Chapter 3
Calculating Topological Invariants with Z2Pack

Dominik Gresch and Alexey Soluyanov

Abstract The topological phase of non-interacting electronic bandstructure can be classified by calculating integer invariants. In this chapter, we introduce the Chern invariant that classifies 2D materials in the absence of symmetry. We then show that this invariant can be used as the building block for the classification of topological insulators, semimetals, and symmetry-protected topological phases. We show how this classification is performed in practice by introducing `Z2Pack`, a tool which allows calculating topological invariants from $\mathbf{k} \cdot \mathbf{p}$ and tight-binding models, as well as first-principles calculations.

3.1 The Chern Number

In this section, we give a coarse introduction to topological invariants in the context of classifying crystalline solids. In the interest of brevity, we will skip many of the mathematical details, instead focusing on conveying an intuitive understanding as required to follow the rest of this chapter. For a more thorough description of the topics covered here, the reader is referred to [1, 2].

3.1.1 Topology in Non-interacting Materials

In this first section, we will introduce the notion of topological properties in the context of non-interacting materials. From their basic definition, we will see that topological phases must exhibit some interesting physical phenomena.

D. Gresch (✉)
ETH Zurich, Institut für Theoretische Physik, Wolfgang-Pauli-Str. 27,
8093 Zürich, Switzerland
e-mail: greschd@phys.ethz.ch

A. Soluyanov
Physik-Institut, Universität Zürich, Winterthurerstrasse 190,
8057 Zurich, Switzerland
e-mail: soluyanov@phys.ethz.ch

D. Bercioux et al. (eds.), *Topological Matter*, Springer Series in Solid-State Sciences 190, https://doi.org/10.1007/978-3-319-76388-0_3

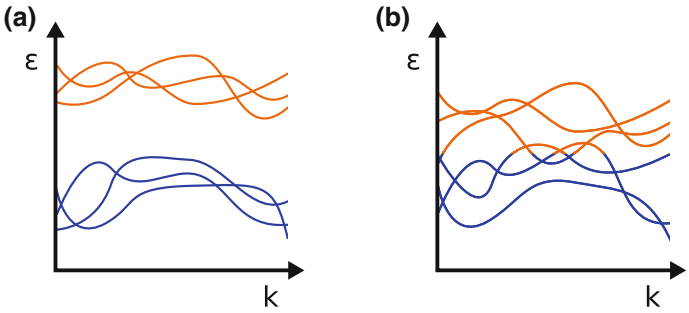


Fig. 3.1 **a** Bandstructure of an insulating material. Occupied (blue) and unoccupied (orange) states are separated by an energy gap for all **k**. **b** Bandstructure of a conducting material. Occupied and unoccupied states touch, and some bands are partially occupied

3.1.1.1 A Short Reminder on Band Theory

We will start with a short reminder about band theory. A more thorough description of the subject can be found in any solid-state physics textbook.

In the non-interacting limit, electronic states in crystalline materials can be described by a single-particle Hamiltonian $H(\mathbf{k})$, which is a smooth function of the crystal wave-vector $\mathbf{k}$. The possible electronic states are given by the solutions of the time-independent Schrödinger equation

$$H(\mathbf{k}) \left|\psi_{n,\mathbf{k}}\right\rangle = \epsilon_{n,\mathbf{k}} \left|\psi_{n,\mathbf{k}}\right\rangle . \tag{3.1}$$

These so-called *Bloch states* $\left|\psi_{n,\mathbf{k}}\right\rangle$ are a superposition of plane waves with wave-vector $\mathbf{k}$. As such, they can be written as

$$\left|\psi_{n,\mathbf{k}}\right\rangle = e^{i\mathbf{k}\cdot\mathbf{r}} \left|u_{n,\mathbf{k}}\right\rangle , \tag{3.2}$$

where $\left|u_{n,\mathbf{k}}\right\rangle$ is cell-periodic. This property is known as the Bloch theorem [3]. The energy eigenvalues $\epsilon_{n,\mathbf{k}}$ are called *energy bands*, with n being their band index.

The bulk properties of materials are determined largely by their bandstructure. For example, a material is insulating if there is an energy gap between the eigenstates which are occupied by electrons, and those that are empty, as shown in Fig. 3.1a. Conversely, a material is conducting if there is no such energy gap, such as when an energy band is only occupied for certain values of $\mathbf{k}$.

It turns out, however, that characterizing materials by their bandstructure does not fully capture their physical properties. Instead, taking into account the shape of the Bloch states $\left|\psi_{n,\mathbf{k}}\right\rangle$ leads to a topological classification of materials.

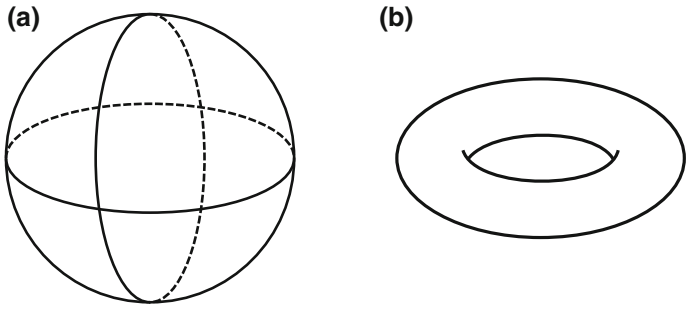


Fig. 3.2 Examples of closed orientable surfaces: **a** A sphere has no holes, and **b** A torus has one hole

3.1.1.2 Topological Properties

To motivate the concept of topological classification, we first show an example from its mathematical origins in geometry: Closed, orientable two-dimensional surfaces can be classified by their number of holes, called *genus*. A sphere, for example, has no holes, while a torus has exactly one (see Fig. 3.2). This property is conserved under smooth deformations of the surface. The only way to add or remove a hole is by tearing and gluing the surface. The genus is an example for a topological invariant – a quantized property that cannot be changed without changing the topological phase. For this reason, topological invariants are commonly used to *identify* topological phases.

In order to define topological phases for materials, we need a geometric object on which the topological properties can be defined. For this purpose, we pick a set of bands B. A very common choice for B is to pick the occupied subspace.[1] The set of states $\{|u_{n,\mathbf{k}}\rangle\}_{n\in B}$ span a vector space $V_{\mathbf{k}}$ (over $\mathbb{C}$) for each $\mathbf{k}$. If $V_{\mathbf{k}}$ is a smooth function of $\mathbf{k}$ and the space where $\mathbf{k}$ itself is defined is a manifold, this defines a so-called fiber bundle.

A simple geometrical example of a fiber bundle is given by a one-dimensional vector space defined on a circle. If the vector space is orthogonal to the plane described by the circle, the resulting object is a cylinder, as shown in Fig. 3.3a. If, however, the basis vector rotates by π as it goes around the circle, the resulting object is a Möbius strip. These two objects cannot be smoothly transformed into each other, making them topologically distinct.

[1] This is not always possible, for example in the case of semimetals where the occupation number changes with $\mathbf{k}$. In these cases, one often picks the N lowest energy bands instead.

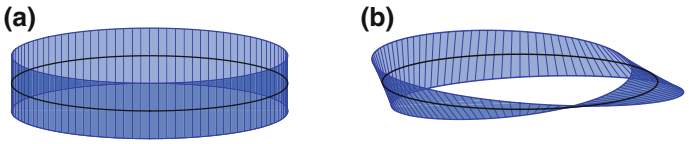


Fig. 3.3 **a** A cylinder, spanned by a vector which does not rotate as it goes around a circle. **b** A Möbius strip, spanned by a vector which rotates by π as it goes around a circle

3.1.1.3 Bulk-Edge Correspondence

In the previous section, the fact that the vector space $V_{\mathbf{k}}$ needs to be a smooth function of $\mathbf{k}$ was mentioned. This has a profound impact on the physical properties of topological states, as we shall now see.

Even though the Hamiltonian $H(\mathbf{k})$ is a smooth function of $\mathbf{k}$, the same is not necessarily true for $V_{\mathbf{k}}$. Consider the following one-dimensional example:

$$H(k) = -\cos(k)\,|a\rangle\langle a| + \cos(k)\,|b\rangle\langle b|\,, \tag{3.3}$$

where $|a\rangle$ and $|b\rangle$ are two arbitrary orthogonal states. For $k = 0$, the energy eigenvalues of $|a\rangle$ and $|b\rangle$ are -1 and 1, respectively. Consequently, $|a\rangle$ has band index 1, while $|b\rangle$ has index 2. As k changes, the energy eigenvalues shift until they are equal at $k = \pi/2$. At this point, the vector space $V_{\mathbf{k}} = \text{span}\left(\{|u_{n,\mathbf{k}}\rangle\}_{n\in\{1\}}\right)$ switches discretely from being spanned by $|a\rangle$ to being spanned by $|b\rangle$. As a result, this space does not meet the criteria for topological categorization.

The smoothness of the vector space $V_{\mathbf{k}}$ can be broken if the order of energy eigenvalues between the states which are in the set B and those which are not changes. This can easily be avoided if we restrict our possible choice of bands B, such that they are always separated from the other bands by a direct energy gap. In other words, topological properties are defined for *isolated* sets of bands, which form smooth fiber bundles.

Another way to frame this is by looking at the possible transformations that can be done to a material without changing its topological properties. In addition to requiring that these transformations smoothly change the Hamiltonian, we impose that the band gap remains open. This definition leads to a remarkable physical property of topological phases: At the boundaries of topologically non-trivial insulating materials, stable conducting edge states must form. In going from the bulk of the topological material to vacuum, the system undergoes a smooth transition from a non-trivial to a trivial (vacuum) state. To allow for this, the aforementioned condition that the bands are separated in energy must be broken. This effect is known as the bulk-boundary correspondence, and variations of this effect govern the interesting transport phenomena to be found in many topological materials [4–6].

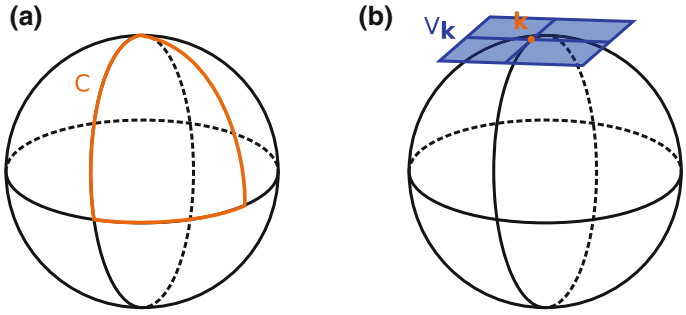
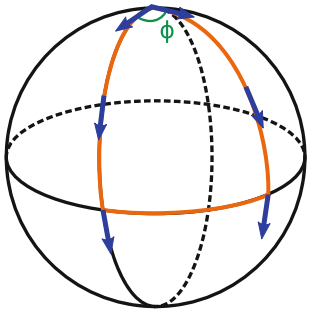


Fig. 3.4 **a** A closed path C on the surface of a sphere. **b** The tangential vector space $V_\mathbf{k}$ for a given point $\mathbf{k}$ on a sphere

Fig. 3.5 Parallel transport of a vector on a closed path on a sphere rotates the vector by an angle ϕ



3.1.2 Defining the Chern Number

In the previous section, we have seen how topological properties in crystalline materials are defined on a conceptual level. Now, we will show an example for a topological invariant, which can be used to classify many topological phases in matter.

3.1.2.1 The Berry Phase and Chern Invariant

The basis for defining a topological invariant for electronic bands is the notion of a *geometric phase*. To illustrate this phase, imagine a closed loop C on a manifold. As an example, we choose a closed loop on a sphere, as shown in Fig. 3.4a. Adding the plane tangential to the sphere at each point gives us a fiber bundle (see Fig. 3.4b).

Now we choose a vector in the tangential space and move it along C in such a way that it remains locally parallel to itself, as shown in Fig. 3.5. This process is called *parallel transport*. We observe that the vector is rotated by some angle ϕ as it traverses the path C. Since this angle depends only on the geometry of the fiber bundle, it is called a geometric phase.

For electronic bands, such a phase, known as Berry's phase,[2] can be written as [7]

$$\gamma_C = i \oint_C \sum_{n \in B} \langle u_{n,\mathbf{k}} | \nabla_{\mathbf{k}} | u_{n,\mathbf{k}} \rangle . \mathrm{d}\mathbf{k}, \tag{3.4}$$

where C is a closed loop in reciprocal space. Unlike the example above, the Berry's phase represents a rotation in the complex phase of a vector, not its real-space direction.[3] It is Gauge invariant up to multiples of 2π [7]. By defining the Berry potential

$$\mathscr{A}(\mathbf{k}) = i \sum_{n \in B} \langle u_{n,\mathbf{k}} | \nabla_{\mathbf{k}} | u_{n,\mathbf{k}} \rangle , \tag{3.5}$$

the Berry phase can be rewritten as

$$\gamma_C = \oint_C \mathscr{A}(\mathbf{k}) . \mathrm{d}\mathbf{k}. \tag{3.6}$$

Note that unlike the Berry phase, the Berry potential is not a Gauge-invariant quantity. If the Berry potential is a smooth function of $\mathbf{k}$ (an important prerequisite, as we shall see soon), we can use Stokes' theorem to rewrite the Berry phase as a surface integral

$$\gamma_C = \int_S \nabla_{\mathbf{k}} \wedge \mathscr{A}(\mathbf{k}) . \mathrm{d}\mathbf{k}, \tag{3.7}$$

where $C = \partial S$. Introducing the *Berry connection*

$$\mathscr{F} = \nabla_{\mathbf{k}} \wedge \mathscr{A}(\mathbf{k}), \tag{3.8}$$

which is again Gauge invariant, we can write this as

$$\gamma_C = \int_S \mathscr{F}(\mathbf{k}) . \mathrm{d}\mathbf{S}. \tag{3.9}$$

For a closed, orientable two-dimensional surface S in reciprocal space, we can now define the Chern invariant as [8, 9]

$$C = \frac{1}{2\pi} \int_S \mathscr{F}(\mathbf{k}) . \mathrm{d}\mathbf{S}. \tag{3.10}$$

Since the edge of a closed surface is a trivial path, (3.6) seems to suggest that the Chern number is always zero. However, we must now remember that (3.7) and (3.9)

[2] For simplicity, we consider the *total* Berry phase of all bands. The Berry phase can also be defined for a single band, in which case the sum over bands is dropped.

[3] To see this, try calculating the Berry phase for $|u_k\rangle = e^{ik/2} \begin{pmatrix} \cos(k) \\ \sin(k) \end{pmatrix}$, for $k \in [0, 2\pi]$.

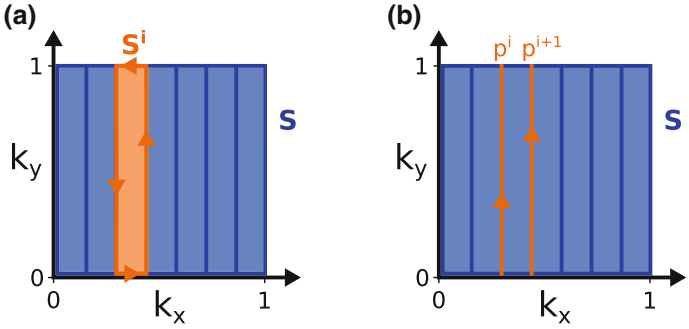


Fig. 3.6 **a** The surface S is divided into segments S^i. For each segment, the flux of Berry connection can be calculated from the Berry phase around its boundary. **b** The top and bottom paths of each boundary cancel, leaving paths p^i which cross the Brillouin zone at a constant k_x

are valid only if the Berry potential $\mathscr{A}(\mathbf{k})$ is smooth. Previously, we discussed that $V_\mathbf{k}$ spanned by $|u_{n,\mathbf{k}}\rangle$ must be a smooth function of $\mathbf{k}$ if we wish to define a topological classification. However, there can still be a winding in the phase of $|u_{n,\mathbf{k}}\rangle$ which makes the Berry potential non-smooth. As a result, the Chern number can take any integer value. In fact, the presence of a nonzero Chern number can be viewed as a topological obstruction to finding a globally smooth Gauge [10, 11].

3.1.2.2 The Chern Number as Change in Berry Phase

Having defined the Chern number in terms of the cell-periodic states $|u_{n,\mathbf{k}}\rangle$, we will now show an alternative form that is easier to calculate numerically and is used within the `Z2Pack` code [12]. For simplicity, we will look at the example where S is the Brillouin zone $\mathbf{k} \in [0, 1)^2$ of a two-dimensional material, in reduced coordinates. The results are equally applicable to other closed two-dimensional surfaces.

We divide the surface integral (3.10) for the Chern number into small segments S^i, as shown in Fig. 3.6a. The segments should be small enough that

$$C^i_{\text{part.}} = \frac{1}{2\pi} \int_{S^i} \mathscr{F}(\mathbf{k}).\mathrm{d}\mathbf{S} \tag{3.11}$$

is much smaller than one. The Chern number is then given as the sum of all segment integrals,

$$C = \sum_i C^i_{\text{part.}} \tag{3.12}$$

Since $\mathscr{A}(\mathbf{k})$ can be made to be *locally* smooth [13, 14], we can use Stokes' theorem to obtain

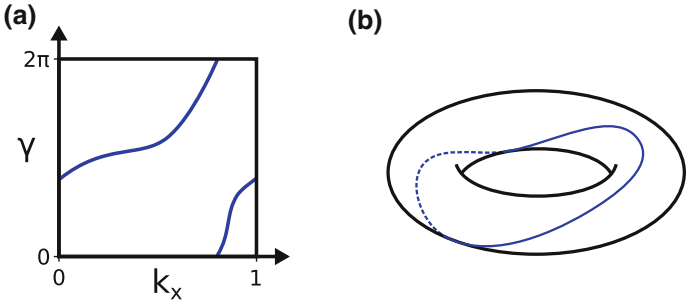


Fig. 3.7 **a** The Berry phase γ as a function k_x for an example system with $C = 1$. **b** Because both k_x and γ are periodic, the Chern number can be seen as the winding number of the Berry phase around a torus

$$C^i_{\text{part.}} \mod 1 = \frac{1}{2\pi}\int_{\partial S^i} \mathscr{A}(\mathbf{k}).\mathrm{d}\mathbf{k} \mod 1 = \frac{\gamma_{\partial S^i}}{2\pi} \mod 1, \tag{3.13}$$

where the modulus comes from the fact that the Berry phase is defined only modulo 2π. Since we imposed that $C^i_{\text{part.}}$ must be much smaller than one, we can still uniquely determine its value from $\gamma_{\partial S^i}/2\pi$ by adding an integer that minimizes the absolute value. Since the top and bottom parts of ∂S^i cancel out due to periodicity, we can write the Berry phase as

$$\gamma_{\partial S^i} = \gamma_{p^{i+1}} - \gamma_{p^i}, \tag{3.14}$$

where p^i and p^{i+1} are the paths at either side of the segment S^i, as shown in Fig. 3.6b. The Berry phase can also be understood as a function of k_x, since each path p^i is given by a fixed k_x. Because both γ and k_x are periodic, the Berry phase describes a line on a torus, as shown in Fig. 3.7. The winding number of this line around the torus is exactly the Chern number [15]. In other words, the Chern number can be calculated by *continuously* tracking the Berry phase on lines of constant k_x as it goes across the Brillouin zone. In practice, enforcing this continuity is a difficult task and is the goal of the convergence options discussed in Sect. 3.2.4.

3.1.2.3 Wilson Loop and Hybrid Wannier Charge Centers

The problem of calculating the Chern number is now reduced to calculating the Berry phase for closed loops in the Brillouin zone. This can be done by calculating the so-called Wilson loop [16] $W(C)$. The Wilson loop can be understood as a matrix that maps the states at a starting point $\mathbf{k}_0$ along the loop onto their images after parallel transport along C. For a discretization $\{\mathbf{k}_0, \ldots, \mathbf{k}_{n-1}, \mathbf{k}_n = \mathbf{k}_0\}$ of the path C, the Wilson loop can be approximated as [12, 16]

$$W(C) = M^{\mathbf{k}_0,\mathbf{k}_1} \cdot \ldots \cdot M^{\mathbf{k}_{n-1},\mathbf{k}_n}, \tag{3.15}$$

where

$$M_{m,n}^{\mathbf{k}_i,\mathbf{k}_j} = \left\langle u_{m,\mathbf{k}_i} \middle| u_{n,\mathbf{k}_j} \right\rangle \tag{3.16}$$

are the overlap matrices between Bloch functions at different $\mathbf{k}$. The eigenvalues λ_i of the Wilson loop are connected to the total Berry phase by [17]

$$\gamma_C = \sum_i \arg \lambda_i . \tag{3.17}$$

This reflects the fact that each λ_i is the rotation angle that is acquired by an eigenstate of the Wilson loop as it traverses the path C. Since the overlap matrices M can be readily computed, this gives a method for calculating the Chern number numerically. Of course, the convergence of the Wilson loop eigenvalues with respect to the discretization of C needs to be accounted for, which will be discussed in Sect. 3.2.4.

Another, equivalent, approach to calculating the Berry phase is by computing so-called hybrid Wannier charge centers [18, 19]. This method is based on the notion of Wannier orbitals, which are given by Fourier transforming the Bloch states:

$$|\mathbf{R}n\rangle = \frac{V}{(2\pi)^d} \int_{BZ} e^{-i\mathbf{k}.\mathbf{R}} \left|\psi_{n,\mathbf{k}}\right\rangle \mathrm{d}\mathbf{k}, \tag{3.18}$$

where d is the dimensionality of the system, and V is the unit cell volume. The resulting orbitals are localized, in contrast to the extended nature of the Bloch waves. Since the Bloch states that are used to construct Wannier orbitals can be changed by a Gauge transformation, the same is true for the Wannier orbitals. Their properties, in particular the localization and position in real space, depend sensitively on this choice of Gauge [20]. For the purposes of computing topological invariants, we introduce *hybrid* Wannier orbitals [19, 21], which are Fourier transformed only in one spatial direction and remain extended in the others:

$$\left|R_x, k_y, k_z; n\right\rangle = \frac{a_x}{2\pi} \int_{-\pi/a_x}^{\pi/a_x} e^{-ik_x R_x} \left|\psi_{n,\mathbf{k}}\right\rangle . \tag{3.19}$$

The average position of such an orbital can be thought of as a function of the remaining reciprocal space variables:

$$\bar{x}_n(k_y, k_z) = \left\langle 0, k_y, k_z; n \middle| \hat{x} \middle| 0, k_y, k_z; n \right\rangle . \tag{3.20}$$

This quantity known as the hybrid Wannier charge center (HWCC) is directly related to the Berry phase:

$$\gamma_C = \frac{2\pi}{a} \sum_n \bar{x}_n , \tag{3.21}$$

where C is the path along which the hybrid Wannier orbitals were Fourier transformed. Moreover, if the Gauge is chosen such that these hybrid Wannier orbitals

are maximally localized, the individual HWCC corresponds to the eigenvalues of the Wilson loop [12]

$$\bar{x}_i = \frac{2\pi}{a} \arg(\lambda_i), \tag{3.22}$$

up to possible reordering.

This equivalence between hybrid Wannier charge centers and the Berry phase gives rise to a physical interpretation of the Chern number C. As the momentum (k_x, in the case of Fig. 3.7) is varied across the Brillouin zone, the average position of the electrons in the orthogonal direction can change. Due to the periodicity of k_x, it must come back to the same position within the unit cell, but it can change into a different unit cell. This represents a *charge pumping* process, where each cycle of k_x moves the charge by C unit cells.

3.2 The Z2Pack Code

Having defined the Chern number and how it can be calculated in theory, we will now see how this knowledge can be applied in practice. First, we will give a brief overview of the Z2Pack code, introducing the necessary components for calculating Chern numbers. Next, we will show two examples, the Haldane model of a Chern insulator and the Weyl semimetal. Finally, we conclude this section with a discussion of the convergence options available in Z2Pack.

3.2.1 Introduction to the Code

Z2Pack is a Python [22] library which provides functionality for computing topological invariants. A basic knowledge of the Python language is required for using the code. For this, the reader is referred to the many excellent Python tutorials available online, in particular the official Python tutorial [23]. In the following, we will give a short introduction to using Z2Pack. For a more detailed description of the classes and functions described here, the reader may wish to consult the online documentation at www.z2pack.ethz.ch/doc.

In order to calculate the Chern number with Z2Pack, two inputs are needed: a description of the material (system) and a parametrization of the surface on which the invariant should be calculated. These inputs are passed to a function which calculates the hybrid Wannier charge center evolution across the surface. Optionally, this function can regularly save its progress to a file, to allow restarting aborted calculations. The result of this calculation can then be used to evaluate the Chern number or other topological invariants and create plots. Figure 3.8 shows an overview of this process and the modules involved in each step. We will now describe these steps in a bit more detail and show some example code.

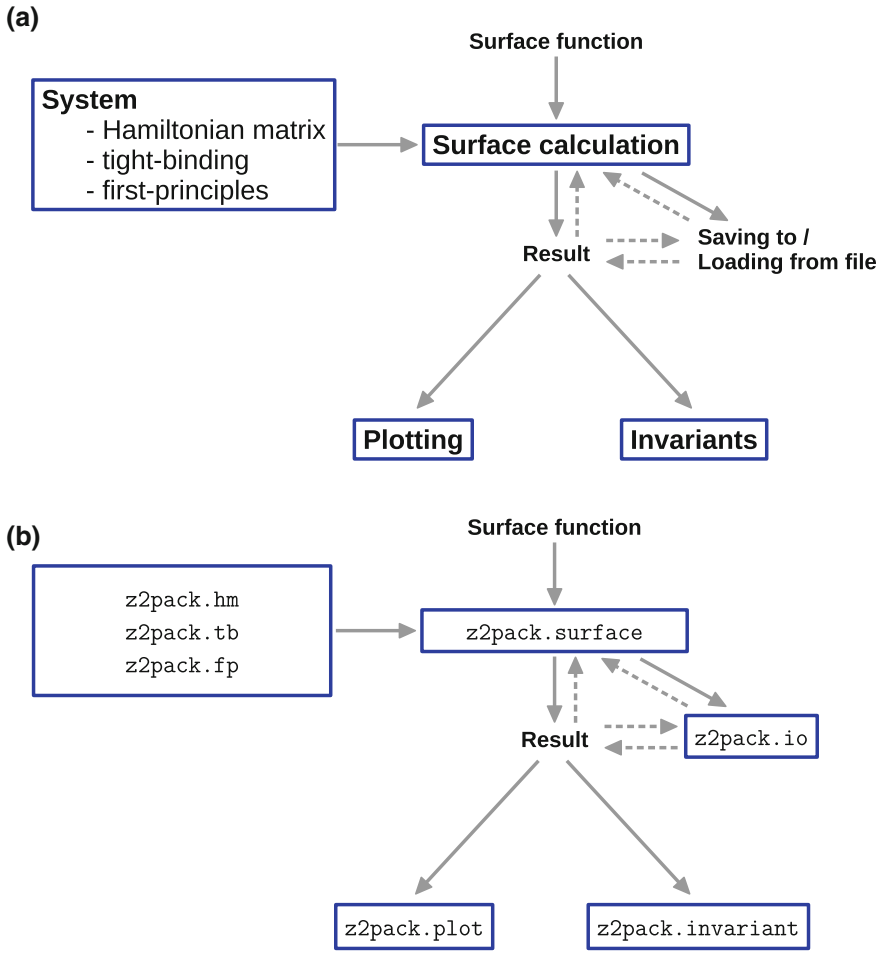


Fig. 3.8 **a** Overview of the process for calculating topological invariants for a reciprocal space surface of a given material. **b** Python modules corresponding to each of the steps in calculating topological invariants

The system for which topological invariants are to be calculated can be given in three different ways: First, it can be defined as an explicit function $H(\mathbf{k})$ describing the Hamiltonian matrix. This is useful for theoretical models, or when using the $\mathbf{k}.\mathbf{p}$ approximation. Listing 3.1 shows how such a system can be created using the `z2pack.hm.System` class. The first, required, input is a function that takes $\mathbf{k}$ and returns the corresponding matrix $H(\mathbf{k})$. An optional keyword argument `bands` can be passed to the class, to describe which band indices the topological invariant should be calculated for. It can be given either as an integer N, such that the lowest N bands will be taken into account, or as an explicit list of band indices. As is customary in

Python, the lowest band has index 0. By default, the lower half of all bands are taken into account.

```
import z2pack

def hamilton(k):
    ...
    # return Hamiltonian matrix for k

system = z2pack.hm.System(hamilton)

# Choose which bands are taken into account
# by specifying the 'bands' keyword

# lowest 2 bands
system = z2pack.hm.System(hamilton, bands=2)

# first and third band
system = z2pack.hm.System(hamilton, bands=[0, 2])
```

Listing 3.1 Example code for creating a `System` class defined with an explicit Hamiltonian matrix.

Second, the system can be given as a tight-binding model. For this, the TBmodels package is used,[4] which allows for defining tight-binding models either manually or from the output of the Wannier90 [24, 25] code. This is shown in Listing 3.2. For more details about how to construct the tight-binding model, we refer to the TBmodels documentation: www.z2pack.ethz.ch/tbmodels.

```
import tbmodels

# Create tight-binding model
model = tbmodels.Model(...)

# Example: Model from Wannier90 output file
model = tbmodels.Model.from_hr_file('wannier90_hr.dat')

system = z2pack.tb.System(model)
```

Listing 3.2 Creating a tight-binding system by using the `tbmodels.Model` class.

Finally, the system can be given as a first-principles calculation. As we have seen in Sect. 3.1.2.3, the overlap matrices $M^{\mathbf{k}_i,\mathbf{k}_j}$ between states at different k-points along a path are needed to calculate the hybrid Wannier charge centers. Z2Pack makes use of the fact that the Wannier90 code [24, 25] also requires these as an input. As a result, Z2Pack is in principle compatible with all DFT codes which interface to Wannier90, and the user needs to create the same input files as for running Wannier90. Since Z2Pack needs to dynamically call the first-principles code for different k-points, a function with which the k-point input can be created also needs to be supplied. For

[4] TBmodels was initially developed as part of Z2Pack, but later separated because it can be used outside of the scope of calculating topological invariants.

some codes, this is implemented in the `z2pack.fp.kpoint` module. Listing 3.3 shows how a first-principles system is defined to be used with VASP [26].

```
system = z2pack.fp.System(
    input_files=[
        'INCAR', 'POSCAR', 'POTCAR', 'wannier90.win'
    ],
    kpt_fct=z2pack.fp.kpoint.vasp,
    kpt_path='KPOINTS',
    command='mpirun $VASP >& log'
)
```

Listing 3.3 Defining a first-principles system for use with the VASP code.

Apart from the system, the only other input required for running a calculation is the surface on which the Chern number should be evaluated. This is simply given as a function

$$\begin{aligned} f : [0,1]^2 &\longrightarrow \mathbb{R}^d \\ (s,t) &\longmapsto \mathbf{k} \end{aligned} \tag{3.23}$$

which parametrizes the surface. Listing 3.4 shows a simple example for a surface function. It is important to note here that $\mathbf{k}$ should be given in reduced coordinates $\mathbf{k} \in [0,1)^d$. The reason for this is that it simplifies many things, for example, checking if the surface is a closed one.

```
# Defining an explicit function
def surface(s, t):
    return [s, t, 0]

# Equivalent expression using a lambda
surface = lambda s, t: [s, t, 0]
```

Listing 3.4 Two ways of defining a simple surface across the BZ at $k_z = 0$.

Given a system and surface, the hybrid Wannier charge centers can be calculated by calling the `z2pack.surface.run` function (see Listing 3.5). The return value of this function can then be passed to the `z2pack.invariant.chern` function to evaluate the Chern number. A simple plot of the sum of HWCC can be created by passing the result to the `z2pack.plot.chern` function. Since the plotting functionality is based on the popular matplotlib [27] library, the appearance of the plots can be fully customized using matplotlib commands.

```
result = z2pack.surface.run(
    system=system,
    surface=lambda s, t: [s, t, 0]
)

# Evaluate Chern number
z2pack.invariant.chern(result)

```

```
# Plot sum of HWCC
fig = z2pack.plot.chern(result)
fig.show()
```

Listing 3.5 Example code for calculating the hybrid Wannier charge centers, evaluating the Chern number and creating a simple plot.

The result object created by the `run` method can be saved into a file using the `z2pack.io.save` method (see Listing 3.6). To retrieve the stored object, the `z2pack.io.load` method can be used.

```
result = ...

# saving
z2pack.io.save(result, 'file_path.json')

# loading
result = z2pack.io.load('file_path.json')
```

Listing 3.6 Saving and loading Z2Pack results to a file.

Since the `run` calculation might take a while—especially for first-principles calculations—it is sometimes necessary to restart from an unfinished calculation. For this purpose, the `save_file` keyword can be specified when calling `run`, which means that the result will periodically be saved into the given file. If the `load` flag is set to `True`, the code will check for an existing result in that file before starting the calculation (see Listing 3.7). One needs to be careful, however, not to load old results the system or surface has changed. Another way to restart calculations is by explicitly passing a result, using the `init_result` keyword.

```
# Restart from file
result = z2pack.surface.run(
    system=system,
    surface=surface,
    save_file='file_path.json',
    load=True
)

# Restart from result
result2 = z2pack.surface.run(
    system=system,
    surface=surface,
    init_result=result
)
```

Listing 3.7 The `run` method can be restarted either from a result saved in a file, or by explicitly passing a result.

Finally, during the `run` call, Z2Pack continuously writes information about the current status to the console. Depending on the use case, this might be an unwanted distraction. Since Z2Pack uses the Python standard module `logging` for this purpose, its verbosity can easily be changed by setting the so-called *level* of the messages that will be printed, as shown in Listing 3.8. The log level describes the severity of a

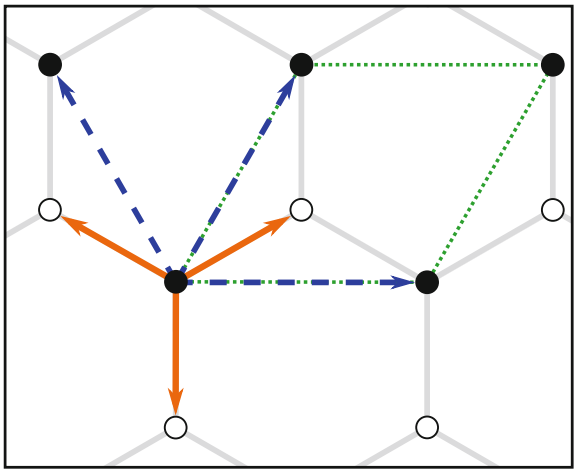


Fig. 3.9 A honeycomb lattice, with A and B sites marked with filled and empty circles, respectively. The unit cell is marked with a dotted green line. Nearest neighbors are indicated with a solid orange arrow, and next-nearest neighbors with a dashed blue arrow

given message. In Z2Pack, the two levels `logging.INFO` (for general messages) and `logging.WARNING` (for convergence issues) are used. Only messages which are at least as severe as the current level will be shown.

```
1 import logging
2
3 # Only show messages with at least 'WARNING' level importance
4 logging.getLogger('z2pack').setLevel(logging.WARNING)
```

Listing 3.8 By setting the log level, the messages printed by Z2Pack can be filtered by severity.

3.2.2 The Haldane Model [5]

The Haldane model [4] is a simple theoretical model for a Chern insulator. It describes two interleaved sub-lattices forming a honeycomb lattice, as shown in Fig. 3.9. The two sub-lattices have opposite on-site energies $\pm M$. Nearest- and next-nearest-neighbor hopping terms are included with strength t_1 and t_2, respectively. In order to break time-reversal symmetry, a microscopic magnetic field is introduced, adding a phase ϕ to the next-nearest-neighbor hopping. The full Hamiltonian of the system is given by

[5] Figures and Text in This Section Are Partly Copied from Previous Work of the Authors [28].

$$H(\mathbf{k}) = 2t_2 \cos\phi \left(\sum_i \cos(\mathbf{k}.\mathbf{b}_i) \right) \mathbb{I} + t_1 \sum_i \left[\cos(\mathbf{k}.\mathbf{a}_i)\sigma^x + \sin(\mathbf{k}.\mathbf{a}_i)\sigma^y \right] \quad (3.24)$$
$$+ \left[M - 2t_2 \left(\sum_i \sin(\mathbf{k}.\mathbf{b}_i) \right) \right] \sigma^z,$$

where $\mathbf{a}_i$ and $\mathbf{b}_i$ are the vectors connecting nearest- and next-nearest neighbors (solid orange / dashed blue arrows in Fig. 3.9), respectively, and σ^i are the Pauli matrices.

In this example, we will calculate the Chern number for the Haldane model for a particular value of the parameters M, t_1, t_2, and ϕ. First, we define a function describing the Hamiltonian, as a function of these parameters and $\mathbf{k}$, as shown in Listing 3.9.

```
# Define the Pauli matrices
IDENTITY = np.identity(2, dtype=complex)
PAULI_X = np.array([[0, 1], [1, 0]], dtype=complex)
PAULI_Y = np.array([[0, -1j], [1j, 0]], dtype=complex)
PAULI_Z = np.array([[1, 0], [0, -1]], dtype=complex)

# Define the function H(k)
def Hamilton(k, m, t1, t2, phi):
    kx, ky = k
    k_a = 2 * np.pi / 3. * np.array([
        kx + ky, -2 * kx + ky, kx - 2 * ky
    ])
    k_b = 2 * np.pi * np.array([kx, ky, ky - kx])
    H = (
        2 * t2 * np.cos(phi) *
        sum([np.cos(val) for val in k_b]) * IDENTITY
    )
    H += t1 * sum([np.cos(val) for val in k_a]) * PAULI_X
    H += t1 * sum([np.sin(val) for val in k_a]) * PAULI_Y
    H += m * PAULI_Z
    H -= (
        2 * t2 * np.sin(phi) *
        sum([np.sin(val) for val in k_b]) * PAULI_Z
    )
    return H
```

Listing 3.9 Defining a function that describes the Haldane Hamiltonian.

Next, we set some constants for the parameters M, t_1, t_2, ϕ and create a Z2Pack system from the Hamiltonian function as shown in Listing 3.10. We will take into account only the lower (occupied) band. Because the `Hamilton` function only takes a two-dimensional $\mathbf{k}$, we specify the dimension using the `dim` keyword. We can then run a surface calculation for this system.

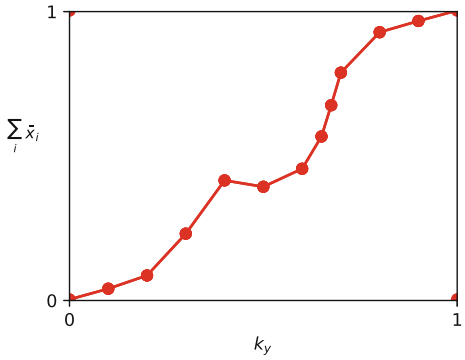


Fig. 3.10 The sum of HWCC as a function of k_x for the Haldane model with $M = 0.1$, $t_1 = 1$, $t_2 = 0.2$, and $\phi = \pi/2$. Since the HWCC winds around the torus once in positive direction, the Chern number is $C = 1$

```
# Set the constants for the Haldane model
M = 0.1
T1 = 1.
T2 = 0.2
PHI = 0.5 * np.pi

# Create a Z2Pack system
system = z2pack.hm.System(
    lambda k: Hamilton(k, m=M, t1=T1, t2=T2, phi=PHI),
    bands=1,
    dim=2
)
# Run the surface calculation
result = z2pack.surface.run(
    system=system,
    surface=lambda s, t: [t, s]
)
```

Listing 3.10 Defining a system and running the surface calculation for specific values of the Haldane parameters.

Finally, we evaluate the Chern number and create a figure that shows the HWCC evolution, as shown in Listing 3.11. This produces the image shown in Fig. 3.10. The complete Haldane model example can be seen in Listing 3.12.

```
# Evaluate the Chern number
print('Chern number:', z2pack.invariant.chern(result))

# Create a figure
fig, ax = plt.subplots(figsize=[4, 3])
z2pack.plot.chern(result, axis=ax)
ax.set_xlabel(r'$k_y$')
ax.set_ylabel(
    r'$\sum_i \,\bar{x}_i$', rotation='horizontal', ha='right'
)
ax.set_xticks([0, 1])
ax.set_yticks([0, 1])
```

```
fig.savefig('haldane.pdf', bbox_inches='tight')
```

Listing 3.11 Evaluating the Chern number and creating a plot from the calculation result.

```
import logging

import z2pack
import numpy as np
import matplotlib.pyplot as plt

logging.getLogger('z2pack').setLevel(logging.WARNING)

# Define the Pauli matrices
IDENTITY = np.identity(2, dtype=complex)
PAULI_X = np.array([[0, 1], [1, 0]], dtype=complex)
PAULI_Y = np.array([[0, -1j], [1j, 0]], dtype=complex)
PAULI_Z = np.array([[1, 0], [0, -1]], dtype=complex)

# Define the function H(k)
def Hamilton(k, m, t1, t2, phi):
    kx, ky = k
    k_a = 2 * np.pi / 3. * np.array([
        kx + ky, -2 * kx + ky, kx - 2 * ky
    ])
    k_b = 2 * np.pi * np.array([kx, ky, ky - kx])
    H = (
        2 * t2 * np.cos(phi) *
        sum([np.cos(val) for val in k_b]) * IDENTITY
    )
    H += t1 * sum([np.cos(val) for val in k_a]) * PAULI_X
    H += t1 * sum([np.sin(val) for val in k_a]) * PAULI_Y
    H += m * PAULI_Z
    H -= (
        2 * t2 * np.sin(phi) *
        sum([np.sin(val) for val in k_b]) * PAULI_Z
    )
    return H

# Set the constants for the Haldane model
M = 0.1
T1 = 1.
T2 = 0.2
PHI = 0.5 * np.pi

# Create a Z2Pack system
system = z2pack.hm.System(
    lambda k: Hamilton(k, m=M, t1=T1, t2=T2, phi=PHI),
    bands=1,
    dim=2
)
# Run the surface calculation
result = z2pack.surface.run(
    system=system,
```

```
    surface=lambda s, t: [t, s]
)

# Evaluate the Chern number
print('Chern number:', z2pack.invariant.chern(result))

# Create a figure
fig, ax = plt.subplots(figsize=[4, 3])
z2pack.plot.chern(result, axis=ax)
ax.set_xlabel(r'$k_y$')
ax.set_ylabel(
    r'$\sum_i\,\bar{x}_i$', rotation='horizontal', ha='right'
)
ax.set_xticks([0, 1])
ax.set_yticks([0, 1])
fig.savefig('haldane.pdf', bbox_inches='tight')
```

Listing 3.12 The complete Haldane model example.

3.2.3 Identifying Weyl Semimetals

So far, we have considered the Chern number in the context of insulating materials. From the discussion in Sect. 3.1.1.3, we know that a Chern number can be defined on any closed two-dimensional surface in the Brillouin zone where the bands are gapped. However, in three-dimensional materials, the band gap can still close outside of that specific surface. This can be used to classify topological semimetals, in particular to identify so-called Weyl nodes.

Weyl nodes are linear touching points of two bands in a single point. Their Hamiltonian can locally be described as [29]

$$H(\mathbf{k}) = \sum_{\substack{i\in\{x,y,z\}\\ j\in\{0,x,y,z\}}} A_{i,j}\, k_i\, \sigma^j, \tag{3.25}$$

where σ^j are the Pauli matrices, and $A_{i,j}$ characterizes the Weyl node. Topologically, Weyl nodes are remarkable because they are a quantized source or sink of Berry connection, depending on their chirality [30]. Since the Chern number measures the flux of Berry connection through a surface, we can determine the chirality of a Weyl node by calculating the Chern number on a sphere enclosing it [12, 31].

Listing 3.13 shows how the Chern number can be calculated for a simple symmetric Weyl node $H(\mathbf{k}) = \sum_i k_i \sigma^i$. The techniques used are the same as for the Haldane example. For defining the surface—a sphere of radius $r = 0.01$—Z2Pack provides a helper function `z2pack.shape.Sphere`, with which a sphere can be defined through its center and radius. The plot created in this example is shown in Fig. 3.11.

```
import numpy as np
import matplotlib.pyplot as plt

import z2pack

# Define Pauli vector
PAULI_X = np.array([[0, 1], [1, 0]], dtype=complex)
PAULI_Y = np.array([[0, -1j], [1j, 0]], dtype=complex)
PAULI_Z = np.array([[1, 0], [0, -1]], dtype=complex)
PAULI_VECTOR = list([PAULI_X, PAULI_Y, PAULI_Z])

def Hamilton(k):
    """simple 2-band hamiltonian k.sigma with a Weyl point at k=0"""
    res = np.zeros((2, 2), dtype=complex)
    for kval, p_mat in zip(k, PAULI_VECTOR):
        res += kval * p_mat
    return res

# Create the System
system = z2pack.hm.System(Hamilton)

# the surface is a sphere around the Weyl point
result = z2pack.surface.run(
    system=system,
    surface=z2pack.shape.Sphere([0., 0., 0.], 0.01)
)
print('Chern number:', z2pack.invariant.chern(result))

# Create plot
fig, ax = plt.subplots(figsize=[4, 3])
z2pack.plot.chern(result, axis=ax)

ax.set_xlabel(r'$\theta$')
ax.set_xticks([0, 1])
ax.set_xticklabels([r'$0$', r'$\pi$'])
ax.set_ylabel(r'$\bar{\phi}$', rotation='horizontal')
ax.set_yticks([0, 1])
ax.set_yticklabels([r'$0$', r'$2\pi$'])
ax.set_title(r'$\vec{k}.\vec{\sigma}$')
plt.savefig('weyl.pdf', bbox_inches='tight')
```

Listing 3.13 Calculating the Chern number for a simple **k**.**p** model of a Weyl node.

3.2.4 Convergence Options

In the previous examples, we have used the `surface.run` function without specifying any convergence options. This means that we relied on the default values defined in Z2Pack. While these should work for a wide range of potential applica-

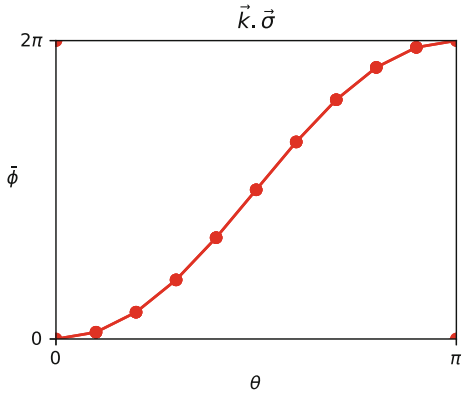


Fig. 3.11 The HWCC on loops around a sphere enclosing a Weyl node, as a function of the altitude angle θ

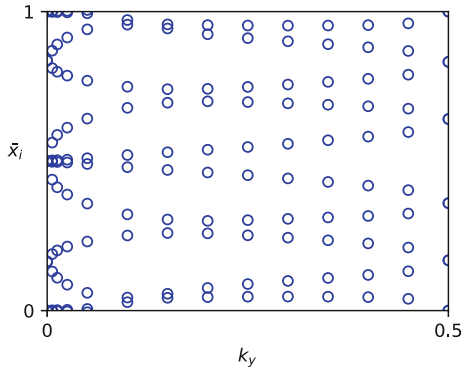


Fig. 3.12 Hybrid Wannier charge center evolution along half of the $k_z = 0$ plane for Bismuth

tions, it is still important to understand the convergence mechanisms and how they can be tuned.

First, we should note that for the convergence criteria described here, the individual hybrid Wannier charge centers are taken into account, not only their sum. Figure 3.12 shows a typical evolution of HWCC $\bar{x}_i$ evaluated at discrete values of k_y. Note that the HWCC is not connected across different k_y, since it is not possible to uniquely identify them.

The first convergence option defined in Z2Pack is convergence with respect to the discretization along the line for which the HWCC is calculated. In rough terms, the number of k-points along the line is increased until the change in HWCC positions is less than a certain threshold, `pos_tol`. How many k-points are used in each step is defined by the `iterator` keyword. This input can be any iterable object (for example a list) of integers. By default, it is set to `range(8, 27, 2)`, meaning that the code will start with eight k-points and then increase in steps of two until 26. If convergence is still not reached after this point, a warning will be generated. In this case, the best course of action is usually to increase the maximum number of k-points and restart the calculation from the previous result.

Fig. 3.13 For comparing their positions, HWCC is indexed starting from the largest gap between any two charge centers

One detail that might be worth noting is how the movement of HWCC is calculated. Since, as mentioned above, the HWCC cannot be uniquely identified, we can not simply calculate the movement for each charge center individually. Because a HWCC can cross from 1 to 0 or vice versa, indexing them by their position from zero also does not work. Instead, the HWCC is indexed starting from the *largest gap*[6] between any two HWCCs (when considering both the old and new charge centers), as shown in Fig. 3.13. The positions of HWCC with the same index are then compared, and the maximum of these differences is computed.

In addition to convergence along a single k-point line, convergence in the orthogonal direction needs to be taken into account. This corresponds to the discretization shown in Sect. 3.1.2.2. Using the same technique as before, the movement of HWCC between two neighboring lines is calculated. If it is larger the threshold value `move_tol`, an additional line is added between the two neighbors. To avoid calculations running indefinitely, which could occur if there is a discontinuity in the HWCC spectrum due to a band gap closure, a minimum allowed distance between neighboring lines `min_neighbor_dist` is defined. Again, a warning is issued if convergence cannot be reached. It is important to note that, due to the way the movement is calculated, two HWCCs that exactly exchange places cannot be detected. This can happen in cases where the band gap becomes very small at some point in the Brillouin zone, and the character of the bands changes very rapidly. For such systems, it is important to increase the initial number of lines through the `num_lines` keyword.

Finally, Z2Pack also monitors the distance between the middle of the largest gap in each line and the HWCC positions in neighboring lines (see Fig. 3.14). If the distance is smaller than `gap_tol` times the size of the largest gap, an additional line is again placed between the two neighboring lines. The reason for this additional test should become obvious in the next section, as it is related to how the $\mathbb{Z}_2$ invariant is calculated. Scaling the tolerance with the size of the largest gap is necessary in this case because otherwise the condition can be impossible to fulfill, especially when there are many evenly spaced HWCC.

[6]Note that this gap in the HWCC spectrum is not related to the band gap of the energy spectrum.

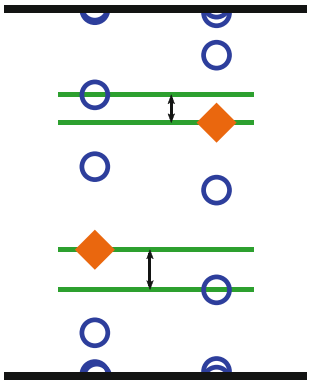


Fig. 3.14 The minimum distance between the middle of the largest gap (orange diamond) and HWCC (blue circle) in neighboring lines determines whether the `gap_tol` criterion is met

3.3 Time-Reversal Symmetry: $\mathbb{Z}_2$ Classification[7]

In the previous sections, we have seen how an isolated set of bands can be classified topologically according to their Chern number. Now, we will show how this classification can be enriched in the presence of symmetries. In particular, we will show that time-reversal invariant materials can be classified according to a $\mathbb{Z}_2$ index. After a theoretical introduction, we describe how the $\mathbb{Z}_2$ index is computed with Z2Pack. Finally, the example of a tight-binding model in the non-trivial $\mathbb{Z}_2$ phase is shown.

3.3.1 Individual Chern Numbers

In Sect. 3.1.1.2, we have seen that topological phases can be defined on manifolds in reciprocal space, if we choose a set of Bloch functions $\{|u_{n,\mathbf{k}}\rangle\}$ such that they span a smooth vector space $V_\mathbf{k}$. The most convenient way of achieving this smoothness, which we have used so far, is by choosing an isolated set of bands. This leads to a classification into topological states which can only be adiabatically changed by closing the band gap and are characterized by the Chern number. However, choosing isolated bands is by no means the only possible way to create a smooth $V_\mathbf{k}$. For the Hamiltonian of (3.3) for example, we could just pick state $|a\rangle$ everywhere.

Here, we aim to find a more complex topological classification by subdividing the occupied states into smooth parts. In general, if the Hilbert space $\mathscr{H}$ of a given problem can be written as a sum of smooth Hilbert spaces,

$$\mathscr{H} = \bigoplus_i \mathscr{H}_i, \tag{3.26}$$

[7] Figures and text in this section were partly copied from previous work of the authors [12, 28].

then each of the Hilbert spaces has a well-defined Chern number C_i. These *individual Chern numbers* [19] sum together to the Chern number of the full Hilbert space:

$$C = \sum_i C_i. \tag{3.27}$$

However, in general, these individual Chern numbers do not carry much meaning, since the choice how to split up the Hilbert space is arbitrary. In the presence of a symmetry S, however, the Hilbert space can be split up according to the symmetry eigenvalues. For example, consider a mirror symmetry with eigenvalues $\pm i$. On the mirror-symmetric surface, S and $H(\mathbf{k})$ commute. Therefore, the Bloch functions $|u_{n,\mathbf{k}}\rangle$ can be separated into $+i$ and $-i$ eigenstates. Both eigenspaces have a well-defined Chern number:

$$C = C_i + C_{-i}. \tag{3.28}$$

This gives rise to a *symmetry-protected* [32–34] topological classification. Materials can have a zero total Chern number, but nonzero individual Chern numbers. Such a topological phase is protected as long as both the band gap remain open and the symmetry is respected. If the symmetry is broken, a mixing of the two eigenspaces can change the topological phase.

Time-reversal symmetry θ leads to a particularly interesting and well-known topological classification. Unlike spatial symmetries, it is an anti-unitary symmetry and squares to -1 in the spinful case. As a result, the Bloch functions come in so-called Kramers pairs [5, 6]

$$\begin{aligned} \theta\,|u^{I}_{m,\mathbf{k}}\rangle &= |u^{II}_{m,\mathbf{k}}\rangle \\ \theta\,|u^{II}_{m,\mathbf{k}}\rangle &= -|u^{I}_{m,\mathbf{k}}\rangle. \end{aligned} \tag{3.29}$$

There is a Gauge in which these states are smooth [14, 19], and thus, they have well-defined, opposite [18] individual Chern numbers

$$C^{I}_{m} = -C^{II}_{m}. \tag{3.30}$$

Furthermore, the hybrid Wannier charge centers are related by [18]

$$\bar{x}^{I}_{m}(k_y) = \bar{x}^{II}_{m}(-k_y), \tag{3.31}$$

meaning that they are degenerate for the time-reversal invariant lines $k_y = 0, \pi$. In order to define a topological invariant, we group the states by their pair indices I, II. The two groups then have individual Chern numbers

$$C^{I} = -C^{II}. \tag{3.32}$$

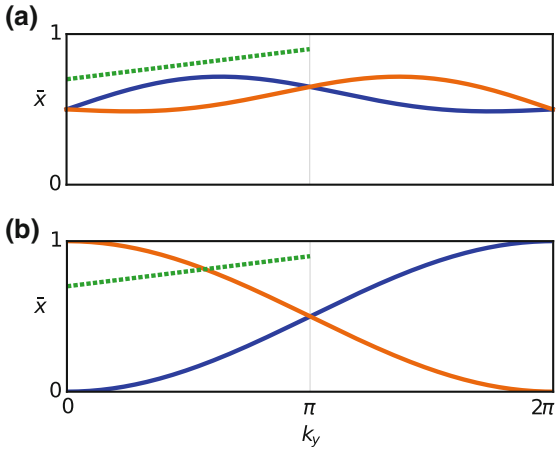


Fig. 3.15 Hybrid Wannier charge centers for a two-band time-reversal invariant system. **a** Trivial phase. The two bands each have a zero individual Chern number. **b** Non-trivial phase. The two bands have individual Chern numbers ± 1

However, these Chern numbers are not Gauge invariant. This can be seen by changing the sign of one of the two states:

$$\begin{aligned} \left|\tilde{u}_m^{II}\right\rangle &= \left|u_m^{I}\right\rangle \\ \left|\tilde{u}_m^{I}\right\rangle &= -\left|u_m^{II}\right\rangle \end{aligned} \tag{3.33}$$

These states still obey equation 3.29, and the individual Chern number of each state remains the same. Yet the two states have switched their pair indices. As a result, the Chern number C^I is changed by $C_m^{II} - C_m^I = 2C_m^{II}$. Since this re-labeling of Kramers pairs can only ever change the Chern numbers by an even number, a topological invariant can be defined as

$$\mathbb{Z}_2 = C_m^I \mod 2. \tag{3.34}$$

In practice, the states do not need to be split by their pair indices to calculate the $\mathbb{Z}_2$ invariant. Instead, we can use the fact that the hybrid Wannier charge centers must be doubly degenerate at the time-reversal invariant momenta. An arbitrary line between zero and π (dotted green line in Fig. 3.15) will cross an even number of HWCC in the topologically trivial case and an odd number in the non-trivial case [18]. This principle is used in Z2Pack to calculate the $\mathbb{Z}_2$ invariant.

When computing the $\mathbb{Z}_2$ invariant numerically, the challenge in using the approach described above is that we cannot uniquely identify hybrid Wannier charge centers. In other words, we do not know how the HWCC connects between two discrete values of k_y. We can get around this issue, however, by cleverly choosing the line $x_{\text{cut}}(k_y)$ for which the number of crossings is counted. Since we want a crossing to be as obvious as possible, we choose it to always be in the middle of the largest gap between any two HWCCs, as shown in Fig. 3.16. The number of crossings is then

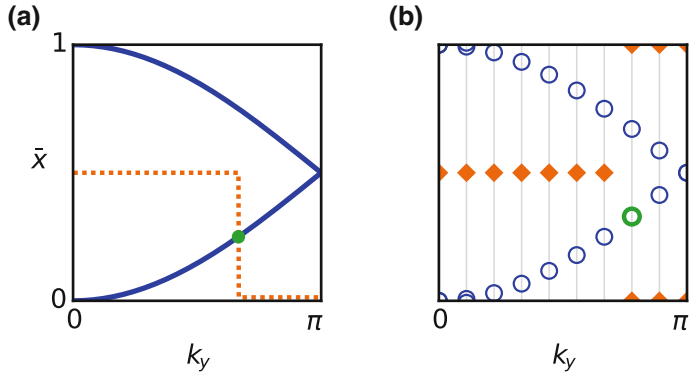


Fig. 3.16 Sketch showing the $\mathbb{Z}_2$ calculation. a Continuous case. The HWCC (solid blue line) is crossed exactly once by x_{cut} (dashed orange line), at the green point. **b** Discrete case. The HWCC (blue circles) and middle of the largest gap (orange diamonds) are known only for discrete k_y. Crossings are counted when the HWCC value lies between the largest gaps of the current and previous lines (green circle)

counted by summing up the HWCCs which lie between the current and previous value of the largest gap.

The interface for calculating the $\mathbb{Z}_2$ invariant in Z2Pack is very similar to that for calculating the Chern number. Given the result of a surface calculation, it can be evaluated with the `z2pack.invariant.z2` function. Note that the surface which is used to calculate the HWCC should cover only half the Brillouin zone.

3.3.2 Tight-Binding Example

For the final example in this chapter, we will consider a system of two interpenetrating square lattices A and B each carrying one electron per unit cell, as shown in Fig. 3.17. Let us take into account nearest and next-nearest-neighbor hopping terms, with strength t_1 and t_2, respectively. Each lattice site has two possible states (spin up / down), both carrying equal on-site energies $+1$ for sub-lattice A, and -1 for sub-lattice B.

Including only hopping terms between orbitals of the same spin direction, let the nearest-neighbor hopping terms from sub-lattice A to B have phases $\{1, i, -i, -1\}$ (counter-clockwise) for the spin-up case and its conjugate for the spin down case. Next-nearest-neighbor hopping terms do not carry a phase, but are positive for sub-lattice A and negative for sub-lattice B. The resulting Hamiltonian is

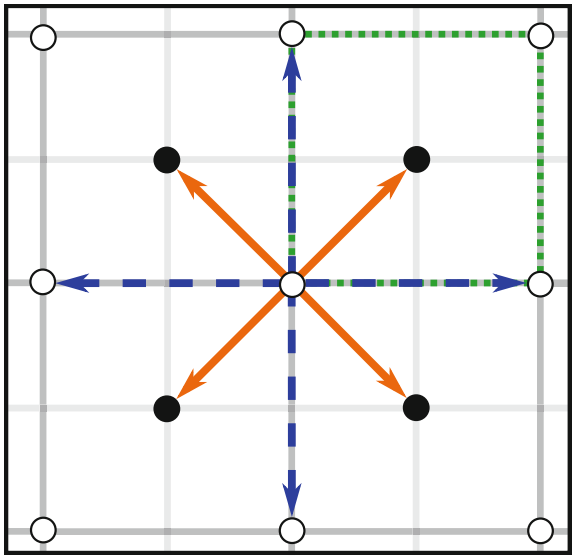


Fig. 3.17 Two inter-penetrating square lattices. The unit cell is shown in green (dotted line). Solid orange arrows connect nearest neighbors, and dashed blue arrows connect next-nearest neighbors

$$H(k_x, k_y) = \left(1 + 2t_2\left[\cos k_x + \cos k_y\right]\right)\sigma_z \otimes \sigma_0 - 2t_1\left[\sin\left(\frac{k_x + k_y}{2}\right)\sigma_y \otimes \sigma_0 + \sin\left(\frac{k_x - k_y}{2}\right)\sigma_x \otimes \sigma_z\right]. \tag{3.35}$$

The tight-binding model is built using the TBmodels code, as shown in listing 3.14. In the constructor of the `tmodels.Model`, the positions, on-site energies, and occupation number are set. Then, the `add_hop` method is used to add all hopping terms. Note that the inverse hopping terms (e.g., nearest-neighbor hopping from sub-lattice B to A) are added automatically. The surface calculation is performed in exactly the same way as for the previous examples, except that the surface now only covers half the Brillouin zone. Finally, the `z2pack.plot.wcc` method is used to plot the HWCC, and `z2pack.invariant.z2` is used to calculate the $\mathbb{Z}_2$ invariant. The resulting plot can be seen in Fig. 3.18, showing the non-trivial $\mathbb{Z}_2$ phase.

```
1  import itertools
2
3  import z2pack
4  import tbmodels
5  import matplotlib.pyplot as plt
6
7  T1, T2 = (0.2, 0.3)
8
9  # Create a ''bare'' tight-binding model, with only
10 # on-site energies.
11 model = tbmodels.Model(
12     on_site=(1, 1, -1, -1),
```

```
    pos=[[0., 0.], [0., 0.], [0.5, 0.5], [0.5, 0.5]],
    occ=2,
)

# Add nearest neighbor hopping terms
for phase, R in zip(
    [1, 1j, -1j, -1],
    itertools.product([0, -1], [0, -1])
):
    model.add_hop(
        overlap=phase * T1,
        orbital_1=0,
        orbital_2=2,
        R=R
    )
    model.add_hop(
        overlap=phase.conjugate() * T1,
        orbital_1=1,
        orbital_2=3,
        R=R
    )

# Add next-nearest neighbor hopping terms
for R in (
    (r[0], r[1]) for r in itertools.permutations([0, 1])
):
    model.add_hop(T2, 0, 0, R)
    model.add_hop(T2, 1, 1, R)
    model.add_hop(-T2, 2, 2, R)
    model.add_hop(-T2, 3, 3, R)

# Create System instance
tb_system = z2pack.tb.System(model, dim=2)

# Run HWCC calculation
result = z2pack.surface.run(
    system=tb_system, surface=lambda s, t: [t, s / 2.]
)

# Create figure
fig, ax = plt.subplots(figsize=[4, 3])
z2pack.plot.wcc(result, axis=ax)
ax.set_xlabel(r'$k_y$')
ax.set_xticks([0, 1])
ax.set_xticklabels([r'$0$', r'$\pi$'])
ax.set_ylabel(r'$\bar{x}_i$', rotation='horizontal')
ax.set_yticks([0, 1])
plt.savefig('tb_wcc.pdf', bbox_inches='tight')

# Calculate Z2 invariant
print("Z2 invariant:", z2pack.invariant.z2(result))
```

Listing 3.14 Calculating the $\mathbb{Z}_2$ invariant for a tight-binding model of two inter-penetrating square lattices.

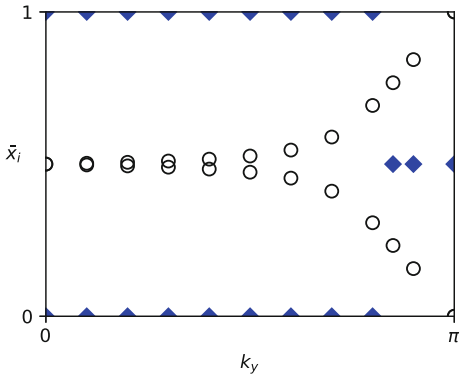


Fig. 3.18 Hybrid Wannier charge center evolution (black circles) and their largest gap (blue diamonds) for the system of two inter-penetrating square lattices, with $t_1 = 0.2$ and $t_2 = 0.3$

Acknowledgements The authors were supported by Microsoft Research, the Swiss National Science Foundation through the National Competence Centers in Research MARVEL and QSIT, and the ERC Advanced Grant SIMCOFE.

References

1. B.A. Bernevig, T.L. Hughes, *Topological Insulators and Topological Superconductors* (Princeton University Press, New Jercy, 2013)
2. M. Nakahara, *Geometry, Topology and Physics* (Taylor and Francis Group, London, 2003)
3. F. Bloch, Über die quantenmechanik der elektronen in kristallgittern. Z. Phys. **52**(7), 555–600 (1929)
4. F.D.M. Haldane, Model for a quantum Hall effect without Landau levels: condensed-matter realization of the parity anomaly. Phys. Rev. Lett. **61**(18), 2015–2018 (1988)
5. C.L. Kane, E.J. Mele, $\mathbb{Z}_2$ topological order and the quantum spin Hall effect. Phys. Rev. Lett. **95**, 146802 (2005)
6. C.L. Kane, E.J. Mele, Quantum spin Hall effect in graphene. Phys. Rev. Lett. **95**, 226801 (2005)
7. J. Zak, Berry's phase for energy bands in solids. Phys. Rev. Lett. **62**(23), 2747–2750 (1989)
8. S.-S. Chern, Characteristic classes of Hermitian manifolds. Ann. Math. 85–121 (1946)
9. D.J. Thouless, M. Kohmoto, M.P. Nightingale, M. den Nijs, Quantized Hall conductance in a two-dimensional periodic potential. Phys. Rev. Lett. **49**(6), 405–408 (1982)
10. T. Thonhauser, D. Vanderbilt, Insulator/Chern-insulator transition in the Haldane model. Phys. Rev. B **74**(23), 235111 (2006)
11. D.J. Thouless, Wannier functions for magnetic sub-bands. J. Phys. C **17**(12), L325 (1984)
12. D. Gresch, G. Autès, O.V. Yazyev, M. Troyer, D. Vanderbilt, B.A. Bernevig, A.A. Soluyanov, Z2Pack: numerical implementation of hybrid Wannier centers for identifying topological materials. Phys. Rev. B **95**, 075146 (2017)
13. A.A. Soluyanov, D. Vanderbilt, Smooth gauge for topological insulators. Phys. Rev. B **85**(11), 115415 (2012)
14. G.W. Winkler, A.A. Soluyanov, M. Troyer, Smooth gauge and Wannier functions for topological band structures in arbitrary dimensions. Phys. Rev. B **93**(3), 035453 (2016)
15. A.A. Soluyanov, Topological Aspects of Band Theory. Ph.D. thesis, Rutgers University-Graduate School-New Brunswick, 2012
16. A. Alexandradinata, X. Dai, B.A. Bernevig, Wilson-loop characterization of inversion-symmetric topological insulators. Phys. Rev. B **89**(15), 155114 (2014)

17. R. Leone, The geometry of (non)-Abelian adiabatic pumping. J. Phys. A Math. Theor. **44**(29), 295301 (2011)
18. A.A. Soluyanov, D. Vanderbilt, Computing topological invariants without inversion symmetry. Phys. Rev. B **83**, 235401 (2011)
19. A.A. Soluyanov, D. Vanderbilt, Wannier representation of $\mathbb{Z}_2$ topological insulators. Phys. Rev. B **83**(3), 035108 (2011)
20. N. Marzari, D. Vanderbilt, Maximally localized generalized Wannier functions for composite energy bands. Phys. Rev. B **56**(20), 12847–12865 (1997)
21. C. Sgiarovello, M. Peressi, R. Resta, Electron localization in the insulating state: application to crystalline semiconductors. Phys. Rev. B **64**(11), 115202 (2001)
22. G. Van Rossum et al., Python programming language. in *USENIX Annual Technical Conference*, vol. 41 (2007)
23. The Python Tutorial, Accessed 14 Jan 2018
24. A.A. Mostofi, J.R. Yates, Y.-S. Lee, I. Souza, D. Vanderbilt, N. Marzari, wannier90: A tool for obtaining maximally-localised Wannier functions. Comput. Phys. Commun. **178**(9), 685–699 (2008)
25. A.A. Mostofi, J.R. Yates, G. Pizzi, Y.-S. Lee, I. Souza, D. Vanderbilt, N. Marzari, An updated version of wannier90: a tool for obtaining maximally-localised Wannier functions. Comput. Phys. Commun. **185**(8), 2309–2310 (2014)
26. G. Kresse, J. Furthmüller, Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. Comput. Mater. Sci. **6**(1), 15–50 (1996)
27. J.D. Hunter, Matplotlib: a 2D graphics environment. Comput. Sci. Eng. **9**(3), 90–95 (2007)
28. D. Gresch, Identifying Topological States in Matter, Master's thesis, ETH Zurich, Zurich, 2015
29. H.B. Nielsen, M. Ninomiya, The Adler–Bell–Jackiw anomaly and Weyl fermions in a crystal. Phys. Lett. B **130**(6), 389–396 (1983)
30. G. Volovik, Zeros in the Fermion spectrum in superfluid systems as diabolical points. JETP Lett. **46**(2) (1987)
31. A.A. Soluyanov, D. Gresch, Z. Wang, Q. Wu, M. Troyer, X. Dai, B.A. Bernevig, Type-II Weyl semimetals. Nature **527**(7579), 495–498 (2015)
32. A. Alexandradinata, B.A. Bernevig, Berry-phase description of topological crystalline insulators. Phys. Rev. B **93**(20), 205104 (2016)
33. L. Fu, Topological crystalline insulators. Phys. Rev. Lett. **106**(10), 106802 (2011)
34. J.C.Y. Teo, L. Fu, C.L. Kane, Surface states and topological invariants in three-dimensional topological insulators: application to $Bi_{1-x}Sb_x$. Phys. Rev. B **78**, 045426 (2008)