

PythTB for (topological) tight-binding models

Jennifer Cano



PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Download Python codes here: <http://samos.martech.fsu.edu/TWS/default.htm>

What is PythTB?

- PythTB is a software package providing a Python implementation for tight-binding models.
- Developed by Sinisa Coh and David Vanderbilt

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Making tight-binding models is easy. Why should I use PythTB?

- Only work in real space.
- Easily compute band structure and get eigenvectors.
- Easily create slab, cube, or other finite boundary conditions.
- Easily compute Berry phase or plot Wilson loop eigenvalues.

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Goals in PythTB

- Plot band structures
- Verify topological phases by
 - Visualizing topological surface states
 - Plotting Berry phase

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Tight-binding degrees of freedom

Atoms located at positions in unit cell: \mathbf{r}_α

Each atom has orbitals labelled by $j = 1, \dots, n$

An atom has infinitely many orbitals – to model a physical system, need to decide which atoms/orbitals are relevant!

$$\phi_{\mathbf{R},\alpha,j}(\mathbf{r}) = \varphi_{\alpha,j}(\mathbf{r} - \mathbf{R} - \mathbf{r}_\alpha)$$

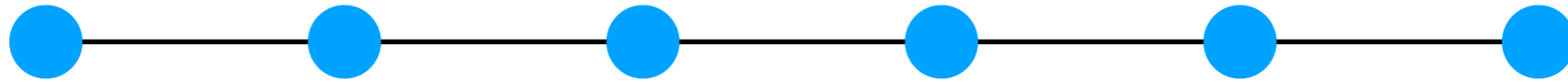
unit cell atom orbital

assume orthonormality: $\langle \phi_{\mathbf{R},\alpha,j} | \phi_{\mathbf{R}',\beta,i} \rangle = \delta_{\mathbf{R},\mathbf{R}'} \delta_{\alpha,\beta} \delta_{ij}$

Hamiltonian consists of “hopping terms”

$$H_{\alpha i, \beta j}(\mathbf{R}) \equiv \langle \phi_{\mathbf{R}',\alpha i} | H | \phi_{\mathbf{R}'+\mathbf{R},\beta j} \rangle = \langle \phi_{\mathbf{0},\alpha i} | H | \phi_{\mathbf{R},\beta j} \rangle$$

Example 1: atoms in 1d

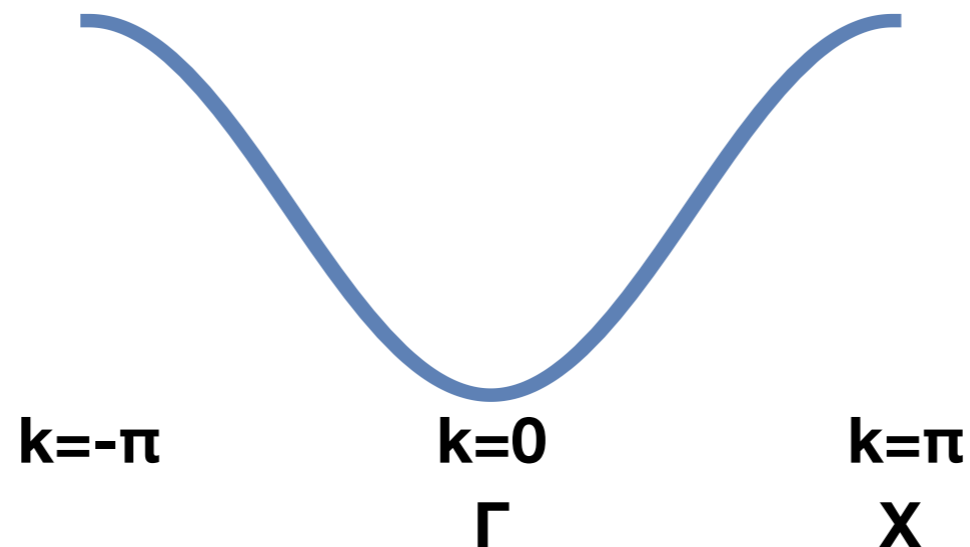


1 orbital/site \Rightarrow trivial subscripts: $\mathbf{r}_\alpha = \mathbf{r}_1 = \mathbf{0}$ $i = 1$

$$H(\mathbf{R} = \pm \hat{x}) = -t$$

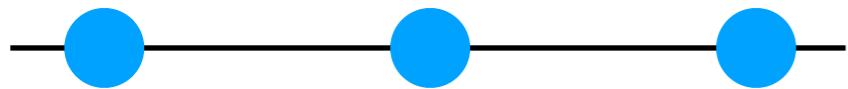
$$H(\mathbf{R} \neq \pm \hat{x}) = 0$$

$$H^{\mathbf{k}} = \sum_{\mathbf{R}} e^{i\mathbf{k} \cdot \mathbf{R}} H(\mathbf{R}) = -t(e^{i\mathbf{k} \cdot \hat{x}} + e^{-i\mathbf{k} \cdot \hat{x}}) = -2t \cos k$$



$$H(\mathbf{R} = \pm \hat{x}) = -t$$

$$H(\mathbf{R} \neq \pm \hat{x}) = 0$$



adaptation from "simple example"

at <http://physics.rutgers.edu/pythtb/examples.html>

Example 1: atoms in 1d

How to define model

specify model

lattice vectors

Lattice vectors (we only have one)

```
lat=[[1.0]]
```

positions of orbitals

Orbitals in units of lattice vecs
(we only have one)

```
orb=[[0.0]]
```

define the model

Define model: (dim k space, dim real
space, lattice vecs, orbital vecs)

```
my_model=tb_model(1,1,lat,orb)
```

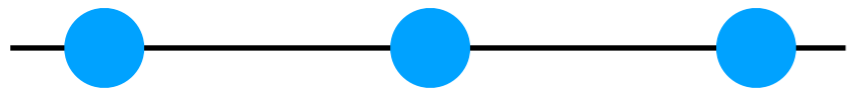
assign hopping terms

Hopping term: (amplitude, $i\alpha$, $j\beta$, \mathbf{R})

```
my_model.set_hop(-1., 0, 0, [1])
```

$$H(\mathbf{R} = \pm \hat{x}) = -t$$

$$H(\mathbf{R} \neq \pm \hat{x}) = 0$$



k path to plot in units of
reciprocal lattice vecs ($-\pi$ to π)

Labels of k-points on path

Repeat this line for more
bands, up to evals[n]

Where to put ticks (from path
and numsteps)

Example 1: atoms in 1d

How to plot

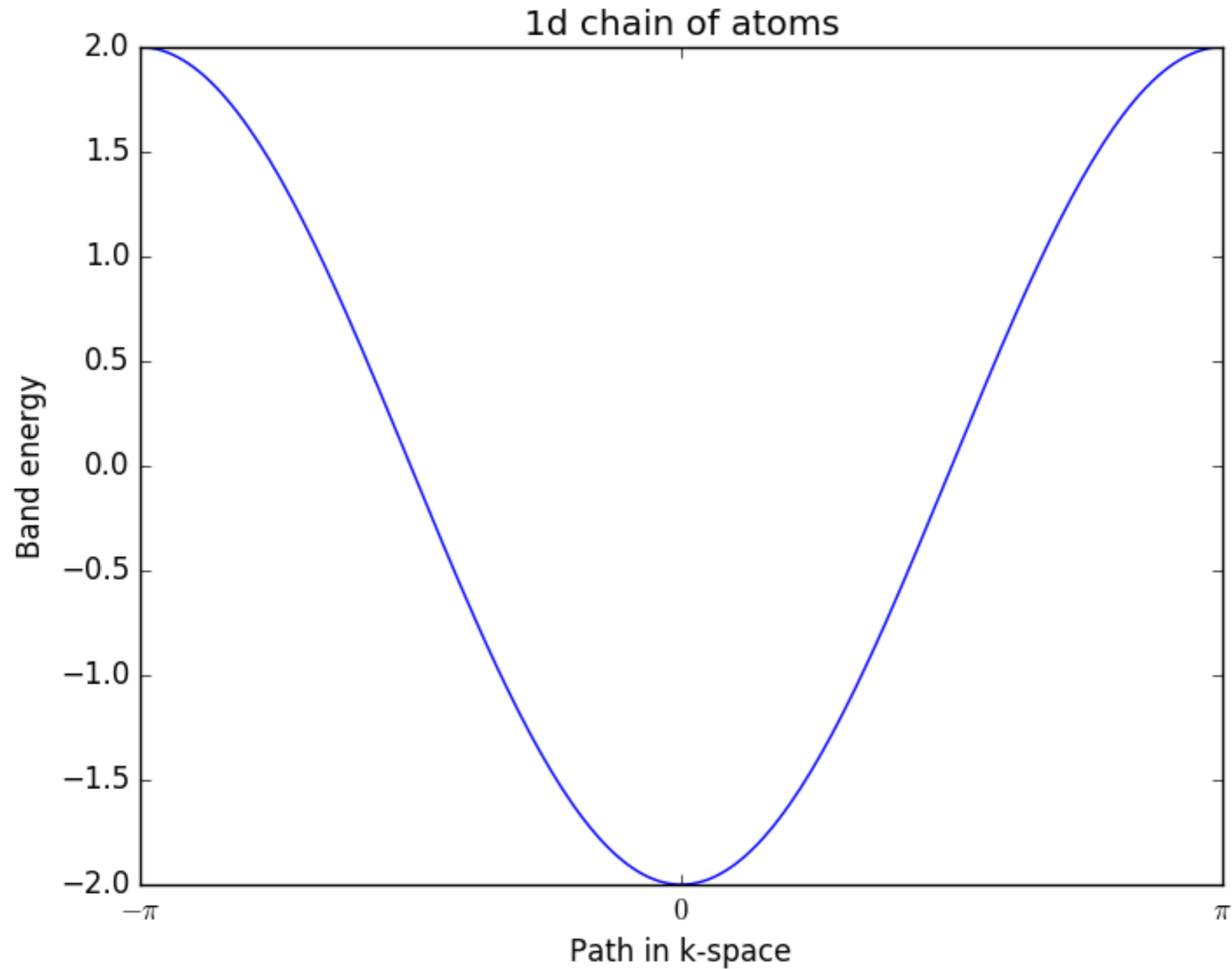
```
# define a path in k-space to plot
path=[[-.5],[0],[.5]]
# label k points
label=(r'$-\pi$',r'$0$',r'$\pi$')
# number of steps between points
numsteps=100
kpts=k_path(path,numsteps)

# solve model
evals=my_model.solve_all(kpts)

# make a figure object
fig=plt.figure()
# plot bands
plt.plot(evals[0])
# put title on top
plt.title("1d chain of atoms")
plt.xlabel("Path in k-space")
plt.ylabel("Band energy")
plt.xticks([0,100,200],label)
```

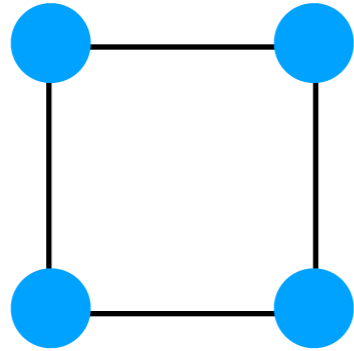
PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Example 1: atoms in 1d



PythTB is based at: <http://physics.rutgers.edu/pythtb/>

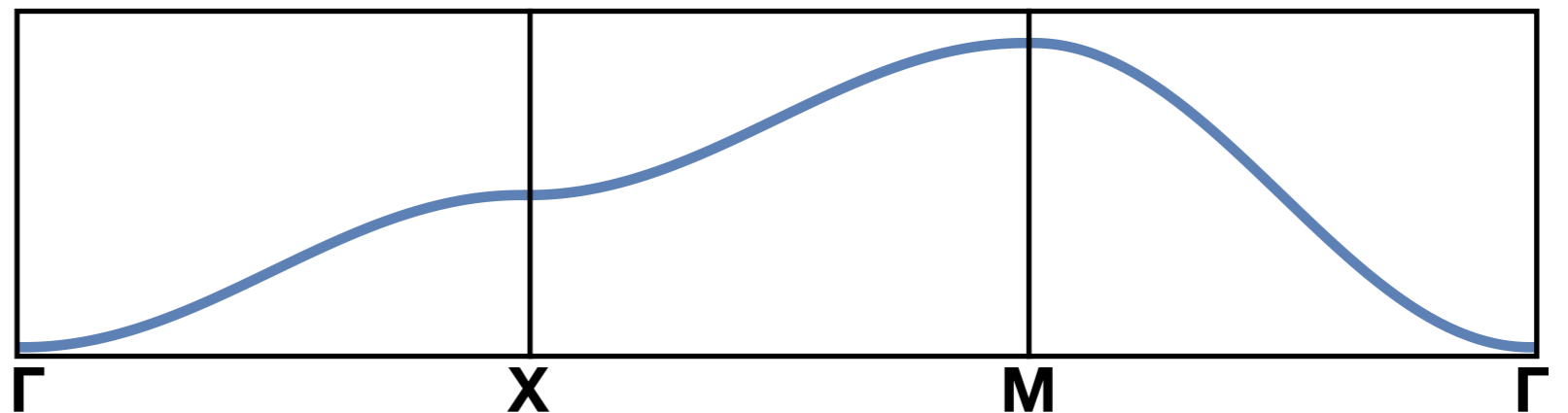
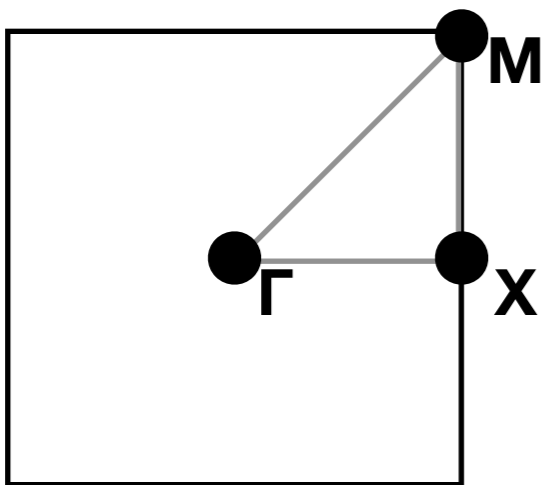
Example 2: square lattice

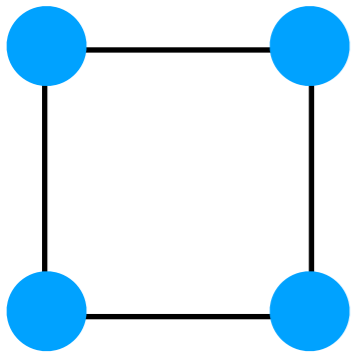


$$H(\mathbf{R}) = \begin{cases} -t & \text{if } \mathbf{R} = \pm\hat{x}, \pm\hat{y} \\ 0 & \text{else} \end{cases}$$

$$H^{\mathbf{k}} = -2t(\cos k_x + \cos k_y)$$

How to plot 2d spectrum? identify high-symmetry path



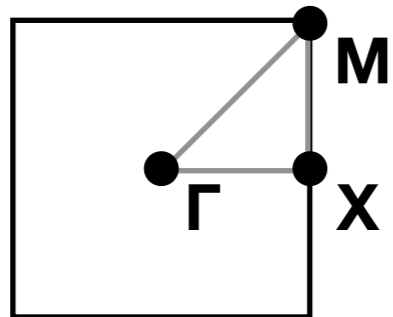


Example 2: square lattice

$$H(\mathbf{R}) = \begin{cases} -t & \text{if } \mathbf{R} = \pm\hat{x}, \pm\hat{y} \\ 0 & \text{else} \end{cases}$$

Two two-component lattice vecs

Still one orbital;
two components



k path: Γ , X, M, Γ

```
# specify model
# lattice vectors
lat=[[1.0,0.0],[0.0,1.0]]
# positions of orbitals
orb=[[0.0,0.0]]

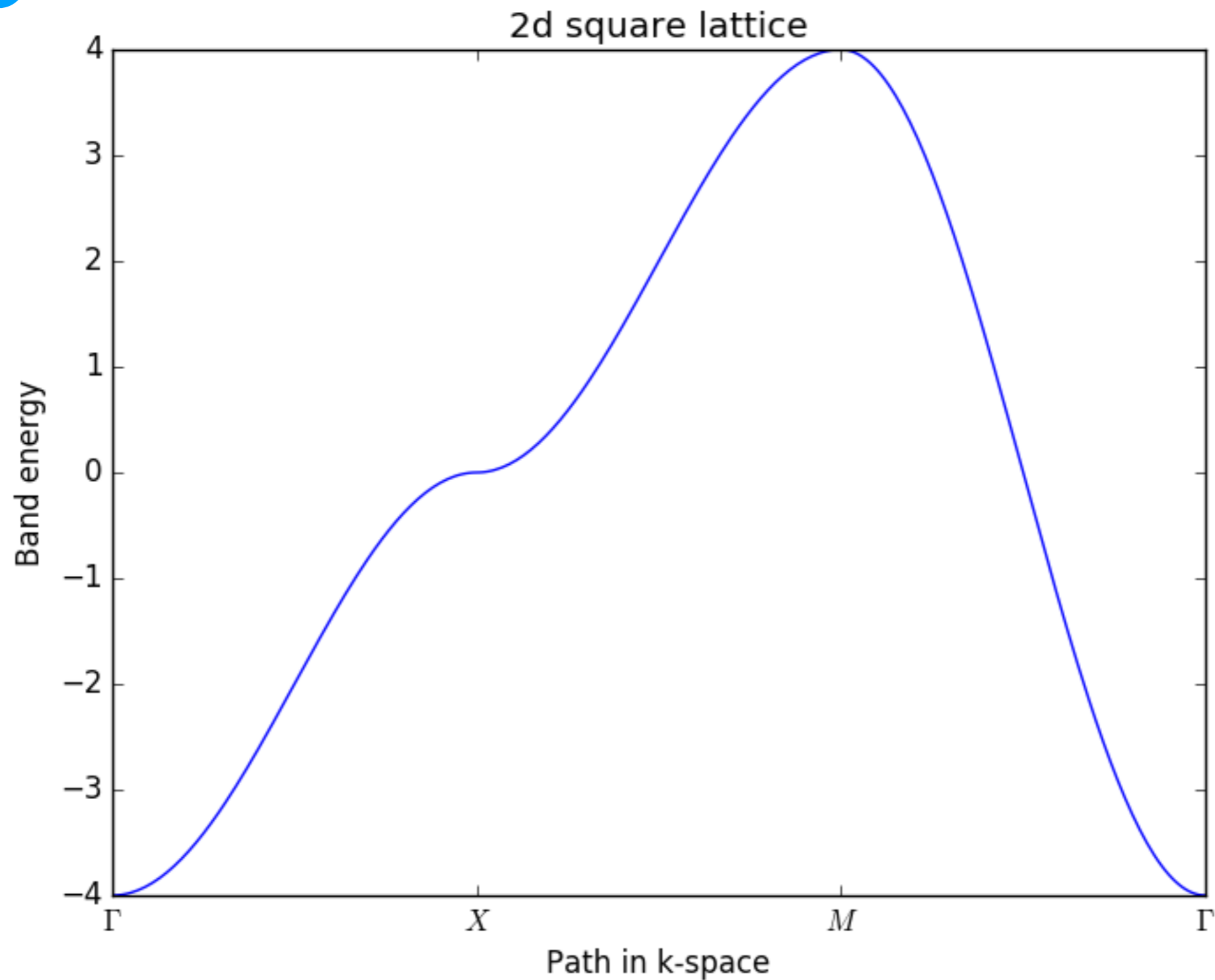
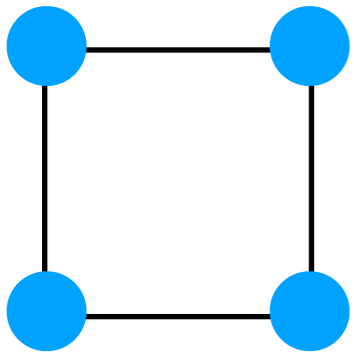
# define the model
my_model=tb_model(2,2,lat,orb)
# assign hopping terms
# x-hopping
my_model.set_hop(-1., 0, 0, [1.0,0])
# y-hopping
my_model.set_hop(-1., 0, 0, [0,1.0])

# define a path in k-space to plot
path=[[0.0,0.0],[.5,0],[.5,.5],[0.0,0.0]]
```

Another tick because more k pts `pl.xticks([0,100,200,300],label)`

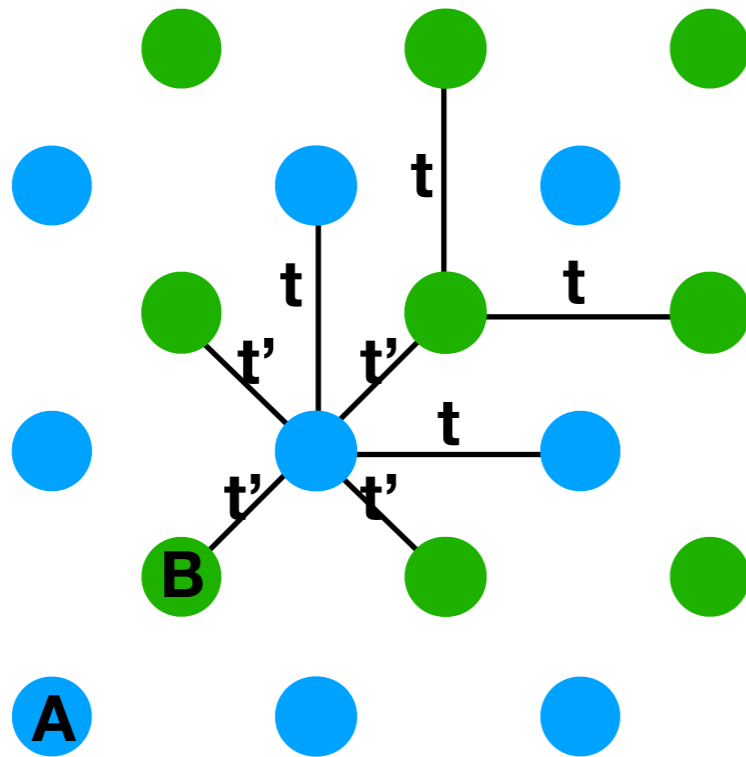
PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Example 2: square lattice



PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Example 3: interpenetrating square lattices



$$\mathbf{r}_A = (0, 0)$$

$$\mathbf{r}_B = (1/2, 1/2)$$

only one orbital/site $\Rightarrow i = 1$

$$H_{AA}(\mathbf{0}) = \mu_A$$

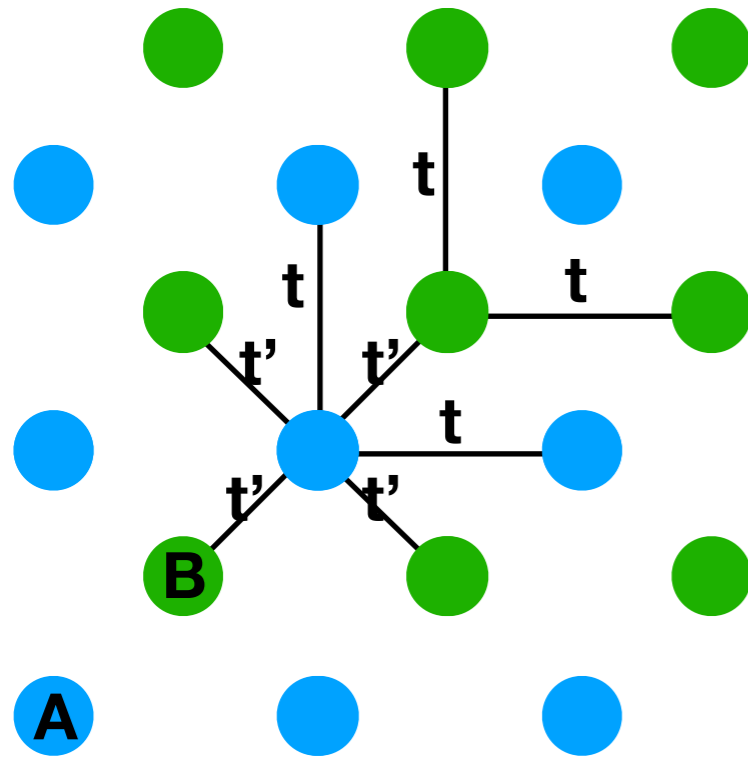
$$H_{BB}(\mathbf{0}) = \mu_B$$

$$H_{AA}(\pm\hat{x}) = H_{AA}(\pm\hat{y}) = H_{BB}(\pm\hat{x}) = H_{BB}(\pm\hat{y}) = -t$$

$$H_{AB}(\mathbf{0}) = H_{AB}(-\hat{x}) = H_{AB}(-\hat{x} - \hat{y}) = H_{AB}(-\hat{y}) = -t'$$

label by \mathbf{R} (unit cell),
not \mathbf{r}_{AB}

Example 3: interpenetrating square lattices



A sublattice hopping

B sublattice hopping

Inter-sublattice hopping
(last term is R)

Now plotting two bands

```
# specify model
# lattice vectors
lat=[[1.0,0.0],[0.0,1.0]]
# positions of orbitals
orb=[[0.0,0.0],[.5,.5]]

# define the model
my_model=tb_model(2,2,lat,orb)

# assign onsite energy
my_model.set_onsite([1.0,-1.0])

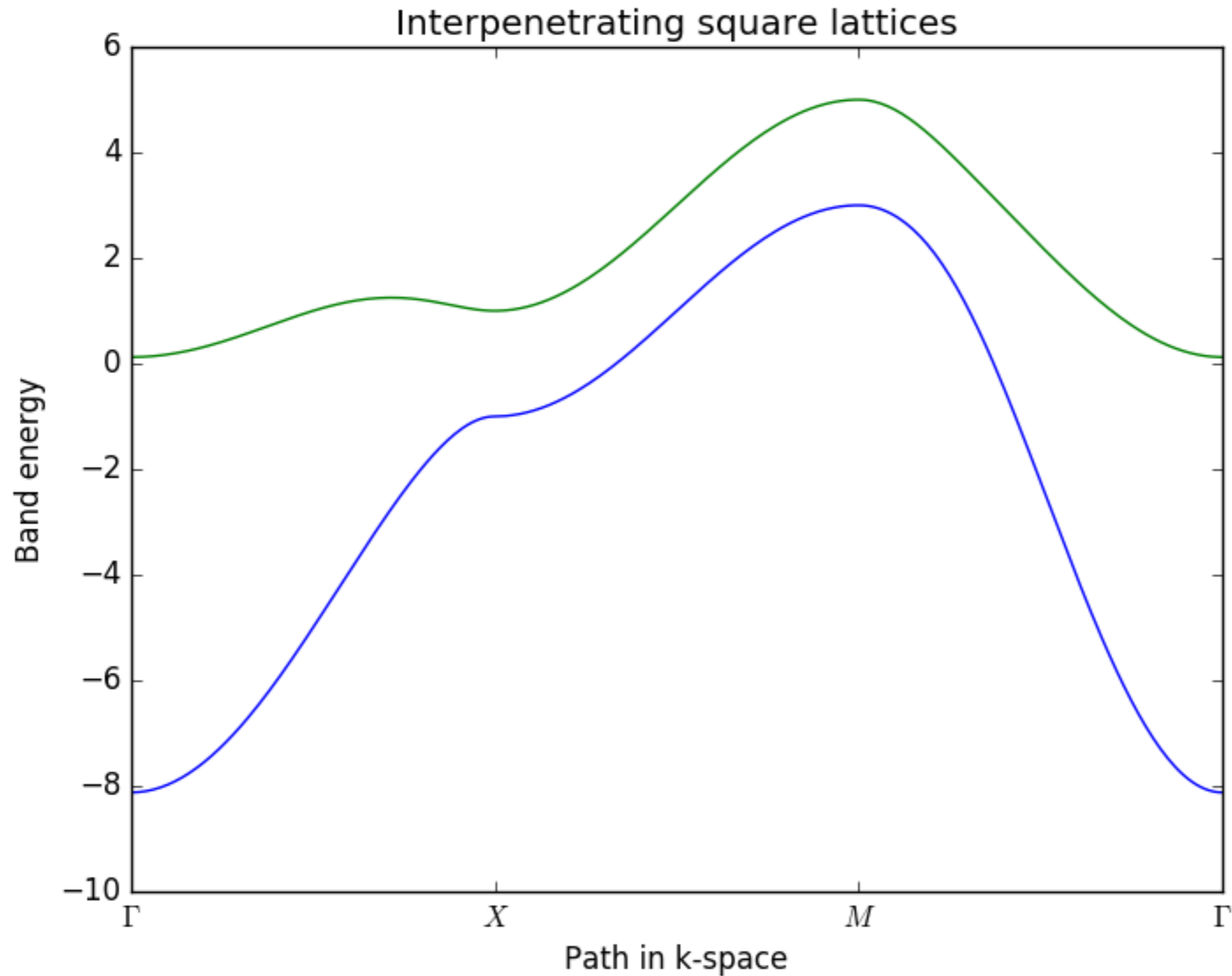
# assign hopping terms
t=1.0
t2=1.0
# x-hopping within sublattice of orbital "0"
my_model.set_hop(-t, 0, 0, [1.0,0])
# y-hopping within sublattice of orbital "0"
my_model.set_hop(-t, 0, 0, [0,1.0])
# x-hopping within sublattice of orbital "1"
my_model.set_hop(-t, 1, 1, [1.0,0])
# y-hopping within sublattice of orbital "1"
my_model.set_hop(-t, 1, 1, [0,1.0])
# four inter-sublattice hopping terms, from "0" to "1"
my_model.set_hop(-t2, 0, 1, [0.0,0.0])
my_model.set_hop(-t2, 0, 1, [-1.0,0.0])
my_model.set_hop(-t2, 0, 1, [-1.0,-1.0])
my_model.set_hop(-t2, 0, 1, [0.0,-1.0])

# plot bands
pl.plot(evals[0])
pl.plot(evals[1])
```

Two orbitals!

Onsite energy
for each orbital

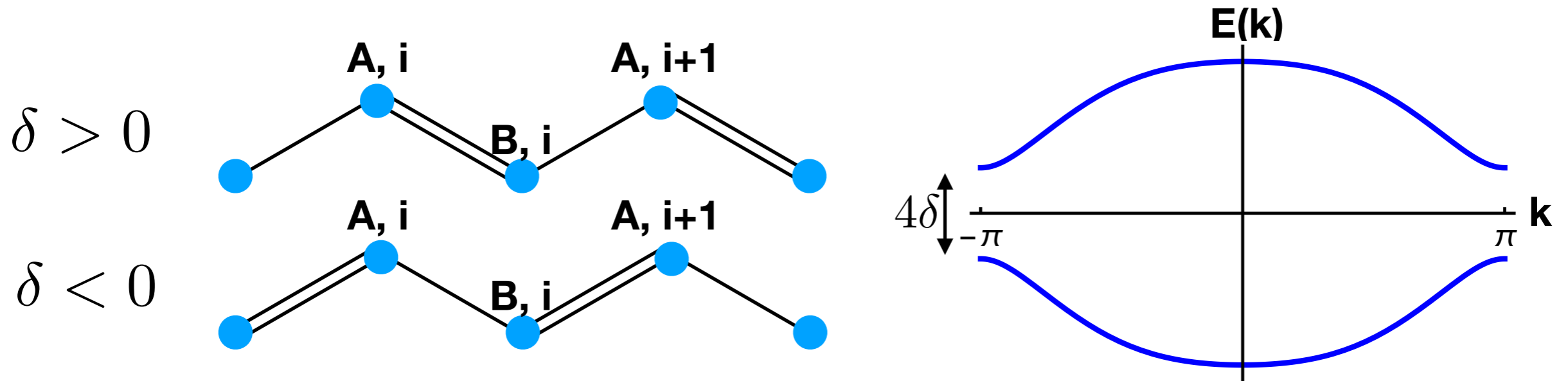
Example 3: interpenetrating square lattices



PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Exercise 1: Implement SSH model

$$H = \sum_i (t + \delta) c_{A,i}^\dagger c_{B,i} + (t - \delta) c_{A,i+1}^\dagger c_{B,i} + \text{h.c.}$$



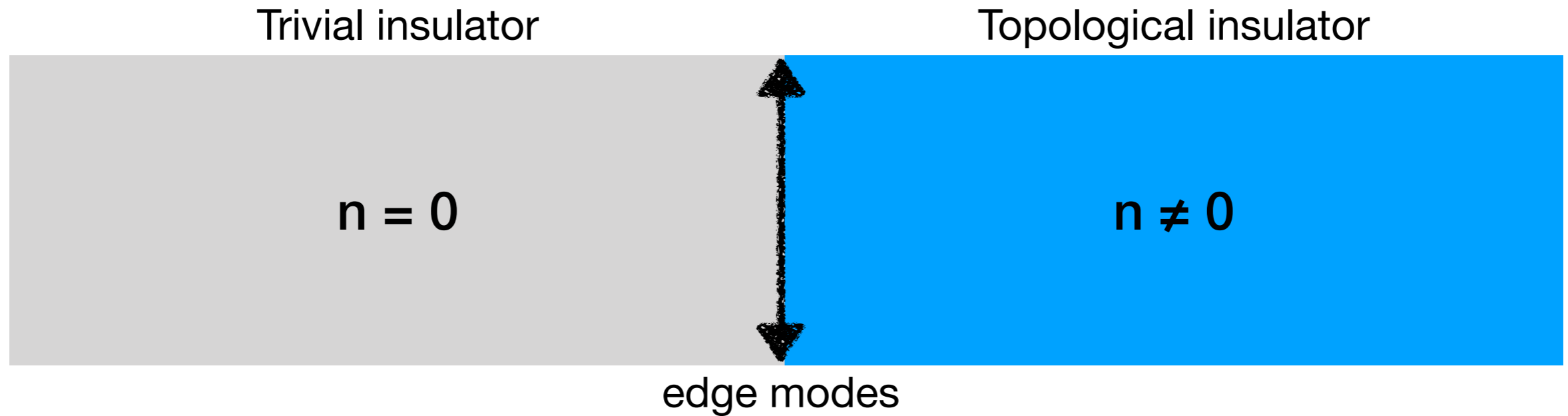
a) reproduce band structure

b) verify Berry phase differs by π when δ changes sign

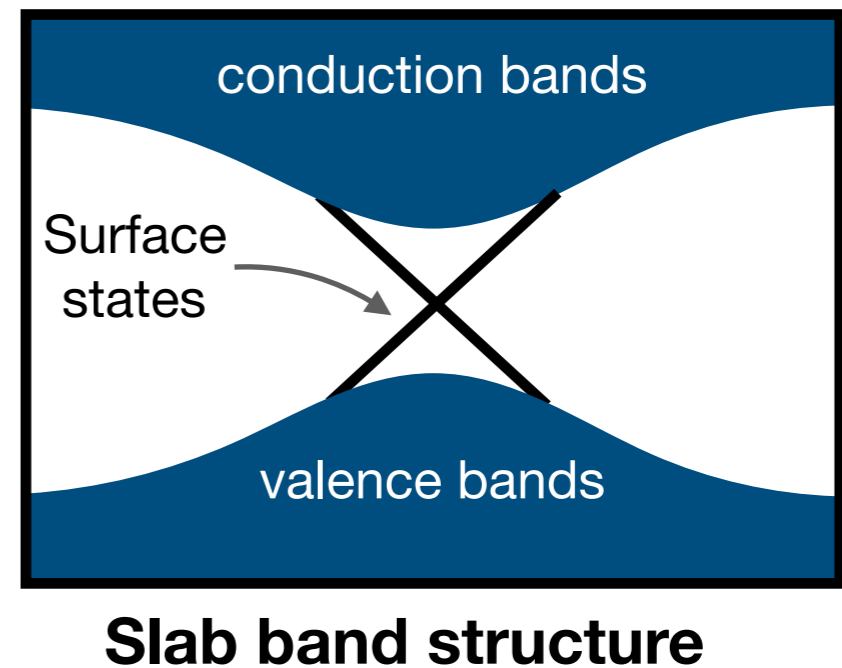
**Berry phase
code snippet:**

```
# Construct wf_array  
# capable of storing 11 wavefunctions  
wf = wf_array(my_model, [11])  
# populate this wf_array with regular  
# grid of points in BZ centered at 0.0  
wf.solve_on_grid([0.0])  
# compute Berry phase of lowest band  
pha = wf.berry_phase([0])
```


Bulk-boundary correspondence

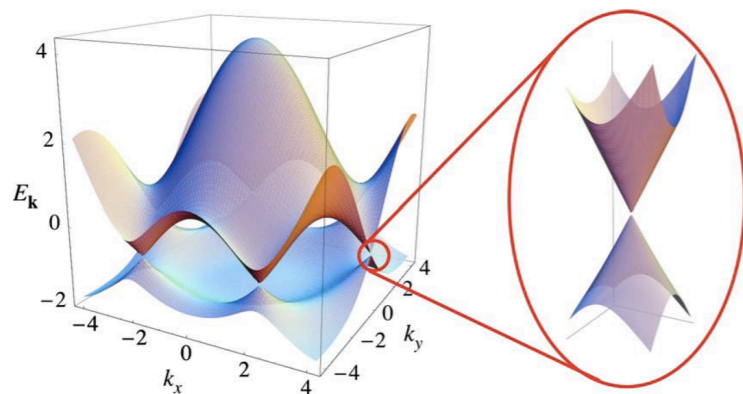


Quantized topological invariant
⇒ gapless edge states



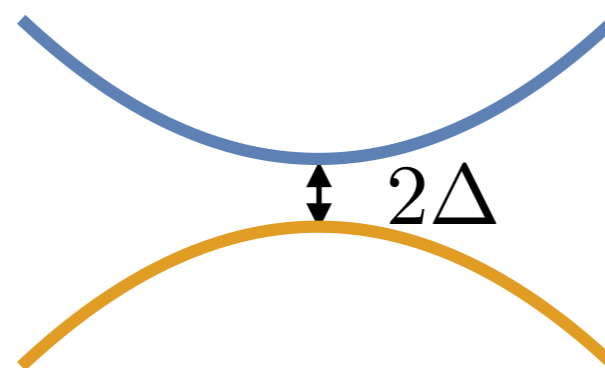
Kane-Mele model

- Recall: the Kane-Mele model is an example of a 2D topological phase protected by time-reversal symmetry (ref: Kane and Mele, PRL 95, 146802 (2005), Eq. (1))
- Sample file: <http://physics.rutgers.edu/pythtb/examples.html>
- We will verify topological nature by plotting surface states



$$\mathcal{H}_0 = -i\hbar v_F \psi^\dagger (\sigma_x \tau_z \partial_x + \sigma_y \partial_y) \psi$$

$\sigma_z = \text{sublattice}$ $\tau_z = \text{valley}$ $s_z = \text{spin}$



open gap:
 $\Delta_{\text{SOC}} \sigma_z \tau_z s_z$

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Kane-Mele model: edge states

New features:

```
# cut into a slab
```

```
fin_model=my_model.cut_piece(numbands, 0, glue_edges=False)
```

layers



real space direction
that is finite in slab

```
# path in surface BZ
```

```
fin_path=k_path([[0.], [.5], [1.0]], numpoints)
```

Number of bands is (# orbitals) x (# layers)

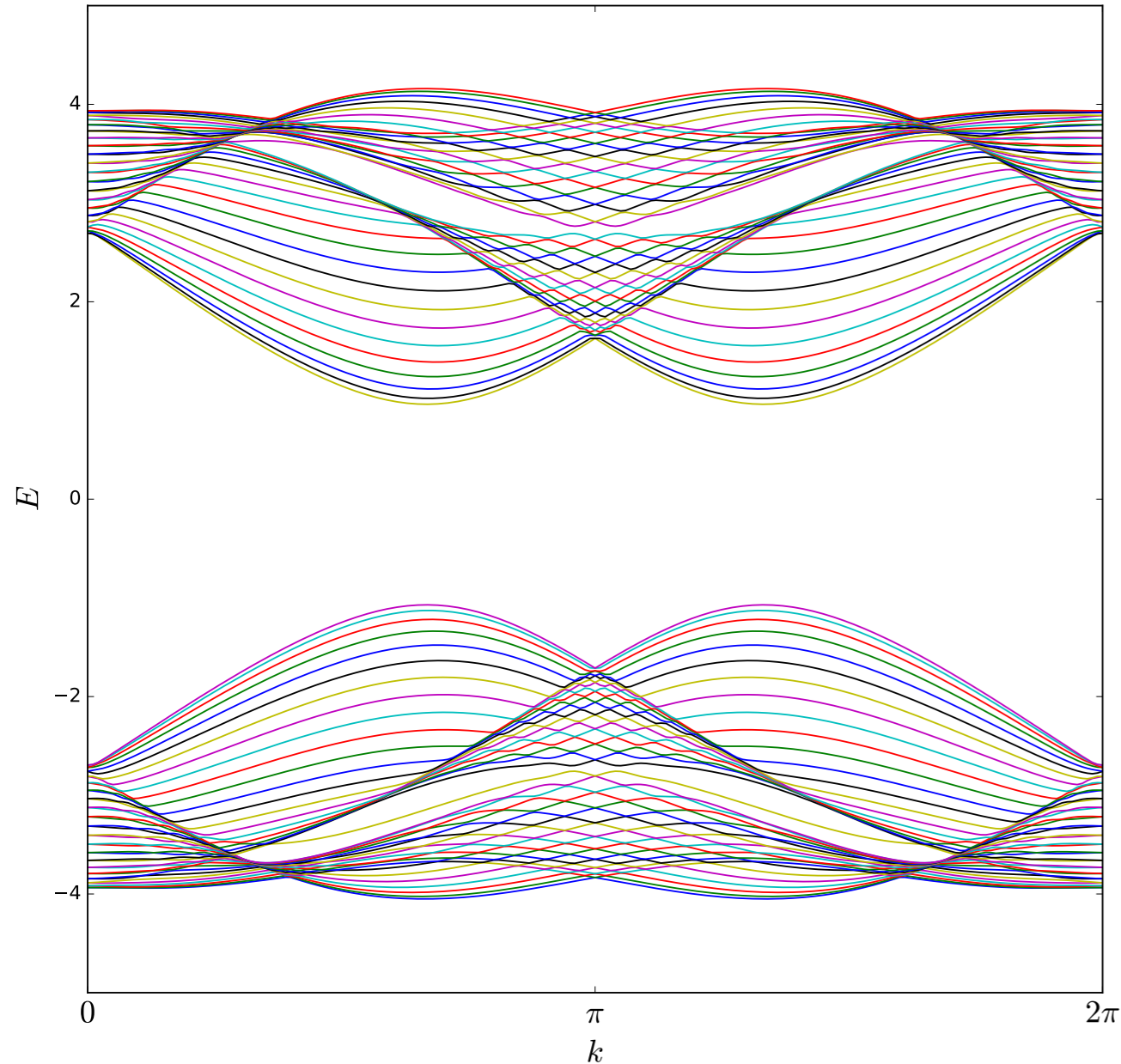
```
for i in range(4*numbands): ## 4 is the number of orbitals  
    pl.plot(fin_evals[i], linewidth=1.0)
```

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

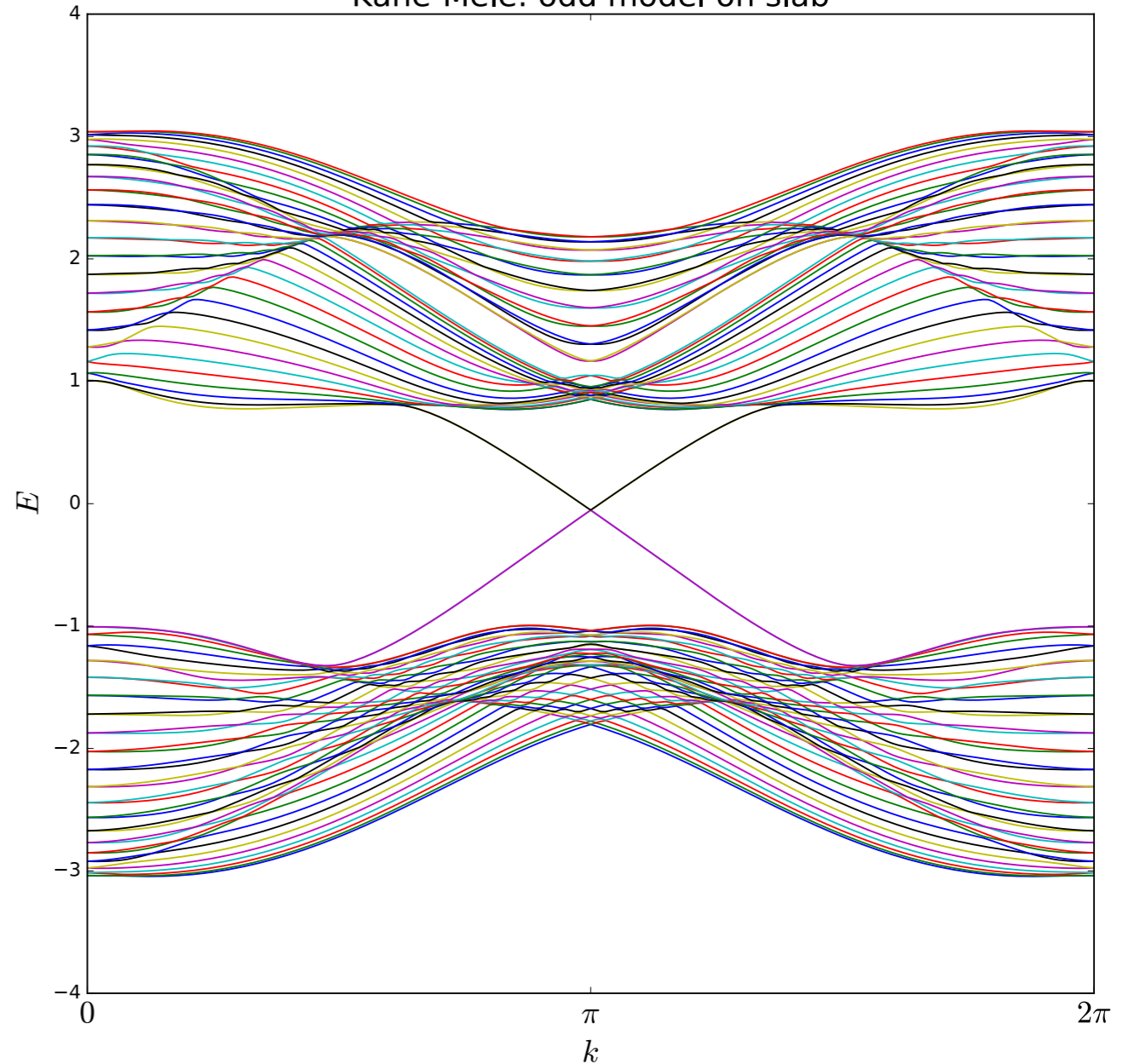
Kane-Mele model: edge states

Output:

Kane-Mele: even model on slab



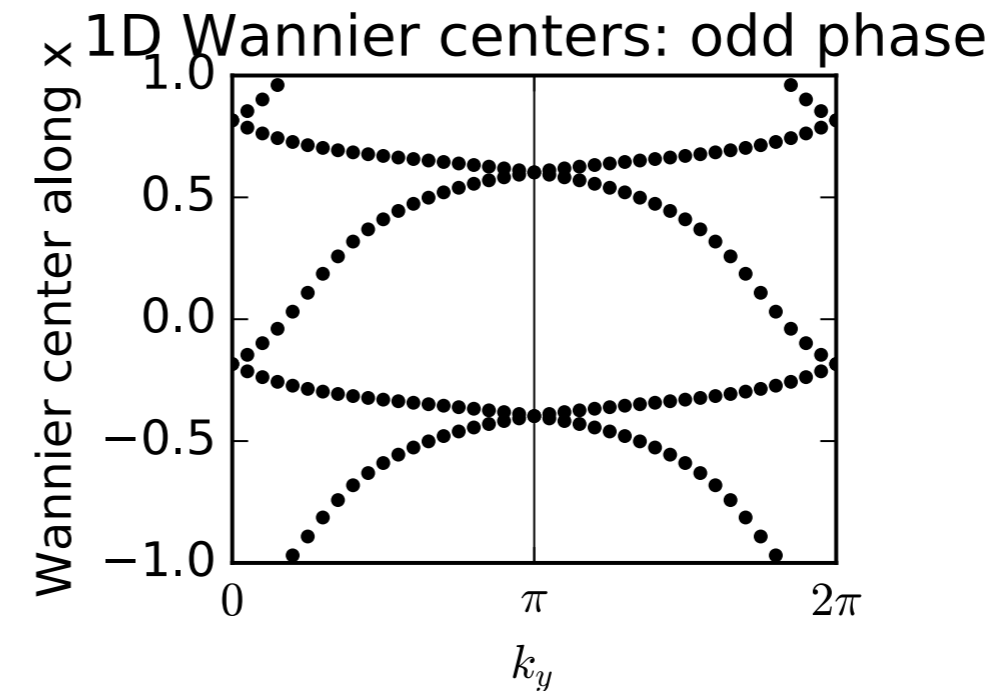
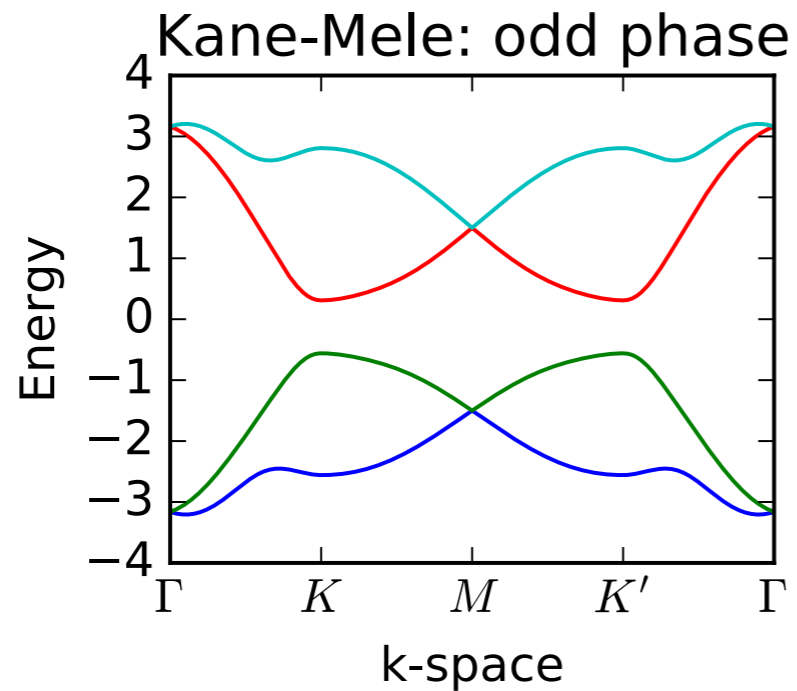
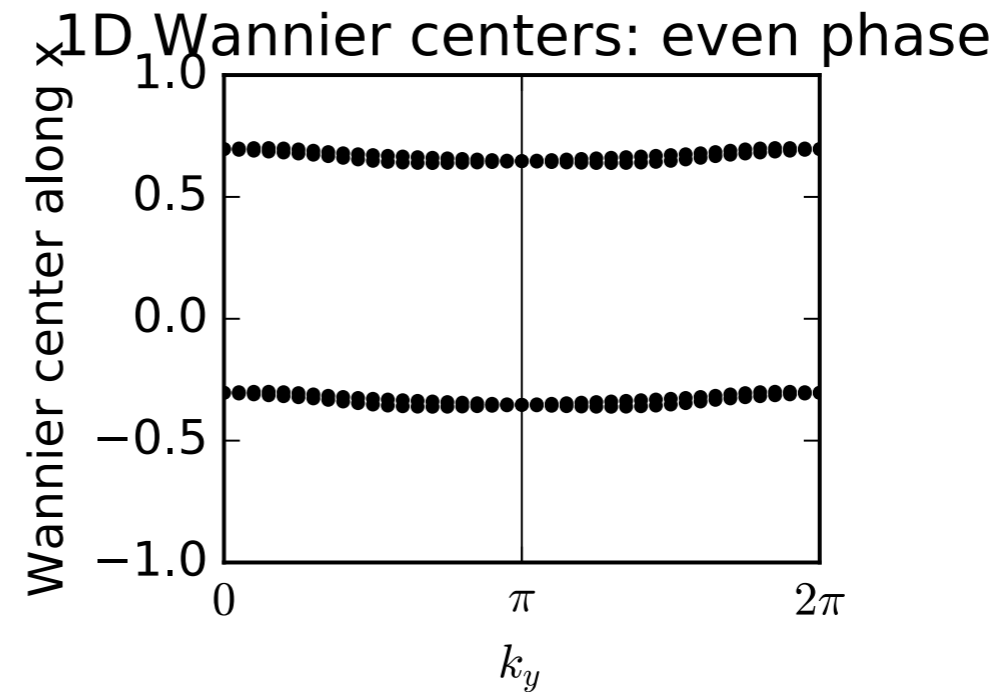
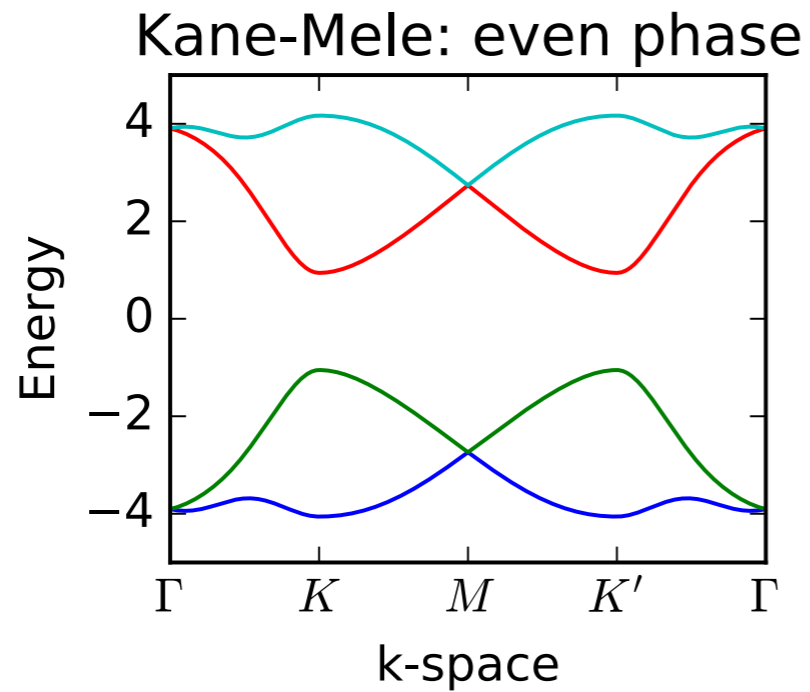
Kane-Mele: odd model on slab



Would get same result for any slab termination

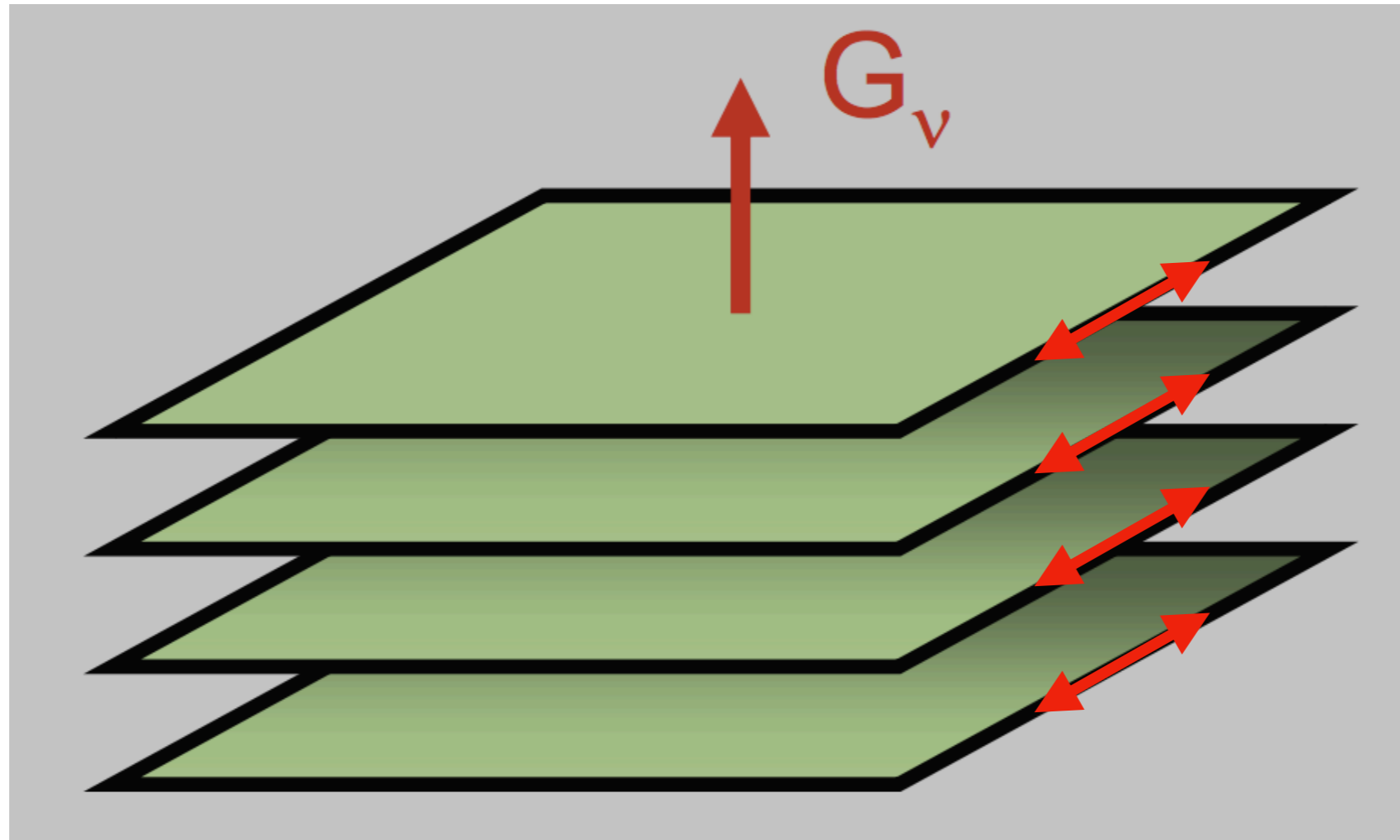
PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Kane-Mele model: winding Berry phase



“Weak” topological insulator from stack of 2D TIs

Ref: Fu, Kane, Mele Phys. Rev. Lett. 98, 106803 (2007)



2 Dirac cones on
side surfaces

Figure: Charlie Kane's, Windsor Summer School slides:
<http://www.physics.upenn.edu/~kane/>

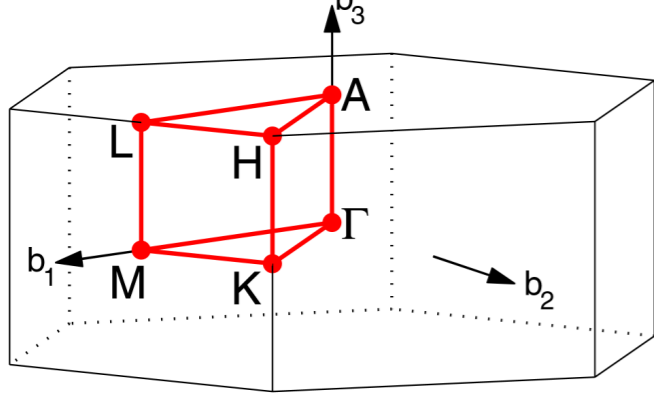
Stacked Kane-Mele

- Add a third dimension (all vectors get a third component!)
- Couple layers (otherwise bands are flat)
- Many ways to couple the layers: I recommend the following coupling term that breaks spin conservation:

```
# add coupling in z direction  
zhop=.1*0  
zsoc=.3*spin_orb  
ret_model.set_hop(-j*zsoc*sigma_z, 0, 0, [ 0., 0., 1.])  
ret_model.set_hop(j*zsoc*sigma_z, 1, 1, [ 0., 0., 1.])
```

(A real term preserves an anti-unitary symmetry that flattens the Dirac cones; details: ArXiv: 1410.4440)

PythTB is based at: <http://physics.rutgers.edu/pythtb/>

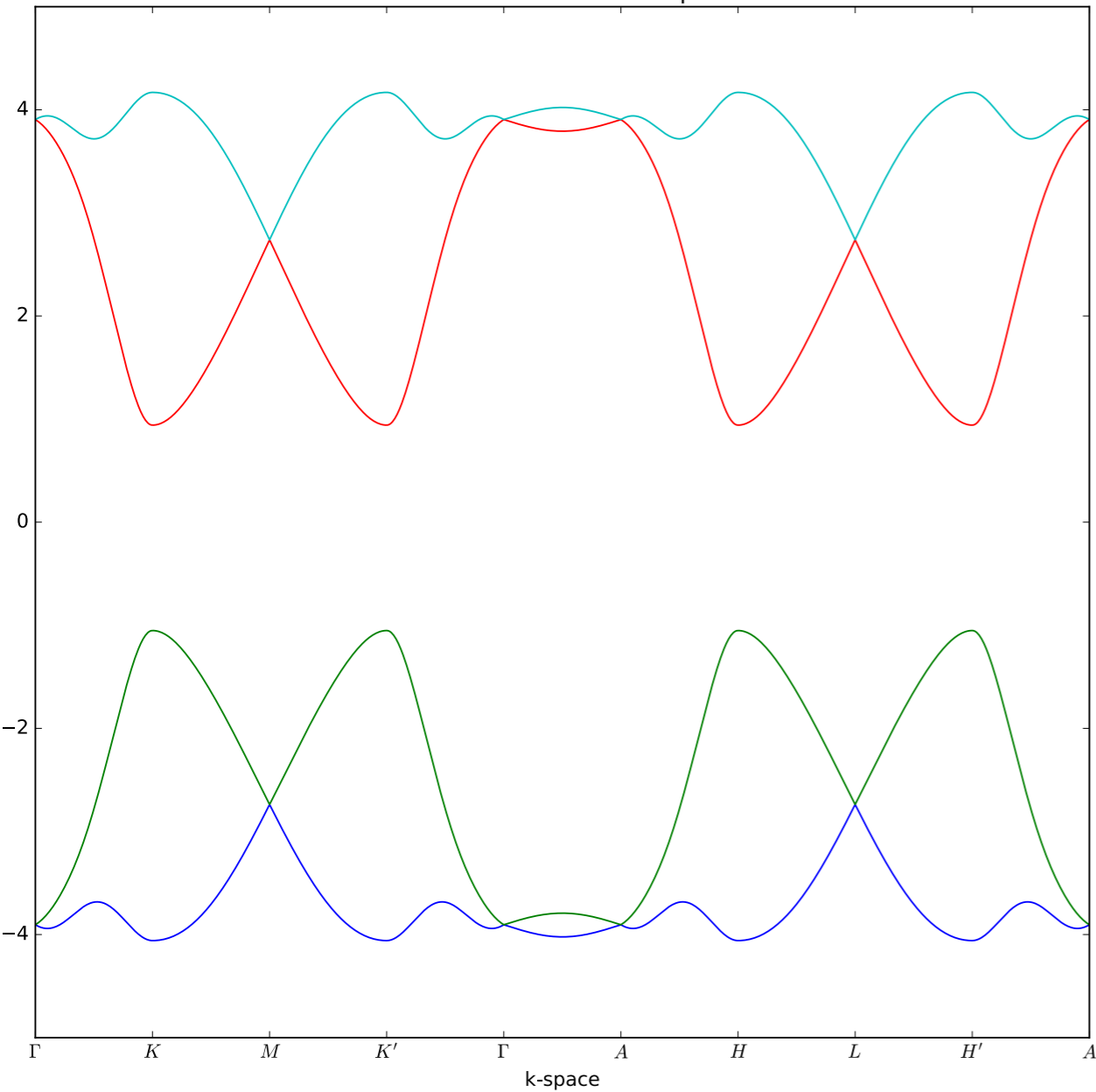


Stacked Kane-Mele

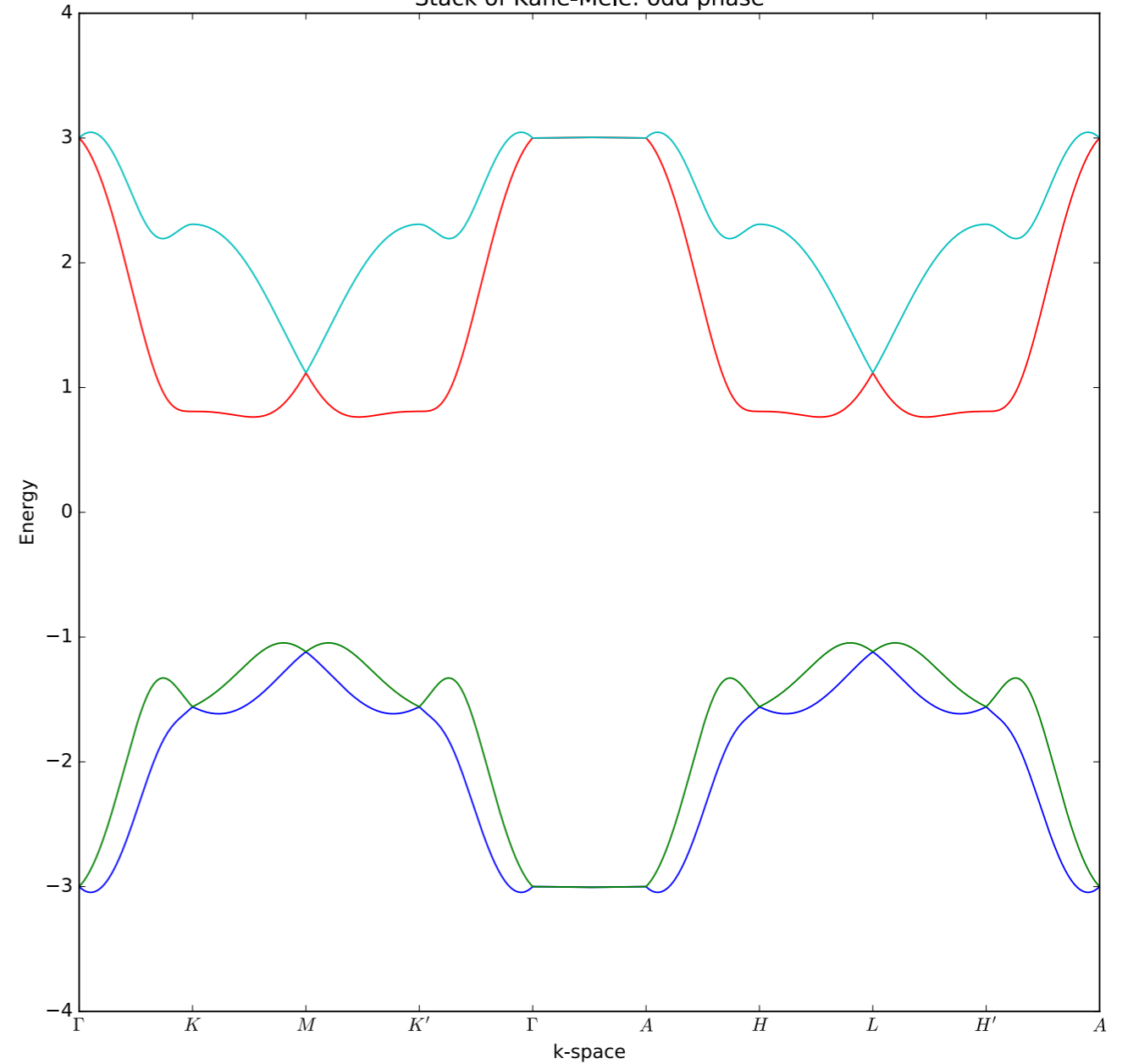
HEX path: Γ -M-K- Γ -A-L-H-A|L-M|K-H

[Setyawan & Curtarolo, DOI: 10.1016/j.commatsci.2010.05.010]

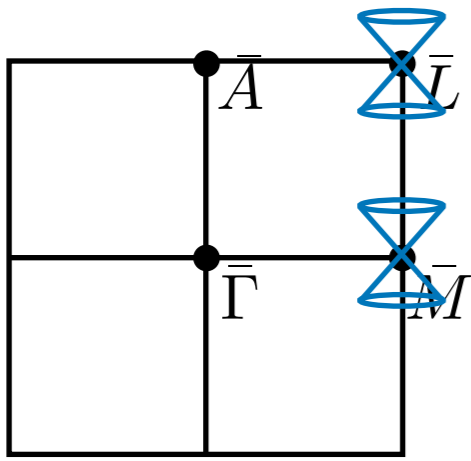
Stack of Kane-Mele: even phase



Stack of Kane-Mele: odd phase

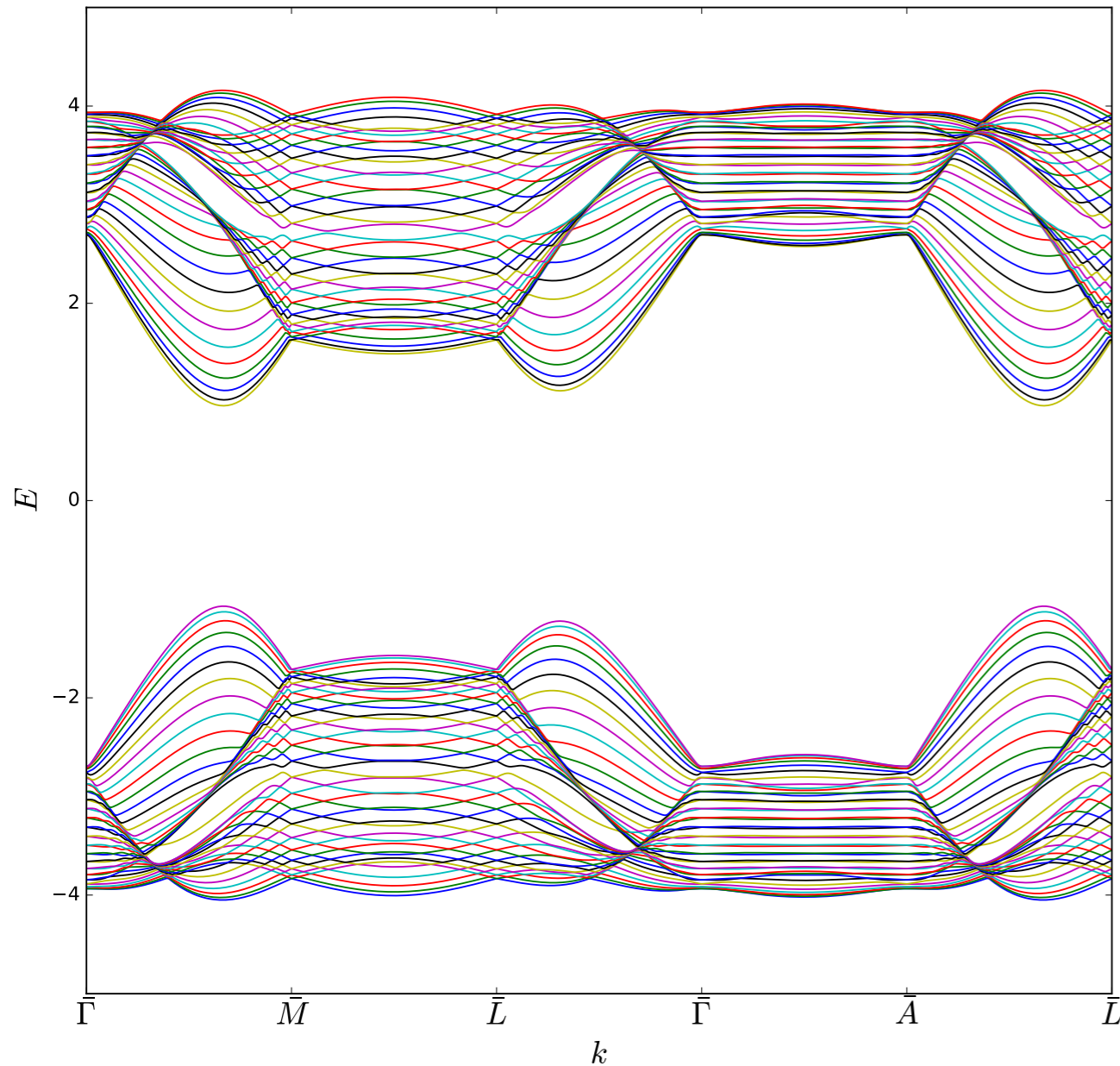


PythTB is based at: <http://physics.rutgers.edu/pythtb/>

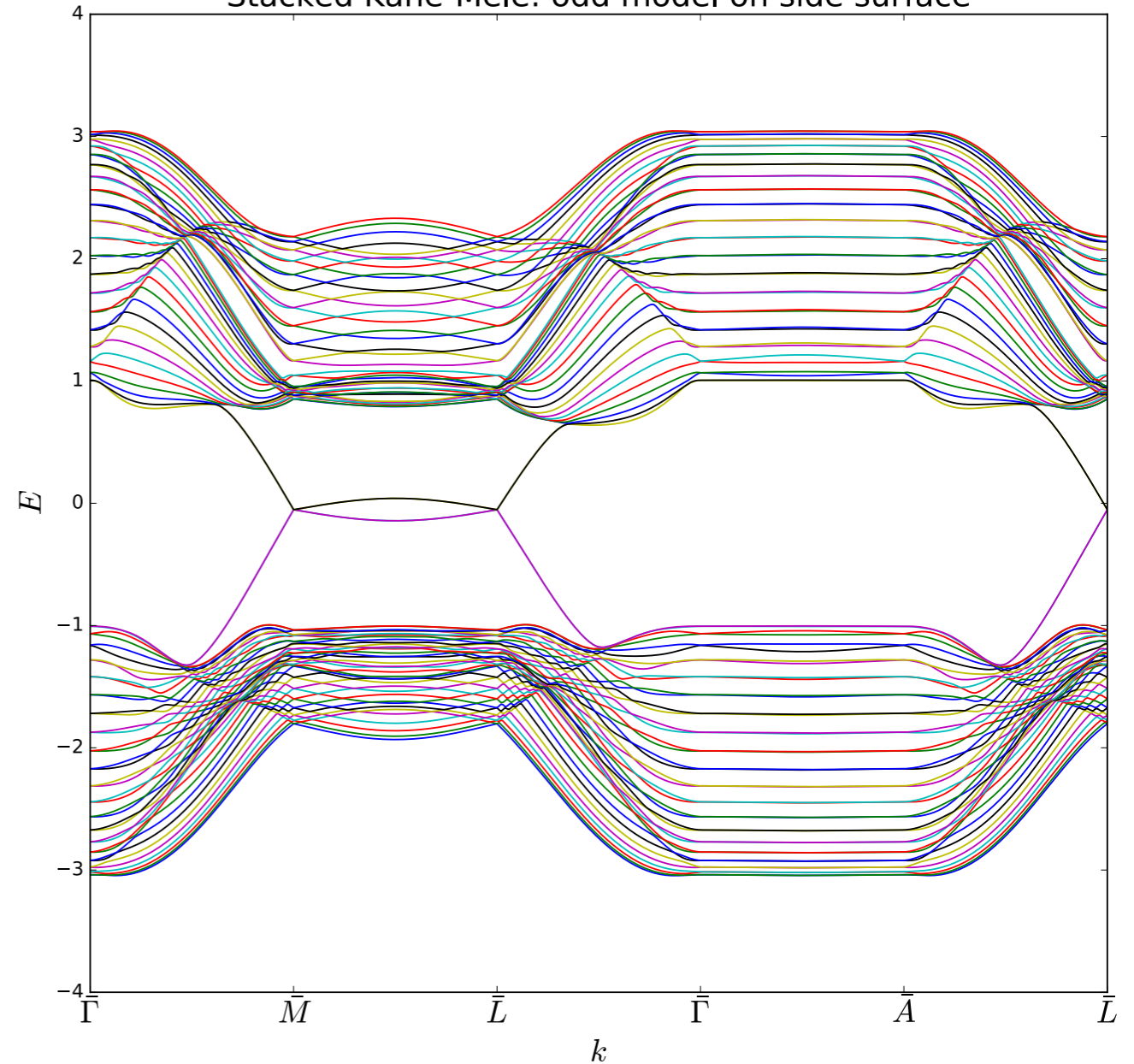


Stacked Kane-Mele: side surfaces have two Dirac cones

Stacked Kane-Mele: even model on side surface



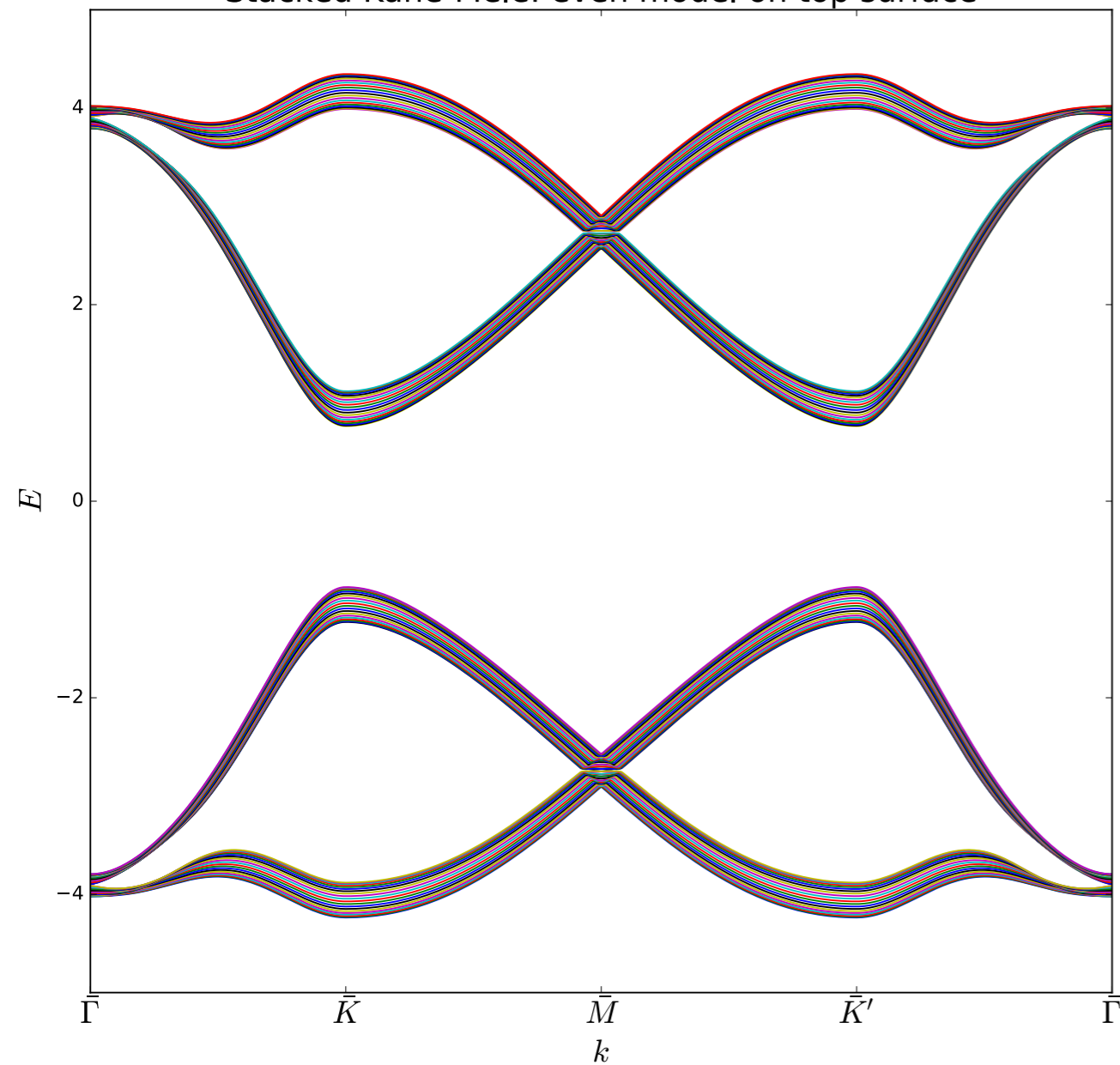
Stacked Kane-Mele: odd model on side surface



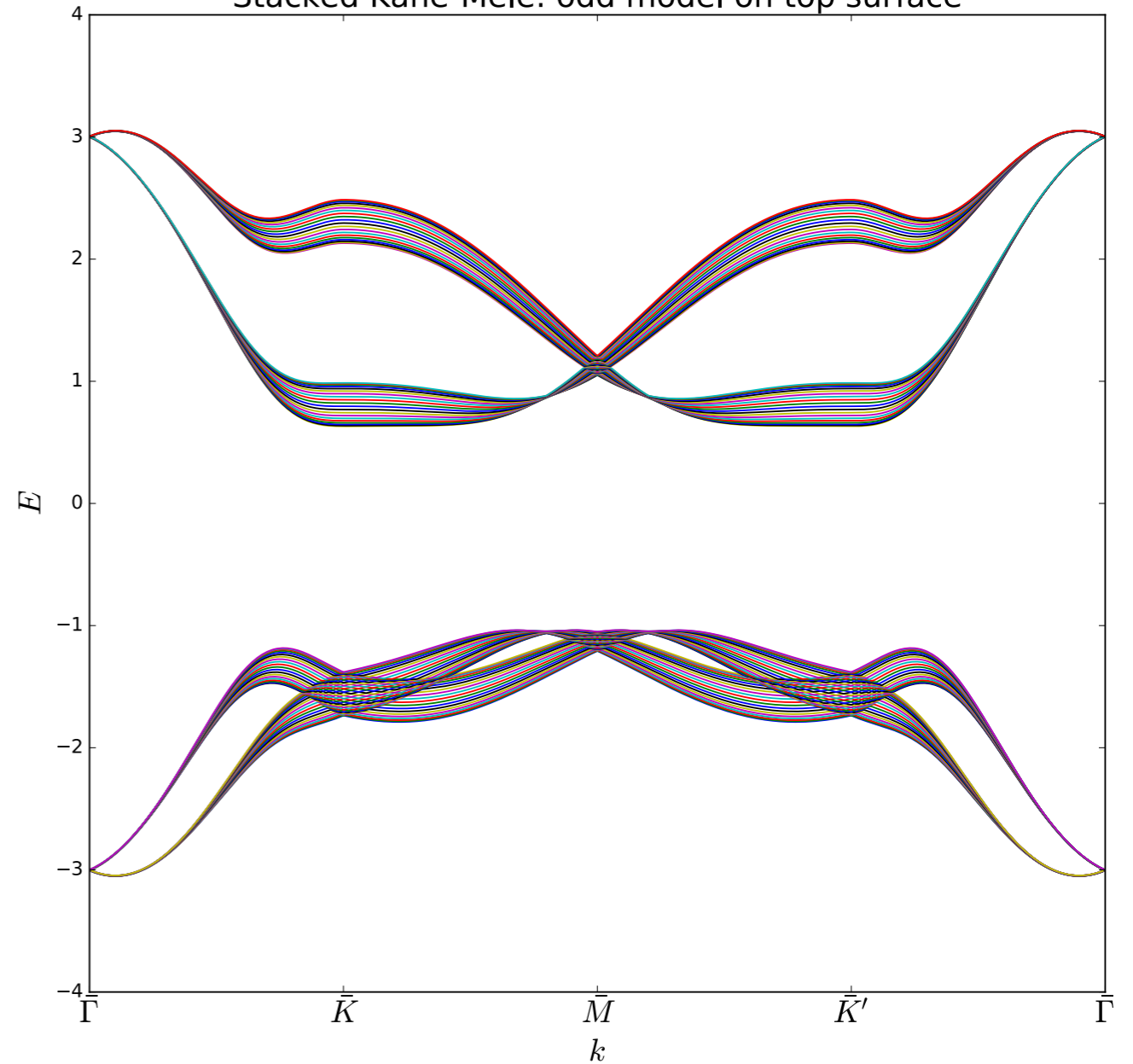
PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Stacked Kane-Mele: top surface is gapped

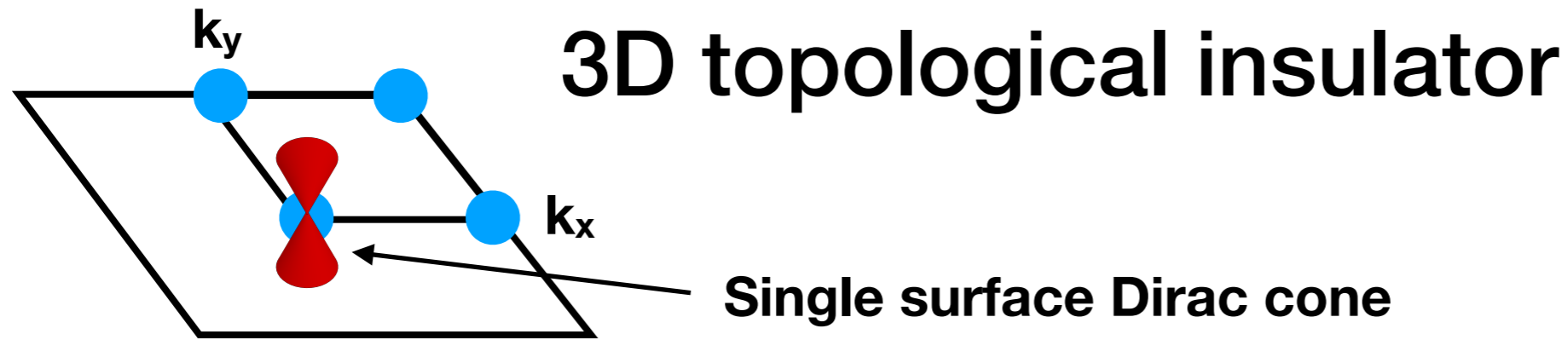
Stacked Kane-Mele: even model on top surface



Stacked Kane-Mele: odd model on top surface

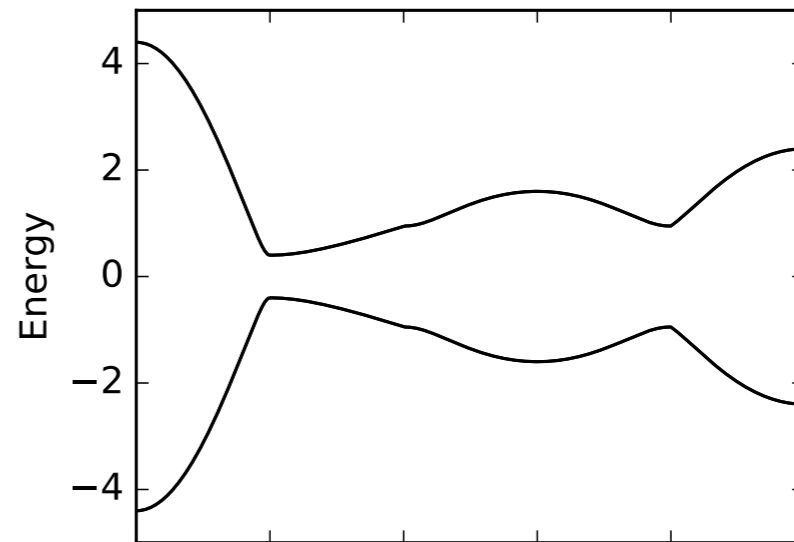


PythTB is based at: <http://physics.rutgers.edu/pythtb/>

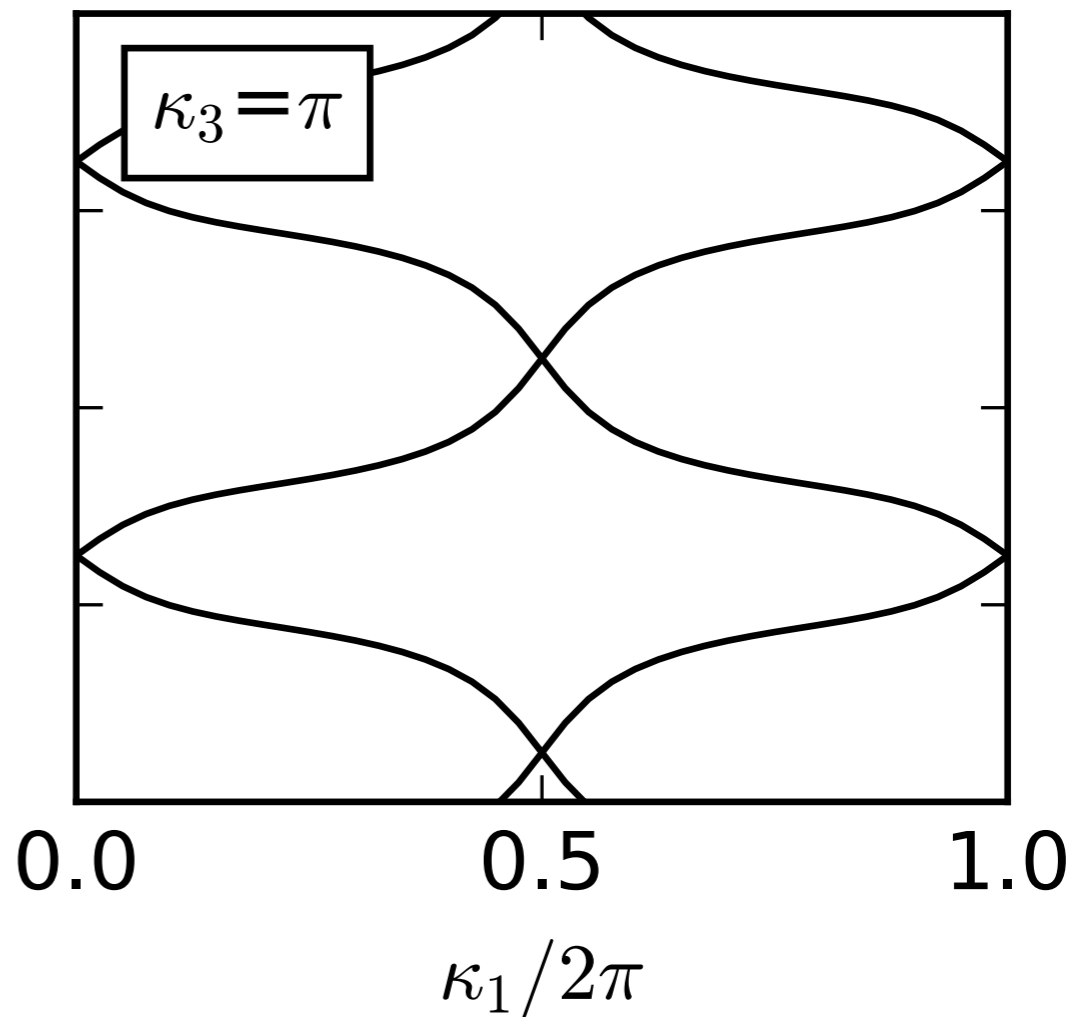
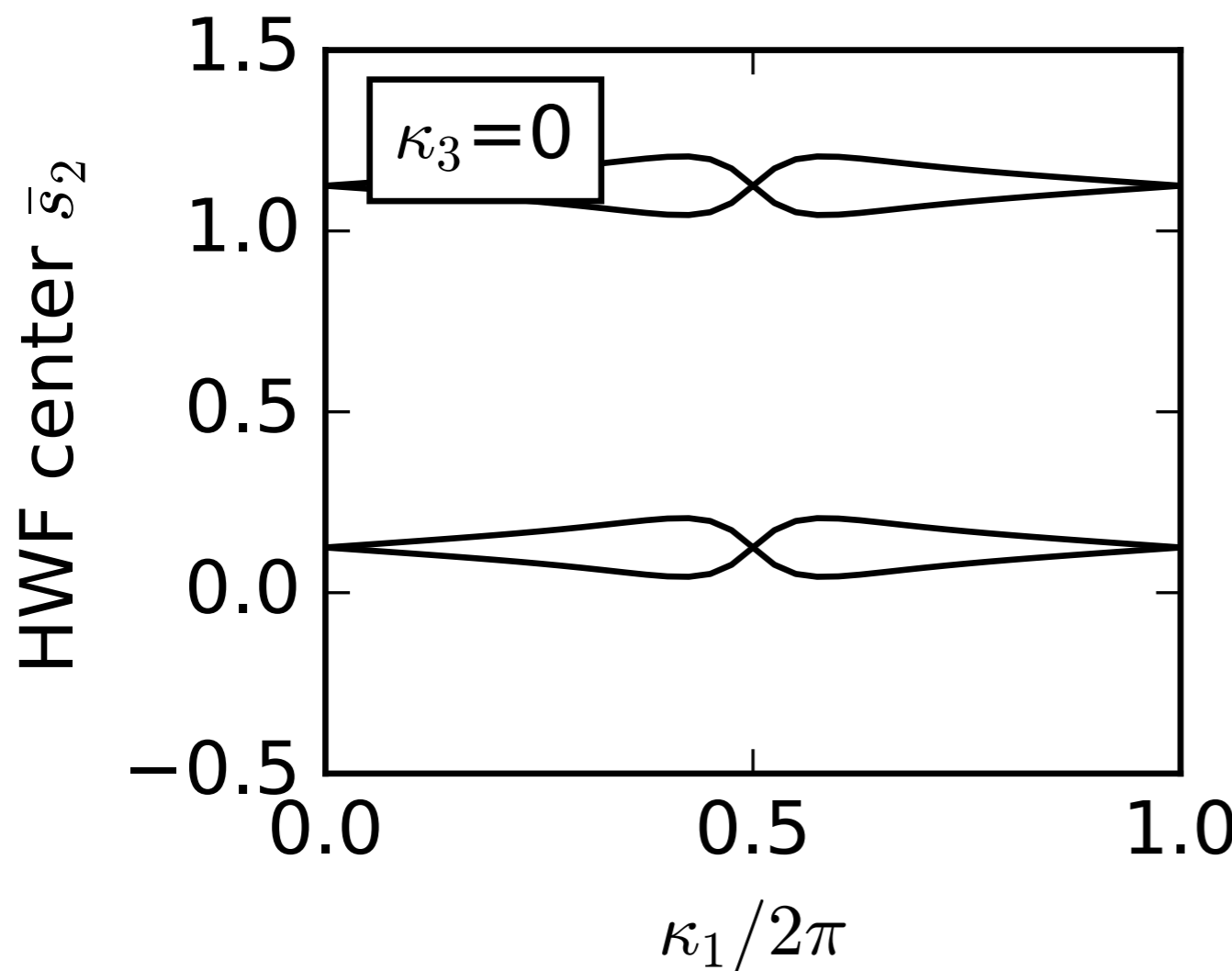


- Fu-Kane-Mele model: Phys. Rev. Lett. 98, 106803 (2007)
- PythTB code, fkm.py, available by David Vanderbilt at:
[http://physics.rutgers.edu/~dhv/pythtb-book-examples/
ptb_samples.html](http://physics.rutgers.edu/~dhv/pythtb-book-examples/ptb_samples.html)

Fu-Kane-Mele model

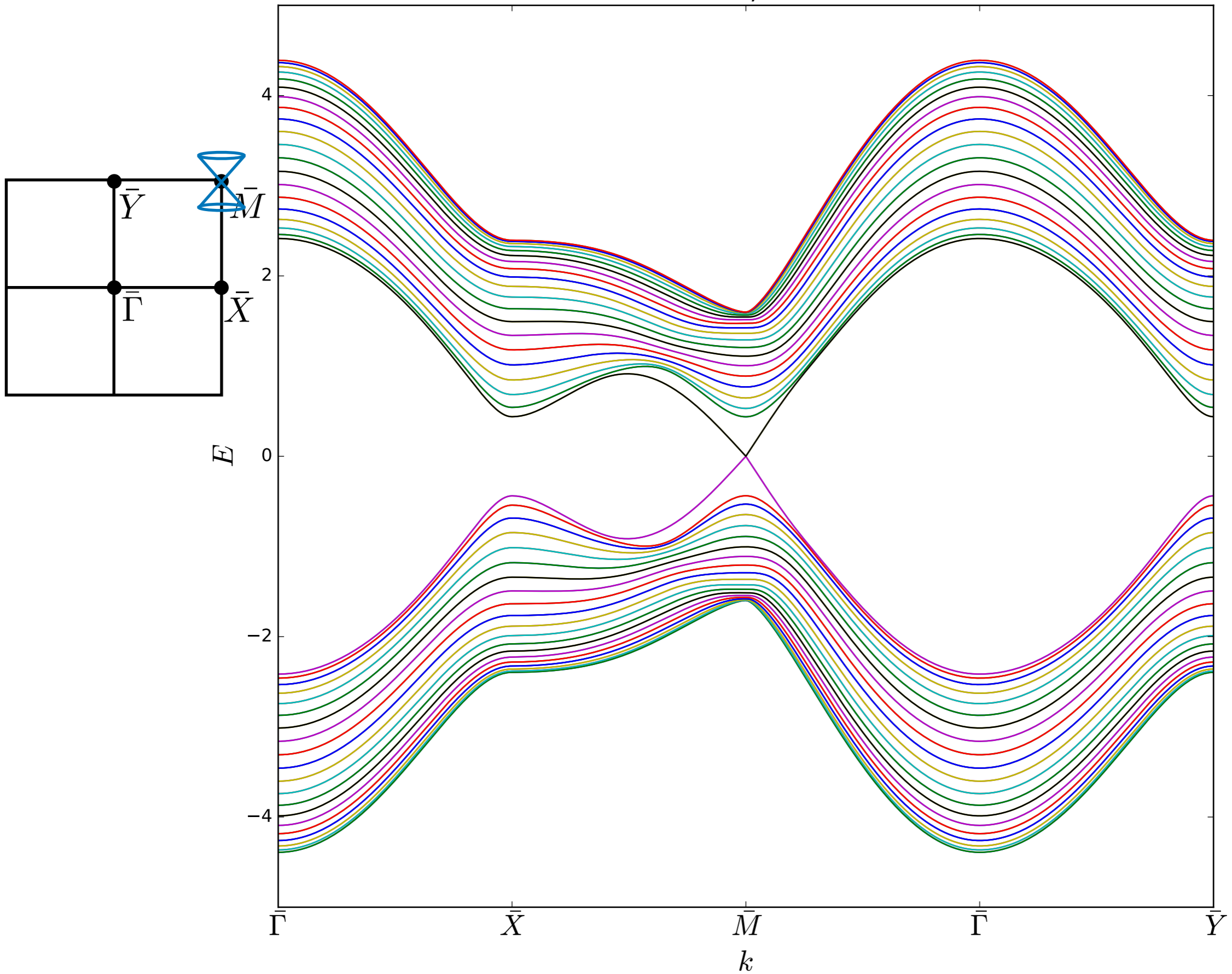


Bulk band structure is gapped



Hybrid Wannier function centers/Berry phase: wind in $k_3=\pi$ plane, not in $k_3=0$
Hallmark of strong TI!!

Fu-Kane-Mele model, surface Dirac cone



PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Exercises

1. Download the Kane-Mele model and plot band structure and Wannier centers. Use “cut_piece” to plot the edge band structure. In my plot I set the onsite energy to zero in the topological phase. What happens when it is non-zero?
2. Add a third dimension and implement a weak TI by stacking layers of the Kane-Mele model. Verify the side surfaces have two surface Dirac cones but the top surface is gapped. Add code to plot the Berry phase in the $k_z=0$ and $k_z=\pi$ planes.
3. Download the 3d Fu-Kane-Mele model from: http://physics.rutgers.edu/~dhv/pythtb-book-examples/ptb_samples.html. Implement the model. Cut into a finite slab to see the surface states. Notice that for a slab in any direction, there is one surface Dirac cone.