Problem 1. Gauge freedom in Bloch wave functions

Wave functions generally have a freedom of what is chosen in the phase. This is also true of Bloch wave functions $\psi_{n\mathbf{k}}(\mathbf{r})$ which we explore here. Define a new set of Bloch wave functions by

$$|\tilde{\psi}_{n\mathbf{k}}\rangle = e^{-i\beta_n(\mathbf{k})}\,|\psi_{n\mathbf{k}}\rangle\,, \tag{1}$$

where β_n is a real function of $\mathbf{k}$. The transformation from the old set to the new set is what we will call a "gauge transformation."

(a) A "periodic gauge" is one for which $|\psi_{n,\mathbf{k}+\mathbf{G}}\rangle = |\psi_{n\mathbf{k}}\rangle$ (i.e., equal with the same phase). What has to be true about $\beta_n(\mathbf{k})$ if it is to preserve the periodicity of the gauge?

(b) Show that the $|u_{n\mathbf{k}}\rangle$ transform in the same way as the $|\psi_{n\mathbf{k}}\rangle$ under a gauge transform. (recall $\psi_{n\mathbf{k}}(\mathbf{r}) = e^{i\mathbf{k}\cdot\mathbf{r}}u_{n\mathbf{k}}(\mathbf{r})$ where $u_{n\mathbf{k}}(r)$ is periodic with respect to a lattice vector $\mathbf{R}$.)

(c) We will soon be introducing the concept of "Berry connection" $\mathbf{A}_n(\mathbf{k}) = i\,\langle u_{n\mathbf{k}}|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\rangle$. Derive an expression that describes how this transforms under a gauge transformation. That is, express $\tilde{\mathbf{A}}_n(\mathbf{k}) = i\,\langle \tilde{u}_{n\mathbf{k}}|\nabla_{\mathbf{k}}\tilde{u}_{n\mathbf{k}}\rangle$ in terms of $\mathbf{A}_n(\mathbf{k})$ plus a correction.

(d) Show that $\nabla_{\mathbf{k}} \times \mathbf{A}_n$ is gauge-invariant.

Problem 2. Bloch's theorem, finite lattices, and flux

Consider a finite lattice given by

$$H = -t\sum_{n=0}^{L-1}(|(n+1)a\rangle\langle na| + |na\rangle\langle(n+1)a|), \tag{2}$$

(a) For the boundary conditions $|L\rangle = |0\rangle$ (periodic boundary conditions), solve this Hamiltonian for eigenvectors and eigenvalues.

(b) For the boundary conditions $|L\rangle = e^{i\vartheta}|0\rangle$ ("twisted" boundary conditions), solve the Hamiltonian for eigenvectors and eigenvalues.

(c) Returning to $|L\rangle = |0\rangle$ (periodic), we can thread magnetic flux through the system, this is often included by *Peierls substitution*, where we transform the hopping terms in the Hamiltonian $|m\rangle\langle n| \to e^{i\int_{na}^{ma} A(x)dx}|m\rangle\langle n|$ in the Hamiltonian, for a vector potential $A(x)$ (defined as the angular component). If this is a ring then $\oint A(x)dx = \Phi$ for an enclosed flux Φ. Assuming that $A(x)$ is constant in x, find the eigenvalues and eigenvectors of this ring. How is Φ related to ϑ from the previous question?

(d) Now, assume we have the simple, infinite tight-binding model

$$H = -t\sum_{n=-\infty}^{\infty}(|(n+1)a\rangle\langle na| + |na\rangle\langle(n+1)a|), \tag{3}$$

we can solve this, inefficiently, by picking a "unit cell" of size L. In particular, first show that Hamiltonian can be rewritten as

$$H = -t\sum_{m=-\infty}^{\infty}\left[\sum_{n=0}^{L-2}(|mL+n+1\rangle\langle mL+n| + |mL+n\rangle\langle mL+n+1|) + |(m+1)L\rangle\langle mL+L-1| + |mL+L-1\rangle\langle(m+1)L|\right]. \tag{4}$$

If we take as our translation operator $\hat{T}_L = e^{-i\hat{k}L}$, and its eigenvalues $0 \le kL < 2\pi$, this defines a Brillouin zone of size $2\pi/L$. What is the L-dimensional k-space Hamiltonian we get by stricting to an eigenspace of $\hat{T}_L$? Relate it back to what we found in (b) and (c). How is k related to Φ and ϑ? (Hint: $|mL+n\rangle = \hat{T}_L^m|n\rangle$)

While we explored this in a simple 1D model, the ideas here easily generalize to higher dimensions and with more complicated unit cells.

Problem 3. Constructing a "protected" edge state

In many models, we can pick a point in the Brillouin zone where we "expand" in small wave vectors and get effective "continuum" models. One such case is the SSH model when $|t_2 - t_1| \ll t_{1,2}$.

For this problem, we can begin with the k-space Hamiltonian

$$H(\mathbf{k}) = \begin{pmatrix} 0 & t_1 + t_2 e^{2ika}, \\ t_1 + t_2 e^{-2ika} & 0 \end{pmatrix}. \tag{5}$$

(a) In the limit of $|t_2 - t_1| \ll t_{1,2}$, what $\mathbf{k}$ vector has the smallest gap between energies? If this happens at $\mathbf{k}_0$, expand the Hamiltonian $H(\mathbf{k}_0 + \mathbf{q})$ for small $|\mathbf{q}|$, keeping only terms linear in $\mathbf{q}$.

(b) The above problem should take the form

$$h = v_{\mathrm{F}} q \sigma_y + m \sigma_x, \tag{6}$$

Find the energy eigenvalues and eigenvectors of this Hamiltonian and how they depend on q.

(c) To make this a continuum model let $q = -i\partial_x$. Further, let m depend on space such that

$$h = -i v_{\mathrm{F}} \sigma_y \partial_x + m_0 \tanh(x) \sigma_x. \tag{7}$$

Find a solution to this equation with zero eigenvalue when $m_0 > 0$ and when $m_0 < 0$; plot its magnitude in space. Finally, relate m_0 back to t_2 and t_1; what is the difference between $x < 0$ and $x > 0$ in the original chain? This continuum solution is due to the "domain wall" in the mass m and is due to Jackiw and Rebbi. It is a topologically protected edge state.

Problem 4. An intro to PythTB.py

We will use this software package to explore tight-binding models, you can find it at `https://www.physics.rutgers.edu/pythtb/`. To install it on your local machine, you will need python, which can be found at `https://conda.io`. Once python is installed (either 2.7 or 3.x), you can install PythTB at the command prompt (terminal) with

```
pip install pythtb --upgrade
```

or if you do not have root permissions

```
pip install pythtb --upgrade --user
```

You will also need some standard python packages like NUMPY (numerics) and MATPLOTLIB (plotting). This problem is to help get you acquainted with this python package. Download `benzene.py` from the course website (or see code on the next page). If you need any help, please do not hesistate to contact me.

In this example, we will look at the tight-binding model of benzene, which is made with p_z orbitals of Carbon atoms. The Hamiltonian is very much like we considered in Problem 2

$$H = E_p \sum_j |j\rangle \langle j| + t \left(\sum_{j=0}^{5} |j+1\rangle \langle j| + \text{h.c.} \right), \quad |6\rangle = |0\rangle . \tag{8}$$

However, these atoms exist in a two-dimensional plane at points $\mathbf{r}_j = r(\cos(j\pi/3), \sin(j\pi/3))$. We will need this for considering external fields.

(a) Identify the parameters in the code, and run the code. It should output the energies and eigenvectors (real part of them). Do these match what you found in Problem 2a?

(b) Now, we mess with the code! Modify it to have a uniform electric field in the $\hat{x}$ or $\hat{y}$ direction (your choice) by raising or lowering site energies in a way that is linear in their spatial coordinates. Make a plot of the six eigenvalues versus electric field strength. Does the behavior make sense? Explain why.

(c) Instead of an external field, now modify model by changing the hopping strength alternate between bonds such that hopping between 0-1, 2-3, and 4-5 have strength $t+\delta$ and 1-2, 3-4, and 5-0 have stengths $t-\delta$. Make a plot of the six eigenvalues versus δ. Does anything special happen at $\delta = t$? Explain why?

BENZENE.PY – From Berry Phases in Electronic Structure Theory, Appendix D.2

```
#!/usr/bin/env python
from __future__ import print_function # python3 style print

# ------------------------------------------------------------------------
# Tight-binding model for p_z states of benzene molecule
# ------------------------------------------------------------------------

from pythtb import *

# set up molecular geometry
lat=[[1.0,0.0],[0.0,1.0]]          # define coordinate frame: 2D Cartesian
r=1.2                              # distance of atoms from center
orb=np.zeros((6,2),dtype=float)    # initialize array for orbital positions
for i in range(6):                 # define coordinates of orbitals
  angle=i*np.pi/3.0
  orb[i,:]= [r*np.cos(angle), r*np.sin(angle)]

# set site energy and hopping amplitude, respectively
ep=-0.4
t=-0.25

# define model
my_model=tbmodel(0,2,lat,orb)
my_model.set_onsite([ep,ep,ep,ep,ep,ep])
my_model.set_hop(t,0,1)
my_model.set_hop(t,1,2)
my_model.set_hop(t,2,3)
my_model.set_hop(t,3,4)
my_model.set_hop(t,4,5)
my_model.set_hop(t,5,0)

# print model
my_model.display()

# solve model and print results
(eval,evec)=my_model.solve_all(eig_vectors=True)

# print results, setting numpy to format floats as xx.xxx
np.set_printoptions(formatter={'float': '{: 6.3f}'.format})
# Print eigenvalues and real parts of eigenvectors, one to a line
print("  n    eigval    eigvec")
```

```
for n in range(6):
    print(" %2i   %7.3f  " % (n,eval[n]), evec[n,:].real)
```