Humboldt-Universität zu Berlin

Mathematisch-Naturwissenschaftliche Fakultät

Institut für Physik

**Bachelorarbeit**

zur Erlangung des akademischen Grades Bachelor of Science (B.Sc.)
im Fach Physik

# Visualization of Fermi Surfaces for Large-Scale Databases

Eingereicht von:

**Jan Hagen Stutz**

| | |
|---|---|
| **Gutachter/innen**: | Prof. Dr. Claudia Draxl |
| | PD PhD Pasquale Pavone |
| **Betreuung**: | Prof. Dr. Claudia Draxl |

Eingereicht am Institut für Physik der Humboldt-Universität zu Berlin
am 15.04.2025

# Contents

# 1 Introduction

The significance of Fermi surfaces arises from a fundamental physical principle: Only energy states in close proximity to the Fermi surface can actively participate in a material's response to external stimuli at temperatures around and below room temperature [1]. Consequently, the topology and characteristics of Fermi surfaces strongly determine many material properties, including electronic, optical, thermal, and magnetic properties [2, 3]. With this knowledge, the concept of Fermi-surface engineering emerged as an approach to design and control material properties for specific applications [4]. Only metallic materials possess a well-defined Fermi surface. All considerations in this thesis are made under the assumption that they take place at 0 Kelvin.

The goal of this thesis is the implementation of an easy-to-use, interactive Fermi-surface visualization software package named **`FSvisual`** [5]. It should be well integrable to web-applications and offer an intuitive user interface for interacting with the visualized Fermi surfaces. To achieve this, **`FSvisual`** pairs every energy value for each band to a corresponding point in a 3D grid. This grid is built after the open-source software **`XCrySDen`**, which established the file type providing the band-energy data. On this grid of band-structure data, the *Marching Cubes* algorithm [6] is applied, that creates the Fermi surface as a triangulated structure.

The band-energy data which are used as starting point for this thesis, are obtained by first-principle calculations performed by Qiang Fu [7] using the software package **`exciting`** [8]. This code implements the density-functional theory for obtaining the electronic band structure [9]. These data, first downloaded for 37 elemental metals from the NOMAD database [10], are then parsed by **`FSvisual`**.

This thesis is structured as follows: In Chapter 2 a brief introduction to direct and reciprocal lattices of a crystal, electronic structure, density-functional theory, and the Marching Cubes algorithm is given. Chapter 3 provides information about the heritage, composition, and access to the band-energy data. In Chapter 4 the implementation of **`FSvisual`** is covered. Chapter 5 presents the results of the thesis, starting with an introduction on how to use **`FSvisual`**. Additionally, all the visualized Fermi surfaces are presented on a website [11]. Chapter 6 provides a summary of the thesis as well as considerations for future work.

# 2 Theoretical Background

This chapter provides an overview of the theoretical background needed in this thesis. Unless not stated otherwise, we follow the detailed descriptions in the textbooks of Gross and Marx [12], Ashcroft [13], and Griffiths [14].

## 2.1 Direct and reciprocal lattice

Many solid materials consist of periodically repeating structural elements, *i.e.*, they are crystals. The *crystal structure* describes the arrangement of atoms in these materials. The repeating unit of atoms is the *unit cell* (UC) with volume:

$$V_{\mathrm{UC}} = |\boldsymbol{a}_1 \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3)|, \tag{2.1}$$

where $\boldsymbol{a}_1$, $\boldsymbol{a}_2$, and $\boldsymbol{a}_3$ are the *unit-cell vectors*. If the UC contains only a single lattice point, it is called *primitive cell*. The lengths of the UC vectors are the *lattice constants*. The set of all vectors $\boldsymbol{R}$ that can be obtained as

$$\boldsymbol{R} = n_1\boldsymbol{a}_1 + n_2\boldsymbol{a}_2 + n_3\boldsymbol{a}_3\,, \tag{2.2}$$

where $n_1$, $n_2$, and $n_3$ have integer values, identifies the *Bravais lattice* or *direct lattice* of the crystal. For every Bravais lattice, there exists a *reciprocal lattice*, composed by the vectors:

$$\boldsymbol{G} = m_1\boldsymbol{b}_1 + m_2\boldsymbol{b}_2 + m_3\boldsymbol{b}_3, \tag{2.3}$$

where $m_1$, $m_2$, and $m_3$ are integers and $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{b}_3$ are basis vectors in reciprocal space. These vectors can be derived from the basis vectors of the direct lattice using the relations:

$$\boldsymbol{b}_1 = \frac{2\pi}{V_{\mathrm{UC}}}\boldsymbol{a}_2 \times \boldsymbol{a}_3\,; \qquad \boldsymbol{b}_2 = \frac{2\pi}{V_{\mathrm{UC}}}\boldsymbol{a}_3 \times \boldsymbol{a}_1\,; \qquad \boldsymbol{b}_3 = \frac{2\pi}{V_{\mathrm{UC}}}\boldsymbol{a}_1 \times \boldsymbol{a}_2\,. \tag{2.4}$$

The (first) Brillouin zone (BZ) is the Wigner-Seitz cell in reciprocal space. It is defined as the region, in which any $\boldsymbol{k}$-point is closer to an arbitrarily chosen lattice point than to any other lattice point. The left panel of Figure 2.1 shows the example of a BZ for the face-centered cubic (fcc) lattice. The right panel of Figure 2.1 shows

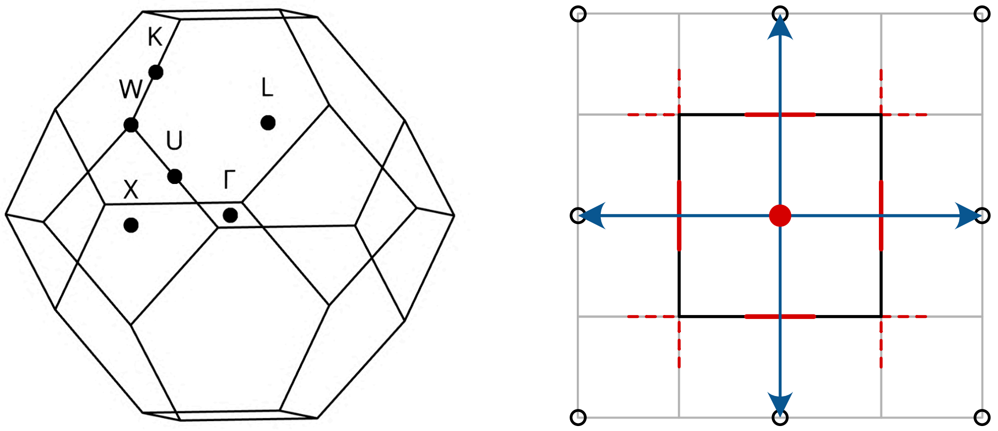


**Figure 2.1:** First BZ of the three-dimensional fcc lattice with high-symmetry points (left) and for a two-dimensional square lattice (right). In the right panel, empty circles (the full red circle) represent reciprocal-lattice points (the Γ point), black/red lines indicate the border of the first BZ, and blue arrows represent reciprocal lattice vectors.

how the BZ is constructed for the representative case of a two-dimensional square lattice. The red circle marks the origin of the first BZ, representing the chosen lattice point. At the midpoint between this lattice point and each of its nearest neighbors, a plane is defined (a line in two dimensions), that is perpendicular to the connecting line (blue) between the neighbor and the BZ origin. The intersections of these planes define the vertices of the BZ. Vertices are defined as points in space where two or more lines, edges, or curves meet. The BZ is identified as the region where every point can be reached from its origin, without crossing any boundary plane.

## 2.2 Electronic band structure

In the following, we introduce the fundamental concepts that are required to introduce the concept of Fermi surfaces. To do so, we first make use of the *free-electron gas* (FEG) approximation. Here, the main assumption is that both electron-electron and electron-nuclei interactions are neglected. Then, the energy states of an electron with periodic boundary conditions in a large cubic volume $V = L^3$, can be calculated by finding the solution of the single-particle time-independent Schrödinger equation:

$$\hat{h}_{\mathrm{FEG}}\, \phi_{\boldsymbol{k}}^{\mathrm{FEG}}(\boldsymbol{r}) \equiv -\frac{\hbar^2\nabla^2}{2m}\, \phi_{\boldsymbol{k}}^{\mathrm{FEG}}(\boldsymbol{r}) = E_{\boldsymbol{k}}^{\mathrm{FEG}}\, \phi_{\boldsymbol{k}}^{\mathrm{FEG}}(\boldsymbol{r}) \tag{2.5}$$

where $m$ is the free-electron mass. The normalized solutions of Eq. (2.5) for a given wave vector $\boldsymbol{k}$ are plane waves of the form:

$$\phi_{\boldsymbol{k}}^{\mathrm{FEG}}(\boldsymbol{r}) = \frac{1}{\sqrt{V}} \exp(i\boldsymbol{k} \cdot \boldsymbol{r}), \tag{2.6}$$

with energies

$$E_{\boldsymbol{k}}^{\mathrm{FEG}} \equiv E^{\mathrm{FEG}}(\boldsymbol{k}) = \frac{\hbar^2 \boldsymbol{k}^2}{2m} = E^{\mathrm{FEG}}(k). \tag{2.7}$$

Due to the periodic boundary conditions, the only allowed values for $\boldsymbol{k}$ are:

$$k_x = \pm\frac{2\pi n_x}{L}, \quad k_y = \pm\frac{2\pi n_y}{L}, \quad k_z = \pm\frac{2\pi n_z}{L}, \tag{2.8}$$

where $n_x, n_y$, and $n_z$ are quantum numbers, indexing the states. Because of Pauli's exclusion principle, every state can be occupied by two electrons of opposite spin, *i.e.*, spin-up and spin-down. The available states are then filled sequentially from the lowest ($n_x$=$n_y$=$n_z$=0 ) to the highest energy. The energy eigenvalue of the highest occupied state, $E_\mathrm{F}$, is called *Fermi energy* or *Fermi level.* Due to the fact that the energy in Eq. (2.7) depends only on the length of the vector $\boldsymbol{k}$, the number of states that share the same energy increases with this length. In the continuous limit ($V \rightarrow \infty$) the $\boldsymbol{k}$-vectors which correspond to the occupied states at the Fermi energy define the surface of a solid which is called *Fermi sphere.*

The Fermi energy, $E_\mathrm{F}$, can be computed from the electronic density-of-states (DOS). For the FEG, it is given by [12]:

$$D(E) = \frac{V}{2\pi^2}\left(\frac{2m}{\hbar^2}\right)^{3/2}\sqrt{E} \tag{2.9}$$

so that the integral of $D(E)$ from 0 to $E_\mathrm{F}$ equals the number of electrons $N$ in the system:

$$\int_0^{E_\mathrm{F}} D(E)\, dE = N. \tag{2.10}$$

Figure 2.2 shows the DOS of the FEG with a separation in occupied (blue) and unoccupied states at the Fermi energy. Inserting Eq. (2.9) into Eq. (2.10) allows to calculate the Fermi level:

$$\begin{aligned} N &= \frac{V}{2\pi^2}\left(\frac{2m}{\hbar^2}\right)^{3/2}\int_0^{E_\mathrm{F}}\sqrt{E}\, dE \\ &= \frac{V}{3\pi^2}\left(\frac{2m}{\hbar^2}\right)^{3/2} E_\mathrm{F}^{3/2} \\ E_\mathrm{F} &= \frac{\hbar^2}{2m}\left(3\pi^2 n\right)^{2/3}. \end{aligned} \tag{2.11}$$

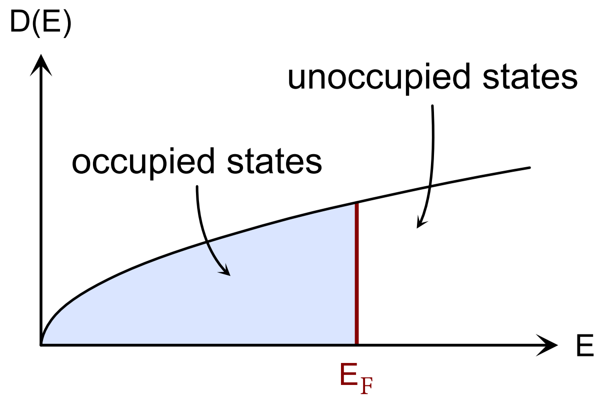


**Figure 2.2:** Density of states $D(E)$ as a function of the energy $E$ for a FEG. Occupied states are indicated in blue and the red line represents the Fermi level $E_\mathrm{F}$.

Only few materials behave like FEGs. In order to improve the theoretical description of the electrons, we introduce a lattice of static positive atomic cores. We consider the periodic potential of that lattice to be very weak (almost vanishing), allowing the electrons to move as in a FEG. This approximation is commonly denoted as *empty lattice* model. In contrast to the energy spectrum of the FEG, the solution of the Schrödinger equation for electrons in the described system yields a *set* of energy values $E_n(\boldsymbol{k})$ for every $\boldsymbol{k}$-vector. Each energy value of that set belongs to an energy band, which is indexed by $n$. These energy bands consist of energy states that electrons are allowed to occupy.

## Reduced-zone scheme

In Figure 2.3, the energy bands of the FEG-like system under consideration are depicted for a one-dimensional example. The dotted lines mark the border of the first BZ. An expression for the FEG energy bands is given by:

$$E_n(\boldsymbol{k}) = \frac{\hbar^2(\boldsymbol{k} + \boldsymbol{G}_n)^2}{2m}\,, \tag{2.12}$$

where $\boldsymbol{G}_n$ is a specific reciprocal lattice vector that extends from the lattice point at the origin to any lattice point in $\boldsymbol{k}$-space. This makes the band structure (BS) appear identical at every lattice point to meet the periodicity of the lattice. In case of $n$ equals zero, $\boldsymbol{G}_0$ is also zero. Since the BS shares the periodicity of the lattice, as demonstrated by Eq. (2.12), it is sufficient to consider the BS within the first BZ of the underlying lattice. This representation is referred to as the *reduced-zone scheme*, shown in Figure 2.4. Each energy band originating from the various lattice points is represented within the reduced zone, collectively forming the overall BS, as illustrated by bands $n$=2 and $n$=3. This behavior is often easier to understand as a folding process at the BZ boundary: When an energy band extends beyond the BZ,

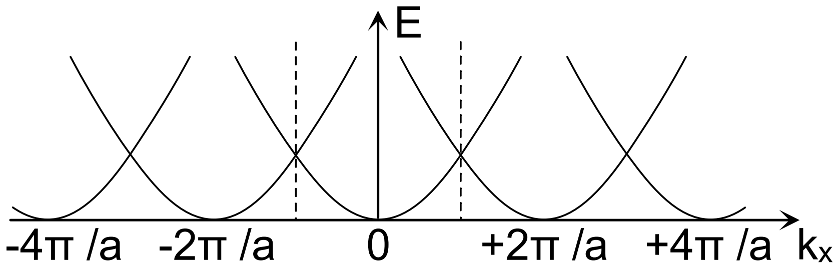


**Figure 2.3:** Electron bands in one dimensional reciprocal space of a FEG in the empty-lattice model. The dotted lines mark the BZ border.

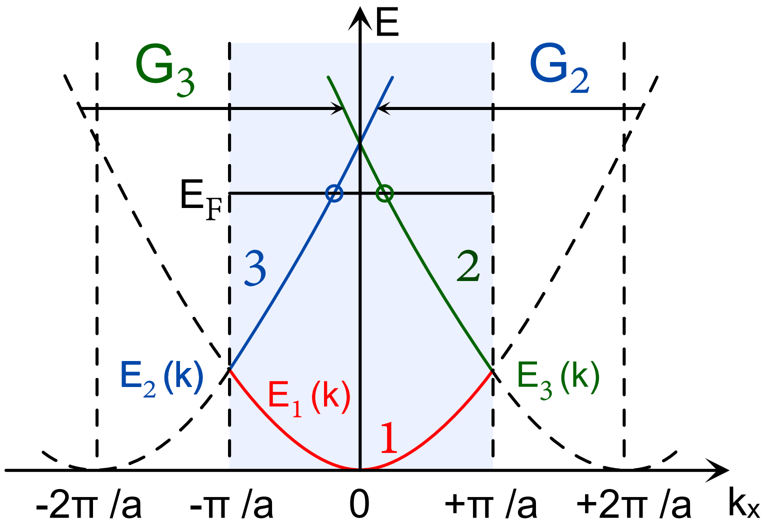


**Figure 2.4:** One-dimensional BS of a FEG in the empty-lattice model within the reduced-zone (light blue) scheme. With the vectors $G_2$ and $G_3$ the folding process of the energy band originating at $k_x = 0$ (red) is illustrated, yielding band 2 (dark blue) and 3 (green).

it is folded back into it by adding the corresponding reciprocal-lattice vector $\boldsymbol{G}_n$, mapping each point outside the reduced zone back into the reduced zone.

### High-symmetry points

Visualizing the BS of a three dimensional material is challenging. This is because, in addition to $\boldsymbol{k}$-space coordinates, the band energies have to be displayed on every point. Thus, for a three-dimensional lattice, a four-dimensional plot would be required to fully represent the BS.

To address this challenge, we introduce *high-symmetry points*, which are specific locations in the BZ that remain invariant under certain symmetry operations of the crystal. Consequently, they vary across different crystal structures, depending on the crystal-structure symmetry. High-symmetry points allow for a visualization of

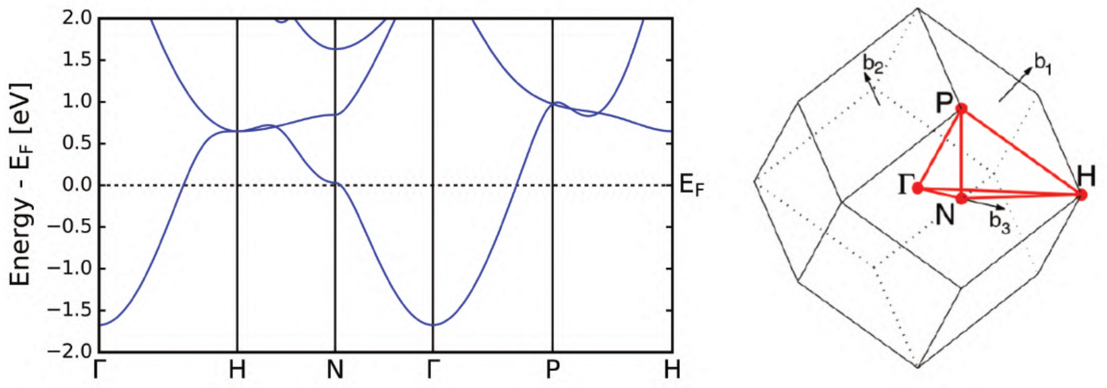


**Figure 2.5:** Left: BS for Cs along the standard bcc high-symmetry path in reciprocal space. The zero of the energy axis is taken at the Fermi energy. Right: BZ, high-symmetry points, and standard $\boldsymbol{k}$-path (solid red lines) for a bcc crystal (reproduced from [15]).

a BS for a three-dimensional material in a two-dimensional plot along the so-called standard paths [15]. Figure 2.5 displays on the left an example of a BS along a path containing the high-symmetry points for Cs in the bcc crystal structure. On the right, the high-symmetry points and the standard path between them [15] are shown for the bcc case.

If an energy band crosses the Fermi level in a three-dimensional space, the resulting cross-sectional surface is called *Fermi surface*. Figure 2.6 shows a schematic example of the BS and their relationship to the Fermi surface (FS) for insulators/semiconductors, metals, and semi-metals simplified in one dimension. From Figure 2.6 it can be seen that FSs are only relevant for metals and semimetals, since the Fermi level of semiconductors and insulators (left panel) lies within the band gap and thus no occupied state can match the Fermi level. On the contrary, for metals (center panel) and semimetals (right panel) there are energy bands that contain occupied as well as unoccupied states resulting in a FS. Furthermore, a FS can have contributions that originates from different bands. In this case the different contributing parts are usually called *branches* of the FS.

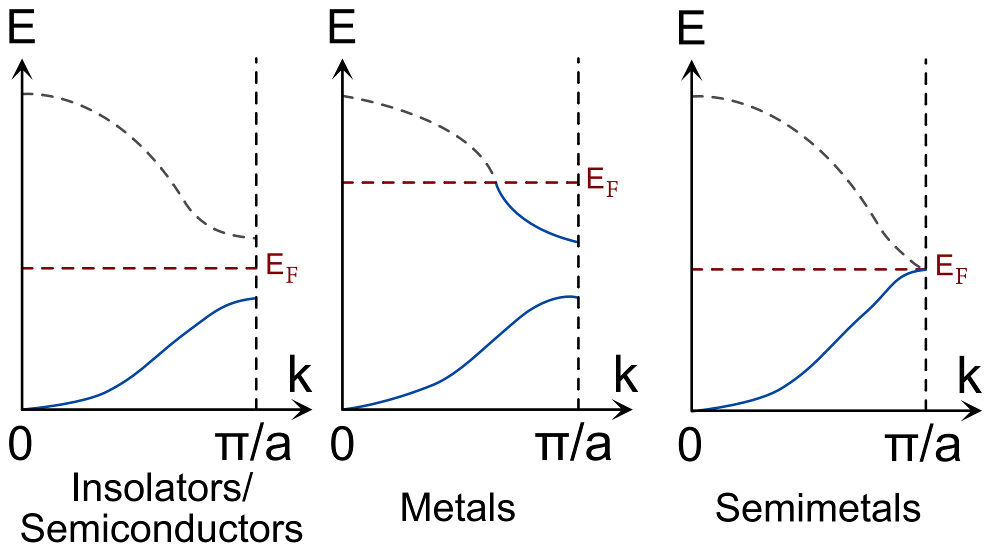


**Figure 2.6:** Schematic representation of the BS around the Fermi level of a one-dimensional insulator/semi-conductor, metal, and semi-metal. The Fermi level is represented by red dotted lines. The intersection between an occupied energy band and the Fermi level (one single point in this example) defines the Fermi "surface". The occupied states are represented by straight blue lines the unoccupied states by dashed gray lines (see text for discussion).

## 2.3 Density-functional theory

The electronic BS of the metals considered in this thesis has been calculated by Qiang Fu [7] using density-functional theory (DFT). DFT is based on the Hohenberg-Kohn theorem [9], which allows to write the ground-state total energy $E$ of a system of interacting electrons in an external potential $v(\boldsymbol{r})$ as a sum of an universal functional $F[n]$ of the ground-state electron density $n(\boldsymbol{r})$ and a contribution which depends explicitly on $v(\boldsymbol{r})$:

$$E[n] = F[n] + \int v(\boldsymbol{r})\, n(\boldsymbol{r})\, d\boldsymbol{r}\,. \tag{2.13}$$

This approach is advantageous over the wave-function based methods in that it solely depends on $n(\boldsymbol{r})$. The electron density is a function of only three coordinates, whereas the many-body wave-function $\Psi(\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_N)$ depends on $3N$ coordinates. An analytical expression for $F[n]$ is unknown, indicating that for practical purposes it must be approximated.

As demonstrated by Kohn and Sham (KS) [16], a many-particle problem of interacting electrons can be mapped onto a system of non-interacting electrons with the same ground-state electron density as the real system. The standard KS approach to DFT starts by separating from $F[n]$ everything that is explicitly known,

$$F[n] = T_0[n] + E_{\mathrm{H}}[n] + E_{\mathrm{XC}}[n]\,, \tag{2.14}$$

where $T_0[n]$ is the kinetic-energy functional of a *non-interacting* electron system with ground-state density $n(\boldsymbol{r})$, $E_{\mathrm{H}}[n]$ is the Hartree energy functional, which covers the classical Coulomb interaction, and $E_{\mathrm{XC}}[n]$ is the unknown exchange-correlation (XC) energy functional. In the KS approach, the ground-state electron density is obtained by solving the single-particle Schrödinger-like KS equations, which effectively describe a system of $N$ non-interacting particles:

$$\left[-\frac{1}{2}\nabla^2 + v_{\mathrm{eff}}(\boldsymbol{r})\right]\psi_i(\boldsymbol{r}) = \epsilon_i\,\psi_i(\boldsymbol{r}) \qquad \text{with } \ i = 1, 2, \ldots, N\,, \tag{2.15}$$

where $\psi_i(\boldsymbol{r})$ is a single-particle orbital and $\epsilon_i$ is the corresponding eigenenergy. The effective potential $v_{\mathrm{eff}}(\boldsymbol{r})$ can be written as follows:

$$v_{\mathrm{eff}}(\boldsymbol{r}) = v(\boldsymbol{r}) + \frac{\delta E_{\mathrm{H}}[n]}{\delta n(\boldsymbol{r})} + \frac{\delta E_{\mathrm{XC}}[n]}{\delta n(\boldsymbol{r})} \equiv v(\boldsymbol{r}) + v_{\mathrm{H}}(\boldsymbol{r}) + v_{\mathrm{XC}}(\boldsymbol{r})\,. \tag{2.16}$$

Equation (2.16) defines the Hartree potential, $v_{\mathrm{H}}(\boldsymbol{r})$, and the XC potential, $v_{\mathrm{XC}}(\boldsymbol{r})$. Then, the ground-state electron density is obtained as:

$$n(\boldsymbol{r}) = \sum_{i=1}^{N} |\psi_i(\boldsymbol{r})|^2\,. \tag{2.17}$$

## Exchange-correlation energy functionals

Since $E_{\mathrm{XC}}[n]$ is unknown, it has to be approximated. Typical approximations for this term are the local-density approximation (LDA) [17] and the generalized-gradient approximation (GGA). Popular GGA functionals are, *e.g.*, the Perdew-Burke-Ernzerhof (PBE) functional [18] and PBEsol [19].

In the simplest form among the three, LDA, the XC energy is given by:

$$E_{\mathrm{XC}}^{\mathrm{LDA}}[n] = \int n(\boldsymbol{r})\,\epsilon_{\mathrm{XC}}^{\mathrm{HEG}}\big(n(\boldsymbol{r})\big)\,d\boldsymbol{r}, \tag{2.18}$$

where $\epsilon_{\mathrm{XC}}^{\mathrm{HEG}}(n)$ represents the exchange-correlation energy per particle of the homogeneous interacting electron gas (HEG) with constant density $n$. The LDA functional typically underestimates unit-cell volumes (or equivalently, overestimates the BZ volume) and thus produces the largest reciprocal-lattice parameters among the XC functionals under consideration.

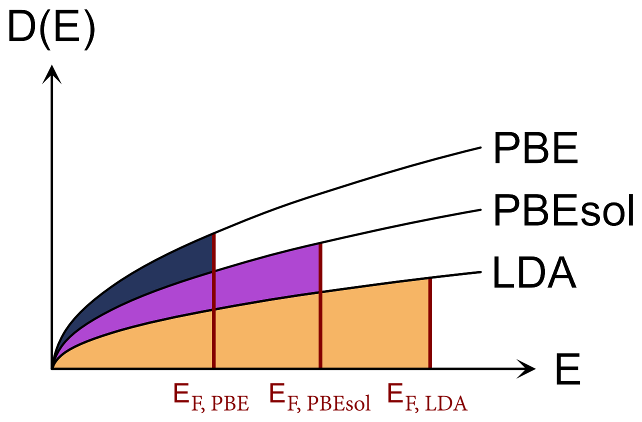


**Figure 2.7:** Schematic illustration of the DOS and Fermi levels of a FEG-like metal obtained by calculations performed using PBE, PBEsol, and LDA. Here, for simplicity, the lowest energy in the bands are aligned.

On the contrary, PBE slightly overestimates unit-cell volumes, while PBEsol typically stays between the two [20]. For metals with FEG-like BS, these differences in the UC volume result in the lowest DOS values at the Fermi energy for LDA and in the highest for PBE (see Eq. (2.9)). Thus, for these systems, a lower DOS leads to a higher Fermi energy, since less states can be occupied. This behavior is represented schematically in Figure 2.7, where the areas below the curves are separated into occupied (colored) and unoccupied states.

### The `exciting` code

The DFT calculation of the electronic band-structure of the metals considered in this thesis were performed using the **`exciting`** code [8]. **`exciting`** is an all-electron full-potential code which implements density-functional theory for the ground-state properties of materials. The **`exciting`** code is not restricted to ground-state calculations, but has a major focus on excited-state properties.

## 2.4 Marching Cubes

The calculated band-energy data are usually available at a grid of points in the BZ. This discretization has the inconvenience that the FS directly calculated from these data can be quite rough. In order to obtain a smooth FS, the data needs to be post-processed. Here, we use the *Marching Cubes* (MC) algorithm [6]. This is a general method to construct triangulated surfaces from a *voxel graphic*, which is a 3D object represented within a regular Cartesian grid of coordinate points. In our case these voxel graphics are the FSs, where each point in $\boldsymbol{k}$-space gets a band-energy value assigned as an attribute. The combination of grid point and attribute is called *voxel*. The MC algorithm for the identification of the FS within the grid of $\boldsymbol{k}$-points (voxel grid in the remainder) consists of four major steps. These steps are

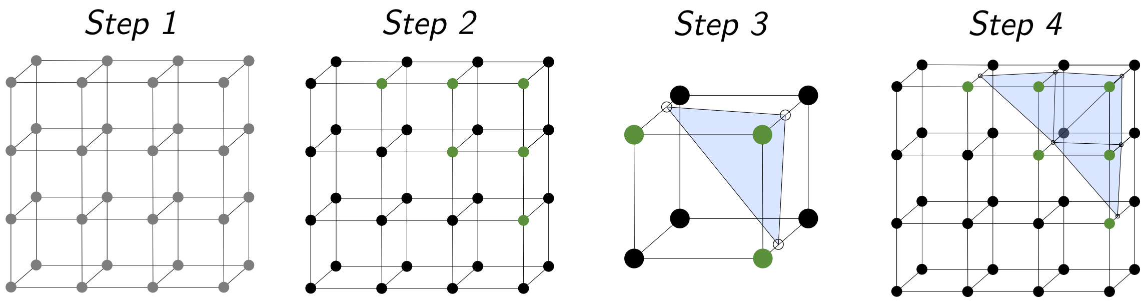


**Figure 2.8:** Illustration of the four steps of the MC algorithm. In the first, second, and fourth step, a voxel grid, partitioned into cubes, is displayed. The third step shows an arbitrarily chosen cube of the grid to illustrate how a triangulated surface is built. The green (black) circles mark vertices with energy values larger (smaller) than the Fermi energy. In Step 3 along the 3 edges connecting marked and unmarked vertices, a corner point of a triangle (light blue) is located, building the FS.

shown in Figure 2.8. First, as shown in the outermost left panel, the MC algorithm partitions the voxel grid into cubes, where each voxel serves as a corner point of a cube. In the second step (center-left panel), the algorithm iterates over all cubes to determine whether the band energy at each corner voxel is greater or less than the Fermi energy. If a corner's value exceeds (or equals) this energy, it is marked. In Figure 2.8, the marked corners are colored in green (in step two, three, and four). The center right panel of Figure 2.8 shows Step 3: For all edges, it is tested if one of their corners is marked and the neighbor is not, which indicates that the FS intersects that edge. Since each of the eight corners can be either marked or unmarked, there are $2^8 = 256$ possible ways the FS can intersect the edges of a cube. Each of the 256 cases can be identified using an 8-digit binary index, which assigns the value 1 for marked corners and 0 for the others. However, when considering symmetry, this number of distinct cases can be reduced to 15, which are shown in Figure 2.9. These 15 configurations are stored in a lookup table that the algorithm uses during computation to construct the surface. Depending on the eight-digit index, the lookup table either returns the stored configuration, or an equivalent symmetric version. For instance, the configuration labeled as index one in Figure 2.9 depicts a single marked vertex located at the front-bottom-left corner of the cube. However, an equivalent triangulation would result if any of the other seven corners were marked instead. Since the lookup table contains only one representative configuration per case of a single marked vertex, the algorithm must apply appropriate symmetry operations, such as mirroring or rotating the stored configuration, depending on the actual position of the marked vertex within the cube. In Step 4, the exact position of each vertex along each edge is determined by linear interpolation.

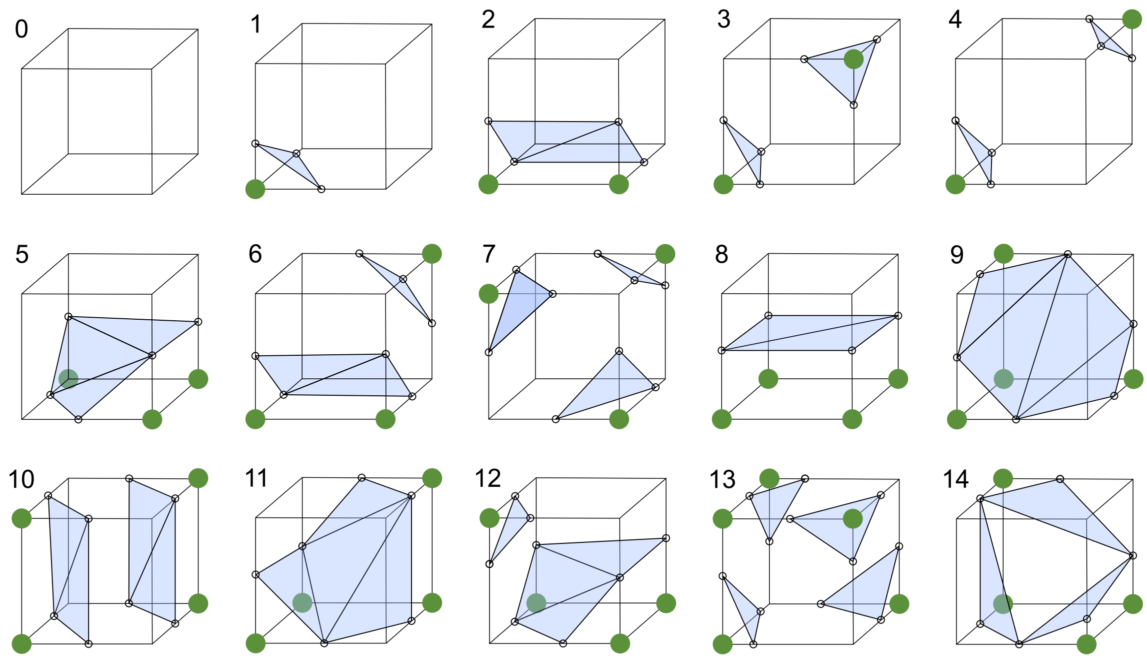


**Figure 2.9:** The 15 possible ways in which a surface composed of triangles (light blue) can intersect a cube.

When the MC algorithm is completed, we retrieve a connected surface consisting of triangles. The continuity is given because of the fact that adjacent cubes always share 4 voxels on their adjacent side with each other. Thus, for these 4 vertices, the linear interpolation yields the same results for both cubes as shown in Step 4 of Figure 2.8. The MC algorithm then outputs all vertices of the triangles, together with the order in which these vertices should be connected to form the triangles. These orders are called *facets*. The accuracy of the MC algorithm is mainly influenced by the resolution of the voxel grid and the chosen interpolation technique.

# 3 Data

In this chapter, we describe the data that is visualized in this work. The band structures (BSs) are computed with **`exciting`** [8]. The functionals used for the BS calculation are PBE, PBEsol, and LDA (see Section 2.3). In total, BS of 37 metals were calculated [7]. The crystal structures under consideration are fcc, bcc, and hcp. These structures are shown in Figure 3.1.

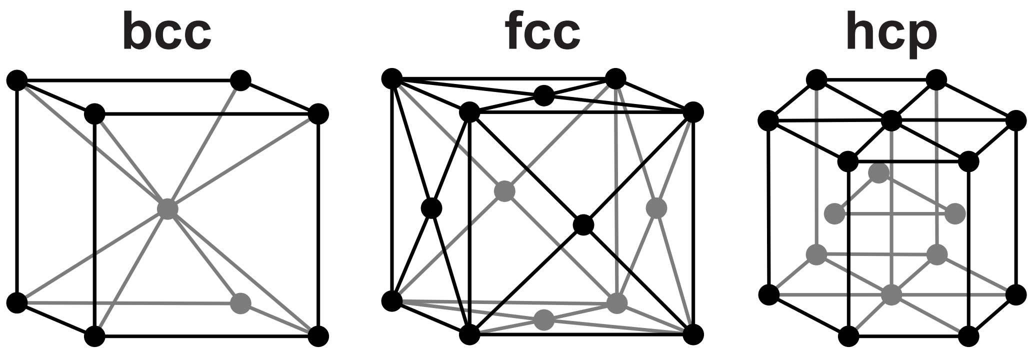


**Figure 3.1:** The crystal structures under consideration in this thesis. The bcc, fcc, and hcp stucture are displayed on the left, middle, and right panel, respectively.

## 3.1 Band-structure files

The band-energy values, used for the determination of the Fermi surface (FS), that are computed with **`exciting`**, are stored in the **`bxsf`** format, which is a file type established by the software project **`XCrySDen`** [21]. These **`bxsf`** files are human readable and contain the energy data in tabular form in the following order:

**i)** The 3 reciprocal basis vectors $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{b}_3$ ;

**ii)** The Fermi energy in units of eV of the considered elemental crystal;

**iii)** The number of energy bands within the file;

**iv)** A triple of integers determining the $\boldsymbol{k}$-point sampling grid;

**v)** The list of energy values in units of eV for each band.

The reciprocal basis vectors are used to calculate the vertices and facets of the first BZ. The Fermi energy is used by the Marching Cubes algorithm as the value at which the Fermi surface is built. The number of energy bands is not needed, since for reading the energy values we use an algorithm which iterates through every band. The knowledge of the partition of the unit cell (UC) in reciprocal space determined by the $\boldsymbol{k}$-point sampling grid is essential for determining at which $\boldsymbol{k}$-point the given band-energy values were calculated. Finally, the list of energy values is parsed in our algorithm for each band as attributes to the mesh grid.

The band-structure files that we used have a very high resolution. The $\boldsymbol{k}$-point sampling grid consists of $161 \times 161 \times 161$ mesh points. This makes for a total of $161^3 = 4173281$ $\boldsymbol{k}$-points for the whole grid.

## 3.2 API access to NOMAD

The data that is used in this work is hosted on NOMAD [10]. NOMAD is the world-wide largest database of DFT calculations, containing more than 15.5 million code runs. More recently, NOMAD has been expanded to include experimental data. NOMAD offers access to the data both through a graphical user-interface (GUI) and a web application programming interface (API). The API allows to search and download the data programmatically. To download the data, we used the latter approach via the **`Python`** package **`requests`**.

To obtain the data files, we retrieved a list from the NOMAD API of all entries associated with Qiang Fu, the author of the dataset. The specific API query [7] used for this purpose is documented in the references. Since each entry contains multiple files, the results had do be filtered to identify the **`bxsf`** files. If an entry included a **`bxsf`** file (which was not always the case), the entry ID and the name of the **`bxsf`** file were recorded. The files were downloaded by sending **`GET`** requests with the corresponding entry IDs and file names to NOMAD. Each downloaded file was categorized based on the XC functional used and was named to include the element name, entry ID, and crystal structure for clear identification [10].

# 4 The Software Package **FSvisual**

The process of visualizing Fermi surfaces (FSs) starting from the raw band-structure data is performed with the help of the software package **`FSvisual`** [5], which we developed for this special purpose. The implementation of **`FSvisual`** for the purpose of obtaining the visualization of the FSs consists of three major steps which are described in this chapter. The first step is to parse the raw band-energy data from **`exciting`** calculations into **`FSvisual`** (see Section 4.1). This action is handled by the **`Python`** script **`input.py`** of the package. At the second step, a $\boldsymbol{k}$-point sampling grid is defined to assign band-energy values to their corresponding $\boldsymbol{k}$-points in the 3D $\boldsymbol{k}$-space (see Section 4.2). This is performed using the **`fermi_surfaces.py`** script. The Marching Cubes (MC) algorithm is then applied to the sampling grid to build the FS. Additionally, the first Brillouin zone (BZ) is constructed and stored using the script **`brillouin_zone.py`**. The third and final step handles the creation of the FS plots (see Section 4.3). In the next sections, the parts of **`FSvisual`** corresponding to the three steps are analyzed in detail.

## 4.1 Parsing the data

Inside the script **`input.py`**, the **`read_energy_numbers`** function reads all values in the **`bxsf`** file (see Section 3.1), excluding the number of energy bands. Since the **`bxsf`** files have a consistent structure, it is possible for the **`read_energy_numbers`** function to iterate over each line and handle each section separately. For instance, the information on the grid size is always located at the 9th line of the **`bxsf`** file for each material. When the function reaches this line, the three integer values defining the $\boldsymbol{k}$-sampling grid are stored to an array. For the band-energy values, the reading process differs slightly. If a line after the previously described sections begins with the keyword "BAND", all subsequent energy values are stored in an array. The process continues until another line starting with "BAND" appears, at which point a new array is created for the following energy values. The arrays generated in this way are appended to a list, which ultimately contains all the band-energy arrays of each band, upon reaching the end of the **`bxsf`** file.

```python
    mesh_axis_one = np.arange(Imin[0], Imax[0] + 1)
    mesh_axis_two = np.arange(Imin[1], Imax[1] + 1)
    mesh_axis_three = np.arange(Imin[2], Imax[2] + 1)

    mesh_axis_one = np.where(mesh_axis_one >= 0, mesh_axis_one,
                   mesh_axis_one + Imax[0])
    mesh_axis_two = np.where(mesh_axis_two >= 0, mesh_axis_two,
                   mesh_axis_two + Imax[1])
    mesh_axis_three = np.where(mesh_axis_three >= 0,
                     mesh_axis_three, mesh_axis_three + Imax[2])

    index_mesh = np.ix_(mesh_axis_one, mesh_axis_two,
                mesh_axis_three)
```

**Listing 4.1:** Code snippet of the **`create_cartesian_mesh`** function to create the $\boldsymbol{k}$-point grid. Here, the abbreviation **`np`** is used for **`NumPy`**. See text for full description.

## 4.2 Building the mesh

To construct the three-dimensional $\boldsymbol{k}$-vectors sampling grid, we re-implemented the algorithm used by **`XCrySDen`**. This step is critical, because there is no explicit information in the **`bxsf`** files about how the band-energy values and $\boldsymbol{k}$-vectors are related to one another. The original implementation in **`XCrySDen`** uses the programming language **`C`**. The direct translation of this code to **`Python`** lead to significant increases in the runtime. This is because, unlike **`C`**, **`Python`** is an interpreted language, meaning that **`Python`** code is executed line by line by the interpreter. This way of execution adds extra processing steps compared to **`C`**, where a code is directly compiled into machine instructions. Consequently, the execution of "for" loops is less efficient with **`Python`**. To improve performance, the function was restructured using **`NumPy`** [22], which replaces explicit loops with efficient matrix operations through vectorization. This allows operations to be applied to entire arrays at once, eliminating the need for element-wise iteration. In addition to enhancing efficiency, **`NumPy`** also makes code easier to read.

The **`create_cartesian_mesh`** function of the script **`fermi_surfaces.py`** creates an index grid in Cartesian coordinates first, containing indices of band-energy values in the band-structure (BS) file at every grid point. This grid is mirrored along each of the reciprocal-space basis vectors, resulting in a grid of eight times the size of the original grid. This is done to ensure that the process of folding the points outside the BZ back into the BZ covers the entire BZ. Subsequently, an energy grid for each band is constructed by replacing the indices of the index grid with the actual band-energy values. A snippet of the source code of this function is shown in Listing 4.1. In the first three lines of Listing 4.1, three one-dimensional sets of ordered integer indices are created, ranging from a negative minimum (**Imin**) and a positive maximum (**Imax** = −**Imin**) value corresponding to the size of the $\boldsymbol{k}$-point sampling in one direction. This is done by the **NumPy.arange** function. Then, the value of **Imax** is added to all negative indices (using **NumPy.where**). The three arrays are subsequently combined in order to generate an ordered list of triples of integer indices by using the **NumPy.ix_** function. This effectively results in the creation of a grid of points that is a $2\times2\times2$ "super" mesh of the original $\boldsymbol{k}$-point sampling grid. Then, the ordering index created by **NumPy.ix_** is used for associating at each grid point the corresponding band energies. For instance, if a grid point has the number 42 as an ordering index, then, the 43rd list of band-energy values is assigned to that point.

## 4.3 Creating Fermi-surface plots

Once the index grid is in place, a grid for each band based on the indices of the index grid is created. The MC algorithm (see Section 2.4) is then applied to these grids, generating surfaces that show the contribution of each band to the FS. Since the MC algorithm can only process inputs from a Cartesian mesh, the grid points are initially not stored in the their reciprocal basis. Consequently, once the algorithm returns the vertices and facets for the FS, a basis transformation is required. Additionally, the surface generated by the MC algorithm may extend outside the first BZ, due to mirroring the grid points of the whole reciprocal UC (see Section 4.2). Since the FS is, by definition, represented only within the BZ boundaries, these external parts are removed using a *slicing algorithm* which is discussed in details in Appendix A. The slicing process is performed using the **Trimesh** [23] library, which is designed to handle triangular meshes. The process addressed in Appendix A is repeated for every energy band listed in the **bxsf** file. The FS resulting from surface contributions of all bands is collected into a list, which is then passed to **Plotly** [24] for visualization.

### 4.3.1 Construction of the first Brillouin zone

**FSvisual** displays the FSs along with the first BZ, requiring the computation of the BZs vertices and facets for the considered crystal structure. This is done using the **Python** package **pymeshlab.core** via the function **get_wigner_seitz_cell()**, which takes the reciprocal basis of the crystal structure as input.

### 4.3.2 Plotting the Fermi surface using `Plotly`

Finally, to create the plots, we implemented a function called **`plot()`** using the routines of the **`Python`** library **`Plotly`** [24]. **`plot()`** takes the vertices and facets of the FS and BZ, as well as the band index as arguments. The function utilizes two types of 3D plots from the **`Plotly.graph_objects`** library: **`Scatter3D`** for point-based visualization and **`Mesh3D`** for surface representation. **`Scatter3D`** takes lists of $(x, y, z)$ coordinates and plots them as points in 3D space. By setting the plot mode to "line" the points are connected by lines in the same order as they are parsed to **`Plotly`**. This feature makes the **`Scatter3D`** an ideal choice for displaying the BZ. **`Mesh3D`**, which is used to visualize the FSs, also takes lists of $(x, y, z)$ coordinates of the vertices along with $i$, $j$, and $l$ indices as facet information. The mesh plot structure expects the facets to be in a triangular shape, which is the case due to the MC algorithm. The indices $i$, $j$, and $l$ correspond to the first, second, an third vertex of the triangle facet, respectively. The **`Mesh3D`** objects of each branch in the **`fermi_surface_list`** are stored in a **`mesh_fermi_surfaces list`**. The **`Scatter3D`** object and the list of **`Mesh3D`** objects are combined into a **`graph_objects`** figure, which handles the final visualization and layout of the FSs. To save the visualized FSs as an **`HTML`** file, we used the **`Plotly.io.write_html()`** function, which takes the figure and a specified file-name as input. The files along with **`svg`** images of the FSs are stored together.

# 5 Results

In this chapter, we present the application of the code **`FSvisual`**, introduced in Chapter 4, to the data presented in Chapter 3, generating interactive 3D visualizations of the 37 metals under consideration. This proceeds through downloading the data from the NOMAD database (Section 3.2), followed by data processing (Sections 4.1 and 4.2), and finally creating Fermi-surface (FS) plots (Section 4.3). In Section 5.1, we provide a guide for using **`FSvisual`**. The interactive website for visual exploration of the FSs is introduced in Section 5.2. Finally, Section 5.3 presents an analysis of the visualized FSs, highlighting cases where they show large variation in dependence of the XC functionals.

## 5.1 Usage of `FSvisual`

If **`bxsf`** files are available, *e.g.*, those generated by **`exciting`**, the basic usage of **`FSvisual`** [5] and the resulting Fermi-surface plots is relatively simple. To start a visualization process, the user has to open the **`program_call.py`** script, which is part of **`FSvisual`**. Within this script, it is necessary to specify the directory path (**`dirpath`**) to indicate where the **`bxsf`** files are located and to set the output directory path (**`save_fermisurf_path`**), where the **`FSvisual`**-generated **`HTML`** scripts and **`SVG`** images of the FS will be stored. The code will automatically iterate through all **`bxsf`** files within the specified directory until all FSs plots and **`SVG`** images are created.

The **`HTML`** scripts for the visualization of the FS can be opened in any web browser. Once the file is loaded, an entire FS will be displayed. The lest panel of Figure 5.1 illustrates the expected appearance of the interface. In this interface, the screen acts as a virtual camera. The user can rotate the camera view of the FS by holding the left mouse button and dragging the cursor in the desired direction. Releasing the button stops the movement. Holding the right mouse button (or pressing the control key on the keyboard + left mouse button) locks the vertical axis of the camera. Subsequent movements will translate the camera along the horizontal axes. Any alteration of the camera position can be reset by reloading the page. The user can zoom in and out using the mouse wheel. An interactive legend is located in the upper-right corner of the screen. As shown in the right panel of

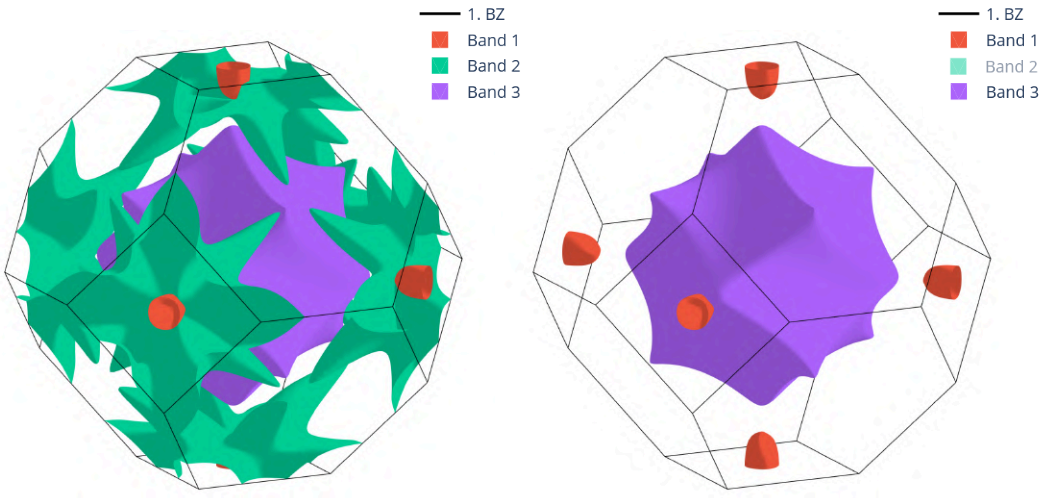


**Figure 5.1:** Left panel: Visualization of the Fermi surface of Pd. The legend at the top of the page rules the appearance of the available bands and BZ. Right panel: Same as the left panel, but with the display of the second band being disabled.

Figure 5.1, the user can hide each FS band, as well as the edges of the Brillouin zone (BZ), by left-clicking on the targeted object in the legend. To reduce the opacity of the FS, the user can adjust the corresponding settings in the **`visualization.py`** file. Lowering the opacity is particularly useful when presenting only images of the FS, as it can improve visibility of internal structures or surface parts facing away from the viewer, without the need of rotating the camera or hiding any branches. It is also possible to assign individual colors for the different FS branches. To do this, add any number of HEX color codes to the **`facecolor`** array in the **`program_call.py`** script. The first color in the array corresponds to the first branch of the **`bxsf`** files, the second to the second branch, and so forth.

## 5.2 Website

The visualizations of the FSs are also available through a specially designed website [11]. This page is created using the *hypertext markup language* (**`HTML`**) and displays a periodic table of elements, where images of the FS are shown for each available element. In Figure 5.2 a snapshot of the periodic table in the website is shown. Every element with an **`SVG`** image is clickable. Once an element is selected, a subpage opens. This subpage shows **`SVG`** images of the FS for all available XC functional for that element. Additionally, the unit-cell volume (UCV) is given together with a link to the entry in the NOMAD database for the original data. Figure 5.3 shows the subpage of Ba as an example. Each image in the subpages can be clicked, to open the interactive visualization of that particular FS.

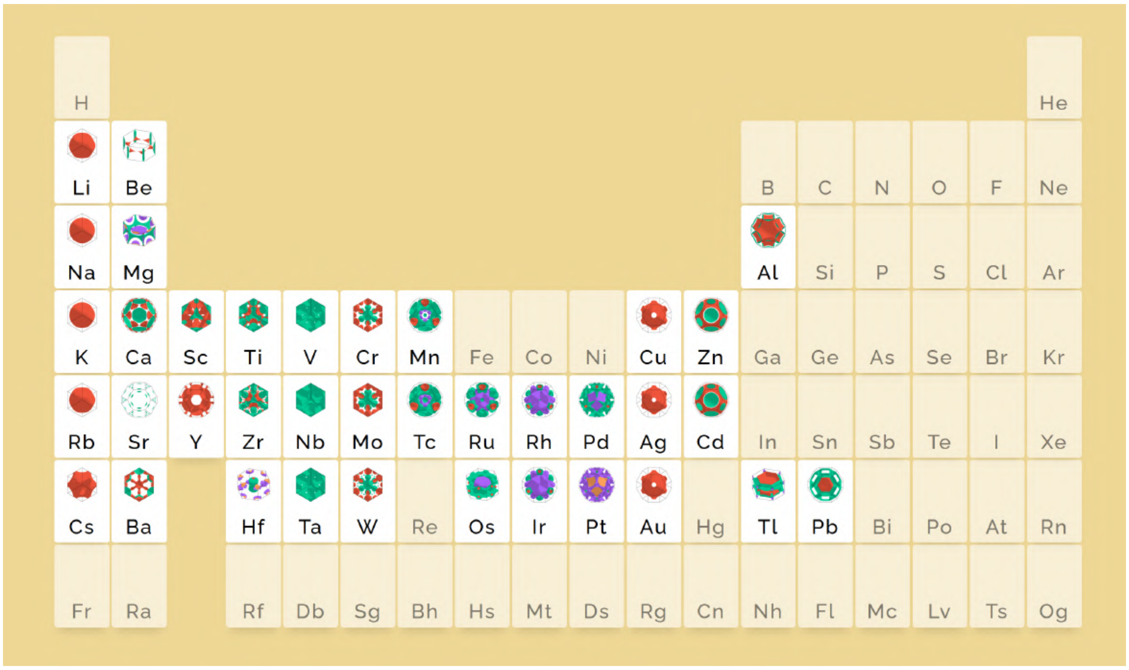
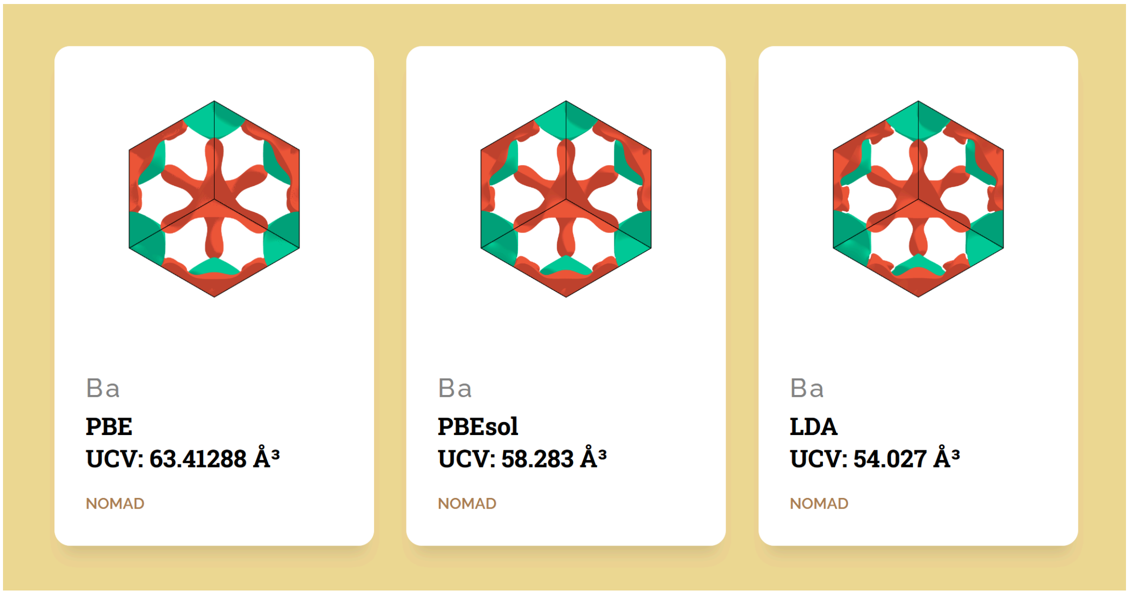


**Figure 5.2:** Snapshot of the periodic table of elements as presented on the website. Every element with a Fermi surface in the `SVG` format can be hovered over and is clickable.



Ba
**PBE**
**UCV: 63.41288 Å³**
NOMAD

Ba
**PBEsol**
**UCV: 58.283 Å³**
NOMAD

Ba
**LDA**
**UCV: 54.027 Å³**
NOMAD

**Figure 5.3:** Snapshot of the subpage showing the Fermi surface of Ba obtained using the PBE, PBEsol, and LDA functionals. The FS for each functional is clickable to open the interactive visualization. Additionally, the unit-cell volume (UCV) in Ångtrom and a direct link to the entry in the NOMAD database are provided.

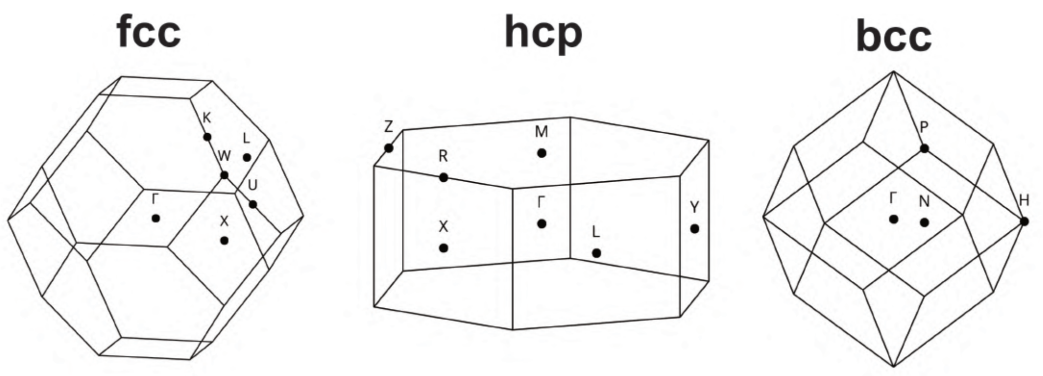


**Figure 5.4:** From left to right: First Brillouin zone and high-symmetry points of the fcc, hcp, and bcc crystal structure, respectively.

## 5.3 Analysis of the Fermi surfaces

In this section, we show examples of FSs generated using **`FSvisual`** for various elements, comparing results across different XC functionals. In order to discuss the topology and features of the FS, we show the first BZ and high-symmetry points of the fcc, hcp, and bcc crystal structure in Figure 5.4.

In our first example, Figure 5.5 presents the FS of Al, which has an fcc crystal structure. The black lines represent the BZ. The panels from left to right refer to calculations with a specific XC functional. Different colors denotes the different branches of the FS. The Fermi surface plots in each panel are very similar. Indeed, Aluminum is representative of the entire set of elemental crystals which show minimal qualitative differences among the FSs obtained with different XC functionals.

For some materials, however, stronger deviations are visible. One example for minor qualitative differences is W in the bcc structure, which is shown in Figure 5.6. From left to right in Figure 5.6, one can see that the small green pockets close to the H point become smaller and smaller in passing from LDA to PBEsol and, finally, to PBE, where they almost vanish.

Figure 5.7 displays the FS of Cd in the hcp crystal structure. Here, the differences between XC functionals are more pronounced: LDA and PBEsol display three branches, while PBE shows four. This fourth branch of PBE (colored in orange) together with an addition of the branch colored in purple, are close to the R and Z point (see Figure 5.4).

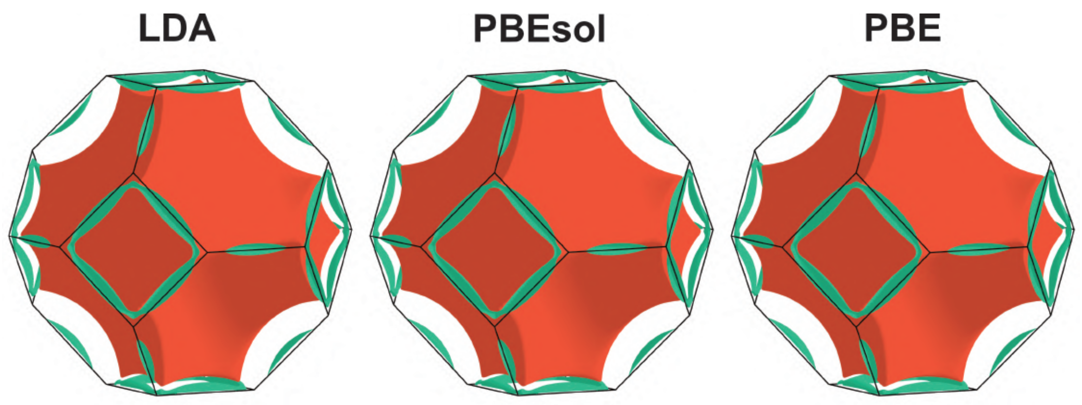


**Figure 5.5:** From left to right: Fermi surface of Al in the fcc structure calculated using LDA, PBEsol, and PBE XC functionals. The FS of Al consists of two branches, here colored red and green.

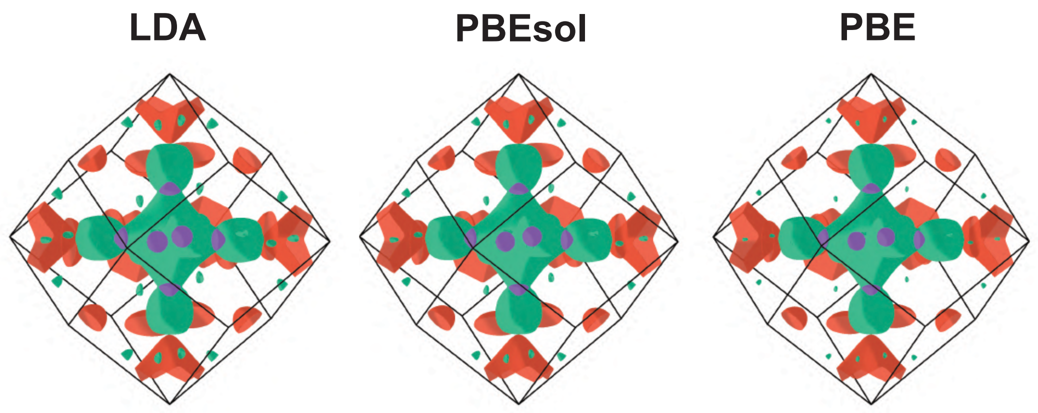


**Figure 5.6:** Same as Figure 5.5 for W in the bcc structure. Here, three branches are found, colored red, green, and purple.

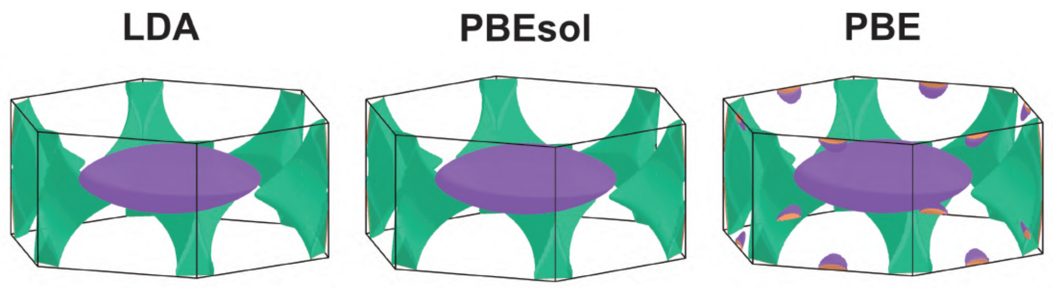


**Figure 5.7:** Same as Figure 5.5 for Cd in the hcp structure. For LDA and PBEsol we find three branches, colored in red, green, and purple. For PBE a fourth branch is displayed in orange.

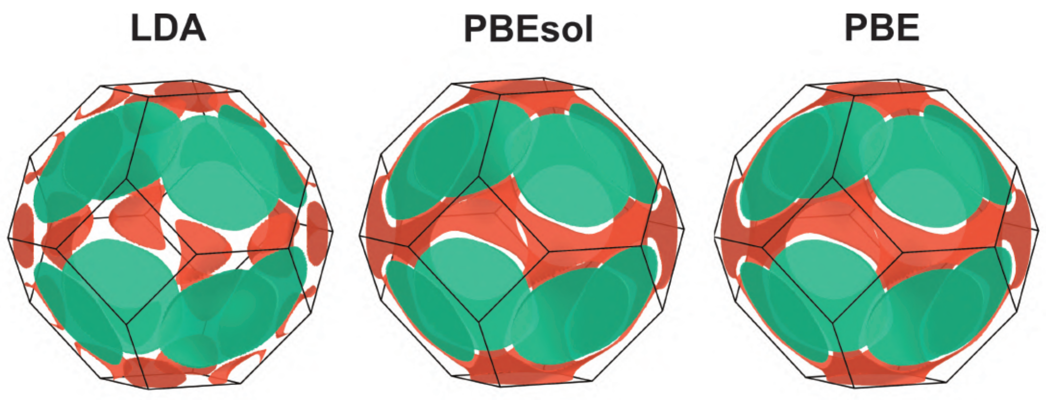


**Figure 5.8:** Same as Figure 5.5 for Ca in the fcc structure. Here, the FS has two branches colored red and green.

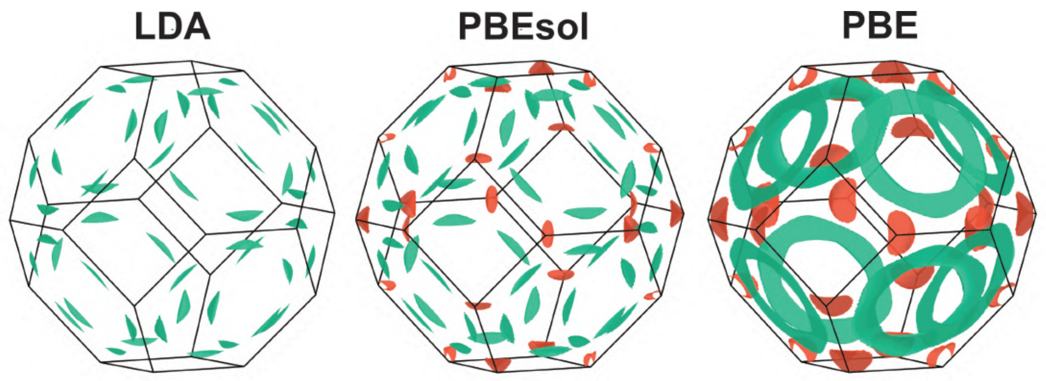


**Figure 5.9:** Same as Figure 5.5 for Sr in the fcc structure. Here, the FS has two branches colored red and green.

In Figure 5.8 the FS of Ca in the fccc structure is shown. This FS has two branches. When comparing the FS computed with PBE and PBEsol, no qualitative differences can be observed. However, for the LDA functional in contrast to with the other two cases, the red branch near the borders of the BZ is not a connected surface.

Figure 5.9 shows the FS of Sr, which has an fcc structure. For this metal, the largest deviation between the three functionals is visible. For PBE and PBEsol, the FS consists of two branches, which are colored in red and green. For LDA only the green branch is present. PBE and PBEsol share similarities, as both show green circle-like structures and red cones. However, for PBEsol, the green circles are not connected as for PBE, but are separated in multiple pieces with gaps in between. Each piece also has a reduced width relative to the boundaries of the green circles. Also, the red cones appear significantly smaller. For LDA, the green branch shows even larger gaps between the pieces and a further reduced size.

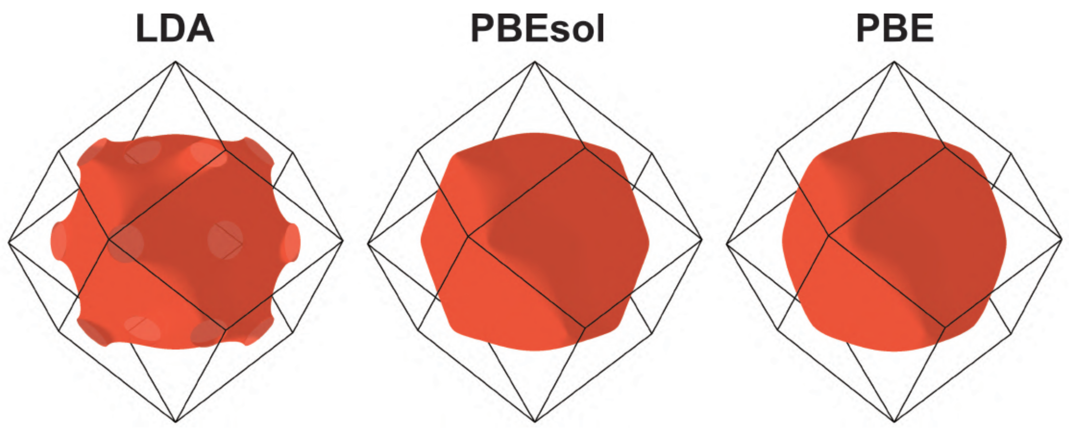


**Figure 5.10:** Same as Figure 5.5 for Cs in the bcc structure. Here, the FS has only one branch (depicted in red).

Figure 5.10 shows the FS of Cs in the bcc structure. It consists of a single branch, colored in red. The FS exhibits one of the most noticeable qualitative differences across PBE, PBEsol, and LDA. While from PBEsol to PBE the corners of the FS become slightly more rounded, for the LDA functional the Fermi surface is no longer closed, but has holes at the border of the BZ.

To understand why the FS of a material can differ when different XC functionals are used, it is important to note that the FSs were computed at the equilibrium UC volume obtained for the respective XC functional. These volumes are shown in Figure 5.11, where triangles (circles, squares) represent the PBE (PBEsol, LDA) equilibrium UC volume of each material. It can be seen that for many materials with small UC volumes ($\sim 100$ Å$^3$), the differences are small. However, the differences increase significantly for larger UC volumes above 200 Å$^3$. In all cases, using LDA results in the smallest volume, while PBE results in the largest. Figure 5.11 suggests a correlation between larger UC volumes and a visible difference in the FSs across the XC functionals. The elements that show qualitative differences in their FSs are depicted in red.

Additionally, metals with larger average UC volumes (average over the volumes of every XC functional for a metal) tend to show greater deviations in UC volume across the XC functionals. To ensure that the observed deviation is not simply proportional to the average UC volume of each metal, the deviations were normalized by the corresponding average values. Figure 5.12 shows the normalized volume deviation for each investigated metal across the XC functionals. For each metal, the average UC volume, as well as the deviation percentage of every XC functional was calculated. In Figure 5.12 elements along the horizontal axis are ordered according to increasing values of the average UC volume. The data points in this figure represent the average of all the UC volume deviation percentages for each metal.

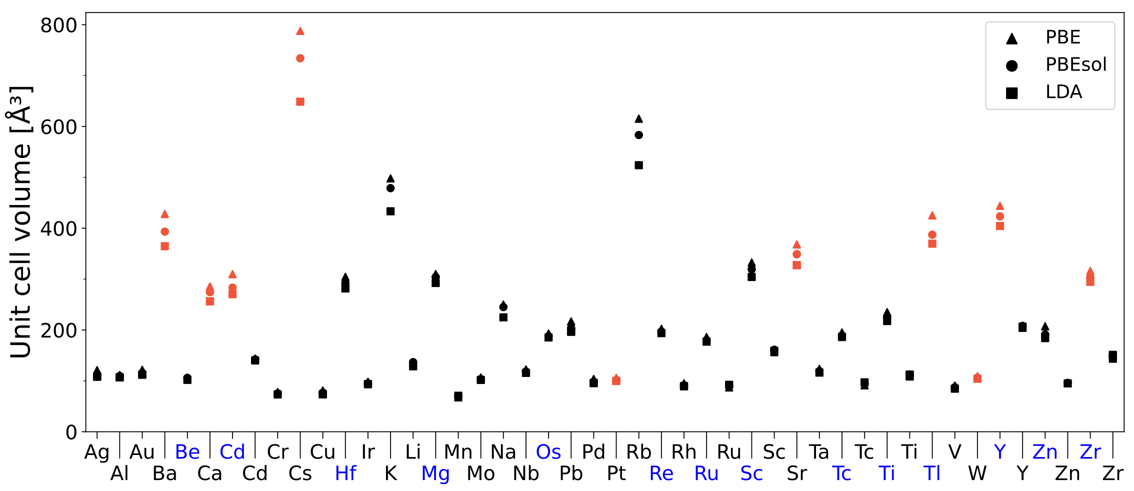


**Figure 5.11:** Equilibrium unit-cell volume of all investigated metals using different XC functional. Elements that show qualitative differences of their FS in dependence on the XC functionals are depicted in red. The blue marked labels indicate that the corresponding metals have an hcp crystal structure.

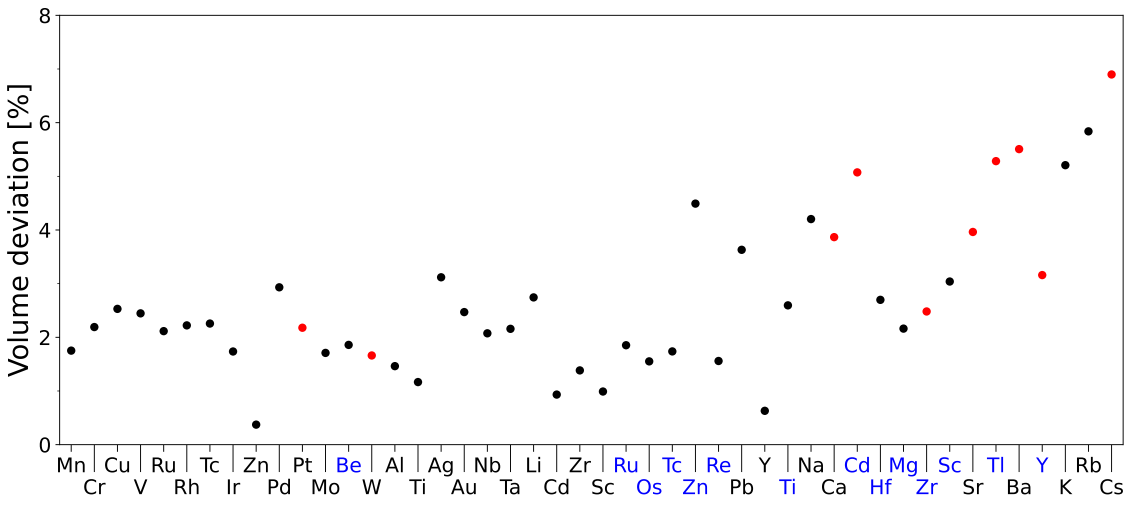


**Figure 5.12:** Relative unit-cell volume deviation across the XC functionals in percent of for each metal. Elements that show qualitative differences on their FSs are colored in red. The blue marked labels indicate, that the corresponding metals have an hcp crystal structure. See text for further details.

The average over every red-marked data point in Figure 5.12 is 4.0%. If we compare this value to the average of the black-marked data points, which is 2.3%, we see a higher deviation of 1.7% for the red-marked data points. Consequently, a relative increase in the relative UC volume deviation can be observed for metals that show qualitative differences in their FSs.

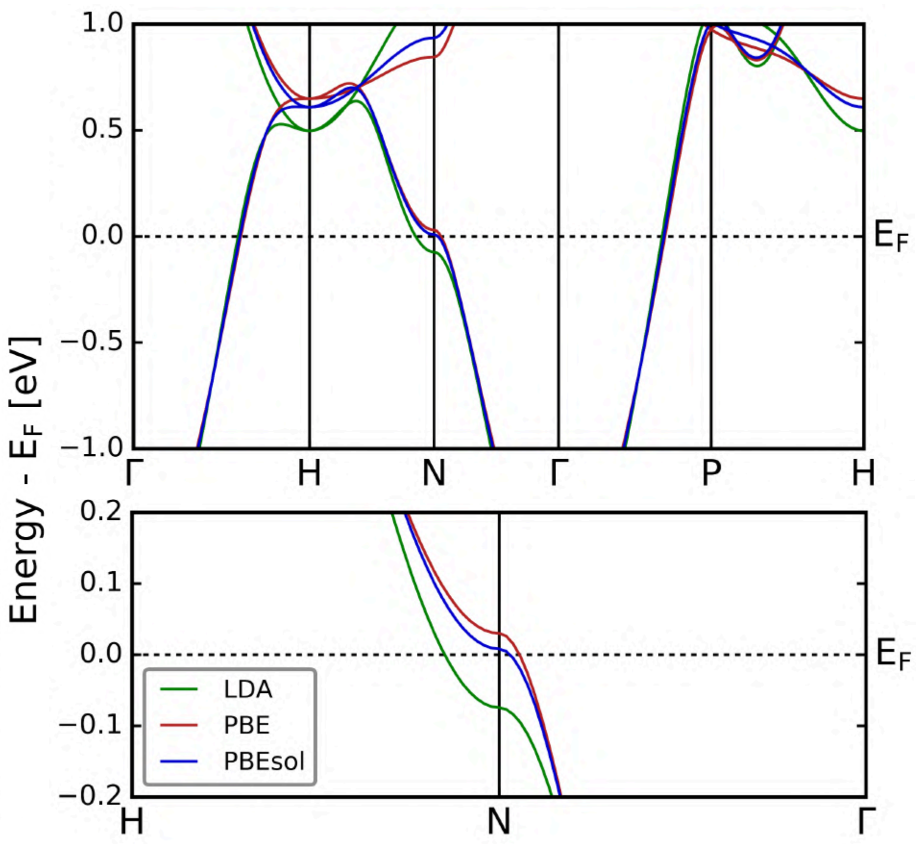


**Figure 5.13:** Electronic band structure (BS) of bcc Cs calculated using the LDA (green), PBE (red), and PBEsol (blue) XC functionals. Top panel: BS along the standard path for bcc crystal structures Γ-H-N-Γ-P-H. Bottom panel: Details of the BS around the Fermi energy along the H-N-Γ path. Notice that while the N-Γ path is fully contained inside the first BZ, the H-N-Γ path lies entirely on a BZ boundary face.

To further investigate these correlations, we focus on bcc Cs, which exhibits the largest average UC volume as well as the largest UC volume deviation. As one can see in Figure 5.10, for PBE and PBEsol the FS fits within the BZ. For LDA, the UC volume is smaller and the bands do not fit within the BZ. Thus, they are folded back. To illustrate that, we show the band structure (BS) of Cs for LDA, PBE, and PBEsol in Figure 5.13. Since a "hole" in the FS of the LDA calculation for Cs (see Figure 5.10) appears close to the N point (see Figure 5.4), we display in the bottom panel of Figure 5.13 the details of the BS around the Fermi energy along a symmetry path close to the N point. Along the H-N direction, which is a line on the border of the BZ, the LDA BS intersects the Fermi energy. This result indicates that in a repeated-zone representation the FS extends itself beyond the boundaries of the first BZ. When using a reduced-zone representation (see Section 2.2) the exceeding part of of the FS is folded back by a specific reciprocal lattice vector. The appearance of characteristic "holes" in the FS at the boundary of the BZ is an effect of this folding process.

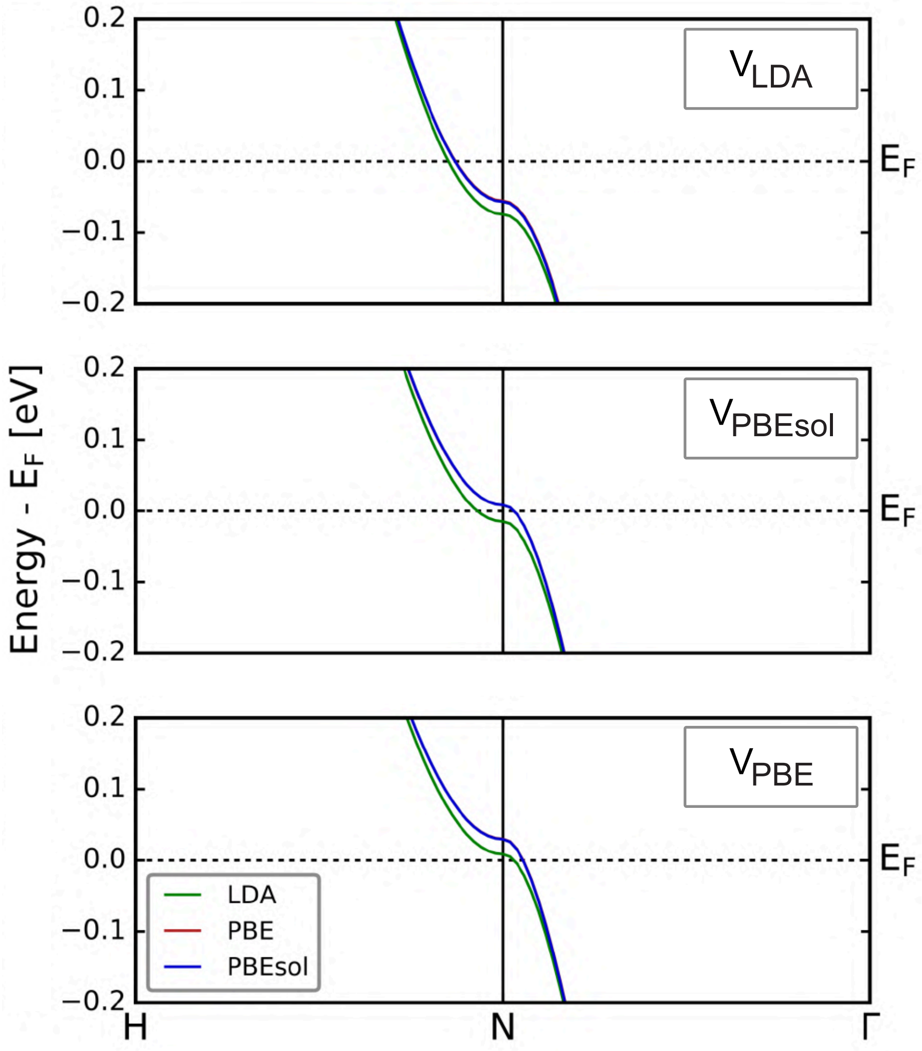


**Figure 5.14:** Details around the Fermi energy of the band structure of bcc Cs calculated with LDA (green lines), PBE (red), and PBEsol (blue) along the H-N-Γ path in the BZ. The panel from top to bottom show the results obtained from calculation performed at the equilibrium volume of LDA, PBEsol, and PBE, respectively.

To confirm if these effects can be solely attributed to the volume, or indicate more fundamental differences between the two types of functionals (LDA and GGA), we compute the BS with all three functionals at the equilibrium volumes obtained with LDA ($V_{\text{LDA}} = 96.11$ Å$^3$), PBE ($V_{\text{PBE}} = 116.74$ Å$^3$), and PBEsol ($V_{\text{PBEsol}} = 108.77$ Å$^3$). In particular, we focus on energy values close to the Fermi energy along the H-N-Γ path in the BZ. The results are shown in Figure 5.14. In all cases, the BS of both GGA functionals, PBE and PBEsol, are visually indistinguishable. Furthermore, it can be seen from the top panel of Figure 5.14 that the "hole" structure of the FS appears for all three XC functionals for calculations performed at $V_{\text{LDA}}$. Finally, the overall trend when passing from GGA to LDA XC functionals at the corresponding equilibrium volumes is a downwards rigid shift of the energy bands relative to the Fermi energy.

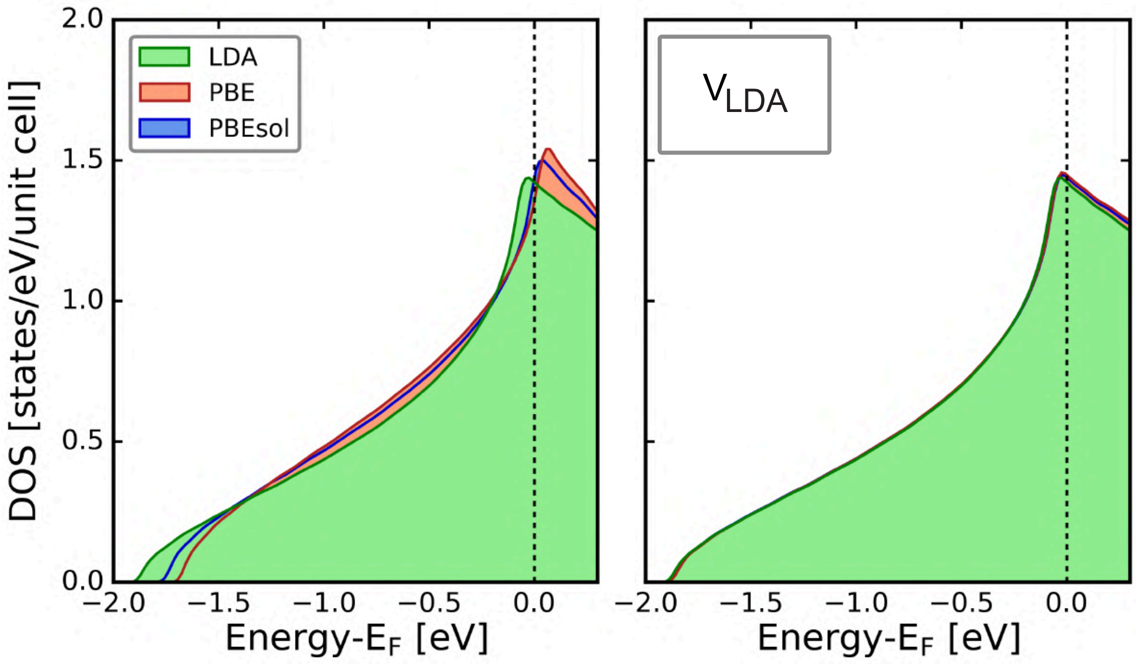


**Figure 5.15:** Left panel: DOS of bcc Cs calculated with LDA, PBE, and PBEsol at the corresponding equilibrium volumes. Right panel: Same as the left panel calculated at the LDA equilibrium volume.

As described in Section 2.3, a change in UC volume also has an impact on the DOS. According to Eq. (2.10), a different DOS yields a different Fermi energy, which also causes changes in the visual appearance of the FS. Therefore, it is also important to compare the DOS across the XC functionals in our test case of bcc Cs. The explicit comparison of the DOS is displayed in Figure 5.15 in the energy region between -2.00 and 0.250 eV, with the Fermi energy $E_F$ set, as indicated by the dotted line, at 0 eV. In the left panel of Figure 5.15, the DOSs corresponding to the three XC functionals exhibit significant differences at energy near the Fermi level. In particular, the maximum of the DOS in the displayed energy range changes its position for the three XC functional from slightly below $E_F$ (LDA) to slightly above it (PBE and PBEsol). If the DOS shape changes close to the Fermi energy, then, the intersection points of the BS with the Fermi level may also change their position, causing visual changes in the FS. The right panel of Figure 5.15 shows the DOS with the three XC functionals calculated with the same UC volume $V_{LDA}$. In this case, the DOS for LDA, PBE, and PBEsol are nearly identical. This indicates that for bcc Cs, the differences in DOS up to the Fermi energy, are predominantly determined by the UC volume. This also explains why the BSs of the different XC functionals, at a fixed volume, appear similar to each other around the Fermi energy.

Following the indication given by the example above, we can argue that the most noticeable qualitative differences for the FS in dependence on the XC functionals are most likely caused by large deviation in UC volume. However, the opposite statement is not always true: The FS of the metals K and Rb (accessible on the website [11]) show almost no visible differences across LDA, PBE, and PBEsol, although their overall UC volume (see Figure 5.11), as well as their UC volume deviation (see Figure 5.12) are very high. This is not in contrast with the previous results, since both these metals only show intersections of the Fermi level with their BSs inside the BZ. Consequently, despite the differences in the BS of the XC functionals is expected to be significant, if the differences from one functional to the other do not cause a BS intersection with the Fermi level outside the BZ (as for Cs), the qualitative differences can be small.

# 6 Conclusions and Outlook

The main task of this thesis was the development of the fully interactive and well-integrable software package **`FSvisual`** for the visualization of the Fermi surface (FS) of metallic systems. Starting from the available raw data consisting of electronic band-structure files in the **`bxsf`** format, **`FSvisual`** implements a three-step procedure for getting **`HTML`** plots and **`svg`** images of the FS. The input data are downloaded from the NOMAD database. The data consist of the results of *ab initio* electronic-structure calculations performed by Qiang Fu [7] using the **`exciting`** code. Included in the data are files for 37 elemental metals, each calculated with three different exchange-correlation (XC) functionals (see Section 2.3) and some with multiple atomic structures.

Concerning the implementation of **`FSvisual`**, the initial step is to parse the downloaded data. Then, the band-energy values are associated to their corresponding $\boldsymbol{k}$-points in the sampling grid (see Section 4.2). Finally, the Fermi surface is constructed by using the Marching Cubes interpolation algorithm (see Section 2.4) and polished to be entirely contained in the first Brillouin zone of the considered crystal by means of the slicing algorithm (see Appendix A). As a result, the Fermi surfaces are visualized with the help of the **`Plotly`** library.

The visualized FSs were compared qualitatively (see Section 5.3) for each available metal, highlighting the differences resulting from the choice of XC functionals. This investigation pointed out a correlation between qualitative changes in the FS of an element across the XC functionals and its average unit-cell (UC) volume. Furthermore, it was shown, that an increasing average UC volume is correlated with an increased UC volume deviation across the XC functionals.

In our major example, the band structure of Cs in the bcc structure was investigated for all three considered XC functionals. It was then shown, that by fixing the UC volume and using various XC functionals, the differences in the BS and density of states near the Fermi level decreases significantly. All the available FS plots created by **`FSvisual`** are made accessible through a website [11].

As for future work, a deeper investigation of the differences of the FSs across the three used XC functionals is possible. Additionally, we plan the integration of **`FSvisual`** into NOMAD services. We also aim to make the implementation more user-friendly and improve the usability of the code. This development is planned and has already started to take place.

# Acknowledgement

I would like to thank Dr. Pasquale Pavone and Prof. Claudia Draxl for their invaluable guidance and insightful discussions throughout this work. Also, I would like to thank Martin Kuban for his mentorship throughout this thesis. His expertise and support have been essential in helping me navigate the challenges of this work. Finally, I would like to thank my colleagues, friends, and family, as well as my girlfriend, for their support during this time.

# Appendix A: Slicing Algorithm

The FSs built with the **`bxsf`** files have parts that extend beyond the borders of the BZ. The purpose of the slicing algorithm is to remove these surface parts. To archive this, the vertices and facets of the FS are converted into a **`Trimesh`** object. This **`Trimesh`** object provides a **`slice_plane`** function, which is capable of slicing a surface in three dimensions into two peaces along a plane. The function takes the planes origin and the normal vector of that slicing plane as input. Since the border of the FS is the BZ, the algorithm iterates over each facet of the BZ and passes the facet's origin and normal vector to the **`slice_plane`** function. The origin of each facet is defined by one of its vertices, which is consistently chosen to be the first vertex of the facet. In order to retrieve the facets normal, we implemented a function called **`face_center_BZ`**. Its purpose is to return the center point of a facet of the BZ. The input of the function are the facets and vertices of the BZ (see Section 4.3.1).

At first, all the facets of the BZ are triangulated, meaning that the surface is partitioned into triangles by the function **`triangulate_faces`** 6, which also takes the vertices and facets of the BZ as input. To achieve this, the function classifies each facet into three categories based on the number of vertices. The simplest case occurs when the facet is already a triangle and requires no further processing. In the second case, the facet is a square, which can always be divided into two triangles. Here, the order in which the vertices of the first triangle are connected is arbitrary. The second triangle consists of the three remaining vertices. For facets with more than four vertices, the third case applies, using a method called *fan-triangulation*. This method selects an arbitrary starting vertex and iterates through every subsequent one. From one iteration to the next, the third vertex of each triangle becomes the second vertex in the next triangle. Figure A.1 shows a pentagonal shape, which is triangulated using the fan-triangulation method. Each created triangle is presented in a different color. Technically, the fan triangulation also works for case one and two, but it is less efficient and, more importantly for **`FSvisual`**, less readable. Once the list of triangles is generated, the **`face_center_BZ`** [6] can proceed.

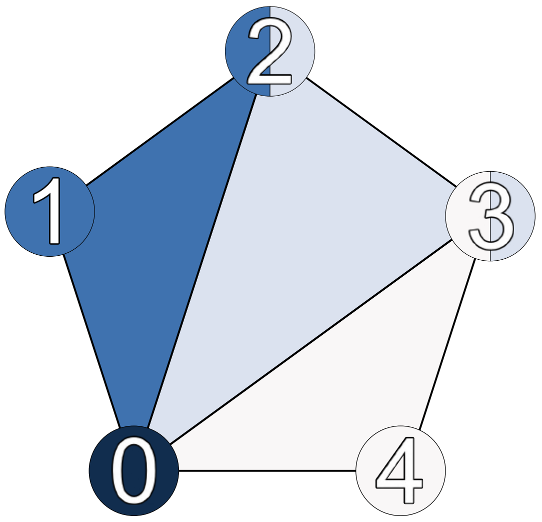


**Figure A.1:** Pentagonal shape that is triangulated using fan-triangulation. Each vertex is indexed starting at the fist vertex with 0. Every triangle is colored differently.

The **`face_center_BZ`** function then iterates through each facet of the BZ. For each triangle inside every of those facets, both its area and center are calculated. Using this information, the function is capable of calculating the centroid of each facet:

$$C_x = \frac{\sum_i C_{x,i} \, A_i}{\sum_i A_i}, \quad C_y = \frac{\sum_i C_{y,i} \, A_i}{\sum_i A_i}, \quad C_z = \frac{\sum_i C_{z,i} \, A_i}{\sum_i A_i}, \tag{.1}$$

where $(C_x, C_y, C_z)$ represent the Cartesian coordinates of the centroid of the entire facet, $(C_{x,i}, C_{y,i}, C_{z,i})$ $(A_i)$ denote the centroid (area) of the $i$-th triangle. With that, the computed facet centroids are ready to be returned to the slicing loop. The **`slicing_plane`** function then slices the FS along each facet, retaining only the part within the BZ.

```python
def triangulate_faces(facets):
        triangles = []
    for facet in facets:
        if len(facet) == 3:
            triangles.append(facet)
        elif len(facet) == 4:
            triangles.append([facet[0], facet[1], facet[2]])
            triangles.append([facet[0], facet[2], facet[3]])
        else:
            # For facets with more than 4 vertices,
            # use a fan triangulation
            for i in range(1, len(facet) - 1):
                triangles.append([facet[0], facet[i],
                                 facet[i + 1]])
    return triangles
```

```python
def face_center_BZ(brillouin_zone_facets):
    triangle_list = triangulate_faces(brillouin_zone_facets)
    face_centers = []
        j = 0
        for facet in brillouin_zone_facets: calculates the center of it
            triangle_areas = []
            triangle_centers = []
            for triangle in triangle_list[j:j + len(facet) - 2]:
                # triangle area:
                if triangle_area(triangle) != 0:
                    triangle_areas.append(triangle_area(triangle))
                    triangle_centers.append(triangle_center(triangle))
            j += len(facet) - 2
            x_coord = np.array([point[0] for point in triangle_centers])
            y_coord = np.array([point[1] for point in triangle_centers])
            z_coord = np.array([point[2] for point in triangle_centers])
            # calculation of the facet center (geometrischer Schwerpunkt)
            xS = np.sum(x_coord * np.array(triangle_areas)) / np.sum(
                triangle_areas)
            yS = np.sum(y_coord * np.array(triangle_areas)) / np.sum(
                triangle_areas)
            zS = np.sum(z_coord * np.array(triangle_areas)) / np.sum(
                triangle_areas)
            face_centers.append([xS, yS, zS])
        return face_centers
```

# Bibliography

[1] Maurizio Mattesini and Martin Magnuson. Electronic correlation effects in the cr2gec mn+1axn phase. *Journal of Physics: Condensed Matter*, 25, 2013.

[2] Vei Wang, Nana Xu, Jin Cheng Liu, Gang Tang, and Wenjing Geng. Vaspkit: A user-friendly interface facilitating high-throughput computing and analysis using vasp code. *Comput. Phys. Commun.*, 267:108033, 2019.

[3] Masnun Naher and Saleh H Naqib. Structural, elastic, electronic, bonding, and optical properties of topological casn3 semimetal. *Journal of Alloys and Compounds*, 2019.

[4] Miroslaw Werwi'nski and Wojciech Marciniak. Ab initio study of magnetocrystalline anisotropy, magnetostriction, and fermi surface of l10 feni (tetrataenite). *Journal of Physics D: Applied Physics*, 50, 2017.

[5] J. Stutz. Fsvisual. `https://git.physik.hu-berlin.de/stutzjan/bachelorarbeit`.

[6] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21:163–169, 1987.

[7] Q. Fu. DFT calculations for metals using LDA, PBE, and PBEsol as XC functionals. The data provided by NOMAD was accessed via the following query: `{"query":{"and":[{"results.method.simulation.program_name:any":["exciting"],"authors.name:any":["Qiang Fu"]},{"quantities:all":["results.method.simulation.program_name"]}]}}`. Any of the returned NOMAD entries to this query, that contains a band structure file, is used in this work.

[8] Exciting. exciting a full-potential all-electron package implementing linearized augmented planewave methods. `https://www.exciting-code.org/` [Accessed: (07.01.2025)].

[9] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864–B871, Nov 1964.

[10] M. Scheidgen, L. Himanen, A. Ladines, D. Sikter, M. Nakhaee, Á. Fekete, T. Chang, A. Golparvar, S. Brockhauser J. Márquez, S. Brückner, L. Ghiringhelli, F. Dietrich, D. Lehmberg, T. Denell, A. Albino 1, H. Näsström, S. Shabih, F. Dobener, M. Kühbach, R. Mozumder, J. Rudzinski, N. Daelman, J. Pizarro, M. Kuban, C. Salazar, P. Ondračka, H.-J. Bungartz, and C. Draxl. NOMAD: A distributed web-based platform for managing materials science research data. `https://nomad-lab.eu` [Accessed: (05.01.2025)].

[11] J. Stutz. Periodic table of elements with interactive fermi surface plots. `https://exciting-code.org/fermi_surfaces`.

[12] Rudolf Gross and Achim Marx. *Festkörperphysik*. DE GRUYTER, 2018.

[13] Neil W. Ashcroft N. David Mermin. *Solid State Physics*. Harcourt College Publishers, 1976.

[14] David J. Griffiths and Darrell F. Schroeter. *Introduction to Quantum Mechanics*. Cambridge University Press, 2018.

[15] Wahyu Setyawan and Stefano Curtarolo. High-throughput electronic band structure calculations: Challenges and tools. *Computational Materials Science*, 49(2):299–312, 2010.

[16] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133, Nov 1965.

[17] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133–A1138, Nov 1965.

[18] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. *Physical review letters*, 77 18:3865–3868, 1996.

[19] Lucian A. Constantin, John P. Perdew, and J. M. Pitarke. Exchange-correlation hole of a generalized gradient approximation for solids and surfaces. *Phys. Rev. B*, 79:075126, Feb 2009.

[20] Philipp Haas, Fabien Tran, Peter Blaha, Luana S. Pedroza, Antônio J. R. da Silva, Mariana M. Odashima, and Klaus Capelle. Systematic investigation of a family of gradient-dependent functionals for solids. *Physical Review B*, 81:125136, 2010.

[21] A. Kokalj and M. Causà. XCrySDen a crystalline- and molecular-structure visualisation program. `http://www.xcrysden.org/XCrySDen.html` [Accessed: (05.01.2025)].

[22] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[23] Dawson-Haggerty et al. trimesh.

[24] Plotly Technologies Inc. Collaborative data science, 2015.

# Statement on the use of AI tools

During the writing of this thesis, generative AI tools like **`ChatGPT`** were employed exclusively for linguistic refinement.

## Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen, Bilder sowie die Nutzung von Künstlicher Intelligenz für die Erstellung von Texten und Abbildungen, sind als solche kenntlich gemacht. Mir ist bekannt,dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den 15.04.2025: [illegible]