CP-PAW Hands-on Course on Density-Functional calculations

Theory of first-principles calculations

Peter E. Blöchl

don't panic!

[1]

[1] To the title page: What is the meaning of ΦSX? Firstly, it sounds like "Physics". Secondly, the symbols stand for the three main pillars of theoretical physics: "X" is the symbol for the coordinate of a particle and represents Classical Physics. "Φ" is the symbol for the wave function and represents Quantum Physics "S" is the symbol for the entropy and represents Statistical Physics.

Contents

Foreword

This booklet is a companion for the "Hands-On Course on first-principles calculations", which we developed to introduce students to first-principles calculations and the use of the CP-PAW code in particular.

The Hands-On course consists of three parts, namely

1. a lecture on the theory behind first principles calculations and on some of the technical details,
2. a lecture on the "Nature of the Chemical Bond", which provides a simple and intuitive understanding of the qualitative features of the electronic structure and finally,
3. the actual hands on course with practical exercises using the CP-PAW code.

This booklet is the reading material for the first part about the theoretical background.

Part I

Physics

Chapter 1

Density-functional theory

1.1 Introduction

On the nanoscale, materials around us have a surprisingly simple structure: The standard model of solid state physics and chemistry only knows of two types of particles, namely the nuclei making up the periodic table and the electrons. Only one kind of interaction between them needs to be considered, namely the electrostatic interaction. Even magnetic forces are important only in rare occasions. All other fundamental particles and interactions are nearly irrelevant for chemistry.

The behavior of the particles can be described by the Schrödinger equation (or better the relativistic Dirac equation), which is easily written down. However, the attempt to solve this equation for any system of interest fails miserably due to what Walter Kohn termed the exponential wall [1].

To obtain an impression of the powers of the exponential wall, imagine the wave function of a N_2 molecule, having two nuclei and fourteen electrons. For N particles, the Schrödinger equation is a partial differential equation in $3N$ dimensions. Let us express the wave function on a grid with about 100 points along each spatial direction and let us consider two spin states for each electron. Such a wave function is represented by $2^{14}100^{3\times16} \approx 10^{100}$ complex numbers. A data server for this amount of data, made of current tera-byte hard disks, would occupy a volume with a diameter of 10^{10} light years!

Treating the nuclei as classical particles turned out to be a good approximation, but the quantum nature of the electrons cannot be ignored. A great simplification is to describe electrons as non-interacting quasi particles. Instead of one wave function in $3N$ dimensions, one only needs to describe N wave functions in three dimensions each, a dramatic simplification from 10^{100} to 10^7 numbers.

While the independent-particle model is very intuitive, and while it forms the basis of most text books on solid-state physics, materials physics, and chemistry, the Coulomb interaction between electrons is clearly not negligible.

Here, density-functional theory [2, 3] comes to our rescue: it provides a rigorous mapping from interacting electrons onto a system of non-interacting electrons. Unfortunately, the exact mapping is utterly complicated, and this is where all the complexity goes. Luckily, there are simple approximations that are both, intuitive and surprisingly accurate. Furthermore, with the help of clever algorithms, density-functional calculations can be performed on current computers for large systems with several hundred atoms in a unit cell or a molecule. The microscopic insight gained from density-functional calculations is a major source of progress in solid state physics, chemistry, material science, and biology.

In the first part of this article, I will try to familiarize the novice reader with the basics of density-functional theory, provide some guidance into common approximations and give an idea of the type of problems that can be studied with density-functional theory.

Beyond this article, I recommend the insightful review and research articles on density-functional theory by Jones and Gunnarsson [4], Baerends [5], von Barth [6], Perdew [7], Yang[8], Truhlar[9] and

their collaborators.

Solving the one-particle Schrödinger equation, which results from density-functional theory, for real materials is a considerable challenge. Several avenues have been developed to their solution. This is the field of electronic structure methods, which will be discussed in the second part of this article. This part is taken from earlier versions by Clemens Först, Johannes Kästner and myself [10, 11].

1.2 Basics of density-functional theory

The dynamics of the electron wave function $|\Psi\rangle$ is governed by the Schrödinger equation $i\hbar\partial_t|\Psi\rangle = \hat{H}|\Psi\rangle$ with the N-particle Hamiltonian $\hat{H}$.

$$\hat{H} = \sum_{j=1}^{N}\left(\frac{-\hbar^2}{2m_e}\vec{\nabla}_j^2 + v_{ext}(\vec{r}_j)\right) + \frac{1}{2}\sum_{i\neq j}^{N}\frac{e^2}{4\pi\epsilon_0|\vec{r}_i - \vec{r}_j|} \; . \tag{1.1}$$

With m_e we denote the electron mass, with ϵ_0 the vacuum permittivity, e is the elementary charge and $\hbar$ is the Planck quantum divided by 2π. The Coulomb potentials of the nuclei have been combined into an external potential $v_{ext}(\vec{r})$.

All N-electron wave functions $\Psi(\vec{x}_1, \ldots, \vec{x}_N)$ obey the Pauli principle, that is they change their sign, when two of its particle coordinates are exchanged.

We use a notation that combines the position vector $\vec{r} \in \mathbb{R}^3$ of an electron with its discrete spin coordinate $\sigma \in \{\uparrow, \downarrow\}$ into a single vector $\vec{x} := (\vec{r}, \sigma)$. Similarly, we use the notation of a four-dimensional integral $\int d^4x := \sum_\sigma \int d^3r$ for the sum over spin indices and the integral over the position. With the generalized symbol $\delta(\vec{x} - \vec{x'}) := \delta_{\sigma,\sigma'}\delta(\vec{r} - \vec{r'})$ we denote the product of Kronecker delta of the spin coordinates and Dirac's delta function for the positions. While, at first sight, it seems awkward to combine continuous and discrete numbers, this notation is less error prone than the notation that treats the spin coordinates as indices, where they can be confused with quantum numbers. During the first reading, the novice can ignore the complexity of the spin coordinates, treating $\vec{x}$ like a coordinate. During careful study, the advanced reader will nevertheless have the complete and concise expressions.

One-particle reduced density matrix and two-particle density

In order to obtain the ground state energy $E = \langle\Psi|\hat{H}|\Psi\rangle$ we need to perform 2^N integrations in $3N$ dimensions each, i.e.

$$E = \int d^4x_1 \cdots \int d^4x_N \; \Psi^*(\vec{x}_1, \ldots, \vec{x}_N)\hat{H}\Psi(\vec{x}_1, \ldots, \vec{x}_N) \; . \tag{1.2}$$

However, only two different types of integrals occur in the expression for the energy, so that most of these integrations can be performed beforehand leading to two quantities of physical significance.

- One of these quantities is the one-particle reduced density matrix $\rho^{(1)}(\vec{x}, \vec{x'})$, which allows one to evaluate all expectation values of one-particle operators such as the kinetic energy and the external potential energy,

$$\rho^{(1)}(\vec{x}, \vec{x'}) := N\int d^4x_2 \ldots \int d^4x_N \; \Psi(\vec{x}, \vec{x}_2, \ldots, \vec{x}_N)\Psi^*(\vec{x'}, \vec{x}_2, \ldots, \vec{x}_N) \; . \tag{1.3}$$

- The other one is the two-particle density $n^{(2)}(\vec{r}, \vec{r'})$, which allows one to determine the interaction between the electrons,

$$n^{(2)}(\vec{r}, \vec{r'}) := N(N-1)\sum_{\sigma,\sigma'}\int d^4x_3 \ldots \int d^4x_N \; |\Psi(\vec{x}, \vec{x'}, \vec{x}_3, \ldots, \vec{x}_N)|^2 \; . \tag{1.4}$$

If it is confusing that there are two different quantities depending on two particle coordinates, note that the one-particle reduced density matrix $\rho^{(1)}$ depends on two $\vec{x}$-arguments of the *same* particle, while the two-particle density $n^{(2)}$ depends on the positions of *two different* particles.

With these quantities the total energy is

$$E = \int d^4x' \int d^4x\, \delta(\vec{x'} - \vec{x}) \left(\frac{-\hbar^2}{2m_e} \vec{\nabla}^2 + v_{ext}(\vec{r}) \right) \rho^{(1)}(\vec{x}, \vec{x'}) + \frac{1}{2} \int d^3r \int d^3r' \, \frac{e^2 n^{(2)}(\vec{r}, \vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} \, , \tag{1.5}$$

where the gradient of the kinetic energy operates on the first argument $\vec{r}$ of the density matrix.

One-particle reduced density matrix and natural orbitals

In order to make oneself familiar with the one-particle reduced density matrix $\rho^{(1)}(\vec{x}, \vec{x'})$, it is convenient to diagonalize it. The eigenstates $\varphi_n(\vec{x})$ are called **natural orbitals** [12] and the eigenvalues $\bar{f}_n$ are their **occupations**. The index n labeling the natural orbitals may stand for a set of quantum numbers.

The density matrix can be written in the form

$$\rho^{(1)}(\vec{x}, \vec{x'}) = \sum_n \bar{f}_n \, \varphi_n(\vec{x}) \varphi_n^*(\vec{x'}) \, . \tag{1.6}$$

The natural orbitals are orthonormal one-particle orbitals, i.e.

$$\int d^4x \, \varphi_m^*(\vec{x}) \varphi_n(\vec{x}) = \delta_{m,n} \, . \tag{1.7}$$

Due to the Pauli principle, occupations are non-negative and never larger than one [13]. The natural orbitals are one-particle wave functions, which already points the way to the world of effectively non-interacting electrons.

The one-particle density matrix provides us with the electron density

$$n^{(1)}(\vec{r}) = \sum_\sigma \rho^{(1)}(\vec{x}, \vec{x}) = \sum_\sigma \sum_n \bar{f}_n \varphi_n^*(\vec{x}) \varphi_n(\vec{x}) \, . \tag{1.8}$$

With the natural orbitals, the total energy Eq. 1.5 obtains the form

$$E = \sum_n \bar{f}_n \int d^4x \, \varphi_n^*(\vec{x}) \frac{-\hbar^2}{2m} \vec{\nabla}^2 \varphi_n(\vec{x}) + \int d^3r \, v_{ext}(\vec{r}) n^{(1)}(\vec{r}) + \frac{1}{2} \int d^3r \int d^3r' \, \frac{e^2 n^{(2)}(\vec{r}, \vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} \, . \tag{1.9}$$

Two-particle density and exchange-correlation hole

The physical meaning of the two-particle density $n^{(2)}(\vec{r}, \vec{r'})$ is the following: For particles that are completely uncorrelated, meaning that they do not even experience the Pauli principle, the two-particle density would be[1] the product of one-particle densities, i.e. $n^{(2)}(\vec{r}, \vec{r'}) = n^{(1)}(\vec{r}) n^{(1)}(\vec{r'})$. If one particle is at position $\vec{r}_0$, the density of the remaining $N-1$ particles is the **conditional density**

$$\frac{n^{(2)}(\vec{r}_0, \vec{r})}{n^{(1)}(\vec{r}_0)} \, . \tag{1.10}$$

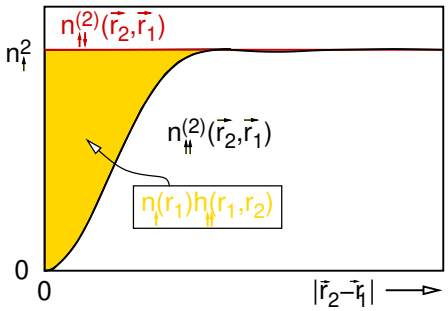


Fig. 1.1: Two-particle density of the non-interacting electron gas for like-spin and opposite spin. Also shown is the relation to the exchange hole $h(\vec{r}_1, \vec{r}_2)$

The conditional density is the electron density as function of $\vec{r}$ seen by one of the electrons at $\vec{r}_0$. This observer electron obviously only sees the remaining $N-1$ electrons.

It is convenient to express the two-particle density by the **hole function** $h(\vec{r}_0, \vec{r})$, i.e.

$$n^{(2)}(\vec{r}_0, \vec{r}) = n^{(1)}(\vec{r}_0)\left[n^{(1)}(\vec{r}) + h(\vec{r}_0, \vec{r})\right] . \tag{1.11}$$

This equation defines the hole function as

$$h(\vec{r}_0, \vec{r}) = \frac{n^{(2)}(\vec{r}_0, \vec{r})}{n^{(1)}(\vec{r}_0)} - n^{(1)}(\vec{r}) . \tag{1.12}$$

One electron at position $\vec{r}_0$ does not "see" the total electron density $n^{(1)}(\vec{r})$ with N electrons, but only the density of the $N-1$ other electrons, because it does not see itself. The hole function $h(\vec{r}_0, \vec{r})$ is simply the difference of the total electron density and the electron density seen by the observer electron at $\vec{r}_0$.

The division of the two-particle density in Eq. 1.11 suggests that we split the electron-electron interaction into the so-called **Hartree energy**

$$E_H \stackrel{\text{def}}{=} \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2 n^{(1)}(\vec{r}) n^{(1)}(\vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} \tag{1.13}$$

and the **potential energy of exchange and correlation**

$$U_{xc} \stackrel{\text{def}}{=} \int d^3r \; n^{(1)}(\vec{r}) \frac{1}{2}\int d^3r' \, \frac{e^2 \; h(\vec{r}, \vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} . \tag{1.14}$$

Keep in mind that U_{xc} is *not* the exchange-correlation energy. The difference between U_{xc} and the exchange-correlation energy E_{xc} is a kinetic energy correction that will be discussed later in Eq. 1.21.

The hole function has a physical meaning: An electron sees the total density minus the electrons accounted for by the hole. Thus, each electron not only experiences the repulsive electrostatic potential of the total electron density $n^{(1)}(\vec{r})$, but also the attractive potential of its own exchange-correlation hole $h(\vec{r}_0, \vec{r})$.

A few facts for this hole density are apparent:

1. Because each electron of a N-electron system sees $N-1$ other electrons, the hole function integrates to exactly minus one electron

$$\int d^3r \; h(\vec{r}_0, \vec{r}) = -1 \tag{1.15}$$

[1] This is correct only up to a term that vanishes in the limit of infinite particle number.

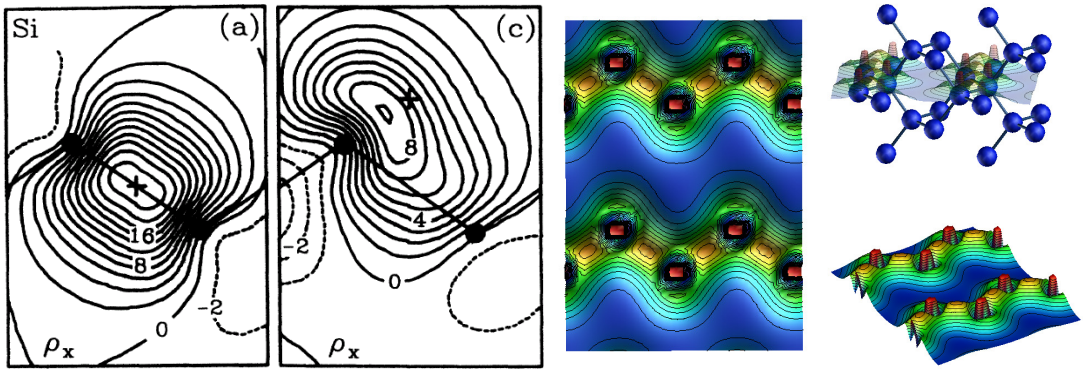


Fig. 1.2: Exchange hole in silicon from [14]. The cross indicates the position of the observer electron. The black spheres and the lines indicate the atomic positions and bonds in the (110) plane.

irrespective of the position $\vec{r}_0$ of the observing electron.

2. The density of the remaining $N-1$ electrons can not be larger than the total electron density. This implies

$$h(\vec{r}_0, \vec{r}) \geq -n^{(1)}(\vec{r}) \; . \tag{1.16}$$

3. Due to the Pauli principle, no other electron with the same spin as the observer electron can be at the position $\vec{r}_0$. Thus the on-top hole $h(\vec{r}_0, \vec{r}_0)$ obeys the limits [15]

$$-\frac{1}{2} n^{(1)}(\vec{r}_0) \geq h(\vec{r}_0, \vec{r}_0) \geq -n^{(1)}(\vec{r}_0) \; . \tag{1.17}$$

4. Assuming locality, the hole function vanishes at large distances from the observer electron at $\vec{r}_0$, i.e.

$$h(\vec{r}_0, \vec{r}) \to 0 \qquad \text{for} \qquad |\vec{r} - \vec{r}_0| \to \infty \; . \tag{1.18}$$

With locality I mean that the density does not depend on the position or the presence of an observer electron, if the latter is very far away.

A selfmade functional

It is fairly simple to make our own density functional[2]: For a given density, we choose a simple shape for the hole function, such as a spherical box. Then we scale the value and the radius such that the hole function integrates to -1, and that its value is opposite equal to the spin density at its center. The electrostatic potential of this hole density at its center is the exchange-correlation energy for the observer electron. Our model has an exchange-correlation energy[3] of

$$U_{xc}[n^{(1)}] \approx -\frac{1}{2} \int d^3r \; n^{(1)}(\vec{r}) \left(\frac{3}{4} \frac{e^2}{4\pi\epsilon_0} \sqrt[3]{\frac{2\pi}{3}} \left(n^{(1)}(\vec{r}) \right)^{\frac{1}{3}} \right) \sim \int d^3r \left(n^{(1)}(\vec{r}) \right)^{\frac{4}{3}} . \tag{1.19}$$

[2] A functional $F[y]$ maps a function $y(x)$ to a number F. It is a generalization of the function $F(\vec{y})$ of a vector $\vec{y}$, where the vector index of $\vec{y}$ is turned into a continuous argument x.

[3] For this model we do not distinguish between the energy of exchange and correlation and its potential energy contribution

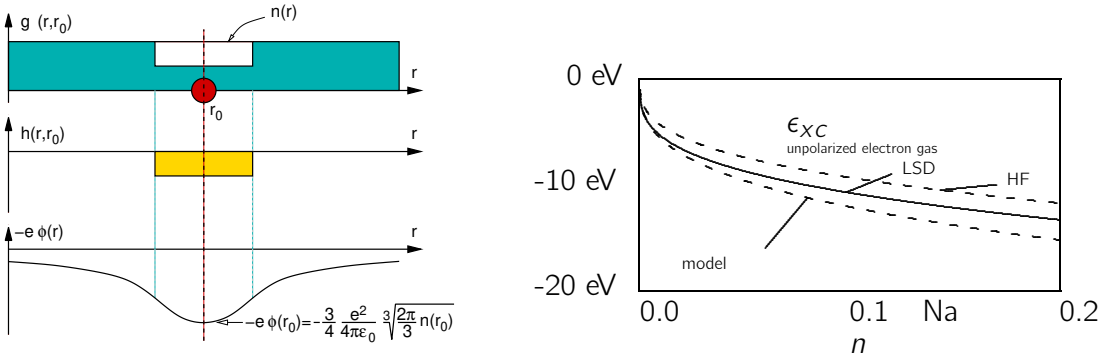


Fig. 1.3: Left: Scheme to demonstrate the construction of the exchange-correlation energy from a simple model. Right: exchange correlation energy per electron ϵ_{xc} as function of electron density from our model, Hartree-Fock approximation and the exact result. The symbol "Na" indicates the density of Sodium.

The derivation is an elementary exercise and is given in the appendix. The resulting energy per electron ϵ_{xc} is given on the right-hand side of Fig. 1.3 indicated as "model" and compared with the exact result indicated as "LSD" [4] and the Hartree-Fock result indicated as "HF" for a homogeneous electron gas.

The agreement with the correct result, which is surprisingly good for such a crude model, provides an idea of how robust the density-functional theory is with respect to approximations. While this model has been stripped to the bones, it demonstrates the way physical insight enters into the construction of density functionals. Modern density functionals are far more sophisticated and exploit much more information [16], but the basic method of construction is similar.

Kinetic energy

While the expression for the kinetic energy in Eq. 1.9 seems familiar, there is a catch to it. In order to know the natural orbitals and the occupations we need access to the many-particle wave function or at least to its reduced density matrix.

A good approximation for the kinetic energy of the interacting electrons is the kinetic energy functional $T_s[n^{(1)}]$ of the ground state of non-interacting electrons with the same density as the true system. It is defined by

$$T_s[n^{(1)}] = \min_{\{f_n\in[0,1],|\psi_n\rangle\}} \operatorname*{stat}_{v_{eff},\mathbf{\Lambda}} \Bigg\{ \sum_n f_n \int d^4x\, \psi_n^*(\vec{x}) \frac{-\hbar^2\vec{\nabla}^2}{2m_e} \psi_n(\vec{x}) + \int d^3r\, v_{eff}(\vec{r}) \left(\left[\sum_n f_n \sum_\sigma \psi_n^*(\vec{x})\psi_n(\vec{x}) \right] - n^{(1)}(\vec{r}) \right) - \sum_{n,m} \Lambda_{m,n} \left(\langle\psi_n|\psi_m\rangle - \delta_{n,m} \right) \Bigg\} . \tag{1.20}$$

Note that $f_n \neq \bar{f}_n$ and that the so-called Kohn-Sham orbitals $\psi_n(\vec{x})$ differ[5] from the natural orbitals $\varphi_n(\vec{x})$. Natural orbitals and Kohn-Sham wave functions are fairly similar, while the occupations f_n of Kohn-Sham orbitals differ considerably from those $\bar{f}_m$ of the natural orbitals. The effective potential $v_{eff}(\vec{r})$ is the Lagrange multiplier for the density constraint. **The effective potential can been seen as that potential needed to "press" the electron density into the shape given by $n^{(1)}(\vec{r})$.** $\Lambda_{m,n}$ are

[4] LSD stands for "local spin-density" and denotes the local spin-density approximation (LSDA). Modern implementations do not only use the total density as fundamental variable, but also the spin density.

[5] To be precise, Kohn-Sham orbitals are the natural orbitals for non-interacting electrons of a given density. They are however different from the natural orbitals of interacting electrons at the same density.

the Lagrange multipliers for the orthonormality constraint of the Kohn-Sham orbitals. Diagonalization of $\mathbf{\Lambda}$ yields a diagonal matrix with the one-particle energies on the diagonal.

This kinetic energy $T_s[n^{(1)}]$ is a unique functional of the density, which is the first sign that we are approaching a density-functional theory. Furthermore, while defining the kinetic-energy functional $T_s[n^{(1)}]$, we referred for the first time to a principle that is limited to the ground state. Density-functional theory as described here is inherently a ground-state theory.

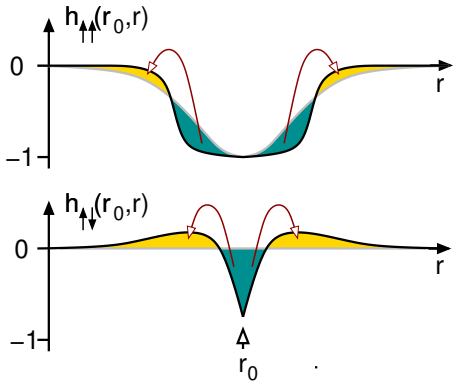


Fig. 1.4: Illustration of the correlation hole. The exchange hole, present without interaction, is deformed by the Coulomb reaction. Because this deformation must respect the charge sum rule, this deformation makes the hole more compact. The dominant effect is on the opposite-spin direction, because the like-spin electrons are already at a distance by the exchange hole.

Why does the true kinetic energy of interacting electrons differ from that of non-interacting electrons?

Consider the hole function of a non-interacting electron gas. When the hole function of non-interacting electrons is inserted into Eq. 1.14 for U_{xc}, the potential energy of exchange and correlation, we obtain a contribution to the total energy that is called exchange energy.

The interaction changes the many-particle wave function and this leads to a second energy contribution, which is called correlation energy: Namely, when the interaction is switched on, the wave function is deformed in such a way that the Coulomb repulsion between the electrons is reduced. This makes the hole function more compact. However, there is a price to pay when the wave functions adjust to reduce the Coulomb repulsion between the electrons, namely an increase of the kinetic energy: **Pushing electrons away from the neighborhood of the reference electrons requires work to be performed against the kinetic pressure of the electron gas, which raises the kinetic energy.** Thus, the system has to find a compromise between minimizing the electrostatic repulsion of the electrons and increasing its kinetic energy. As a result, the correlation energy has a potential-energy contribution and a kinetic-energy contribution.

This tradeoff between kinetic and interaction energy can be observed in Fig. 1.3. The correct exchange-correlation energy is close to our model at low densities, while it becomes closer to the Hartree-Fock result at high densities. This is consistent with the fact that the electron gas can easily be deformed at low densities, while the deformation becomes increasingly costly at high densities due to the larger pressure of the electron gas.

The difference between T_s and the true kinetic energy is combined with the potential energy of exchange and correlation U_{xc} from Eq. 1.14 into the exchange-correlation energy E_{xc}, i.e.

$$E_{xc} \stackrel{\text{def}}{=} U_{xc} + \sum_n \bar{f}_n \int d^4x\, \varphi_n^*(\vec{x}) \frac{-\hbar^2}{2m} \vec{\nabla}^2 \varphi_n(\vec{x}) - T_s[n^{(1)}] \,. \tag{1.21}$$

Note, that the $\varphi_n(\vec{x})$ and the $\bar{f}_n$ are natural orbitals and occupations of the interacting electron gas, and that they differ from the Kohn-Sham orbitals $\psi_n(\vec{x})$ and occupations f_n. We will later see

that the exchange-correlation energy can be expressed as a functional of the density alone. At this point, however, Eq. 1.21 requires the knowledge of the many-particle wave function[6] to evaluate the exchange-correlation energy.

Total energy

The total energy obtains the form

$$E = \min_{|\Phi\rangle,\{|\psi_n\rangle,f_n\in[0,1]\}} \operatorname*{stat}_{v_{eff},\Lambda}\Bigg\{\sum_n f_n \int d^4x\ \psi_n^*(\vec{x})\frac{-\hbar^2}{2m}\vec{\nabla}^2\psi_n(\vec{x}) + \int d^3r\ v_{eff}(\vec{r})\Bigg(\Bigg[\sum_n f_n \sum_\sigma \psi_n^*(\vec{x})\psi_n(\vec{x})\Bigg] - n^{(1)}_{|\Phi\rangle}(\vec{r})\Bigg) + \int d^3r\ v_{ext}(\vec{r})n^{(1)}_{|\Phi\rangle}(\vec{r}) + \frac{1}{2}\int d^3r \int d^3r'\ \frac{e^2 n^{(1)}_{|\Phi\rangle}(\vec{r})n^{(1)}_{|\Phi\rangle}(\vec{r'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + E_{xc}[|\Phi\rangle] - \sum_{n,m}\Lambda_{m,n}\Big(\langle\psi_n|\psi_m\rangle - \delta_{n,m}\Big)\Bigg\}. \qquad (1.22)$$

This equation shall be read as follows: For a given many-particle wave function $|\Phi\rangle$ we extract the density $n^{(1)}_{|\Phi\rangle}(\vec{r})$. Kohn-Sham orbitals $|\psi_n\rangle$ and occupations f_n are obtained by an independent minimization of the kinetic energy for the density $n^{(1)}_{|\Phi\rangle}(\vec{r})$. Only the exchange-correlation energy $E_{xc}[|\Phi\rangle]$ needs yet to be calculated directly from the many-particle wave function. This is done for all many-particle wave functions until the minimum of the total energy has been found.

Because we still need the full many particle wave function to determine the expression above, Eq. 1.22 is not yet a functional of the density.

If, however, we were able to express the exchange-correlation energy E_{xc} as a functional of the density alone, there would be no need for the many-particle wave function at all and the terrors of the exponential wall would be banned. We could minimize Eq. 1.22 with respect to the density, Kohn-Sham orbitals and their occupations.

Self-consistent equations

Let us, for the time being, simply assume that the exchange-correlation energy E_{xc} is a functional of the electron density and explore the consequences of this assumption. Later, I will show that this assumption is actually valid. With this assumption, the ground-state total energy Eq. 1.22 has the form

GROUND-STATE ENERGY IN DENSITY FUNCTIONAL THEORY

$$E = \min_{n^{(1)}} \min_{\{|\psi_n\rangle,f_n\in[0,1]\}} \operatorname*{stat}_{v_{eff},\Lambda}\Bigg\{\sum_n f_n \int d^4x\ \psi_n^*(\vec{x})\frac{-\hbar^2}{2m}\vec{\nabla}^2\psi_n(\vec{x}) + \int d^3r\ v_{eff}(\vec{r})\Bigg(\Bigg[\sum_n f_n \sum_\sigma \psi_n^*(\vec{x})\psi_n(\vec{x})\Bigg] - n^{(1)}(\vec{r})\Bigg) + \int d^3r\ v_{ext}(\vec{r})n^{(1)}(\vec{r}) + \frac{1}{2}\int d^3r \int d^3r'\ \frac{e^2 n^{(1)}(\vec{r})n^{(1)}(\vec{r'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + E_{xc}[n^{(1)}] - \sum_{n,m}\Lambda_{m,n}\Big(\langle\psi_n|\psi_m\rangle - \delta_{n,m}\Big)\Bigg\}. \qquad (1.23)$$

The minimization in Eq. 1.23 with respect to the one-particle wave functions yields the Kohn-Sham

[6] The potential energy of exchange and correlation U_{xc}, the density $n^{(1)}$, natural orbitals φ_n and occupations $\bar{f}_n$ are obtained from the many-particle wave function.

equations

$$\left[\frac{-\hbar^2}{2m_e}\vec{\nabla}^2 + v_{eff}(\vec{r}) - \epsilon_n\right]\psi_n(\vec{x}) = 0 \qquad \text{with} \qquad \int d^4x\, \psi_m(\vec{x})\psi_n(\vec{x}) = \delta_{m,n}\,. \tag{1.24}$$

The Kohn-Sham energies ϵ_n are the eigenvalues of the Lagrange multiplier $\mathbf{\Lambda}$.

The requirement that the derivative of the total energy Eq. 1.22 with respect to the density vanishes, yields an expression for the effective potential

$$v_{eff}(\vec{r}) = v_{ext}(\vec{r}) + \int d^3r'\, \frac{e^2 n^{(1)}(\vec{r'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + \frac{\delta E_{xc}[n^{(1)}]}{\delta n^{(1)}(\vec{r})}\,. \tag{1.25}$$

Both equations, together with the density constraint

$$n^{(1)}(\vec{r}) = \sum_n f_n \sum_\sigma \psi_n^*(\vec{x})\psi_n(\vec{x})\,, \tag{1.26}$$

form a set of coupled equations, that determine the electron density and the total energy. This set of coupled equations, Eqs. 1.24, 1.25, and 1.26, is what is solved in the so-called self-consistency loop. Once the set of self-consistent equations has been solved, we obtain the electron density and we can evaluate the total energy.

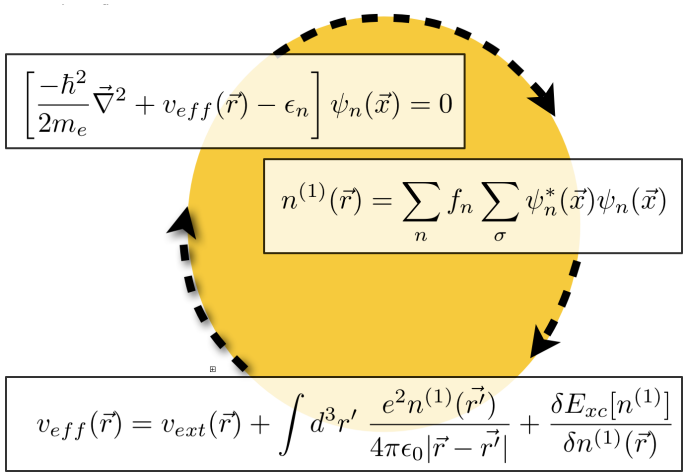


Fig. 1.5: Self-consistency cycle.

In practice, one often makes the assumption that the non-interacting electrons in the effective potential closely resemble the true interacting electrons, and extracts a wealth of other physical properties from the Kohn-Sham wave functions $|\psi_n\rangle$ and the Kohn-Sham energies ϵ_n. However, there is little theoretical backing for this approach and, if it fails, one should not blame density-functional theory!

Is there a density functional?

The argument leading to the total-energy expression Eq. 1.23, from which we obtained the self-consistent equations, Eqs. 1.24, 1.25, and 1.26, relied entirely on the hope that exchange-correlation functional can be expressed as a functional of the electron density. In fact, this can easily be shown, if we restrict us to ground state densities. The proof goes back to the seminal paper by Levy [17, 18]. It is pictorially represented in Fig. 1.6.

Imagine that one could construct all fermionic many-particle wave functions. For each of these wave functions, we can determine in a unique way the electron density

$$n^{(1)}(\vec{r}) = N \sum_{\sigma} \int d^3x_2 \ldots \int d^3x_N \, |\Psi(\vec{x}, \vec{x}_2, \ldots \vec{x}_N)|^2 \, . \tag{1.27}$$

Having the electron densities, we sort the wave functions according to their density. For each density, I get a mug $M[n^{(1)}]$ that holds all wave functions with that density, that is written on the label of the mug.

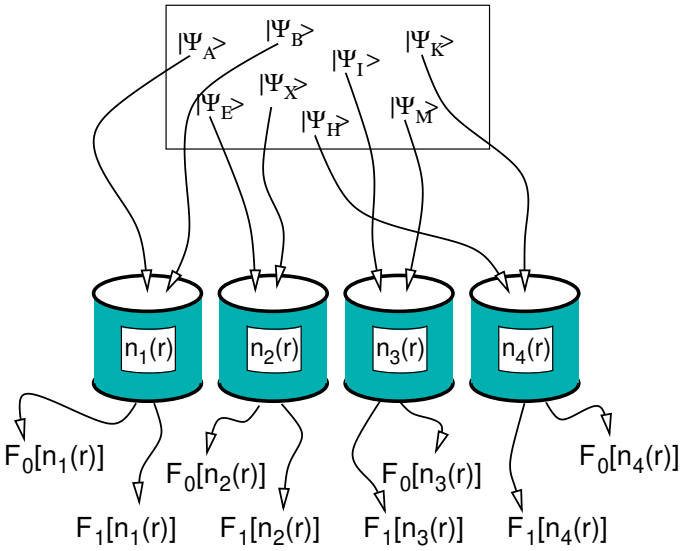


Fig. 1.6: Illustration for Levy's proof that there exists a density functional.

Now we turn to each mug $M[n^{(1)}]$ in sequence and determine for each the wave function with the lowest energy. Because the external potential energy is the same for all wave functions with the same density, we need to consider only the kinetic energy operator $\hat{T}$ and the operator $\hat{W}$ of the electron-electron interaction[7] , and we do not need to consider the external potential.

$$F^{\hat{W}}[n^{(1)}] = \min_{|\Psi\rangle \in M[n^{(1)}]} \langle \Psi | \hat{T} + \hat{W} | \Psi \rangle \tag{1.29}$$

$F^{\hat{W}}[n^{(1)}]$ is the universal density functional. It is universal in the sense that it is an intrinsic property of the electron gas and absolutely independent of the external potential.

Next, we repeat the same construction as that for a universal density functional, but now we leave out the interaction $\hat{W}$ and consider only the kinetic energy $\hat{T}$.

$$F^{0}[n^{(1)}] = \min_{|\Psi\rangle \in M[n^{(1)}]} \langle \Psi | \hat{T} | \Psi \rangle \tag{1.30}$$

The resulting functional $F^0[n^{(1)}]$ is nothing but the kinetic energy of non-interacting electrons $T_s[n^{(1)}]$.

Now we can write down the total energy as functional of the density

$$E[n^{(1)}] = F^{\hat{W}}[n^{(1)}] + \int d^3r \; v_{ext}(\vec{r}) n^{(1)}(\vec{r}) \tag{1.31}$$

[7] The electron-electron interaction is the second term of Eq. 1.1

$$\hat{W} \stackrel{\text{def}}{=} \frac{1}{2} \sum_{i \neq j}^{N} \frac{e^2}{4\pi\epsilon_0 |\vec{r}_i - \vec{r}_j|} \tag{1.28}$$

When we compare Eq. 1.31 with Eq. 1.22, and use 1.20, and 1.30, we obtain an expression for the exchange-correlation energy.

$$\begin{aligned} \underbrace{E}_{F^W+E_{ext}} &\overset{Eqs.\ 1.22,1.20}{=} \underbrace{T_s}_{F^0} + E_{ext} + E_H + E_{xc} \\ E_{xc}[n^{(1)}] &= F^{\hat{W}}[n^{(1)}(\vec{r})] - F^0[n^{(1)}(\vec{r})] - \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2 n^{(1)}(\vec{r}) n^{(1)}(\vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} \end{aligned} \tag{1.32}$$

This completes the proof that the exchange-correlation energy is a functional of the electron density. The latter was the assumption for the total energy expression Eq. 1.23 from which we derived the set of self-consistent equations, Eqs. 1.24, 1.25, and 1.26 for the Kohn-Sham wave functions $\psi_n(\vec{x})$.

With this, I finish the description of the theoretical basis of density-functional theory. We have seen that the total energy can rigorously be expressed as a functional of the density or, in practice, as a functional of a set of one-particle wave functions, the Kohn-Sham wave functions and their occupations. Density-functional theory per se is not an approximation and, in contrast to common belief, it is not a mean-field approximation. Nevertheless, we need to introduce approximations to make density-functional theory work. This is because the exchange-correlation energy $E_{xc}[n^{(1)}]$ is not completely known. These approximations will be discussed in the next section.

1.3 Adiabatic connection

At this point I want to demonstrate a theorem that is central to the thinking within density-functional theory, the **adiabatic connection formula** [19, 20] [8] It allows one to express the difference between the kinetic energy of the interacting and the non-interacting wave function by the exchange-correlation hole for various strengths of the electron-electron interaction.

Let us proceed as in Levy's proof, but now we use a Hamiltonian with a scaled interaction

$$\hat{H}(\lambda) = \hat{T} + \lambda \hat{W} \tag{1.33}$$

The extreme values, $\lambda = 0$ and $\lambda = 1$ describe the non-interacting and the interacting electron gas, respectively.

For each value of λ we determine

$$F^{\lambda\hat{W}} = \min_{|\Psi\rangle \in M[n^{(1)}]} \langle \Psi | \hat{T} + \lambda \hat{W} | \Psi \rangle = \langle \bar{\Psi}_\lambda[n^{(1)}] | \hat{T} + \lambda \hat{W} | \bar{\Psi}_\lambda[n^{(1)}] \rangle \tag{1.34}$$

Note, that the wave function $|\bar{\Psi}_\lambda[n^{(1)}]\rangle$ of the minimum depends on λ.

The fully interacting functional can be obtained as an integral over the derivative

$$F^{\hat{W}}[n^{(1)}] = F^0[n^{(1)}] + \int_0^1 d\lambda \, \frac{\partial F^{\lambda\hat{W}}[n^{(1)}]}{\partial \lambda} \tag{1.35}$$

Using the minimum condition the derivative of the functional with respect to λ can be expressed entirely by the interaction energy.

In order to make the minimum condition explicit, we include the density and the normalization constraint in the constrained search. Thus the functional is written as

$$\begin{aligned} F^{\lambda\hat{W}}[n^{(1)}] = \min_{|\Psi\rangle} \Bigg[& \langle \Psi | \hat{T} + \lambda \hat{W} | \Psi \rangle \\ & + \int d^3r \; v_{eff,\lambda}(\vec{r}) \Big(\langle \Psi | \hat{n}(\vec{r}) | \Psi \rangle - n^{(1)}(\vec{r}) \Big) - E \Big(\langle \Psi | \Psi \rangle - 1 \Big) \Bigg] \end{aligned} \tag{1.36}$$

[8] For historical notes on the adiabatic connection theorem see Jones[21] and Musher[22].

The minimum condition is

$$\left[\hat{T} + \lambda\hat{W} + \int d^3r\ v_{eff,\lambda}(\vec{r})\hat{n}(\vec{r}) - E_\lambda\right]|\bar{\Psi}_\lambda\rangle = 0 \tag{1.37}$$

Now we can perform the derivative of

$$\begin{aligned} F^{\lambda\hat{W}}[n^{(1)}] &= \langle\bar{\Psi}_\lambda|\hat{T} + \lambda\hat{W}|\bar{\Psi}_\lambda\rangle \\ &\quad + \int d^3r\ v_{eff,\lambda}(\vec{r})\Big(\langle\bar{\Psi}_\lambda|\hat{n}(\vec{r})|\bar{\Psi}_\lambda\rangle - n^{(1)}(\vec{r})\Big) - E_\lambda\Big(\langle\bar{\Psi}_\lambda|\bar{\Psi}_\lambda\rangle - 1\Big) \end{aligned} \tag{1.38}$$

with respect to λ. The λ-dependent quantities are the wave functions, and the lagrange multipliers $v_{eff,\lambda}(\vec{r})$ and E_λ. We obtain

$$\begin{aligned} \frac{d}{d\lambda}F^{\lambda\hat{W}}[n^{(1)}] &= \langle\frac{d\bar{\Psi}_\lambda}{d\lambda}|\underbrace{\left(\hat{T} + \lambda\hat{W} + \int d^3r\ v_{eff,\lambda}(\vec{r})\hat{n}(\vec{r}) - E_\lambda\right)}_{=0}|\bar{\Psi}_\lambda\rangle \\ &= \underbrace{\langle\bar{\Psi}_\lambda|\hat{T} + \lambda\hat{W} + \int d^3r\ v_{eff,\lambda}(\vec{r})\hat{n}(\vec{r}) - E_\lambda|}_{=0}\frac{d\bar{\Psi}_\lambda}{d\lambda}\rangle \\ &= \langle\bar{\Psi}_\lambda|\hat{W}|\bar{\Psi}_\lambda\rangle + \int d^3r\ \frac{dv_{eff,\lambda}}{d\lambda}(\vec{r})\underbrace{\Big(\langle\bar{\Psi}_\lambda|\hat{n}(\vec{r})|\bar{\Psi}_\lambda\rangle - n^{(1)}(\vec{r})\Big)}_{=0} - \frac{dE_\lambda}{d\lambda}\underbrace{\Big(\langle\bar{\Psi}_\lambda|\bar{\Psi}_\lambda\rangle - 1\Big)}_{=0} \\ &= \langle\bar{\Psi}_\lambda|\hat{W}|\bar{\Psi}_\lambda\rangle \end{aligned} \tag{1.39}$$

Thus we can express the functional as

$$\begin{aligned} F^{\hat{W}}[n^{(1)}] &= T_s[n^{(1)}] + \int_0^1 d\lambda\ \langle\bar{\Psi}_\lambda|\hat{W}|\bar{\Psi}_\lambda\rangle \\ &= T_s[n^{(1)}] + \frac{1}{2}\int d^3r\int d^3r'\ \frac{e^2n^{(1)}(\vec{r})n^{(1)}(\vec{r'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + \int d^3r n^{(1)}(\vec{r})\int d^3r'\frac{e^2h_{av}(\vec{r},\vec{r'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} \end{aligned} \tag{1.40}$$

where

$$h_{av}(\vec{r},\vec{r'}) = \int_0^1 d\lambda\ h_\lambda(\vec{r},\vec{r'}) \tag{1.41}$$

1.4 Jacob's ladder of density functionals

The development of density functionals is driven by mathematical analysis of the exact exchange-correlation hole [16, 7], physical insight and numerical benchmark calculations on real systems. The functionals evolved in steps from one functional form to another, with several parameterizations at each level. Perdew pictured this development by Jacob's ladder leading up to heaven [23, 7]. In his analogy the different rungs of the ladder represent the different levels of density functionals leading to the unreachable, ultimately correct functional.

JACOB'S LADDER OF DENSITY FUNCTIONALS

The different stairs of Jacob's Ladder of density functionals are

1. LDA (local-density approximation): the exchange-correlation hole is taken from the free electron gas. These functionals exhibit strong overbinding. While the van-der Waals bond is not considered in these functionals, the overbinding mimicked van-der-Waals bonding, albeit for the wrong reason.
2. GGA (generalized gradient approximation): These functionals not only use the electron density but also its gradient to estimate the asymmetry of the exchange-correlation hole with respect to the reference electrons. As a result surfaces are energetically favored compared to LDA and the overbinding is strongly reduced.
3. meta-GGA: in addition to the gradient also the kinetic energy density is used as a parameter. The kinetic energy is a measure for the flexibility of the electron gas.
4. Hybrid functionals: Hybrid functional include a fraction of exact exchange. This functionals improve the description of left-right correlations.
5. Exact: An exact density functional can be obtained using the constrained search formalism using many-particle techniques.

1.5 X_α method

Even before the invention of density-functional theory per se, the so-called X_α method has been introduced. Today, the X_α method has mostly historical value. The X_α method uses the expression for the exchange of a homogeneous electron gas[24] instead of the exchange-correlation energy. However, the exchange energy has been scaled with a parameter, namely X_α, that has been adjusted to Hartree Fock calculations. The results are shown in Fig. 1.7.

The rationale behind the X_α-method is a dimensional argument. Choose a given shape for the exchange-correlation hole, but scale it according to the density and the electron sum rule. Then the exchange-correlation energy per electron always scales like $n^{\frac{1}{3}}$. Each shape corresponds to a specific pre-factor.

Let us proceed through this argument: Consider a given shape described by a function $f(\vec{r})$ with

$$f(\vec{0}) = 1$$
$$\int d^3r\ f(\vec{r}) = 1$$

Now, we express the hole function by the function f by scaling its magnitude at the origin such that the amplitude of the hole cancels the electron density. Secondly, we stretch the function in space so that the sum rule, which says that the hole must integrate to -1, is fulfilled. These conditions yield the model for the exchange-correlation hole.

$$h(\vec{r}_0, \vec{r}) = -n(\vec{r}_0)\, f\left(\frac{\vec{r} - \vec{r}_0}{n(\vec{r}_0)^{\frac{1}{3}}}\right) \tag{1.42}$$

The corresponding exchange-correlation energy per electron is

$$\epsilon_{xc}(\vec{r}) = -\frac{1}{2}\int d^3r'\ \frac{e^2}{4\pi\epsilon_0|\vec{r} - \vec{r'}|} f\left(\frac{\vec{r} - \vec{r'}}{n(\vec{r})^{\frac{1}{3}}}\right) \tag{1.43}$$

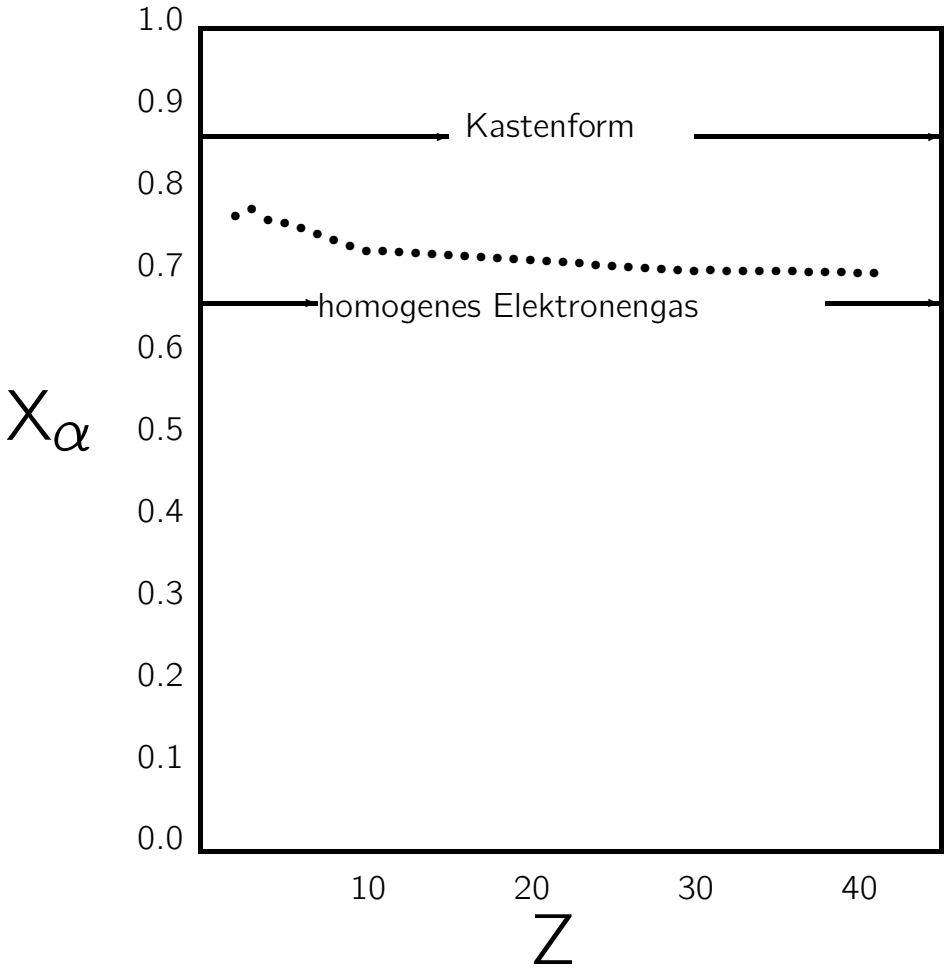


Fig. 1.7: X_α value obtained by comparison of the atomic energy with exact Hartree-Fock calculations as function of atomic number of the atom.[25, 26]

If we introduce a variable transform

$$\vec{y} = \frac{\vec{r} - \vec{r'}}{n(\vec{r})^{\frac{1}{3}}} \tag{1.44}$$

we obtain

$$\epsilon_{xc}(\vec{r}) = -n(\vec{r})^{\frac{1}{3}} \underbrace{\frac{1}{2} \int d^3y' \, \frac{e^2}{4\pi\epsilon_0 |\vec{y}|} f(\vec{y})}_{=:C} = -C n^{\frac{1}{3}} \tag{1.45}$$

where C is a constant that is entirely defined by the shape function $f(\vec{r})$.

In general, the X_α method yields larger band gaps than density-functional theory. The latter severely underestimates band gaps. This is in accord with the tendency of Hartree Fock to overestimate band gaps. In contrast to Hartree Fock, however, the X_α method is superior for the description of metals because it does not lead to a vanishing density of states at the Fermi level, a well-known problem of the Hartree-Fock approximation.

LDA, the big surprise

The first density functionals used in practice were based on the local-density approximation (LDA). The hole function for an electron at position $\vec{r}$ has been approximated by the one of a homogeneous electron gas with the same density as $n^{(1)}(\vec{r})$. The exchange-correlation energy for the homogeneous electron gas has been obtained by quantum Monte Carlo calculations [27] and analytic calculations [28]. The local-density approximation has been generalized early to local spin-density approximation (LSD) [29].

Truly surprising was how well the theory worked for real systems. Atomic distances could be determined within a few percent of the bond length and energy differences in solids were surprisingly good.

This was unexpected, because the density in real materials is far from homogeneous. Gunnarsson and Lundquist [20] explained this finding with sum rules that are obeyed by the local-density approximation: Firstly, the exchange-correlation energy depends only on the spherical average of the exchange-correlation hole. Of the radial hole density only the first moment contributes, while the second moment is fixed by the sum-rule that the electron density of the hole integrates to -1. Thus we can use

$$\int d^3r \, \frac{e^2 h(\vec{r}_0, \vec{r})}{4\pi\epsilon_0 |\vec{r} - \vec{r}_0|} = -\frac{e^2}{4\pi\epsilon_0} \frac{\int_0^\infty dr \; r \left\langle h(\vec{r}_0, \vec{r}') \right\rangle_{|\vec{r}' - \vec{r}_0| = r}}{\int_0^\infty dr \; r^2 \left\langle h(\vec{r}_0, \vec{r}') \right\rangle_{|\vec{r}' - \vec{r}_0| = r}} \tag{1.46}$$

where the angular brackets imply the angular average of $\vec{r}' - \vec{r}_0$. This dependence on the hole density is rather insensitive to small changes of the hole density. Even for an atom, the *spherically averaged* exchange hole closely resembles that of the homogeneous electron gas [4], as demonstrated in Fig. 1.8.

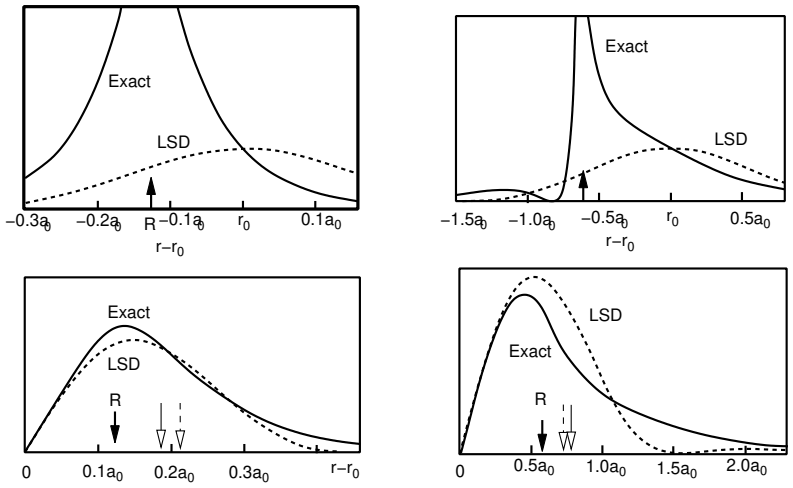


Fig. 1.8: Exchange hole of the nitrogen atom for an electron close to the nucleus (left) and further from the nucleus (right). As seen in the top figures, the true hole is centered at the nucleus, and the exchange hole of the free electron gas is centered on the electron. Despite the differences the integrands for the exchange energy, i.e. the spherical average multiplied by $\frac{1}{2} \cdot 4\pi r^2 \cdot \frac{e^2}{4\pi\epsilon_0 r}$, shown at the bottom, closely resembles each other. Redrawn from [4].

The main deficiency of the LDA was the strong overbinding with bond energies in error by about one electron volt. On the one hand, this rendered LDA useless for most applications in chemistry. On the other hand, the problem was hardly visible in solid state physics where bonds are rarely broken, but rearranged so that errors cancelled.

GGA, entering chemistry

Being concerned about the large density variations in real materials, one tried to include the first terms of a Taylor expansion in the density gradients. These attempts failed miserably. The culprit has been a violation of the basic sum rules as pointed out by Perdew [30]. The cure was a cutoff for the gradient contributions at high gradients, which lead to the class of generalized gradient approximations (GGA) [31].

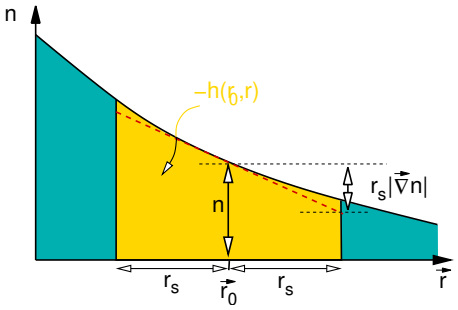


Fig. 1.9: Definition of the reduced gradient. The density gradient is made dimension-less by multiplication with the Seitz radius r_s representing the average size of the exchange-correlation hole and by division by the density. The Seitz radius is defined a by $\frac{4\pi}{3} r_s^2 n = 1$, where n is the electron density. If the reduced gradient exceeds one the center of the exchange-correlation hole moves away from the electron to avoid negative two-particle densities.

Becke [32] provides an intuitive description for the workings of GGA's, which I will sketch here in a simplified manner: Becke uses an ansatz $E_{xc} = \int d^3r\, A(n(\vec{r}))F(x(\vec{r}))$ for the exchange-correlation energy where $n(\vec{r})$ is the local density and $x = |\vec{\nabla} n|/n^{\frac{4}{3}}$ is a dimensionless reduced gradient. Do not confuse this symbol with the combined position-and-spin coordinate $\vec{x}$. The function A is simply the LDA expression and $F(x)$ is the so-called enhancement factor. The large-gradient limit of $F(x)$ is obtained from a simple physical argument:

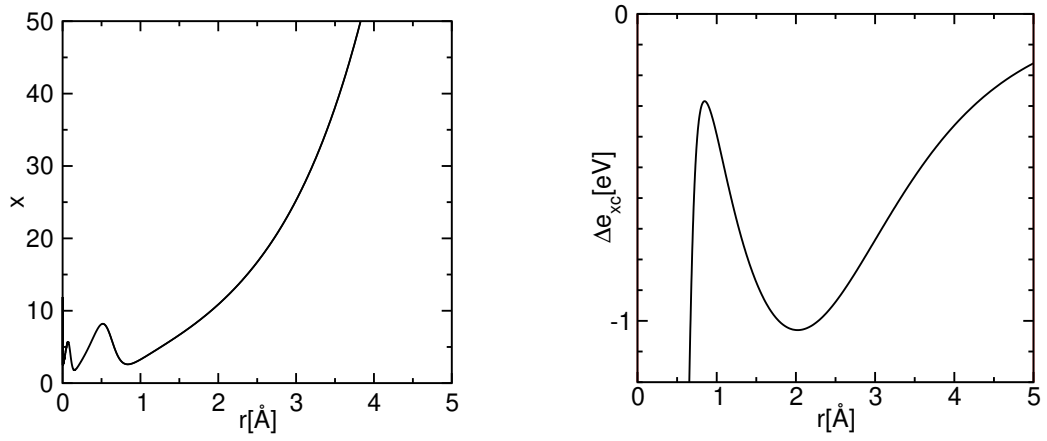


Fig. 1.10: Left figure: reduced density gradient $x = |\vec{\nabla} n|/n^{\frac{4}{3}}$ of a silicon atom as function of distance from the nucleus demonstrating that the largest reduced gradients occur in the exponential tails. Right figure: additional contribution from the gradient correction (PBE versus PW91 LDA) of the exchange-correlation energy per electron. The figure demonstrates that the gradient correction stabilizes the tails of the wave function. The covalent radius of silicon is at 1.11 Å.

Somewhat surprisingly, the reduced gradient is largest not near the nucleus but in the exponentially

decaying charge-density tails as shown in Fig. 1.10. For an electron that is far from an atom, the hole is on the atom, because a hole can only be dug where electrons are. Thus the Coulomb interaction energy of the electron with its hole is $-\frac{e^2}{4\pi\epsilon_0 r}$, where r is the distance of the reference electron from the atom. As shown in appendix 1.7.2, the enhancement factor can now be obtained by enforcing this behavior for exponentially decaying densities.

As a result, the exchange and correlation energy per electron in the tail region of the electron density falls of with the inverse distance in GGA, while it has a much faster, exponential decay in the LDA. Thus, the tail region is stabilized by GGA. This contribution acts like a negative "surface energy".

When a bond between two atoms is broken, the surface is increased. In GGA this bond-breaking process is more favorable than in LDA, and, hence, the bond is weakened. Thus the GGA cures the overbinding error of the LDA.

These gradient corrections greatly improved the bond energies and made density-functional theory useful also for chemists. The most widely distributed GGA functional is the Perdew-Burke-Ernzerhof (PBE) functional [33].

Meta GGA's

The next level of density functionals are the so-called meta GGA's [34, 35, 36] that include not only the gradient of the density, but also the second derivatives of the density. These functionals can be reformulated so that the additional parameter is the kinetic-energy density instead of the second density derivatives. Perdew recommends his TPSS functional [37].

Hybrid functionals

Another generation of functionals are hybrid functionals [38, 39], which replace some of the exchange energy by the exact exchange

$$E_X^{HF} = -\frac{1}{2}\sum_{m,n} \bar{f}_m \bar{f}_n \int d^4x \int d^4x' \frac{e^2 \psi_m^*(\vec{x})\psi_n(\vec{x})\psi_n^*(\vec{x'})\psi_m(\vec{x'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} \tag{1.47}$$

where $\bar{f}_n$ and the $\psi_n(\vec{x})$ are the Kohn-Sham occupations and wave functions, respectively.

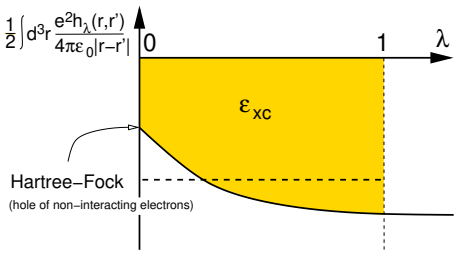


Fig. 1.11: Demonstration of the guiding idea of behind hybrid functionals, namely adiabatic connection. The exchange-correlation energy can be written as an integral of the potential energy of exchange of interaction over the integration strength. This integrand may be approximated by an weighted average of the

The motivation for this approach goes back to the adiabatic connection formula [19, 40, 20]

$$E_{xc}[n(\vec{r})] = \int_0^1 d\lambda\, U_{xc}^{\lambda\hat{W}}[n(\vec{r})] = \int d^3r\, n(\vec{r}) \int_0^1 d\lambda \frac{1}{2} \int d^3r' \frac{h_\lambda(\vec{r},\vec{r'})}{4\pi\epsilon|\vec{r}-\vec{r'}|} \tag{1.48}$$

which expresses the exchange-correlation energy as an integral of the potential energy of exchange and correlation over the interaction strength λ. Here the interaction in the Hamiltonian is scaled by a factor λ, leading to a λ-dependent universal functional $F^{\lambda\hat{W}}[n^{(1)}]$. The interaction energy can be expressed by

$$\begin{aligned} F^{\hat{W}}[n] &= F^0[n] + \int_0^1 d\lambda \, \frac{d}{d\lambda} F^{\lambda\hat{W}}[n] \\ &= T_s[n] + \frac{1}{2} \int d^3r \int d^3r' \, \frac{e^2 n(\vec{r}) n(\vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} + \int_0^1 d\lambda \, U_{xc}^{\lambda\hat{W}}[n] \end{aligned} \tag{1.49}$$

which leads via Eq. 1.32 to Eq. 1.48. Using perturbation theory, the derivative of $F^{\lambda\hat{W}}[n]$ simplifies to the expectation value $\langle\Psi(\lambda)|\hat{W}|\Psi(\lambda)\rangle$ of the interaction, which is the potential energy of exchange and correlation evaluated for a many-particle wave function obtained for the specified given interaction strength.

The underlying idea of the hybrid functionals is to interpolate the integrand between the end points. In the non-interacting limit, i.e. for $\lambda = 0$ the integrand $U_{xc}^{\lambda\hat{W}}$ is exactly given by the exact exchange energy of Eq. 1.47. For the full interaction, on the other hand, the LDA or GGA functionals are considered correctly. Thus a linear interpolation would yield

$$E_{xc} = \frac{1}{2}\left(U_{xc}^0 + U_{xc}^{\hat{W}}\right) = \frac{1}{2}\left(E_X^{HF} + U_{xc}^{\hat{W}}\right) = E_{xc}^{GGA} + \frac{1}{2}\left(E_X^{HF} - E_X^{GGA}\right). \tag{1.50}$$

Depending on whether the λ-dependence is a straight line or whether it is convex, the weight factor may be equal or smaller than $\frac{1}{2}$. Perdew [41] has given arguments that a factor $\frac{1}{4}$ would actually be better than a factor $\frac{1}{2}$.

Hybrid functionals perform substantially better than GGA functionals regarding binding energies, band gaps and reaction energies. However, they are flawed for the description of solids. The reason is that the exact exchange hole in a solid is very extended. These long-range tails are screened away quickly when the interaction is turned on, because they are cancelled by the correlation. Effectively, we should use a smaller mixing factor for the long range part of the exchange hole. This can be taken into account, by cutting off the long-range part of the interaction for the calculation of the Hartree-Fock exchange [42]. This approach improves the results for band gaps while reducing the computational effort [43].

The effective cancellation of the long-ranged contribution of exchange with a similar contribution from correlation, which is also considered properly already in the LDA, is one of the explanation for the superiority of the LDA over the Hartree-Fock approximation.

The most widely used hybrid functional is the B3LYP functional [44], which is, however, obtained from a parameter fit to a database of simple molecules. The functional PBE0 [45, 46] is born out of the famous PBE GGA functional and is a widely distributed parameter-free functional.

LDA+U and local hybrid functionals

Starting from a completely different context, Anisimov et. al. [47] introduced the so-called LDA+U method, which, as described below, has some similarities to the hybrid functionals above.

The main goal was to arrive at a proper description of transition metal oxides, which tend to be Mott insulators, while GGA calculations predict them often to be metals. The remedy was to add a correlation term[9] [48] borrowed from the Hubbard model and to correct the resulting double counting

[9] The expression given here looks unusually simple. This is due to the notation of spin orbitals, which takes care of the spin indices.

of the interactions by E_{dc}.

$$E = E^{GGA} + \frac{1}{2}\sum_R \sum_{\alpha,\beta,\gamma,\delta \in \mathcal{C}_R} U_{\alpha,\beta,\gamma,\delta}\left(\rho_{\gamma,\alpha}\rho_{\delta,\beta} - \rho_{\delta,\alpha}\rho_{\gamma,\beta}\right) - E_{dc} \tag{1.51}$$

$$U_{\alpha,\beta,\gamma,\delta} = \int d^4x \int d^4x' \frac{e^2 \chi^*_\alpha(\vec{x})\chi^*_\beta(\vec{x'})\chi_\gamma(\vec{x})\chi_\delta(\vec{x'})}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} \tag{1.52}$$

$$\rho_{\alpha,\beta} = \langle\pi_\alpha|\psi_n\rangle f_n \langle\psi_n|\pi_\beta\rangle\,, \tag{1.53}$$

where $|\chi_\alpha\rangle$ are atomic tight-binding orbitals and $|\pi_\alpha\rangle$ are their projector functions.[10] The additional energy is a Hartree-Fock exchange energy, that only considers the exchange for specified sets of local orbitals. The exchange term does only consider a subset of orbitals $\mathcal{C}_R$ for each atom R and it ignores the contribution involving orbitals centered on different atoms.

Novak et al. [49] made the connection to the hybrid functionals explicit and restricted the exact exchange contribution of a hybrid functional to only a shell of orbitals. While in the LDA+U method the bare Coulomb matrix elements are reduced by a screening factor, in the hybrid functionals it is the mixing factor that effectively plays the same role. Both LDA+U and the local hybrid method have in common that they radically remove the contribution of off-site matrix elements of the interaction. Tran et al. [50] applied this method to transition metal oxides and found results that are similar to those of the full implementation of hybrid functionals.

Van der Waals interactions

One of the major difficulties for density functionals is the description of van der Waals forces, because it is due to the quantum mechanical synchronization of charge fluctuations on distinct molecules. I refer the reader to the work made in the group of Lundqvist [51, 52, 53].

1.6 Benchmarks, successes and failures

The development of density functionals has profited enormously from careful benchmark studies. The precondition is a data set of test cases for which reliable and accurate experimental data exist. The most famous data sets are the G1 and G2 databases [54, 55, 56, 57] that have been set up to benchmark quantum-chemistry codes. Becke [58, 59, 39, 60, 61] set a trend by using these large sets of test cases for systematic studies of density functionals. In order to separate out the accuracy of the density functionals, it is vital to perform these calculations on extremely accurate numerical methods. Becke used basis-set free calculations that were limited to small molecules while being extremely accurate. Paier et al. [62, 63, 64, 43] have later performed careful comparisons of two methods, Gaussian and the projector augmented-wave method, to single out the error of the electronic structure method.

Overall, the available density functionals predict molecular structures very well. Bond distances agree with the experiment often within one percent. Bond angles come out within a few degrees.

The quality of total energies depends strongly on the level of functionals used. On the LDA level bonds are overestimated in the 1 eV range, on the GGA level these errors are reduced to about 0.3 eV, and hybrid functionals reduce the error by another factor of 2. The ultimate goal is to reach chemical accuracy, which is about 0.05 eV. Such an accuracy allows one to predict reaction rates at room temperature within a factor of 10.

Band gaps are predicted to be too small with LDA and GGA. The so-called band gap problem has been one of the major issues during the development of density functionals. Hybrid functionals clearly

[10] Projector functions obey the bi-orthogonality condition $\langle\chi_\alpha|\pi_\beta\rangle = \delta_{\alpha,\beta}$. Within the sub-Hilbert space of the tight-binding orbitals, i.e. for wave functions of the form $|\psi\rangle = \sum_\alpha |\chi_\alpha\rangle c_\alpha$, the projector functions decompose the wave function into tight binding orbitals, i.e. $|\psi\rangle = \sum_\alpha |\chi_\alpha\rangle\langle\pi_\alpha|\psi\rangle$. A similar projection is used extensively in the projector augmented-wave method described later.

improve the situation. A problem is the description of materials with strong electron correlations. For LDA and GGA many insulating transition metal oxides are described as metals. This changes again for the hybrid functionals, which turns them into antiferromagnetic insulators, which is a dramatic improvement.

1.7 Appendix to chapter DFT

1.7.1 Model exchange-correlation energy

We consider a model with a constant density and a hole function that describes a situation, where all electrons of the same spin are repelled completely from a sphere centered at the reference electron

The hole function has the form

$$h(\vec{r},\vec{r}_0) = \begin{cases} -\frac{1}{2}n(\vec{r}_0) & \text{for } |\vec{r}-\vec{r}_0| < r_h \\ 0 & \text{otherwise} \end{cases}$$

where $n(\vec{r})$ is the electron density and the hole radius $r_h = \sqrt[3]{\frac{2}{4\pi n}}$ is the radius of the sphere, which is determined such that the exchange-correlation hole integrates to -1, i.e. $\frac{4\pi}{3}r_h^3\left(\frac{1}{2}n\right) = 1$.

The potential of a homogeneously charged sphere with radius r_h and one positive charge is

$$v(r) = \frac{e^2}{4\pi\epsilon_0}\begin{cases} -\frac{3}{2r_h} + \frac{1}{2r_h}\left(\frac{r}{r_h}\right)^2 & \text{for } r \le r_h \\ -\frac{1}{r} & \text{for } r > r_h \end{cases}$$

where $r = |\vec{r}-\vec{r}_0|$.

With Eq. 1.14 we obtain for the potential contribution of the exchange-correlation energy

$$U_{xc} = -\int d^3r\; n(\vec{r})v(r=0) = -\int d^3r\; \frac{e^2}{4\pi\epsilon_0}\frac{3}{4}\sqrt[3]{\frac{2\pi}{3}}\cdot n^{\frac{4}{3}}$$

1.7.2 Large-gradient limit of the enhancement factor

An exponentially decaying density

$$n(r) = \exp(-\lambda r) \tag{1.54}$$

has a reduced gradient

$$x := \frac{|\vec{\nabla}n|}{n^{\frac{4}{3}}} = \lambda\exp(+\frac{1}{3}\lambda r) \tag{1.55}$$

We make the following ansatz for the exchange-correlation energy per electron

$$\epsilon_{xc}(n,x) = -Cn^{\frac{1}{3}}F(x) \tag{1.56}$$

where only the local exchange has been used and C is a constant.

Enforcing the long-distance limit of the exchange-correlation energy per electron for exponentially decaying densities

$$\epsilon_{xc}((n(r),x(r)) = -\frac{1}{2}\frac{e^2}{4\pi\epsilon_0 r} \tag{1.57}$$

yields

$$F(x) = \frac{e^2}{4\pi\epsilon_0 r(x)2Cn^{\frac{1}{3}}(r(x))} \tag{1.58}$$

Using Eqs. 1.54 and 1.55, we express the radius and the density by the reduced gradient, i.e.

$$r(x) = -\frac{3}{\lambda}\Big(\ln[\lambda] - \ln[x]\Big) \tag{1.59}$$

$$n(x) = n(r(x)) = \lambda^3 x^{-3}\ , \tag{1.60}$$

and obtain

$$\begin{aligned} F(x) &= \frac{e^2}{4\pi\epsilon_0\left[-\frac{3}{\lambda}\Big(\ln[\lambda]-\ln[x]\Big)\right]\Big[2C\lambda x^{-1}\Big]} = \big(\frac{e^2}{4\pi\epsilon_0\cdot 6C}\big)\frac{x^2}{x\ln(\lambda)-x\ln(x)} \\ &\overset{x\to\infty}{\to} -\left(\frac{e^2}{4\pi\epsilon_0\cdot 6C}\right)\frac{x^2}{x\ln(x)} \end{aligned} \tag{1.61}$$

Now we need to ensure that $F(0) = 1$, so that the gradient correction vanishes for the homogeneous electron gas, and that $F(x) = F(-x)$ to enforce spin reversal symmetry. There are several possible interpolations for these requirements, but the simplest is

$$F(x) = 1 - \frac{\beta x^2}{1 + \frac{4\pi\epsilon_0}{e^2}\cdot 6C\beta x\cdot \mathrm{asinh}(x)} \tag{1.62}$$

This is the enhancement factor for exchange used by Becke in his B88 functional [32].

Chapter 2

Electronic structure methods and the PAW method

This section is related to earlier versions [10, 11] written together with J. Kästner and C. Först.

2.1 Introduction

The main goal of electronic structure methods is to solve the Schrödinger equation for the electrons in a molecule or solid, to evaluate the resulting total energies, forces, response functions and other quantities of interest. In this chapter we describe the basic ideas behind the main electronic structure methods such as the pseudopotential and the augmented wave methods and provide selected pointers to contributions that are relevant for a beginner. We give particular emphasis to the Projector Augmented Wave (PAW) method developed by one of us, an electronic structure method for ab-initio molecular dynamics with full wavefunctions. We feel that provides the best insight into the common conceptional basis of the most widespread electronic structure methods in materials science.

The methods described below require as input only the charge and mass of the nuclei, the number of electrons and an initial atomic geometry. They predict binding energies accurate within a few tenths of an electron volt and bond-lengths in the 1-2 percent range. Currently, systems with few hundred atoms per unit cell can be handled. The dynamics of atoms can be studied up to tens of pico-seconds. Quantities related to energetics, the atomic structure and to the ground-state electronic structure can be extracted.

In order to lay a common ground and to define some of the symbols, let us briefly touch upon the density-functional theory [2, 3]. It maps a description for interacting electrons, a nearly intractable problem, onto one of non-interacting electrons in an effective potential. Within density-functional theory, the total energy is written as

$$E\Big[\{\psi_n(\vec{r})\},\{\vec{R}_R\}\Big] = \sum_n f_n \langle\psi_n|\frac{\hat{\vec{p}}^2}{2m_e}|\psi_n\rangle + \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2\Big(n(\vec{r})+Z(\vec{r})\Big)\Big(n(\vec{r'})+Z(\vec{r'})\Big)}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + E_{xc}[n] \;, \tag{2.1}$$

where $Z(\vec{r}) = -\sum_R \mathcal{Z}_R \delta(\vec{r}-\vec{R}_R)$ is the nuclear charge density expressed in electron charges. Z_R is the atomic number of a nucleus at position $\vec{R}_R$.

The electronic ground state is determined by minimizing the total energy functional $E[\Psi_n]$ of Eq. 2.1 at a fixed ionic geometry. The one-particle wave functions have to be orthogonal. This constraint is implemented with the method of Lagrange multipliers. We obtain the ground-state

wave functions from the extremum condition for

$$Y\Big[\{|\psi_n\rangle\},\mathbf{\Lambda}\Big] = E\Big[\{|\psi_n\rangle\}\Big] - \sum_{n,m}\Big[\langle\psi_n|\psi_m\rangle - \delta_{n,m}\Big]\Lambda_{m,n} \tag{2.2}$$

with respect to the wavefunctions and the Lagrange multipliers $\Lambda_{m,n}$. The extremum condition for the wavefunctions has the form

$$\hat{H}|\psi_n\rangle f_n = \sum_m |\psi_m\rangle\Lambda_{m,n} \ , \tag{2.3}$$

where $\hat{H} = \frac{1}{2m_e}\hat{\vec{p}}^2 + \hat{v}_{\text{eff}}$ is the effective one-particle Hamilton operator.

The corresponding effective potential depends itself on the electron density via

$$v_{eff}(\vec{r}) = \int d^3r' \ \frac{e^2\Big(n(\vec{r'}) + Z(\vec{r'})\Big)}{4\pi\epsilon_0|\vec{r}-\vec{r'}|} + \mu_{xc}(\vec{r}) \ , \tag{2.4}$$

where $\mu_{xc}(\vec{r}) = \frac{\delta E_{xc}[n(\vec{r})]}{\delta n(\vec{r})}$ is the functional derivative of the exchange and correlation functional.

After a unitary transformation that diagonalizes the matrix of Lagrange multipliers $\mathbf{\Lambda}$, we obtain the Kohn-Sham equations

$$\hat{H}|\psi_n\rangle = |\psi_n\rangle\epsilon_n \ . \tag{2.5}$$

The one-particle energies ϵ_n are the eigenvalues of the matrix with the elements $\Lambda_{n,m}(f_n+f_m)/(2f_nf_m)$ [65].

The one-electron Schrödinger equations, namely the Kohn-Sham equations given in Eq. 1.24, still pose substantial numerical difficulties: (1) in the atomic region near the nucleus, the kinetic energy of the electrons is large, resulting in rapid oscillations of the wavefunction that require fine grids for an accurate numerical representation. On the other hand, the large kinetic energy makes the Schrödinger equation stiff, so that a change of the chemical environment has little effect on the shape of the wavefunction. Therefore, the wavefunction in the atomic region can be represented well already by a small basis set. (2) In the bonding region between the atoms the situation is opposite. The kinetic energy is small and the wavefunction is smooth. However, the wavefunction is flexible and responds strongly to the environment. This requires large and nearly complete basis sets.

Combining these different requirements is non-trivial and various strategies have been developed.

- The atomic point of view has been most appealing to quantum chemists. Basis functions are chosen that resemble atomic orbitals. This choice exploits that the wavefunction in the atomic region can be described by a few basis functions, while the chemical bond is described by the overlapping tails of these atomic orbitals. Most techniques in this class are a compromise of, on the one hand, a well adapted basis set, where the basis functions are difficult to handle, and, on the other hand, numerically convenient basis functions such as Gaussians, where the inadequacies are compensated by larger basis sets.

- Pseudopotentials regard an atom as a perturbation of the free electron gas. The most natural basis functions for the free electron gas are plane waves. Plane-wave basis sets are in principle complete and suitable for sufficiently smooth wavefunctions. The disadvantage of the comparably large basis sets required is offset by their extreme numerical simplicity. Finite plane-wave expansions are, however, absolutely inadequate to describe the strong oscillations of the wavefunctions near the nucleus. In the pseudopotential approach the Pauli repulsion by the core electrons is therefore described by an effective potential that expels the valence electrons from the core region. The resulting wavefunctions are smooth and can be represented well by plane waves. The price to pay is that all information on the charge density and wavefunctions near the nucleus is lost.

- Augmented-wave methods compose their basis functions from atom-like wavefunctions in the atomic regions and a set of functions, called envelope functions, appropriate for the bonding in between. Space is divided accordingly into atom-centered spheres, defining the atomic regions, and an interstitial region in between. The partial solutions of the different regions are matched with value and derivative at the interface between atomic and interstitial regions.

The projector augmented-wave method is an extension of augmented wave methods and the pseudopotential approach, which combines their traditions into a unified electronic structure method.

After describing the underlying ideas of the various approaches, let us briefly review the history of augmented wave methods and the pseudopotential approach. We do not discuss the atomic-orbital based methods, because our focus is the PAW method and its ancestors.

2.2 Augmented wave methods

The augmented wave methods have been introduced in 1937 by Slater [66]. His method was called augmented plane-wave (APW) method. Later Korringa [67], Kohn and Rostoker [68] modified the idea, which lead to the so-called KKR method. The basic idea behind the augmented wave methods has been to consider the electronic structure as a scattered-electron problem: Consider an electron beam, represented by a plane wave, traveling through a solid. It undergoes multiple scattering at the atoms. If, for some energy, the outgoing scattered waves interfere destructively, so that the electrons can not escape, a bound state has been determined. This approach can be translated into a basis-set method with energy- and potential-dependent basis functions. In order to make the scattered wave problem tractable, a model potential had to be chosen: The so-called muffin-tin potential approximates the true potential by a potential that is spherically symmetric in the atomic regions, and constant in between.

Augmented-wave methods reached adulthood in the 1970s: O. K. Andersen [69] showed that the energy dependent basis set of Slater's APW method can be mapped onto one with energy independent basis functions by linearizing the partial waves for the atomic regions with respect to their energy. In the original APW approach, one had to determine the zeros of the determinant of an energy dependent matrix, a nearly intractable numerical problem for complex systems. With the new energy independent basis functions, however, the problem is reduced to the much simpler generalized eigenvalue problem, which can be solved using efficient numerical techniques. Furthermore, the introduction of well defined basis sets paved the way for full-potential calculations [70]. In that case, the muffin-tin approximation is used solely to define the basis set $|\chi_i\rangle$, while the matrix elements $\langle\chi_i|H|\chi_j\rangle$ of the Hamiltonian are evaluated with the full potential.

In the augmented wave methods one constructs the basis set for the atomic region by solving the radial Schrödinger equation for the spherically averaged effective potential

$$\left[\frac{-\hbar^2}{2m_e}\vec{\nabla}^2 + v_{eff}(\vec{r}) - \epsilon\right]\phi_{\ell,m}(\epsilon,\vec{r}) = 0 \tag{2.6}$$

as function of the energy. Note that a partial wave $\phi_{\ell,m}(\epsilon,\vec{r})$ is an angular-momentum eigenstate and can be expressed as a product of a radial function and a spherical harmonic. The energy-dependent partial wave is expanded in a Taylor expansion about some reference energy $\epsilon_{\nu,\ell}$

$$\phi_{\ell,m}(\epsilon,\vec{r}) = \phi_{\nu,\ell,m}(\vec{r}) + (\epsilon-\epsilon_{\nu,\ell})\dot{\phi}_{\nu,\ell,m}(\vec{r}) + O((\epsilon-\epsilon_{\nu,\ell})^2)\ , \tag{2.7}$$

where $\phi_{\nu,\ell,m}(\vec{r}) = \phi_{\ell,m}(\epsilon_{\nu,\ell},\vec{r})$. The energy derivative of the partial wave $\dot{\phi}_{\nu,\ell,m}(\vec{r}) = \left.\frac{\partial\phi_{\ell,m}(\epsilon,\vec{r})}{\partial\epsilon}\right|_{\epsilon_{\nu,\ell}}$ is obtained from the energy derivative of the Schrödinger equation

$$\left[\frac{-\hbar^2}{2m_e}\vec{\nabla}^2 + v_{eff}(\vec{r}) - \epsilon_{\nu,\ell}\right]\dot{\phi}_{\nu,\ell,m}(\vec{r}) = \phi_{\nu,\ell,m}(\vec{r})\ . \tag{2.8}$$

Next, one starts from a regular basis set, such as plane waves, Gaussians or Hankel functions. These basis functions are called envelope functions $|\tilde{\chi}_i\rangle$. Within the atomic region they are replaced by the partial waves and their energy derivatives, such that the resulting wavefunction $\chi_i(\vec{r})$ is continuous and differentiable. The augmented envelope function has the form

$$\chi_i(\vec{r}) = \tilde{\chi}_i(\vec{r}) - \sum_R \theta_R(\vec{r})\tilde{\chi}_i(\vec{r}) + \sum_{R,\ell,m} \theta_R(\vec{r}) \left[\phi_{\nu,R,\ell,m}(\vec{r})a_{R,\ell,m,i} + \dot{\phi}_{\nu,R,\ell,m}(\vec{r})b_{R,\ell,m,i}\right] . \quad (2.9)$$

$\theta_R(\vec{r})$ is a step function that is unity within the augmentation sphere centered at $\vec{R}_R$ and zero elsewhere. The augmentation sphere is atom-centered and has a radius about equal to the covalent radius. This radius is called the muffin-tin radius, if the spheres of neighboring atoms touch. These basis functions describe only the valence states; the core states are localized within the augmentation sphere and are obtained directly by a radial integration of the Schrödinger equation within the augmentation sphere.

The coefficients $a_{R,\ell,m,i}$ and $b_{R,\ell,m,i}$ are obtained for each $|\tilde{\chi}_i\rangle$ as follows: The envelope function is decomposed around each atomic site into spherical harmonics multiplied by radial functions

$$\tilde{\chi}_i(\vec{r}) = \sum_{\ell,m} u_{R,\ell,m,i}(|\vec{r} - \vec{R}_R|)Y_{\ell,m}(\vec{r} - \vec{R}_R) . \quad (2.10)$$

Analytical expansions for plane waves, Hankel functions or Gaussians exist. The radial parts of the partial waves $\phi_{\nu,R,\ell,m}$ and $\dot{\phi}_{\nu,R,\ell,m}$ are matched with value and derivative to $u_{R,\ell,m,i}(|\vec{r}|)$, which yields the expansion coefficients $a_{R,\ell,m,i}$ and $b_{R,\ell,m,i}$.

If the envelope functions are plane waves, the resulting method is called the linear augmented plane-wave (LAPW) method. If the envelope functions are Hankel functions, the method is called linear muffin-tin orbital (LMTO) method.

A good review of the LAPW method [69] has been given by Singh [71]. Let us now briefly mention the major developments of the LAPW method: Soler [72] introduced the idea of additive augmentation: While augmented plane waves are discontinuous at the surface of the augmentation sphere if the expansion in spherical harmonics in Eq. 2.9 is truncated, Soler replaced the second term in Eq. 2.9 by an expansion of the plane wave with the same angular momentum truncation as in the third term. This dramatically improved the convergence of the angular momentum expansion. Singh [73] introduced so-called local orbitals, which are non-zero only within a muffin-tin sphere, where they are superpositions of ϕ and $\dot{\phi}$ functions from different expansion energies. Local orbitals substantially increase the energy transferability. Sjöstedt [74] relaxed the condition that the basis functions are differentiable at the sphere radius. In addition she introduced local orbitals, which are confined inside the sphere, and that also have a kink at the sphere boundary. Due to the large energy cost of kinks, they will cancel, once the total energy is minimized. The increased variational degree of freedom in the basis leads to a dramatically improved plane-wave convergence [75].

The second variant of the linear methods is the LMTO method [69]. A good introduction into the LMTO method is the book by Skriver [76]. The LMTO method uses Hankel functions as envelope functions. The atomic spheres approximation (ASA) provides a particularly simple and efficient approach to the electronic structure of very large systems. In the ASA the augmentation spheres are blown up so that the sum of their volumes is equal to the total volume. Then, the first two terms in Eq. 2.9 are ignored. The main deficiency of the LMTO-ASA method is the limitation to structures that can be converted into a closed packed arrangement of atomic and empty spheres. Furthermore, energy differences due to structural distortions are often qualitatively incorrect. Full potential versions of the LMTO method, that avoid these deficiencies of the ASA have been developed. The construction of tight binding orbitals as superposition of muffin-tin orbitals [77] showed the underlying principles of the empirical tight-binding method and prepared the ground for electronic structure methods that scale linearly instead of with the third power of the number of atoms. The third generation LMTO [78] allows to construct true minimal basis sets, which require only one orbital per electron pair for insulators. In addition, they can be made arbitrarily accurate in the valence band region, so that a matrix diagonalization becomes unnecessary. The

first steps towards a full-potential implementation, that promises a good accuracy, while maintaining the simplicity of the LMTO-ASA method, are currently under way. Through the minimal basis-set construction the LMTO method offers unrivaled tools for the analysis of the electronic structure and has been extensively used in hybrid methods combining density-functional theory with model Hamiltonians for materials with strong electron correlations [79].

2.3 Pseudopotentials

Pseudopotentials have been introduced to (1) avoid describing the core electrons explicitly and (2) to avoid the rapid oscillations of the wavefunction near the nucleus, which normally require either complicated or large basis sets.

The pseudopotential approach can be traced back to 1940 when C. Herring invented the orthogonalized plane-wave method [80]. Later, Phillips [81] and Antončík [82] replaced the orthogonality condition by an effective potential, which mimics the Pauli repulsion by the core electrons and thus compensates the electrostatic attraction by the nucleus. In practice, the potential was modified, for example, by cutting off the singular potential of the nucleus at a certain value. This was done with a few parameters that have been adjusted to reproduce the measured electronic band structure of the corresponding solid.

Hamann, Schlüter and Chiang [83] showed in 1979 how pseudopotentials can be constructed in such a way that their scattering properties are identical to that of an atom to first order in energy. These first-principles pseudopotentials relieved the calculations from the restrictions of empirical parameters. Highly accurate calculations have become possible especially for semiconductors and simple metals. An alternative approach towards first-principles pseudopotentials by Zunger and Cohen[84] even preceded the one mentioned above.

The idea behind the pseudopotential construction

In order to construct a first-principles pseudopotential, one starts out with an all-electron density-functional calculation for a spherical atom. Such calculations can be performed efficiently on radial grids. They yield the atomic potential and wavefunctions $\phi_{\ell,m}(\vec{r})$. Due to the spherical symmetry, the radial parts of the wavefunctions for different magnetic quantum numbers m are identical.

For the valence wavefunctions one constructs pseudo wavefunctions $|\tilde{\phi}_{\ell,m}\rangle$. There are numerous ways [85, 86, 87, 88] to construct those pseudo wavefunctions: Pseudo wave functions are identical to the true wave functions outside the augmentation region, which is called core region in the context of the pseudopotential approach. Inside the augmentation region the pseudo wavefunction should be nodeless and have the same norm as the true wavefunctions, that is $\langle\tilde{\phi}_{\ell,m}|\tilde{\phi}_{\ell,m}\rangle = \langle\phi_{\ell,m}|\phi_{\ell,m}\rangle$ (compare Figure 2.1).

From the pseudo wavefunction, a potential $u_\ell(\vec{r})$ can be reconstructed by inverting the respective Schrödinger equation, i.e.

$$\left[-\frac{\hbar^2}{2m_e}\vec{\nabla}^2 + u_\ell(\vec{r}) - \epsilon_{\ell,m}\right]\tilde{\phi}_{\ell,m}(\vec{r}) = 0 \quad \Rightarrow \quad u_\ell(\vec{r}) = \epsilon + \frac{1}{\tilde{\phi}_{\ell,m}(\vec{r})}\cdot\frac{\hbar^2}{2m_e}\vec{\nabla}^2\tilde{\phi}_{\ell,m}(\vec{r})\ . \quad (2.11)$$

This potential $u_\ell(\vec{r})$ (compare Figure 2.1), which is also spherically symmetric, differs from one main angular momentum ℓ to the other. Note, that this *inversion of the Schrödinger equation* works only if the wave functions are nodeless.

Next we define an effective pseudo Hamiltonian

$$\hat{\tilde{H}}_\ell = -\frac{\hbar^2}{2m_e}\vec{\nabla}^2 + v_\ell^{ps}(\vec{r}) + \int d^3r'\frac{e^2\left(\tilde{n}(\vec{r}') + \tilde{Z}(\vec{r}')\right)}{4\pi\epsilon_0|\vec{r}-\vec{r}'|} + \mu_{xc}([n(\vec{r})],\vec{r})\ , \quad (2.12)$$

where $\mu_{xc}(\vec{r}) = \delta E_{xc}[n]/\delta n(\vec{r})$ is the functional derivative of the exchange and correlation energy with respect to the electron density. Then, we determine the pseudopotentials v_ℓ^{ps} such that the

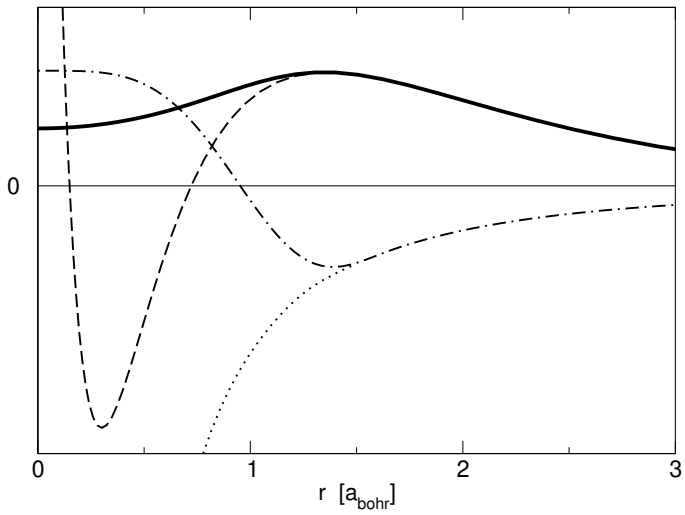


Fig. 2.1: Illustration of the pseudopotential concept at the example of the 3s wavefunction of Si. The solid line shows the radial part of the pseudo wavefunction $\tilde{\phi}_{\ell,m}$. The dashed line corresponds to the all-electron wavefunction $\phi_{\ell,m}$, which exhibits strong oscillations at small radii. The angular momentum dependent pseudopotential u_ℓ (dash-dotted line) deviates from the all-electron potential v_{eff} (dotted line) inside the augmentation region. The data are generated by the fhi98PP code [89].

pseudo Hamiltonian produces the pseudo wavefunctions, that is

$$v_\ell^{ps}(\vec{r}) = u_\ell(\vec{r}) - \int d^3r' \, \frac{e^2\left(\tilde{n}(\vec{r}') + \tilde{Z}(\vec{r}')\right)}{4\pi\epsilon_0|\vec{r} - \vec{r}'|} - \mu_{xc}([\tilde{n}(\vec{r})], \vec{r}) \; . \tag{2.13}$$

This process is called "unscreening".

$\tilde{Z}(\vec{r})$ mimics the charge density of the nucleus and the core electrons. It is usually an atom-centered, spherical Gaussian that is normalized to the charge of nucleus and core electrons of that atom. In the pseudopotential approach, $\tilde{Z}_R(\vec{r})$ does not change with the potential. The pseudo density $\tilde{n}(\vec{r}) = \sum_n f_n \tilde{\psi}_n^*(\vec{r})\tilde{\psi}_n(\vec{r})$ is constructed from the pseudo wavefunctions.

In this way, we obtain a different potential for each angular momentum channel. In order to apply these potentials to a given wavefunction, the wavefunction must first be decomposed into angular momenta. Then each component is applied to the pseudopotential v_ℓ^{ps} for the corresponding angular momentum.

The pseudopotential defined in this way can be expressed in a semi-local form[1]

$$v^{ps}(\vec{r}, \vec{r}') = \bar{v}(\vec{r})\delta(\vec{r} - \vec{r}') + \sum_{\ell,m} \left[Y_{\ell,m}(\vec{r}) \left[v_\ell^{ps}(\vec{r}) - \bar{v}(\vec{r}) \right] \frac{\delta(|\vec{r}| - |\vec{r}'|)}{|\vec{r}|^2} Y_{\ell,m}^*(\vec{r}') \right] \; . \tag{2.14}$$

The local potential $\bar{v}(\vec{r})$ only acts on those angular momentum components that are not already considered explicitly in the non-local, angular-momentum dependent pseudopotentials v_ℓ^{ps}. Typically it is chosen to cancel the most expensive nonlocal terms, the one corresponding to the highest physically relevant angular momentum.

The pseudopotential $v^{ps}(\vec{r}, \vec{r}')$ is non-local as its depends on two position arguments, $\vec{r}$ and $\vec{r}'$. The expectation values are evaluated as a double integral

$$\langle\tilde{\psi}|\hat{v}_{ps}|\tilde{\psi}\rangle = \int d^3r \int d^3r' \; \tilde{\psi}^*(\vec{r}) v^{ps}(\vec{r}, \vec{r}')\tilde{\psi}(\vec{r}') \tag{2.15}$$

[1] A semi-local potential is local in the radial coordinate, but non-local in the angular coordinates.

The semi-local form of the pseudopotential given in Eq. 2.14 is computationally expensive. Therefore, in practice one uses a separable form of the pseudopotential [90, 91, 92]

$$\hat{v}^{ps} \approx \sum_{i,j} \hat{v}^{ps}|\tilde{\phi}_i\rangle \left[\langle\tilde{\phi}_j|\hat{v}^{ps}|\tilde{\phi}_i\rangle\right]^{-1}_{i,j} \langle\tilde{\phi}_j|\hat{v}^{ps} \ . \tag{2.16}$$

Thus, the projection onto spherical harmonics used in the semi-local form of Eq. 2.14 is replaced by a projection onto angular momentum dependent functions $\hat{v}^{ps}|\tilde{\phi}_i\rangle$.

The indices i and j are composite indices containing the atomic-site index R, the angular momentum quantum numbers ℓ, m and an additional index α. The index α distinguishes partial waves with otherwise identical indices R, ℓ, m when more than one partial wave per site and angular momentum is allowed. The partial waves may be constructed as eigenstates of the hamiltonian with the pseudopotential $\hat{v}^{ps}_{\ell}$ for a set of energies.

One can show that the identity of Eq. 2.16 holds by applying a wavefunction $|\tilde{\psi}\rangle = \sum_i |\tilde{\phi}_i\rangle c_i$ to both sides. If the set of pseudo partial waves $|\tilde{\phi}_i\rangle$ in Eq. 2.16 is complete, the identity is exact. The advantage of the separable form is that $\langle\tilde{\phi}|\hat{v}^{ps}$ is treated as one function, so that expectation values are reduced to combinations of simple scalar products $\langle\tilde{\phi}_i|\hat{v}^{ps}|\tilde{\psi}\rangle$.

The total energy of the pseudopotential method can be written in the form

$$\begin{aligned} E = &\sum_n f_n\langle\tilde{\psi}_n|\frac{\hat{\vec{p}}^2}{2m_e}|\tilde{\psi}_n\rangle + E_{self} + \sum_n f_n\langle\tilde{\psi}_n|\hat{v}_{ps}|\tilde{\psi}_n\rangle \\ &+\frac{1}{2}\int d^3r \int d^3r' \frac{e^2\Big(\tilde{n}(\vec{r}) + \tilde{Z}(\vec{r})\Big)\Big(\tilde{n}(\vec{r}') + \tilde{Z}(\vec{r}')\Big)}{4\pi\epsilon_0|\vec{r}-\vec{r}'|} + E_{xc}[\tilde{n}(\vec{r})] \ . \end{aligned} \tag{2.17}$$

The constant E_{self} is adjusted such that the total energy of the atom is the same for an all-electron calculation and the pseudopotential calculation.

For the atom, from which it has been constructed, this construction guarantees that the pseudopotential method produces the correct one-particle energies for the valence states and that the wave functions have the desired shape.

While pseudopotentials have proven to be accurate for a large variety of systems, there is no strict guarantee that they produce the same results as an all-electron calculation, if they are used in a molecule or solid. The error sources can be divided into two classes:

- Energy transferability problems: Even for the potential of the reference atom, the scattering properties are accurate only in given energy window.
- Charge transferability problems: In a molecule or crystal, the potential differs from that of the isolated atom. The pseudopotential, however, is strictly valid only for the isolated atom.

The plane-wave basis set for the pseudo wavefunctions is defined by the shortest wave length $\lambda = 2\pi/|\vec{G}|$, where $\vec{G}$ is the wave vector, via the so-called plane-wave cutoff $E_{PW} = \frac{\hbar^2 G^2_{max}}{2m_e}$ with $G_{max} = \max\{|\vec{G}|\}$. It is often specified in Rydberg (1 Ry=$\frac{1}{2}$ H≈13.6 eV). The plane-wave cutoff is the highest kinetic energy of all basis functions. The basis-set convergence can systematically be controlled by increasing the plane-wave cutoff.

The charge transferability is substantially improved by including a nonlinear core correction [93] into the exchange-correlation term of Eq. 2.17. Hamann [94] showed how to construct pseudopotentials also from unbound wavefunctions. Vanderbilt [95, 96] generalized the pseudopotential method to non-normconserving pseudopotentials, so-called ultra-soft pseudopotentials, which dramatically improves the basis-set convergence. The formulation of ultra-soft pseudopotentials has already many similarities with the projector augmented-wave method. Truncated separable pseudopotentials suffer sometimes from so-called ghost states. These are unphysical core-like states, which render the pseudopotential useless. These problems have been discussed by Gonze [97]. Quantities such as hyperfine parameters that depend on the full wavefunctions near the nucleus, can be extracted approximately

[98]. Good reviews of the pseudopotential methodology have been written by Payne *et al.*[99] and Singh [71].

In 1985 R. Car and M. Parrinello published the ab-initio molecular dynamics method [100]. Simulations of the atomic motion have become possible on the basis of state-of-the-art electronic structure methods. Besides making dynamical phenomena and finite temperature effects accessible to electronic structure calculations, the ab-initio molecular dynamics method also introduced a radically new way of thinking into electronic structure methods. Diagonalization of a Hamilton matrix has been replaced by classical equations of motion for the wavefunction coefficients. If one applies friction, the system is quenched to the ground state. Without friction truly dynamical simulations of the atomic structure are performed. By using thermostats [101, 102, 103, 104], simulations at constant temperature can be performed. The Car-Parrinello method treats electronic wavefunctions and atomic positions on an equal footing.

2.4 Projector augmented-wave method

The Car-Parrinello method had been implemented first for the pseudopotential approach. There seemed to be insurmountable barriers against combining the new technique with augmented wave methods. The main problem was related to the potential dependent basis set used in augmented wave methods: the Car-Parrinello method requires a well defined and unique total energy functional of atomic positions and basis set coefficients. Furthermore the analytic evaluation of the first partial derivatives of the total energy with respect to wave functions, $\frac{\partial E}{\partial \langle \psi_n |} = \hat{H}|\psi_n\rangle f_n$, and atomic positions, the forces $\vec{F}_j = -\vec{\nabla}_j E$, must be possible. Therefore, it was one of the main goals of the PAW method to introduce energy and potential independent basis sets, which were as accurate as the previously used augmented basis sets. Other requirements have been: (1) The method should at least match the efficiency of the pseudopotential approach for Car-Parrinello simulations. (2) It should become an exact theory when converged and (3) its convergence should be easily controlled. We believe that these criteria have been met, which explains why the use of the PAW method has become increasingly widespread today.

Transformation theory

At the root of the PAW method lies a transformation that maps the true wavefunctions with their complete nodal structure onto auxiliary wavefunctions that are numerically convenient. We aim for smooth auxiliary wavefunctions, which have a rapidly convergent plane-wave expansion. With such a transformation we can expand the auxiliary wave functions into a convenient basis set such as plane waves, and evaluate all physical properties after reconstructing the related physical (true) wavefunctions.

Let us denote the physical one-particle wavefunctions as $|\psi_n\rangle$ and the auxiliary wavefunctions as $|\tilde{\psi}_n\rangle$. Note that the tilde refers to the representation of smooth auxiliary wavefunctions and n is the label for a one-particle state and contains a band index, a k-point and a spin index. The transformation from the auxiliary to the physical wave functions is denoted by $\hat{\mathcal{T}}$, i.e.

$$|\psi_n\rangle = \hat{\mathcal{T}}|\tilde{\psi}_n\rangle \,. \tag{2.18}$$

Now we express the constrained density functional F of Eq. 2.2 in terms of our auxiliary wavefunctions

$$F\Big[\{\hat{\mathcal{T}}|\tilde{\psi}_n\rangle\}, \{\Lambda_{m,n}\}\Big] = E\Big[\{\hat{\mathcal{T}}|\tilde{\psi}_n\rangle\}\Big] - \sum_{n,m}\Big[\langle\tilde{\psi}_n|\hat{\mathcal{T}}^\dagger\hat{\mathcal{T}}|\tilde{\psi}_m\rangle - \delta_{n,m}\Big]\Lambda_{m,n} \,. \tag{2.19}$$

The variational principle with respect to the auxiliary wavefunctions yields

$$\hat{\mathcal{T}}^\dagger\hat{H}\hat{\mathcal{T}}|\tilde{\psi}_n\rangle = \hat{\mathcal{T}}^\dagger\hat{\mathcal{T}}|\tilde{\psi}_n\rangle\epsilon_n \,. \tag{2.20}$$

Again, we obtain a Schrödinger-like equation (see derivation of Eq. 2.5), but now the Hamilton operator has a different form, $\hat{\tilde{H}} = \hat{\mathcal{T}}^\dagger \hat{H} \hat{\mathcal{T}}$, an overlap operator $\hat{\tilde{O}} = \hat{\mathcal{T}}^\dagger \hat{\mathcal{T}}$ occurs, and the resulting auxiliary wavefunctions are smooth.

When we evaluate physical quantities, we need to evaluate expectation values of an operator $\hat{A}$, which can be expressed in terms of either the true or the auxiliary wavefunctions, i.e.

$$\langle \hat{A} \rangle = \sum_n f_n \langle \psi_n | \hat{A} | \psi_n \rangle = \sum_n f_n \langle \tilde{\psi}_n | \hat{\mathcal{T}}^\dagger \hat{A} \hat{\mathcal{T}} | \tilde{\psi}_n \rangle \,. \tag{2.21}$$

In the representation of auxiliary wavefunctions we need to use transformed operators $\hat{\tilde{A}} = \hat{\mathcal{T}}^\dagger \hat{A} \hat{\mathcal{T}}$. As it is, this equation only holds for the valence electrons. The core electrons are treated differently, as will be shown below.

The transformation takes us conceptionally from the world of pseudopotentials to that of augmented wave methods, which deal with the full wavefunctions. We will see that our auxiliary wavefunctions, which are simply the plane-wave parts of the full wavefunctions, translate into the wavefunctions of the pseudopotential approach. In the PAW method the auxiliary wavefunctions are used to construct the true wavefunctions and the total energy functional is evaluated from the latter. Thus it provides the missing link between augmented wave methods and the pseudopotential method, which can be derived as a well-defined approximation of the PAW method.

In the original paper [65], the auxiliary wavefunctions were termed pseudo wavefunctions and the true wavefunctions were termed all-electron wavefunctions, in order to make the connection more evident. We avoid this notation here, because it resulted in confusion in cases, where the correspondence is not clear-cut.

Transformation operator

So far we have described how we can determine the auxiliary wave functions of the ground state and how to obtain physical information from them. What is missing is a definition of the transformation operator $\hat{\mathcal{T}}$.

The operator $\hat{\mathcal{T}}$ has to modify the smooth auxiliary wave function in each atomic region, so that the resulting wavefunction has the correct nodal structure. Therefore, it makes sense to write the transformation as identity plus a sum of atomic contributions $\hat{\mathcal{S}}_R$

$$\hat{\mathcal{T}} = \hat{1} + \sum_R \hat{\mathcal{S}}_R. \tag{2.22}$$

For every atom, $\hat{\mathcal{S}}_R$ adds the difference between the true and the auxiliary wavefunction.

The local terms $\hat{\mathcal{S}}_R$ are defined in terms of solutions $|\phi_i\rangle$ of the Schrödinger equation for the isolated atoms. This set of partial waves $|\phi_i\rangle$ will serve as a basis set so that, near the nucleus, all relevant valence wavefunctions can be expressed as superposition of the partial waves with yet unknown coefficients as

$$\psi(\vec{r}) = \sum_{i \in R} \phi_i(\vec{r}) c_i \quad \text{for} \quad |\vec{r} - \vec{R}_R| < r_{c,R} \,. \tag{2.23}$$

With $i \in R$ we indicate those partial waves that belong to site R.

Since the core wavefunctions do not spread out into the neighboring atoms, we will treat them differently. Currently we use the frozen-core approximation, which imports the density and the energy of the core electrons from the corresponding isolated atoms. The transformation $\hat{\mathcal{T}}$ shall produce only wavefunctions orthogonal to the core electrons, while the core electrons are treated separately. Therefore, the set of atomic partial waves $|\phi_i\rangle$ includes only valence states that are orthogonal to the core wavefunctions of the atom.

For each of the partial waves we choose an auxiliary partial wave $|\tilde{\phi}_i\rangle$. The identity

$$\begin{aligned} |\phi_i\rangle &= (\hat{1} + \hat{\mathcal{S}}_R)|\tilde{\phi}_i\rangle \quad \text{for} \quad i \in R \\ \hat{\mathcal{S}}_R|\tilde{\phi}_i\rangle &= |\phi_i\rangle - |\tilde{\phi}_i\rangle \end{aligned} \tag{2.24}$$

defines the local contribution $\hat{\mathcal{S}}_R$ to the transformation operator. Since $\hat{1} + \hat{\mathcal{S}}_R$ should change the wavefunction only locally, we require that the partial waves $|\phi_i\rangle$ and their auxiliary counter parts $|\tilde{\phi}_i\rangle$ are pairwise identical beyond a certain radius $r_{c,R}$.

$$\phi_i(\vec{r}) = \tilde{\phi}_i(\vec{r}) \quad \text{for} \quad i \in R \quad \text{and} \quad |\vec{r} - \vec{R}_R| > r_{c,R} \tag{2.25}$$

Note that the partial waves are not necessarily bound states and are therefore not normalizable unless we truncate them beyond a certain radius $r_{c,R}$. The PAW method is formulated such that the final results do not depend on the location where the partial waves are truncated, as long as this is not done too close to the nucleus and identical for auxiliary and all-electron partial waves.

In order to be able to apply the transformation operator to an arbitrary auxiliary wavefunction, we need to be able to expand the auxiliary wavefunction locally into the auxiliary partial waves

$$\tilde{\psi}(\vec{r}) = \sum_{i \in R} \tilde{\phi}_i(\vec{r}) c_i = \sum_{i \in R} \tilde{\phi}_i(\vec{r}) \langle \tilde{p}_i | \tilde{\psi} \rangle \quad \text{for} \quad |\vec{r} - \vec{R}_R| < r_{c,R} \ , \tag{2.26}$$

which defines the projector functions $|\tilde{p}_i\rangle$. The projector functions probe the local character of the auxiliary wave function in the atomic region. Examples of projector functions are shown in Figure 2.2. From Eq. 2.26 we can derive $\sum_{i \in R} |\tilde{\phi}_i\rangle\langle\tilde{p}_i| = 1$, which is valid within $r_{c,R}$. It can be shown by insertion, that the identity Eq. 2.26 holds for any auxiliary wavefunction $|\tilde{\psi}\rangle$ that can be expanded locally into auxiliary partial waves $|\tilde{\phi}_i\rangle$, if

$$\langle \tilde{p}_i | \tilde{\phi}_j \rangle = \delta_{i,j} \quad \text{for} \quad i, j \in R \, . \tag{2.27}$$

Note that neither the projector functions nor the partial waves need to be mutually orthogonal. The projector functions are fully determined with the above conditions and a closure relation that is related to the unscreening of the pseudopotentials (see Eq. 90 in [65]).

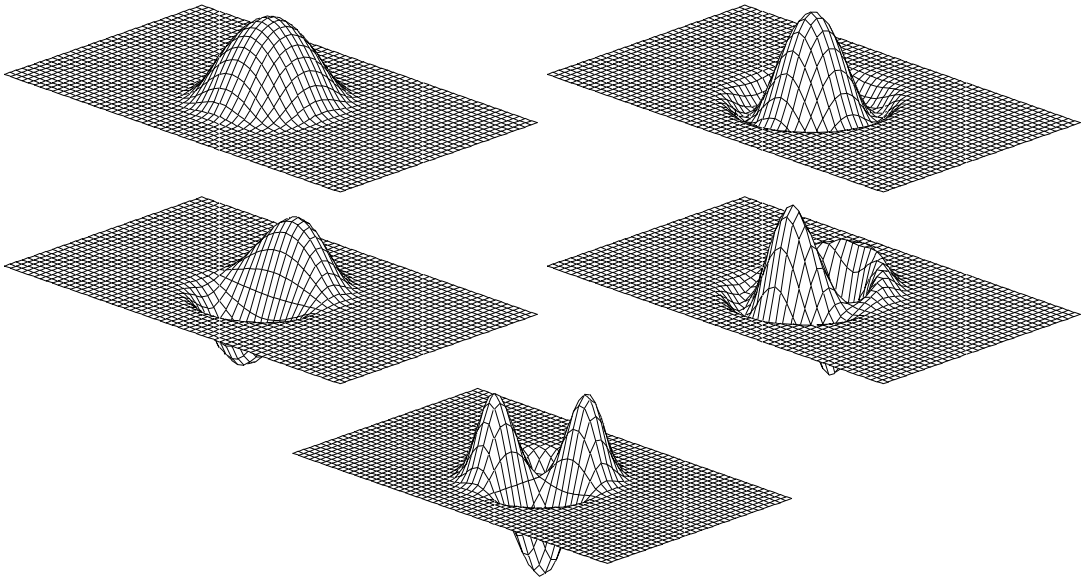


Fig. 2.2: Projector functions of the chlorine atom. Top: two s-type projector functions, middle: p-type, bottom: d-type.

By combining Eq. 2.24 and Eq. 2.26, we can apply $\hat{\mathcal{S}}_R$ to any auxiliary wavefunction.

$$\hat{\mathcal{S}}_R|\tilde{\psi}\rangle = \sum_{i\in R} \hat{\mathcal{S}}_R|\tilde{\phi}_i\rangle\langle\tilde{p}_i|\tilde{\psi}\rangle = \sum_{i\in R}\Big(|\phi_i\rangle - |\tilde{\phi}_i\rangle\Big)\langle\tilde{p}_i|\tilde{\psi}\rangle \ . \tag{2.28}$$

Hence, the transformation operator is

$$\hat{\mathcal{T}} = \hat{1} + \sum_i \Big(|\phi_i\rangle - |\tilde{\phi}_i\rangle\Big)\langle\tilde{p}_i| \,, \tag{2.29}$$

where the sum runs over all partial waves of all atoms. The true wave function can be expressed as

$$|\psi\rangle = |\tilde{\psi}\rangle + \sum_i \Big(|\phi_i\rangle - |\tilde{\phi}_i\rangle\Big)\langle\tilde{p}_i|\tilde{\psi}\rangle = |\tilde{\psi}\rangle + \sum_R \Big(|\psi^1_R\rangle - |\tilde{\psi}^1_R\rangle\Big) \tag{2.30}$$

with

$$|\psi^1_R\rangle = \sum_{i\in R} |\phi_i\rangle\langle\tilde{p}_i|\tilde{\psi}\rangle \tag{2.31}$$

$$|\tilde{\psi}^1_R\rangle = \sum_{i\in R} |\tilde{\phi}_i\rangle\langle\tilde{p}_i|\tilde{\psi}\rangle \,. \tag{2.32}$$

In Fig. 2.3 the decomposition of Eq. 2.30 is shown for the example of the bonding p-σ state of the Cl_2 molecule.

To understand the expression Eq. 2.30 for the true wave function, let us concentrate on different regions in space. (1) Far from the atoms, the partial waves are, according to Eq. 2.25, pairwise identical so that the auxiliary wavefunction is identical to the true wavefunction, that is $\psi(\vec{r}) = \tilde{\psi}(\vec{r})$. (2) Close to an atom R, however, the auxiliary wavefunction is, according to Eq. 2.26, identical to its one-center expansion, that is $\tilde{\psi}(\vec{r}) = \tilde{\psi}^1_R(\vec{r})$. Hence the true wavefunction $\psi(\vec{r})$ is identical to $\psi^1_R(\vec{r})$, which is built up from partial waves that contain the proper nodal structure.

In practice, the partial wave expansions are truncated. Therefore, the identity of Eq. 2.26 does not hold strictly. As a result, the plane waves also contribute to the true wavefunction inside the atomic region. This has the advantage that the missing terms in a truncated partial wave expansion are partly accounted for by plane waves. This explains the rapid convergence of the partial wave expansions. This idea is related to the additive augmentation of the LAPW method of Soler [72].

Frequently, the question comes up, whether the transformation Eq. 2.29 of the auxiliary wavefunctions indeed provides the true wavefunction. The transformation should be considered merely as a change of representation analogous to a coordinate transform. If the total energy functional is transformed consistently, its minimum will yield auxiliary wavefunctions that produce the correct wave functions $|\psi\rangle$.

Expectation values

Expectation values can be obtained either from the reconstructed true wavefunctions or directly from the auxiliary wave functions

$$\begin{aligned}\langle\hat{A}\rangle &= \sum_n f_n\langle\psi_n|\hat{A}|\psi_n\rangle + \sum_{n=1}^{N_c}\langle\phi^c_n|\hat{A}|\phi^c_n\rangle \\ &= \sum_n f_n\langle\tilde{\psi}_n|\hat{\mathcal{T}}^\dagger\hat{A}\hat{\mathcal{T}}|\tilde{\psi}_n\rangle + \sum_{n=1}^{N_c}\langle\phi^c_n|\hat{A}|\phi^c_n\rangle \,,\end{aligned} \tag{2.33}$$

where f_n are the occupations of the valence states and N_c is the number of core states. The first sum runs over the valence states, and second over the core states $|\phi^c_n\rangle$.

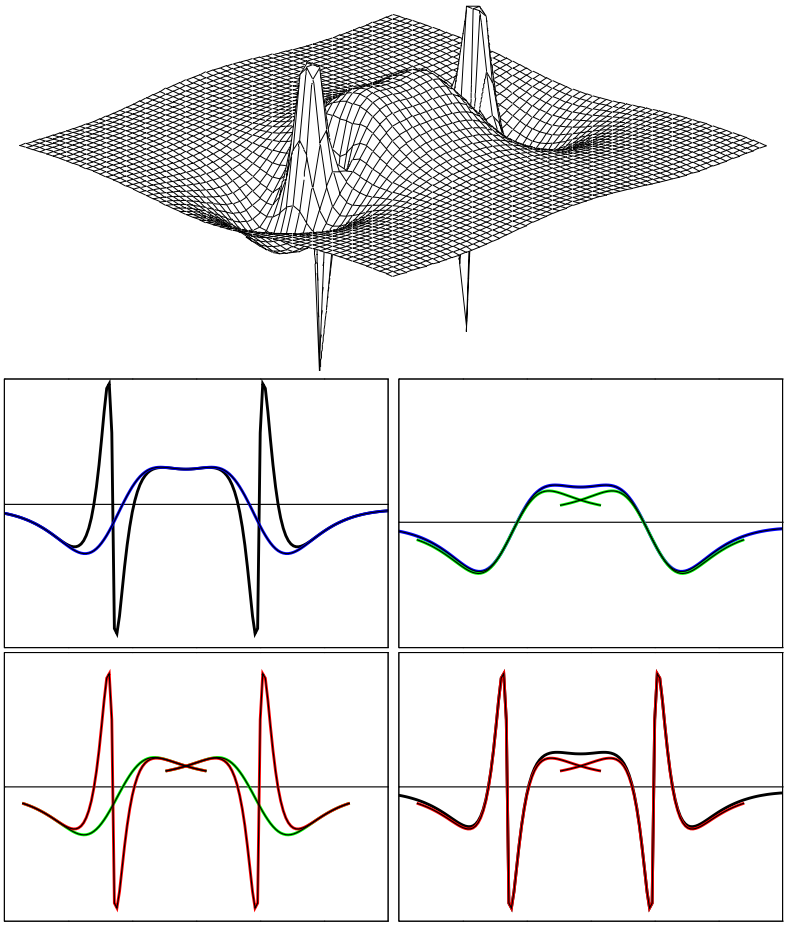


Fig. 2.3: Bonding p-σ orbital of the Cl_2 molecule and its decomposition into auxiliary wavefunction and the two one-center expansions. Top-left: True and auxiliary wave function; top-right: auxiliary wavefunction and its partial wave expansion; bottom-left: the two partial wave expansions; bottom-right: true wavefunction and its partial wave expansion.

Now we can decompose the matrix element for a wavefunction ψ into its individual contributions according to Eq. 2.30.

$$\begin{aligned}
\langle\psi|\hat{A}|\psi\rangle &= \langle\tilde{\psi} + \sum_R(\psi^1_R - \tilde{\psi}^1_R)|\hat{A}|\tilde{\psi} + \sum_{R'}(\psi^1_{R'} - \tilde{\psi}^1_{R'})\rangle \\
&= \underbrace{\langle\tilde{\psi}|\hat{A}|\tilde{\psi}\rangle + \sum_R\Big(\langle\psi^1_R|\hat{A}|\psi^1_R\rangle - \langle\tilde{\psi}^1_R|\hat{A}|\tilde{\psi}^1_R\rangle\Big)}_{\text{part 1}} \\
&+ \underbrace{\sum_R\Big(\langle\psi^1_R - \tilde{\psi}^1_R|\hat{A}|\tilde{\psi} - \tilde{\psi}^1_R\rangle + \langle\tilde{\psi} - \tilde{\psi}^1_R|\hat{A}|\psi^1_R - \tilde{\psi}^1_R\rangle\Big)}_{\text{part 2}} \\
&+ \underbrace{\sum_{R\neq R'}\langle\psi^1_R - \tilde{\psi}^1_R|\hat{A}|\psi^1_{R'} - \tilde{\psi}^1_{R'}\rangle}_{\text{part 3}}
\end{aligned} \tag{2.34}$$

Only the first part of Eq. 2.34 is evaluated explicitly, while the second and third parts of Eq. 2.34 are neglected, because they vanish for sufficiently local operators as long as the partial wave expansion is

converged: The function $\psi^1_R - \tilde{\psi}^1_R$ vanishes per construction beyond its augmentation region, because the partial waves are pairwise identical beyond that region. The function $\tilde{\psi} - \tilde{\psi}^1_R$ vanishes inside its augmentation region, if the partial wave expansion is sufficiently converged. In no region of space are both functions $\psi^1_R - \tilde{\psi}^1_R$ and $\tilde{\psi} - \tilde{\psi}^1_R$ simultaneously nonzero. Similarly the functions $\psi^1_R - \tilde{\psi}^1_R$ from different sites are never non-zero in the same region in space. Hence, the second and third parts of Eq. 2.34 vanish for operators such as the kinetic energy $\frac{-\hbar^2}{2m_e}\vec{\nabla}^2$ and the real space projection operator $|r\rangle\langle r|$, which produces the electron density. For truly nonlocal operators the parts 2 and 3 of Eq. 2.34 would have to be considered explicitly.

The expression, Eq. 2.33, for the expectation value can therefore be written with the help of Eq. 2.34 as

$$\begin{aligned}
\langle\hat{A}\rangle &= \sum_n f_n\Big(\langle\tilde{\psi}_n|\hat{A}|\tilde{\psi}_n\rangle + \langle\psi^1_n|\hat{A}|\psi^1_n\rangle - \langle\tilde{\psi}^1_n|\hat{A}|\tilde{\psi}^1_n\rangle\Big) + \sum_{n=1}^{N_c}\langle\phi^c_n|\hat{A}|\phi^c_n\rangle \\
&= \sum_n f_n\langle\tilde{\psi}_n|\hat{A}|\tilde{\psi}_n\rangle + \sum_{n=1}^{N_c}\langle\tilde{\phi}^c_n|\hat{A}|\tilde{\phi}^c_n\rangle \\
&+ \sum_R\Big(\sum_{i,j\in R} D_{i,j}\langle\phi_j|\hat{A}|\phi_i\rangle + \sum_{n\in R}^{N_{c,R}}\langle\phi^c_n|\hat{A}|\phi^c_n\rangle\Big) \\
&- \sum_R\Big(\sum_{i,j\in R} D_{i,j}\langle\tilde{\phi}_j|\hat{A}|\tilde{\phi}_i\rangle + \sum_{n\in R}^{N_{c,R}}\langle\tilde{\phi}^c_n|\hat{A}|\tilde{\phi}^c_n\rangle\Big)\,, \qquad (2.35)
\end{aligned}$$

where $\boldsymbol{D}$ is the one-center density matrix defined as

$$D_{i,j} = \sum_n f_n\langle\tilde{\psi}_n|\tilde{p}_j\rangle\langle\tilde{p}_i|\tilde{\psi}_n\rangle = \sum_n\langle\tilde{p}_i|\tilde{\psi}_n\rangle f_n\langle\tilde{\psi}_n|\tilde{p}_j\rangle\,. \qquad (2.36)$$

The auxiliary core states, $|\tilde{\phi}^c_n\rangle$ allow us to incorporate the tails of the core wavefunction into the plane-wave part, and therefore assure that the integrations of partial wave contributions cancel exactly beyond r_c. They are identical to the true core states in the tails, but are a smooth continuation inside the atomic sphere. It is not required that the auxiliary wave functions are normalized.

Following this scheme, the electron density is given by

$$n(\vec{r}) = \tilde{n}(\vec{r}) + \sum_R\Big(n^1_R(\vec{r}) - \tilde{n}^1_R(\vec{r})\Big) \qquad (2.37)$$

$$\begin{aligned}
\tilde{n}(\vec{r}) &= \sum_n f_n\tilde{\psi}^*_n(\vec{r})\tilde{\psi}_n(\vec{r}) + \tilde{n}_c(\vec{r}) \\
n^1_R(\vec{r}) &= \sum_{i,j\in R} D_{i,j}\phi^*_j(\vec{r})\phi_i(\vec{r}) + n_{c,R}(\vec{r}) \\
\tilde{n}^1_R(\vec{r}) &= \sum_{i,j\in R} D_{i,j}\tilde{\phi}^*_j(\vec{r})\tilde{\phi}_i(\vec{r}) + \tilde{n}_{c,R}(\vec{r})\,, \qquad (2.38)
\end{aligned}$$

where $n_{c,R}$ is the core density of the corresponding atom and $\tilde{n}_{c,R}$ is the auxiliary core density, which is identical to $n_{c,R}$ outside the atomic region, but smooth inside.

Before we continue, let us discuss a special point: The matrix elements of a general operator with the auxiliary wavefunctions may be slowly converging with the plane-wave expansion, because the operator $\hat{A}$ may not be well behaved. An example of such an operator is the singular electrostatic potential of a nucleus. This problem can be alleviated by adding an "intelligent zero": If an operator $\hat{B}$ is purely localized within an atomic region, we can use the identity between the auxiliary wavefunction and its own partial wave expansion

$$0 = \langle\tilde{\psi}_n|\hat{B}|\tilde{\psi}_n\rangle - \langle\tilde{\psi}^1_n|\hat{B}|\tilde{\psi}^1_n\rangle\,. \qquad (2.39)$$

Now we choose an operator $\hat{B}$ so that it cancels the problematic behavior of the operator $\hat{A}$, but is localized in a single atomic region. By adding $\hat{B}$ to the plane-wave part and the matrix elements with its one-center expansions, the plane-wave convergence can be improved without affecting the converged result. A term of this type, namely $\hat{\bar{v}}$ will be introduced in the next section to cancel the Coulomb singularity of the potential at the nucleus.

Total energy

Like wavefunctions and expectation values, also the total energy can be divided into three parts.

$$E\Big[\{|\tilde{\psi}_n\rangle\},\{R_R\}\Big] = \tilde{E} + \sum_R \Big(E^1_R - \tilde{E}^1_R\Big) \tag{2.40}$$

The plane-wave part $\tilde{E}$ involves only smooth functions and is evaluated on equi-spaced grids in real and reciprocal space. This part is computationally most demanding, and is similar to the expressions in the pseudopotential approach.

$$\begin{aligned}\tilde{E} = &\sum_n \langle\tilde{\psi}_n|\frac{\hat{\tilde{p}}^2}{2m_e}|\tilde{\psi}_n\rangle + \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2\Big(\tilde{n}(\vec{r})+\tilde{Z}(\vec{r})\Big)\Big(\tilde{n}(\vec{r}')+\tilde{Z}(\vec{r}')\Big)}{4\pi\epsilon_0|\vec{r}-\vec{r}'|} \\ &+ \int d^3r \; \bar{v}(\vec{r})\tilde{n}(\vec{r}) + E_{xc}[\tilde{n}]\end{aligned} \tag{2.41}$$

$\tilde{Z}(\mathbf{r})$ is an angular-momentum dependent core-like density that will be described in detail below. The remaining parts can be evaluated on radial grids in a spherical-harmonics expansion. The nodal structure of the wavefunctions can be properly described on a logarithmic radial grid that becomes very fine near the nucleus,

$$\begin{aligned}E^1_R = &\sum_{i,j\in R} D_{i,j}\langle\phi_j|\frac{\hat{\tilde{p}}^2}{2m_e}|\phi_i\rangle + \sum_{n\in R}^{N_{c,R}}\langle\phi^c_n|\frac{\hat{\tilde{p}}^2}{2m_e}|\phi^c_n\rangle \\ &+ \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2\Big(n^1(\vec{r})+Z(\vec{r})\Big)\Big(n^1(\vec{r}')+Z(\vec{r}')\Big)}{|\vec{r}-\vec{r}'|} + E_{xc}[n^1]\end{aligned} \tag{2.42}$$

$$\begin{aligned}\tilde{E}^1_R = &\sum_{i,j\in R} D_{i,j}\langle\tilde{\phi}_j|\frac{\hat{\tilde{p}}^2}{2m_e}|\tilde{\phi}_i\rangle + \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2\Big(\tilde{n}^1(\vec{r})+\tilde{Z}(\vec{r})\Big)\Big(\tilde{n}^1(\vec{r}')+\tilde{Z}(\vec{r}')\Big)}{4\pi\epsilon_0|\vec{r}-\vec{r}'|} \\ &+ \int d^3r \; \bar{v}(\vec{r})\tilde{n}^1(\vec{r}) + E_{xc}[\tilde{n}^1]\,.\end{aligned} \tag{2.43}$$

The compensation charge density $\tilde{Z}(\vec{r}) = \sum_R \tilde{Z}_R(\vec{r})$ is given as a sum of angular momentum dependent Gauss functions, which have an analytical plane-wave expansion. A similar term occurs also in the pseudopotential approach. In contrast to the norm-conserving pseudopotential approach, however, the compensation charge of an atom $\tilde{Z}_R$ is non-spherical and constantly adapts instantaneously to the environment. It is constructed such that

$$n^1_R(\vec{r}) + Z_R(\vec{r}) - \tilde{n}^1_R(\vec{r}) - \tilde{Z}_R(\vec{r}) \tag{2.44}$$

has vanishing electrostatic multipole moments for each atomic site. With this choice, the electrostatic potentials of the augmentation densities vanish outside their spheres. This is the reason why there is no electrostatic interaction of the one-center parts between different sites.

The compensation charge density as given here is still localized within the atomic regions. A technique similar to an Ewald summation, however, allows it to be replaced by a very extended charge density. Thus we can achieve, that the plane-wave convergence of the total energy is not affected by the auxiliary density.

The potential $\bar{v} = \sum_R \bar{v}_R$, which occurs in Eqs. 2.41 and 2.43 enters the total energy in the form of "intelligent zeros" described in Eq. 2.39

$$0 = \sum_n f_n \left(\langle \tilde{\psi}_n | \bar{v}_R | \tilde{\psi}_n \rangle - \langle \tilde{\psi}_n^1 | \bar{v}_R | \tilde{\psi}_n^1 \rangle \right) = \sum_n f_n \langle \tilde{\psi}_n | \bar{v}_R | \tilde{\psi}_n \rangle - \sum_{i,j \in R} D_{i,j} \langle \tilde{\phi}_i | \bar{v}_R | \tilde{\phi}_j \rangle \,. \tag{2.45}$$

The main reason for introducing this potential is to cancel the Coulomb singularity of the potential in the plane-wave part. The potential $\bar{v}$ allows us to influence the plane-wave convergence beneficially, without changing the converged result. $\bar{v}$ must be localized within the augmentation region, where Eq. 2.26 holds.

Approximations

Once the total energy functional provided in the previous section has been defined, everything else follows: Forces are partial derivatives with respect to atomic positions. The potential is the derivative of the non-kinetic energy contributions to the total energy with respect to the density, and the auxiliary Hamiltonian follows from derivatives $\tilde{H}|\tilde{\psi}_n\rangle$ with respect to auxiliary wave functions. The fictitious Lagrangian approach of Car and Parrinello [100] does not allow any freedom in the way these derivatives are obtained. Anything else than analytic derivatives will violate energy conservation in a dynamical simulation. Since the expressions are straightforward, even though rather involved, we will not discuss them here.

All approximations are incorporated already in the total energy functional of the PAW method. What are those approximations?

- Firstly we use the frozen-core approximation. In principle this approximation can be overcome.
- The plane-wave expansion for the auxiliary wavefunctions must be complete. The plane-wave expansion is controlled easily by increasing the plane-wave cutoff defined as $E_{PW} = \frac{1}{2}\hbar^2 G_{max}^2$. Typically we use a plane-wave cutoff of 30 Ry.
- The partial wave expansions must be converged. Typically we use one or two partial waves per angular momentum (ℓ, m) and site. It should be noted that the partial wave expansion is not variational, because it changes the total energy functional and not the basis set for the auxiliary wavefunctions.

We do not discuss here numerical approximations such as the choice of the radial grid, since those are easily controlled.

Relation to pseudopotentials

We mentioned earlier that the pseudopotential approach can be derived as a well defined approximation from the PAW method: The augmentation part of the total energy $\Delta E = E^1 - \tilde{E}^1$ for one atom is a functional of the one-center density matrix $\boldsymbol{D}$ defined in Eq. 2.36. The pseudopotential approach can be recovered if we truncate a Taylor expansion of ΔE about the atomic density matrix after the linear term. The term linear in $\boldsymbol{D}$ is the energy related to the nonlocal pseudopotential.

$$\begin{aligned} \Delta E(\boldsymbol{D}) &= \Delta E(\boldsymbol{D}^{at}) + \sum_{i,j} \left. \frac{\partial \Delta E}{\partial D_{i,j}} \right|_{\boldsymbol{D}^{at}} (D_{i,j} - D_{i,j}^{at}) + O(\boldsymbol{D} - \boldsymbol{D}^{at})^2 \\ &= E_{self} + \sum_n f_n \langle \tilde{\psi}_n | \hat{v}^{ps} | \tilde{\psi}_n \rangle - \int d^3r \; \bar{v}(\vec{r}) \tilde{n}(\vec{r}) + O(\boldsymbol{D} - \boldsymbol{D})^2 \,, \end{aligned} \tag{2.46}$$

which can directly be compared with the total energy expression Eq. 2.17 of the pseudopotential method. The local potential $\bar{v}(\vec{r})$ of the pseudopotential approach is identical to the corresponding potential of the projector augmented-wave method. The remaining contributions in the PAW total

energy, namely $\tilde{E}$, differ from the corresponding terms in Eq. 2.17 only in two features: our auxiliary density also contains an auxiliary core density, reflecting the nonlinear core correction of the pseudopotential approach, and the compensation density $\tilde{Z}(\vec{r})$ is non-spherical and depends on the wave function. Thus we can look at the PAW method also as a pseudopotential method with a pseudopotential that adapts instantaneously to the electronic environment. In the PAW method, the explicit nonlinear dependence of the total energy on the one-center density matrix is properly taken into account.

What are the main advantages of the PAW method compared with the pseudopotential approach?

Firstly all errors can be systematically controlled, so that there are no transferability errors. As shown by Watson [105] and Kresse [106], most pseudopotentials fail for high spin atoms such as Cr. While it is probably true that pseudopotentials can be constructed that cope even with this situation, a failure can not be known beforehand, so that some empiricism remains in practice: A pseudopotential constructed from an isolated atom is not guaranteed to be accurate for a molecule. In contrast, the converged results of the PAW method do not depend on a reference system such as an isolated atom, because PAW uses the full density and potential.

Like other all-electron methods, the PAW method provides access to the full charge and spin density, which is relevant, for example, for hyperfine parameters. Hyperfine parameters are sensitive probes of the electron density near the nucleus. In many situations they are the only information available that allows us to deduce atomic structure and chemical environment of an atom from experiment.

The plane-wave convergence is more rapid than in norm-conserving pseudopotentials and should in principle be equivalent to that of ultra-soft pseudopotentials [95]. Compared to the ultra-soft pseudopotentials, however, the PAW method has the advantage that the total energy expression is less complex and can therefore be expected to be more efficient.

The construction of pseudopotentials requires us to determine a number of parameters. As they influence the results, their choice is critical. Also the PAW methods provides some flexibility in the choice of auxiliary partial waves. However, this choice does not influence the converged results.

Recent developments

Since the first implementation of the PAW method in the CP-PAW code [65], a number of groups have adopted the PAW method. The second implementation, called PWPAW, was done by the group of Holzwarth[107, 108, 109, 110]. Several codes, previously using pseudopotentials have extended their code to PAW. Among them are the VASP code with the PAW implementation of Kresse and Joubert[106]. The PAW implementation of the ABINIT code[111] has been done by Torrent et al.[112]. An independent PAW code has been developed by Valiev and Weare [113]. This implementation has entereed the NWChem code [114, 115]. The PAW method has also been implemented by W. Kromen [116] into the EStCoMPP code of Blügel and Schröder. Other implementations are in the Quantum Espresso code [117][2] and Socorro[3]. A real-space-grid based version of the PAW method is the code GPAW developed by Mortensen et al. [118].

Another branch of methods uses the reconstruction of the PAW method, without taking into account the full wavefunctions in the energy minimization. Following chemists' notation, this approach could be termed "post-pseudopotential PAW". This development began with the evaluation for hyperfine parameters from a pseudopotential calculation using the PAW reconstruction operator [98] and is now used in the pseudopotential approach to calculate properties that require the correct wavefunctions such as hyperfine parameters.

The implementation of the PAW method by Kresse and Joubert [106] has been particularly useful as they had an implementation of PAW in the same code as the ultra-soft pseudopotentials, so that they could critically compare the two approaches. Their conclusion is that both methods compare well in most cases, but they found that magnetic energies are seriously – by a factor two – in error

[2] Quantum Espresso is maintained by Stefano Gironcoli and Lorenzo Paulatto

[3] `http://dft.sandia.gov/socorro`. Socorro is maintained by Alan Wright and Normand Modine

in the pseudopotential approach, while the results of the PAW method were in line with other all-electron calculations using the linear augmented plane-wave method. As an aside, Kresse and Joubert claim incorrectly that their implementation is superior as it includes a term that is analogous to the non-linear core correction of pseudopotentials [93]: this term, however, is already included in the original version in the form of the pseudized core density. Recently, a careful comparison[119] has shown that the original formulation[65] is more reliable.

Several extensions of the PAW have been done in the recent years: For applications in chemistry truly isolated systems are often of great interest. As any plane-wave based method introduces periodic images, the electrostatic interaction between these images can cause serious errors. The problem has been solved by mapping the charge density onto a point charge model, so that the electrostatic interaction could be subtracted out in a self-consistent manner [120]. In order to include the influence of the environment, the latter was simulated by simpler force fields using the quantum-mechanics molecular-mechanics (QM-MM) approach [121].

In order to overcome the limitations of the density-functional theory several extensions have been performed. Bengone [122] implemented the LDA+U approach into our CP-PAW code. Soon after this, Arnaud [123] accomplished the implementation of the GW approximation into our CP-PAW code. The VASP-version of PAW [124] and our CP-PAW code have now been extended to include a non-collinear description of the magnetic moments. In a non-collinear description, the Schrödinger equation is replaced by the Pauli equation with two-component spinor wavefunctions.

The PAW method has proven useful to evaluate electric field gradients [125] and magnetic hyperfine parameters with high accuracy [126]. Invaluable will be the prediction of NMR chemical shifts using the GIPAW method of Pickard and Mauri [127], which is based on their earlier work [128]. While the GIPAW is implemented in a post-pseudopotential manner, the extension to a self-consistent PAW calculation should be straightforward. An post-pseudopotential approach has also been used to evaluate core level spectra [129] and momentum matrix elements [130].

Chapter 3

Ab-initio molecular dynamics

3.1 Fictitious Lagrangian approach to ab-initio molecular dynamics

Until 1985, electronic structure calculations were performed only for given atomic structures. Those could be improved in a tedious manner using the calculated forces. Already those single-point calculations were at that time among the largest computer calculations done. One of the reasons was the self-consistency loop. It was nearly inconceivable to study the dynamics of a system.

In 1985 Roberto Car and Michele Parrinello[100] from (Scuola internazionale Superiore di Studi Avanzati) SISSA in Trieste, changed this situation in a radical manner. By introducing ideas from classical molecular dynamics into the field of electronic structure calculations they showed that calculations with moving classical nuclei are feasible.

The equations are derived as Euler Lagrange equations from the Lagrangian

$$\mathcal{L} = \sum_n f_n \langle \dot{\psi}_n | m_\psi | \dot{\psi}_n \rangle + \frac{1}{2} \sum_i M_i \dot{\vec{R}}_i^2 + E_{DFT}[\psi_n, \vec{R}_i] - \sum_{n,m} \Lambda_{n,m} \left(\langle \psi_n | \psi_m \rangle - \delta_{n,m} \right) \tag{3.1}$$

- The first term is the fictitious kinetic energy of the wave functions. There is no physical correspondence for this term. Ideally one would choose the "fictitious mass" or wave function mass equal to zero. Of course this is not feasible in practice, but the approximation can be checked by a convergence test with decreasing wave function mass. The presence of this unphysical quantity also resulted in naming the Lagrangian fictitious.
- The second term describes the classical kinetic energy of the nuclei.
- the third term is the density-functional total energy, which is a functional of the wave functions and the atomic positions.
- The last term is the constraint of orthonormal wave functions $|\psi_n\rangle$. The matrix $\mathbf{\Lambda}$ are the corresponding Lagrange multipliers.

The Euler-Lagrange equations are

$$\begin{aligned} m_\psi |\ddot{\psi}_n\rangle f_n &= -\underbrace{\frac{\delta E}{\delta \langle \psi_n |}}_{\hat{H}|\psi_n\rangle f_n} + \sum_m |\psi_m\rangle \Lambda_{m,n} \\ M_i \ddot{\vec{R}}_i &= \underbrace{-\vec{\nabla}_{\vec{R}_i} E}_{\vec{F}_i} \end{aligned} \tag{3.2}$$

3.1.1 Adiabatic principle

During the simulation the atoms have a finite, physical temperature, while the wave functions should ideally be in the ground state, that is at $T = 0$. Thus we need to perform a nonequilibrium simulation with a continuous heat transfer from the atoms to the wave functions. Fortunately, the heat transfer is relatively small. The reason for this is the **adiabatic principle**, which says that that the frequencies of the atomic motion are much lower than those of the wave function dynamics. The two types of dynamics are therefore out of resonance and the heat transfer is small.

This adiabatic principle can be illustrated on a simple system of two coupled pendula as shown in Fig. 3.1

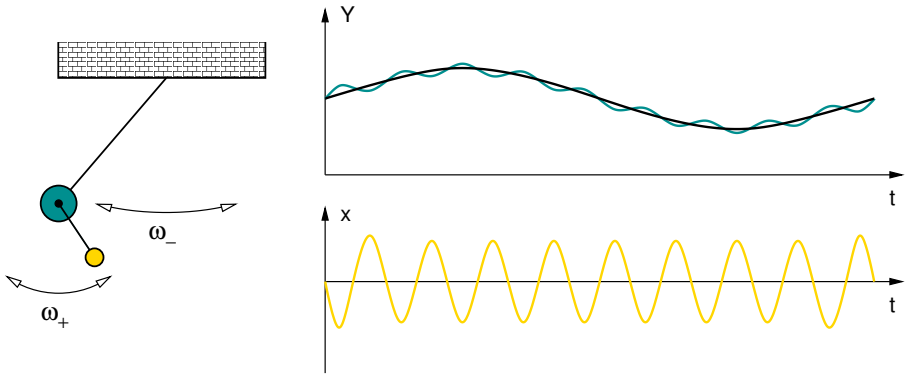


Fig. 3.1:

The bigger green ball represents the nuclei, having a rather large mass and low eigenfrequency. The smaller yellow ball represents the wave functions with a small mass and high eigenfrequency.

While the heat transfer is small it does not vanish. In the early days the method was to interrupt the calculation from time to time, optimize the wave functions, and continue the calculation. A better alternative to this technique was to add a small friction to the wave function dynamics.

Today we use two thermostats[103, 104]. One is a real thermostat acting on the atomic motion, which ensures that the atomic trajectories form a canonical ensemble. This is the Nose thermostat described below. The other thermostat acts on the wave functions. While being formed in close analogy to the Nose thermostat, this is actually not a thermostat: It does nothing until the wave functions acquire more kinetic energy than required to follow the atomic motion. Only then, it will shuffle energy from the wave function motion back into the atomic motion.

3.1.2 Mass renormalization

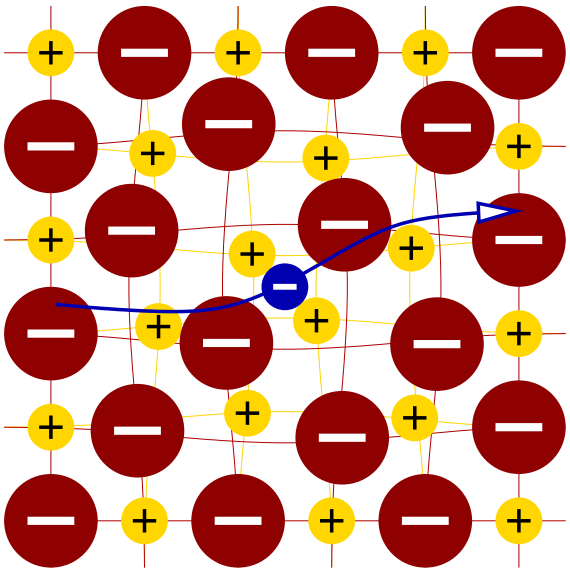
The fictitious kinetic energy of the wave functions has another important effect: The atoms become heavier. This is important to consider when extracting vibrational frequencies from a dynamical trajectory. Without renormalizing the nuclear masses, frequencies from an ab-initio molecular dynamics simulation will be too low.

Physically we can consider the atom as a **quasiparticle** made out of a nucleus and a deformation of the electron gas. This sounds a little mystic: Therefore let us begin with the prototype of a quasi-particle, namely the **polaron**.

Consider an electron, which travels through an ionic solid such as NaCl. As the electron is placed into the crystal the crystal is distorted because the negative ions are repelled and the positive ions are attracted to the electron. A lattice distortion is a **phonon** (Lattice vibration). Thus the electron forms a bound state with a phonon. This is quite analogous to a chemical bond only that it is not two atoms that bind, but an electron and an elementary excitation of the crystal, the phonon. Both

the electron and the phonon can travel, and when they bind, they travel together.

It is not surprising that the electron that carries along a phonon, will travel differently from one that is on its own. There is still translational symmetry, so there is no additional friction that will slow down the polaron. But it will cost more energy to accelerate the electron, because momentum needs to be supplied not only to the electron, but also to the phonon. Thus the polaron is heavier than the bare electron. It will have an effective mass that is larger than that of the bare mass o the electron.



We can also understand the increased mass from the dispersion relation. An electron in the conduction band will have the dispersion relation of a free electron (even though the mass may also be affected by placing the electron into the static crystal). An optical phonon has a low energy and a fairly flat dispersion relation. If the phonon and the electron do not interact the two dispersion relations cross. Due to the interaction, the crossing is changed into an **avoided crossing**. This will push the lower branch downward, making it flatter. Because the effective mass is the curvature of the dispersion relation, this implies a larger mass.

We have seen that particles and elementary excitations can form quasi-particle, and this affects the effective mass of the quasi-particle.

Now we consider a new type of quasi-particle, namely a nucleus, that flies through a homogeneous electron gas. Again it forms between between a particle, the nucleus and an elementary excitation of the electron gas, a charge-density wave. We call the quasi-particle an atom. In the Car Parrinello method it is not exactly the electrons, but the wave functions that form the quasi particle with the nucleus.

Because the wave functions have a fictitious kinetic energy, they have a dispersion relation and thus they have a mass. This mass must be carried along by the atom. The atom is heavier than the nucleus alone. Because the effect is fictitious, we must correct it. If the masses are too heavy, the vibrational modes will be too low.

We are able to estimate the effect considering the dispersion relation of an isolated atom, and thus determine the mass of the wave function cloud following the nucleus[104, 103]. Now we renormalize the mass by simply making the nucleus artificially lighter, so that the effective mass equals the true atomic mass.

3.1.3 Fictitious Lagrangian vs. Born-Oppenheimer dynamics

In the CP-PAW code we use the fictitious Lagrangian formalism. Most other codes use what is called Born-Oppenheimer dynamics. Both methods have their own advantages and disadvantages, and the decision in favor of one or the other is not clear cut.

The basic principle of the Born-Oppenheimer dynamics is to converge the wave functions into the ground state for each molecular dynamics step for the atoms.

3.2 Equations of motion: The Verlet Algorithm

The Verlet algorithm has been popularized by the french physicist Loup Verlet.[131] It has been used earlier by Carl Störmer.

Consider the following second order differential equation.

$$m\ddot{\vec{x}} = \vec{F}(t) - m\alpha\dot{\vec{x}} \tag{3.3}$$

In order to solve it on a computer, we discretize the time axis. The individual time slices are at $t_j = j\Delta$. Δ is called the **time step**. It turns out that at any given time we only need to consider three successive time slices. Therefore, we introduce the following short-hand notation: relative to a given time slice j_0, we denote

$$\begin{aligned}\vec{x}(+) &= \vec{x}(t_{j_0} + \Delta)\\ \vec{x}(0) &= \vec{x}(t_{j_0})\\ \vec{x}(-) &= \vec{x}(t_{j_0} - \Delta)\end{aligned}$$

The recipe for the Verlet algorithm is to replace first and second time derivatives by differential quotients in the following way:

REPLACEMENT RULE OF THE VERLET ALGORITHM

$$\begin{aligned}\dot{\vec{x}} &\to \frac{\vec{x}(+) - \vec{x}(-)}{2\Delta} + O(\Delta^2)\\ \ddot{\vec{x}} &\to \frac{\vec{x}(+) - 2\vec{x}(t) + \vec{x}(-)}{\Delta^2} + O(\Delta^2)\end{aligned} \tag{3.4}$$

This result can be verified using the Taylor expansion of $x(t \pm \Delta)$ in Δ. It is important to exactly use the differential quotients given above and not to replace the first derivative with the differential quotient between two neighboring time slices.

Inserting the replacement rules into the equation of motion Eq. 3.3, we obtain

$$\begin{aligned}m\frac{\vec{x}(+) - 2\vec{x}(t) + \vec{x}(-)}{\Delta^2} &= F(0) - m\alpha\frac{\vec{x}(+) - \vec{x}(-)}{2\Delta}\\ \vec{x}(+) - 2\vec{x}(0) + \vec{x}(-) &= \vec{F}(0)\frac{\Delta^2}{m} - \underbrace{\frac{\alpha\Delta}{2}}_{:=a}\Big(\frac{\vec{x}(+) - \vec{x}(-)}{2\Delta}\Big)\\ x(+) &= \vec{x}(0)\frac{2}{1+a} - \vec{x}(-)\frac{1-a}{1+a} + \vec{F}(0)\frac{\Delta^2}{m}\frac{1}{1+a}\end{aligned}$$

VERLET ALGORITHM

$$\vec{x}(+) = \vec{x}(0)\frac{2}{1+a} - \vec{x}(-)\frac{1-a}{1+a} + \vec{F}(0)\frac{\Delta^2}{m}\frac{1}{1+a} \tag{3.5}$$

The parameter Δ is the time step and the parameter

$$a \stackrel{\text{def}}{=} \frac{\alpha\Delta}{2} \tag{3.6}$$

is the friction parameter. Note, that $m\alpha$ is what is called the friction coefficient and not a. In order to obtain a feeling for the parameter a, let us discuss its extreme values

- If we choose the friction parameter a equal to zero, we obtain an energy conserving dynamics without friction.
- If we set the friction parameter a equal to one, the resulting trajectory corresponds to infinite friction. This can be seen as follows: The discretized equation of motion with $a = 1$ has the form

$$\vec{x}(+) = \vec{x}(0) + \vec{F}(0)\frac{\Delta^2}{2m}$$
$$\dot{x} \approx \frac{\vec{x}(+) - \vec{x}(0)}{\Delta'} = \vec{F}(0)$$

  where the new time step has is $\Delta' = \frac{\Delta^2}{2m}$. This equation is **steepest descent**, that is one follows the forces downhill. One can call it an equation with infinite friction, because the inertia, respectively the mass, drops out of the equation for $a = 1$. In contrast to infinite friction, however, this does not imply that the motion comes to halt.

3.2.1 Advantages and disadvantages of the Verlet algorithm

The Verlet algorithm is probably the most simple method for discretizing the equations of motion. There are more sophisticated integrators. Typically they are more accurate for rather small time steps. Simple integrators such as the Verlet algorithm have the advantage that they are more robust and that they provide reasonable results also when the time step is fairly large.

The Verlet algorithm has one major advantage over most other techniques: It is absolutely invariant against time inversion. This implies that there is absolutely no energy drift, unless there is an explicit friction.

The Verlet algorithm can be derived from a discretized action with Hamilton's principle.

3.2.2 Stability of the Verlet algorithm

The time step in the Verlet algorithm has a stability limit. This means that if we choose the time step too large, the simulation will be unstable. The stability limit is given by

STABILITY LIMIT OF THE VERLET ALGORITHM

$$\Delta < T_0/\pi = 2/\omega_0 \tag{3.7}$$

where ω_0 is the maximum circular frequency of the modes in the system and T_0 is the corresponding

period. One needs to understand the frequency spectrum of the system in order to set the time step in a valid an economical manner. The instability occurs fairly immediate, so that a breakdown is observed soon by a kinetic energy that diverges exponentially.

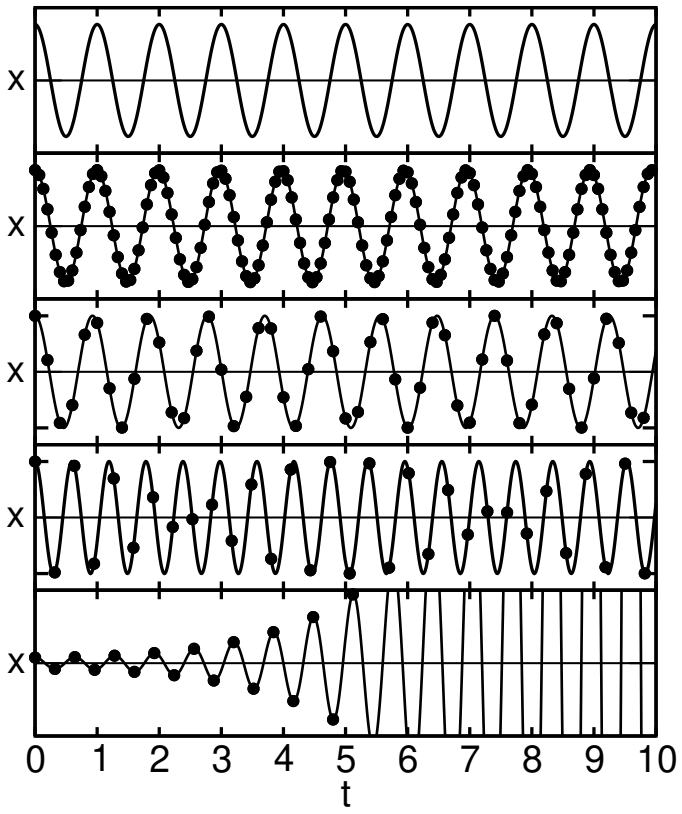


Fig. 3.2: Trajectory, i.e. displacement versus time, of a 1-dimensional harmonic oscillator for different choices of the time step Δ. The time is measured in terms of periods T_0 of the harmonic oscillator. Lines are the interpolated trajectories, while dots indicate the points on the discretized trajectory. From top to bottom: (1) analytical result (2) $\Delta = T_0/15$, where frequencies are accurate to within 1 %, (3) $\Delta = T_0/5$ close to the limit where an interpolation provides a reasonable trace of the trajectory, (4) $\Delta = 0.99 \cdot T_0/\pi$ just below the stability limit and (5) $\Delta = 1.01 \cdot T_0/\pi$, just beyond the stability limit. The stability limit is at T_0/π.

3.2.3 Accuracy of the Verlet algorithm

Accuracy is different from stability. Even if the system is stable, the frequencies are still affected by the choice of the time step. One can estimate that the frequencies of the system are accurate to about 1 % if the time step is smaller than

$$\Delta < T/15 \, . \tag{3.8}$$

This is seen in Fig. 3.3

3.2.4 Convergence

While the Verlet algorithm has been developed to calculate trajectories, we use it often in combination with a friction to bring the system into the ground state, i.e. the minimum of the potential energy

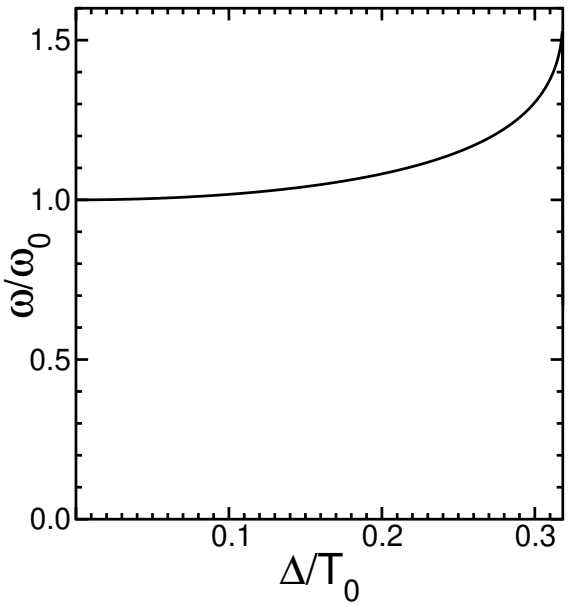


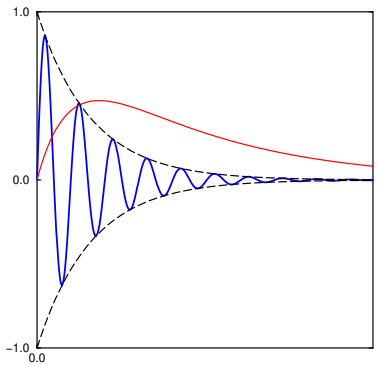
Fig. 3.3: Numerical frequency ω as function of time step Δ as obtained in Eq. 3.19. The correct analytical circular frequency is ω_0. At $\Delta = 2/\omega_0$, that is at the right boundary of the graph, is the stability limit of the Verlet algorithm. Frequencies tend to be overestimated by the Verlet algorithm. The frequencies are within 1 % of the correct value, if the time step is less one-tenth of the oscillation period.

surface.

In order to reach the ground state efficiently, we need to minimize the number of time step one needs to approach the minimum to within a certain distance. For that purpose, it is again useful to analyze the trajectories. As our model system we use again the harmonic oscillator.

A damped harmonic oscillator has three different forms of trajectories

- damped oscillation or weakly damped oscillator
- critical damping or aperiodic limit
- overdamped trajectory or strongly damped oscillator



During optimization, a common reaction of a beginner is to increase the friction as much as possible in order to obtain fastest possible convergence. This is, however, a very poor choice. Let us analyze this pitfall by looking at the energy dissipation $\frac{dE_{tot}}{dt}$. Convergence will be fastest, if the dissipation is at its maximum. The energy dissipation is *friction-times-velocity*. If the friction is very large, the velocity will become very small, so that little energy will be dissipated as a consequence. If, as the opposite extreme, a small friction parameter is chosen, the velocity will remain large but, due the small friction, the dissipation will be small again. The optimum lies, as usual, in the middle.

For each of the three types of trajectories of a damped oscillator, , namely the weakly damped, the aperiodic and the overdamped case, we can determine an exponential decay constant λ of the envelope function, so that

$$|x(t)| < Ae^{-\lambda t} \qquad \text{or} \qquad E_{tot}(t) \approx E_0 + cA^2 e^{-2\lambda t} \tag{3.9}$$

A large decay constant λ corresponds to fast convergence. Now let us inspect the decay constant as function of friction shown in Fig. 3.4.

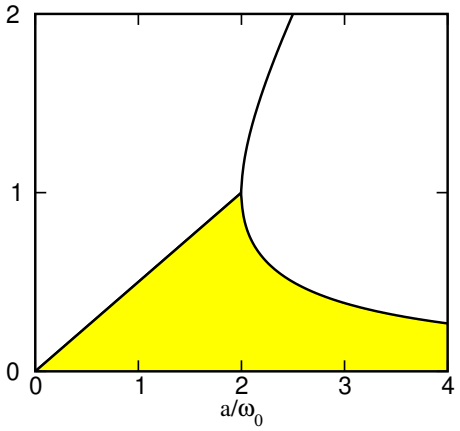


Fig. 3.4: Decay constant λ as calculated in Eq. 3.20 as function of friction coefficient a. ω_0 is the eigenfrequency of the harmonic oscillator.

We see that the maximum lies at $a = 2\omega_0$, that is at critical damping. Increasing the friction only slightly beyond that value leads to a dramatic deterioration of the convergence rate. Thus, if the friction is only chosen a little too large the convergence rate will break in dramatically.

OPTIMUM FRICTION

The best convergence is achieved for critical damping, that is

$$a = \omega_0 \Delta \tag{3.10}$$

What does this mean if we have to optimize the friction for a system with a wide range of frequencies? In figure 3.5, I show the convergence rate of the total energy[1] $E = E_{kin} + E_{pot} \sim e^{-t/T}$, now expressed by the decay time $T = \frac{1}{2 \cdot \mathrm{Im}(\omega)}$ as function of frequency for a fixed common friction factor. Each line in Fig. 3.5 corresponds to a different value of the friction. Each line has a transition between a flat segment in the high-frequency region and a sharp increase below a certain frequency. Thus, for each friction value, only those modes with a frequency above a certain cutoff are effectively converged at all. By increasing the friction we pass from the blue line via the green line to the red

[1] We consider the partial solution of a harmonic oscillator with the smallest decay constant. The total energy is then $E(t) = \frac{1}{2} m |\dot{x}|^2 + \frac{1}{2} c |x|^2 = E(0) \exp(-2 \cdot \mathrm{Im}(\omega) t)$.

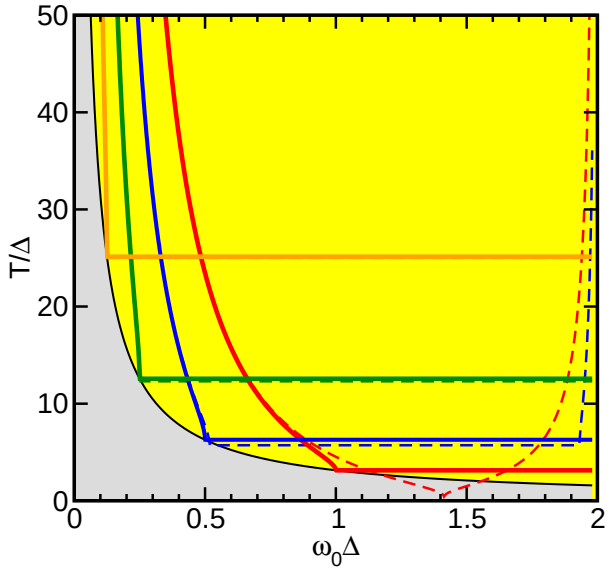


Fig. 3.5: Decay time T of the total energy as function of vibrational frequency ω_0. The decay time describes the convergence of the total energy as $E(T) \sim \exp(-\frac{t}{T})$. The friction parameters $a = \alpha\Delta/2$ are red: $a = 1$, blue: $a = 0.5$ green: $a = 0.25$ and orange: $a = 0.125$. $\omega_0\Delta = 2$ is the stability limit of the Verlet algorithm. The dashed lines are the results from the discretized equations, while the full lines are from the continuous trajectories. The grey region is not accessible with any friction parameter.

line. During this process we improve the convergence in the high-frequency part, but at the same time also the onset for convergence shifts towards higher frequency. This observation tells us that we should decide on the minimum frequency mode we wish to converge and then choose the friction that converges this mode and all higher ones. Often, this implies choosing a very small friction. If one is concerned about removing some modes from the convergence altogether, note, that these slow modes also carry little energy. If these modes are relevant, our analysis tells simply to be patient.

3.2.5 Numerical analysis of stability, accuracy, and convergence

In the previous sections we summarized some findings for the discretized trajectory. These results have been derived from a study of the damped one-dimensional harmonic oscillator. The harmonic oscillator can easily be solved analytically not only in the continuous form, but also in the discretized version.

In order to explore the stability and the accuracy one explores the damped harmonic oscillator.

$$m\ddot{x} = -cx - m\alpha\dot{x}$$

With the eigenfrequency $\omega_0 = \sqrt{c/m}$ we can rewrite this equation as

$$\ddot{x} = -\omega_0^2 x - \alpha\dot{x}$$

Next, we discretize the equation of motion using the Verlet replacement rule Eq. 3.4 with the time

step Δ and the friction parameter $a = \frac{\alpha\Delta}{2}$. The result is

$$\begin{aligned} x(+) - 2x(0) + x(-) &\overset{\text{Eq. 3.4}}{=} -(\omega_0\Delta)^2 x(0) - a\Big(x(+) - x(-)\Big) \\ \Rightarrow \qquad (1+a)x(+) + (1-a)x(-) &= \left[2 - (\omega_0\Delta)^2\right] x(0) \end{aligned}$$

With the ansatz $x(t) = \exp(i\omega t)$, we can use $x(+) = \exp(i\omega\Delta)x(0)$ and $x(-) = \exp(-i\omega\Delta)x(0)$, so that we obtain

$$(1+a)\exp(i\omega\Delta) + (1-a)\exp(-i\omega\Delta) = 2 - (\omega_0\Delta)^2 \tag{3.11}$$

which is a quadratic equation for $\exp(i\omega\Delta)$

$$(1+a)\exp^2(i\omega\Delta) - \Big(2 - (\omega_0\Delta)^2\Big)\exp(i\omega\Delta) + (1-a) = 0 \tag{3.12}$$

The two solutions for $\exp(i\omega\Delta)$ yield the two allowed values for ω, which I denote with ω_1 and ω_2. Their values allows one to construct the entire discretized trajectory for $t_j = \Delta j$ as

$$x(t_j) = A\exp(i\omega_1 t_j) + B\exp(i\omega_2 t_j) \tag{3.13}$$

with

$$A = \frac{x(0) - \mathrm{e}^{i\omega_2\Delta}x(-)}{1 - \mathrm{e}^{i(\omega_2-\omega_1)\Delta}} \qquad \text{and} \qquad B = \frac{x(0) - \mathrm{e}^{i\omega_1\Delta}x(-)}{1 - \mathrm{e}^{-i(\omega_1-\omega_1)\Delta}} \tag{3.14}$$

The two solutions of the quadratic equation Eq. 3.12 for $\exp(i\omega\Delta)$ are

$$\exp(i\omega\Delta) = \frac{1}{1+a}\left[\left(1 - \frac{(\omega_0\Delta)^2}{2}\right) \pm \sqrt{\left(1 - \frac{(\omega_0\Delta)^2}{2}\right)^2 - 1 + a^2}\ \right] \tag{3.15}$$

We distinguish three cases, one with positive argument of the square root and one with a negative one, and one with vanishing argument.

1. For a positive argument of the square root, we obtain a real value for $\exp(i\omega\Delta)$. If this value for $\exp(i\omega\Delta)$ is positive, the frequency is either purely imaginary so that $\omega = i\lambda$. If the value for $\exp(i\omega\Delta)$ is negative $\omega = i\lambda + \frac{\pi}{\Delta}$. The second result is obtained for $|\omega_0\Delta| > \sqrt{2}$.

   Each partial solution is either an exponentially decaying function or an alternating function with an exponentially decaying envelope. The solution is thus **overdamped**.

   The two values for $\lambda = \mathrm{Im}[\omega]$ are

$$\lambda_\pm \overset{\text{Eq. 3.15}}{=} \frac{1}{\Delta}\ln\left[\frac{1}{1+a}\left[\left(1 - \frac{(\omega_0\Delta)^2}{2}\right) \pm \sqrt{\left(1 - \frac{(\omega_0\Delta)^2}{2}\right)^2 - 1 + a^2}\ \right]\right] - i\frac{\pi}{\Delta}\theta\Big(|\omega_0\Delta| - \sqrt{2}\Big) \tag{3.16}$$

   where $\theta(x)$ is Heaviside's step function defined by $\theta(x) = 0$ for negative arguments and $\theta(x) = 1$ for positive arguments.

2. For a negative argument of the square root, we obtain a **damped oscillation**. One can separate real an imaginary part of the equation. We use $\omega = \bar{\omega} + i\lambda$, where $\bar{\omega}$ and λ are purely real. With $\mathrm{e}^{i\omega\Delta} = \Big[\cos(\bar{\omega}\Delta) + i\sin(\bar{\omega}\Delta)\Big]\mathrm{e}^{-\lambda\Delta}$ we obtain from Eq. 3.15

$$\cos(\bar{\omega}\Delta)\mathrm{e}^{-\lambda\Delta} = \frac{1 - \frac{(\omega_0\Delta)^2}{2}}{1+a} \tag{3.17}$$

$$\sin(\bar{\omega}\Delta)\mathrm{e}^{-\lambda\Delta} = \pm\frac{1}{1+a}\sqrt{1 - a^2 - \left(1 - \frac{(\omega_0\Delta)^2}{2}\right)^2} \tag{3.18}$$

Division of the two equations Eqs. 3.17,3.18 yields an expression for $\bar{\omega}$.

$$\tan(\bar{\omega}\Delta) = \pm\sqrt{\frac{1-a^2}{\left(1-\frac{(\omega_0\Delta)^2}{2}\right)^2}-1}$$

$$\Rightarrow \qquad \bar{\omega} = \frac{1}{\Delta}\text{atan}\left[\sqrt{\frac{1-a^2}{\left(1-\frac{(\omega_0\Delta)^2}{2}\right)^2}-1}\right] + \frac{\pi}{\Delta}\theta\left(|\omega_0\Delta|-\sqrt{2}\right) \tag{3.19}$$

where $\theta(x)$ is Heaviside's step function defined by $\theta(x) = 0$ for negative arguments and $\theta(x) = 1$ for positive arguments.

Adding the absolute squares of the real and imaginary part from Eqs. 3.17,3.18 removes the $\bar{\omega}$-dependence and provides the value of λ.

$$\underbrace{\left(\cos(\bar{\omega}\Delta)e^{-\lambda\Delta}\right)^2+\left(\sin(\bar{\omega}\Delta)e^{-\lambda\Delta}\right)^2}_{=e^{-2\lambda\Delta}} = \left[\frac{1-\frac{(\omega_0\Delta)^2}{2}}{1+a}\right]^2\left[1+\frac{1-a^2}{\left(1-\frac{(\omega_0\Delta)^2}{2}\right)^2}-1\right]$$

$$\Rightarrow e^{-2\lambda\Delta} = \left[\frac{1-\frac{(\omega_0\Delta)^2}{2}}{1+a}\right]^2\frac{1-a^2}{\left(1-\frac{(\omega_0\Delta)^2}{2}\right)^2} = \frac{1-a}{1+a}$$

$$\Rightarrow \lambda = \frac{1}{2\Delta}\ln\left|\frac{1+a}{1-a}\right| \tag{3.20}$$

The result is shown in Fig. 3.4. Note that the decay constant for the damped oscillation is independent of the vibrational frequency and only depends on the friction constant.

3. When the argument of the square root in Eq. 3.15 vanishes, we obtain a trajectory with **critical damping**.

$$1-a^2 = \left(1-\frac{\omega_0^2\Delta^2}{2}\right)^2$$

$$\Rightarrow \qquad a = \sqrt{1-\left(1-\frac{\omega_0^2\Delta^2}{2}\right)^2} = \sqrt{1-\left(1-2\frac{\omega_0^2\Delta^2}{2}+\left(\frac{\omega_0^2\Delta^2}{2}\right)^2\right)}$$

$$= \sqrt{\omega_0^2\Delta^2-\left(\frac{\omega_0^2\Delta^2}{2}\right)^2}$$

$$= \omega_0\Delta + O\left(\omega_0^2\Delta^2\right)^2$$

Thus if the time step is sufficiently small, we can choose the friction for critical damping, the optimum friction to $a = \omega_0\Delta$ as specified in Eq. 3.10

Once the complex frequencies are known, the trajectory is obtained via Eq. 3.13. From here the stability limit Eq. 3.7, the measure for the accuracy Eq. 3.8 and the value for the optimum friction Eq. 3.10 can be readily extracted.

In Fig. 3.6 the regions of distinct behavior of the Verlet algorithm are shown.

3.3 Constraints in the Verlet algorithm

We often encounter constraints on the trajectories in a molecular-dynamics simulation. A constraint may confine bond-lengths to avoid high-frequency oscillations in a classical molecular-dynamics simulation or it may constrain the wave functions to be orthonormal in a Car-Parrinello simulation.

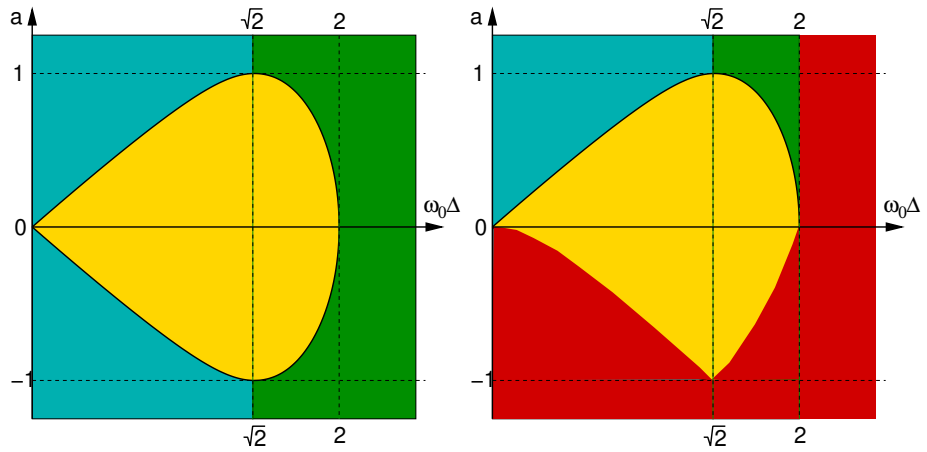


Fig. 3.6: Regions with distinct qualitative behavior of the discretized Verlet algorithm for an harmonic oscillator. Left graph: yellow: damped oscillation; blue exponential solutions (overdamped or exponential growth), green, i.e. $\omega_0\Delta > \sqrt{2}$, exponential behavior with alternating sign. Right graph: same as left graph but regions with unstable trajectories marked in red.

There is a strict recipe for including constraints invented by Ryckaert, Cicotti and Berendsen[132].

Constraints are formulated as the zero hypersurface of a function $G_\alpha(\vec{x})$ of the coordinates. There may be several constraints at the same time, which are labeled by the index α. Thus, each constraint is a condition of the form

$$G_\alpha(\vec{x}(t)) = 0 \; .$$

The first step is to extend our Lagrangian by adding the constraints using the method of Lagrange multipliers

$$\bar{\mathcal{L}}(\vec{x}, \vec{v}, t) = \mathcal{L}(\vec{x}, \vec{v}, t)) + \sum_\alpha \lambda_\alpha G_\alpha(\vec{x})$$

where $\mathcal{L}$ is the Lagrangian without the constraints and $\lambda(t)$ are the Lagrange multipliers.

The Euler-Lagrange equations are

$$\frac{d}{dt}\frac{\partial \bar{\mathcal{L}}}{\partial v_i} = \frac{\partial \bar{\mathcal{L}}}{\partial x_i} \qquad \Rightarrow \qquad \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial v_i} = \frac{\partial \mathcal{L}}{\partial x_i} + \underbrace{\sum_\alpha \frac{\partial G_\alpha}{\partial x_i}\lambda_\alpha}_{\text{constraint forces}}$$

The terms at the very right of the last equation are the constraint forces, that act on the particle in such a way that it never violates the constraints. The art of the procedure is to determine the Lagrange multipliers $\lambda_\alpha(t)$, that scale the constraint forces.

It is interesting to see that the constraint forces do not perform work on the system, so that energy conservation is not affected by adding constraint forces. Because $G_\alpha(\vec{x}(t)) = 0$ for all times, we find $\frac{d}{dt}G_\alpha(\vec{x}(t)) = \vec{v}\cdot\vec{\nabla}G_\alpha(\vec{x}(t)) = 0$. Thus, the constraint forces are always perpendicular to the trajectory. As they do not have a projection on the direction of the motion, they neither accelerate nor decelerate the motion.

To be less abstract, let us choose a simple specific Lagrangian

$$\mathcal{L}(\vec{x}, \vec{v}) = \frac{1}{2}m\vec{v}^2 - V(\vec{x}) \; .$$

The corresponding equation of motion has the form

$$m\ddot{\vec{x}} = \vec{F}(\vec{x}) + \sum_\alpha \lambda_\alpha \vec{\nabla} G_\alpha(\vec{x})$$

where $\vec{F} = -\vec{\nabla} V$ is the force.

Let us now discretize this equation of motion

$$\vec{x}(+) = \underbrace{2\vec{x}(0) - \vec{x}(-) + \vec{F}(\vec{x}(0))\frac{\Delta^2}{2m}}_{\vec{y}} + \sum_\alpha \lambda_\alpha \vec{\nabla} G_\alpha(\vec{x}(0))\frac{\Delta^2}{2m} = \vec{y} + \sum_\alpha \vec{\nabla} G_\alpha \frac{\Delta^2}{2m}\lambda_\alpha$$

We use the usual notation $\vec{x}(t+\Delta) = \vec{x}(+)$, $\vec{x}(t) = \vec{x}(0)$, $\vec{x}(t-\Delta) = \vec{x}(-)$, where Δ is the time step.

Next, we determine the Lagrange multipliers λ_α such that the constraints are exactly fulfilled in the next time step, that is

$$G_\beta(\vec{x}(+)) = G_\beta\left(\vec{y} + \sum_\alpha \vec{\nabla} G_\alpha \frac{\Delta^2}{2m}\lambda_\alpha\right) = 0$$

We may now expand $G_\beta(\vec{x}(+))$ in a Taylor expansion about some set of positions $\vec{x} + \sum_\alpha \vec{\nabla} G_\alpha \frac{\Delta^2}{m}\lambda_\alpha$

$$\begin{aligned} G_\beta(\vec{x}(+)) &= G_\beta(\vec{y}) + \left(\sum_\alpha \vec{\nabla} G_\alpha \frac{\Delta^2}{m}\lambda_\alpha\right)\vec{\nabla} G_\beta + O(\lambda^2) \\ &= G_\beta(\vec{y}) + \sum_\alpha \left(\frac{\Delta^2}{m}[\vec{\nabla} G_\beta][\vec{\nabla} G_\alpha]\right)\lambda_\alpha + O(\lambda^2) \end{aligned}$$

This equation is solved iteratively using

$$G_\beta(\vec{y}^{(n)}) + \sum_\alpha A_{\alpha,\beta}\lambda_\alpha^{(n)} = 0 \qquad \text{and} \qquad \vec{y}^{(n+1)} = \vec{y}^{(n)} + \sum_\alpha \vec{\nabla} G_\alpha \frac{\Delta^2}{m}\lambda_\alpha^{(n)}$$

where the matrix $A_{\alpha,\beta}$ is given as

$$A_{\alpha,\beta} = \frac{\Delta^2}{m}[\vec{\nabla} G_\beta][\vec{\nabla} G_\alpha]$$

by the constraint-gradients evaluated at $\vec{x}(0)$.

The converged value of $\vec{y}^{(\infty)}$ is simply the next set of coordinates $\vec{x}(+)$.

3.3.1 Simple example code

The following code describes a simple pendulum to demonstrate the use of constraints in a molecular-dynamics environment.

```
!*************************************************************************
!**  DESCRIBES A PENDULUM WITH A GIVEN LENGTH,MASS AND FORCE            **
!**  OF GRAVITY AS EXAMPLE FOR A MOLECULAR DYNAMICS SIMULATION          **
!**  WITH CONSTRAINTS                                                   **
!**                                                                     **
!*************************************************************************
MODULE MODEL
REAL(8),PARAMETER    :: PERIOD=11.D0
```

```
REAL(8),PARAMETER   :: OMEGA=6.28D0/PERIOD
REAL(8),PARAMETER   :: MASS=1.D0
REAL(8),PARAMETER   :: RAD2=1.D0
REAL(8),PARAMETER   :: GRAV=MASS*OMEGA**2/RAD2**2
END MODULE MODEL
!     ..............................................................
      PROGRAM MAIN
      USE MODEL
      IMPLICIT NONE
      REAL(8),PARAMETER :: DT=1.D0           ! TIME STEP
      REAL(8)           :: RM(3),R0(3),RP(3) ! POSITIONS
      REAL(8)           :: FORCE(3)          ! FORCE
      REAL(8)           :: LAMBDAM,LAMBDA0,LAMBDAP !LAGR. MULTIPLIER
      INTEGER(4)        :: ITER              ! ITERATION COUNTER
      INTEGER(4)        :: NITER=2000        ! #(ITERATIONS)
      REAL(8)           :: T                 ! TIME
      REAL(8)           :: EPOT,EKIN,ETOT    ! ENERGIES
      REAL(8)           :: DLAMBDA           ! CORRECTION FOR LAMBDA
      REAL(8)           :: V(3)              ! VELOCITY
      REAL(8)           :: err               ! deviation from constraint
!     ****************************************************************
!
!     ================================================================
!     == INITIALIZE VALUES                                          ==
!     ================================================================
      R0=0.D0
      R0(1)=SQRT(RAD2)
      RM=R0
      LAMBDA0=0.D0
      LAMBDAM=0.D0
      LAMBDAP=0.D0
!
!     ================================================================
!     ==  NOW START THE SIMULATION LOOP                              ==
!     ================================================================
      T=0
      DO ITER=1,NITER
!       == KINETIC ENERGY AND FORCES ===================================
        CALL ENERGY(R0,LAMBDA0,EPOT,FORCE)
!       == PROPAGATE ===================================================
        RP=2.D0*R0-RM+FORCE*DT**2/MASS
!       ==  APPLY CONSTRAINT ===========================================
        CALL CONSTRAINT(R0,RP,DLAMBDA,DT,ERR)
        LAMBDA0=LAMBDA0+DLAMBDA
        IF(ITER.EQ.1) then
          LAMBDAM=LAMBDA0
        end if
        LAMBDAP=2.D0*LAMBDA0-LAMBDAM
!       ==KINETIC ENERGY AND PRINT =====================================
        V=(RP-RM)/(2.D0*DT)
        EKIN=0.5D0*MASS*DOT_PRODUCT(V,V)
        WRITE(*,FMT='(10F10.5)')T,EKIN+EPOT,EKIN,EPOT,LAMBDA0,DLAMBDA,err,R0
!       == SWITCH ======================================================
        T=T+DT
```

```
        RM=R0
        R0=RP
        LAMBDAM=LAMBDA0
        LAMBDA0=LAMBDAP
      ENDDO
      STOP
      END
!
!     ..................................................................
      SUBROUTINE CONSTRAINT(R0,RP,DLAMBDA,DT,ERR)
      USE MODEL
      IMPLICIT NONE
      REAL(8),INTENT(IN)    :: R0(3)
      REAL(8),INTENT(INOUT) :: RP(3)
      REAL(8),INTENT(IN)    :: DT
      REAL(8),INTENT(OUT)   :: DLAMBDA
      REAL(8),INTENT(OUT)   :: ERR
      REAL(8)               :: FC(3)
      REAL(8)               :: A,B,C
!     ******************************************************************
      FC=-2.D0*R0*DT**2/MASS
      A=DOT_PRODUCT(RP,RP)-RAD2
      B=2.D0*DOT_PRODUCT(RP,FC)
      C=DOT_PRODUCT(FC,FC)
      DLAMBDA=-0.5D0*B/C+SQRT((0.5D0*B/C)**2-A/C)*SIGN(1.D0,B)
      RP=RP+FC*DLAMBDA
      ERR=DOT_PRODUCT(R0,R0)-RAD2
      RETURN
      END
!
!     ..................................................................
      SUBROUTINE ENERGY(R,LAMBDA,ETOT,FORCE)
      USE MODEL
      IMPLICIT NONE
      REAL(8),INTENT(IN) :: R(3)
      REAL(8),INTENT(IN) :: LAMBDA
      REAL(8),INTENT(OUT):: ETOT
      REAL(8),INTENT(OUT):: FORCE(3)
!     ******************************************************************
      ETOT=GRAV*R(3)+LAMBDA*(DOT_PRODUCT(R,R)-RAD2)
      FORCE(:)=-2.D0*R*LAMBDA
      FORCE(3)=FORCE(3)-GRAV
      RETURN
      END
```

3.3.2 Orthonormality constraint

The tricky point in solving the equations of motion is how to describe the constraints. Let us consider the equation of motion discretized according the Verlet algorithm. We also add a friction term, which

is often used in practice

$$
\begin{aligned}
m_\psi|\ddot{\tilde{\psi}}_n\rangle &= -\hat{H}|\psi_n\rangle - m_\psi|\dot{\tilde{\psi}}_n\rangle\alpha_\Psi + \sum_m |\psi_m\rangle\Lambda_{m,n}\frac{1}{f_n} \\
m_\psi\frac{|\psi_n(+)\rangle - 2|\psi_n(0)\rangle + |\psi_n(-)\rangle}{\Delta^2} &= -\hat{H}(0)|\psi_n(0)\rangle - m_\psi\frac{|\psi_n(+)\rangle - |\psi_n(-)\rangle}{2\Delta}\alpha_\Psi \\
&\quad + \sum_m |\psi_m(0)\rangle\Lambda_{m,n}\frac{1}{f_n} \\
|\psi_n(+)\rangle - 2|\psi_n(0)\rangle + |\psi_n(-)\rangle &= -m_\psi^{-1}\hat{H}(0)|\psi_n(0)\rangle\Delta^2 - (|\psi_n(+)\rangle - |\psi_n(-)\rangle)\underbrace{\frac{\alpha_\Psi\Delta}{2}}_{=:a_\psi} \\
&\quad + \sum_m m_\psi^{-1}|\psi_m(0)\rangle\Lambda_{m,n}\frac{1}{f_n}\Delta^2
\end{aligned}
\tag{3.21}
$$

$$
\begin{aligned}
|\psi_n(+)\rangle &= \underbrace{|\psi_n(0)\rangle\frac{2}{1+a_\psi} - |\psi_n(-)\rangle\frac{1-a_\psi}{1+a_\psi} - m_\psi^{-1}\hat{H}(0)|\psi_n(0)\rangle\frac{\Delta^2}{1+a_\psi}}_{|\bar{\psi}_n\rangle} \\
&\quad + \sum_m \underbrace{m_\psi^{-1}|\psi_m(0)\rangle\frac{\Delta^2}{1+a_\psi}}_{|\chi_m\rangle}\Lambda_{m,n}\frac{1}{f_n}
\end{aligned}
\tag{3.22}
$$

Thus we obtain an equation of the form:

$$
|\psi_n(+)\rangle = |\bar{\psi}\rangle + \sum_n |\chi_m\rangle\Lambda_{m,n}\frac{1}{f_n} \tag{3.23}
$$

The Lagrange parameters are then determined from the condition

$$\langle\psi_n(+)|\psi_m(+)\rangle = \delta_{n,m}$$

which results in an equation for the Lagrange multiplicators

$$
\begin{aligned}
&\underbrace{\left(\langle\bar{\psi}_n|\bar{\psi}_m\rangle - \delta_{n,m}\right)}_{A_{n,m}} + \sum_i \underbrace{\langle\bar{\psi}_n|\chi_i\rangle}_{B_{n,i}}\underbrace{\Lambda_{i,m}\frac{1}{f_m}}_{X_{i,m}} + \sum_i \underbrace{\frac{1}{f_n}\Lambda^*_{i,n}}_{X^\dagger_{n,i}}\underbrace{\langle\chi_i|\bar{\psi}_m\rangle}_{B^\dagger_{m,i}} \\
&\qquad + \sum_{i,j} \underbrace{\frac{1}{f_n}\Lambda^*_{i,n}}_{X^\dagger_{n,i}}\underbrace{\langle\chi_i|\chi_j\rangle}_{C_{i,j}}\underbrace{\Lambda_{j,m}\frac{1}{f_m}}_{X_{j,m}} = 0 \\
&\qquad \mathbf{A} + \mathbf{BX} + \mathbf{X}^\dagger\mathbf{B}^\dagger + \mathbf{X}^\dagger\mathbf{CX} = 0
\end{aligned}
$$

In addition to this quadratic equation, we need to ensure that the Lagrange multipliers are hermitian. This requirement follows from condition of energy conservation.

$$\mathbf{\Lambda} = \mathbf{\Lambda}^\dagger$$

In contrast to the one-dimensional quadratic equation, the quadratic equation with matrices is nontrivial to solve and requires an iterative procedure. If this loop does not converge, the CP-PAW code stops with an error message “loop for orthogonalization not converged”.

3.4 Constant temperature: the Nose thermostat

The seemingly simple but ingenious **Nose thermostat**[101] has a thorough physical foundation. Namely, it constructs an exact canonical ensemble.

$$M_i \ddot{\vec{R}}_i = \vec{F}_i - M_i \dot{\vec{R}}_i \dot{x}_R \tag{3.24}$$

$$Q_R \ddot{x}_R = 2\left(\sum_R \frac{1}{2} M_i \dot{\vec{R}}^2 - \frac{1}{2} g k_B T\right) \tag{3.25}$$

The equations of Nose are substantially more complicated, but they can be rewritten in the form given above. This form is closely related to that given by Hoover[102]. I have modified his expression so that a second order differential equation for the thermostat variable results[103, 104]

Even though the equations of motion appear like a feedback equation, the thermostat does not come to rest, but performs fluctuations completely consistent with the canonical ensemble.

3.4.1 Analysis of the equations of motion

The dynamics of the Nose thermostat can be in a two-dimensional representation, that includes only the kinetic energy of the system and the variable of the Nose thermostat.

To understand the equations of motion let us multiply the first equation with the velocities.

$$\underbrace{\sum_i M_i \dot{\vec{R}}_i \ddot{\vec{R}}_i}_{\partial_t E_{kin}} = \underbrace{\sum_i \vec{F}_i \dot{\vec{R}}_i}_{-\partial_t E_{pot}} - 2\underbrace{\left(\sum_i \frac{1}{2} M_i \dot{\vec{R}}_i^2\right)}_{E_{kin}} \dot{x}_R \tag{3.26}$$

$$Q_R \ddot{x}_R = 2\Big(\underbrace{\sum_R \frac{1}{2} M_i \dot{\vec{R}}^2}_{E_{kin}} - \frac{1}{2} g k_B T\Big) \tag{3.27}$$

Let us now introduce the variables $E_{kin}(t) = \frac{1}{2} g k_B T + e(t)$ and $y(t) = \dot{x}_R(t)$, which yields

$$\dot{e} = -g k_B T y - 2ey - \partial_t E_{pot}(t) \tag{3.28}$$

$$\dot{y} = \frac{2}{Q_R} e \tag{3.29}$$

By inserting the second equation of motion into the first, w obtain

$$\frac{Q_R}{2} \ddot{y} = -g k_B T y - 2\frac{Q_R}{2} y \dot{y} - \partial_t E_{pot}(t) \tag{3.30}$$

which has to linear order an eigenfrequency of

$$\omega = \sqrt{\frac{2 g k_B T}{Q_R}} \tag{3.31}$$

3.4.2 Characteristics of the Nose dynamics

The typical behavior of the Dynamics under a Nose thermostat is shown in figure 3.7.

The analysis shows that the kinetic energy oscillates around the target value with a given frequency. In our program we exploit this to hide the variable Q_R from the user. The latter is calculated directly from a specified target frequency. The frequency is valid in the limit of small oscillations.

If the oscillations are large, we find time intervals, where the motion is effectively damped out, which are interrupted by a short energy fluctuation. Such a behavior is often encountered in the initial phase of the simulation, where it can produce problems by breaking up the system under study.

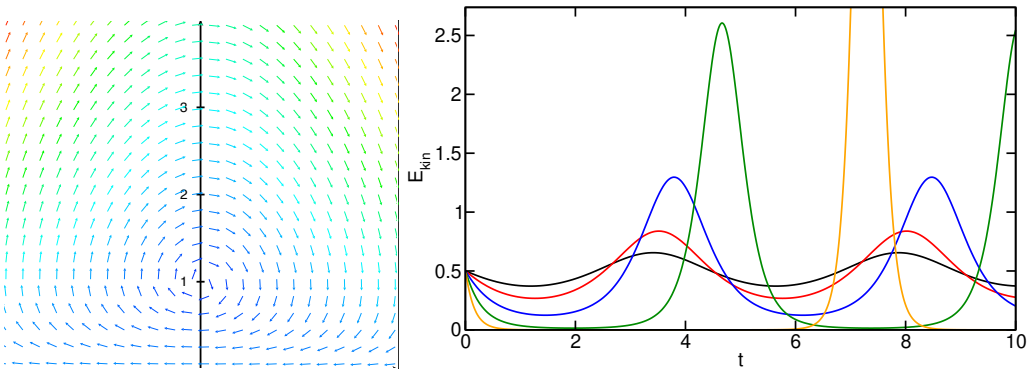


Fig. 3.7: Dynamics of the Nose thermostat in the absence of a potential energy. Left figure: vertical axis: kinetic energy in units of $\frac{1}{2}gk_BT$. Horizontal axis: velocity of the Nose variable x_R. The vector field corresponds to the velocities of the system in this coordinate system. The drawing indicates the direction of the vector field by the direction of the arrows and the size of the vectors by the color coding. Right figure: kinetic energy in the absence of forces as driven by the Nose thermostat. The kinetic energy is given in units of gk_BT.

As long as the forces acting on the system are neglected, the system will stay on a given orbit about the stable point. The only intrinsic effect that determines the size of the orbit, are the changes in the potential energy. The potential energy pushes the system more or less randomly from one to the other of the trajectories.

Chapter 4

Plane waves

4.1 Basissets

In order to deal with wave functions we need a numerical representation that allows us to represent the continuous wave function by a discrete set of numbers. For this purpose we introduce a so-called **basisset**. The basisset is a set of functions $\chi_\alpha(\vec{r},\sigma)$, which are identified by an index α and which depend on the spatial variable $\vec{r}$ and a spin variable σ.

A wave function $\psi(\vec{r})$ can be expanded into this basisset

$$\psi(\vec{r},\sigma) = \sum_\alpha \chi_\alpha(\vec{r},\sigma) c_\alpha \tag{4.1}$$

The choice of a basisset is critical for the accuracy and efficiency of an electronic structure method. Let me name the most common basissets:

- Slater functions

  $$\chi_\alpha(\vec{r},\sigma) = |\vec{r}|^{\ell+2n_\alpha} \mathrm{e}^{-\lambda_\alpha|\vec{r}|} Y_{\ell_\alpha,m_\alpha}(\vec{r}) \delta_{\sigma,\sigma_\alpha} \tag{4.2}$$

  are derived from the wave functions of a generalized hydrogen atom. Analytical integrations are, however, complex, so that Slater functions. A code using Slater functions is the Amsterdam Density-Functional (ADF) code.

- Gauss functions

  $$\chi_\alpha(\vec{r},\sigma) = |\vec{r}|^{\ell+2n_\alpha} \mathrm{e}^{-\lambda_\alpha|\vec{r}|^2} Y_{\ell_\alpha,m_\alpha}(\vec{r}) \delta_{\sigma,\sigma_\alpha} \tag{4.3}$$

  are used in the majority of quantum chemistry calculations. Compared to atomic orbitals they fall off too fast and they do not exhibit a cusp at the origin, which is present in the true wave functions. However, they have the big advantage that most integrations can be performed analytically. Most well known is the GAUSSIAN code package, Turbomole, ORCA, etc.

- Plane waves

  $$\chi_\alpha(\vec{r},\sigma) = \mathrm{e}^{i\vec{G}_\alpha\vec{r}} \delta_{\sigma,\sigma_\alpha} \tag{4.4}$$

  are probably the most simple basisset. It is usually used in conjunction with the pseudopotential approximation. Plane waves are, however, also used as envelope function in augmented wave methods such as the augmented plane wave (APW) method, the linear augmented plane wave method (LAPW) and the projector augmented wave (PAW) method.

- Augmented wave methods can use other basis sets. Because they replace the behavior of the so-called envelope function in the atomic region by numerical solutions of the Schrödinger equation, they are very flexible in the choice of the basisset for the envelope function. One widely used basisset are spherical Hankel functions, which are solutions to the Helmholtz equation. Like plane waves, spherical Hankel functions solve the Schrödinger equation for a constant potential. They are used in methods such as the Linear augmented muffin tin orbital (LMTO) method, the closely related Augmented spherical wave (ASW) method, and the Korringa-Kohn-Rostokker (KKR) method.

The underlying reason for the success of plane waves is firstly, that a basis set of plane waves can be made complete in a systematic way. Secondly, most operations with plane waves are extraordinarily simple. Most importantly, the transformation between the expansion coefficients of a function and its real space representation can be done with a highly efficient numerical technique, the so-called **Fast Fourier transform**.

4.2 Plane waves

The pseudo part of the wave functions, densities and potentials are expanded into plane waves. A **plane wave** is a function having the form

$$\chi_\alpha(\vec{r}) = e^{i\vec{G}_\alpha \vec{r}} \tag{4.5}$$

where $\vec{G}$ is called the **wave vector**. To simplify the discussion, we drop the spin dependence from here on.

Thus, a general wave function has the form

$$\psi(\vec{r}) = \sum_\alpha e^{i\vec{G}_\alpha \vec{r}} \psi(\vec{G}_\alpha) \tag{4.6}$$

The plane wave coefficients are $\psi(\vec{G}_\alpha)$. The notation may be somewhat misleading, because we use the same symbol for the wave function and its plane wave coefficients. Furthermore, we use $\psi(\vec{G}_\alpha)$ instead of ψ_α.[1]

In order to deal with wave functions in a plane wave representation on the computer, we need to limit the number of coefficients $\psi(\vec{G}_\alpha)$ to a finite set. This involves two steps:

- We select a discrete grid of G-vectors $\vec{G}_\alpha$. This is equivalent to choosing wave functions for periodic crystals. Such wave functions are the so-called **Bloch states**.
- we limit the length of the wave vectors considered. For this purpose we introduce the so-called **plane wave cutoff** E_{PW} and impose the requirement

  $$\frac{1}{2}\vec{G}_\alpha^2 < E_{PW} \tag{4.7}$$

  A finite plane wave cutoff limits the strength of the corrugation of the wave function, and the convergence with this parameter needs to be checked.

4.3 Real-space lattice

In our codes, we mostly deal with functions that are periodic with the lattice vectors of the crystal. If $\vec{t}$ is any of the real space lattice vectors, the periodicity of a function $f(\vec{r})$ with $\vec{t}$ implies

$$f(\vec{r} + \vec{t}) = f(\vec{r}) \tag{4.8}$$

[1] The origin for this notation is probably that $\psi(\vec{r}) = \langle \vec{r} | \psi \rangle$ and $\psi(\vec{G}) = \langle \vec{G} | \psi \rangle$.

The lattice vectors $\{\vec{t}\}$ define a regular grid of points in real space. The general lattice vectors $\vec{t}_{i,j,k} = \vec{T}_1 i + \vec{T}_2 j + \vec{T}_3 k$ are multiples of the three primitive lattice vectors $\vec{T}_1, \vec{T}_2$ and $\vec{T}_3$. It is often convenient to combine the primitive lattice vectors into a matrix $\boldsymbol{T}$.

$$\boldsymbol{T} \hat{=} \begin{pmatrix} T_{1,x} & T_{2,x} & T_{3,x} \\ T_{1,y} & T_{2,y} & T_{3,y} \\ T_{1,z} & T_{2,z} & T_{3,z} \end{pmatrix} \tag{4.9}$$

because the general lattice vector can be represented as a matrix vector product

$$\vec{t}_{i,j,k} = \boldsymbol{T} \begin{pmatrix} i \\ j \\ k \end{pmatrix} \ . \tag{4.10}$$

4.4 Reciprocal-space lattice

Any periodic function $f(\vec{r})$ can be expanded into a series of plane waves

$$f(\vec{r}) = \sum_G \mathrm{e}^{i\vec{G}\vec{r}} F(\vec{G})$$

where the G-vectors are taken from the **reciprocal lattice**. The constants $F(\vec{G})$ are called the **plane-wave coefficients** of the function $f(\vec{r})$.

The reciprocal lattice is formed by all G-vectors, for which the plane wave $\mathrm{e}^{i\vec{G}\vec{r}}$ is periodic with the lattice, i.e.

$$\mathrm{e}^{i\vec{G}(\vec{r}+\vec{t})} = \mathrm{e}^{i\vec{G}\vec{r}} \tag{4.11}$$

This implies $\mathrm{e}^{i\vec{G}\vec{t}} = 1$ or

$$\vec{G}\vec{t} = 2\pi n \ , \tag{4.12}$$

where n is some integer.

The general reciprocal space vectors $\vec{G}_{i,j,k} = \vec{g}_1 i + \vec{g}_2 j + \vec{g}_3 k$ are multiples of three primitive reciprocal lattice vectors $\vec{g}_1, \vec{g}_2$ and $\vec{g}_3$. They can be combined into a (3×3) matrix $\boldsymbol{g}$, which obeys[2]

$$\boldsymbol{g}^\top \boldsymbol{T} = 2\pi \mathbf{1} \qquad \text{or} \qquad \boldsymbol{g} = 2\pi \left(\boldsymbol{T}^{-1}\right)^\top \tag{4.13}$$

To the beginner, the reciprocal lattice appears like a fairly abstract object. However, it can be connected directly to physical properties. Some of them are illustrated in Fig. fig:graphite. They can be summarized as follows:

PHYSICAL MEANING OF RECIPROCAL LATTICE VECTORS

The directions of the reciprocal lattice vectors are perpendicular to the lattice planes. The length of the *primitive* reciprocal lattice vectors is 2π divided by the distance of the lattice planes.

4.5 Fourier Transform

With the help of the concept of the reciprocal lattice we can formulate the Fourier transform for periodic functions.

[2] The transpose of a matrix $\boldsymbol{A}$ is written as $\boldsymbol{A}^\top$ and has elements $\left(\boldsymbol{A}^\top\right)_{i,j} = A_{j,i}$.

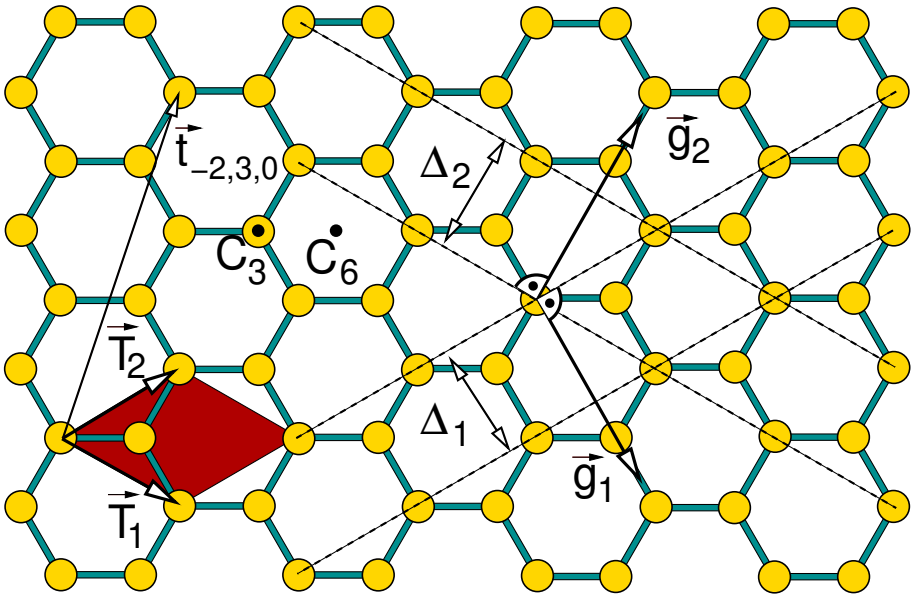


Fig. 4.1: Translational and rotational symmetry of a single graphite sheet and demonstration of reciprocal lattice vectors. Graphite is a layered material, which consists of sheets as the one shown. The yellow balls indicate the carbon atoms and the blue sticks represent the bonds. In graphite these two-dimensional sheets are stacked ontop of each other, where every second sheet is shifted such that an atom falls vertically below the point indicated by C_6. Here we only consider the symmetry of a single sheet. The elementary unit cell is indicated by the red parallelogram, which is spanned by the elementary lattice vectors $\vec{T}_1$ and $\vec{T}_2$. The third lattice vector points perpendicular to the sheet towards you. An example for a general lattice vector is $\vec{t}_{3,-2,0} = 3\vec{T}_1 - 2\vec{T}_2 + 0\vec{T}_3$. The lattice planes indicated by the dashed lines. The lattice planes are perpendicular to the sheet. The distance of the lattice planes are indicated by Δ_1 and Δ_2. The elementary reciprocal lattice vectors are $\vec{g}_1$ and $\vec{g}_2$. The third reciprocal lattice vector points perpendicular out of the plane towards you. Note that the reciprocal lattice vectors have the unit "inverse length". Thus their length is considered irrelevant in this real space figure. The axis through C_3 standing perpendicular to the plane is a three-fold rotation axis of the graphite crystal. The axis through C_6 perpendicular to the plane is a 6-fold rotation axis of the graphite sheet, but only a three-fold rotation axis of the graphite crystal. (Note that a rotation axis for the graphite crystal must be one for both sheets of the crystal). In addition there is a mirror plane lying in the plane. Furthermore there are several mirror planes perpendicular to the plane: One passing through every atom with one of the three bonds lying in the plane and one perpendicular each bond passing through the bond center.

FOURIER TRANSFORM OF PERIODIC FUNCTIONS

For a function that is periodic on a lattice specified by the primitive lattice vectors $\vec{T}_1, \vec{T}_2, \vec{T}_3$ the forward and back transforms are

$$f(\vec{r}) = \sum_{\vec{G}} \mathrm{e}^{i\vec{G}\vec{r}} F(\vec{G}) \tag{4.14}$$

$$F(G) = \frac{1}{V} \int_V d^3r \, \mathrm{e}^{-i\vec{G}\vec{r}} f(\vec{r}) \tag{4.15}$$

where $V = \det(\boldsymbol{T})$ is the volume of the primitive real-space unit cell. The integration is performed over one unit cell of the real-space lattice. The sum over G-vectors includes all reciprocal lattice vectors.

4.6 Hamiltonian contributions and FFT's

The main advantage of the plane wave basisset is that the kinetic energy and the Coulomb interaction are diagonal in reciprocal space and that the potential energy in a potential is diagonal in r space. Furthermore the transform from real to reciprocal space can be done with enormous efficiency using **Fast Fourier Transforms (FFT)**.

Normally such a transformation takes n^2 operations, where n is the number of grid points. A fast fourier transform on the other hand takes only $n \log n$ operations.

The kinetic energy of the pseudo wave functions has the form

$$\tilde{E}_{kin} = \sum_n f_n \langle \psi_n | \frac{\hat{\vec{p}}^2}{2m_e} | \psi_n \rangle_V = V \sum_n f_n \sum_{\vec{G}} \frac{(\hbar \vec{G})^2}{2m_e} |\tilde{\psi}_n(\vec{G})|^2 \tag{4.16}$$

I have introduced here the subscript V at the bracket to indicate that the scalar product is defined as an integral over a single unit cell only. Usually this restriction is silently assumed, where appropriate.

The electrostatic energy is calculated from the charge density as

$$E = \frac{1}{2} \int_V d^3r \int_\infty d^3r' \, \frac{\rho(\vec{r})\rho(\vec{r'})}{4\pi\epsilon_0 |\vec{r} - \vec{r'}|} = V \sum_{\vec{G}, \Gamma \in \{\vec{G}\}} \frac{-1}{\epsilon_0 |\vec{G}|^2} |\rho(\vec{G})|^2 \tag{4.17}$$

Only one of the two real-space integrations is restricted to the first unit cell, while the other is performed over the entire space. The charge density $\rho(\vec{r})$ is e times the total electronic charge density plus the charge density of the nuclei.

We observe the singularity at the Γ point, i.e. $\vec{G} = \vec{0}$. This singularity is only avoided, if the charge density $\rho(\vec{G})$ vanishes at the Γ-point, which implies that the average charge density must vanish. This is not surprising because in this case the total charge of the entire system becomes infinite, which results in an infinite potential and therefore an infinite electrostatic interaction between the density of one unit cell with the rest.

In order to remedy this problem, a **compensating charge background** is automatically added to ensure that the average density vanishes.[3] In practice, the Γ-point is simply excluded from the sum for the electrostatic self energy.

The electron density is constructed in real space. The wave functions are Fourier transformed into real space, where the product is formed and the density is summed up. Once this has been done, the result is Fourier transformed into reciprocal space.

Similarly, the wave functions are multiplied with the potentials in real space.

4.7 Plane wave cutoff

The maximum length G_{max} of all wave vectors $\vec{G}$ of a plane wave is a measure of the length scale $\Delta = \pi / G_{max}$ on which the wave function can vary.

This value is usually expressed by the plane wave cutoff E_{PW} which is the maximum kinetic energy of all plane waves.

$$E_{PW} = \frac{(\hbar G_{max})^2}{2m_e}$$

It is often specified in the **Rydberg energy unit** Ry. One Rydberg is the total energy of a hydrogen atom, namely the binding energy of electron and proton.

[3] For finite systems, the CP-PAW code has a mechanism to deal with truly charged systems without a compensating charge background.

It is vital to use a spherical plane wave cutoff to avoid breaking the spherical symmetry. Thus all plane coefficients $\psi(\vec{G})$ with

$$\vec{G} \leq \frac{1}{\hbar}\sqrt{2m_e E_{PW}} \tag{4.18}$$

are included in the expansion and all other are excluded. Remember, that this cutoff applies to the displaced lattice, which includes $\vec{k}$ and not the unshifted lattice which includes the Γ-point.

4.8 Plane wave cutoff for the density

For a give plane wave cutoff for the wave functions the plane wave cutoff for the density should be four times higher. This follows from the expression for the electron density $n(\vec{r})$.

$$n(\vec{r}) = \sum_n f_n \psi_n^*(\vec{r})\psi_n(\vec{r}) = \sum_n f_n \sum_{G,G'} \psi_n^*(\vec{G})\psi_n(\vec{G})e^{i(\vec{G}-\vec{G'})\vec{r}} \tag{4.19}$$

We see that the largest value for $\vec{G}-\vec{G'}$ is $2G_{max}$, when $|\vec{G}|, |\vec{G'}| < G_{max}$. Because the maximum G-vector enters quadratically into the expression for the plane wave cutoff we find

$$E_{PW}(n) = 4E_{PW}(\psi) \tag{4.20}$$

In practice we reduce the plane wave cutoff to a smaller value to speed up the calculation.

4.9 Plane wave convergence

The number of plane waves can be calculated from the plane wave cutoff as

$$N_{PW} = \frac{V}{(2\pi)^3}\frac{4\pi}{3}\left(\frac{1}{\hbar}\sqrt{2m_e E_{PW}}\right)^3 = \underbrace{\frac{1}{(2\pi\hbar)^3}\frac{4\pi}{3}}_{0.01688 a.u.} \cdot V\left(2m_e E_{PW}\right)^{\frac{3}{2}} \tag{4.21}$$

where V is the volume of the unit cell.

Example our cell for malonaldehyde has an fcc unit cell with a lattice constant of 2*5.9944 Åand thus a volume of $V = 2*(5.9944/0.529177)^2 a_0^3$, which leads to a prediction of 8067 plane waves for the wave functions and 22816 plane waves for the density using a plane wave cutoff of 60 Ry for the latter. The program calculates 7991 plane waves for the wave functions and 11498 for the density. The lower value for the density is because we can exploit that the density is real so that only one-half of all g-vectors needs to be kept.

4.10 Sawtooth

One big advantage of plane waves is that the basis set does not depend on the atomic structure. However it depends on the lattice vectors. When the lattice periodicity is changed, for example, when we change the volume of the unit cell, the changes of the basis set need to be taken into account.

If one fixes the size of the basis set, the crystal will artificially contract. The reason is that the quality of the basisset increases as the lattice vectors become smaller: The wave functions can describe corrugations with a shorter characteristic length.

Therefore, it is important to fix the plane wave cutoff when changing the lattice vectors. The shape of the energy versus volume curve will converge faster with plane wave cutoff.

However as we change the volume, the number of basis functions does not change continuously. Each time additional basis functions are allowed, the energy will drop because the basis set has a

larger variational degree of freedom. If one calculates the total energy as function of volume, one observes a **Sawtooth** behavior.

In order to obtain reliable values it is important to choose points for the volumes that are sufficiently far apart so that the sawtooth behavior of the energy can be ignored.

The sawtooth disappears if the calculation is well converged with the number of place waves. The sawtooth can also be made smaller by using a fine k-point grid, because then the distance between the "teeth" becomes smaller.

Chapter 5

K-points and Brillouin zone integration

5.1 Bloch states

In a crystal the potential and the charge density have the periodicity of the crystal lattice[1]. The wave functions, however, are not periodic.

Still, because the Hamiltonian is periodic, the wave functions can be chosen as eigenstates of the lattice translation operator $\hat{S}(\vec{t})$ defined by

$$\hat{S}(\vec{t}) \stackrel{\text{def}}{=} \int d^3r \, |\vec{r}+\vec{t}\rangle\langle\vec{r}| \qquad \Leftrightarrow \qquad \hat{S}(\vec{t})\psi(\vec{r}) = \psi(\vec{r}-\vec{t}) \tag{5.1}$$

The eigenvalue equation has the form

$$\psi(\vec{r}-\vec{t}) = \psi(\vec{r})\mathrm{e}^{-i\vec{k}\vec{t}} \tag{5.2}$$

where $\mathrm{e}^{-i\vec{k}\vec{t}}$ is the eigenvalue.

The eigenvalue equation implies[2] that

$$u(\vec{r}) \stackrel{\text{def}}{=} \psi(\vec{r})\mathrm{e}^{i\vec{k}\vec{r}} \tag{5.4}$$

is periodic.

BLOCH THEOREM

The one-particle wave functions of a crystal can be written as **Bloch states**

$$\psi(\vec{r}) = u(\vec{r})\mathrm{e}^{i\vec{k}\vec{r}} \tag{5.5}$$

namely as product of a periodic function $u(\vec{r})$, which obeys

$$u(\vec{r}+\vec{t}) = u(\vec{r}) \, , \tag{5.6}$$

and a phase factor $\mathrm{e}^{i\vec{k}\vec{r}}$. The **Bloch vector** $\vec{k}$ is a good quantum number of the system.

[1] We do not consider symmetry breaking.

[2] We start from the eigenvalue equation

$$\psi(\vec{r}-\vec{t}) = \psi(\vec{r})\mathrm{e}^{-i\vec{k}\vec{t}} \qquad \Rightarrow \qquad \psi(\vec{r}-\vec{t})\mathrm{e}^{-i\vec{k}\vec{t}} = \psi(\vec{r}) \qquad \Rightarrow \qquad \underbrace{\psi(\vec{r}-\vec{t})\mathrm{e}^{i\vec{k}(\vec{r}-\vec{t})}}_{u(\vec{r}-\vec{t})} = \underbrace{\psi(\vec{r})\mathrm{e}^{i\vec{k}\vec{r}}}_{u(\vec{r})} \tag{5.3}$$

The periodic part can be represented by a Fourier expansion with G-vectors $\vec{G} = \vec{g}_1 i + \vec{g}_2 j + \vec{g}_3 k$ from the reciprocal lattice

$$u(\vec{r}) = \sum_{\vec{G};\Gamma\in\{\vec{G}\}} U(\vec{G}) \mathrm{e}^{i\vec{G}\vec{r}} \tag{5.7}$$

The Γ point is the origin $(0,0,0)$ of the reciprocal lattice.

The full wave function may also be expressed by a sum over G vectors that are displaced away from the Γ-point so that $\vec{k}$ is part of the shifted lattice.

$$\psi(\vec{r}) = \Big(\sum_{\vec{G};\Gamma\in\{\vec{G}\}} \mathrm{e}^{i\vec{G}\vec{r}} U(\vec{G}) \Big) \mathrm{e}^{i\vec{k}\vec{r}} = \Big(\sum_{\vec{G};\Gamma\in\{\vec{G}\}} \mathrm{e}^{i(\vec{k}+\vec{G})\vec{r}} U(\vec{G}) \Big) = \sum_{\vec{G};\vec{k}\in\{\vec{G}\}} \mathrm{e}^{i\vec{G}\vec{r}} \psi(\vec{G}) \tag{5.8}$$

where $\psi(\vec{G}) = U(\vec{G} - \vec{k})$.

I have purposely written up the wave function in several different notations, because one can easily be confused by the varying notation of different authors. I, personally, prefer a rather uncommon notation, namely the last one in Eq. 5.8. This is important to keep in mind while reading my notes.

5.2 Bloch vector as good quantum number

The Hamiltonian matrix elements between two arbitrary Bloch states, the result vanishes whenever the two Bloch states differ. The Hamiltonian in a basis of Bloch states is block diagonal. This allows one to obtain the eigenstates for each Bloch-vector individually.

Expectation values, which are sums over all one-particle states, need to be obtained as a sum over a unit cell in reciprocal space. This integration is called **Brillouin-zone integration** for historical reasons.[3]

The matrix elements of the Hamiltonian as function of the Bloch vector are what is called the **band structure**.

Bloch states for a given Bloch vector $\vec{k}$ can be transformed into equivalent states for a Bloch vector $\vec{k}' = \vec{k} + \vec{G}$ shifted by a reciprocal lattice vector by multiplying the periodic function with a periodic plane wave.

$$\psi(\vec{r}) = u(\vec{r}) \mathrm{e}^{i\vec{k}\vec{r}} = \underbrace{\Big(u(\vec{r}) \mathrm{e}^{-\vec{G}\vec{r}} \Big)}_{u'(\vec{r})} \mathrm{e}^{i(\vec{k}+\vec{G})\vec{r}} \tag{5.9}$$

This immediately tells us that the band structure, as well as the matrix element of any one-particle Hamiltonian between Bloch states is periodic with the reciprocal lattice. Hence a single periodic unit cell contains the complete information on the system.

5.3 Band structures

This is the reason why band structures are not represented in an **extended zone scheme** but in a **reduced zone scheme**.

- in the extended zone scheme the band structure of free electrons would be a single parabola, and would extend to infinity in reciprocal space.
- in the **periodic zone scheme**, the free electron would consists of many parabola centered at the reciprocal lattice points. Thus the reciprocal zone scheme is periodic with the periodicity of the reciprocal lattice.

[3] The Brillouin zone is a symmetric representation of a unit cell in reciprocal space. The Brillouin zone consists of all points in reciprocal space, that are closer to the origin than to any other reciprocal lattice point.

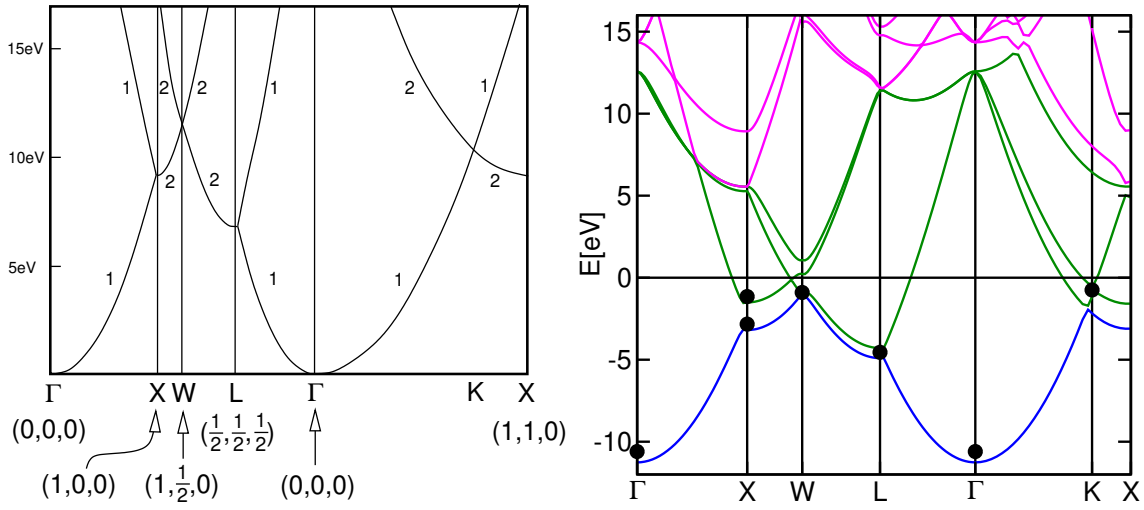


Fig. 5.1: Band structure of free, non-interacting electrons. The lattice is an fcc-cell with a lattice constant of 4.05 Å corresponding to aluminum. The high symmetry points are given in units of $\frac{2\pi}{a_{lat}}$. The numbers indicate the degeneracy beyond spin-degeneracy. On the right-hand side, the band structure of aluminum is shown in comparison.

- in the reduced zone scheme, only the irreducible part of the periodic zone scheme is shown, that is one single repeat unit. The band structures shown in Fig. 5.1 are in a reduced zone scheme. There are several choices for the repeat units of the reciprocal lattice. One choice is simply the primitive unit cell of the reciprocal lattice. However, the shape of the unit cell does not have the point-group symmetry of the reciprocal lattice. Therefore one instead chooses the **Wigner Seitz cell**[4] of the reciprocal lattice, which is called the **Brillouin zone**.

In the reduced zone scheme, all wave vectors connected by a reciprocal lattice vector, fall ontop of the same point in the Brillouin zone. Thus a Hamiltonian that has lattice periodicity couples all points that lie at the same point in the Brillouin zone. This is demonstrated in Fig. 5.2.

From perturbation theory we know that the splitting of two interacting states is large when the two states are close or even degenerate. Thus the largest effect in the band structure occurs right at the boundary of the irreducible zone, where the degeneracy of the free electron gas is lifted by the periodic potential. Thus local band gaps appear at the surface of the Brillouin zone. This band gap, however, is usually warped, so that there is no energy window, that completely lies in all local band gaps. In that case the material remains a metal.

5.4 Brillouin-zone integrations

When we evaluate matrix elements of one-particle operators, we perform a Brillouin zone integration of the form

$$\langle A \rangle = \sum_n \frac{1}{\Omega_G} \int_{\Omega_G} d^3k \; f(\epsilon_{\vec{k},n}) \langle \psi_{\vec{k},n} | \hat{A} | \psi_{\vec{k},n} \rangle \tag{5.10}$$

where the function $f(\epsilon) = \left[1 + e^{\frac{1}{k_B T}(\epsilon - \mu)}\right]^{-1}$ is the Fermi distribution function, which determines if the state $|\psi_{\vec{k},n}\rangle$ is occupied or not. It depends on the chemical potential μ for the electrons and the temperature T. In the low-temperature case relevant for us, the Fermi distribution function is a step

[4] The Wigner Seitz cell of a lattice consists of all points that are closer to the origin than to any other lattice point. Thus, it is enclosed by planes perpendicular to the lattice vectors cutting the lattice vector in half.

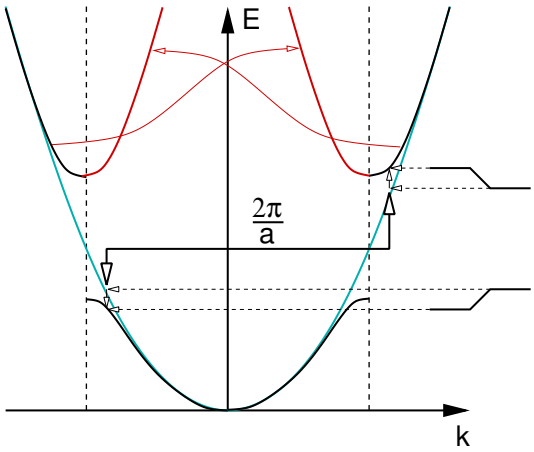


Fig. 5.2: Schematic demonstration how a periodic potential results in a coupling between states with the same wave vector in the reduced zone scheme. If a free particle experiences a potential with periodicity a, states with wave vectors that differ by a reciprocal lattice vector $G_n = \frac{2\pi}{a}n$ couple. The coupling shifts the upper state up and the lower state down, which results in an opening of the gap. As a result of the coupling, the wave vector of a free particle state is no more a good quantum number. However, the wave vector in the reduced zone scheme remains to be a good quantum number.

function that vanished for states above the Fermi level and is equal to one (or two, if spin degeneracy is included) if the energy is below the Fermi level.

Ω_G is the volume of the reciprocal unit cell. It is given by the primitive real space lattice vectors as

$$\Omega_G = \frac{(2\pi)^2}{\det|\mathbf{T}|} \tag{5.11}$$

The integral of the matrix element is an integral over the real-space unit cell

$$\langle\psi_{\vec{k},n}|\hat{A}|\psi_{\vec{k},n}\rangle = \int_{\Omega_T} d^3r\ \psi^*_{\vec{k},n}(\vec{r})\hat{A}\psi_{\vec{k},n}(\vec{r}) \tag{5.12}$$

and the states are normalized such that

$$\langle\psi_{\vec{k},n}|\psi_{\vec{k},n'}\rangle = \delta_{n,n'} \tag{5.13}$$

5.4.1 Sampling

The most simple integration scheme on the market is **k-point sampling**. The integral is discretized on a lattice over which the result is summed.

We first select the so-called Monkhorst-Pack lattice

$$\vec{k}_{i,j,k} = \vec{G}_1\frac{i}{N_1} + \vec{G}_2\frac{j}{N_2} + \vec{G}_3\frac{k}{N_3} \tag{5.14}$$

Then we evaluate the expectation value as sum over all k-points

$$\langle A\rangle = \sum_n \frac{1}{N_1N_2N_3}\sum_{i,j,k} f(\epsilon_{\vec{k}_{i,j,k},n})\langle\Psi_{\vec{k}_{i,j,k},n}|\hat{A}|\Psi_{\vec{k}_{i,j,k},n}\rangle \tag{5.15}$$

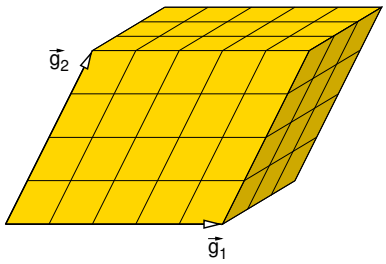


Fig. 5.3: k-point grid in the unit cell of the reciprocal lattice. The primitive reciprocal lattice vectors are $\vec{g}_1, \vec{g}_2$ and $\vec{g}_3$.

Very important is that the k-points form a regular lattice, because the result is equivalent to a Fourier interpolation followed by an integration. Remember therefore that the way the k-points are chosen is not arbitrary! For completely filled bands, the result of the discretized sum is the same as the integral of the Fourier interpolated matrix elements in reciprocal space. As a consequence the integral for filled bands converges very fast, namely exponentially. The results for partially occupied bands are not as accurate, because the sampling method is not able to capture the correct shape of the Fermi surface. In this case the improved tetrahedron method[133], described below, is the method of choice.

5.4.2 Tetrahedron method

For metals there is a better strategy than sampling, namely the improved tetrahedron method[133]. The improved tetrahedron method builds upon the linear tetrahedron method invented independently by Jepsen and Andersen and by Lehman and Taut[134, 135].

The tetrahedron method also starts out from the same regular grid as the sampling method. However, it divides divides space between the grid points into tetrahedra and interpolates the energy $\epsilon_n(\vec{k})$ and the matrix elements $A_n(\vec{k})$ linearly inside each tetrahedron. All the information for the linear interpolation is obtained from the values at the four corners of a tetrahedron. The occupied states fill a polyhedron with a triangulated surface.

The modern tetrahedron method[133] is formulated such that filled bands are treated identically to the sampling method on a Monckhorst-Pack mesh. Thus the method profits from the exponential convergence for all filled bands. Furthermore there is no difference between this implementation of the tetrahedron method and the zero-temperature sampling method for insulators.

While the bands and matrix elements are piecewise linear, the modern tetrahedron method[133] contains an additional correction, which goes beyond the linear interpolation of the matrix elements and effectively includes quadratic terms.

The notion behind this correction[133] is that the linear approximation overestimates bands with a positive curvature, while it underestimates those with negative curvature. For filled bands the average curvature vanishes so that these errors cancel. For partially filled bands, it is possible to estimate the average curvature of the bands in the occupied region from the slope of the bands at the Fermi surface. The latter is accessible from the linearly interpolated bands. Thus the error can be estimated and used as a correction for the results.

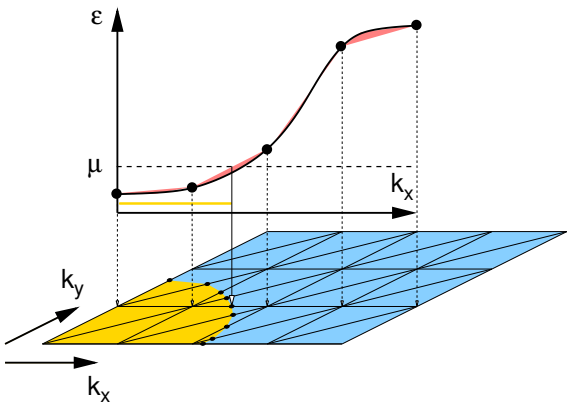


Fig. 5.4: Scheme to demonstrate the interpolation errors of the conventional tetrahedron method and the correction in the improved tetrahedron method.

5.5 Grids and such

5.5.1 Shifted k-point grids

It is possible to shift the k-point grid so that the Γ-point is avoided[5]. The Γ-point is a high symmetry point and therefore describes either the maxima or the minima of a band. This is usually a bad estimate for the average value of the band that we finally need. Therefore it us better to shift the grid away from Gamma.

5.5.2 k-point spacing and real-space cutoff

The sampling method with $N_1 N_2 N_3$ k-points is equivalent to a calculation in a supercell in real space with $N_1 \times N_2 \times N_3$ unit cells. Thus even the discretized k-point set describes a physical system.

A supercell can be characterize by the minimum distance between periodic images in real space. This minimum distance is ideally as large as possible. Therefore, we use this minimum distance to characterize the quality of the k-point grid.

We define the parameter slightly differently, namely completely analogous to the plane wave cutoff. A matrix element in k-space is expanded in plane waves in reciprocal space

$$A_n(\vec{k}) = \sum_{\vec{t}} a_n(\vec{t}) e^{i\vec{t}\vec{k}} \tag{5.16}$$

where $a_n(\vec{t})$ are the Fourier components. Because of the periodicity in of the matrix elements in reciprocal space, only Fourier coefficients at the real space lattice vectors contribute. In order to find a cutoff that does not break the symmetry, we only include those lattice vectors for which

$$\vec{t} < \vec{R} \tag{5.17}$$

where R is a parameter defining number of Fourier coefficients. R should be taken so large that a perturbation is screened out over this distance. Such a criterion depends of course on the requirements on the accuracy of a calculations, but in general the value will be a few nanometers.

[5] The Γ-point is the origin of the reciprocal lattice, i.e. $\vec{k}_\Gamma = (0, 0, 0)$.

5.5.3 Time-inversion symmetry and real wave functions

While we do not exploit spatial symmetries in the CP-PAW code, we do exploit **time-inversion symmetry**.

The Schrödinger equation for a charged particle in an electromagnetic field is symmetric under simultaneous time-inversion and reversal of magnetic fields. That is, the Schrödinger is invariant under the following transformation (P. Blöchl, ΦSX: The electronic structure of matter)

$$\begin{aligned}\psi'(\vec{r},t) &= \psi^*(\vec{r},-t)\\ \vec{A}'(\vec{r},t) &= -\vec{A}(\vec{r},-t)\\ \Phi'(\vec{r},t) &= \Phi(\vec{r},-t)\end{aligned} \tag{5.18}$$

Here $\vec{A}$ and Φ are the electromagnetic vector potential and the scalar electromagnetic potential respectively. With $\psi(\vec{r},t) = \mathrm{e}^{-i\omega_n t}\psi_n(\vec{r})$ this implies

$$\psi'_n(\vec{r}) = \psi^*_n(\vec{r}) \tag{5.19}$$

and for the Fourier coefficients

$$\psi'_n(\vec{G}) = \psi^*_n(-\vec{G}) \tag{5.20}$$

Time-inversion symmetry thus results in the relation

$$\Psi_{-\vec{k},n}(\vec{r}) = \Psi^*_{\vec{k},n}(\vec{r}) \tag{5.21}$$

Note, that this does not hold for non-collinear calculations, because time-inversion symmetry is broken by the effective magnetic field.

The last equation says that if we know an eigenstate of the Hamiltonian with a Bloch vector $\vec{k}$, we can immediately can construct a corresponding eigenstate at $-\vec{k}$. As demonstrated in Fig. 5.5, this has the following consequences:

- only the wave function for about half of the Bloch vectors need to be explicitly calculated, while the other half is obtained via Eq. 5.20. Hence, the matrix elements of about half of the k-points is identical to another one. Therefore only the matrix elements at the irreducible k-points, i.e. only one from a pair of related k-points, are explicitly calculated and their sampling weight is doubled.

- For each lattice there are 8 k-points for which $-\vec{k}$ and $+\vec{k}$ differ by exactly one reciprocal lattice vector. For these k-points the wave functions can be chosen real. In other words: if a wave function for this k-point is complex, both, real and imaginary part are independent solutions of the Schrödinger equation.

There are special k-points that are mapped onto each other. They do not obtain the doubled integration weight, as they are related only to itself but not to another k-point. The special k-points are those that can be mapped onto itself by an inversion at the Γ point (the origin) and a lattice

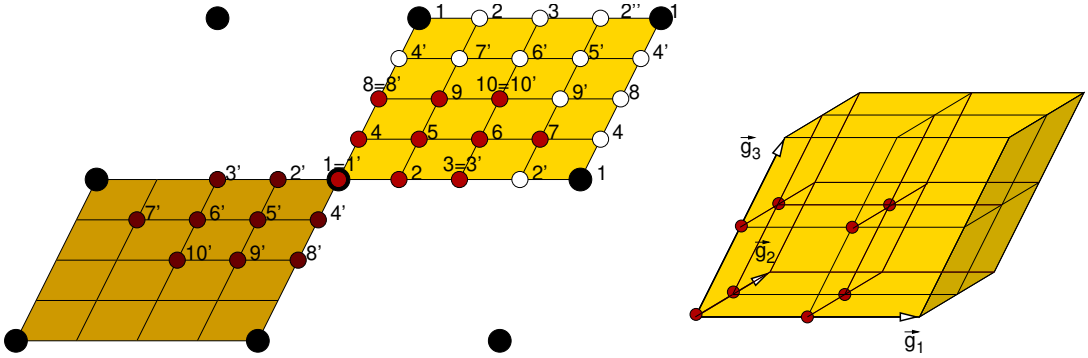


Fig. 5.5: Demonstration of the relation of k-points by time-inversion symmetry. Time inversion symmetry corresponds to an inversion symmetry in reciprocal space. a k-point x is related by inversion at a reciprocal lattice point to point x'. All points that are related by a reciprocal lattice translation are indicated by the same number x or x'. Thus in a grid with a 4×3 division, only 10 k-points are independent instead of 12. Four of the points are special that is they only require half the computational effort than the general k-points. The computational effort is this reduced by a factor $\frac{8}{12}$

translation in reciprocal space. They are:

$$\vec{k} = \vec{0}$$
$$\vec{k} = \frac{1}{2}\vec{g}_1$$
$$\vec{k} = \frac{1}{2}\vec{g}_2$$
$$\vec{k} = \frac{1}{2}\vec{g}_3$$
$$\vec{k} = \frac{1}{2}\vec{g}_2 + \frac{1}{2}\vec{g}_3$$
$$\vec{k} = \frac{1}{2}\vec{g}_1 + \frac{1}{2}\vec{g}_3$$
$$\vec{k} = \frac{1}{2}\vec{g}_1 + \frac{1}{2}\vec{g}_2$$
$$\vec{k} = \frac{1}{2}\vec{g}_1 + \frac{1}{2}\vec{g}_2 + \frac{1}{2}\vec{g}_3$$

If time-inversion symmetry is exploited, the wave functions can be assumed to be real. This reduces the amount of data in half.

In fact, we can superimpose two real wave function into a complex **super wave function**. Thus we need to consider only half of the wave functions.

$$|\Psi^{sup}\rangle := |\Psi_1\rangle + i|\Psi_2\rangle \tag{5.22}$$

In real space the two real wave functions are obtained from

$$\langle\vec{r}|\Psi_2\rangle = \mathrm{Im}\langle\vec{r}|\Psi^{sup}\rangle \tag{5.23}$$
$$\langle\vec{r}|\Psi_1\rangle = \mathrm{Re}\langle\vec{r}|\Psi^{sup}\rangle \tag{5.24}$$

The original wave functions can also be obtained in reciprocal space as follows, when we realize that

for a real wave function the relation $\langle -\vec{G}|\Psi\rangle = \langle \vec{G}|\Psi\rangle^*$

$$\langle \vec{G}|\Psi^{sup}\rangle = \langle \vec{G}|\Psi_1\rangle + i\langle \vec{G}|\Psi_2\rangle \tag{5.25}$$

$$\langle -\vec{G}|\Psi^{sup}\rangle = \langle -\vec{G}|\Psi_1\rangle + i\langle -\vec{G}|\Psi_2\rangle = \langle \vec{G}|\Psi_1\rangle^* + i\langle \vec{G}|\Psi_2\rangle^* \tag{5.26}$$

$$\tag{5.27}$$

$$\langle -\vec{G}|\Psi^{sup}\rangle^* = \langle \vec{G}|\Psi_1\rangle - i\langle \vec{G}|\Psi_2\rangle \tag{5.28}$$

$$\langle \vec{G}|\Psi_1\rangle = \frac{1}{2}\left(\langle \vec{G}|\Psi^{sup}\rangle + \langle -\vec{G}|\Psi^{sup}\rangle^*\right) \tag{5.29}$$

$$\langle \vec{G}|\Psi_2\rangle = \frac{-i}{2}\left(\langle \vec{G}|\Psi^{sup}\rangle - \langle -\vec{G}|\Psi^{sup}\rangle^*\right) \tag{5.30}$$

Chapter 6

Supercells

This whole chapter is only a draft

Not all materials are crystals, for which we can exploit translational symmetry. Consider for example amorphous materials, quasi-crystals, liquids, molecules, surfaces, interfaces.

Even though these materials are no crystals, the existence or the absence of a long-range order may not be relevant for the problems at hand.

In that case we may introduce an artificial long-range order by forming super-cells.

Purely electronic effects are typically screened after a distance of about 1 nm.

6.1 Supercells for molecules

In order to describe a molecule we need to place it on a lattice and ensure that the molecules on the lattice to not interact with each other. The identical molecules that are introduced to form a lattice are called **periodic images** of each other.

There are two effects to consider

- If the wave functions of periodic images overlap, they may form the onset of chemical bonds and antibonds.
- An electrostatic interaction between the periodic images is present if the molecule is charged or if it has electrostatic multipoles.

6.1.1 Electrostatic decoupling

If the molecule is charged or if it creates multipole moments, the periodic images interact by the Coulomb interaction. The Coulomb interaction is long ranged, which implies that, for monopoles and dipoles, the interaction does not even vanish for infinite distances. This implies that it is impossible to determine ionization potentials without eliminating this artificial electrostatic interaction between the periodic images.

In order to remedy this in a simple fashion, I developed the electrostatic decoupling[120] of the periodic images.

The idea is the following:

1. First we construct a model for the instantaneous charge distribution, by fitting a set of atom-centered spherical Gaussian charge densities to the actual charge density. This fit is done in such a way that the long range contribution of the electrostatic potential is particularly well described.

Instead of a fit by atom centered Gaussians one might also consider to make a multipole expansion of the charge density. This is however not convenient, because it converges poorely for more complex molecules. The multi-center monopole expansion chosen in our method is superior to the one-center multipole expansion.

2. The model charge density can be converted into a point charge model, which has identical interaction between the periodic images.

3. For the point charge model, we can determine the electrostatic interaction for an infinite array of molecules, and we can calculate the electrostatic energy of a single molecule. The difference is the electrostatic interaction between the periodic images.

4. the interaction energy between the periodic images is subtracted from the total energy. This defines an additional term to the total energy functional which is analytically specified by the atomic positions and the wave functions.

5. By forming the analytic derivatives of the energy correction we obtain consistent contributions to the forces acting on the atoms and to the potential contributing to the Hamiltonian.

Here we did not mention the contribution of the interaction with the charge compensating background, which is discussed in the original paper.[120]

6.1.2 Wave function overlap

As a rule of thumb we should use a minimum of 6 Å of vacuum between the periodic images. This is the minimum atom-atom distance for a molecule saturated by hydrogen atoms. More space is required if orbitals with weakly bound electrons point away from the molecule. These orbitals are, for example, lone-pairs and dangling bonds, in particular of the second and higher rows of the periodic table. Similarly, if we consider metal clusters, the orbitals can reach extremely far out, so that the cell-size convergence must be carefully tested.

6.1.3 Shape of the supercell

For molecules it is convenient to use an fcc-type supercell. The fcc crystal corresponds to a closed packe sphere packing. Thus the Wigner-Seitz cell, the region of all points that are closer to the origin than to any other lattice point, is very symmetrical and has the best ratio of distance between periodic images and supercell volume.

6.2 Supercells for crystals

The most simple approach from a crystal to a supercell is to simply multiply each lattice vector by a factor. The disadvantage of this approach is that the steps are often too large. Doubling the cell in each direction leads to an 8-fold increase in the volume and that in turn leads to and approximately 100-fold increase in computer time. Therefore, it is important to consider also other types of supercells.

I demonstrate the approach for the most common case, namely that of cubic crystals. There are three types of cubic supercells, namely simple cubic (sic), face-centered cubic (fcc) and body centered cubic (bcc). The lattice vectors are

$$\begin{array}{lll} \vec{T}_1 = (0, \frac{1}{2}, \frac{1}{2})a & \vec{T}_1 = (-\frac{1}{2}, \frac{1}{2}, \frac{1}{2})a & \vec{T}_1 = (1, 0, 0)a \\ \vec{T}_2 = (\frac{1}{2}, 0, \frac{1}{2})a \quad \text{for fcc,} & \vec{T}_2 = (\frac{1}{2}, -\frac{1}{2}, \frac{1}{2})a \quad \text{for bcc, and} & \vec{T}_2 = (0, 1, 0)a \quad \text{for sic,} \\ \vec{T}_3 = (\frac{1}{2}, \frac{1}{2}, 0)a & \vec{T}_3 = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2})a & \vec{T}_3 = (0, 0, 1)a \end{array}$$

where a is the simple cubic lattice constant. The volumes are $V = \frac{1}{4}a^3$ for the fcc lattice, $V = \frac{1}{2}a^3$ for the bcc lattice, and $V = a^3$ for the sic lattice.

By multiplying the axes we obtain the following supercells

Type	Multiplicity	V/a^3
fcc	1^3	1/4
bcc	1^3	1/2
sic	1^3	1
fcc	2^3	2
bcc	2^3	4
fcc	3^3	6.75
sic	2^3	8
bcc	3^3	13.5
fcc	4^3	16
sic	3^3	27
fcc	5^3	31.25
bcc	4^3	32
fcc	6^3	54
bcc	5^3	62.5
sic	4^3	64

Only those supercells can be used for which the volume is a multiple of the elementary (smallest) unit cell. The ratio of V/a^3 for the selected supercell and the elementary unit cell provides us with the multiplication factor of for the number of atoms.

The procedure as follows.

- First one selects a cell of the required size.
- Then one constructs the smallest supercell having the same type as the selected supercell.
- Finally one multiplies the lattice vectors by the factors given under multiplicity.

6.3 Slab calculation

te Velde and Baerends investigated the convergence of adsorption energies with respect to slab thickness[136]. They also established that adsorption energies cannot be converged with cluster calculations, while rapid convergence is reached for slabs.

Discuss the following points: Dangling bonds, Polar surfaces, Electron count, Saturation by capping-hydrogen-atoms with fractional atomic number.

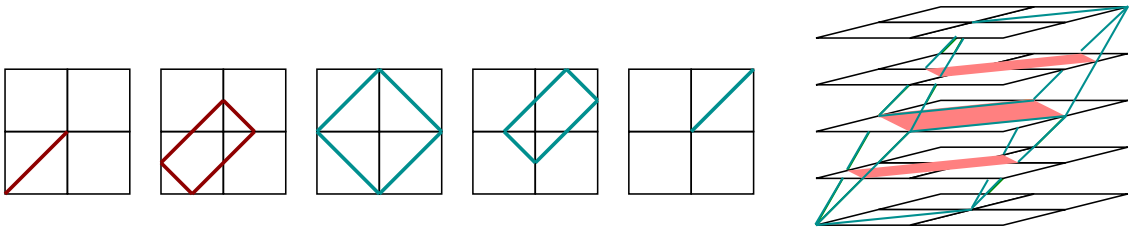


Fig. 6.1: Fcc supercell in a super-cube. Schemes as this are helpful to select the relevant atoms in the supercell, if the atomic positions in the cubic cell are easily constructed.

Chapter 7

Energy versus volume equation of state

7.1 Murnaghan equation of state

Murnaghan[137] proposed to interpolate the energy versus volume curve on the assumption that the bulk modulus varies linearly with pressure. The resulting interpolation formula is used to extract equilibrium energy and volume, bulk modulus and its derivative. Here we will derive the explicit form of it:

MURNAGHAN'S ASSUMPTION

The assumption of Murnaghan is

$$B(p) = B_0 + B'p \tag{7.1}$$

where $B(p)$ is the bulk modulus, B_0 is its value at equilibrium volume, and B' is the pressure derivative of the bulk modulus at equilibrium volume.

The isotropic pressure is defined as the volume derivative of the internal energy

$$p = -\frac{dE}{dV} \tag{7.2}$$

and the bulk modulus is defined as

$$B = -V\frac{dp}{dV} \tag{7.3}$$

As shown below, Murnaghan's assumption leads to

MURNAGHAN'S EQUATION OF STATE

$$E(V) \overset{\text{Eq. 7.6}}{=} E_0 + \frac{B_0 V_0}{B'(B'-1)}\left[\left(\frac{V}{V_0}\right)^{-(B'-1)} - 1\right] + \frac{B_0 V_0}{B'}\left[\frac{V}{V_0} - 1\right] \tag{7.4}$$

where V_0 is the equilibrium volume, E_0 is the energy at equilibrium volume, B_0 is the bulk modulus at equilibrium volume and B' is the pressure derivative of the Bulk modulus.

7.1.1 Derivation of Murnaghan's equation of state

Expressing Murnaghan's assumption Eq. 7.1 by the energy, we obtain a differential equation for $p(V)$.

$$B(p) \overset{\text{Eq. 7.3}}{=} -V\frac{dp(V)}{dV} \overset{\text{Eq. 7.1}}{=} B_0 + B'p$$

We rewrite this expression as differential equation of $V(p)$

$$-\frac{1}{V}\frac{dV(p)}{dp} = \frac{1}{B_0 + B'p}$$

We transform to the logarithm of the volume

$$-\frac{d}{dp}\ln[V(p)] = \frac{1}{B_0 + B'p}$$

which can be integrated. We introduce the equilibrium volume V_0.

$$\begin{aligned}
-\left(\ln[V(p)] - \ln[V_0]\right) &= \int_0^p dp' \frac{1}{B_0 + B'p'} = \left[\frac{1}{B'}\ln[B_0 + B'p']\right]_0^p \\
&= \frac{1}{B'}\ln[B_0 + B'p] - \frac{1}{B'}\ln[B_0] \\
\Rightarrow \left(\frac{V_0}{V}\right)^{B'} &= \frac{B'}{B_0}p + 1 \\
\Rightarrow p(V) &= \frac{B_0}{B'}\left[\left(\frac{V}{V_0}\right)^{-B'} - 1\right] \qquad (7.5)
\end{aligned}$$

Using the relation Eq. 7.2 between energy and pressure, we see that a simple integration takes us to the form for the energy versus volume. We introduce the equilibrium energy $E_0 = E(V_0)$.

$$\begin{aligned}
E(V) &= E_0 - \int_{V_0}^{V} dV'\, p(V') \\
&= E_0 - \int_{V_0}^{V} dV' \left\{\frac{B_0}{B'}\left[\left(\frac{V}{V_0}\right)^{-B'} - 1\right]\right\} \\
&= E_0 - \frac{B_0}{B'}\left[-\frac{V_0}{B'-1}\left(\frac{V}{V_0}\right)^{-(B'-1)} - V\right]_{V_0}^{V} \\
&= E_0 + \frac{B_0 V_0}{B'(B'-1)}\left[\left(\frac{V}{V_0}\right)^{-(B'-1)} - 1\right] + \frac{B_0 V_0}{B'}\left(\frac{V}{V_0} - 1\right) \qquad (7.6)
\end{aligned}$$

Let us test the result: From Eq. 7.6, it is evident that $E(V_0) = E_0$.

$$\begin{aligned}
p(V) \overset{\text{Eq. 7.2}}{=} -\frac{dE}{dV} \overset{\text{Eq. 7.6}}{=} \frac{B_0}{B'}\left[\left(\frac{V}{V_0}\right)^{-B'} - 1\right] \qquad (\Rightarrow p(V_0) = 0) \\
\Rightarrow \frac{dp}{dV} = -\frac{B_0}{V_0}\left(\frac{V}{V_0}\right)^{-B'-1} \qquad \left(\Rightarrow V\frac{dp}{dV}\Big|_{V_0} = B_0\right) \\
\Rightarrow B = -V\frac{dp}{dV} = B_0\left(\frac{V}{V_0}\right)^{-B'} \overset{\text{Eq. 7.5}}{=} B_0\left[\frac{B'}{B_0}p + 1\right] = B_0 + B'p
\end{aligned}$$

Thus we have reproduced Murnaghan's assumption.

Chapter 8

Time, length and energy scales

It is important to develop a intuitive feeling for the size of physical quantities. This helps not only to detect obvious mistakes, but to develop strategies for the calculations. In the following, I am collecting a few numbers that I find helpful.

Disclaimer: The numbers in this section need to be checked.

8.1 Time scales

Hartree atomic time unit	0.024 fs
Car-Parrinello time step	0.12 fs
Period of visible light	1.3-2.3 fs
H_2 vibration	8 fs
XH stretch vibration	15 fs
H_2O bend vibration	20 fs
optical phonon of Si	70 fs
1 nm/speed-of-sound in iron	200 fs
Waiting time for reactions E_a=0.2 eV T=1000 K	1 ps
Waiting time for reactions E_a=0.5 eV T=1000 K	30 ps
Waiting time for reactions E_a=0.5 eV T=300 K	25 μs
Single-electron transfer rate in an STM (0.03 nA)	5 ns

Waiting times for chemical reactions: The reaction rate can be estimated by $\Gamma = \frac{2\pi}{\omega} e^{-E_A/k_B T}$, where ω related to the attempt frequency, and E_A is the activation energy. The attempt frequency can be approximated by $\omega = 2\pi/20fs$. This number is reasonable within a factor 2-5. At room temperature the waiting time for a reaction with a small barrier of 10 kJ/mol typical for hydrogen bonds is about 1.4 ps. If the barrier increases to a 50 kJ/mol typical for a low barrier chemical reaction the waiting time would be already 10^7 ps. Above 1000 K the waiting time for reactions with barriers of 50 kJ/mol decreases in the picosecond range.[138]

8.2 Length scales

- atomic distances: 1-3 Å
- distance of van-der Waals bonded systems 3-4 Å
- amplitude of thermal vibrations (only bond distances) 0.1 Å

- 1nm=10 Å
- wave-length of visible light 400-700 nm

8.3 Energy scales

- chemical accuracy 0.05 eV. A change of a reaction barrier by 0.05 eV changes the reaction rate at room temperature by a factor of 10.
- energies of van der Waals bonds 0.2 eV
- energies of covalent bonds 2-9 eV
- Madelung energy of NaCl: 3.6 eV per ion
- band gap of silicon: 1.14 eV
- thermal energy of a free molecule at room temperature $\frac{3}{2}k_BT = 38$ meV, thermal energy $k_BT = 25$ meV for vibrations.
- energies of photons of visible light 1.6-3.27 eV.

$$\hbar\omega = \frac{1239.84187eV}{\lambda/nm} \tag{8.1}$$

Source:`http://en.wikipedia.org/wiki/Electronvolt`

Chapter 9

Structure of the CP-PAW code

The spirit of the CP-PAW code is that it iteratively solves a differential equation in time for a large amount of variables. This leads to a fairly simple repetitive structure of the code, which is very similar to a classical molecular dynamics code. The simple structure has the advantage that we can add complexity in the description of the system, such as including quantum and classical variables, including thermostats, adding an environment with a different method, including many-particle electronic correlations, and many more. There is a price, namely that one needs to understand the global features and time scales of the dynamics in order to be able to effectively control the dynamics.

The second speciality of the CP-PAW code is its object oriented structure. As a consequence, flow-charts are not a useful description for many parts of the code. Rather one should think of the structure as a number of agents, that each are responsible for a particular topic, and that "collaborate" on the progress of the computation.

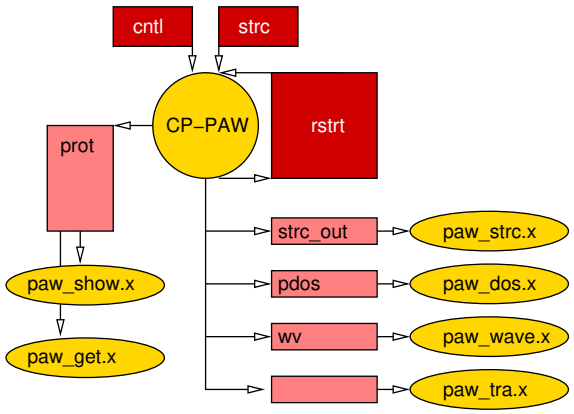


Fig. 9.1: Sketch of the CP-PAW code. Codes are denoted by spheres and elipses. Data files are denoted by rectangles. For a description see text.

An overview on the workings of the CP-PAW package is given in figure 9.1, which shows the information exchange between the codes (sphers and elipses) and files (rectangles). The central part is the CP-PAW simulation code. It takes two files, the structure file "strc" and the control file "cntl" as input. In order to continue a calculation it writes and rewrites the current state to the restart file "rstrt". The central output file is the protocoll file "prot". The protocoll file reports the settings of the input data and the progress of the calculation. It can be inspected by tools such as "paw_show.x" and "paw_get". More involved data are written on machine readable files. Some of these are written per default. Others need a request by the control file. The data from these files are converted with

dedicated tools into a form that can be accessed, for example, by generic visualization tools.

9.1 General structure of a molecular dynamics code

The general structure of an MD code can be expressed as a simple flow chart. The general problem is that of solving the equations of motion for a given Lagrangian

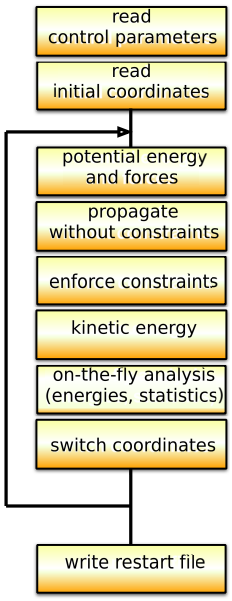
$$\mathcal{L}(\vec{x},\vec{v},t) = \frac{1}{2}\vec{v}\boldsymbol{M}\vec{v} - E_{pot}(\vec{x}) - \sum_j \lambda_j G_j(\vec{x}) \tag{9.1}$$

The constraints are described by the conditions $G_j(\vec{x}) = 0$, and they are enforced by the Lagrange parameters λ_j. The Euler-Lagrange equations are

$$\partial_t \vec{x} = \vec{v} \qquad \text{and} \qquad \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial v_i} = \frac{\partial \mathcal{L}}{\partial x_i} \tag{9.2}$$

$$\boldsymbol{M}\partial_t^2 \vec{x}(t) = \underbrace{-\vec{\nabla}E_{pot}(\vec{x})}_{\text{force } \vec{F}} \underbrace{- \sum_j \vec{\nabla}G_j(\vec{x})\lambda_j}_{\text{forces of constraint}} \tag{9.3}$$

This differential equation is solved iteratively using the Verlet algorithm.



1. read control parameters, such as timestep, frictions, choice of variables to be optimized, frozen or treated dynamically.

2. read initial coordinates: this contains static information, such as the general description of the system and the initial structure.

3. read restart file, if available. Otherwise use initial coordinates.

4. Potential energy and forces

$$F_i = \left.\frac{\partial E_{pot}(\vec{x})}{\partial x_i}\right|_{\vec{x}(0)}$$

5. propagate without constraints

$$\bar{\vec{x}} = \vec{x}(0)\frac{2}{1+a} + \vec{x}(-)\frac{1-a}{1+a} + \boldsymbol{M}^{-1}\vec{F}\frac{\Delta^2}{1+a}$$

6. enforce constraints

$$\vec{x}(+) = \bar{\vec{x}} - \sum_j \boldsymbol{M}^{-1}\vec{\nabla}G_i\lambda_i \qquad \text{with } \{\lambda_j\} \text{ determined from} \qquad G_j(\vec{x}(+)) = 0$$

7. kinetic energy

$$E_{kin} = \frac{1}{2}\frac{\vec{x}(+)-\vec{x}(-)}{2}\boldsymbol{M}\frac{\vec{x}(+)-\vec{x}(-)}{2}$$

8. switch coordinates

$$\vec{x}(0) \to \vec{x}(-) \qquad \text{and} \qquad \vec{x}(+) \to \vec{x}(0)$$

9. write restart file: The restart file contains all information about the dynamical state of the system. Usually it is two sets of variables, namely $\vec{x}(0)$ and $\vec{x}(-)$, and some of the Lagrange multipliers.

9.2 Types of objects in the CP-PAW code

While being written in Fortran, The CP-PAW code has an object oriented structure. This implies that a overall flow-chart is not the best way to describe the code. Rather there are parts of the code, objects, that act like more-or-less independent agents. These agents will be initially instructed with the information (the data) they need to perform a well defined set of tasks. It will then be triggered to perform certain operations in their field of duty.

In order to describe the structure of the code I am trying to categorize the objects, which is done below.

9.2.1 Dynamical objects

The dynamical objects listed here have the general structure described in section 9.1.

- `paw_atoms`
- `paw_waves`
- `paw_thermostat`
- `paw_cell`
- `paw_cosmo`
- `paw_classical` through the interface `paw_qmmm`
- `paw_occupations`
- `paw_ci`

Energy terms may also be attached to the interfaces

- `paw_isolate`
- `paw_lmto`

9.2.2 Control objects

- `paw_assist`
- `paw_extpot`
- `paw_optfric`
- `paw_vext`

9.2.3 Analysis objects

- `paw_graphics`
- `paw_efg`
- `paw_opteels`

9.2.4 I/O objects

- `paw_ioroutines`
- `paw_iotra`
- `paw_ionew`

9.3 The PAW library

While writing the CP-PAW code, a number of useful general-purpose tools accumulated and are used everywhere in the code.

9.3.1 Basic library objects

The most basic set objects in the paw library are the following. Anyone dealing with the code of CP-PAW must have a understanding of the following objects.

- `paw_library`: calls to external libraries are called trough an interface of the CP-PAW code of the form `call lib$`*function*(x,y,z...). The interfaces are collected in one place, namely `paw_library`. This makes it possible to replace the libraries without interfering with the code itself.
- `paw_filehandler`: Fortran files are managed by the file handler. Files are addressed through id's. Opening and closing files, as well as the choice of file units is done through the file handler.
- `paw_constants`: Units and conversion factors between constants provided through `paw_constants`. This ensures consistency of the conversion factors and enforces the use of constants from the official *CODATA* dataset `http://physics.nist.gov/cuu/index.html`.
- `paw_periodictable`: makes data available that can be attributed to the periodic table of the elements.
- `paw_error`: frequent consistency checks are done frequently. In order to enforce a common layout for error messages a set of helper routines supplies calls for reporting and for bringing down a code, which can be non-trivial on a parallel computer.
- `paw_trace`: trace calls can be used to report entering and leaving certain subroutines. The reporting can be switched on and off with a central call.

- `paw_timing`: clocks certain code sequences between breakpoints. clocks are identified by id's. A central report can be produced with one call.
- `paw_linkedlist`: manages a complex linked-list/tree structure, mostly used for handling input files with an xml-like format.
- `paw_strings`: Overloads the '+' and '-' sign for strings with the uppercase and lowercase functions respectively.

9.3.2 Specific library objects

More specific are the following library objects:

- `paw_clock`
- `paw_lock`
- `paw_specialfunctions`
- `paw_usage`
- `paw_dft`
- `paw_dftaddendum`
- `paw_spherical`
- `paw_polynom`
- `paw_generalpurpose`
- `paw_report`
- `paw_radial`
- `paw_schroedinger`
- `paw_atomlib`
- `paw_selftest`
- `paw_strcio`
- `paw_cell`
- `paw_pdos`
- `paw_dimer`
- `paw_lmtobasics`
- `paw_debug`
- `paw_brillouin`
- `paw_gaussian`

Part II

Computing

Chapter 10

Emacs Editor

EMACS [1] is an extremely powerful full screen editor to write text files. Furthermore, it also has additional inbuilt functionality such allowing to maneuver around in the directory structure and to manipulate it, execute commands directly from within Emacs. Most importantly it allows one to move around arbitrary rectangular areas in your file, which is very convenient to manipulate for example tables of atomic coordinates.

Emacs is free software and it runs on all systems I know of. Thus your investment of learning it will be preserved.

Emacs is driven by key-combinations. It takes a while until one is used to it, but after a while of practice, you will do them automatically without even remembering the key combinations. They enter what is called "muscle memory". Whatever I do in Emacs, I can do it until someone asks me how....

To get started with Emacs, I recommend the following tutorial

`http://xahlee.org/emacs/emacs.html`

Another useful page is

`http://www.math.uh.edu/~torok/math_6298/emacs/`

In the following chapter, I try to mention the most important steps.

10.0.1 Basic concepts and commands

Create a file and open it by typing the following on the command line.

`touch` *filename*
`emacs` *filename* `&`

The first command creates a file, if it is not there.[2] The `&` sign at the end of the second line make Emacs start in the background, so that you can use the command line after the Emacs window has opened.

Now, enter some text by normal typing. Continue to fill a few lines so that you can try some commands on them. You can move around in the text using the arrow keys.

On my computer using the function key together with the arrow lets me jump to the beginning of the line (fn-left), the end of the line (fn-right), up one page (fn-up) or one page down (fn-down). **Editor: Is this a general solution?**

The essence of Emacs is to understand the key combinations:

[1] Note that there is a very similar editor, namely "xemacs" which is almost identical, but not quite.
[2] If a file with the name is present, touch gives it a new modification date.

- The key combinations are written such as "C-x d", which means: first type the keys "Cntl" and *x* simultaneously, let them go, and then type "d". "C" stands for the key "Cntl". Whenever two keys are connected by a dash as in "C-x" it means that the two keys must be typed simultaneously. Without a dash, the previous keys are let go. On my computer it is the "Strg" key. Usually it is one of the special keys to the very left at the bottom of your keyboard.

- There is another special key named "M", which typically is the "Alt" key, also at the bottom left of the keyboard. It occurs most commonly in the key combination "M-x" which allows one to enter a command explicitly instead of typing a key combination. When typing a command you can use the autocompletion option by typing the spacebar at any time in the command. This will complete the command up to the point to which its continuation is unique.

- The third special key is the <ESC> key

Some Emacs concepts:

- "point" is the current position of the cursor

- "mark" is a position in a text that has been marked with command C-space

- "region" is the text between point and mark

- "buffer" is usually a file that has been opened in Emacs.

Now try the following commands on your text.

C-u	undo
C-g	cancel last command
C-x C-s	save file (do this often!)
C-x C-k	close file without saving
C-x C-c	close Emacs window
M-g g	Go to line
C-l	center page at cursor
Esc <	Beginning of file
Esc >	end of file
C-s	incremental search forward
C-r	incremental search backward
C-d	kill letter under the cursor (forward kill)
backspace	backward kill
C-k	Kill rest of line
M-!	open shell
C-space	mark the starting point for cut/copy
C-w	Kill region from last mark (C-space) to cursor
C-y	"Yank" (insert) previously killed region (C-w) or line (C-k)
C-x i	insert file
M-%	replace string (see below)
M-x occur	list all lines containing a selected string (see below)
C-x d	open Dired (directory editor) (see below)
M-x M-u	Make text in region uppercase
M-x M-l	Make text in region lowercase

10.0.2 Collect all occurrences of a string (M-x occur)

M-x occur: Type the M-key and x simultaneously and enter the command "occur" at the line that opens at the bottom of the window. Hit return and follow the invitation to enter the search string in the line at the bottom.

This will open a second frame in the window, showing all lines containing the search string, which is highlighted. Left-click on a line in the list, makes Emacs jump to that line in the original frame.

10.0.3 Replace strings (M-%)

Replace string (M-%). typing the M key and % simultaneously will open a line at the bottom inviting you to insert the string to be replaced, after entering this string it will invite you to insert the replacement string.

Then Emacs hops from occurrence to occurrence and you need to enter the following keys to decide on the action to be done.

y	(yes) replace instance
n	(no) skip instance (do nothing and continue to the next string)
q	quit
!	replace all remaining matches

10.0.4 Cut and Paste rectangles

One of the powerful features of Emacs is the ability to operate on rectangles in the text, which is very convenient to adjust and change column-based data files.

Go to the left upper corner and set a mark with "C-space". Then move to the right lower corner and type "C-x r k" to kill the rectangle. If you want to retain the rectangle use "C-x u" to undo the kill. The rectangle remains nevertheless in an intermediate buffer. You can place the rectangle at the position of the cursor by typing "C-x r y". The cursor position will be the left upper corner of the rectangle.

10.0.5 Dired

Basics: Moving around

First, we learn how to maneuver within your directory structure and how to select filed. "Dired" stands for "directory editor". It allows you to move around efficiently in your directory structure and also to manipulate it.

```
/home/myuid/Tree/PhiSX/Praktikum/Book/Chapters:
insgesamt 48
drwxr-xr-x  2 myuid users  4096 2006-11-26 12:35 .
drwxr-xr-x  4 myuid users  4096 2006-11-16 10:57 ..
-rw-r--r--  1 myuid users  6972 2006-11-12 14:29 carparrinello.tex
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 carparrinello.tex~
-rw-r--r--  1 myuid users  2449 2006-11-26 12:37 #computing.tex#
-rw-r--r--  1 myuid users  2227 2006-11-26 12:35 computing.tex
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 computing.tex~
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 dft.tex
-rw-r--r--  1 myuid users 23393 2006-11-12 11:42 kpoints.tex
-rw-r--r--  1 myuid users     0 2006-11-11 19:03 kpoints.tex~
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 mermin.tex
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 supercells.tex
```

```
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 thermostat.tex
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 tst.tex
-rw-r--r--  1 myuid users     0 2006-11-12 11:44 verlet.tex
```

At the very top, Emacs tells you where in your directory structure you are. Then follows a list of the files in the current directory At the top of the list you will find " .' and ".", which stands for current and the parent directory.

You will also see filed that start and end with a hatch such as `#computing.tex#` and those that end with a tilde such as "`computing.tex~`". They are special backup files of Emacs, which allow you to recover a recent version of the file.

You can now select a file or a directory with typing "f" at the corresponding line. If it is a file, the file opens and you are ready to modify it. We will not do that now, but move back into dired.

- Either you call the dired again by typing by C-x d which brings you back to the same place where you left dired, that is when you entered the file.
- or you move back into the previous buffer, typing C-x b. It may ask you in the bottom line

  ```
  Switch to buffer: (default Chapters)
  ```

  Chapters is the directory from where you entered into the file. Instead of "Chapters" there will be another name in your case. You confirm by typing the ENTER key. to return to the previous buffer.

One last command is very handy: By typing C-x B you will open a so-called buffer list. It is a list of all directories and files you have opened recently. You can select files as if you had opened a directory. Note that you may have opened tow different files with the same name, which however are located in different directories. The buffers are called the same but you will see an ending <2>, which tells you that this is the second buffer with this name. Behind the file the entire path of the file is listed so that you can exactly identify the file.

Advanced: changing

The dired allows you much more than just moving around efficiently. You can

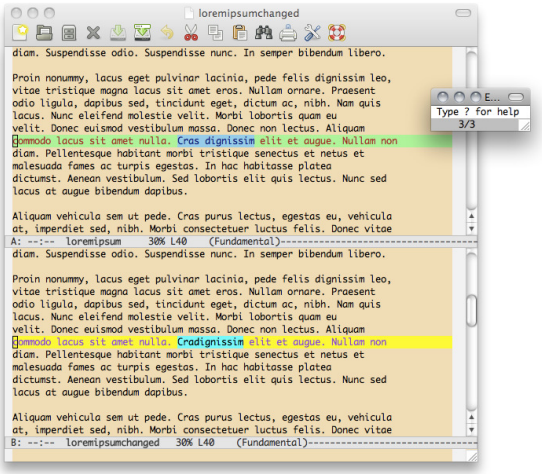
- rename files or move files and directories around
- copy files
- delete files

These options and more you will also see in the menu at the top of the Emacs window under operate.

10.1 Compare files with ediff

Emacs contains a convenient tool for comparing two files. It is accessed from the "Tools" pulldown menu. Select "Two files" and then select the two files to be compared.

When the two files open, also a tiny window comes up. In order to maneuver up and down the changes, type letters into this tiny window: Use "n" for next, "p" for previous and "q" for quit. When you quit, you will be asked at the bottom of the main window whether you want to accept or delete the changes.



If a certain change is selected you transfer the version from file A to file B by simply typing “a” into the small window. By typing “b” one selects the version from file B for file A.

10.2 .emacs file

The .emacs file is a profile. Its content allow to adjust the general behavior of Emacs. This section is not for the student, but rather for the administrator, who sets up the software for your computer.

Sometimes the .emacs file only redirects to another file, which plays the role of the profile for Emacs. This has been necessary on some linux systems to run Emacs and Xemacs simultaneously.

The double semicolon is a sign for comment.

10.2.1 My .emacs file

```
(setq enable-local-variables nil)
;;-------------------------------------------------------
(set 'printer-name' "Printer1")
(global-set-key [home] 'beginning-of-line)
(global-set-key [end] 'end-of-line)
(global-set-key [next] 'scroll-up)
(global-set-key [prior] 'scroll-down)
(global-set-key [up] 'previous-line)
(global-set-key [down] 'next-line)
(global-set-key [left] 'backward-char)
(global-set-key [right] 'forward-char)
(global-set-key [insert] 'overwrite-mode)
(global-set-key "\C-c\C-e" 'LaTeX-close-environment)

;;(global-set-key \C-d 'del-char)
;; =====================================================
;; set foreground and background color
;; =====================================================
(set-background-color "wheat")
;;(set-foreground-color "yellow")
;; =====================================================
;; Enable Highlighting
;; =====================================================
```

```
(cond (window-system
      ;;;; use preferably font-lock mode
      (require 'font-lock)
      (global-font-lock-mode t) ; works only in emacs20
      (add-hook 'bibtex-mode-hook '(lambda()
     (make-local-variable 'font-lock-maximum-size) (set font-lock-maximum-size 1000000)))
      ))
;;-------------------------------------------------------
;; allow to uppercase and lowercase regions
(put 'downcase-region 'disabled nil)
(put 'upcase-region 'disabled nil)


;;(global-set-key [end] 'compile)
;;(put 'downcase-region 'disabled nil)
;;(put 'upcase-region 'disabled nil)
;;(custom-set-variables
;; '(compilation-read-command nil)
;; '(compilation-ask-about-save nil)
;; '(compile-command "make -k ")
;; '(delete-selection-mode nil nil (delsel))
;; '(scroll-bar-mode (quote right)))
;;(custom-set-faces)

(setq ispell-program-name "/usr/local/bin/aspell")

;;(setq-default ispell-program-name "aspell")
;;(setq-default ispell-extra-args '("--reverse"))
```

10.2.2 Common-User-Access (CUA) mode

Emacs uses a number of cryptic commands for historical reasons such as different keyboard layouts. Today the majority of applications use a common standard, namely the "Common User access". CUA is for example standard in Windows.

C-x	kill
C-c	copy
C-v	paste
C-z	undo

Recently Emacs includes the possibility to use this standard by enabling CUA mode.

I have not yet adopted this convention because there are key-combinations in Emacs that collide with the CUA conventions:

Special caution for (Common User access) CUA commands: The meaning of C-x varies completely if there is highlighted text or not. The command C-x is the command for "delete highlighted text". It works that way only if there is highlighted text. C-x is also part of many Emacs commands. These commands do not work if there is highlighted text. This may be quite confusing. A region is highlighted for example by simply dragging the cursor. A highlighted region can also be created by C-space and moving the cursor with the arrow keys. In order to define a region without highlighting set the mark with C-space and the move the cursor with the mouse to the new position.

Chapter 11

Visualization

11.1 Xmgrace

Two-dimensional plots such as a function $y(x)$ are conveniently plotted by Xmgrace. The main advantage is that it is sufficiently flexible to create publication-ready graphs. Xmgrace also allows one to perform some elementary operation such as Fourier transforms, polynomial fits etc.

Xmgrace starts from a simple data file, where the data are written in column format such as

x1 y1(x1) y2(x1) y3(x1) ...
x2 y1(x2) y2(x2) y3(x2) ...
x3 y1(x3) y2(x3) y3(x3) ...
⋮

The program is started simply by

```
xmgrace -nxy infile.dat
```

The user guide can be found at
`http://plasma-gate.weizmann.ac.il/Grace/doc/UsersGuide.html`

11.1.1 Special symbols

(From `http://blog.louic.nl/?p=249`)

To type special symbols in a text field of Xmgrace, select the text field and type `ctrl-e`. This will bring up a front dialog box. Select the desired font, for example "symbol", and type the character, respectively string and accept to include it in the text field.

Otherwise you may use the following table

Switch to superscripts	\S
Switch to subscripts	\s
Switch back to normal font	\N
Switch to greek letters	\f{*Symbol*}
Switch back to latin letters	\f{}.
Å	\cE\C
°	\c0\C; \c:\C
±	\c1\C
^{2}	\c2\C
^{3}	\c3\C
$\frac{1}{2}$	\c=\C
$\frac{1}{4}$	\c<\C
μ	\c5\C

11.1.2 Settings

Xmgrace uses three special files, namely

`gracerc`	`templates/Defaults.agr`	`fonts/FontDataBase`

It searches the file names specified in the following order in the directories

(1)	(2)	(3)
`./`	`.grace/`	`~/.grace/$GRACE_HOME/`

relative to the current directory "`./`".

The following settings are not necessary. They are a matter of convenience.

- define the grace home directory `GRACE_HOME`.
  - see whether `GRACE_HOME` is set by typing "`echo $GRACE_HOME`" on the command line. If this does not help,
  - search for a directory `grace/templates` somewhere on your system. If this does not help,
  - ask your system administrator, who installed Xmgrace.

  If the environment variable `GRACE_HOME` has not been set, define it in your profile (e.g. `~.profile`, `~.bashrc`) by introducing a line

  ```
  export GRACE_HOME=...
  ```

- set an alias in your profile (e.g. `~.profile`, `~.bashrc`)

  ```
  alias xmgrace='xmgrace -free -noask'
  ```

  The parameter "-free" chooses the page size such that the graph fits into the window chosen. The parameter "-noask" avoids that Xmgrace asks for confirmation each time when you close a window.

- Set defaults: Create a directory `~/.grace/templates` in your home directory and copy the file `$GRACE_HOME/templates/Defaults.agr` into the new directory. This file can now be edited:
  - I personally like the font "Helvetica" better than "Times Roman". This can be changed by interchanging the numbers in the font setting.

```
@map font 0 to "Helvetica", "Helvetica"
@map font 1 to "Helvetica-Oblique", "Helvetica-Oblique"
@map font 2 to "Helvetica-Bold", "Helvetica-Bold"
@map font 3 to "Helvetica-BoldOblique", "Helvetica-BoldOblique"
@map font 4 to "Times-Roman", "Times-Roman"
@map font 5 to "Times-Italic", "Times-Italic"
@map font 6 to "Times-Bold", "Times-Bold"
@map font 7 to "Times-BoldItalic", "Times-BoldItalic"
```

– Next, change the colors by changing the numbers on them. Colors like yellow and grey are hardly visible, which is annoying. Therefore I give them a large number so that they are selected as the last colors.

```
@map color 0 to (255, 255, 255), "white"
@map color 1 to (0, 0, 0), "black"
@map color 2 to (255, 0, 0), "red"
@map color 3 to (0, 0, 255), "blue"
@map color 4 to (0, 139, 0), "green4"
@map color 5 to (255, 165, 0), "orange"
@map color 6 to (64, 224, 208), "turquoise"
@map color 7 to (0, 255, 0), "green"
@map color 8 to (255, 0, 255), "magenta"
@map color 9 to (0, 255, 255), "cyan"
@map color 10 to (148, 0, 211), "violet"
@map color 11 to (114, 33, 188), "indigo"
@map color 12 to (188, 143, 143), "brown"
@map color 13 to (103, 7, 72), "maroon"
@map color 14 to (220, 220, 220), "grey"
@map color 15 to (255, 255, 0), "yellow"
```

– increase the line width to 2.5: Change all occurrences of "`linewidth 1.0`" to "`linewidth 2.5`".

– increase the font and symbol size to 1.5: Change all occurrences of "`char size 1.0`" to "`char size 1.5`" and of "`symbol size 1.0`" to "`symbol size 1.5`" and

Once you understand the syntax, you can also do other adjustments and try the result by opening Xmgrace. You can also open Xmgrace, adjust the settings, save the session and inspect the corresponding file, which has the extension ".agr".

11.1.3 A handy script

By default, Xmgrace reads from a file only two columns, one for x and one for y. In order to make Xmgrace read several y-columns, one needs to prepend -nxy to each file such as

```
xmgrace -nxy file
```

For multiple files, this is annoying, in particular, when one wishes to process a number of files simultaneously using wild cards. In order to make this more convenient, I prepared a small script which I am adding here.

```
#!/bin/sh
# executes xmgrace with files for multiple dos files
#
#  mygrace a b c ...
#
#  xmgrace -nxy a.ddos -nxy b.ddos -nxy c.ddos ...
#
```

```
CMD='xmgrace -free -noask'
for X in $* ; do
  CMD=`echo $CMD -nxy $X`
done
$CMD
```

Copy these lines into a file, say "mygrace", make the file executable by `chmod +x mygrace`, and place it somewhere where your executables are found, such as your 'bin' directory.

Then you can plot all files ending with '.ddos' by

```
mygrace *.ddos
```

or you simply list the files

```
mygrace file1 file2 file3
```

and all will be read with the multicolumn option.

11.2 Gnuplot

11.2.1 Rubbersheet plots

If you wish to plot a surface, let us say $z(x, y)$, create an input file `example.dat` by simply writing the triples (x,y,z) each on one line. The choice of the (x,y) pairs and the order of the triples is not relevant, because the values will interpolated onto a new grid.

Now copy the following file into a file example.gnu and adjust the values in the data section.

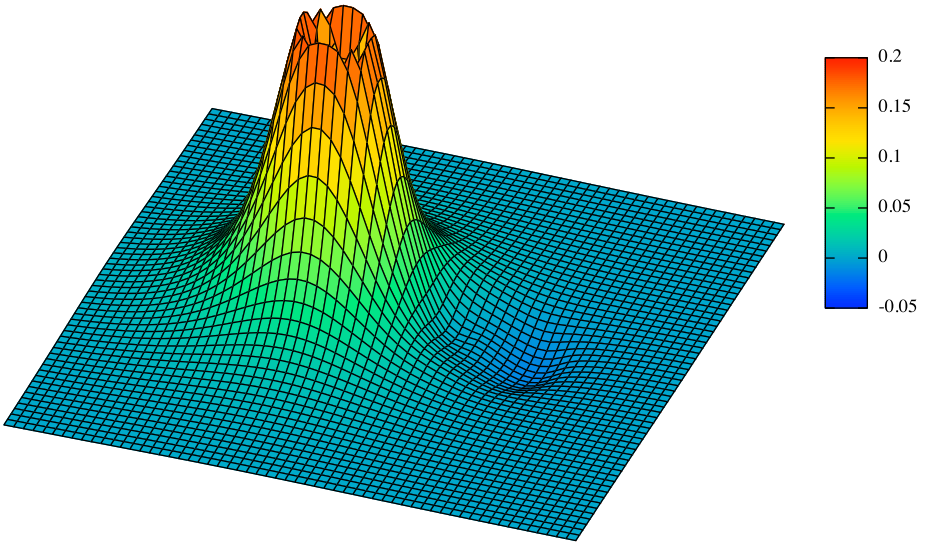
```
#
#================================================
# data section to be changed by the user
#================================================
xmin=-4.
xmax=6.
ymin=-5.
ymax=5.
zmin=-0.015
zmax=0.3
rot_x=30
rot_z=20
scale=1.0
scale_z=2.5
infile="example.dat"
#
#================================================
# define line styles to be used with ls in splot
#================================================
# define a linestyle for splot (used with ls)
set style line 1 lt 1 lc rgb "black" lw 1
# map hight values to colors
set palette rgbformula 22,13,-31
#
#=================================
# data related statements
#=================================
```

```
set xrange [xmin:xmax]
set yrange [ymin:ymax]
set zrange [zmin:zmax]
# sample input data onto a 60x60 grid
set dgrid3d  60,60,1
#
#==================================
# surface plot
#==================================
set surface
set hidden3d
set data style lines
# places the zero of the z-axis into the xy plane
set xyplane at 0.
# remove axes
unset border
#  remove tics from the axes
unset xtics
unset ytics
unset ztics
# no title written
set key off
# place contours onto the surface
set pm3d explicit hidden3d 1
#
#  angle, angle, overall scale, scale z-axis
set view rot_x,rot_z,scale,scale_z
#
splot  infile with pm3d
```

The batch file is executed with

```
gnuplot example.gnu
```

Depending on the input file the result could look like this



11.2.2 Contour plots

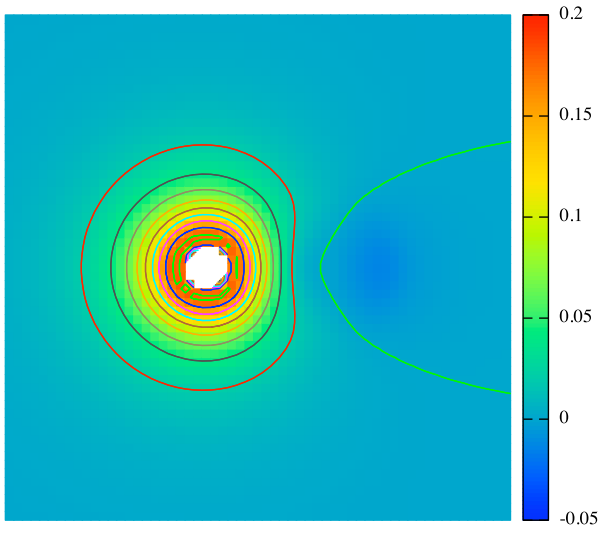
A variation of the rubbersheet plots just demonstrated are the contour plots, which are basically rubbersheets seen from above, while the grid is replaced by contours.

An input file is

```
#================================================
# data section to be changed by the user
#================================================
xmin=-4.
xmax=6.
ymin=-5.
ymax=5.
zmin=-0.015
zmax=0.3
ncontour=30
infile="example.dat"
#
#================================================
# define line styles to be used with ls in splot
#================================================
# define a linestyle for splot (used with ls)
set style line 1 lt 1 lc rgb "black" lw 1
set palette rgbformula 22,13,-31
#
#=================================
# data related statements
#=================================
```

```
set xrange [xmin:xmax]
set yrange [ymin:ymax]
set zrange [zmin:zmax]
# sample input data onto a 60x60 grid
set dgrid3d  60,60,1
set cntrparam levels auto ncontour
#
#==================================
# surface plot
#==================================
set surface
# remove axes
unset border
#  remove tics from the axes
unset xtics
unset ytics
unset ztics
# no title written
set key off
set contour both
set view map
set size square
splot  infile with pm3d
```

Depending on the input file the result could look like this



If you do not want to see the colored background, change the command “set surface” to “set nosurface”.

11.2.3 Terminals

In order to create the image in a particular format and to write it to a file use a command like

```
set terminal postscript
set output 'image.eps
```

Terminal accepts may different styles

- pdf (.pdf) portable document format developed by Adobe Systems. (According to Wiki, it is now an open standard.)
- latex
- png (.png) portable network graphics file. portable and free bitmap format that replaces the older and restricted gif format.
- povray (.pov) Ray tracing scene
- vrml (.wrl) Virtual Reality Modeling Language
- x11
- fig (.fig) input file for XFIG graphics editor

Appendix A

Fluctuations of the kinetic energy

Here we show, how the kinetic-energy fluctuations in a canonical ensemble are related to the number g is the number of vibrational degrees of freedom in an MD simulation,

$$\frac{\sqrt{\left\langle \left(E_{kin} - \langle E_{kin} \rangle \right)^2 \right\rangle}}{\left\langle E_{kin} \right\rangle} = \left(\frac{g}{2} \right)^{-\frac{1}{2}} . \tag{A.1}$$

This relation is useful to explore whether the simulation has reached thermal equilibrium. Before thermal equilibrium is reached, the fluctuations are usually larger than expected, because the number of actually excited degrees of freedom is smaller than the total number.

We represent the velocities of the g-degrees of freedom g-dimensional vector $\vec{v}$. The masses are represented by a mass tensor, which normally only contains diagonal elements.

The mean value of the kinetic energy is

$$\left\langle E_{kin} \right\rangle = \frac{\int d^g v \ \frac{1}{2} \vec{v} \boldsymbol{m} \vec{v} \mathrm{e}^{-\beta \frac{1}{2} \vec{v} \boldsymbol{m} \vec{v}}}{\int d^g v \ \mathrm{e}^{-\beta \frac{1}{2} \vec{v} \boldsymbol{m} \vec{v}}} = k_B T \frac{\int d^g x \ \sum_j x_j^2 \mathrm{e}^{-\sum_j x_j^2}}{\int d^g x \ \mathrm{e}^{-\sum_j x_j^2}} = g k_B T \frac{\int dx \ x^2 \mathrm{e}^{-x^2}}{\int dx \ \mathrm{e}^{-x^2}} \tag{A.2}$$

$$\begin{aligned} \left\langle E_{kin}^2 \right\rangle &= \frac{\int d^g v \ (\frac{1}{2} \vec{v} \boldsymbol{m} \vec{v})^2 \mathrm{e}^{-\beta \frac{1}{2} \vec{v} \boldsymbol{m} \vec{v}}}{\int d^g v \ \mathrm{e}^{-\beta \frac{1}{2} \vec{v} \boldsymbol{m} \vec{v}}} = (k_B T)^2 \frac{\int d^g x \ \sum_{i,j} x_i^2 x_j^2 \mathrm{e}^{-\sum_j x_j^2}}{\int d^g x \ \mathrm{e}^{-\sum_j x_j^2}} \\ &= (k_B T)^2 \left(g \frac{\int dx \ x^4 \mathrm{e}^{-x^2}}{\int dx \ \mathrm{e}^{-x^2}} + g(g-1) \left(\frac{\int dx \ x^2 \mathrm{e}^{-x^2}}{\int dx \ \mathrm{e}^{-x^2}} \right)^2 \right) \end{aligned} \tag{A.3}$$

Now, we use the expressions[139]

$$\int_{-\infty}^{\infty} dx \ \mathrm{e}^{-x^2} = \sqrt{\pi} \tag{A.4}$$

$$\int_{-\infty}^{\infty} dx \ x^2 \mathrm{e}^{-x^2} = \frac{1}{2} \sqrt{\pi} \tag{A.5}$$

$$\int_{-\infty}^{\infty} dx \ x^4 \mathrm{e}^{-x^2} = \frac{3}{4} \sqrt{\pi} \tag{A.6}$$

and obtain

$$\left\langle E_{kin} \right\rangle = \frac{1}{2} g k_B T \tag{A.7}$$

$$\left\langle E_{kin}^2 \right\rangle = (k_B T)^2 \left(g \frac{3}{4} + g(g-1) \frac{1}{4} \right) = \left(\frac{1}{2} g k_B T \right)^2 + \frac{1}{2} g (k_B T)^2 \tag{A.8}$$

Thus we obtain

$$\frac{\sqrt{\left\langle \left(E_{kin} - \langle E_{kin} \rangle \right)^2 \right\rangle}}{\left\langle E_{kin} \right\rangle} = \frac{\sqrt{\left\langle E_{kin}^2 \right\rangle - \left\langle E_{kin} \right\rangle^2}}{\left\langle E_{kin} \right\rangle} = \frac{\sqrt{\frac{1}{2} g (k_B T)^2}}{\frac{1}{2} g k_B T} = \left(\frac{g}{2} \right)^{-\frac{1}{2}} \tag{A.9}$$

which proves Eq. A.1.

Appendix B

Extract frequencies from a MD simulation

An simple way to analyze the frequency spectrum of a simple trajectory $x(t)$, such as a bond-length, is to measure the time delay between repeated zeros. Note, that one first has to ensure by inspection that the trajectory is suitable for such an analysis.

The following is the code `freqextract.f90`.

```
program main
implicit none
real(8)   :: xm
real(8)   :: x
real(8)   :: tm
real(8)   :: t
real(8)   :: f
real(8)   :: pi
pi=4.d0*atan(1.d0)
xm=0.d0
tm=0.d0
do
  read(*,*)t,x
  if(x*xm.lt.0.d0) then
    f=2.d0*pi/(t-tm)
    write(*,*) t,f
    tm=t
  end if
  xm=x
enddo
stop
end
```

Appendix C

Tips and tricks on the computer

C.1 Floating point operations in a shell

The bash shell per se can only do integer mathematics but does not know of floating point operations. However one can use the internal arbitrary precision calculator "bc".

For example if have defined an equilibrium lattice constant ALAT0 and want to scan a number of smaller and larger lattice constants you can do the following

```
#!/bin/sh
ALAT0=10.3976
for X in 96 97 98 99 100 101 102 103 104; do
  ALAT=‘ echo "$X / 100 * $ALAT0 " |bc -l ‘
  echo $ALAT
done
```

Note the ALAT0, ALAT, and X are simple string variables. Ensure that you are using the correct apostrophe which is leaning backward. It differs from the one used to enclose text.

This tip has been found at `http://phoxis.org/2009/12/23/floatmathbash/`. For further a description of `bc`, see `http://www.gnu.org/software/bc/manual/html_mono/bc.html`.

C.2 create temporary files in a bash script

The command `mktemp` creates a file and returns the name of the file to standard out. It takes as argument a model for the file name, which ends with a sequence of X's. The X's are replaced by a unique string.

```
export TMPFILE=$(mktemp /tmp/tmp.XXXXXX)
```

The resulting file name is is kept in the variable TMPFILE

C.3 Process options of a shell script

`getopts` is a method to scan through a set of options of a shell script in a professional manner. Below I provide a simple example of using it. One needs to familiarize onself with getops by using one of the many tutorials on getopts on the internet.

The following variables have fixed meanings and are processed in a hidden manner.

- `OPTIND` is the index to the next argument.

- OPTARG is an argument to an option

```
while getopts ":a:b" OPT ; do
 case $OPT in
    a) echo "this is option a with argument $OPTARG"
       case $OPTARG in
         aa) echo "argument aa to option a"
         ab) echo "argument ab to option a"
         *) echo "error in $0: argument $OPTARG not recognized" >&2
            exit 1
            ;;
       esac
       ;;
       ;;
    b) echo "this is option b"
       ;;
    \?) # unknown option
       echo "error in $0: invalid option -$OPTARG" >&2
       echo "retrieve argument list with:" >&2
       echo "$0 -h" >&2
       exit 1
       ;;
    :)    # no argument passed to option requiring one
       echo "error in $0" >&2
       echo "option -$OPTARG requires an additional argument" >&2
       exit 1
       ;;
  esac
done
shift $(($OPTIND - 1))
```

Appendix D

Frozen-core Approximation

D.1 Introduction

The frozen-core approximation has two purposes. First, it reduces the dimensions of the algebraic operations, such as diagonalizations, matrix products,etc to the number of valence electrons. Secondly, it reduces rounding errors by reducing the overall size of the total energy. The energy of the valence electrons are only a small fraction of the energy of the core electrons. Finally, by freezing degrees of freedom, on which the energy depends on sensitvely, the stability of the calculations is improved.

The frozen core approximation is discussed by von Barth and Gelatt.[140].

D.2 Energy functional in the frozen core approximation

Let me start with the total energy of density functional theory. The density functional (in curly brackets) is expressed in terms of the Kohn-Sham wave functions $|\varphi_n\rangle$ and their occupations.

$$E_{tot} = \min_{|\varphi_n\rangle, f_n} \operatorname*{stat}_{\mathbf{\Lambda},\mu} \Bigg\{ \sum_n f_n \langle\varphi_n| \frac{\hat{\vec{p}}^2}{2m_e} |\varphi_n\rangle + E_{pot}[\hat{\rho}^{(1)}] - \sum_{m,n} \Lambda_{m,n} \Big(\langle\varphi_n|\varphi_m\rangle - \delta_{n,m} \Big) - \mu \Big(\sum_n f_n - N \Big) \Bigg\} \tag{D.1}$$

The one-particle reduced density matrix of the non-interacting reference system is obtained from the Kohn-Sham wave functions and their occupations as

$$\hat{\rho}^{(1)} = \sum_n |\varphi_n\rangle f_n \langle\varphi_n| \; . \tag{D.2}$$

The potential energy can be written in the form

$$E_{pot} = E_H + E_{xc} \tag{D.3}$$

I use here the one-particle reduced density matrix rather than the density, because it allows me to also consider hybrid functionals, that use the (statically) screened exchange term of the form

$$E_{\text{scrX}} = \frac{1}{2} \int d^4x \int d^4x' \; \frac{e^2 \rho^{(1)}(\vec{x}, \vec{x'}) \rho^{(1)}(\vec{x'}, \vec{x})}{4\pi\epsilon_0 \epsilon_r (|\vec{r} - \vec{r'}|) |\vec{r} - \vec{r'}|} \tag{D.4}$$

The position $\vec{r}$ and the spin index σ of the electron are combined in the composite index $\vec{x} = (\vec{r}, \sigma)$. The four-dimensional integral is a short hand for $\int d^4x = \sum_{\sigma \in \{\uparrow,\downarrow\}} \int d^3r$.

The potential energy contains the Hartree energy

$$E_H = \frac{1}{2}\int d^3r \int d^3r' \, \frac{e^2\Big(n(\vec{r}) + Z(\vec{r})\Big)\Big(n(\vec{r'}) + Z(\vec{r'})\Big)}{4\pi\epsilon_0|\vec{r} - \vec{r'}|} \tag{D.5}$$

which depends on the electron density, the exchange correlation energy

$$E_{xc} = E_{xc}[n] \tag{D.6}$$

and a double-counting term

$$E_{dc} = E_{dc}[n] \tag{D.7}$$

as counterpart if the screened exchange energy

$$n(\vec{r}) = \sum_{\sigma\in\{\uparrow,\downarrow\}} \rho^{(1)}(\vec{x},\vec{x}) \tag{D.8}$$

and the charge density $-eZ(\vec{r})$ of the nuclei[1]. We use the composite space-and-spin coordinate $\vec{x} = (\vec{r},\sigma)$ mentioned above.

The density $\vec{Z}(\vec{r}) = -Z_R\delta(\vec{r} - \vec{R})$ is defined by the positions $\vec{R}$ of the nuclei and their atomic number Z_R. (A more explicit notation would be to use $\vec{R}_R$ rather than $\vec{R}$.)

D.2.1 Basissets

The minimization of the density functional is typically constrained to a basiset $\{|\chi_\alpha\rangle\}$, which gives the Kohn-Sham wave functions as

$$|\varphi_n\rangle = \sum_\alpha |\chi_\alpha\rangle c_{\alpha,n} \, . \tag{D.9}$$

In the PAW method, the basis functions are the projector-augmented plane waves, and the frozen core states. This means that the index α represents a wave vector (Bloch vector $\vec{k}$ plus reciprocal lattice vector) and a spin index.

The wave function coefficients $c_{\alpha,n}$ are optimized to obtain the energy minimum while satisfying orthonormality of the wave functions.

$$\begin{aligned} E_{tot} = \min_{c_{\alpha,n},\vec{f}_n} \operatorname*{stat}_{\mathbf{\Lambda},\mu} \Bigg\{ & \sum_n f_n\Big(\sum_{\alpha,\beta} c^*_{\alpha,n} T_{\alpha,\beta} c_{\alpha,n}\Big) + E_{pot}[\hat{\rho}^{(1)}] \\ & - \sum_{m,n} \Lambda_{m,n}\Big(\Big(\sum_{\alpha,\beta} c^*_{\alpha,n} O_{\alpha,\beta} c_{\beta,n}\Big) - \delta_{n,m}\Big) - \mu\Big(\sum_n f_n - N\Big)\Bigg\} \end{aligned} \tag{D.10}$$

The one-particle reduced density matrix has the form

$$\hat{\rho}^{(1)} = \sum_{\alpha,\beta} |\chi_\alpha\rangle\Big(\sum_n c_{\alpha,n} f_n c^*_{\beta,n}\Big)\langle\chi_\beta| \tag{D.11}$$

The matrix elements of kinetic energy and overlap are

$$T_{\alpha,\beta} = \langle\chi_\alpha|\frac{\hat{\vec{p}}^2}{2m_e}|\chi_\beta\rangle \qquad \text{and} \qquad O_{\alpha,\beta} = \langle\chi_\alpha|\chi_\beta\rangle \tag{D.12}$$

[1]The minus sign seems incorrect. However, $Z(\vec{r})$ is defined such that it is negative, so that the related change density is positive again. The underlying idea is that $Z(\vec{r})$ is considered as equivalent electron-number density.

D.2.2 Restrictions of the frozen-core approximation

In the frozen core approximation, the basis functions for the core states are the core states of an isolated atom or ion. These states are, per construction, orthonormal among each other.

$$\langle\chi_\alpha|\chi_\beta\rangle = \delta_{\alpha,\beta} \quad \text{for } \alpha,\beta \in C_R \tag{D.13}$$

Furthermore the orbitals for the valence states are orthogonalized to the core states, i.e.

$$|\chi_\alpha\rangle = |\chi_\alpha^{(0)}\rangle - \sum_{\beta\in C} |\chi_\beta\rangle\langle\chi_\beta|\chi_\alpha^{(0)}\rangle \quad \text{for } \alpha \in V \tag{D.14}$$

Because of the requirement that valence and core wave functions are orthonormal, the expansion of the valence wave function is limited to core-orthogonalized valence basis functions

$$|\varphi_n\rangle = \sum_{\alpha\in V} |\chi_\alpha\rangle c_{\alpha,n} \quad \text{for } n \in V \tag{D.15}$$

Any admixture of the core states would violate the orthogonality with the core states and is prohibited. This results in a considerable simplification of the expressions, because the effective Hamiltonian becomes block diagonal in the valence and core contributions.

Let me now define the core energy as

$$E_{core} = \sum_R \Big\{ \sum_{\alpha\in C_R} \langle\chi_\alpha|\frac{\hat{\vec{p}}^2}{2m_e}|\chi_\alpha\rangle + E_{pot}[\hat{\rho}^{(1)}_{core,R}] \Big\} \tag{D.16}$$

with the one-particle reduced density matrix of the (frozen) core states

$$\hat{\rho}^{(1)}_{core,R} = \sum_{\alpha\in C_R} |\chi_\alpha\rangle\langle\chi_\alpha| \tag{D.17}$$

The core energy does not represent any physically meaningful quantity. Rather it represents a constant, that does not change when atoms move or react. Thus it can be subtracted from the total energy functional without affecting physical quantities. Doing so cancels the kinetic-energy contribution of the core electrons.

D.2.3 Energy functional in the frozen-core approximation

The valence energy can now be composed as

$$\begin{aligned} E_{tot,val} &= E_{tot} - E_{core} \\ &= \min_{c_{\alpha,n},f_n} \operatorname*{stat}_{\mathbf{\Lambda},\mu} \Big\{ \sum_{n\in V} f_n \Big(\sum_{\alpha,\beta\in V} c^*_{\alpha,n} T_{\alpha,\beta} c_{\alpha,n} \Big) + E_{pot}[\hat{\rho}^{(1)}] - \sum_R E_{pot,R}[\hat{\rho}^{(1)}_{core,R}] \\ &\qquad - \sum_{m,n\in V} \Lambda_{m,n} \Big[\Big(\sum_{\alpha,\beta\in V} c^*_{\alpha,n} O_{\alpha,\beta} c_{\beta,n} \Big) - \delta_{n,m} \Big] - \mu \Big[\sum_{n\in V} f_n - \Big(N - \sum_R N_{core,R} \Big) \Big] \Big\} \\ \hat{\rho}^{(1)} &= \sum_{\alpha,\beta\in V} |\chi_\alpha\rangle \Big(\sum_{n\in V} c_{\alpha,n} f_n c^*_{\beta,n} \Big) \langle\chi_\beta| + \hat{\rho}^{(1)}_{core} \end{aligned} \tag{D.18}$$

This functional only depends on the wave function components of the valence electrons, while those of the core electrons are frozen.

This functional will always produce an upper bound (after adding the core energies in again), because we evaluate the correct functional, but only constrain the minimization.

D.2.4 Schrödinger equation

The conditions for the minimum is that the derivatives of the functional vanish. The exception is where the minimum lies at the boundary of the domain of allowed values for the functional. This is the case for the occupations, which are restricted to lie between zero and one.

Kohn-Sham potential

Before we continue let me define the Kohn-Sham potential

$$\hat{V}_{KS} = \frac{\delta E_{pot}}{\delta \hat{\rho}^{(1)}} \tag{D.19}$$

This term can be rationalized by writing the first variation of the potential energy as

$$\begin{aligned} E_{pot}\Big[\hat{\rho}^{(1)} + \delta\hat{\rho}^{(1)}\Big)] &= E_{pot}[\hat{\rho}^{(1)}] + \mathrm{Tr}\left[\frac{\delta E_{pot}}{\delta \hat{\rho}^{(1)}}\delta\hat{\rho}^{(1)}\right] + O\Big(\Big(\delta\hat{\rho}^{(1)}\Big)^2\Big) \\ &= E_{pot}[\hat{\rho}^{(1)}] + \mathrm{Tr}\Big[\hat{V}_{KS}\delta\hat{\rho}^{(1)}\Big] + O\Big(\Big(\delta\hat{\rho}^{(1)}\Big)^2\Big) \end{aligned} \tag{D.20}$$

For a density functional, the Kohn Sham potential has the form

$$\hat{V}_{KS} = \sum_{\alpha,\beta} |\pi_\alpha\rangle \Big\{ \sum_{\sigma,\sigma'\in\{\uparrow,\downarrow\}} \int d^3r\, \chi^*_\alpha(\vec{r},\sigma)\Big(v_H(\vec{r})\delta_{\sigma,\sigma'} + v_{xc}(\vec{r},\sigma,\sigma')\chi_\beta(\vec{r},\sigma')\Big\}\langle\pi_\beta| \tag{D.21}$$

where $\langle\pi_\alpha|$ are the projector functions for the basis orbitals that obey the bi-orthogonality condition

$$\langle\pi_\alpha|\chi_\beta\rangle = \delta_{\alpha,\beta} \tag{D.22}$$

$v_H(\vec{r}) = \frac{\delta E_H}{\delta n(\vec{r})}$ is the Hartree potential and $v_{xc}(\vec{r},\sigma,\sigma') = \frac{\delta E_H}{\delta n(\vec{r},\sigma',\sigma)}$ is the exchange-correlation potential. (I have generalized the density to include also the spin dependence.)

Kohn-Sham equations

The conditions for the minimization, I call equilibrium conditions, while the others, for which the optimum is a stationary point, are the constraint conditions.

The equilibrium conditions are

$$\begin{aligned} 0 &= \frac{1}{f_n}\frac{\delta E}{\delta c^*_{\alpha,n}} = T_{\alpha,\beta}c_{\beta,n} + \langle\chi_\alpha|\hat{V}_{KS}|\chi_\beta\rangle c_{\beta,n} - \sum_{m\in V} O_{\alpha,\beta}c_{\beta,m}\Lambda_{m,n} \\ 0 &= \frac{\delta E}{\delta f_n} = \sum_{\alpha,\beta\in V} c^*_{\alpha,n}T_{\alpha,\beta}c_{\beta,n} + c^*_{\alpha,n}\langle\chi_\alpha|\hat{V}_{KS}|\chi_\beta\rangle c_{\beta,n} - \mu \end{aligned} \tag{D.23}$$

The last equation only holds for partially occupied orbitals, but not for integer occupations. This implies that all states below the fermi level are filled and those above are empty.

D.2.5 Forces in the frozen-core approximation

It is important to calculate the forces consistent with the frozen-core approximation. The force is given (up to the sign) by the partial derivative of the energy functional.

$$\delta E = \sum_j \Big\{ \underbrace{\frac{\partial E}{\delta R_j}}_{-F^{(Z)}}\delta R_j + \underbrace{\mathrm{Tr}\Big[\frac{\partial E}{\partial\hat{\rho}^{(1)}}\frac{\delta\hat{\rho}^{(1)}}{\delta R_j}\Big]}_{-F^{(C)}_{R_j}-F^{(V)}_{R_j}}\delta R_j \Big\} \tag{D.24}$$

The first term describes the force acting on the nucleus.

$$\begin{aligned} F_j^{(Z)} &= -\int d^3r\; v_H(\vec{r})\frac{\delta Z(\vec{r})}{\delta R_j} = -\int d^3r\; v_H(\vec{r})\Big(-Z_{R_j}\vec{\nabla}_{R_j}\delta(\vec{r}-\vec{R}_j)\Big) \\ &= -\int d^3r\; v_H(\vec{r})\Big(+Z_{R_j}\vec{\nabla}_r\delta(\vec{r}-\vec{R}_j)\Big) = +Z_{R_j}\int d^3r\;\delta(\vec{r}-\vec{R}_j)\Big)\vec{\nabla}_r v_H(\vec{r}) \\ &= +eZ_{R_j}\underbrace{\vec{\nabla}_{R_j}\frac{1}{e}v_H(\vec{R}_j)}_{-\vec{E}(\vec{R}_j)} \end{aligned} \tag{D.25}$$

which is related to the electric field $\vec{E}$ at the nucleus times the nuclear charge density.

The second term describes the force acting on the electrons. It requires the response function for the electron density due to a change of the atomic positions. However, when the equilibrium is reached, that is, when the Kohn-Sham equations are obeyed, we can exploit that the forces acting on the electrons vanish when the equilibrium condition are obeyed. Thus, it is not necessary to evaluate the response function. This is what is called the Hellmann-Feynman theorem.

In the frozen-core approximation, however, only the equilibrium conditions for the valence wave functions are obeyed. The forces on the core electrons do not vanish and must be taken into account. This is however, unproblematic as the nuclear density is tied to the atomic position and it is independent of the valence electrons.

$$F_j^{(C)} = -\int d^3r\; v_H(\vec{r})\frac{\delta n^C(\vec{r})}{\delta R_j} = -\int d^3r\; n_{R_j}^{(C)}(\vec{r})\vec{\nabla}_r v_H(\vec{r}) \tag{D.26}$$

The force F_j^V acting on the valence electrons does not vanish when a finite basisset is employed. The minimum is obtained only for the wave function components $c_{\alpha,n}$ not for the entire Kohn-Sham wave function. When the basisset depends on the atomic positions, this derivative must be taken into account. These are the so-called **Pulay forces**. The force acting on the core electrons can actually also be described as a special kind of Pulay force. In the PAW method, the augmentation depends on the atomic positions and the corresponding derivatives are taken into account.[2]

D.3 Errors of the total energy due to the frozen-core approximation

Because of the variational princicple, the energy in the frozen-core approximation deviates only in second order in the difference of the density obtained in the frozen core approximation from the density at the exact minimum. This is probably the main reason for the good quality of the frozen-core approximation.

It is important that this principle holds only for the total energy, but not for individual contributions.

An important aspect is that the density is invariant with respect to a unitary transformation of the occupied Kohn-Sham states among each other. The frozen-core approximation does not make the assumption, that the individual states are frozen. Rather, the Kohn-Sham wave functions in the frozen-core approximations must be obtained after diagonalizing the Kohn-Sham Hamiltonian calculated from the frozen-core states and the filled valence states obtained from the frozen-core approximation. The comparison of frozen atomic wave functions with those of a crystal is poor. While they may have similar shape, their contribution to the total energy is very different.

Editor: Commented out is an incomplete analysis of the errors.

[2] They are not immediately evident, because many of the terms are used to cancel forces acting on the nucleus and the core density.

D.4 Individual core levels

D.4.1 Core-levels

The frozen core approximation does not imply that the core states are individually frozen. Two sets of core orbitals, which are related by a unitary transformation produce the same one-particle reduced density matrix. Thus such a unitary transformation does neither affect the density, nore the total energy.

We can use this flexibility to obtain individual core states.

$$\sum_{\beta\in C}\left[T_{\alpha,\beta}+\langle\chi_\alpha|\hat{V}_H|\chi_\beta\rangle-\epsilon\delta_{\alpha,\beta}\right]c_{\beta,n}=0\quad\text{for } n\in C \tag{D.27}$$

This shows, that, even in the frozen-core approximation, the core wave functions can deviate grossly from the atomic core wave functions from which they are constructed. Therefore, a comparison of individual energy levels between a frozen core and one with a relaxed core, does injustice to the frozen core approximation

D.4.2 Hyperfine parameters

Tere are cases where the frozen-core approximation has a considerable impact on the results. Core-polarization effects are dominant for the calculation of hyperfine parameters for systems where the s-orbital is not present in the spin density. This has been investigated in the group of Ursula Roethlisberger.[141]

Part III

Additions 1

Appendix E

Transition states: SN_2 Reaction

Lernziele: Zwangsbedingungen, Ubergangszustandstheorie, Reaktionsraten Bestimmung von Übergangszuständen und Aktivierungsenergien.

E.1 Retro Diels-Alder Reaktion von 2-Azanorbornen

Appendix F

Point defects and super cells

Rubin (oktahedrisches Cr^{3+} in Al2O3)

Supercells and interaction of defects

In a supercell calculation the concentration of defects often has unrealistically large values. Nevertheless we often claim to simulate isolated defects or defects in the limit of low concentrations.

The question is related to the interaction of defects. In a supercell that is sufficiently large, the separation of defects is so large that their interaction is negligible. Thus we effectively obtain an array of isolated defects. The nature of one such defect is the same as that of an isolated defect in an infinite, otherwise perfect host material.

In contrast to a supercell calculation, defects in a real material may wander around and come close to another defect. At this point the interaction between defects becomes important. If defects attract each other, they may form clusters unless they are driven apart from each other at finite temperatures by the larger entropy of individual defects. In this way the interaction between defects is responsible for their concentration dependent effects. However this interaction is absent in a proper supercell calculation. In order to study concentration dependent effects we would not only study individual defects, but also defect pairs or more complex clusters in a supercell calculation.

The caveat in this argument is however the requirement of a sufficient distance between defects. This means that we need to understand the range of differnet interactions.

- Electronic interactions extend typically to a distance of 1 nm. Thus supercells of a cube-nanometer are the typical choice. Note however that there are exceptions where there are long-ranged interactions in certain directions, as has been noticed for defects in silicon.
- the Coulomb interaction of charged defects is long ranged. This implies that the self energy of the charge distribution is infinite. In our calculations, a homogeneous charge background is automatically added, so that the total charge of the system vanishes. Corrections for charged unit cells have been developed for example by Blöchl for isolated molecules[142, 126] and for solids[142, 126] and by Makov and Payne[143].
- Elastic interactions can be long ranged. They may be investigated by investigating the defect under isotropic expansions or anisotropic strain. Usually elastic interactions can be ignored.

In case of doubt, it is necessary to explore how the results depend on the size of the supercell.

Defect states

Usually supercell calculations are performed with a single k-point. Rather than adding k-points it may often be advantageous to increase the supercell.

However, if we consider the band structure of molecules in a supercell calculation or of a defect level in the band gap, we often find a substantial dispersion. That is the band, which ideally is just a single energy level, has a certain band width. The band width is due to the defect that the repeated defects still form bonding and antibonding states, which may be several electron volt apart. The "ideal" position of the defect level would be in the center of the band, that is between bonding and antibonding orbitals.

The finite dispersion of defect levels in a supercell calculation leads to some situations that may not be obvious to handle.

If a defect-level is partially occupied it is important to occupy the state for each k-point equally. Filling only the lowest states, as in a metal, would describe a bond between the defects that is an artifact of the calculation.

A difficult situation appears, if the band overlaps with the valence or with the conduction band. Filling such a state equally for all k-points may lead to the situation that we occupy states from the conduction band, which may fall below the defect level, that crossed into the conduction state. That implies that we occupy a qualitatively different state. Only a careful analysis or special method will lead to a unique answer in this case.

Appendix G

Surfaces

Note for the Author: exercise with capping hydrogen atoms with fractional charge. Set file with it !CNTL!FILES!FILE:ID='PARMS_STP'. The example is H_3/4, which is a capping hydrogen with a charge of 3/4 e. [1]

G.1 Learning goals

- Learn how to treat surfaces or thin films with PAW.
- Understand the subdivision of the k-points into $k_{\|}$ and k_{normal} (or shorter k_z).
- Understand convergence with number of k-points.
- Understand the development of bulk behavior with the number of layers.
  Due to reduced coordination number, the band width is reduced, and
  for magnetic moments μ_B we get:$\mu_B(atom) > \mu_B(monolayer)) > \mu_B(surfacelayer) > \mu_B(bulk)$.
- Learn how to treat surface relaxation.
- Learn how to treat surface reconstruction.
- Understand why for polar systems capping hydrogen atoms with fractional charge are necessary.

G.2 Simulation of Surfaces

The description of the surface of a bulk crystal would include a half space, which is not treatable with PAW or other ab-initio electronic-structure methods. We simulate this system with a number of layers of the respective bulk planes parallel to the surface. This is the so-called slab-geometry (sometimes also called thin films). The more layers are included, the more the slab should resemble the truncated bulk system.
We now have a system with periodicity only in the plane parallel to the surface, which we choose as the x-y-plane spanned by the first two lattice vectors in the `strc`-file. The normal component of the k-vector is therefore naturally denoted by k_z.

The slab electronic wavefunctions are accordingly classified by a two-dimensional k ($=k_{\|}$) of a two-dimensional Brillouin-zone. If we add more and more parallel layers, the dispersion $E(k_{\|})$ should become a projection of the normal direction of the bulk bandstructure $E(k_{\|}, k_z)$ onto that $k_{\|}$. An

[1] FiXme Note: I have no experience with "capping hydrogen atoms with fractional charge"

example would be, that for a (001)-fcc surface all bulk $E(\vec{k} = 0, 0, k_z)$-values are projected to $k_{\|} = (0, 0)$ (now called $\bar{\Gamma}$), $0 \leq k_z \leq 1$.

Obviously, a slab has always two surfaces which should not interact directly or via periodic images. Thus, additionally to the slab thickness we have to introduce a large enough vacuum region between the periodic images normal to the slab. The third lattice vector in the `strc`-file according to the atomic positions within the unit cell has to be chosen in that way, that there exists a vacuum region of about 20 Bohr radii. This is an approved value for most systems. To determine the distance necessary one would have to do several test calculations with different lattice vectors normal to the surface till the results getting independent of the vacuum length. Another possibility is to do a calculation with two inequivalent k_z values. If the respective energies for these k-points do not differ, the vacuum is chosen large enough. Note however, that this should only be true for self-consistent results. A further restriction: You have to compare only occupied and deep lying exited states. Higher states have energies above the vacuum potential and are non-localized, thus the Bloch-condition for the respective k_z may lead to different eigenvalues.

G.3 Surface Relaxation and Reconstruction

The positions of the atoms within a crystal are determined by the forces from all the other atoms, especially from those in the neighborhood. On surfaces, the neighborhood changes drastically, the resulting forces may drive the atoms to other positions compared to those of an ideal cleavage of a crystal. If the symmetry parallel to the surface is kept and only the interlayer distance between the surface layers are changing, then this effect is called "surface relaxation". We will study that effect for a 7-layer aluminum 001-slab.

If there is also a lateral movement of the atoms associated, which may result in a smaller and/or larger spacing of the surface atoms or even in a symmetry change of the surface layers, then this behavior is referred to as "surface reconstruction". Both effects may also be induced by adsorption of other atoms.

The Wood notations for simple superstructures is as follows:
The 2-dimensional basis vectors of the reconstructed surface are given in multiples of the original lattice. If there is also a rotation with respect to the original lattice., then "R" (for **r**otated) and a rotation angle are also given. "p" means "**p**rimitive unit cell", "c" stands for "**c**entered" and denotes, that the cell center is equivalent to the corners.
Examples:

- (1×1) no superstructure
- $p(2 \times 1)$ the first lattice vector of the superstructure is twice as large as that of the original lattice, the second lattice vector equals the original one
- $(\sqrt{2} \times \sqrt{2})R45^\circ$

[2] One remark: for metals, the "healing effect" is strong, i.e., about three layers below the surface, the electronic density resembles very much that of the bulk, if one has only to do with surface relaxation.

G.4 Clean Al (001) surface

We will start with aluminum(001)- slabs with different number of layers to get some feeling on treating surface structures. Since here we only have to do with surface relaxation, we have to take

[2] FiXme Note: Figures necessary

one atom per layer (p(1 × 1) within the unit cell (p(1 × 1)).
Let's start with a **monolayer**. Here, is an example `strc`-file:

```
!STRUCTURE
 !GENERIC LUNIT=7.6534    !END
 !KPOINTS DIV=9 9 2      !END
 !OCCUPATIONS EMPTY=5 NSPIN=1 SPIN[HBAR]=0.0   !END
 !LATTICE T= 0.5 0.5  0.
            -0.5 0.5 0.
             0.0 0.0 5. !END
!SPECIES NAME='Al' NPRO=2 2 1 LRHOX=2 ID='AL_HBS' !END
 !ATOM  NAME= 'Al1'  R= 0.0 0.0 0.0       !END
!END
!EOB
```

Tasks:

- Do a paw-calculation with that `cntl`-file. Compare the energies for k-points with the same $k_{\|}$. What is your conclusion?
- Change the vacuum length to about 20 Bohr radii.
  Do the shortest test to prove, that this (nearly) prevents the periodic images to interact.
- Now do a self-consistent calculation for that monolayer.

Since we want to follow the growing of bulk behavior with the number of layers, we do calculations for 3, 5, 7, and 9 layers. For every system we have to change the `strc`-file: adding atoms into the unit cell and enhancing the third lattice vector accordingly to keep the vacuum length nearly constant. For all slab-calculations, search in the `prot`-file the respective values for the lowest occupied state (it's denoted by $\bar{\Gamma}_1$) and for the Fermi-level E_F (i.e. the `CHEMICAL POTENTIAL`). Fill in the table, the difference refers to the bandwidth and can be compared with the respective value for the bulk calculation Q.4 in the solids chapter. Interprete the results.

No. of layers	1	3	5	7	9	bulk
$\bar{\Gamma}_1$						
E_F						
E_F-$\bar{\Gamma}_1$						

Ergebnis:

No. of layers	1	3	5	7	9	bulk
$\bar{\Gamma}_1$	-8.693	-9.024	-7.579	-6.573	-5-839	-1.482
E_F	-1.483	1.457	3.247	4.401	5.238	9.689
E_F-$\bar{\Gamma}_1$	7.210	10.481	10.826	10.874	11.077	11.171

G.4.1 Relaxation of the Al (001) surface

We will study the relaxation on the example of the 9 layer slab. We take the results from the self-consistent calculation and change the `cntl`-file and the `strc`-file. Think about the changes in the `cntl`-file, the branch `RDYN` comes into play [3].

Within the `strc`-file there are now additional constraints. To mimic the surface, we have to fix some layers within the bulk structure. This is best done by keeping the atomic positions of the first 5 layers and let the atoms of the next 4 layers relax. The new branch of the `strc`-file looks like

```
!CONSTRAINTS
   !FREEZE ATOM='Al1' !END
   !FREEZE ATOM='Al2' !END
   !FREEZE ATOM='Al3' !END
   !FREEZE ATOM='Al4' !END
   !FREEZE ATOM='Al5' !END
 !END
```

If the atom in the 6-th layer is denoted by 'Al6' and case.prot is the respective `prot`-file, the relaxation can best be seen via the command "grep -A4 Al6 case.prot". Interprete the result.

This result is not typical for all atoms. For example, platinum shows a 5×1 reconstruction for the (001)-surface.

G.4.2 Density of states

For aluminum layers we do not study the two-dimensional bandstructure, the computer time necessary is large with respect to the learning effect. For the density of states, we have instead a descriptive possibility to study the development of bulk behavior with the number of layers. In chapter 7 we have seen that the aluminum bulk DOS starts with a square root behavior due to the nearly-free electron like valence band. If we have only a two-dimensional periodicity, the respective two-dimensional constant-energy areas are circles, thus the gradient also gives $\frac{1}{m^*}k$ (c.f. equation (Q.1)) and

$$\begin{aligned} d^2k &\equiv k\,d\varphi \\ \longrightarrow \int_{S_{\vec{k}(\epsilon)}} \frac{d^2k}{|\vec{\nabla}_{\vec{k}}\epsilon(\vec{k})|} &= \int_0^{2\pi} m^* \frac{k}{k} d\varphi \\ &= 2\pi m^*, \end{aligned}$$

the density of states is constant. Since for every additional layer a new band appears, we got an additive succession of constant lines with successive starting points, which form at least the square root behavior of the bulk density of states. To show this, is the main goal at this point and therefore we leave out the calculation and plotting of the atom resolved momenta weights of the density of states. But at least, you should always have a look into the `al.dprot-` file and understand the projective charge information given there.

Tasks:

- Do a DOS-calculation for the selfconsistent monolayer. Proceed as for the respective metal calculation in chapter 7. The `al.bcntl`-file looks like

```
!BCNTL
  !INPUTFILE NAME="al.banddata" !END
```

[3] FiXme Note: compare with section Superstructures?

```
  !METHOD
    MODE=2
    METHOD_DIAG=1
  !END
  !PDOS
    PDOSINFILE="al.pdos"
    PDOSOUTFILE="al.pdosout"
    NKDIV=10 10 1
    NB=12
    TUSESYM=F
    SPACEGROUP=225
  !END
  !DOS
    FILE="al.dos"
    EMIN=-10.0
    EMAX=20.0
    NE=1000
  !END
!END
!EOB
```

and the `al.dntl`-file

```
!DCNTL
  !GRID  EMIN[EV]=-10.0 EMAX[EV]=10. DE[EV]=0.04  !END
  !WEIGHT ID='Al'    TYPE='TOTAL' !END
  !WEIGHT ID='Al1'
   !ATOM NAME='Al1 ' TYPE='ALL' !END
  !END
 !OUTPUT ID='Al'  FILE='Altotal.dos' !END
 !OUTPUT ID='Al1'  FILE='Al1.dos' !END
!END
!EOB
```

- Do the same for 3,5,7,9 layers. For all these layers, the respective "al.pdosout"-files have already been calculated (the computer time is too high to do all calculations during the tutorial). You have to be sure that the onset of the valence band is within your given energy window of the `al.dcntl`-file. Plot for each layer the atom-resolved local density of states. Look at the broadening of these local densities. Do you see a tendency?

- Plot the total density of the bulk together with the total densities for all layers. Arrange the layer densities of states in such way, that they start on the onset of the bulk valence band. [4] For the layers you have to divide the respective DOS by the number of layers, which give the DOS per atom and is therefore comparable to the bulk density of states. [5]
  Can you explain, why at the onset region the monolayer DOS exceeds all others by far? Describe the deviations of the layer total density of states from the bulk DOS depending on the number of layers.

6

[4] FiXme Note: calculate the difference edif in the onsets of bulk and layer valence band , click in xmgrace on "Data", "Transformations", "Evaluate expression", in the opened window click the respective " Source" graph of the layer and the same "Destination" graph and input x=x+edif

[5] FiXme Note: click in xmgrace on "Data", "Transformations", "Evaluate expression", in the opened window click the respective " Source" graph of the layer and the same "Destination" graph and input y=y/nl, if nl is the number of the layers

[6] FiXme Note: Hier sollte auch erkennbar sein, dass sich aus dem zusätzlichen Auftreten von konstanten LDOS ein Wurzelverhalten entwickelt

G.5 Fe (001) surface

G.5.1 Monolayer

For that surface orientation, the next nearest neighbors are missing. The unit cell is a square with the bulk lattice constant.
Tasks:

- Do a paw-calculation with

```
 !KPOINTS DIV=9 9 1   !END
!OCCUPATIONS EMPTY=14 NSPIN=2 SPIN[HBAR]=2.0     !END
!SPECIES NAME='Fe' NPRO=1 1 2 LRHOX=2 ID='FE_HBS'  !END
```

- Generate the data for a **bandstructure** plot ($\bar{M}, \bar{\Gamma}, \bar{X}, \bar{M}$) with

```
!BCNTL
  !INPUTFILE NAME="fe.banddata" !END
  !METHOD
    ID=1
    METHOD_DIAG=1
  !END
  !LINE FILE='Fe_spinup.dat'
      KVEC1=0.5 0.5  0. KVEC2=0.0 0.0 0.0   kvecscale= 5.423      NK=10
      NB=7 spin=1  !END
  !LINE FILE='Fe_spinup.dat'
      KVEC1=0. 0.  0. KVEC2=0.5 0.0 0.0   kvecscale= 5.423      NK=8
      NB=7 spin=1 TATTACH=T !END
  !LINE FILE='Fe_spinup.dat'
      KVEC1=0.5 0.  0. KVEC2=0.5 0.5 0.0   kvecscale= 5.423      NK=8
      NB=7 spin=1 TATTACH=T !END
  !LINE FILE='Fe_spindown.dat'
      KVEC1=0.5 0.5  0. KVEC2=0.0 0.0 0.0   kvecscale= 5.423      NK=10
      NB=7 spin=2  !END
  !LINE FILE='Fe_spindown.dat'
      KVEC1=0. 0.  0. KVEC2=0.5 0.0 0.0   kvecscale= 5.423      NK=8
      NB=7 spin=2 TATTACH=T !END
  !LINE FILE='Fe_spindown.dat'
      KVEC1=0.5 0.  0. KVEC2=0.5 0.5 0.0   kvecscale= 5.423      NK=8
      NB=7 spin=2 TATTACH=T !END
!EOB
```

- Plot the bandstructure in the following way:
  (a) `xmgrace -nxy Fe_spinup.dat &`
  (b) choose one color for all these sets
  (c) import `Fe_spindown.dat` and choose another color for that respective sets
  (e) plot the vertical lines at $\bar{\Gamma}$ and $\bar{X}$
  (f) plot the Fermi-energy
  Maybe the interpretation of the plot is facilitated by marking all calculated points of the bands by a small symbol.

- Explain, why for the lowest band of both spin-directions the spin splitting changes drastically going from $\bar{\Gamma}$ to $\bar{X}$. [7]

[7] FiXme Note: Solution: On $\bar{\Gamma}$ the wavefunction has s-character, on $\bar{X}$ it's d

G.5.2 3,5, and 7 layers

Tasks:

- Do a paw-calculation for 3 and 5 layers.
- The DOS-output files from the `paw_bands.x`-run are given, also for the 9-layer slab (see also that respective `prot`-file).
- Calculate the corresponding DOS.
  Here is as an example the 7 layer `dcntl`-file:

```
!DCNTL
  !GRID  EMIN[EV]=-2. EMAX[EV]=13. DE[EV]=0.04  !END
  !WEIGHT ID='Fe'
   !ATOM NAME='Fe1 ' TYPE='ALL' !END
   !ATOM NAME='Fe2 ' TYPE='ALL' !END
   !ATOM NAME='Fe3 ' TYPE='ALL' !END
   !ATOM NAME='Fe4 ' TYPE='ALL' !END
   !ATOM NAME='Fe5 ' TYPE='ALL' !END
   !ATOM NAME='Fe6 ' TYPE='ALL' !END
   !ATOM NAME='Fe7 ' TYPE='ALL' !END
  !END
  !WEIGHT ID='Fe1'
   !ATOM NAME='Fe1 ' TYPE='ALL' !END
  !END
 !WEIGHT ID='Fe2'
   !ATOM NAME='Fe2 ' TYPE='ALL' !END
  !END
 !WEIGHT ID='Fe3'
   !ATOM NAME='Fe3 ' TYPE='ALL' !END
  !END
 !WEIGHT ID='Fe4'
   !ATOM NAME='Fe4 ' TYPE='ALL' !END
  !END
 !OUTPUT ID='Fe'  FILE='Fetotal.dos' !END
 !OUTPUT ID='Fe1'  FILE='Fe1.dos' !END
 !OUTPUT ID='Fe2'  FILE='Fe2.dos' !END
 !OUTPUT ID='Fe3'  FILE='Fe3.dos' !END
 !OUTPUT ID='Fe4'  FILE='Fe4.dos' !END
!END
!EOB
```

- Look into the `dprot`-files. Discuss the projected charges and the spin-analysis.
- Plot for every number of layers the spin resolved total DOS together with those ones for the atoms.
- Plot into another plot all total densities. Proceed analogously to aluminum, however, shift here all data, such that the Fermi-energy is equal for all slabs.

G.6 Clean Si (001) surface

The atomic and electronic structures of clean silicon(001) surfaces has been a problem of great technological interest because of their importance in solid-state devices. For the ideal surface, there

are two broken bonds per surface atom making the surface highly unstable. Pairing of surface atoms - resulting in so-called "dimers"- re- duces the number of dangling bonds by a factor of 2 and leads to a (2×1) periodicity.
This system cannot be calculated in an acceptable time within this course, therefore we will do a calculation for a 4 layer Si(001)-slab in a (2×1) geometry. In this case, we have to take account for the unsaturated bottom silicon atoms. For each of them we include two hydrogen atoms placed into the direction, where the now missing next silicon atoms would sit. This also ensures the bulk behavior of the bottom silicon atoms.

Here is the `strc`-file:

```
!STRUCTURE
  !GENERIC LUNIT=10.2612  !END
  !KPOINTS  DIV=6 12 1  !END
  !OCCUPATIONS EMPTY=40 NSPIN=1 SPIN[HBAR]=0.  !END
  !LATTICE T= 1.4142136 0.   0.
              0.00000  0.7071068 0.00000
              0.0000  0.00000  3.0 !END
  !SPECIES NAME= 'SI' ID='SI_HBS' NPRO= 1 2 0  lrhox=2
  !END
  !SPECIES NAME='H_' ID='H_HBS' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= 'SI1'  R= 0.00  0.00  0.00 !END
  !ATOM    NAME= 'SI2'  R= 0.7071068  0.0  0.0 !END
  !ATOM    NAME= 'SI3'  R= 0.00   0.3535534  0.25 !END
  !ATOM    NAME= 'SI4'  R= 0.7071068   0.3535534  0.25 !END
  !ATOM    NAME= 'SI5'  R= 0.3535534   0.3535534  0.5 !END
  !ATOM    NAME= 'SI6'  R= 1.0606602   0.3535534  0.5 !END
  !ATOM    NAME= 'SI7'  R= 0.3535534   0.0  0.75 !END
  !ATOM    NAME= 'SI8'  R= 1.0606602   0.0  0.75 !END
  !ATOM    NAME= 'H_21'   R= -0.2227386 0. -0.1575 !END
  !ATOM    NAME= 'H_22'   R= 0.2227386 0. -0.1575 !END
  !ATOM    NAME= 'H_23'   R= 0.4843682 0. -0.1575 !END
  !ATOM    NAME= 'H_24'   R= 0.9298454 0. -0.1575 !END
!END
!EOB
```

(paw-Lauf 20min) Durchsuchen des prot files auf Bandlücke –> metallic (entspricht nicht Experiment) Relaxation würde keine anderen Ergebnisse liefern, da lokales Extremum

Auslenkung der obersten Atome –> metallic aus prot RDYN (46min)–> liefert band gap und struktur, die mit experiment vergleichbar

Anschauen und Vergleich mit avogadro

Appendix H

Grenzflächen

Determine band offsets:

Appendix I

Tips and Tricks

I.1 Moving rectangles with Emacs

Often we have the atomic structure in some list form such as an xyz file. We take the example for malonaldehyde.

```
9
 Energy:      11.7225213
O          -2.82079        4.29839        4.04847
C          -3.63600        4.49555        4.92063
C          -3.90161        5.81347        5.56210
C          -3.12195        6.77266        5.05510
O          -2.19628        6.54119        4.06260
H          -2.23833        5.57570        3.82952
H          -4.22951        3.65891        5.27624
H          -4.64581        5.94446        6.34433
H          -3.09654        7.81558        5.32691
```

In addition we have a structure file for water, which we would like to turn into one for malonaldehyde.

```
!STRUCTURE
  !GENERIC LUNIT=1.88796 !END
  !OCCUPATIONS EMPTY=5 !END
  !LATTICE T= 0.00000 6.00000  6.00000
              6.00000 0.00000  6.00000
              6.00000 6.00000  0.0000 !END
  !SPECIES NAME='O_' ID='O_.75_6.0' M=5. NPRO=1 1 0 LRHOX=2  !END
  !SPECIES NAME='H_' ID='H_.75_6.0' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= 'O_1'  R= 0.00  0.00  0.00 !END
  !ATOM    NAME= 'H_2'  R= 1.05  0.00  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ISOLATE !END
!END
!EOB
```

First we create additional atom blocks, os that they can hold all the nine atoms of malonaldehyde. This is done by moving the curser to the beginning of one of the lines holding an "!atom" block, killing the line with C-k (control-kill), and inserting the lines repeatedly with C-y (control-yank). If you created more than you need, you can delete those later.

The file looks like

```
!STRUCTURE
  !GENERIC LUNIT=1.88796 !END
  !OCCUPATIONS EMPTY=5 !END
  !LATTICE T= 0.00000 6.00000  6.00000
              6.00000 0.00000  6.00000
              6.00000 6.00000  0.0000 !END
  !SPECIES NAME='O_' ID='O_.75_6.0' M=5. NPRO=1 1 0 LRHOX=2  !END
  !SPECIES NAME='H_' ID='H_.75_6.0' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= 'O_1'  R= 0.00  0.00  0.00 !END
  !ATOM    NAME= 'H_2'  R= 1.05  0.00  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ATOM    NAME= 'H_3'  R= 0.00  1.05  0.00 !END
  !ISOLATE !END
!END
!EOB
```

Now we delete the atomic positions an element symbols using rectangle kill. First we need to mark a rectangle in the file, which should contain all atomic positions but no other symbols. The rectangle is marked by "setting a mark" at the top left corner of the rectangle. We move the curser to the top left corner, that is ontop of the "0" of the x-coordinate of oxygen. Then we hit C-spacebar. In the bottom of the emacs window it will say "Mark set". This information is now kept by emacs. Now you move with the cursor to the bottom right corner of the rectangle, that is just **behind** the last digit of the z-coordinate of the last atom. Now hit "C-x r k". (control is held while x is pressed, while r an k are typed sequentially).

This should remove all atomic coordinates. The file looks like

```
!STRUCTURE
  !GENERIC LUNIT=1.88796 !END
  !OCCUPATIONS EMPTY=5 !END
  !LATTICE T= 0.00000 6.00000  6.00000
              6.00000 0.00000  6.00000
              6.00000 6.00000  0.0000 !END
  !SPECIES NAME='O_' ID='O_.75_6.0' M=5. NPRO=1 1 0 LRHOX=2  !END
  !SPECIES NAME='H_' ID='H_.75_6.0' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= 'O_1'  R=  !END
  !ATOM    NAME= 'H_2'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ATOM    NAME= 'H_3'  R=  !END
  !ISOLATE !END
!END
```

```
!EOB
```

Now we do the same to remove first digit of the element symbols. (The underscore will be needed, and the numbers have to be done individually later.)

In another emacs window we now open the xyz file and kill the rectangle with element symbol and the atomic positions, as described before.

Then we move to the new structure file and move the cursor just behind the first "R=" of the first atom. Now we hit "C-x r y" (control-x rectangle yank).

```
!STRUCTURE
  !GENERIC LUNIT=1.88796 !END
  !OCCUPATIONS EMPTY=5 !END
  !LATTICE T= 0.00000 6.00000  6.00000
              6.00000 0.00000  6.00000
              6.00000 6.00000  0.0000 !END
  !SPECIES NAME='O_' ID='O_.75_6.0' M=5. NPRO=1 1 0 LRHOX=2  !END
  !SPECIES NAME='H_' ID='H_.75_6.0' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= '_1'  R=O          -2.82079        4.29839        4.04847  !END
  !ATOM    NAME= '_2'  R=C          -3.63600        4.49555        4.92063  !END
  !ATOM    NAME= '_3'  R=C          -3.90161        5.81347        5.56210  !END
  !ATOM    NAME= '_3'  R=C          -3.12195        6.77266        5.05510  !END
  !ATOM    NAME= '_3'  R=O          -2.19628        6.54119        4.06260  !END
  !ATOM    NAME= '_3'  R=H          -2.23833        5.57570        3.82952  !END
  !ATOM    NAME= '_3'  R=H          -4.22951        3.65891        5.27624  !END
  !ATOM    NAME= '_3'  R=H          -4.64581        5.94446        6.34433  !END
  !ATOM    NAME= '_3'  R=H          -3.09654        7.81558        5.32691  !END
  !ATOM    NAME= '_3'  R=  !END
  !ATOM    NAME= '_3'  R=  !END
  !ISOLATE !END
!END
!EOB
```

We remove the two atom blocks that are not required. Then we move the rectangle with the element symbols into the atom name. We can also use rectangle-kill to remove empty spaces. The result looks as follows.

```
!STRUCTURE
  !GENERIC LUNIT=1.88796 !END
  !OCCUPATIONS EMPTY=5 !END
  !LATTICE T= 0.00000 6.00000  6.00000
              6.00000 0.00000  6.00000
              6.00000 6.00000  0.0000 !END
  !SPECIES NAME='O_' ID='O_.75_6.0' M=5. NPRO=1 1 0 LRHOX=2  !END
  !SPECIES NAME='H_' ID='H_.75_6.0' M=2. NPRO=1 1 0 LRHOX=2  !END
  !ATOM    NAME= 'O_1'  R=  -2.82079   4.29839   4.04847  !END
  !ATOM    NAME= 'C_2'  R=  -3.63600   4.49555   4.92063  !END
  !ATOM    NAME= 'C_3'  R=  -3.90161   5.81347   5.56210  !END
  !ATOM    NAME= 'C_3'  R=  -3.12195   6.77266   5.05510  !END
  !ATOM    NAME= 'O_3'  R=  -2.19628   6.54119   4.06260  !END
  !ATOM    NAME= 'H_3'  R=  -2.23833   5.57570   3.82952  !END
  !ATOM    NAME= 'H_3'  R=  -4.22951   3.65891   5.27624  !END
  !ATOM    NAME= 'H_3'  R=  -4.64581   5.94446   6.34433  !END
  !ATOM    NAME= 'H_3'  R=  -3.09654   7.81558   5.32691  !END
  !ISOLATE !END
```

```
!END
!EOB
```

Finally we adjust the atom numbers of the atom names, and modify the rest of the structure file.

I.2 Optimize structures with Openbabel

- using the commandline-tool `obminimize` from OpenBabel (http://openbabel.org/wiki/Obminimize) you can do the optimization on the console without Avogadro.
- http://openbabel.org/wiki/Babel

Part IV

Additions 2

Appendix J

Computing

J.1 Introduction

We use a number of open source software package that come along with the Linux operating system. We selected a set of software packages that are free, have a good functionality and often can be run on other operating systems as well. An important ingredient is that the data developed can, at least in principle, be recovered without that software. This allows, at least in principle, to avoid loss of valuable data, if the software loses compatibility with old versions, is no more maintained or is superseded by a better technology.

Much of this software is developed by the Free Software Organization, a non-profit organization, that finances itself mostly by donations and service. In contrast to common expectations, this software, even though freely available, is much more reliable than commercial software.

- linux (file completion, history, search path)
- Emacs editor
- Xmgrace
- xfig
- latex

J.1.1 Shell-script programming

The following example demonstrates how to perform a sequence of similar PAW calculations and to create a list containing the corresponding total energies

```
#!/bin/sh
# perform a loop where the variable X is set equal to the values in the list
for X in 3_1 3_2 3_3 3_4 3_45 3_5
do
 Y=h2_‘echo -n $X | sed  s/_//g‘  # Y=h2_345
 Z=‘echo -n $X | sed  s/_/./g‘    # Z=3.45
 CNTL=$Y.cntl                     # CNTL=h2_345.cntl
 STRC=$Y.strc                     # STRC=h2_345.strc
 PROT=$Y.prot                     # PROT=h2_345.prot
 cp h2.cntl $CNTL
#  copy h2.strc into h2_345.strc and replace string "1.29" by "3.45"
 sed s/1.29/$Z/g h2.strc > $STRC
```

```
 echo doing $Y ...
# execute paw code
 paw_fast.x $CNTL 1>$Y.out 2>&1
# RES is the content of the last line containing the string "TOTAL ENERGY"
# this line contains the last value of the total energy
 RES=‘grep "TOTAL ENERGY"  $PROT |tail -n 1‘
# append this line to the file summary
 echo $Z $RES >>summary
 echo .... $Y done
done
echo ... finished!
```

Here the control file used in the above example

```
!CONTROL
 !GENERIC NSTEP=2000 NWRITE=500 START=T DT=5. !END
 !FOURIER EPWPSI=30. CDUAL=2.0 !END
 !DFT     TYPE=10 !END
 !PSIDYN STOP=T  FRIC=0.01 SAFEORTHO=T
   !AUTO FRIC(-)=0.3 FACT(-)=0.97  FRIC(+)=0.3 FACT(+)=1. minfric=0.01 !END
 !END
!END
!EOB
```

Here the structure file used in the above example

```
!STRUCTURE
  !GENERIC LUNIT=1.8897269 !END
  !OCCUPATIONS EMPTY=2 NSPIN=2 CHARGE[E]=0.0 SPIN[HBAR]=0.0 !END
  !LATTICE T= 0.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 0.0 !END
  !SPECIES NAME='H_' ID='H_.75_6.0' NPRO=1 1 !END
  !ATOM NAME='H_1'  R=   0.   0.  0.   !END
  !ATOM NAME='H_2'  R=   1.29  0.  0.   !END
  !ISOLATE !END
!END
!EOB
```

The resulting file summary may look like this

```
3.1 TOTAL ENERGY : -0.9423612 H
3.2 TOTAL ENERGY : -0.9398863 H
3.3 TOTAL ENERGY : -0.9377142 H
3.4 TOTAL ENERGY : -0.9358009 H
3.45 TOTAL ENERGY : -0.9349315 H
3.5 TOTAL ENERGY : -0.9080667 H
```

In order to visualize the result we need to get rid of the text. Tis can be done using the column editor “awk” or the rectangle editing capabilities of Emacs.

J.2 Shell programming

J.3 Tips and Tricks

- Name completion: file names and command names can be completed automatically using the tabulator key. It completes as far as it is unambiguous. Typing tab twice shows a selection of

possible completions.

- Command history: the up and down arrows on the command line allow one to maneuver through recently used commands.

J.4 Fortran

The CP-PAW code is written in Fortran.

For an introduction to Fortran look up the book by Metcalf and Reid *Modern Fortran explained*. (Watch for newer versions of that book.)

J.5 Python

Skip this section on python. It is a very early draft

Python is a high-level programming language, which is more powerful than shell-skripts, but simpler than conventional programming language. The advantages are:

- It is an interpreted language. Compilation takes place at runtime. Each line of the program can be executed after it is written. Thus it can be used even like a simple calculator.
- It takes only a minimum of declarations. The model behind is that python tries to guess whenever possible. Furthermore, arrays do not have fixed bounds.
- python probably replaces text editing such as awk,sed, a graphics tool such as gnuplot.

For us, Python bridges the gap between simple shell scripting with bash or tcsh and a compiled language such as Fortran and C++.

For a course on python see `http://www.wspiegel.de/pykurs/`.

NumPy is a library containing matrix manipulation and linear algebra operations such as diagonalization, solving eigenvalue problems etc. It basically has similar range of functions as MATLAB.

One executes a python script with "python script.py", where script.py is for example the following.

```
#from numpy import *

import numpy as num
import numpy.matlib as M
from numpy.matlib import rand,zeros,ones,empty,eye
from numpy.linalg import eig

A=num.mat("1,2;3,4")
b=num.mat("5;6")
print "A is \n ", A
print "b is \n", b
print "the element (1,2) of A  is ", A[0,1]
print "the first row of A  is ", A[0,:]
print "the first columnof A  is \n", A[:,0]

#transpose
print "transpose of A\n",A.T
print "transpose of b\n",b.T

#eigenvalues
```

```
e,U=eig(A)
print "eigenvalues of A ",e
print "eigenvectors of A\n",U

#singular value decomposition
U,S,V=num.linalg.svd(A)
print "singular values of A:",S
#?????print "test singular value decomposition. Should be zero:\n",U*S*V.T

# inverse or pseudo inverse
AINV=num.linalg.pinv(A)
print "inverse of A\n",AINV
print "test of inversion. Should be unity:\n", A*AINV

#solve equation system
x=num.linalg.solve(A,b)
print 'x from A*x=b\n',x
print 'test equation system solver. Should be zero:\n ',A*x-b
# for non-square matrices A
y=num.linalg.lstsq(A,b)
print 'x from A*x=b\n',y

#  eye(n) constructs unit matrix in n dimensions. It is however a 2-d array
# and does not allow matrix multiplication. the function num.mat converts
# into the matrix format
print " construct a 3-d unit matrix ",num.mat(eye(3))
print " construct a 4-d zero matrix\n " ,num.mat( num.zeros((4,4)) )
print " array containing the diagonal elements of A" ,num.diag(A)

print "norm of b",num.linalg.norm(b)
print "dot product of x and b ",x.T*b
print "dyadic product of x and b ",x*b.T

u=[1,2]
v=[2,3]
print 'dot_product ',num.dot(u,v)


file="input.dat"
input_file=open(file,"r")
text=input_file.read()
input_file.close()
print text

file="output.dat"
output_file=open(file,"w")
output_file.write(text)
output_file.close()

import os
os.popen('touch newfile')
os.popen('echo done')
import subprocess
subprocess.Popen('echo done1; echo done2',shell=True)
```

The file input.dat residing in the same directory is

```
this is file "input.dat"
it is meant to test file io with python
```

Appendix K

Templates for shell scripts

K.1 Scan lattice constants

Caution! This shell script removes files! execute in a dedicated directory!

Name the following file "scanlat", make it executanble with "chmod +x scanlat" and execute it with the rootname of the substance.

```
#!/bin/sh
#  performs set of calculations for different lattice constants
#  and atomic calculations for all files of ending with "_atom.strc"
#
#  requirement: nio.cntl
#               nio.strc
#               ni_atom.strc
#               o_atom.strc
#
#  nio is to be replaced by the name of the substace and
#  ni and o must be replaced by the corresponding atom ids.
#
#  nio.strc must contain exactly one line containing "LUNIT="
#  followed by the lattice constant in atomic units, followed by "!END".
#  this occurrance will be replaced by the varied values for the
#  individual runs.
#
#  result       etot.dat
#               gap.dat
#
#=============================================================================
# SET THESE VARIABLES. THE STRING ALAT0 must match with all digits the value
# given in the structure file
#=============================================================================
NAME=$1
#
#=============================================================================
# pick out the equilibrium lattice constant from the structure file
#=============================================================================
X=‘grep ’LUNIT=’ $NAME.strc‘  #pick the line containing LUNIT
X=${X#*LUNIT=}
X=${X%%!END*}
```

```
echo $X
ALAT0=$X
#
#=============================================================================
# loop crystal calculation through different lattic constants                ==
#=============================================================================
for X in 96 97 98 99 100 101 102 103 104; do
  #===========================================================================
  # construct structure control file from template by replacing lattice constant
  #===========================================================================
  ALAT=‘echo "$X /100 * $ALAT0 " | bc -l‘
  sed ’s/$ALAT0/$ALAT/g ${NAME}.strc’ >${NAME}_${X}.strc
  #===========================================================================
  # run paw_fast.x and place data "gap.dat" and "etot.dat"
  #===========================================================================
  cp ${NAME}.cntl ${NAME}_${X}.cntl
  rm ${NAME}_${X}.prot
  paw_fast.x ${NAME}_${X}.cntl 1>${NAME}_${X}.out 2>&1
  grep "ABSOLUTE GAP" ${NAME}_${X}.prot \
      | tail -n 1 \
      | awk ’{print ’${X}’  " " $3}’ >> gap.dat
  grep "TOTAL ENERGY" ${NAME}_${X}.prot \
      | tail -n 1 \
      | awk ’{print ’${X}’  " " $4}’ >>etot.dat
  rm ${NAME}_${X}_constr.report
  rm ${NAME}_${X}_r.tra
  rm ${NAME}_${X}.pdos
  rm ${NAME}_${X}.strc_out
  rm ${NAME}_${X}_stpfor*.myxml
done
#
#=============================================================================
# do atomic calculations                                                    ==
#=============================================================================
echo ’doing isolated atoms....’
for X in *_atom.strc; do
  ATOM=${X%_atom.strc}_atom
  echo $ATOM
  rm ${ATOM}.prot
  cp ${NAME}.cntl ${ATOM}.cntl
  paw_fast.x ${ATOM}.cntl 1>${ATOM}.out 2>&1
  grep "TOTAL ENERGY" ${ATOM}.prot \
      | tail -n 1 \
      | awk ’{print "atom " $4}’ >atometot.dat
  rm ${ATOM1}_constr.report
  rm ${ATOM1}_r.tra
  rm ${ATOM1}.pdos
done
echo ’.....done’
exit
```

K.2 Scan lattice constants

This shell script actually writes a make file, which actually does the work.

Make

A make file is a set of recipes for constucting targets from its dependencies. A recipy has the following form

```
target : dependencies
[tab]1 command of rule
[tab]2 command of rule
```

The recipy defines on the first line the target and its dependencies. If one of the dependencies, which usually is a file, has changed since the targed has been made the last time, make will execute the recipy, that is it will process the commands. Most important is that each command must be preceded by a tab, not by a number of spaces!

The dependencies themselfes may be targets in other rules. Make will check if a dependency appears as target in the makefile, and it will rebuild it before using it as dependency. This means that the order in which the recipies are applied, is given by the logical connections of the recipies and not by the order in which they occur in the makefile.

Take the example that we wish to make an executable do.x from two fortran files code1.f90 and code2.f90

```
do.x : code1.o code2.o
[tab]f90 -o do.x code1.o code2.o
#
code1.o : code1.f
[tab]f90 -o code1.o code1.f
#
code2.o : code2.f
[tab]f90 -o code2.o code2.f
```

Important: Do not type out [tab] but replace it by the invisible tabulator.

The Makefile is executed by the command

make -k -j8 do.x

The name "Makefile" is the default name, so that we do not have to specify it. The option -k (keep going) tells make to not abort upon the first error, but to execute as many recipies as possible. The option -j8 (8 jobs) tells make to execute 8 rules in parallel, which is suitable if you have several processors on your computer. The number 8 can be replaced by any other number. The argument "do.x" is the target that shall be calculated. That is. make will work on all rules that lead to the target all, but it ignores all others. On default it takes the first target mentioned.

Shellscript to write the makefile

```
#!/bin/bash
#*****************************************************************************
#*****************************************************************************
#**    make -j8 -k all
#               # -j8 (jobs) tells make to do 8 recipes in parallel
#               # -k (keep going) tries to satisfy as many recipes as possible
#*****************************************************************************
```

```
#*******************************************************************************
BASEDIR=‘pwd‘
SCALES="96 97 98 99 100 101 102 103 104"
MAKEFILE=${BASEDIR}/Makefile
#
#==============================================================================
#== construct targe all and construct relevant subdirectories
#==============================================================================
SUBDIRS=""
for SUBDIR in * ;do
  DIR=$BASEDIR/$SUBDIR/ALAT
  if test -e $DIR; then
    SUBDIRS="$SUBDIRS $SUBDIR"
  fi
done
echo -e "all : $SUBDIRS"                                          > $MAKEFILE
                                              # existing version is overwritten
#===============target clean ================================
echo -e ""                                                        >>$MAKEFILE
echo -e "clean: "                                                 >>$MAKEFILE
for SUBDIR in $SUBDIRS ;do
  DIR=$BASEDIR/$SUBDIR/ALAT
  echo -e "\t-rm $DIR/*_constr.report"                            >>$MAKEFILE
  echo -e "\t-rm $DIR/*.pdos"                                     >>$MAKEFILE
  echo -e "\t-rm $DIR/*_r.tra"                                    >>$MAKEFILE
  echo -e "\t-rm $DIR/*.out"                                      >>$MAKEFILE
                      # the "-" in front of the command lets make continue
                      # with the other commands
                      #
                      # the \t is a tap. it requires the -e option in echo
done
#
#=================flag to delete file on interupt
  echo -e ""                                                      >>$MAKEFILE
  echo -e ".DELETE_ON_ERROR :"                                    >>$MAKEFILE
                      # partially completed targets are removed on interrupt
                      # see gnu make manul p49.
#
#==============================================================================
#
#==============================================================================
cd $BASEDIR
for SUBDIR in ${SUBDIRS} ;do
  DIR=$BASEDIR/$SUBDIR/ALAT
  D=${DIR#$BASEDIR/}/
  if test -e $DIR; then     # does file $DIR exist?
    cd $DIR
    echo doing ‘pwd‘ ...
    #======== collect main substance================
    NAME=""
    for X in *.strc ; do
      case "$X"  in
        *_*) #strc files with an underscore are ignored
            ;;
```

```
        *)
#          if test -n $NAME; then
#            echo main substance is not unique
#            echo two strc files without underscore present
#            echo NAME1=$NAME NAME2=${X%.strc}
#            exit
#          fi
           NAME=${X%.strc}
           ;;
      esac
    done
    #=========================================================================
    # pick out the equilibrium lattice constant from the structure file
    #=========================================================================
    X=‘grep ’LUNIT=’ $NAME.strc‘  #pick the line containing LUNIT
    X=${X#*LUNIT=}
    X=${X%%!END*}
    ALAT0=$X
#
    #=====================================================================
    #==== constuct make file  ===============
    #=====================================================================
    echo ..adding to make file ...
    echo -e ""                                                    >>$MAKEFILE
    echo -e "$SUBDIR : ${D}etot.dat ${D}gap.dat"                  >>$MAKEFILE
    #==== target: $NAME_$SCALE.prot ========================================
    PROTS=""
    for X in ${SCALES}; do
      PROTS="$PROTS $D${NAME}_${X}.prot"
      ALAT=‘echo "$X /100 * $ALAT0 " | bc -l‘   # current lattice constant
#     == target structure file ============================================
      echo -e ""                                                  >>$MAKEFILE
      echo -e "$D${NAME}_${X}.strc : $D${NAME}.strc"              >>$MAKEFILE
      echo -e "\tsed \"s/${ALAT0}/${ALAT} /g\" $D${NAME}.strc"\
               "> $D${NAME}_${X}.strc"                            >>$MAKEFILE
#     == target protfile ==================================================
      echo -e ""                                                  >>$MAKEFILE
      echo -e "$D${NAME}_${X}.prot : $D${NAME}_${X}.strc $D${NAME}.cntl" \
                                                                  >>$MAKEFILE
      echo -e "\tcp $D${NAME}.cntl $D${NAME}_${X}.cntl"           >>$MAKEFILE
      echo -e "\t-rm $D${NAME}_${X}.prot"                         >>$MAKEFILE
      echo -e "\tpaw_fast.x $D${NAME}_${X}.cntl 1>$D${NAME}_${X}.out 2>&1" \
                                                                  >>$MAKEFILE
    done
    #==== target: etot.dat ================================
    echo ... doing target etot.dat ...
    echo -e ""                                                    >>$MAKEFILE
    echo -e "${D}etot.dat : $PROTS"                               >>$MAKEFILE
    echo -e "\t-rm  ${D}etot.dat"                                  >>$MAKEFILE
    for X in ${SCALES}; do
      echo -e "\tgrep ’TOTAL ENERGY’ $D${NAME}_${X}.prot" \
              "| tail -n 1 " \
              "| awk ’{print ${X} \" \" \$\$4}’ >> ${D}etot.dat"   >>$MAKEFILE
                # make requires two dollars for awk to avoid
```

```
                # interpreting is as value of the variable 4 (which is empty)
    done
    #==== target: gap.dat ================================
    echo ... doing target gap.dat ...
    echo -e ""                                                    >>$MAKEFILE
    echo -e "${D}gap.dat : $PROTS"                               >>$MAKEFILE
    echo -e "\t-rm  ${D}gap.dat"                                  >>$MAKEFILE
        for X in ${SCALES}; do
      echo -e "\tgrep 'ABSOLUTE GAP' $D${NAME}_${X}.prot " \
              "| tail -n 1 " \
              "| awk '{print ${X}  \" \" \$\$3}' >> ${D}gap.dat "  >>$MAKEFILE
    done

    #==== execute makefile ===================
    echo  ... `pwd` done
    cd $BASEDIR
  fi
done
exit
```

Once the doit file is written, make it executable by the command `chmod +x doit` and execute it by saying `doit`. As a result a file "Makefile" will be constructed, which can be excuted by `make -k -j8 all`.

K.3 Automatic replacement of parts in files

The following script "myresolve" allows the automatic replacement of parts in the structure files. Consider the case that many different calculations shall use the same species parameters.

```
  !SPECIES NAME='PR' ID='PR_HBS_SC' NPRO=2 1 1 1
    !NTBO TORB=T F T T T FOCKSETUP=T CV=T
          NDDO=F 31=F BONDX=F TAILLAMBDA=4.D0 2.D0 !END
  !END
```

It may be useful to have all this information at a central place. Therefore we place for each species the above sequance into a file named for example "`pr.species`" and place it in directory "/home/me/Specs".

Now we prepare a template for the structrue file `x.strc0` and replace the sequence simply by `@pr.species@` in the structure file.

This template structure file is then expanded by myresolve -p/home/me/Specs x.strc0 >x.
into the final structure file.

myresolve

The script myresolve is the following:

```
#!/bin/bash
##########################################################################
## replaces every line containing @file@ by the content of ${PRE}/file
##  insspecies -p. pro2.strc
##########################################################################
PRE=$1
IN=$2
```

```
PRE=${PRE#-p}
# collect
VAR=‘sed -n -e ’/@*@/p’ $IN‘
LIST=""
for X in $VAR; do
  X=${X%@}
  X=${X#@}
  LIST="$LIST $X"
done
CHS1=""
for X in $LIST; do
   CHS1="$CHS1 -e/@$X@/r$PRE/$X "
done
#
CHS2=""
for X in $LIST; do
   CHS2="$CHS2 -e/@$X@/d "
done
sed $CHS1 ${IN} |sed $CHS2
exit
```

K.4 Make atomic calculations

```
#!/bin/bash
#*****************************************************************************
#*****************************************************************************
#**    make -j8 -k all
#               # -j8 (jobs) tells make to do 8 recipes in parallel
#               # -k (keep going) tries to satisfy as many recipes as possible
#*****************************************************************************
#*****************************************************************************
BASEDIR=‘pwd‘
MAKEFILE=${BASEDIR}/make_atoms
#
#============================================================================
#== construct targe all and construct relevant subdirectories
#============================================================================
SUBDIRS=""
for X in Atoms/* ;do
  if test -d $X ; then
    SUBDIR=${X#Atoms/}
    SUBDIRS="$SUBDIRS $SUBDIR"
  fi
done
echo -e "all : $SUBDIRS" >  $MAKEFILE  # existing version is overwritten
echo -e "\t-rm Atoms/atometot.dat"                              >> $MAKEFILE
for SUBDIR in $SUBDIRS ;do
  X=$BASEDIR/Atoms/$SUBDIR/etot.dat
  echo -e "\tcat $X >> Atoms/atometot.dat"                      >> $MAKEFILE
done

#==============target clean ================================
```

```
echo -e ""                                                          >> $MAKEFILE
echo -e "clean: "                                                   >> $MAKEFILE
for SUBDIR in $SUBDIRS ;do
  DIR=$BASEDIR/Atom/$SUBDIR
  echo -e "\t-rm $DIR/*_constr.report"     >> $MAKEFILE
  echo -e "\t-rm $DIR/*.pdos"              >> $MAKEFILE
  echo -e "\t-rm $DIR/*_r.tra"             >> $MAKEFILE
  echo -e "\t-rm $DIR/*.out"               >> $MAKEFILE
                      # the "-" in front of the command lets make continue
                      # with the other commands
                      #
                      # the \t is a tap. it requires the -e option in echo
done
#
#=================flag to delete file on interupt
  echo -e ""                                >> $MAKEFILE
  echo -e ".DELETE_ON_ERROR :"              >> $MAKEFILE
                      # partially completed targets are removed on interrupt
                      # see gnu make manul p49.
#
#============================================================================
#
#============================================================================
echo $SUBDIRS
cd $BASEDIR
for SUBDIR in ${SUBDIRS} ;do
  DIR=$BASEDIR/Atoms/$SUBDIR
  cd $DIR
  echo doing `pwd` ...
  #======== collect main substance================
  NAME=""
  for X in *_atom.strc ; do
    case "$X"  in
        *)
          NAME=${X%_atom.strc}
          ;;
      esac
    done
    ROOT=$BASEDIR/Atoms/${SUBDIR}/${NAME}_atom
echo $NAME
    echo -e ""                                                      >>$MAKEFILE
    echo -e "$SUBDIR : ${DIR}/etot.dat"                            >>$MAKEFILE
    echo -e ""                                                      >>$MAKEFILE
    echo -e "${DIR}/etot.dat : ${ROOT}.strc ${ROOT}.cntl"          >>$MAKEFILE
    echo -e "\t-rm ${ROOT}.prot "                                   >>$MAKEFILE
    echo -e "\tpaw_fast.x ${ROOT}.cntl 1>${ROOT}.out 2>&1 "        >>$MAKEFILE
    echo -e "\tgrep 'TOTAL ENERGY' ${ROOT}.prot" \
              "| tail -n 1 " \
              "| awk '{print \""${NAME}"\" \" \" \$\$4}' > $DIR/etot.dat" >>$MAKEFILE
    #==== execute makefile ==================
    echo  ... `pwd` done
    cd $BASEDIR
done
exit
```

Appendix L

Course planning

The course is planned for one week. Lectures will be

- 9:00-10:30 Theory of first-principles calculations
- 11:00-12:30 Nature of the Chemical Bond
- 12:30-14:00 lunch break
- 14:00-17:00 Hands-on Class

In the second week, the students will perform projects amd will report them on friday morning of the second week. The presentation will be on an individual basis and it will be graded.

One ECTS amounts to 25-30 hours of work. Thus a two-week course can be 2-3 ECTS. (25 h*3ECTS/10days=7.5 h/day; 30 h*2 ECTS/10 days=6h/day)

Problem: A modul should have 5 ECTS minimum.

L.1 Teaching Goals

After the course, the students are able to perform electronic structure calculations, evaluate their quality, and evaluate their results.

L.2 Praktikum

1. introduction to the computer, bash scripting, electronic and structure optimization of H_2O, plot wave functions.
2. ab-initio molecular dynamics of malonaldehyde at T>0, extract trajectories, extract frequency density of states, frequencies with small displacements
3. SN2 reaction (Transition states)
4. Solids: silicon, aluminum, MgO, Pt, Density of states, band structure
5. Magnetism Fe, Density of states, band structure, broken symmetry state for the magnetic molecule $(SH)_2Fe_2S_2(SH)_2$. Hyperfine parameters
6. Point defects, Supercells, Surfaces, Interfaces (band offsets)

L.3 Lecture Theoretical backgroud

1. Density functional theory I
2. Density functional theory II
3. Introduction, molecular dynamics, Verlet algorithm, stability, convergence ab-initio molecular dynamics
4. Transition state theory
5. Electronic structure methods: Pseudopotentials, Augmented waves, PAW.
6. K-points, supercells
7. Numerics: Sawtooth, planewave convergence (absolute/difference), k-point convergence, optimization techniques.
8. Analysis: Hyperfine parameters, electric field gradients, EELS spectra, band offsets,

L.4 Lecture: Chemical bond

1. Bonding patterns: Two-center bonds, three center bonds, rings and chains
2. Symmetry orbitals
3. Hybrid orbitals
4. Frontier orbitals
5. From bonds to bands: Tight bindig picture
6. From atoms to solids, canonical band structures
7. Instabilities: Jahn-Teller distortion, Peierls distortion, magnetism
8. Ionic cmpounds, Electronegativity Softness, Hardness
9. Crystal structures

L.5 Todo: Notes to the author

- include other explanatory chapters obout: mermin, supercells, thermostat, tst, verlet, dft
- Internet resources: how to find starting structures
- Include a section “time, length and energy scales”. Inlcude figures (Started but not completed)
- In the Lectures include a list of topics and include topics slide when going from one to the next.
- In the tests also test the size of the super cell for molecules.
- How to test the convergence: Let system evolve without friction. Estimate energy distance from minimum. Estimate energy tolerance from force.
- make md simulation for polyacetylene
- discuss safeortho. Safeortho=F does not lead to energy conservation in the optimization.

- There is an energy drift during the simulation. Robert found for Malonaldehyde that the energy of a converged structure without any friction shakes up. We should analyze the origin of the lack of energy conservation: One possible cause is that the projections are not calculated in the beginning of the iteration but projected from the previous ones. Another possible cause is that the the next positions are calculated with an extrapolated lagrange multiplier for the orthogonalization constraint. Analysis 1: The error is already present without atomic motion. Analysis 2: The error is not related to isolate. Analysis 3: The error occurs also if the wave function is fully converged and only the wave functions are allowed to move without initial velocity.

Part V

Trash (Alte Versuchsbeschreibungen etc.)

L.6 Hilfsprogramme

L.7 Hilfsprogramme

L.7.1 Latex

Latex ist ein frei verfügbares Textverarbeitungsprogramm, welches für Linux wie auch für Windows (Mik-Tex) erhältlich ist. Es ist zwar schon relativ alt, aber in puncto mathematische Formeln ungeschlagen, weshalb es in der Mathematik und Physik sehr häufig verwendet wird. Anders als bei den meisten üblichen Textverarbeitungsprogrammen kannst du hier dein Dokument nicht durch Klicken u.Ä. verändern, sondern du musst ein Textfile (mit der Endung .tex) schreiben, welches zum einen den Text und zum Anderen Formatierungsbeschreibungen, die meistens mit einem backslash (\) beginnen, enthält. Da latex ein sehr umfangreiches Programm ist, hier nur ein Beispiel für ein Latex-Dokument:

```
\documentclass{article}
\usepackage{german}

\begin{document}

\section{Ein einfaches Latex-Dokument}
Hier soll gezeigt werden, wie

\begin{itemize}
\item ein Latex-Dokument aufgebaut ist.
\item mathematische Formeln geschrieben werden.
\end{itemize}

\subsection{Etwas Text}
Um einen neuen Absatz zu beginnen, muss man im Latex-Code eine Zeile frei
lassen, so wie hier.

Will man nur eine neue Zeile beginnen, ohne einen neuen Absatz, geht das\\
so wie hier.

\subsection{Eine Formel}
Mathematische Formeln werden (beispielsweise) in Dollarzeichen eingeschlossen.

$i \hbar \partial_t \Psi (\vec r, t) = (- \frac {\hbar^2} {2m} \nabla^2 + V
(\vec r, t) ) \Psi(\vec r, t)$

\end{document}
```

Kompiliere dieses File nun mit Hilfe des Befehls `latex` *name*`.tex`. Dabei entsteht ein File *name*`.dvi`, welches du beispielsweise mit dem Befehl `xdvi` *name*`.dvi &` ansehen kannst. Du kannst es auch mit dem Befehl `dvips` *name*`.dvi` in ein Postscript-File umwandeln.

L.7.2 xmgrace

Aufruf und Funktion

xmgrace ist ein Programm zur Darstellung von Diagrammen und Kurven. Berechnet man zum Beispiel die Zustandsdichte eines Atoms, so erhält man die Funktion $D(E)$. E läuft dann in den

Intervallgrenzen mit fester Schrittweite ΔE durch und zu jedem Punkt wird D ausgerechnet. In der Ausgabedatei stehen dann Zahlenreihen der Form:

```
x.xxxxx   y.yyyyyy
x.xxxxx   y.yyyyyy
x.xxxxx   y.yyyyyy
x.xxxxx   y.yyyyyy
x.xxxxx   y.yyyyyy
```

Ruft man

```
xmgrace daten_file
```

auf, bekommt man die Zustandsdichte geplottet. Für den Fall, dass man nicht nur ein File der Form

$x \quad f_1(x)$

hat, sondern mehrere Funktionen überlagern möchte

$x \quad f_1(x) \quad f_2(x) \ldots,$

startet man xmgrace mit

```
xmgrace -nxy daten_file
```

So wird sowohl f_1 als auch f_2 in einen Graphen geplottet.

L.7.3 FORTRAN

Übung 1

Folgendes Fortran-Programm liefert Dir eine Wertetabelle für den Sinus.

```
PROGRAM wert_tab

IMPLICIT NONE

real(8)            :: x, f1
real(8), PARAMETER :: d=0.001d0
integer(4)         :: i, max
real(8)            :: u_grenze, o_grenze, anzahl

open(unit=10, FILE="tabelle.dat", STATUS="UNKNOWN")

u_grenze = -6.5d0
o_grenze = 6.5d0

anzahl = (o_grenze - u_grenze) / d
x = u_grenze
max = abs(anzahl)

do i=1,max
f1 = SIN(x)
```

```
write(10, FMT='(2F8.4)') x, f1
x = x + d
enddo

close(10)
STOP

END PROGRAM wert_tab
```

Fortran-Programme solltest du in Files mit der Endung .f90 speichern. Um sie ausführen zu können, musst du sie zunächst kompilieren (übersetzen). Hierür gibt es zwei Compiler, und zwar das GNU-Programm `g95` und den Intel-Fortran-Compiler `ifc7`. Tippst du also `g95` *`name`*`.f95` oder `ifc7` *`name`*`.f95`, so erzeugt der Compiler ein File mit dem Namen `a.out`, welches du mit `./a.out` ausführen kannst. Möchtest du deinem Programm einen anderen Namen geben, benutze die Option `-o`: `g95 -o` *`programmname name`*`.f95`.

1. Teste es und variiere die Parameter. Schau Dir die Ergebnisse mit `xmgrace` an.
2. Verändere das Programm so, dass es Sinus- und Cosinusfunktion gleichzeitig darstellt.
3. Stelle beide Kurven gleichzeitig in `xmgrace` dar und beschrifte die x-Achse mit $\pi/2$, π, $3/2\pi$ usw.

Texteingabe

Wichtige Befehle für Texteingabe:

\ **s** subscript

\ **S** superscript

\ **x** Griechische Buchstaben EIN

\ **B** Lateinische Buchstaben EIN

\ **o** overline

\ **u** underline

\ **c** Sonderzeichen EIN

Will man beispielsweise eine Achse beschriften mit

Δd/Ångstrom

so wählt man `Window/Drawing Objects` und dann Text. Dann klickt man in dem Graphen an die Stelle, wo der Text stehen soll und schreibt:

```
\xD\Bd / \cE\Bngstrom
```

Übung 2

Beschrifte eine Achse mit:

$$\frac{\Delta E}{\text{kJ mol}^{-1}}$$

Hinweis: Den Bruchstrich kann man auch mit den drawing objects konstruieren.

Möchtest du mehr über Programmierung in Fortran lernen, so seien dir die Übungen zur Vorlesung "Fortgeschrittene Themen der Theoretischen Physik 1" empfohlen.

L.7.4 molden

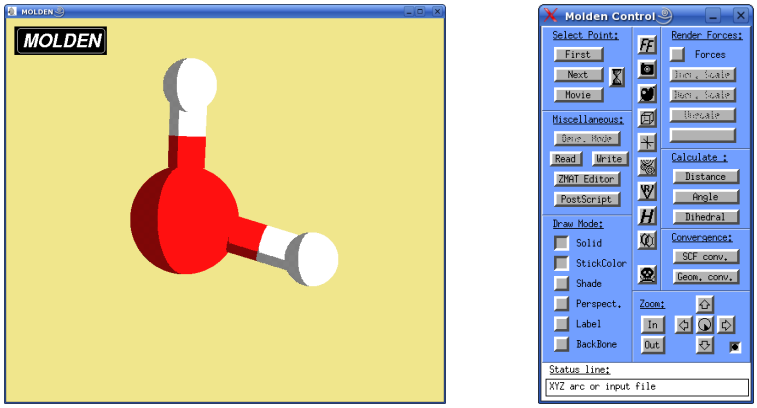


Fig. L.1: Das Programm molden mit einem Wassermolekül

Hat man etwa ein Molekül mit PAW berechnet, erhält man als Ergebnis oftmals "Bilder" vom Molekülen oder Movies von Rechnungsabläufen, welche im Format .xyz gespeichert sind. Solche Files enthalten lediglich die Namen und Positionen der einzelnen Atome. Man kann sie beispielsweise mit dem Programm molden ansehen und damit auch beispielsweise Bindungsabstände und Bindungswinkel berechnen oder "richtige" Bilder daraus erzeugen.

Das Kommando zum Aufrufen vom molden ist oft versionsabhängig. Möchtest du beispielsweise ein File mit dem Namen `case.xyz` mit der Version 4.1 des Programmes ansehen, so lautet das Kommando `molden4.1 case.xyz`. Es öffnen sich zwei Fenster, in einem wird das Molekül angezeigt, im anderen kannst du Kommandos ausführen.

Um die Moleküle wie in der Abbildung im Ball-Stick-Modell darzustellen, klicke im Kontrollfenster unten links unter Draw Mode auf Solid und anschließend auf Ball&Stick. Zum Beenden des Programmes klicke auf den Totenkopf in der Mitte des Kontrollfensters. Hast du ein Movie geöffnet, so kannst du dieses Klicken auf "Movie" oben links ansehen. Bindungsabstände und -winkel kannst du rechts unter Calculate bestimmen. Ansonsten viel Spaß beim Ausprobieren.

L.7.5 jmol

jmol ist eine Alternative zu molden. Das Kommando zum Ausführen ist typischerweise `jm`, `jmol4` (für Version 4) oder `jmol10` (für Version 10), wie üblich mit dem Namen des zu öffnenden Files als Argument.

L.7.6 DataExplorer

Mit dem DataExplorer der Firma IBM können Bilder von Molekülorbitalen, die von PAW erzeugt werden, angesehen werden. Solche Files haben die Endung `.dx`. Starte den DataExplorer mit dem Kommando `dx -image &`. Nun musst du zunächst einige Makros laden. Diese befinden sich unterhalb deiner PAW-Installationsdirectory typischerweise in einer Directory mit dem Namen dx. Evtl. kannst du mit dem Befehl `find` danach suchen, sie haben die Endung *.net*. Zum Laden der Makros führst du unter "File" "Load Macro..." aus. Wechsele in die soeben gefundene Directory, klicke auf "Load all Macros" und schließe dieses Fenster wieder. Mit "File" "Open" öffnest du in derselben Directory das File `dx_wave.net`. Sollten Fehlermeldungen auftreten, kannst du diese ignorieren.

Im Main Control Panel kannst du oben ein .dx-File deiner Wahl laden. Um dies nun zu sehen, musst du im anderen Fenster unter "Execute" "Execute Once" anklicken, oder du benutzt gleich "Execute on Change". Im Data Explorer gibt es zwei Darstellungsmöglichkeiten für Orbitale, nämlich die dir

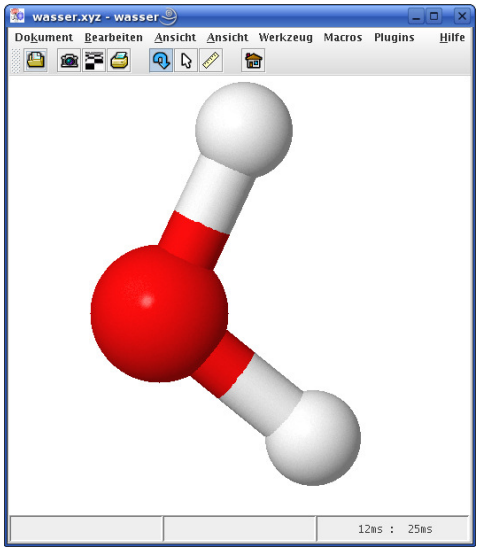


Fig. L.2: Das Programm jmol mit einem Wassermolekül

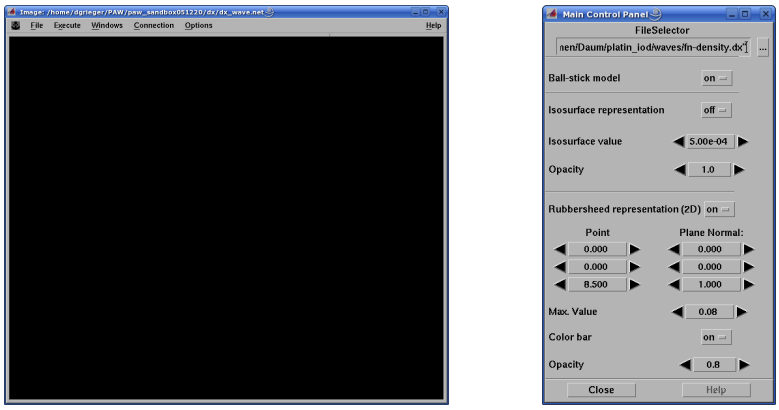


Fig. L.3: DataExplorer ohne geladenes File

wahrscheinlich bekannte Isosurface Representation, in der Flächen gleicher Wahrscheinlichkeitsdichte eingezeichnet werden, und die Rubbersheet Representation, bei welcher in zwei Dimensionen die Wahrscheinlichkeitsdichte in Abhängigkeit vom Ort angezeigt wird. Du kannst die von dir gewünschte im Main Control Panel auswählen. Darunter findest du Parameter, die du ebenfalls entsprechend anpassen kannst.

L.7.7 MSModeling

MSModeling ist ein Zeichenprogramm zur Konstruktion von Molekülen und mehr. Will man ein System mit PAW berechnen, braucht man ein Strukturfile, in dem die einzelnen Atome und deren Koordinaten stehen. Diese von Hand zu erzeugen klappt für Systeme wie H_2O oder CO_2 (wohlbemerkt nur ein einziges Molekül im System!), jedoch stößt man sehr schnell an die Grenze, wo dieses nicht mehr möglich ist. Benutzt man ein Zeichenprogramm geht dies viel schneller und effektiver. Durch die eingebaute Clean-Funktion können Handskizzen in relativ gute Strukturen umgewandelt werden.

Leider ist MSModeling ein Windows-Programm, so dass du es auf dem Windows-Rechner `balkan` ausführen musst. Von deinem eigenen Rechner aus kannst du dich mit dem Kommando `rdesktop -a24 -g800x600 balk`

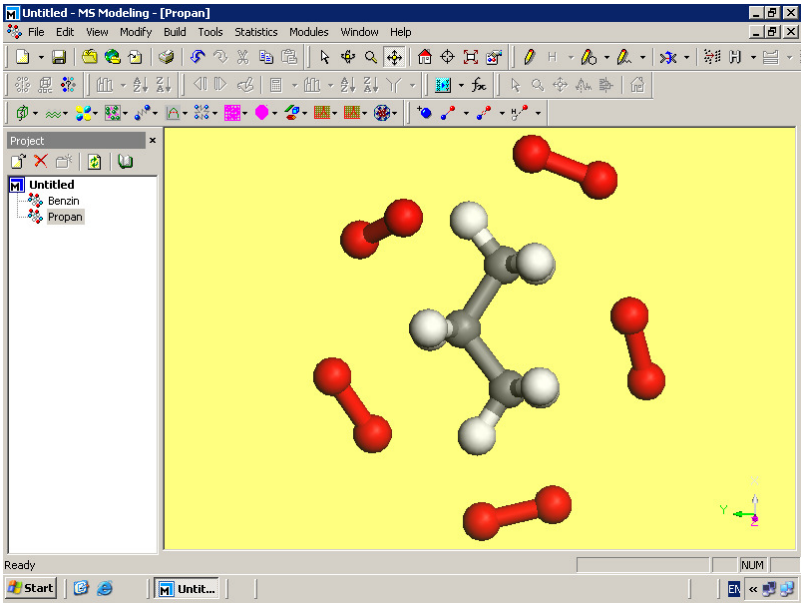


Fig. L.4: MSModeling mit einer einfachen Struktur

auf `balkan` anmelden. Für `balkan` brauchst du ein eigenes Passwort. Nun kannst du MSModeling starten. Beim Starten erscheint evtl. eine Fehlermeldung, auf die du mit nein antworten kannst. Zeichne nun mit MSModeling deine Struktur. Neben der schon erwähnten Clean-Funktion gibt es z.B. noch die Funktion "Adjust Hydrogen", mit der alle noch offenen Bindungen mit Wasserstoff besetzt werden.

Exportiere die fertige Zeichnung in das Format .car. Dieses File kannst du nun mit Hilfe des Programmes WinSCP auf deinen eigenen Rechner kopieren. Zurück auf deinem eigenen Rechner kannst du dieses .car-File mit dem zu PAW gehörenden Programm `paw_tostrc.x` in ein .prestrc-File umwandeln (`paw_tostrc.x` *`name`*`.car`). Dieses .prestrc-File enthält genau die Daten, die du benötigst, um daraus ein .strc-File für eine PAW-Rechnung zu machen.

Appendix M

Vibrational Frequencies

Lernziele: Gesamtenergieoberfläche, klassische Schwingungsfrequenz, Programmierung eines einfachen Programms, Ab-initio Molekulardynamik.

M.1 Learning goals

- Point-by-Point exploration of the total energy surface
- vibrational frequencies from the total energy surface
- vibrational frequencies and dynamics
- mass renormalization

We will determine the vibrational frequency of the CO molecule once with static calulations and once using dynamics. Later we will analyze the eigenmodes of the water molecule.

M.2 Background:

M.3 Berechnung der Schwingungsfrequenz aus dem Morsepotential

Das einfachste Modell für ein Molekül ist ein System von Massepunkten, welche durch Federn miteinander verbunden sind. Für ein zweiatomiges Molekül erhält man für dieses Modell ein Parabelpotential der Form

$$V(R) = \frac{1}{2}c(R - R_e)^2$$

mit dem Gleichgewichtsabstand R_e und der Federkonstanten c. Daraus ergibt sich eine Schwingungsfrequenz ω_0 mit

$$\omega_0 = \sqrt{\frac{c}{\mu}}$$

wobei für die reduzierte Masse μ gilt:

$$\frac{1}{\mu} = \frac{1}{m_1} + \frac{1}{m_2}$$

Dieses Potential beschreibt die Schwingung für kleine Auslenkungen recht gut, ein Fehler ist aber insbesondere, dass das Potential für große Auslenkungen unendlich wird und nicht, wie zu erwarten, gegen die Dissoziationsenergie konvergiert. Ein Potential, welches dies korrekt beschreibt, ist das empirisch gefundene Morsepotential

$$V(R) = V_D \cdot \left(1 - \mathrm{e}^{-\alpha(R-R_e)}\right)^2$$

wobei V_D die Dissoziationsenergie ist. Dabei hängt die Konstante α mit der Schwingungsfrequenz wie folgt zusammen:

$$\alpha = \sqrt{\frac{\mu}{2V_D}}\,\omega_0$$

Hier ist der Nullpunkt des Potentials gerade die Energie des Moleküls im Gleichgewichtsabstand.

Um die klassische Schwingungsfrequenz des einfachen linearen Moleküls CO aus dem Morsepotential zu bestimmen, muss zunächst der Gleichgewichtsabstand R_e und die Dissoziationsenergie D berechnet werden. Den Gleichgewichtsabstand kannst du durch eine einfache PAW-Rechnung bestimmen: Das Strukturfile für CO ist schnell geschrieben:

```
!STRUCTURE
  !GENERIC LUNIT=1.8897259926 !END
  !OCCUPATIONS NSPIN=1 EMPTY=5 !END

  !LATTICE T=10.0  0.0  0.0
              0.0  6.0  0.0
              0.0  0.0  6.0
  !END

  !SPECIES NAME='C_' ZV=4. npro=2 2 1 lrhox=2
      FILE='/home/dgrieger/PAW/Setups_031213/PBE/006C/c_.75_6.0.out' !END
  !SPECIES NAME='O_' ZV=6. npro=2 2 1 lrhox=2
      FILE='/home/dgrieger/PAW/Setups_031213/PBE/008O/o_.75_6.0.out' !END

  !ATOM NAME='C_1' R= 0.0 0.0 0.0 !END
  !ATOM NAME='O_2' R= 1.0 0.0 0.0 !END

  !ISOLATE NF=3 RC=0.5 RCFAC=1.5 GMAX2=3.0 DECOUPLE=T !END

  !CONSTRAINTS
     !TRANSLATION  !END
     !ROTATION     !END
  !END
!END
!EOB
```

Die weitere Vorgehensweise ist analog zu den vorhergehenden Versuchen. Die Rechnungen sollten allesamt sehr schnell ablaufen. Eventuell musst du die `!ROTATION`-Zwangsbedingung hinausnehmen, falls PAW beim Loslassen der Kerne eine Fehlermeldung liefert, dass die Zwangsbedingungen nicht angewendet werden können. Ist die Struktur vollständig konvergiert, kannst du mit dem Befehl `strc2xyz -o` ein .xyz-File erzeugen und daraus mit jmol, molden oder CryMolCAD den Gleichgewichtsabstand bestimmen.

Die Dissoziationsenergie erhältst du folgendermaßen: Starte zwei weitere PAW-Rechnungen, in welchen du jeweils nur ein C-Atom bzw. nur ein O-Atom im Strukturfile angibst. Wenn beide Rechnungen konvergiert sind, erhältst du für beide Atome einzeln einen Wert für die Gesamtenergie.

Addierst du beide Werte, entspricht dies der Energie des dissoziierten Moleküls. Die Dissoziationsenergie ist also die Differenz aus dieser Energie und der aus der vorhergehenden Rechnung ermittelten Gesamtenergie des CO-Moleküls. Damit hast du D.

M.3.1 Die !BOND-Zwangsbedingung

Nun hast du zwei der drei in das Morsepotential eingehenden Parameter bestimmt. Um den dritten zu ermitteln, kannst du nun beispielsweise den Abstand der beiden Atome mit einem konstanten Wert festhalten. Hierfür benötigst du die !BOND-Zwangsbedingung. Diese kannst du im !CONSTRAINTS-Block wie folgt aktivieren, hier als Beispiel für einen vorgegebenen Bindungsabstand von 1,5 atomaren Einheiten:

```
!CONSTRAINTS
   !BOND ATOM1='C_1' ATOM2='O_2' MOVE=T VALUE=1.5 NSTEP=150 SHOW=T !END
   !TRANSLATION  !END
   !ROTATION     !END
!END
```

Die !TRANSLATION- und !ROTATION-Zwangsbedingung bleibt davon selbstverständlich unverändert. Hier die Erklärungen der einzelnen Parameter der !BOND-Zwangsbedingung:

ATOM1 ist der Name des ersten an der feszuhaltenden Bindung beteiligten Atoms, wie er im !ATOM-Block angegeben ist.

ATOM2 ist der Name des zweiten Atoms.

VALUE gibt den Bindungsabstand in atomaren Einheiten (Bohr-Radien, $a_0 = 0,529$ Ångstrom) an, den du erreichen möchtest.

NSTEP gibt an, in wie vielen Rechenschritten der neue Abstand erreicht werden soll. Du solltest hier nicht zu wenige Schritte angeben, da sonst evtl. das Elektronensystem nicht den Kernen folgen kann und somit das Molekül instabil wird. 150 ist hier ein praktikabler Wert.

SHOW gibt an, dass der momentan erreichte Bindungsabstand und die dazu verwendete Kraft in das File mit der Endung "_constr.report" eingetragen werden soll. Dazu später mehr.

MOVE bestimmt, ob die Atome auf ihre neuen Plätze bewegt werden sollen (für MOVE=T) oder lediglich auf ihren alten festgehalten werden sollen (für MOVE=F), hierfür brauchst du VALUE und NSTEP nicht angeben. MOVE=F kannst du auch benutzen, um dir den aktuellen Bindungsabstand anzeigen zu lassen (wenn du das nicht schon mit beispielsweise "jmol" getan hast), da er mit SHOW=T ins _constr.report-File geschrieben wird.

Es ist sinnvoll, bevor du die Zwangsbedingungen anwendest, eine neue Directory anzulegen, in die du dein Control-File, das neue Struktur-File und, um mit der konvergierten Struktur weiterrechnen zu können, das Restart-File (also das File mit der Endung ".rstrt") der vorangegangenen Rechnung kopierst. Starte nun in dieser neuen Directory die Rechnung (mit START=F) für einen Atomabstand, der dir sinnvoll erscheint (z.B. der 1,5-fache Gleichgewichtsabstand). Achte dabei aber darauf, dass deine Einheitszelle ausreichend groß ist. Es wird, wie schon erwähnt, ein Constraints-Report-File (mit der Endung "_constr.report") angelegt, welches du dir wie das Protokollfile mit `tail -f` ansehen kannst. Dort siehst du für jeden Rechenschritt sowohl den momentan erreichten Bindungsabstand als auch die dafür notwendige Zwangskraft, welche nach

$$\vec{F} = -\text{grad}\, V$$

die negative Ableitung des Potentials ist.

Wenn diese Rechnung konvergiert ist, erhältst du aus der errechneten Gesamtenergie (abzüglich der Gleichgewichtsenergie) einen Wert für das Morsepotential an einer bekannten Stelle. Nach kurzer Rechnung erhältst du daraus den noch unbekannten Parameter α und daraus die Eigenfrequenz von CO.

M.4 Exkurs: Programmierung in Fortran 90/95

Wenn du nun den realen Potentialverlauf (also den nicht durch ein Modellpotential genäherten Verlauf) darstellen möchtest, hast du hierfür eine sehr einfache Möglichkeit: Benutze die `!BOND`-Zwangsbedingung nacheinander für mehrere vorgegebene Abstandswerte. Du erhältst als Ergebnis Werte für das Potential und dessen Ableitung (die negative Kraft) für diese Werte. Hast du genügend Werte berechnet, musst du das Potential zwischen diesen Werten interpolieren. Das kannst du beispielsweise mit einem kleinen Fortran-Programm erledigen. Hierin erhältst du nun eine kleine Einführung:

In unserer Arbeitsgruppe wird zur Programmierung die Programmiersprache Fortran in ihrer Version aus dem Jahre 1990 bzw. 1995 (diese unterscheiden sich nicht wesentlich) verwendet. Ein erstes Fortran-Programm hast du bereits in Abschnitt L.7.3 gesehen. Dort hast du ebenfalls gelernt, wie man es kompilieren kann.

Eine Interpolation kann (im einfachsten Fall) mit Polynomen dritter Ordnung geschehen, die an den berechneten Stellen (Numeriker sprechen von Stützstellen) in Funktionswert und 1. Ableitung mit den berechneten Werten (Stützwerten) übereinstimmen. Seien also x_i deine (der Größe nach geordneten) Stützstellen und f_i bzw. f_i' die zugehörigen Werte bzw. Ableitungen. Für das Intervall $[x_i, x_{i+1}]$ sei das interpolierende Polynom P_i gegeben durch

$$P_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

Durch Einsetzen erhält man für die Koeffizienten

$$\begin{aligned} a_i &= f_i \\ b_i &= f_i' \\ c_i &= 3\frac{f_{i+1}-f_i}{(x_{i+1}-x_i)^2} - \frac{f_{i+1}'+2f_i'}{x_{i+1}-x_i} \\ d_i &= 2\frac{f_i-f_{i+1}}{(x_{i+1}-x_i)^3} + \frac{f_{i+1}'+f_i'}{(x_{i+1}-x_i)^2} \end{aligned}$$

Hier nun das Fortran-Programm:

```
PROGRAM INTERPOLATION
IMPLICIT NONE

real(8), DIMENSION(256) :: x, f, fprime, c, d
real(8)                 :: t, y
real(8), PARAMETER      :: inc = 1d-3
integer(4)              :: i = 1, j, n

open(unit=25, file='interpolation.dat', status='replace')

write(*,'(A$)') 'Zahl der Stuetzstellen (max. 256): '
read(*,*) n

DO i = 1, min(n, 256)
   write(*,'(I3,A$)') i, '. Stuetzstelle:  '
   read(*,*) x(i)
   write(*,'(I3,A$)') i, '. Funktionswert: '
```

```
      read(*,*) f(i)
      write(*,'(I3,A$)') i, '. Steigungswert: '
      read(*,*) fprime(i)

      IF (i .eq. 1) CYCLE

      c(i-1) = 3 * (f(i) - f(i-1))/((x(i) - x(i-1))**2) - &
&       (fprime(i) + 2*fprime(i-1))/(x(i) - x(i-1))
      d(i-1) = 2 * (f(i-1) - f(i))/((x(i) - x(i-1))**3) + &
&       (fprime(i) +  fprime(i-1))/((x(i) - x(i-1))**2)

      t = 0
      DO j = 1, INT((x(i) - x(i-1)) / inc)
         y = f(i-1) + fprime(i-1) * t + c(i-1) * t**2 + d(i-1) * t**3
         write(25, '(2F8.4)') t + x(i-1), y
         t = t + inc
      END DO
   END DO

   close(unit=25)
   STOP
   END PROGRAM INTERPOLATION
```

Hier noch einige Erläuterungen zu diesem Programm. Wenn du noch nie programmiert hast, sind diese Erläuterungen für dich wahrscheinlich nicht ausreichend. (Literaturangabe?)

- Bei vielen Fortran-Programmen beginnt (fast) jede Zeile mit sechs Leerzeichen. Dies ist eine Konvention, die im Wesentlichen noch aus der Lochkartenzeit stammt, sie muss heute nicht mehr beachtet werden.
- Es ist in Fortran egal, ob du eine Anweisung in Groß- oder in Kleinbuchstaben schreibst. Historisch wurden ausschließlich Großbuchstaben verwendet, was man heute noch ab und zu findet.
- Jedes Fortran-Programm beginnt mit `PROGRAM name_des_programms` und endet mit `END PROGRAM name_des_programms`
- Mit der Anweisung `STOP` beendest du die Ausführung eines Programmes.
- `write(...,...)` schreibt Daten auf den Bildschirm oder in Files. Der erste Parameter ist dabei der Ort, wohin du schreiben möchtest: `*` für den Bildschirm, eine Zahl für ein File. Der zweite Parameter beschreibt das Format. Gibst du auch hier `*` an, benutzst du das Standardformat.
- `read(...,...)` liest Daten. Die Parameter haben hier die gleiche Bedeutung wie bei `write`
- Für dieses Programm wurden Variablen der Typen `integer(4)` (ganze Zahlen mit 4 Byte Genauigkeit, also von $-2^{31}+1$ bis $+2^{32}-1$) und `real(8)` (reelle Dezimalzahlen mit 8 Byte Genauigkeit, also etwa 16 Nachkommastellen). `dimension(256)` erzeugt dabei ein Array mit 256 Einträgen (ein Array ist eine Sammlung von in diesem Fall 256 aufeinander folgenden Variablen), alternativ könnte auch an den Variablennamen `(256)` angehängt werden. Das beispielsweise fünfte Element des Arrays `c` erhält man dann durch `c(5)`
- `IMPLICIT NONE` bedeutet, dass jede Variable deklariert werden muss, ohne diese Anweisung würde Fortran je nach Namen der Variable automatisch annehmen, dass es sich um integer- bzw. real-Variablen handelt, was eine häufige Fehlerquelle ist.
- Die Angabe `1d-3` bedeutet $1 \cdot 10^{-3}$.

- Mit `open` öffnest du ein File zum daraus Lesen oder Hineinschreiben, mit `close` schließt du es wieder. Die Angabe `unit=` ist die Zahl, die als erster Parameter für `read` oder `write` benötigt wird.
- `DO` kennzeichnet eine Schleife. Die Syntax ist `DO VARIABLE=ANFANGSWERT, ENDWERT`. Eine Schleife ist eine wiederholte Ausführung eines Programmteils. Dieser wird genau so oft wiederholt, dass die Variable im ersten Durchlauf den Anfangswert, im letzten den Endwert annimmt und nach jedem Durchlauf der Wert um 1 erhöht wird. Dabei müssen sowohl Anfangswert als auch Endwert ganze Zahlen sein. Dafür sorgt die Funktion `INT`, indem sie Nachkommastellen einfach abschneidet.

Wie du sicher schnell erkannt hast, erzeugt dieses Programm ein File interpolation.dat, welches du mit xmgrace ansehen kannst.

Übung 3

Ändere dieses Programm so ab, dass die Interpolation nicht mit Polynomen 3. Ordnung, sondern lediglich mit Geradenstücken, also Polynomen 1. Ordnung, erfolgt. Hierfür kannst du die 1. Ableitung an den Stützstellen nicht verwenden.

M.5 Berechnung der Schwingungsfrequenz aus der Schwingung selbst

Im Folgenden soll die Schwingungsfrequenz nicht aus einem theoretischen Potentialverlauf heraus bestimmt werden, sondern daraus, dass die Atome ausgelenkt werden und anschließend beobachtet wird, wie sich der Abstand verhält. Du musst also die Atome zunächst auslenken. Dazu erzeugst du wieder eine neue Directory und berechnest dort eine Struktur, in der du mit der `!BOND`-Zwangsbedingung einen Atomabstand vorgibst, der größer als der Gleichgewichtsabstand, aber auch nicht zu groß ist, höchstens etwa das 1,5-fache des Gleichgewichtsabstandes. Ist diese konvergiert, gehe folgendermaßen vor:

- Schalte die Reibung aus (also `FRIC=0.0`)
- Setzt, falls nocht nicht geschehen, die tatsächlichen Massen der Atome ein (also im `!SPECIES`-Block die Angabe `M=` durch ein `_off` auskommentieren oder ganz wegnehmen).
- Schalte die Zwangsbedingung aus (also mit `!BOND_off` auskommentieren).
- Schalte die Massenrenormalisierung ein (siehe unten).

Da es sich hier um eine Dynamiksimulation handelt, müssen die Atommassen renormalisiert werden. Deine `!SPECIES`-Blöcke sollten also folgendermaßen aussehen:

```
!SPECIES NAME='C_' ZV=4. npro=2 2 1 lrhox=2
    FILE='/home/dgrieger/PAW/Setups_031213/PBE/006C/c_.75_6.0.out'
    PS<G2>=5.64386 PS<G4>=20.34949 !END

!SPECIES NAME='O_' ZV=6. npro=2 2 1 lrhox=2
    FILE='/home/dgrieger/PAW/Setups_031213/PBE/008O/o_.75_6.0.out'
    PS<G2>=15.18367 PS<G4>=92.08135 !END
```

Die Parameter `PS<G2>` und `PS<G4>` sorgen für die Massenrenormalisierung. Die zu verwendenden Zahlen erhältst folgendermaßen: Gehe in die Directory, in der deine Atom-Setups für das gewünschte Atom liegen (also in diesem Beispiel für den Sauerstoff `~/PAW/Setups_031213/PBE/008O/`), öffne

dort das File mit der Endung `.prot` und suche nach `G**2` bzw. `G**4` (typischerweise in Form von `SUM:F*<PS-PSI|G**2|PS-PSI>`). In dieser Zeile findest du den Parameter.

Du erkennst nun schon am Temperaturverlauf deiner Rechnung, dass das Molekül eine Schwingung ausführt. Lasse die Rechnung etwas laufen (10.000 Schritte sollten praktikabel sein). Hast du die Rechnung beendet, kannst du dir mit Hilfe des dir schon bekannten Programms `paw_tra.x` den Bindungsabstand gegen die Zeit plotten lassen. Benutze dazu ein wie folgt aussehendes `.tcntl`-File:

```
!TCNTL
   !MODE ID='C1O2'
      !BOND
        ATOM1='C_1'
        ATOM2='O_2'
        SCALE=1.
      !END
   !END
   !OUTPUT ID='C1O2' FILENAME='c1o2.xyz' !END
!END
!EOB
```

Um dir mit `paw_tra.x` einen Bindungsabstand oder einen Bindungswinkel gegen die Zeit auftragen zu lassen, benötigst du `!MODE`. Deiner "Mode" musst du nun mit `ID=` einen Namen geben (dieser kann beliebig sein), so kannst du mit einem `.tcntl`-File durch Angabe von verschiedenen !MODE-Blöcken verschiedene Werte plotten lassen. Um anzugeben, dass du einen Bindungsabstand plotten möchtest, benötigst du `!BOND`, gefolgt von den beiden an der Bindung beteiligten Atomen (wie bei der gleichnamigen Zwangsbedingung). Wolltest du alternativ einen Bindungswinkel plotten lassen, müsstest du hier `!ANGLE` angeben, gefolgt von drei Atomen. Es folgt noch die Angabe eines Faktors `SCALE`, mit diesem werden die Bindungsabstände multipliziert.

Die letzte Anweisung ist `!OUTPUT`. Hiermit gibst du mit `FILENAME` den Namen des zu erzeugenden Files zu dem mit `ID` angegebenen Plot an. Wenn du mehrere `!MODE`-Einträge hast, kannst du auch `!OUTPUT` mehrfach verwenden.

Manchmal kann auch `!SEQUENCE` hilfreich sein. Diese Anweisung benötigst du, wenn du nicht alle (im File mit der Endung `_r.tra`) zur Verfügung stehenden Daten, sondern nur einen Ausschnitt davon plotten möchtest. Dies sieht etwa so aus:

```
!SEQUENCE T1[PS]=0.753 T2[PS]=1.549 DT[FS]=100 !END
```

Mit `T1[PS]` und `T2[PS]` legst du den Ausschnitt fest (Simulationszeit in Picosekunden). `DT[FS]` kann weggelassen werden, es gibt die Schrittweite zwischen zwei zu zeichnenden Punkten an. Dies ist beispielsweise nützlich, wenn du sehr viele Daten hast und keine allzu großen Files erzeugen möchtest. Übrigens kannst du `!SEQUENCE` auch beispielsweise für Movies verwenden.

Das erzeugte XYZ-File kannst du nun beispielsweise mit xmgrace ansehen. Aus der Differenz zweier oder mehrerer Schwingungen erhältst du die Schwingungsdauer und daraus die Kreisfrequenz.

Übung 4

Führe diese Dynamiksimulation ohne Massenrenormalisierung (also ohne die Parameter `PS<G2>` und `PS<G4>`) aus. Wie ändert sich die Schwingungsfrequenz?

Appendix N

Versuch 5: Infrarot-Spektroskopie von CO

Programmierung eines einfachen quantenmechanischen Programms. Dipolmomente, Obertöne, ir aktive und ir passive Moden.

Schrittweite, Massenrenormalisierung, Elektronenreibung

1. Bestimmen Sie die Wellenfunktionen eines harmonischen oszillators.
2. Bestimmen sie die Dipolmatrixelemente

Appendix O

The computer, the stranger next to you

O.1 Learning goals

This section is meant for the people who have very little experience with computers of who are new to the Unix operating system.

- Steuerung über die Konsole
  - Datei- und Verzeichnisoperationen
  - Wo finde ich Hilfe?
  - wichtige Tools (grep, tail, ...)
- Shellscripte

O.2 Preparation

I recommend to anyone working on Unix systems to have the book "Linux in a Nutshell"[144] on your desk. It is a very compact reference of Linux commands, which are nearly identical on all Unix operating systems including OS-X.

It is expected that you know how to switch a computer on and off, and to log in and log out and that you have been instructed about changing passwords. If you are not familiar with these steps obtain an instruction from your system administrator.

O.3 Linux, the operating system

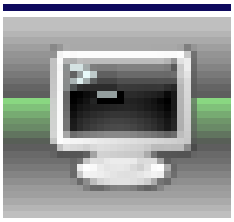
The operating system is the basic language of the computer which allows the user to interact with the hardware of the computer. If you want to create, view, change, remove or print a file, you can tell it using commands from the operating system. If one wants to execute a program, one has to tell the computer using commands of the operating system. A program itself interacts at first with an operating system.

The most common operating systems can be divided into Unix-based operating systems such as Linux and Mac OS-X on the one side and the Microsoft Windows operating system on the other side. We are using Unix based operating systems such as Linux or OS-X.

In the following a brief introduction to the LINUX operating system will be given. While the beginner will be used to execute programs by mouse-clicks, we will use more by explicitely typing commands into a terminal.[1] At first this seems old-fashioned and cumbersome, you will find out that this is far more efficient once you get used to it. However, it requires that you know some of the Linux commands. There are not that many that you will use regularly. Nevertheless it is adviseable to have a reference book on your desk when working. The reference book I use myself is "Linux in a Nutshell" by O'Reilly.

If you are working with OS X, the operating system of Apple, you will find little differences. Both Linux and OS-X belong to the same class of operating systems.

You open a terminal application either by mouse clicking on the terminal icon, which may look like a old-fashioned computer screen.



Sollte dieses Symbol bei dir nicht vorhanden sein, findest du die Konsole auch im Startmenu (das Icon ganz links in der Startleiste) unter System -> Terminals. Sollte dies alles fehlschlagen, wähle "Befehl ausführen" aus dem Startmenu und tippe `konsole`.

If you are using OS-X you may start with Terminal.app. It is however recommended that you have someone install the program iTerm.app (`http://iterm.sourceforge.net/`) instead.

O.3.1 Directories und Files

Wie bei Windows sind die wesentlichen Einheiten Files (Dateien), welche in Directories (Verzeichnissen) organisiert sind. Jedoch kennt Linux nicht das Konzept von Laufwerken zur Organisation von Partitionen u. Ä., sondern alle Files und Directories sind in einer Directory, der sogenannten Root Directory enthalten. Desweiteren besitzt jeder Benutzer eine spezielle Directory, auf die er Zugriff hat (auf die meisten anderen Directories können nur Administratoren zugreifen). Für diese Directories gibt es Kürzel:

~	Home Directory
/	Root Directory
.	Aktuelle Directory (Working Directory)
..	Directory, welche die aktuelle Directory enthält, also das in der Hierachie eins drüber stehende.

Möchtest du Pfade angeben, werden die Directories, anders als unter Windows, mit einem / getrennt. Beispiel für einen Pfad wäre also `/usr/bin` für die Directory bin unterhalb der Directory usr in der Root-Directory.

Mit den folgenden Kommandos kannst du dich im Filesystem bewegen:

pwd sagt Dir, in welcher Directory Du Dich befindest. (Print Working Directory)

ls gibt Dir eine Ansicht von Files in deinem aktuellen Verzeichnis. (LiSt)

[1] A terminal is a window, that allows to type text commands and that may produce textual output. This is the most primitive interface that allows you to communicate with the computer. In the old days, that is before Apple and Xerox Palo Alto research center invented windows, the entire computer screen was nothing but a terminal window, which explains the name and the icon.

Linux Kommandos können von Argumenten und Optionen gefolgt sein. Am Beispiel von `ls` ist das Argument die Directory welche Du Dir ansehen möchtest. Zum Beispiel kannst du das Kommando `ls /` verwenden, um den Inhalt der Root-Directory anzusehen. Möchtest du den Inhalt einer Directory in der Root-Directory, z.B. der Directory `bin`, ansehen, tippst Du `ls /bin`. Beachte dass `ls bin` nach einer Directory in deiner aktuellen Directory sucht, also nach ./bin.

Optionen sind spezielle Argumente, welche mit einem Minuszeichen beginnen. Für das Kommando `ls` gibt es zum Beispiel die Option `-l`. Als Kommando sieht das beispielsweise so aus: `ls -l`. Damit gibt `ls` zusätzlich Auskunft über die Zugangsberechtigungen (mehr dazu später), den Besitzer, die Größe der Files, das Datum und die Uhrzeit der letzten Änderung. Eine weitere Option ist `-a`. Diese macht auch Files sichtbar, deren Name mit einem Punkt beginnt, und die damit sonst unsichtbar sind. Die beiden Optionen können als `ls -al` kombiniert werden. Verwende den Skrollbar auf der rechten Seite um die ganze Liste zu sehen.

ll, dir sind gleichbedeutend mit `ls -l` (sog. alias), allerdings nicht immer verfügbar.

mkdir Tree erzeugt eine Directory mit Namen Tree. Achte darauf, dass Linux - im Gegensatz zu Windows - zwischen Groß- und Kleinschreibung unterscheidet! (MaKe DIRectory)

cd Tree bringt Dich in die neue Directory. Als Parameter kannst du selbstverständlich jede beliebige Directory angeben. Als Kurzformen gibt es: `cd` ohne Parameter bringt dich in deine Home-Directory, `cd -` bringt dich in die Directory zurück, in der du vorher warst. (Change Directory)

rmdir Tree löscht die neue Directory wieder, allerdings nur, wenn sie leer ist. Zum Löschen von nicht-leeren Directories siehe den `rm`-Befehl weiter unten. (ReMove DIRectory)

O.3.2 Hilfe!

man Kommando liefert dir Hilfe zu jedem beliebigen Kommando (die sogenannte Manpage). Falls du also z.B. wissen möchtest, wie genau man das `ls`-Kommando benutzt, tippe `man ls`. Beenden kannst du diese Anzeige entweder mit `q` oder mit Strg-C. Übrigens kannst du fast jeden Befehl mit Strg-C wieder beenden. (MANual)

Eine allgemeine Bemerkung: Sollte ein Kommando nicht so funktionieren, wie du es wünschst, ist es oft schon hilfreich, die auftretende Fehlermeldung genau zu lesen.

O.3.3 File-Operationen

rm Filename löscht das File mit dem Namen *Filename*. `rm` hat beispielsweise die Option `-i` (interactive), die bewirkt, dass du nochmal gefragt wirst, bevor ein File gelöscht wird. Mit der Option `-r` (recursive) kannst du Directories löschen und zwar selbst dann, wenn noch Files darin enthalten sind. Diese Option kann allerdings gefährlich werden, weil man damit sehr viele Files auf einmal löschen kann. Stelle dir z.B. vor, du befindest dich in deiner Home-Directory und möchtest eine Directory .bla löschen, vertippst dich aber, so dass dein Kommando `rm -r . bla` lautet. Damit hast du alle deine Daten gelöscht. (ReMove)

Übrigens kannst du, wie bei allen Kommandos, Teile eines Filenamens durch ein '*' ersetzen. [rm -i *.strc] löscht beispielsweise alle Files mit der Endung .strc

mv alterName neuerName benennt ein File oder eine Directory um. (MoVe)

mv File Directory verschiebt ein (oder mehrere) File(s) in eine Directory.

cp alterName neuerName kopiert ein File. Möchtest du eine Directory mit allen darin enthaltenen Files kopieren, benötigst du die Option `-r`. (CoPy)

touch Filename erzeugt ein leeres File mit dem angegebenen Namen. Existiert das File mit dem angegebenen Namen bereits, setzt `touch` das Datum und die Uhrzeit der letzten Änderung auf die aktuelle Uhrzeit.

cat Filename zeigt den Inhalt eines (oder mehrerer) Files an. (conCATenate)

more Filename, less Filename tun im Prinzip das gleiche wie `cat`, jedoch kann man hier mit der Leertaste weiterblättern, wenn der gesamte Inhalt des Files nicht auf einmal in das Konsolenfenster passt. `cat` würde den gesamten Inhalt auf einmal anzeigen. Beenden kannst du diese Anzeige wiederum mit `q` oder Strg-C.

tail Filename zeigt die letzten 10 Zeilen eines Textfiles an. Solltest du eine andere Anzahl von Zeilen sehen wollen, kannst du dies als Option übergeben, also etwa `tail -30` *Filename* für 30 Zeilen. Oft wird dieses Kommado mit der Option `-f` benutzt, in diesem Fall wartet `tail`, bis am Ende des Files eine Zeile hinzugekommen ist, zeigt diese dann an und wartet weiter. Abbrechen lässt sich dies mit STRG-C. Nützlich ist dies etwa für Protokollfiles, wie sie auch von PAW erzeugt werden, somit kannst du sofort sehen, wann PAW etwa wieder einen Rechenschritt gemacht hat.

head Filename ist äquivalent zu `tail`, zeigt jedoch die ersten statt die letzten Zeilen eines Files an.

grep Text Filename zeigt aus dem angegebenen File/den angegebenen Files nur die Zeilen an, die den angegebenen Text enthalten. `grep` kennt als Option `-i` (Insensitive), dabei wird auf Groß- und Kleinschreibung keine Rücksicht genommen.

find Directory -name Filename sucht unterhalb der angegebenen Directory nach einem File mit dem angegebenem Namen und gibt den vollständigen Pfad dahin aus. Beim Filenamen können auch Wildcards verwendet werden, zum Beispiel liefert `find . -name *.net` alle Files unterhalb der Working Directory, die die Endung `.net` haben.

lpr -P Drucker file druckt ein File vom Typ PostScript (ps). Als Drucker kannst du den im Kopierraum stehenden Drucker ir2200 verwenden. (Das `-P` ist ein großes P.)

a2ps -P Drucker file druckt ein Text-File, also z.B. Programm-Quellcode, PAW-Steuerungsfiles etc. (Das `-P` ist ein großes P.) (Ascii TO PostScript) Ersetzt du `-P` *`Drucker`* durch `-o` *`filename`*`.ps`, kannst du "in ein File drucken", d.h. du erhältst aus deinem Text-File ein Postscript-File. Da die Ausgabe von `a2ps` standardmäßig zweispaltig ist, hier als Beispiel das Kommando für einspaltigen Druck, und zwar um das File `case.pdb` in das Postscript-File `case.ps` umzuwandeln:
`a2ps -R -o case.ps case.pdb --chars-per-line=120 --columns=1`
`a2ps` kennt noch viele weitere Optionen. Für Details siehe in der zugehörigen manpage.

O.3.4 Zugriffsrechte

chmod ändert die Zugriffsrechte für ein File. (CHange MODe) Normalerweise kann unter Linux nicht jeder Benutzer eines Computers beliebig auf ein File zugreifen, sondern dies ist durch Zugriffsrechte (modes) geregelt. Man kann die folgenden Zugriffsrechte verteilen:

- `r` ein File bzw. den Inhalt einer Directory lesen (Read)
- `w` in ein File schreiben bzw. den Inhalt einer Directory verändern (Write)
- `x` ein File ausführen (eXecute), macht nur bei Programmen und Directories Sinn (Directories müssen ausführbar sein, um darauf zugreifen zu können).

Zugriffsrechte können für die folgenden Benutzergruppen verteilt werden:

- `u` der Besitzer eines Files (User), der vom Kommando `ls -l` angezeigt wird, i.d.R. also derjenige, der das File angelegt hat.
- `g` die Gruppe (Group), zu der der Besitzer gehört, also unsere gesamte Arbeitsgruppe.
- `o` alle Anderen (other).
- `a` alle (All), eine Kurzform für ugo

`chmod` wird nun folgendermaßen benutzt: Möchtest du beispielsweise zusätzlich zu den bereits bestehenden Rechten allen in deiner Gruppe und allen Anderen das Recht geben, ein File zu lesen und in ein File zu schreiben, sieht das Kommando wie folgt aus: `chmod go+rw` *Filename*. Du erkennst: Auf der linken Seite stehen jeweils die Benutzergruppen, für die du Rechte verteilen möchtest, auf der rechten Seite die Rechte und dazwischen ein '+', wenn du Rechte dazugeben möchtest, ein '-', um Rechte wegzunehmen, und ein '=', wenn das File anschließend genau diese Rechte haben soll. Alternativ gibt es auch eine Schreibweise mit dreistelligen oktalen Zahlen, hier steht 4 für read, 2 für write und 1 für execute, die erste Ziffer repräsentiert den user, die zweite die group und die dritte other. Beispielsweise erreichst du mit dem Befehl `chmod 660` *Filename*, dass das angegebene File vom Besitzer und dessen Gruppe gelesen und geschrieben (6=2+4), von allen anderen jedoch weder gelesen noch geschrieben werden kann.

Eine nützliche Option von `chmod` ist `-R`, diese bewirkt, dass, wenn man als Argument eine Directory angibt, automatisch auch alle in der Directory enthaltenen Files geändert werden. (Recursive)

Die Zugriffsrechte für ein File kannst du an den ersten 10 Zeichen der Ausgabe des Kommandos `ls -l` sehen. Steht dort beispielsweise `-rw-r-r-`, handelt es sich um ein File, dass der Besitzer lesen und schreiben kann (Zeichen 2-4), die Gruppe (Zeichen 5-7) und alle Anderen (Zeichen 8-10) nur lesen können. Das erste Zeichen ist für reguläre Files stets ein '-', für Directories ein 'd'.

O.3.5 Ein- und Ausgabekanäle, Pipes

programm >file leitet alles, was programm auf den Bildschirm ausgeben würde (außer Fehlermeldungen) (stdout), in file um. Dies kann z.B. für PAW hilfreich sein, PAW produziert sehr viel Bildschirmausgabe, die in den meisten Fällen nicht weiter relevant ist. So kann man beispielsweise, indem man dem PAW-Kommando ein `>out` nachstellt, diese Ausgabe in das File out umleiten, und falls es Probleme geben sollte, auf dieses File zurückgreifen, anstatt auf dem Bildschirm suchen zu müssen. Solltest du die Ausgabe eines Programmes nicht benötigen, kannst du sie mit `>/dev/null` löschen.

programm <file liest das, was programm von der Tastatur lesen würde (stdin), aus file. Eher selten gebräuchlich.

programm 2>file leitet die Fehlermeldungen von programm (stderr) in file um.

programm1 | programm2 ist eine Konstruktion, die sich pipe nennt. Hierbei erhält programm2 das als Tastaturdaten (stdin), was programm1 auf den Bildschirm ausgeben würde (stdout). Hilfreich ist dies z.B. für die Kommandos `more`, `less` oder `grep`, welche, wenn kein File als Parameter angegeben ist, das benutzen, was über die Tastatur eingetippt wird. Beispiele:

- `ls -l | more` ist nützlich, falls die Ausgabe von [ls] nicht vollständig auf den Bildschirm passt, so kann man sich mit der Leertaste durchbewegen.
- `ls -l | grep .strc` zeigt alle Files an, deren Name .strc enthält.

O.3.6 Shell-Scripts, .bashrc, Umgebungsvariablen

Shell-Scripts sind das Linux-Pendant zu Batch-Files unter Windows. Ein Shell-Script ist also nichts Anderes als ein File, in dem Linux-Kommandos aufgelistet sind, die nacheinander ausgeführt werden. Zum Anfang ein triviales Beispiel:

```
#! /bin/bash

mkdir ../Muell
mv * ../Muell
```

Wie du wahrscheinlich schnell erkannt hast, erzeugt dieses Shell-Script eine Directory Muell in der in der Hierarchie eine Stufe höher gelegenen Directory und verschiebt alle Files in deiner Working-Directory in diese neu erzeugte Directory.

Um Shell-Scripts zu erstellen, nutze z.B. den emacs (siehe unten). Wie du das File nennst, ist egal, im Folgenden heiße es `shellscript`. Um nun dein Shell-Skript auszuführen, muss es noch ausführbar gemacht werden. Das Kommando hierfür kennst du schon: `chmod a+x shellscript`. Ausführen geht jetzt mit `./shellscript`. Dieses überflüssig aussehende `./` am Anfang wird benötigt, damit dein File auch gefunden wird.

Sicherlich ist dir die erste Zeile des Shell-Scripts aufgefallen. Mit solch einer Zeile beginnt jedes Shellscript. Sie hat den folgenden Sinn: Wenn du dem Betriebssystem sagst, du möchtest ein bestimmtes File ausführen, weiß das Betriebssystem per se nicht, ob es sich dabei um ein Script oder ein richtiges Programm handelt. Dass es sich um ein Script handelt, sagen die Zeichen `#!` (die sog. Magic Number). Der Rest der Zeile gibt an, welche Shell (oder allgemeiner: welches Programm) das Script ausführen soll. Shell heißt unter Linux das Programm, welches die Kommandos ausführt, welche du entweder eintippst oder in ein Script schreibst. Hier wird die Bourne Again Shell (bash) benutzt, sie hat den Pfad `/bin/bash`.

- http://www.tfh-berlin.de/ s17779/bash/bashscripting.html
- http://www.tldp.org/LDP/abs/html/

Ein besonderes Shell-Script hat den Namen `.bashrc` und liegt in deiner Home-Directory. Dieses wird jedes Mal ausgeführt, wenn du eine neue Shell startest, also insbesondere, wenn du ein Konsolenfenster öffnest oder ein Shellscript startest. (Dieses muss nicht mit `#! /bin/bash` beginnen und nicht ausführbar sein.)

Eine typische Anwendung von `.bashrc` ist es, Umgebungsvariablen zu setzen. Umgebungsvariablen sind etwas Ähnliches wie Variablen in Programmen, nur dass sie z.B. in einer Konsolensitzung oder in einem Shellscript zur Verfügung stehen. Zum Setzen von Umgebungsvariablen gibt es das Kommando `export`. Möchtest du z.B. der Umgebungsvariablen *test* den Wert *hallo* zuordnen, sieht das so aus: `export test=hallo`. Einige Umgebungsvariablen haben eine speziele Bedeutung: Die Variable `PATH` (in Großbuchstaben) enthält alle Directories, in denen die Shell nach ausführbaren Programmen oder Shell-Scripts suchen soll. Beispiel PAW: Oftmals liegen die PAW-Programme in einer Directory wie z.B. `~PAW/paw_sandbox051220/bin/ifc/`. Möchtest du zum Starten von PAW dies nicht jedes Mal vollständig eintippen müssen, sondern nur das jeweilige Kommando, kannst du z.B. `PATH` folgendermaßen setzen:
`export PATH=~/PAW/paw_sandbox051220/bin/ifc/:$PATH`. (Das `$PATH` am Ende ist das, was schon in `PATH` steht.) Damit du diese Zeile nun nicht jedes Mal eintippen musst, wenn du ein neues Konsolenfenster öffnest, schreibe sie in das File `.bashrc`.

O.4 Editors

Ein Editor ist ein Prorgamm zum Schreiben von Text-Dateien. Der am Institut am häufigsten gebrauchte ist Emacs, aber es gibt noch einige andere.

O.4.1 Emacs

Emacs ist ein frei verfügbarer, unglaublich vielseitiger Editor. Er ist anfangs etwas gewöhnungsbedurftig, aber nach einiger Anlernzeit unschlagbar.

Öffnen und Schließen; Tastenkombinationen

Schreibe `emacs &` in ein Konsolenfenster, was ein emacs Fenster öffnet. (Das `&` am Ende des Kommandos bedeutet, dass du gleich nachdem du dieses Kommando eingetippt hast, gleich wieder ein neues eintippen kannst und nicht erst warten musst, bis emacs wieder beendet ist.)

Nun schließen wir den Editor wieder mit Hilfe der Tastenkombination C-x C-c. Dies benötigt einige Erklärungen: Emacs wird weitgehend durch Tastenkombinationen angesprochen. Hat man diese einmal erlernt, dann ist die Handhabung deutlich schneller als über Menüs. Es gibt zwei spezielle Tasten C steht für die Control-Taste. Auf deutschen Tastaturen heißt diese meist Strg für Steuerung. Die zweite spezielle Taste ist die Alt-Taste. Sie wird durch M- bezeichnet. (Die Abkürzung M- steht für Meta, was nichts Anderes bedeutet.) C-x C-c bedeutet, dass man die Tasten C und x gleichzeitig tippt, und anschließend gleichzeitig die Tasten C und c. Damit schließt sich das emacs Fenster wieder.

Maneuvrieren mit Dired

Jetzt öffne wieder ein emacs-Fenster und starte den directory-editor dired mit Hilfe der Tastenkombination C-x d. Ganz unten im Fenster erscheint eine Zeile die sagt "Dired (directory):" gefolgt von einem Pfad. Diesen Pfad kann man editieren und dann mit einem Return bestätigen. Danach erhält man eine Liste von Directories und Files in der ausgewählten Directory. Man kann nun ein File oder eine Directory auswählen, indem man über den Filenamen ein `f` tippt.

Files können auch direkt ohne dired geöffnet werden. Dies geht mit Hilfe der Tastenkombination C-x C-f, woraufhin emacs nach dem Namen des zu öffnenden Files fragt, den Du mit return bestätigst. Außerdem kannst du Filenamen als Argument übergeben, wenn du emacs startest, also etwa `emacs case.strc`, um ein File `case.strc` zu öffnen.

Editieren und Sichern von Files

Nachdem du ein File mit emacs editiert hast, vergiss nicht, es mit der Tastenkombinatin C-x C-s zu speichern. Möchtest du es unter einem anderen Namen speichern, benutze die Tastenkombination C-x C-w.

Die meisten emacs-Kommandos zum Editieren heißen anders, als du es möglicherweise von Windows gewohnt bist. Z.B. ist C-w cut, M-w copy und C-y paste. Wie du möglicherweise schon gemerkt hast, ist es nicht möglich, eine Stelle im Text mit der Maus zu markieren und sie dann mit der Backspace-Taste zu löschen, hierfür musst du C-w benutzen. Andere nützliche Kommandos sind C-s zum Suchen von Text vorwärts, C-r zum Suchen rückwärts und M-% zum Ersetzen von Text. Hierzu musst du nach der Eingabe von M-% zunächst den alten Text eingegeben, mit return bestätigen, den neuen Text eingeben, wieder mit return bestätigen und dann jede Ersetzung mit y bestätigen oder mit n ablehnen oder du kannst mit ! angeben, dass alle weiteren Ersetzungen ohne Nachfrage durchgeführt werden sollen.

Ein weiteres nützliches Kommando ist C-x r k. Hiermit löschst du nicht den gesamten markierten Text, sondern nur ein Rechteck mit den Eckpunkten Beginn der Markierung und Cursorposition. Mit C-x r y kannst du solch ein Rechteck wieder einfügen.

Es soll noch das Kommando ediff erwähnt werden: Angenommen, du hast zwei ähnliche Files, von denen du die Unterschiede wissen musst. Hierzu kannst du im Emacs das Kommando M-x ediff <Return> benutzen. Du wirst nun nach zwei Filenamen zum Vergleichen gefragt und siehst anschließend beide Files im Emacs. Um nun den ersten Unterschied im File markiert zu bekommen, drücke die Leertaste. So kannst du dich mit der Leertaste durch die Files bewegen. Um das Kommando wieder zu beenden, drücke q.

Appendix P

COSMO

Learning goals: Estimate energies of solvation using the COnductor-like Screening MOdel"

```
!CONTROL
  !GENERIC NSTEP=300 DT=5. START=F  !END
  !FOURIER EPWPSI=30. CDUAL=2.0 !END
  !DFT  TYPE=10 !END
  !PSIDYN  STOP=T
    !AUTO FRIC(-)=0.05 FACT(-)=0.98 FRIC(+)=0.3 FACT(+)=1.00 minfric=0.03 !END
  !END
  !RDYN_OFF STOP=F FRIC=0.01
    !AUTO FRIC(-)=0.0 FACT(-)=1.00 FRIC(+)=0.001 FACT(+)=1.00 !END
  !END
  !COSMO STOP=T M=1000. ADIABATIC=T OPTFRIC=T !END
!END
!EOB
```

Note the solvent radii must be given in atomic units.

```
!STRUCTURE
  !GENERIC LUNIT=1.8897259926  !END
  !OCCUPATIONS  NBAND=8 NSPIN=1 !END
  !LATTICE T=   13.0    0.0    0.0
                0.0   13.0    0.0
                0.0    0.0   13.0 !END
  !SPECIES NAME= 'N_' ID='N_.75_6.0' NPRO=1 1 1 !END
  !SPECIES NAME= 'H_' ID='H_.75_6.0'  NPRO=1 !END
  !ATOM    NAME= 'N_1' R=    -2.519   2.806   0.011     !END
  !ATOM    NAME= 'H_1' R=    -3.337   2.179   0.018     !END
  !ATOM    NAME= 'H_2' R=    -2.798   3.737   0.351     !END
  !ATOM    NAME= 'H_3' R=    -1.786   2.421   0.623     !END
  !COSMO   EPSILON=78.
    !ATOM NAME='N_1'  RAD=2.93 !END
    !ATOM NAME='H_1'  RAD=2.26 !END
    !ATOM NAME='H_2'  RAD=2.26 !END
    !ATOM NAME='H_3'  RAD=2.26 !END
  !END
  !ISOLATE NF=3 RC=0.5 RCFAC=1.5 GMAX=3.0 DECOUPLE=T !END
!END
!EOB
```

Appendix Q

Solids (Replace this!

[1]

Q.1 Learning goals

- K-point sampling: k-points must form an equi-spaced grid, because the background is a Fourier interpolation of the bands in reciprocal space. Specify the k-points in the `strc`-file either by `DIV=` or by `R=`. The k-point density should be about equal in all directions.
- Understand k-points given in the protocol (absolute and relative coordinates).
- Understand convergence with number of k-points.
- Relate real-space vectors with reciprocal space vectors. If one reciprocal vector [2] is much longer than the others, it should be the last vector, for that is assumed in the PAW-Programm for optimal parallelization.
- For metals use in the `cntl`-file `!CONTROL!MERMIN`. (Number of occupied electronic states for different k-points are different. Otherwise total energy is to high.) For metals at zero temperature use `TETRA+`. Understand retardation of energy eigenvalues. For finite temperatures start the occupation-dynamics. Understand that retardation destroys exact energy conservation. Use `SAFEORTHO=F` and understand why. [3]
- Understand true Mermin functional with dynamical occupations. SAFEORTHO=F. One-particle energies are not retarded, but dynamical. Additional term -TS. [4]
- Be able to plot band structures and density of states for solids. Various groups can do different materials, but they should be discussed together. (Maybe a group discussion where the members are asked to simply say something that they can read from a given band structure. Band gap, effective mass, color? etc.) [5]
- Sawtooth problem.[6]

[1] FiXme Note: we need figures of the Brillouin-zones for fcc and bcc with the respective notations
[2] FiXme Note: or lattice vector?
[3] FiXme Note: Wo ist dies beschrieben?
[4] FiXme Note: Wo ist dies beschrieben?
[5] FiXme Note: Was sind Gruppen, wann und wo werden band Gap, effektive Masse und Farbe besprochen?
[6] FiXme Note: Ist das so wichtig, dass dazu Rechnungen gemacht werden sollen? Dann müssten Bestimmung der Gitterkonstante auch gemacht werden. Durch Gesamtenergien? Durch Cell-Optimierung?

[7]

Q.2 Solids

```
Tasks:\\
Silicon:\\
paw-Rechnung mit anschliessender Cell f\"ur unterschiedliche Anzahl von k-Punkten\\
Heraussuchen von einigen erzeugten Daten aus prot-file\\
Mit den ‘‘si.banddata’’ f\"ur den Cell-Lauf mit gr\"o{\ss}ter k-Anzahl soll
mit paw_bands die Total Dos berechnet und dargestellt werden, dies mit
Synmmetrieangabe.\\
Berechnung der Bandstruktur mit gleichem  ‘‘si.banddata’’ aus
paw_bands.\\
Gap untersuchen und Vergleich Maxima und Minima von DOS mit
Bandstruktur.\\
Die Rechnungen liegen unter \home\ptjn\Solids\Si\Sicalc \\

Aluminum:\\
paw-Rechnung mit Mermin, Heraussuchen von einigen erzeugten Daten aus
prot-file\\
Bandstrukturberechnung, wie Si, aber mit Einzeichnen der
Fermi-Energie.\\
Zustandsdichteberechnung mit paw_bands, Kopie von erzeugter
"al.pdosout"-Datei und damit Berechnung partieller Zustandsdichten mit paw_dos.\\
Die Rechnungen liegen unter \home\ptjn\Solids\Al\Alcalc \\

Copper:\\
wie Aluminum. Vergleich der Werte der Zustandsdichten und B\"ander im
Valenzbereich.\\
Die Rechnungen liegen unter \home\ptjn\Solids\Cu\Cucalc

MgO:\\
paw-Rechnung, Gap-Vergleich mit Experiment.\\
Bandstruktur mit paw_bands berechnen und darstellen.\\
Partielle Zustandsdichten berechnen analog zu Aluminum.\\
Zuordnung der Energiebereiche zu atomaren Zust\"anden.
Die Rechnungen liegen unter \home\ptjn\Solids\MgO\MgOcalc
```

[7] FiXme Note: If possible this would be the point to discuss the problems with changing the size and shape of the unit cell. (I.e. sawtooth problem,) Potentially do cell dynamics. (Use isotropic and do not rely yet on anisotropic dynamics.) Everybody should do one calculation for an insulator first and then one calculation for a metal. Each of this calculations is done from scratch. In order to cover additional materials one could prepare the restart files of converged calculations and distribute them among the participants to do the band structure and density of states plots. In the end one could discuss the obtained results in a group.

Q.2.1 Geometry

First, you have to specify the crystal structure. This is done by choosing the unit cell, the respective lattice vectors and the coordinates of the atoms in the unit cell. It's just as for molecules, however, the lattice constant is that of the real crystal and, of course, the unit cell isn't isolated, we have 3-dimensional periodic structures.
In the following table we give the lattice vectors and the atomic positions for the most common crystal symmetry structures of the elements, a denoting the lattice constant.

space group	lattice vectors			atomic positions		
fcc	0	$a/2$	$a/2$	0	0	0
	$a/2$	0	$a/2$			
	$a/2$	$a/2$	0			
bcc	$-a/2$	$a/2$	$a/2$	0	0	0
	$a/2$	$-a/2$	$a/2$			
	$a/2$	$a/2$	$-a/2$			
hcp	0	$-a$	0	0	0	0
	$a\sqrt{3}/2$	$a/2$	0	0	$-a/\sqrt{3}$	$c/2$
	0	0	c			
diamond	0	$a/2$	$a/2$	0	0	0
	$a/2$	0	$a/2$	$a/4$	$a/4$	$a/4$
	$a/2$	$a/2$	0			

This geometry-part of the `strc`-file for an fcc silicon crystal with a lattice constant of 7.18 Bohr radii (experimental value) would be something like:

```
!GENERIC LUNIT=7.18  !END
!LATTICE   T= 0.50000  0.50000  0.00000
             0.00000  0.50000  0.50000
             0.50000  0.00000  0.50000 !END
  !SPECIES NAME= 'SI' ID='SI_HBS' NPRO= 2 2 1  lrhox=0
  !END
  !ATOM    NAME= 'SI1'  R= 0.00  0.00  0.00 !END
```

As explained in chapter 2.5, the variable `LRHOX` describes the expansion of the one-center expansion of the density in terms of spherical harmonics. Since an fcc- (or a bcc-)crystal has maximal cubic symmetry and the one-center expansion of the density (and the potential) naturally shows also this symmetry, the next non-zero term after `l=0` ist `l=4`, which is much to high for giving any noticeable contribution for the PAW method, in contrast to some other commonly used electronic structure methods. Thus, for such symmetric systems one can reduce this expansion to only one term.
We are now able to calculate the theoretical lattice constant by varying `LUNIT` and searching the minimum of the total energy. But CAUTION: with the lattice constant the number of planes waves in the expansion of the wavefunctions may also vary which may result in the so-called SAWTOOTH-problem (see chapter ???): The dependence of the total energy on the lattice constant is not a smooth curve.
The PAW-Code provides another possibility to get the lattice constant for minimal total energy:
After the the self-consistent iterations bringing the wavefunctions into the ground state, one can add a block in the respective `cntl`-file:

```
!CELL MOVE=T STOP=T FRIC=0.05 CONSTRAINTTYPE="ISOTROPIC" !END
```

(see the explanation in the manual).
We could also choose other cell-geometries (bcc, hcp, diamond, ...) and calculate the respective

cohesive energies to get the theoretically favoured crystal structure.

For determining the atomic structure of a crystal with more than one atom in the basis, you have to proceed like in the water-example: First do the iteration only for the wavefunctions, then add the RDYN-branch into the cntl-file.

Q.2.2 K-points

If the lattice constants for **molecules** are chosen large enough, we do not have a dependence of the eigenvalues on the k-points, the wavefunctions for the molecules in neighbouring periodic cells don't overlap. Thus, the Bloch-condition is automatically fullfilled and independent of k. k does not appear in the strc-file for molecules, the default value (1 k-point with $\vec{k} = (0, 0, 0)$, denoted by Γ) is taken for the iterations. For **solids**, the wavefunctions for states of adjacent unit cells overlap more or less strongly, Bloch's theorem together with the requirement of continuity and continuous differentiability of the wavefunction is determining the dispersion relations $E(\vec{k})$. The k-point density, which should be about equal in all directions, can be set in the strc-file either by DIV= or by R=.

```
!KPOINTS  DIV=7 7 7  !END
```

means, that the k-points are equally spaced within the Brilloin-zone as fractions of 7 of the reciprocal lattice vectors.

```
!KPOINTS  R=40.  !END
```

defines the density of k-points for the automatic generation of k-points, $\Delta k = \frac{2\pi}{R}$, in this case $\Delta k = \frac{2\pi}{40}$.

As already stated, for calculating f.e. the charge density we have to do a summation over the k-points chosen, instead of an integral over the Brillouin-zone. Clearly, the results will getting more accurate with a higher number of k-points (k-convergence). On the other hand, the computer time and the storage necessary are growing as well. A compromise between accuracy of the results and cost of computer resources has to be found. We will do such a study for silicon, which is the first task in this tutorium.

Q.2.3 Variable occupations

For a metal, there is one complication, the energy bands are partially occupied, which means that the number of occupied states differs from k-point to k-point. Worse, this number can change from iteration step to iteration step. Thus, we need to determine the occupations from the one-particle energies in every iteration step.

Variable occupations are enabled in the cntl-file using the branch

```
 !MERMIN START=T T[K]=0. adiabatic=T tetra+=T   retard=10.  !END
```

The name Mermin is only indirectly related to what we are actually doing. Mermin[145] extended the density functional theory to a free-energy functional theory. Minimizing the free energy functional for a given temperature with respect to the occupation yields the Fermi distribution.

With the option TETRA+=T we switch on an interpolation between the grids of the k-points using the improved tetrahedron method[133]. We need to specify T[K]=0., for this interpolation only works for zero temperature. The tetrahedron method requires the one-particle energies as input. We need to specify the option ADIABATIC=T, which predicts the energy eigenvalues from the previous

few iterations. RETARD=10 specifies the number of iterations that are considered in the prediction. A value that is too small may cause a noisy behavior, in particlular if very localized d- or f-levels are involved or we have to do with spin holes at the Fermi level, which may change from unoccupied to occupied states in successive iteration steps.

Q.2.4 Energy eigenstates

For metals, it is a must to specify in the cntl-file the variable SAFEORTHO=F in the branch PSIDYN.

The normal Car-Parrinello method does not directly produce eigenstates of the Hamiltonian, so that the eigenvalues are not available. This is because the Car-Parrinello method searches for a minimum of the total energy, and a scrambling of occupied states does not affect the total energy. The CP-PAW code uses a modified equation of motion that ensures that the wave function only come to rest, when they are eigenstates of the Hamiltonian. In that case the Lagrange multipliers are the eigenvalues, which can then be used to determine the occupations.

This option is not as clearcut as the simulation for insulators. There is the fact that there is no strict energy conservation. The Lagrange multipliers need to be extrapolated from the previous time step, etc. This, and because the method rests on a modification of the orthogonalization of wave functions, is why the parameter is called SAFEORTHO=F.

Q.2.5 Density of states

If there are in an energy interval $d\epsilon$ several bands, numbered by an index n, then the density of states is given by

$$D(\epsilon) = \frac{V}{(2\pi)^3} \sum_n \int_{S_{\vec{k}(\epsilon)}} \frac{d^2k}{|\vec{\nabla}_k \epsilon_n(\vec{k})|} \tag{Q.1}$$

For simple metals we have for the lower enery part of the first valence band nearly free electron behaviour, that is

$$\epsilon(\vec{k}) = \frac{1}{2m^*} k^2$$

with

$$\begin{aligned} \vec{\nabla}_k \epsilon(\vec{k}) &= \frac{1}{m^*} \vec{k} \\ \rightarrow |\vec{\nabla}_k \epsilon(k)| &= \frac{1}{m^*} k \end{aligned} \tag{Q.2}$$

The constant-energy areas are spheres in $\vec{k}$-space. Thus, we get in (Q.1):

$$d^2k = k^2 \sin\vartheta \, d\vartheta \, d\varphi$$

and therefore

$$\begin{aligned} \longrightarrow \int_{S_{\vec{k}(\epsilon)}} \frac{d^2k}{|\vec{\nabla}_k \epsilon(\vec{k})|} &= k m^* \int_0^{2\pi} \int_0^{\pi} \sin\vartheta \, d\vartheta \, d\varphi \\ &= 4\pi k m^* \\ &= 4\pi m^* \sqrt{2m^* \epsilon}, \end{aligned}$$

i.e. a square-root behaviour for the bulk density of states.

We will use the paw_dos-tool with the block
!DCNTl! WEIGHT
in the dcntl-control file. This option allows us to calculate the total density of states, the denisty of states for the atoms of the system (often called "local density of states"), or the angular momentum weights ("partial density of states" for the respective atom).

Q.3 Silicon

Q.3.1 Determination of the electronic structure and the lattice constant

For semiconductors like silicon the bandstructure is relatively easy to determine. All valence band states are fully occupied, thus, we have for every k-point the same number of occupied states. The sampling over the Brillouin-zone for evaluating expectation values or getting the new charge density in the course of the self-consistent iterations can be restricted to a small number of k-points. We start using the Monkhorst-Pack k-Point division with a subdivision of 3 for the reciprocal basic lattice vectors (cf. `DIV= 3 3 3` in the `si.strc` file). With respect to time-inversion symmetry, this results into 14 independent k-points in the Brillouin-zone. Within PAW, we do not include further symmetry considerations, which would lead to taking only k-points of an irreducible part of the Brilloin-zone. The emphasis of PAW ist to describe non-symmetric physical systems. On the other hand, the reduction to the irreducible part does not result in a remarkable time gain in the course of selfconsistency.

Silicon crystallizes in the diamond structure, which can be described as an fcc-lattice with two Si-atoms in the unit cell. The experimental lattice constant is 5.43 Å or 10.2612 Bohr radii. Silicon is not spinpolarized, therefore we take NSPIN=1 SPIN[HBAR]=0. We choose EMPTY=5 (5 empty bands are calculated), thus, we can calculate the band gap between valence and conduction bands, the experimental value is $1.17e\,V$.
Silicon is used to become familiar with the PAW-treatment of crystals, especially of semiconductor type. In order to save computertime, we will not do a fully converged calculation with respect to the energy cutoff `EPWPSI` of the wavefunction.

The `cntl`-file and the `strc`-file are given in the directory

```
/palau/ptjn/Solids/Si
```

Calculations have been done in the directory

```
/palau/ptjn/Solids/Si/Sicalc
```

```
!CONTROL
 !GENERIC NSTEP=400 DT=10.0 NWRITE=100 START=T !END
 !FOURIER EPWPSI=30. CDUAL=2.0  !END
 !DFT     TYPE=10     !END
 !PSIDYN STOP=T  FRIC=0.01
   !AUTO FRIC(-)=0.3 FACT(-)=0.97  FRIC(+)=0.3 FACT(+)=1. minfric=0.01 !END
 !END
!END
!EOB

!STRUCTURE
  !GENERIC LUNIT=10.2612  !END
  !KPOINTS  DIV=3 3 3  !END
  !OCCUPATIONS EMPTY=5 NSPIN=1 SPIN[HBAR]=0.  !END
  !LATTICE T= 0.00000  0.50000  0.50000
             0.50000  0.00000  0.50000
             0.50000  0.50000  0.00000 !END
  !SPECIES NAME= 'SI' ID='SI_HBS' NPRO= 2 2 1  lrhox=0
  !END
  !ATOM    NAME= 'SI1'  R= 0.00  0.00  0.00 !END
  !ATOM    NAME= 'SI2'  R= 0.25  0.25  0.25 !END
  !END
!EOB
```

Tasks:

(1) Do this calculation. Find out of the protocol-file the number of k-points, the total energy, and the absolute band gap. Look on every information regarding the k-points.

(2) Create a new directory (e.g. CELL), copy si.strc, si.cntl, si.rstrt (jobscript?) into this new directory. Go into this directory and do the following changes in si.cntl:
(a) insert the branch `!CELL MOVE=T STOP=T FRIC=0.05 CONSTRAINTTYPE="ISOTROPIC" !END`
(b) set `NSTEP` to 600, `START` to `F`
(c) shut off the auto-pilot for the wavefunctions.
Let PAW run and observe the changes of the lattice vectors T1,T2,T3. What do the new values mean for the lattice constant?

(3) Do all the same steps (tasks(1) and (2)) in new directories for `DIV= n n n`, n=5,7,9,...
Stop, if the total energy differs from the (k-)converged value $-8.0350H$ by less than 10 meV.

(4) Fill in the following table for all calculations done, ie. n of DIV, number of k-points, total energy, absolute gap, and the lattice constant.

(5) How does the calculated band gap (for your highest n used) compare to experiment (value and respective k-points)?

n	3						17
No.k-points							2457
E_{total}/H							-8.0350
Gap/eV							0.7445
a_0/Bohr radii							10.3059

The table with the respective data should look like

n	3	5	7	9			17
No.k-points	14	63	172	365			2457
E_{total}/H	-8.0136	-8.0329	-8.0347	-8.0350			-8.0350
Gap/eV	0.8650	0.7452	0.7451	0.7588			0.7445
a_0/Bohr radii	10.3704	10.3123	10.36065	10.3059			10.3059

Q.3.2 Visualization of the bandstructure

In the following we want to visualize the density of states and the bandstructure for the CELL-calculation with the largest number of k-points necessary in the last study. Since we have used also for that case only few k-points for the selfconsistency iterations, we have to generate energies for a denser grid of k-points with a special program. We will do both visualizations in respective directories, therefore we have to create directories named f.e. BANDS and DOS . There is a tool named paw_bands.x which generates the bandstructure ore the DOS data. The command is
paw_bands.x case.bcntl
where case stands for the root name of the calculation (i.e. "si" in this case).

[8] We want to have the bandstructure $E(\vec{k})$ on the respective symmetry-lines of the Brillouin-zone for an fcc lattice[9]. For all lines we start with KVEC1 and end with KVEC2, all given in absolute coordinates. In this bcntl-file the first point is Γ, via the symmetrie-line Δ we end at X(=1.0 0.0 0.0). The next lines and points are $Z, W, Q, L, \Lambda, \Gamma, \Sigma, K, S, X$. NK gives the number of grid-points on the respective line.

For the calculation we have to add the lattice constant (here called kvecscale), which has been calculated from the "CELL"-output. We have to change the following si.bcntl-file accordingly.

```
!BCNTL
  !INPUTFILE NAME="si.banddata" !END
  !METHOD
    MODE=1
    METHOD_DIAG=1
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0. 0. 0. KVEC2=1.0 0.0 0.0   kvecscale= 10.111      NK=10
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1. 0. 0. KVEC2=1.0 0.5 0.0   kvecscale= 10.111     NK=5
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1.0 0.5 0.0  KVEC2=0.5 0.5 0.5  kvecscale= 10.111      NK=5
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0.5 0.5 0.5  KVEC2=0.0 0.0 0.0  kvecscale= 10.111      NK=10
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0.0 0.0 0.0  KVEC2=0.75 0.75 0.0  kvecscale= 10.111      NK=10
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1.0 0.25 0.25  KVEC2=1.0 0.0 0.0  kvecscale= 10.111      NK=4
      TATTACH=T
  !END
!END
!EOB
```

As a result, in the file denoted by the variable FILE for all calculated k-points the distance to the

[8] FiXme Note: Dauer auf palau frontend ca 2 1/2 min
[9] FiXme Note: hier Hinweis auf die Darstellung dieser Brillouinzone mit den eingezeichneten Symmetriepunkten

first k-point and the energies for that k-point are listed. With TATTACH=T one can achieve that the results for all symmetry lines are witten into the same file, in this example BANDS_all.dat. The default value for the number of energies for each k-point is 10, it can be changed f.e. to 6 by adding NB=6 in the LINE-branch.

After running paw_bands.x with that bcntl-file, it's worth to look into BANDS_all.dat to see how it is composed. The visualization with xmgrace (xmgrace -nxy BANDS_all.dat &), displays the bandstructure, however, we have to beautify this figure. We want to have vertical lines at each high-symmetry point. This can be done with the import of the vertices of that line. Create a file line1.dat with

```
1.0 -5.0
1.0 20.0
```

which gives a line at the point with x.value 1.0 from $y = -5.0$ to $y = 20.0$. Then click Data, then Import,ASCII, and choose "line1.dat". By creating a new file with the respective x-value of the next high-symmetry point (it's to be seen in the file BANDS_all.dat, here it's 1.5) and so on, one has now the vertical lines. To have them as black lines, click "Plot", "Set appearence", select the last 6 sets and choose for them under "Line properties" "Color" "black". If one wants also the bands as black lines, select all sets.
We are still not content with the standard tick labels and tick marks on the X axis. Therefore we enter "Plot", "Axis properties" and change at first for this axis "STOP" to 4.48735, this is the x-value for the last k-point. Then we go two lines down and click at "Special". There, from "Special Ticks:" we choose "Tick marks and labels" and enter for each main symmetry-point the x-value and the respective label. The greek letters have to be given with an extra \x (e.g. Γ is represented by \xG). Additionally, one can also supply the data for the lines (the x- value just in the middle of the respective two main symmetry-points.) Choose the right number of "Number of user ticks to use" and we are done.

Q.3.3 Visualization of the density of states

For the calculation of the density of states we have to copy si.banddata and si.pdos into this directory. We use again the paw_bands-tool.

The respective control file si.bcntl looks like this:

```
!BCNTL
  !INPUTFILE NAME="si.banddata" !END
  !METHOD
    MODE=2
    METHOD_DIAG=1
  !END
  !PDOS
    PDOSINFILE="si.pdos"
    NKDIV=16 16 16
    NB=12
    TUSESYM=T
    SPACEGROUP=225
  !END
  !DOS
    FILE="si.dos"
    EMIN=-10.0
    EMAX=20.0
    NE=1000
  !END
```

```
!END
!EOB
```

Here, NKDIV has the same meaning as DIV in the paw strc file. In the program paw_bands the energies are calculated via diagonalization of the Hamiltonian matrix. In order to minimize computer time we do this calculation only in the irreducible part of the Brillouin-zone. This is managed with the settings of the variables TUSESYM=T and the international number for the fcc spacegroup SPACEGROUP=225. Be careful, that is only possible, if one is not interested in the partial densities of states. For generating the partial densities of states one may proceed as described in the Aluminum section.

The density of states is calculated in the energy window [EMIN,EMAX] on a grid of NE points. Be aware that the valence and conduction bands you are interested in are lying within this energy window.
The output is written to si.bprot, the density of states to si.dos and may be visualized in this case with
xmgrace si.dos.

Further tasks:

(6) After that more technical section have a look what kind of band gap appears for silicon, i.e., search for the k-points with highest valence energy and lowest conduction energy, respectively.

(7) Study, from which k-point regions the maxima and minima in the density of states stem from.

Q.4 Aluminum

Aluminum is our first example for a metal. It's a simple metal and the bandstructure is nearly-free electron like, i.e. free-electron like ($E \propto k^2$) within the Brilloin-Zone, but with a splitting of the degenerate free-electron bands at the Brillouin-zone boundary. Since we treat a metal, we have to include the MERMIN branch in the cntrl-file.

```
!CONTROL
  !GENERIC NSTEP=300 START=T !END
  !FOURIER EPWPSI=30. CDUAL=2.0 !END
  !DFT     TYPE=10 !END
  !PSIDYN  STOP=T  FRIC=0.01 SAFEORTHO=F
    !AUTO FRIC(-)=0.3 FACT(-)=0.97  FRIC(+)=0.3 FACT(+)=1. MINFRIC=0.01 !END
  !END
  !MERMIN START=T T[K]=0. ADIABATIC=T RETARD=10. TETRA+=T !END
  !CELL_X MOVE=T STOP=T FRIC=.01 M=5000. CONSTRAINTTYPE="ISOTROPIC" !END
!END
!EOB
```

The structure of aluminum is fcc with a lattice constant of 7.6534 Bohr radii. Thus a strc-file could be

```
!STRUCTURE
  !GENERIC LUNIT= 7.6534 !END
  !OCCUPATIONS EMPTY=9 NSPIN=1  SPIN[HBAR]=0. !END
  !KPOINTS DIV=11 11 11 !END
  !LATTICE T= 0.0 0.5  0.5 0.5 0.0 0.5  0.5 0.5 0.0 !END
  !SPECIES NAME= 'Al'  ID='AL_HBS' NPRO=2 2 1 LRHOX=0 !END
  !ATOM    NAME= 'Al1'  R=  0.00    0.00      0.00   !END
!END
!EOB
```

Task:
Do this calculation. Find out the number of k-points, the total energy, the Fermi-energy (i.e. the chemical potential), and the difference between the onset of the valence band (=Γ_1) and the Fermi-energy E_F. [10]

Q.4.1 Bandstructure

Create the bandstructure plot for Al, analogously to Si (the maximal energy should be 35 eV). Manage, that the Fermi-Level E_F is also to be seen in the figure.

Q.4.2 Density of states

Determine the total density of states (`TYPE='TOTAL'`). Additionally, calculate as partial density of states the angular momentum weight via `TYPE='S'`,`TYPE='P'`, and `TYPE='D'`. For that, we have to change the procedure compared to silicon. Instead of calculating all necessary data in one step via the `paw_bands`-tool, we now have two steps. For the first step, `TUSESYM=F` must be set in the `al.bcntl`-file. Running `paw_bands.x al.bcntl` creates a new pdos-file, defined by the variable `PDOSOUTFILE`.

```
!BCNTL
  !INPUTFILE NAME="al.banddata" !END
  !METHOD
    MODE=2
    METHOD_DIAG=1
    EPWPSI_OFF=10.
    TFIXSYMMETRYBREAK_OFF=T
  !END
  !PDOS
    PDOSINFILE="al.pdos"
    PDOSOUTFILE="al.pdosout"
    NKDIV=11 11 11
    NB=10
    TUSESYM=F
    SPACEGROUP=225
    ISHIFT_OFF=0 0 0
    TSHIFT_OFF=F
  !END
  !DOS
    FILE="casenosymm.dos"
    EMIN=-5.0
    EMAX=20.0
    NE=1000
  !END
!END
!EOB
```

This `PDOSOUTFILE` replaces `al.pdos` for the calculation of the partial density of states. Therefore, we create a new directory and copy `al.pdosout` as `al.pdos` into this new directory. Then, we have to construct an input file `al.dcntl` for the `paw_dos`-tool:

```
!DCNTL
  !GRID EMIN[EV]=-5. EMAX[ev]=22. DE[ev]=0.02   !END
```

[10] FiXme Note: 666 k-points, TOTAL ENERGY -2.1280 H, E_F=9.68875 eV, valence bandwidth 11.17 eV

```
  !WEIGHT ID='TOTAL' TYPE='TOTAL' !END
  !WEIGHT ID='AL-S'
      !ATOM NAME='Al1' TYPE='S' !END
  !END
  !WEIGHT ID='AL-P'
      !ATOM NAME='Al1' TYPE='P' !END
  !END
  !WEIGHT ID='AL-D'
      !ATOM NAME='Al1' TYPE='D' !END
  !END
  !OUTPUT ID='TOTAL'  FILE='total.dos' !END
  !OUTPUT ID='AL-S'   FILE='al-s.dos' !END
  !OUTPUT ID='AL-P'   FILE='al-p.dos' !END
  !OUTPUT ID='AL-D'   FILE='al-d.dos' !END
!END
!EOB
```

We proceed now as follows

```
paw_dos.x al.dcntl
paw_stripnos *.dos
xmgrace -nxy total.ddos
Running xmgrace, we have to import al-s.ddos,al-p.ddos,al-d.ddos.
```

Q.5 Copper

Copper is a so-called d-band metal, it's also fcc. Take the experimental lattice constant of 6.83 Bohr radii and proceed as for aluminum.
Compare especially the valence band density of states for both metals. Look at the hybridization of the copper d-bands with the s-p band. Density of states and the bandstructure always give a hint to the respective type of the physical system studied.

Q.6 MgO

MgO represents a II-IV compound. Since all valence states are occupied, we can take the repective `cntl` file fom the silicon calculation and reduce the k-point number, this time, we will take `R=17`.. The experimental lattice constant is 7.91 Bohr radii.

Compare the resulting band gap with the experimental value ($7.8eV$).

Q.6.1 Bandstructure

Take the following `bcntl` file for the plot of the bandstructure

```
!BCNTL
  !INPUTFILE NAME="mgo.banddata" !END
  !METHOD
    MODE=1
    METHOD_DIAG=1
    EPWPSI_OFF=10.
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0. 0. 0. KVEC2=1.0 0.0 0.0   kvecscale= 7.91      NK=11
```

```
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1. 0. 0. KVEC2=1.0 0.5 0.0   kvecscale= 7.91      NK=6
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1.0 0.5 0.0  KVEC2=0.5 0.5 0.5  kvecscale= 7.91      NK=6
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0.5 0.5 0.5  KVEC2=0.0 0.0 0.0  kvecscale= 7.91      NK=11
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=0.0 0.0 0.0  KVEC2=0.75 0.75 0.0  kvecscale= 7.91      NK=11
      TATTACH=T
  !END
  !LINE FILE='BANDS_all.dat'
      KVEC1=1.0 0.25 0.25  KVEC2=1.0 0.0 0.0  kvecscale= 7.91      NK=4
      TATTACH=T
  !END
!END
!EOB
```

Q.6.2 Density of states

This time, a lot more work has to be done. We want to have all partial densities for both atoms. Change for the `mgo.bcntl` file that respective one for aluminum accordingly. [11]
The file `mgo.dcntl` should look like that:

```
!DCNTL
  !GRID  EMIN[EV]=-10.0 EMAX[EV]=30. DE[EV]=0.02 !END
  !WEIGHT ID='TOTAL'   TYPE='TOTAL' !END
  !WEIGHT ID='Mg-S'    !ATOM NAME='MG1' TYPE='S' !END
  !END
  !WEIGHT ID='Mg-P'    !ATOM NAME='MG1' TYPE='P' !END
  !END
  !WEIGHT ID='Mg-D'    !ATOM NAME='MG1' TYPE='D' !END
  !END
  !WEIGHT ID='Mg'    !ATOM NAME='MG1' TYPE='ALL' !END
  !END
  !OUTPUT ID='TOTAL' FILE='total.dos' !END
  !OUTPUT ID='Mg-S'  FILE='Mg-s.dos' !END
  !OUTPUT ID='Mg-P'  FILE='Mg-p.dos' !END
  !OUTPUT ID='Mg-D'  FILE='Mg-d.dos' !END
  !OUTPUT ID='Mg'  FILE='Mg.dos' !END
  !WEIGHT ID='O-S'    !ATOM NAME='O_1' TYPE='S' !END
  !END
  !WEIGHT ID='O-P'    !ATOM NAME='O_1' TYPE='P' !END
  !END
  !WEIGHT ID='O-D'    !ATOM NAME='O_1' TYPE='D' !END
  !END
```

[11] FiXme Note: nkdiv=10 10 10 könnte etwas zulang dauern

```
  !WEIGHT ID='O'   !ATOM NAME='O_1' TYPE='ALL' !END
  !END
  !OUTPUT ID='O-S'  FILE='O-s.dos' !END
  !OUTPUT ID='O-P'  FILE='O-p.dos' !END
  !OUTPUT ID='O-D'  FILE='O-d.dos' !END
  !OUTPUT ID='O'  FILE='O.dos' !END
!END
!EOB
```

Plot in one figure the total density of states together with Mg.dos and O.dos. From what atom stem the bands in the different energy regimes?
Plot in the next two figures all densities for the respective atom (local density of states together with the angular momenta partial density of states). Of which l-type are the energy regimes?

Bibliography

[1] W. Kohn. Nobel Lectures, Chemistry, 1996-2000, chapter Electronic Structure of Matter - Wave functions and Density Functionals, page 213. World Scientific, Singapore, 2003.

[2] Pierre Hohenberg and Walter Kohn. Inhomogeneous electron gas. Phys. Rev., 136:B864, 1964. doi: 10.1103/PhysRev.136.B864. URL http://link.aps.org/doi/10.1103/PhysRev.136.B864.

[3] Walter Kohn and Lu J. Sham. Self-consistent equations including exchange and correlation effects. Phys. Rev., 140:A1133, 1965. doi: 10.1103/PhysRev.140.A1133. URL http://link.aps.org/doi/10.1103/PhysRev.140.A1133.

[4] Robert O. Jones and Olle Gunnarsson. The density functional formalism, its applications and prospects. Rev. Mod. Phys., 61:689, 1989.

[5] Evert Jan Baerends and Oleg V. Gritsenko. A quantum chemical view of density functional theory. J. Phys. Chem. A, 101:5383, 1997.

[6] Ulf von Barth. Basic density-functional theory—an overview. Physica Scripta, 2004(T109):9, 2004. URL http://stacks.iop.org/1402-4896/2004/i=T109/a=001.

[7] J.P. Perdew, A. Ruzsinszky, J. Tao, V.N. Staroverov, G.E. Scuseria, and G.I. Csonka. Prescription for the design and selection of density functional approximations: More constraint satisfaction with fewer fits. J. Chem. Phys., 123:62201, 2005.

[8] Aron J. Cohen, Paula Mori-Sánchez, and Weitao Yang. Challenges for density functional theory. Chem. Rev., 112(1):289, 2012. doi: 10.1021/cr200107z. URL http://dx.doi.org/10.1021/cr200107z.

[9] Christopher J. Cramer and Donald G. Truhlar. Density functional theory for transition metals and transition metal chemistry. Phys. Chem. Chem. Phys., 11:10757–10816, 2009. doi: 10.1039/B907148B. URL http://dx.doi.org/10.1039/B907148B.

[10] Peter E. Blöchl, Clemens J. Först, and Johannes Kästner. Handbook for Materials Modeling, chapter Electronic Structure Methods: Augmented Waves, Pseudopotentials and the Projector augmented wave method, page 93. Springer, 2005.

[11] P. E. Blöchl, C. J. Först, and J. Schimpl. Projector augmented wave method: Ab-initio molecular dynamics with full wave functions. Bull. Mater. Sci., 26:33, 2003.

[12] Per-Olov Löwdin. Quantum theory of many-particle systems. i. physical interpretations by means of density matrices, natural spin-orbitals, and convergence problems in the method of configurational interaction. Phys. Rev., 97:1474, 1955. doi: 10.1103/PhysRev.97.1474. URL http://link.aps.org/doi/10.1103/PhysRev.97.1474.

[13] A.J. Coleman. Structure of fermion density matrices. Rev. Mod. Phys., 35:668, 1963. doi: 10.1103/RevModPhys.35.668. URL http://link.aps.org/doi/10.1103/RevModPhys.35.668.

[14] Mark S. Hybertsen and Steven G. Louie. Electron correlation in semiconductors and insulators: Band gaps and quasiparticle energies. Phys. Rev. B, 34:5390, 1986.

[15] Kieron Burke, John P. Perdew, and Matthias Ernzerhof. Why semilocal functionals work: Accuracy of the on-top pair density and importance of system averaging. J. Chem. Phys., 109: 3760, 1998.

[16] John P. Perdew and Kieron Burke. Comparison shopping for a gradient-corrected density functional. Int. J. Quant. Chem., 57:309, 1996.

[17] Mel Levy. Universal variational functionals of electron densities, first order density matrixes and natural spin-orbitals and solution of the v-representability problem. Proc. Nat'l Acad. Sci. USA, 76:6062, 1979. URL `http://www.pnas.org/content/76/12/6062.abstract`.

[18] Elliott H. Lieb. Density functionals for coulomb systems. Int. J. Quantum Chem., 24(3): 243–277, 1983. ISSN 1097-461X. doi: $10.1002/\mathrm{qua}.560240302$. URL `http://dx.doi.org/10.1002/qua.560240302`.

[19] J. Harris and R.O. Jones. The surface energy of a bounded electron gas. J. Phys. F: Met. Phys., 4:1170, 1974.

[20] Olle Gunnarsson and Bengt I. Lundquist. Exchange and correlation in atoms, molecules, and solids by the spin-density-functional formalism. Phys. Rev. B, 13:4274, 1976.

[21] R.O. Jones. Density functional theory: Past, present, ... future? PsiK Newsletter, 124:0, 2015.

[22] J.I. Musher. Comment on some theorems of quantum chemistry. Am. J. Phys., 34:267, 1966.

[23] John P. Perdew and Karla Schmidt. Jacob's ladder of density functional approximations for the exchange-correlation energy. AIP Conf. Proc., 577:1, 2001.

[24] P.A.M. Dirac. Note on the exchange phenomena in the thomas atom. Proc. Cambridge Philos. Soc., 26:376, 2930. local Hartree Fock exchange.

[25] K. Schwarz. Optimization of the statistical exchange parameter α for the free atoms h through nb. Phys. Rev. B, 5:2466, 1971.

[26] I. Lindgren and K. Schwarz. Analysis of the electronic exchange in atoms. Phys. Rev. A, 5: 542, 1972.

[27] D.M. Ceperley and B.J. Alder. Ground state of the electron gas by a stochastic method. Phys. Rev. Lett., 45:566, 1980.

[28] Sh.-K. Ma and K.A. Brueckner. Correlation energy of an electron gas with a slowly varying high density. Phys. Rev., 165:18, 1968.

[29] U. von Barth and L. Hedin. A local exchange-correlation potential for the spin polarized case. i. J. Phys. C: Solid State Phys., 5:1629, 1972.

[30] John P. Perdew. Accurate density functional for the energy: Real-space cutoff of the gradient expansion for the exchange hole. Phys. Rev. Lett, 55:1665, 1985.

[31] D.C. Langreth and M.J. Mehl. Beyond the local-density approximation in calculations of ground-state electronic properties. Phys. Rev. B, 28:1809, 1983.

[32] A. D. Becke. Density-functional exchange energy with correct asymptotic behavior. Phys. Rev. A, 38:3098, 1988.

[33] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. Phys. Rev. Lett, 77:3865, 1996.

[34] E.I. Proynov, E. Ruiz, A. Vela, and D. R. Salahub. Determining and extending the domain of exchange and correlation functionals. Int. J. Quant. Chem., 56, S29:61, 1995.

[35] T. Van Voorhis and G.E. Scuseria. A novel form for the exchange-correlation energy functional. J. Chem. Phys., 109:400, 1998.

[36] Axel D. Becke. A new inhomogeneity parameter in density-functional theory. J. Chem. Phys., 109:2092, 1998.

[37] Jianmin Tao, John P. Perdew, Viktor N. Staroverov, and Gustavo E. Scuseria. Climbing the density functional ladder: Nonempirical meta-generalized gradient approximation designed for molecules and solids. Phys. Rev. Lett, 91:146401, 2003.

[38] Axel D. Becke. A new mixing of hartree-fock and local density-functional theories. J. Chem. Phys., 98:1372, 1993. doi: $10.1063/1.464304$.

[39] Axel D. Becke. Density functional thermochemistry. iii. the role of exact exchange. J. Chem. Phys, 98:5648, 1993. doi: $10.1063/1.464913$.

[40] David C. Langreth and John P. Perdew. The exchange-correlation energy of a metallic surface. Sol. St. Commun., 17:1425, 1975.

[41] John P. Perdew, Matthias Ernzerhof, and Kieron Burke. Rationale for mixing exact exchange with density functional approximations. J. Chem. Phys., 105:9982, 1996.

[42] J. Heyd, G.E. Scuseria, and M. Ernzerhof. Hybrid functionals based on a screened coulomb potential. J. Chem. Phys., 118(18):8207, 2003. doi: $\mathrm{doi}{:}10.1063/1.1564060$. URL `http://scitation.aip.org/content/aip/journal/jcp/118/18/10.1063/1.1564060`.

[43] M. Marsman, J. Paier, A. Stroppa, and G. Kresse. Hybrid functionals applied to extended systems. J. Phys.: Condens. Matter, 20:64201, 2008.

[44] P.J. Stephens, F.J. Devlin, C.F. Chabalowski, and M.J. Frisch. Ab-initio calculation of vibrational adsorption and circular dichroism spectra using density-functional force fields. J. Phys. Chem, 98:11623, 1994.

[45] M. Ernzerhof and G.E. Scuseria. Assessment of the perdew-ernzerhof exchange-correlation functional. J. Chem. Phys., 110:5029, 99.

[46] C. Adamo and V. Barone. Toward reliable density functional methods without adjustable parameters: The pbe0 model. J. Chem. Phys., 110:6158, 1999.

[47] Vladimir I. Anisimov, Jan Zaanen, and Ole K. Andersen. Band theory and mott insulators: Hubbard *U* instead of stoner *I*. Phys. Rev. B, 44:943–954, Jul 1991. doi: $10.1103/\mathrm{PhysRevB}.44.943$. URL `http://link.aps.org/doi/10.1103/PhysRevB.44.943`.

[48] A. I. Liechtenstein, V. I. Anisimov, and J. Zaanen. Density-functional theory and strong interactions: Orbital ordering in mott-hubbard insulators. Phys. Rev. B, 52:R5467–R5470, Aug 1995. doi: $10.1103/\mathrm{PhysRevB}.52.\mathrm{R}5467$. URL `http://link.aps.org/doi/10.1103/PhysRevB.52.R5467`.

[49] Pavel Novak, Jan Kunes, Laurent Chaput, and Warren E. Pickett. Exact exchange for correlated electrons. Phys. Stat. Sol. B, 243:563, 2006.

[50] Fabien Tran, Peter Blaha, Karlheinz Schwarz, and Pavel Novák. Hybrid exchange-correlation energy functionals for strongly correlated electrons: Applications to transition-metal monoxides. Phys. Rev. B, 74:155108, 2006.

[51] M. Dion, H. Rydberg, E. Schröder, D. C. Langreth, and B. I. Lundqvist. Van der waals density functional for general geometries. Phys. Rev. Lett, 92:246401, 2004.

[52] T. Thonhauser, Valentino R. Cooper Shen Li, Aaron Puzder, P. Hyldgaard, and D.C. Langreth. Van der waals density functional: Self-consistent potential and the nature of the van der waals bond. Phys. Rev. B, 76:125112, 2007.

[53] Kyuho Lee, Éamonn D. Murray, Lingzhu Kong, Bengt I. Lundqvist, and David C. Langreth. Higher-accuracy van der waals density functional. Phys. Rev. B, 82:81101, 2010.

[54] John A. Pople, Martin Head-Gordon, Douglas J. Fox, Krishnan Raghavachari, and Larry A. Curtiss. Gaussian-1 theory: A general procedure for prediction of molecular energies. J. Chem. Phys., 90:5622, 1989.

[55] Larry A. Curtiss, Christopher Jones, Gary W. Trucks, Krishnan Raghavachari, and John A. Pople. Gaussian-1 theory of molecular energies for second-row compounds. J. Chem. Phys., 93:2537, 1990.

[56] Larry A. Curtiss, Krishnan Raghavachari, Paul C. Redfern, and John A. Pople. Assessment of gaussian-2 and density functional theories for the computation of enthalpies of formation. J. Chem. Phys., 106:1063, 1997.

[57] Larry A. Curtiss, Paul C. Redfern, Krishnan Raghavachari, and John A. Pople. Assessment of gaussian-2 and density functional theories for the computation of ionization potentials and electron affinities. J. Chem. Phys., 109:42, 1998.

[58] A. D. Becke. Density-functional thermochemistry. i. the effect of the exchange-only gradient correction. J. Chem. Phys., 96:2155, 1992.

[59] A. D. Becke. Density-functional thermochemistry. ii. the effect of the perdew-wang generalized-gradient correlation correction. J. Chem. Phys., 97:9173, 1992.

[60] Axel D. Becke. Density-functional thermochemistry. iv. a new dynamical correlation functional and implications for exact-exchange mixing. J. Chem. Phys., 104:1040, 1996.

[61] Axel D. Becke. Density-functional thermochemistry. v. systematic optimization of exchange-correlation functionals. J. Chem. Phys., 107:8554, 1997.

[62] J. Paier, R. Hirschl, M. Marsman, and G. Kresse. The perdew-burke-ernzerhof exchange-correlation functional applied to the g2-1 test set using a plane-wave basis set. J. Chem. Phys., 122:234102, 2005.

[63] J. Paier, M. Marsman, K. Hummer, G. Kresse, I.C. Gerber, and J.G. Ángyán. Screened hybrid density functionals applied to solids. J. Chem. Phys., 124:154709, 2006.

[64] J. Paier, M. Marsman, K. Hummer, G. Kresse, I. C. Gerber, and J. G. Ángyán. Erratum: "screened hybrid density functionals applied to solids" [j. chem. phys. 124, 154709 (2006). J. Chem. Phys., 125:249901, 2006.

[65] P. E. Blöchl. Projector augmented-wave method. Phys. Rev. B, 50:17953–17979, Dec 1994. doi: $10.1103/\mathrm{PhysRevB}.50.17953$. URL `http://link.aps.org/doi/10.1103/PhysRevB.50.17953`.

[66] J.C. Slater. Wave functions in a periodic potential. Phys. Rev., 51:846, 1937.

[67] J. Korringa. On the calculation of the energy of a bloch wave in a metal. Physica, 13:392, 1947.

[68] W. Kohn and N. Rostocker. Solution of the schrödinger equation in periodic lattices with an application to metallic lithium. Phys. Rev., 94:1111, 1954.

[69] O. Krogh Andersen. Linear methods in band theory. Phys. Rev. B, 12:3060–3083, Oct 1975. doi: 10.1103/PhysRevB.12.3060. URL http://link.aps.org/doi/10.1103/PhysRevB.12.3060.

[70] H. Krakauer, M. Posternak, and A. J. Freeman. Linearized augmented plane-wave method for the electronic band structure of thin films. Phys. Rev. B, 19:1706, 1979.

[71] David J. Singh. Planewaves, pseudopotentials and the LAPW method. Kluwer, 1994.

[72] J.M. Soler and A.R. Williams. Simple formula for the atomic forces in the augmented-plane-wave method. Phys. Rev. B, 40:1560, 1989.

[73] David Singh. Ground-state properties of lanthanum: Treatment of extended-core states. Phys. Rev. B, 43:6388, 1991.

[74] E. Sjöstedt, L. Nordström, and D. J. Singh. An alternative way of linearizing the augmented plane-wave method. Solid State Commun., 114:15, 2000.

[75] Georg K. H. Madsen, Peter Blaha, Karlheinz Schwarz, Elisabeth Sjoestedt, and Lars Nordstroem. Efficient linearization of the augmented plane-wave method. Phys. Rev. B, 64:195134, 2001.

[76] Hans L. Skriver. The LMTO method: muffin-tin orbitals and electronic structure. Springer, 1984.

[77] Ole K. Andersen and Ove Jepsen. Explicit, first-principles tight-binding theory. Phys. Rev. Lett, 53:2571, 1984.

[78] O. K. Andersen, T. Saha-Dasgupta, and S. Ezhof. Third-generation muffin–tin orbitals. Bull. Mater. Sci., 26:19, 2003.

[79] Karsten Held, Igor A. Nekrasov, Georg Keller, Volker Eyert, Nils Blümer, Andrew K. McMahan, Richard T. Scalettar, Thomas Pruschke, Vladimir I. Anisimov, and Dieter Vollhardt. The lda+dmft approach to materials with strong electronic correlations. NIC Series (John von Neumann Institute for Computing), 10:175, 2002.

[80] C. Herring. A new method for calculating wave functions in crystals. Phys. Rev., 57:1169, 1940.

[81] J.C. Phillips and L. Kleinman. New method for calculating wave functions in crystals and molecules. Phys. Rev., 116:287, 1959.

[82] E. Antoncik. Approximate formulation of the orthogonalized plane-wave method. J. Phys. Chem. Solids, 10:314, 1959.

[83] D. R. Hamann, M. Schlüter, and C. Chiang. Norm-conserving pseudopotentials. Phys. Rev. Lett, 43:1494, 1979.

[84] Alex Zunger and Marvin L. Cohen. First-principles nonlocal-pseudopotential approach in the density-functional formalism: Development and application to atoms. Phys. Rev. B, 18:5449, 1978. doi: 10.1103/PhysRevB.18.5449.

[85] G.P. Kerker. Non-singular atomic pseudopotentials for solid state applications. J. Phys. C: Solid State Phys., 13:L189, 1980.

[86] Giovanni B. Bachelet, Don R. Hamann, and Michael Schlüter. Pseudopotentials that work: From h to pu. Phys. Rev. B, 26:4199, 1982.

[87] N. Troullier and J.L. Martins. Efficient pseudopotentials for plane-wave calculations. Phys. Rev. B, 43:1993, 1991.

[88] J.S. Lin, A. Qteish, M.C. Payne, and V. Heine. Optimized and transferable nonloeal separable ab initio pseudopotentials. Phys. Rev. B, 47:4174, 1993.

[89] M. Fuchs and M. Scheffler. Ab initio pseudopotentials for electronic structure calculations of poly-atomic systems using density-functional theory. Comp. Phys. Comm., 119:67, 1998.

[90] Leonard Kleinman and D. M. Bylander. Efficacious form for model pseudopotentials. Phys. Rev. Lett, 48:1425, 1982.

[91] Peter E. Blöchl. Generalized separable potentials for electronic-structure calculations. Phys. Rev. B, 41:5414, 1990.

[92] D. Vanderbilt. Soft self-consistent pseudopotentials in a generalized eigenvalue formalisem. Phys. Rev. B, 41:7892, 1990.

[93] S. G. Louie, S. Froyen, and M. L. Cohen. Nonlinear ionic pseudopotentials in spin-density-functional calculations. Phys. Rev. B, 26:1738, 1982.

[94] D. R. Hamann. Generalized norm-conserving pseudopotentials. Phys. Rev. B, 40:2980, 1989.

[95] D. Vanderbilt. Soft self-consistent pseudopotentials in a generalized eigenvalue formalism. Phys. Rev. B, 41:17892, 1990.

[96] K. Laasonen, A. Pasquarello, R. Car, C. Lee, and D. Vanderbilt. Implementation of ultrasoft pseudopotentials in ab-initio molecular dynamics. Phys. Rev. B, 47:110142, 1993.

[97] Xavier Gonze, Roland Stumpf, and Matthias Scheffler. Analysis of separable potentials. Phys. Rev. B, 44:8503, 1991.

[98] C.G. Van de Walle and P.E. Blöchl. First-principles calculations of hyperfine parameters. Phys. Rev. B, 47:4244, 1993.

[99] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos. Iterative minimization techniques for ab-initio total-energy calculations: molecular dynamics and conjugate-gradients. Rev. Mod. Phys, 64:11045, 1992.

[100] Roberto Car and Michele Parrinello. Unified approach for molecular dynamics and density-functional theory. Phys. Rev. Lett, 55:2471, 1985. doi: $10.1103/\mathrm{PhysRevLett.55.2471}$. URL `http://link.aps.org/doi/10.1103/PhysRevLett.55.2471`.

[101] S. Nose. A unified formulation of the constant temperature molecular dynamics methods. J. Chem. Phys., 81:511, 1984.

[102] William G. Hoover. Canonical dynamics: Equilibrium phase space distributions. Phys. Rev. A, 31:1695, 1985.

[103] Peter E. Blöchl and Michele Parrinello. Adiabaticity in first-principles molecular dynamics. Phys. Rev. B, 45:9413, 1992.

[104] Peter E. Blöchl. Second-generation wave-function thermostat for ab-initio molecular dynamics. Phys. Rev. B, 65:104303, 2002.

[105] S. C. Watson and E. A. Carter. Spin-dependent pseudopotentials. Phys. Rev. B, 58:R13309, 1998.

[106] Georg Kresse and D. Joubert. From ultrasoft pseudopotentials to the projector augmented-wave method. Phys. Rev. B, 59:1758, 1999.

[107] N. A. W. Holzwarth, G. E. Mathews, R. B. Dunning, A. R. Tackett, and Y. Zheng. Comparison of the projector augmented-wave, pseudopotential, and linearized augmented-plane-wave formalisms for density-functional calculations of solids. Phys. Rev. B, 55:2005, 1997.

[108] N. A. W. Holzwarth, G. E. Matthews, A. R. Tackett, and R. B. Dunning. Orthogonal polynomial projectors for the projector augmented wave method of electronic structure calculations. Phys. Rev. B, 57:11827, 1998.

[109] N.A.W. Holzwarth, A.R. Tackett, and G.E. Matthews. A projector augmented wave (paw) code for electronic structure calculations, part i: atompaw for generating atom-centered functions. Comp. Phys. Comm., 135:329, 2001.

[110] A.R. Tackett, N.A.W. Holzwarth, and G.E. Matthews. A projector augmented wave (paw) code for electronic structure calculations, part ii: pwpaw for periodic solids in a plane wave basis. Comp. Phys. Comm., 135:348, 2001.

[111] X. Gonze, B. Amadon, P. M. Anglade, J. M. Beuken, F. Bottin, P. Boulanger, F. Bruneval, D. Caliste, R. Caracas, M. Cote, T. Deutsch, L. Genovese, Ph. Ghosez, M. Giantomassi, S. Goedecker, D. R. Hamann, P. Hermet, F. Jollet, G. Jomard, S. Leroux, M. Mancini, S. Mazevet, M. J. T. Oliveira, G. Onida, Y. Pouillon, T. Rangel, G. M. Rignanese, D. Sangalli, R. Shaltaf, M. Torrent, M. J. Verstraete, G. Zerah, and J. W. Zwanziger. Abinit: First-principles approach to material and nanosystem properties. Computer Physics Communications, 180(12):2582–2615, 2009. doi: 10.1016/j.cpc.2009.07.007. URL http://dx.doi.org/10.1016/j.cpc.2009.07.007. Code is available at the website http://www.abinit.org.

[112] Marc Torrent, Francois Jollet, Francois Bottin, Gilles Zerah, and Xavier Gonze. Implementation of the projector augmented-wave method in the abinit code: Application to the study of iron under pressure. Comp. Mat. Sci., 42:337, 2008.

[113] M. Valiev and J. H. Weare. The projector-augmented plane wave method applied to molecular bonding. J. Phys. Chem. A, 103:10588, 1999.

[114] J. Bylaska, Michel Dupuis, So Hirata, Lisa Pollack, Dayle M. Smith, Tjerk P. Straatsma, and Edoardo Aprá. Nwchem: New functionality. Lecture Notes in Computer Science, 2660:168, 2003.

[115] M. Valiev, E. J. Bylaska, N. Govind, K. Kowalski, T. P. Straatsma ahd H. J. J. Van Dam, D. Wang, J. Nieplocha, E. Apra, T. L. Windus, and W. A. de Jong. Nwchem: A comprehensive and scalable open-source solution for large scale molecular simulations. Computer Physics Communications, 181:1477–1489, 2010.

[116] Winfried Kromen. Die Projector Augmented Wave-Methode: Ein schnelles Allelektronenverfahren für die ab-initio-Molekulardynamik. PhD thesis, RWTH Aachen, 2001.

[117] Paolo Giannozzi, Stefano Baroni, Nicola Bonini, Matteo Calandra, Roberto Car, Carlo Cavazzoni, Davide Ceresoli, Guido L Chiarotti, Matteo Cococcioni, Ismaila Dabo, Andrea Dal Corso, Stefano Fabris Stefano de Gironcoli and, Guido Fratesi, Ralph Gebauer, Uwe Gerstmann, Anton Kokalj Christos Gougoussis5, Michele Lazzeri, Layla Martin-Samos, Nicola Marzari, Francesco Mauri, Riccardo Mazzarello, Stefano Paolini, Alfredo Pasquarello, Lorenzo Paulatto, Carlo Sbraccia, Sandro Scandolo, Gabriele Sclauzero, Ari P Seitsonen, Alexander Smogunov, Paolo Umari, and Renata M Wentzcovitch. Quantum espresso: a modular and open-source software project for quantum simulations of materials. J. Phys.: Condens. Matter, 21:395502, 2009.

[118] J. J. Mortensen, L. B. Hansen, and K. W. Jacobsen. Real-space grid implementation of the projector augmented wave method. Phys. Rev. B, 71:35109, 2005.

[119] Marc Torrent, N.A.W. Holzwarth, Francois Jollet, David Harris, Nicholas Lepley, and Xiao Xu. Electronic structure packages: Two implementations of the projector augmented wave (paw) formalism. Comp. Phys. Comm., 181:1862, 2010.

[120] P.E. Blöchl. Electrostatic decoupling of periodic images of plane wave expanded densities and derived atomic point charges. J. Chem. Phys., 103:7422, 1995.

[121] T.K. Woo, P.M. Margl, P.E. Blöchl, and T. Ziegler. A combined car-parrinello qm/mm implementation for ab initio molecular dynamics simulations of extended systems: Application to transition metal catalysis. J. Phys. Chem. B, 101:7877, 1997.

[122] O. Bengone, M. Alouani, P.Blöchl, and J. Hugel. Implementation of the projector augmented-wave lda+u method: Application to the electronic structure of nio. Phys. Rev. B, 62:16392, 2000.

[123] B. Arnaud and M. Alouani. All-electron projector-augmented-wave gw approximation: Application to the electronic properties of semiconductors. Phys. Rev. B., 62:4464, 2000.

[124] D. Hobbs, G. Kresse, and J. Hafner. Fully unconstrained noncollinear magnetism within the projector augmented-wave method. Phys. Rev. B, 62:11556, 2000.

[125] H.M. Petrilli, P.E. Blöchl, P.Blaha, and K. Schwarz. Electric-field gradient calculations using the projector augmented wave method. Phys. Rev. B, 57:14690, 1998.

[126] P.E. Blöchl. First-principles calculations of defects in oxygen-deficient silica exposed to hydrogen. Phys. Rev. B, 62:6158, 2000.

[127] Chris J. Pickard and Francesco Mauri. All-electron magnetic response with pseudopotentials: Nmr chemical shifts. Phys. Rev. B, 63:245101, 2001.

[128] F. Mauri, B.G. Pfrommer, and S.G. Louie. Ab initio theory of nmr chemical shifts in solids and liquids. Phys. Rev. Lett., page 5300, 1996.

[129] D. N. Jayawardane, C. J. Pickard, L. M. Brown, and M. C. Payne. Cubic boron nitride: Experimental and theoretical energy-loss near-edge structure. Phys. Rev. B, 64:115107, 2001.

[130] H. Kageshima and K. Shiraishi. Momentum-matrix-element calculation using pseudopotentials. Phys. Rev. B, 56:14985, 1997.

[131] L. Verlet. Computer "experiments" on classical fluids i. thermodynamical properties of lennard-jones molecules. Phys. Rev., 159:98, 1967.

[132] Jean-Paul Ryckaert, Giovanni Ciccotti, and Herman J. C. Berendsen. Numerical integration of cartesian equations of motion of a system with constraints: molecular dynamics of alkenes. J. Comput. Phys., 23:327, 1977. doi: doi:10.1016/0021-9991(77)90098-5.

[133] Peter E. Blöchl, O. Jepsen, and O. K. Andersen. Improved tetrahedron method for brillouin-zone integrations. Phys. Rev. B, 49:16223–16233, Jun 1994. doi: 10.1103/PhysRevB.49.16223. URL http://link.aps.org/doi/10.1103/PhysRevB.49.16223.

[134] O. Jepsen and O.K. Andersen. The electronic structure of h.c.p. ytterbium. Sol. St. Commun., 9:1763, 1971.

[135] G. Lehmann and M. Taut. On the numerical calculation of the density of states and related properties. Phys. Stat. Sol. B, 54:469, 1972.

[136] G. te Velde and E.J. Baerends. Slab versus cluster approach for chemisorption studies. cooncu(loo). Chem. Phys., 177:399, 1993.

[137] F.D. Murnaghan. The compressibility of media under extreme pressures. Proc. Natl. Acad. Sci. U.S.A., 30:244, 1944.

[138] Peter E. Blöchl, Peter Margl, and Karlheinz Schwarz. Ab Initio Molecular Dynamics with the Projector Augmented Wave Method, volume 629 of ACS Symposium Series, chapter 5, pages 54–69. American Chemical Society, 1996. doi: 10.1021/bk-1996-0629.ch004. URL http://pubs.acs.org/doi/abs/10.1021/bk-1996-0629.ch004.

[139] I.N. Bronstein and K.A. Semendjajew. Taschenbuch der Mathematik. BSB B.G. eubner Verlagsgesellschaft, Leipzig, 1983.

[140] U. von Barth and C. D. Gelatt. Validity of the frozen-core approximation and pseudopotential theory for cohesive energy calculations. Phys. Rev. B, 21:2222, 1980.

[141] O.V. Yazyev, I. Tavernelli, L. Helm, and U. Röthlisberger. Core spin-polarization correction in pseudopotential-based electronic structure calculations. Phys. Rev. B, 71:115110, 2005.

[142] P. Carloni, P.E. Blöchl, and M. Parrinello. Electronic structure of the cu, zn superoxide dismutase active site and its interactions with the substrate. J. Phys. Chem., 99:1338, 1995.

[143] G. Makov and M. C. Payne. Periodic boundary conditions in ab initio calculations. Phys. Rev. B, 51:4014, 1995.

[144] E. Siever et al. Linux in a Nutshell, A desktop Quick Reference. O'Reilly, 1997.

[145] N.D. Mermin. Thermal properties of the inhomogeneous electron gas. Physical Review, 137:A1441, 1965. doi: 10.1103/PhysRev.137.A1441. URL http://link.aps.org/doi/10.1103/PhysRev.137.A1441.

Index