

PythTB for (topological) tight-binding models

Jennifer Cano

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

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/>

But I don't know Python....

- Learning Python is a good time investment.
- Let this be your excuse to learn it.
- Truth: the code is so easy, you don't even need to know Python to use it.

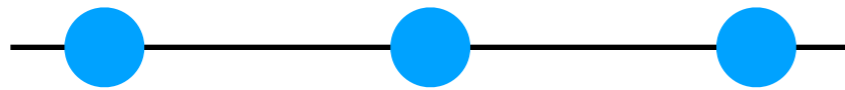
Goals

- Today: plot band structures for models from the previous lecture.
- Tomorrow: implement topological tight-binding models in PythTB to visualize topological surface states and Berry phase.

Example 1: atoms in 1d

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

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



Lattice vectors (we only have one)

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

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

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

k path in units of reciprocal lattice vecs

Labels of k-points on path

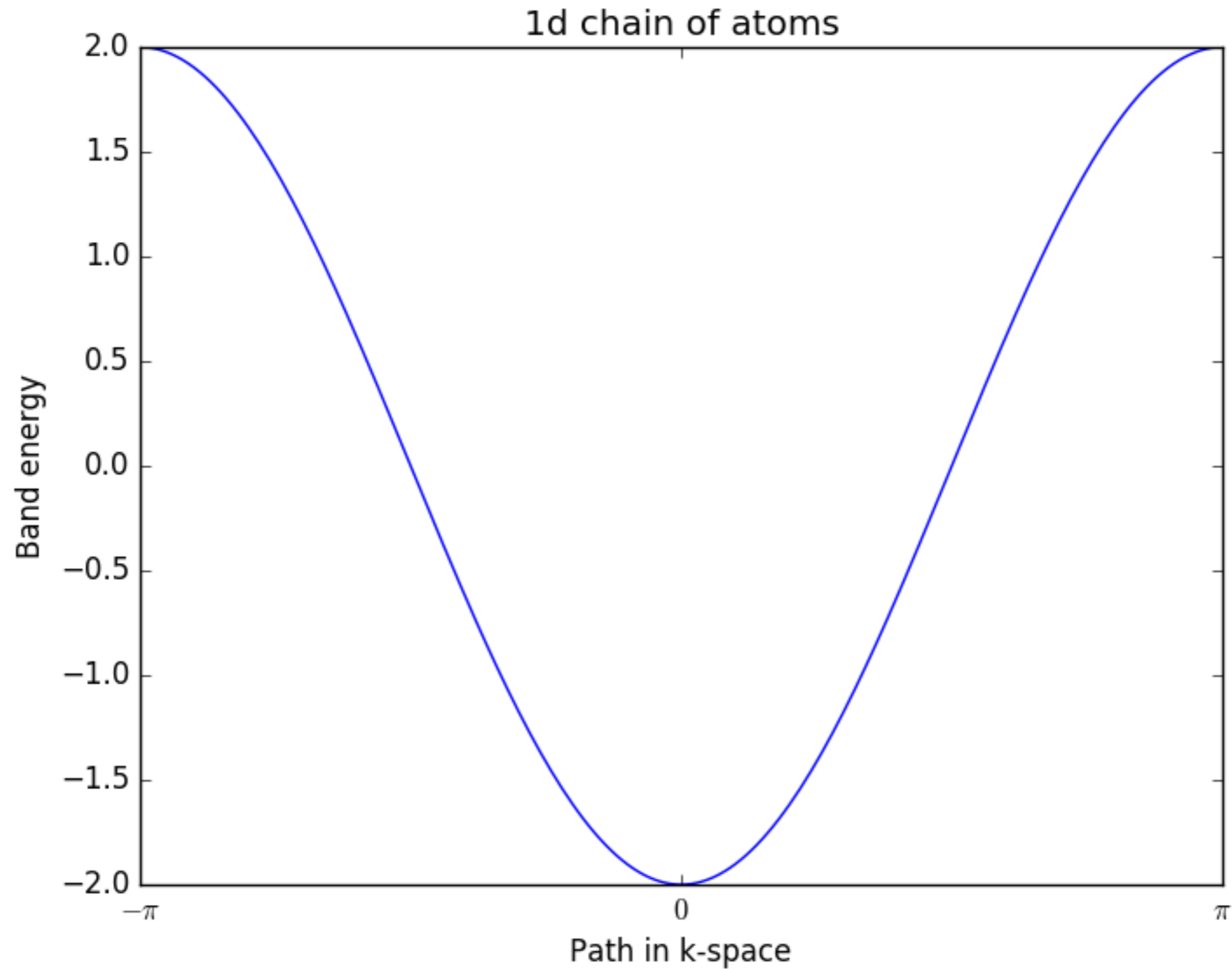
Repeat this line for more bands, up to evals[n]

Vector of ticks based on path and numsteps

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Aug 24 21:23:16 2018
4
5 @author: jennifercano
6 """
7
8 # adaptation from "simple example"
9 # at http://physics.rutgers.edu/pythtb/examples.html
10
11 from pythtb import * # import TB model class
12 import numpy as np
13 import matplotlib.pyplot as plt
14
15 # specify model
16 # lattice vectors
17 lat=[[1.0]]
18 # positions of orbitals
19 orb=[[0.0]]
20
21 # define the model
22 my_model=tb_model(1,1,lat,orb)
23 # assign hopping terms
24 my_model.set_hop(-1., 0, 0, [1])
25
26 # define a path in k-space to plot
27 path=[[-.5],[0],[.5]]
28 # label k points
29 label=(r'$-\pi$',r'$0$',r'$\pi$')
30 # number of steps between points
31 numsteps=100
32 kpts=k_path(path,numsteps)
33
34 # solve model
35 evals=my_model.solve_all(kpts)
36
37 # make a figure object
38 fig=plt.figure()
39 # plot bands
40 plt.plot(evals[0])
41 # put title on top
42 plt.title("1d chain of atoms")
43 plt.xlabel("Path in k-space")
44 plt.ylabel("Band energy")
45 plt.xticks([0,100,200],label)
```

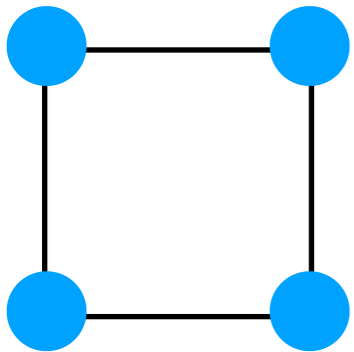
PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Example 1: atoms in 1d



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

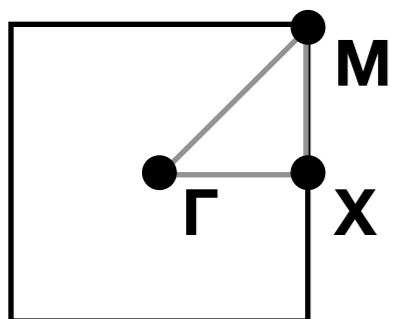
Example 2: square lattice



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

Two two-component lattice vecs

Still one orbital; now two components



k path: Γ, X, M, Γ

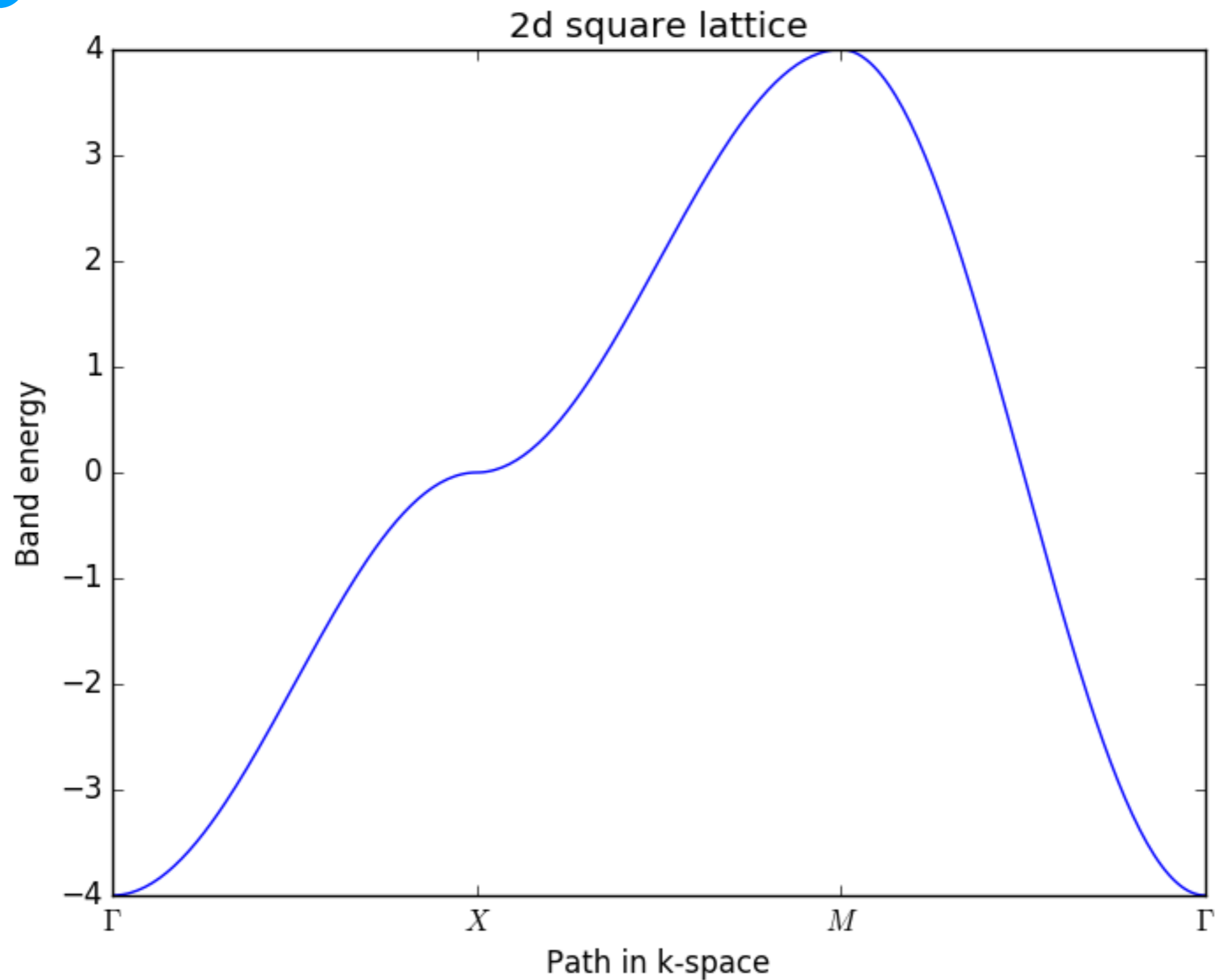
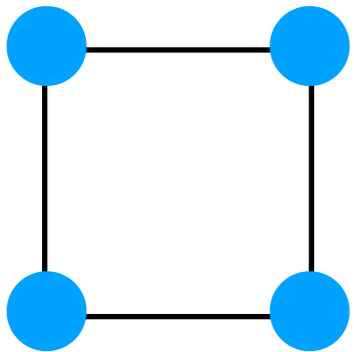
```

11 from pythtb import * # import TB model class
12 import numpy as np
13 import matplotlib.pyplot as plt
14
15 # specify model
16 # lattice vectors
17 lat=[[1.0,0.0],[0.0,1.0]]
18 # positions of orbitals
19 orb=[[0.0,0.0]]
20
21 # define the model
22 my_model=tb_model(2,2,lat,orb)
23 # assign hopping terms
24 # x-hopping
25 my_model.set_hop(-1., 0, 0, [1.0,0])
26 # y-hopping
27 my_model.set_hop(-1., 0, 0, [0,1.0])
28
29 # define a path in k-space to plot
30 path=[[0.0,0.0],[.5,0],[.5,.5],[0.0,0.0]]
31 # label k points
32 label=(r'\Gamma', r'X', r'M', r'\Gamma')
33 # number of steps between points
34 numsteps=100
35 kpts=k_path(path,numsteps)
36
37 # solve model
38 evals=my_model.solve_all(kpts)
39
40 # make a figure object
41 fig=pl.figure()
42 # plot bands
43 pl.plot(evals[0])
44 # put title on top
45 pl.title("2d square lattice")
46 pl.xlabel("Path in k-space")
47 pl.ylabel("Band energy")
48 pl.xticks([0,100,200,300],label)

```

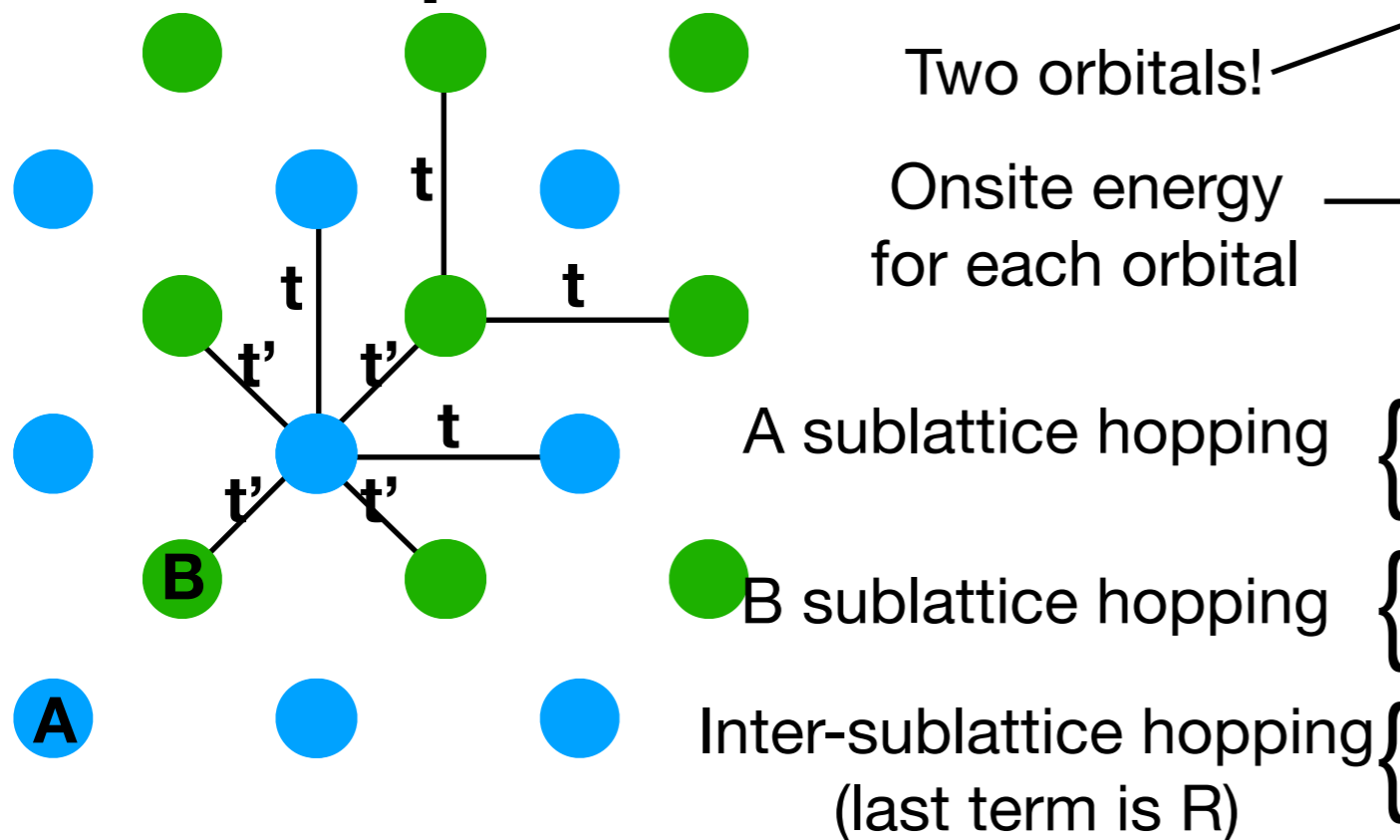
Added another tick because longer k path

Example 2: square lattice



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

Example 3: interpenetrating square lattices



```

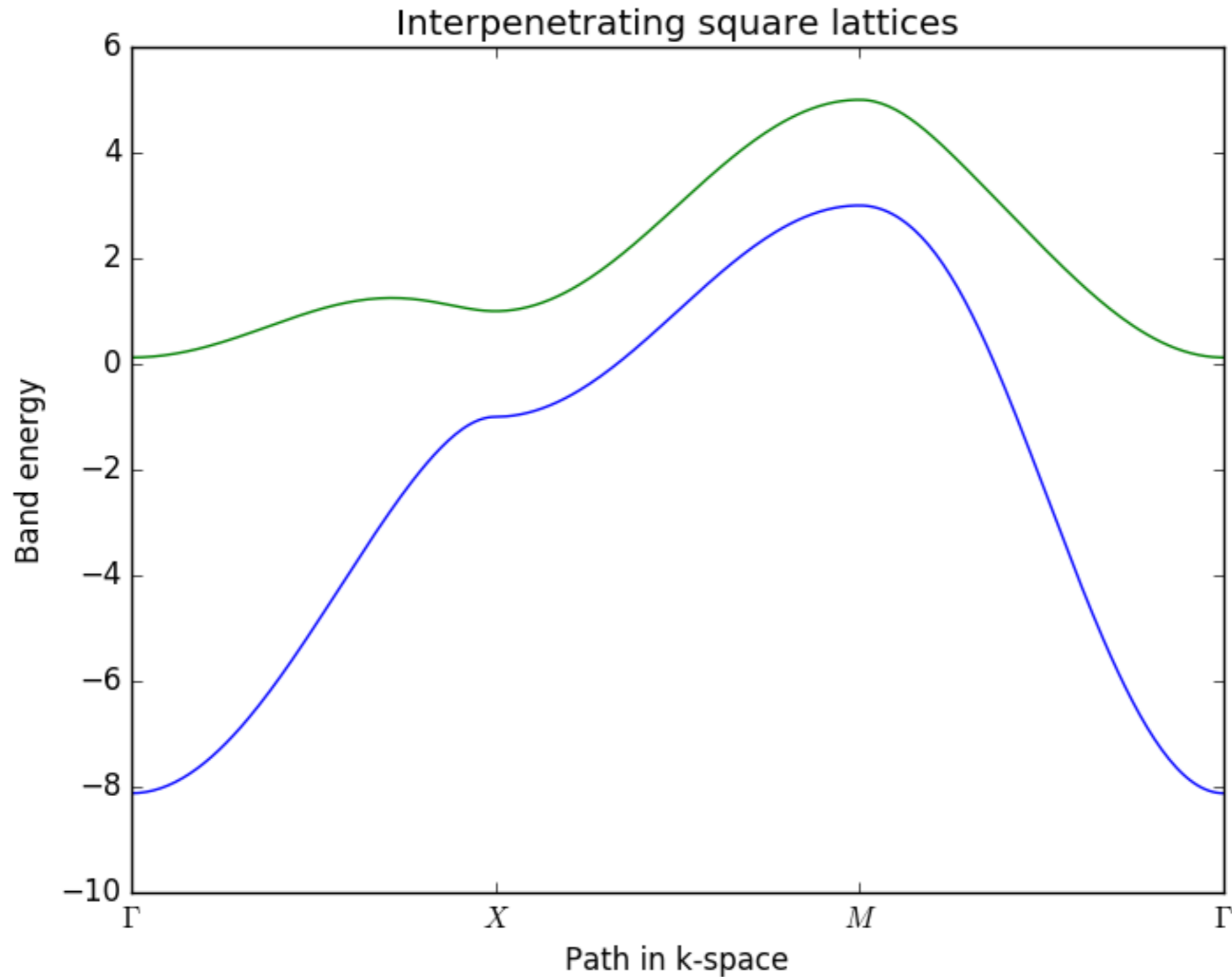
12 # specify model
13 # lattice vectors
14 lat=[[1.0,0.0],[0.0,1.0]]
15 # positions of orbitals
16 orb=[[0.0,0.0],[.5,.5]]
17
18 # define the model
19 my_model=tb_model(2,2,lat,orb)
20
21 # assign onsite energy
22 my_model.set_onsite([1.0,-1.0])
23
24 # assign hopping terms
25 t=1.0
26 t2=1.0
27 # x-hopping within sublattice of orbital "0"
28 my_model.set_hop(-t, 0, 0, [1.0,0])
29 # y-hopping within sublattice of orbital "0"
30 my_model.set_hop(-t, 0, 0, [0,1.0])
31 # x-hopping within sublattice of orbital "1"
32 my_model.set_hop(-t, 1, 1, [1.0,0])
33 # y-hopping within sublattice of orbital "1"
34 my_model.set_hop(-t, 1, 1, [0,1.0])
35 # four inter-sublattice hopping terms, from "0" to "1"
36 my_model.set_hop(-t2, 0, 1, [0.0,0.0])
37 my_model.set_hop(-t2, 0, 1, [-1.0,0.0])
38 my_model.set_hop(-t2, 0, 1, [-1.0,-1.0])
39 my_model.set_hop(-t2, 0, 1, [0.0,-1.0])
40
41 # define a path in k-space to plot
42 path=[[0.0,0.0],[.5,0],[.5,.5],[0.0,0.0]]
43 # label k points
44 label=(r'\Gamma',r'X',r'M',r'\Gamma')
45 # number of steps between points
46 numsteps=100
47 kpts=k_path(path,numsteps)
48
49 # solve model
50 evals=my_model.solve_all(kpts)
51
52 # make a figure object
53 fig=pl.figure()
54 # plot bands
55 pl.plot(evals[0])
56 pl.plot(evals[1])
57 # put title on top
58 pl.title("Interpenetrating square lattices")
59 pl.xlabel("Path in k-space")
60 pl.ylabel("Band energy")
61 pl.xticks([0,100,200,300],label)

```

Now plotting two bands

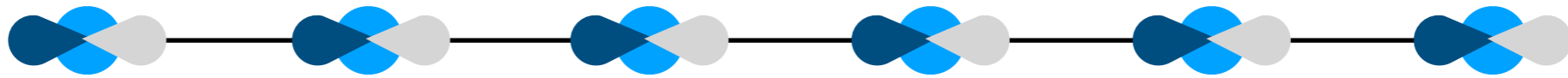
PythTB is based at: <http://physics>

Example 3: interpenetrating square lattices



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

Exercise 1: implement 1d chain with inversion, s and p orbitals



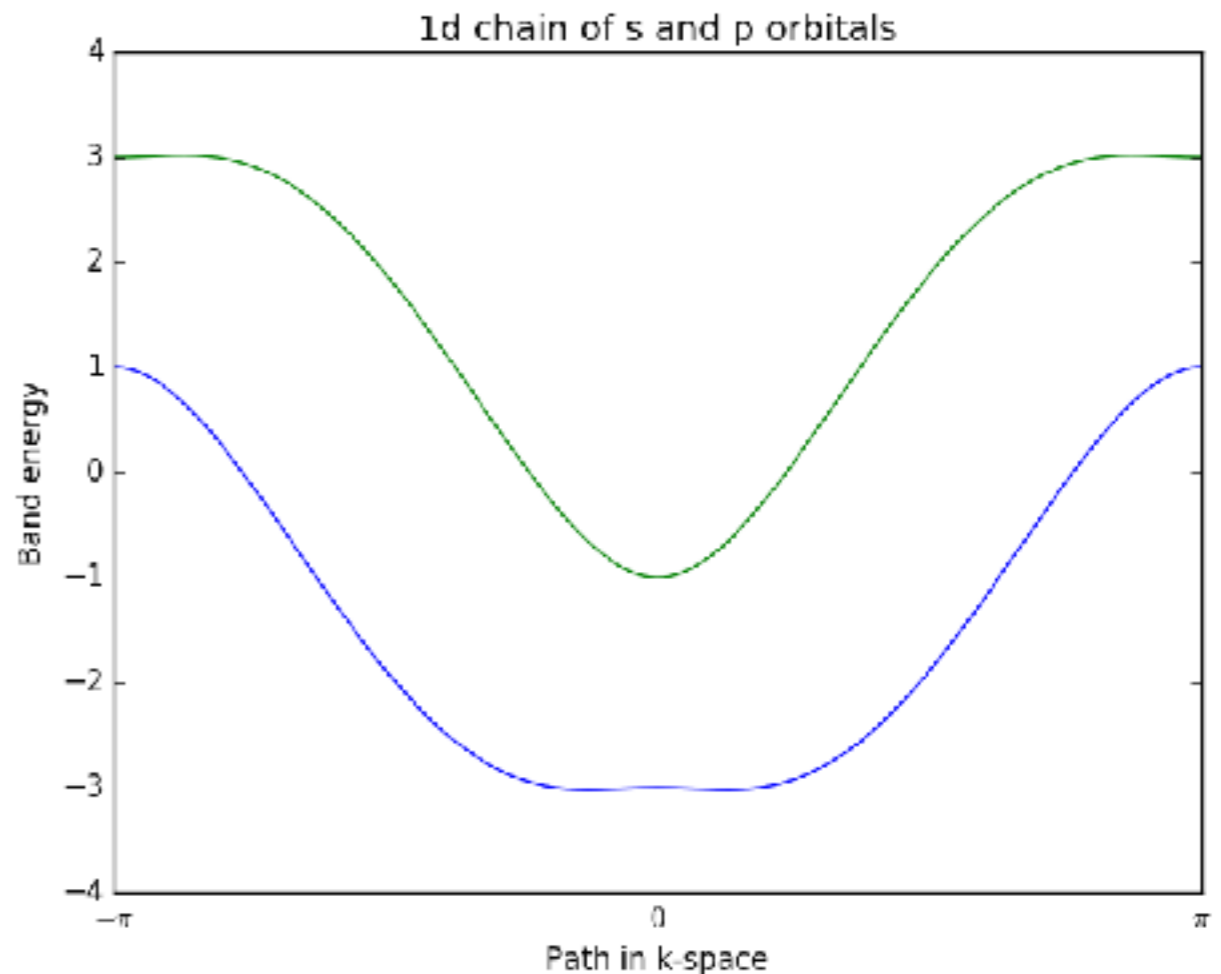
$$U = \sigma_z \quad H^k = \begin{pmatrix} \mu_s & 0 \\ 0 & \mu_p \end{pmatrix} + \begin{pmatrix} -t_s & 0 \\ 0 & -t_p \end{pmatrix} \cos k + \begin{pmatrix} 0 & -it_{sp} \\ it_{sp} & 0 \end{pmatrix} \sin k$$

onsite
nearest neighbor, same orbital
nearest neighbor, different orbital

$$H_{ss(pp)}(\hat{x}) = H_{ss(pp)}(-\hat{x})$$

$$H_{sp}(\hat{x}) = -H_{sp}(-\hat{x})$$

Sample spectrum:

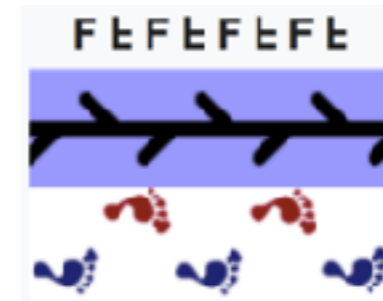


PythTB is based at: <http://physics.rutgers.edu/pythtb/>

Exercise 2: implement pg Hamiltonian and plot band structure

(Non-symmorphic) group generated by $\{m_y | \frac{1}{2} 0\}$

$$\mathbf{r}_1 = 0, \mathbf{r}_2 = 1/2$$



“Frieze”
group

glide exchanges orbitals

$$U_g = \sigma_x$$

$$\sigma_x H^k \sigma_x = H^k$$

$$H^k = \mu \mathbb{I} + \left(t \cos \frac{k}{2} + t' \sin \frac{k}{2} \right) \sigma_x$$

- both sites have same chemical potential
- two types of nearest neighbor hopping

Sample spectrum: $t = t' = 1$

