Srivani_HW5

July 5, 2018

1 Homework: 5 (Graphene bands)

1.0.1 Problem1. Find the analytical form of graphene's Hamiltonian and plots the band structure using the nearest neighbor tight-binding model with t=2.6eV and t=3.4 eV. (a) Represent the bands as surface plots and along the symmtry lines in the Brillouin zone. (c) Represent the bands considering an additional second nearest neighbor hopping term t=0.2eV and describe its effects. (d) Represent the bands in the presence of VA=0.5 eV and VB=0.5 eV site potential differences.

1.0.2 (a) Represent the bands along the symmtry lines

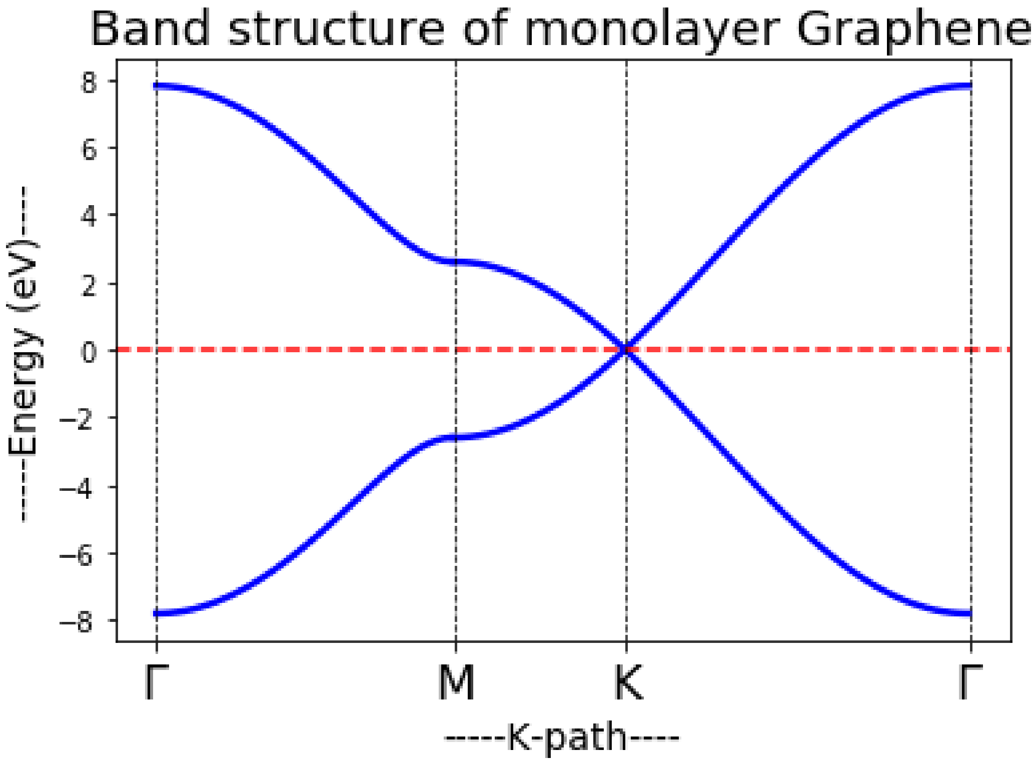
at t = -2.6 eV

```
In [17]: import numpy as np
         from numpy import linalg as LA
         import matplotlib.pyplot as plt

         Dim = 2 # Number of atoms in primitive cell
         VA = 0
         VB = 0
         t = -2.6 #eV  -> nearest neighbor hopping term

         wG2M1 = []
         wG2M2 = []
         kxG2M =[]

         wM2K1 = []
         wM2K2 = []
         kxM2K = []

         wK2G1 = []
         wK2G2 = []
         kxK2G = []

         a = 2.46 # Ao ->lattice parameter
         acc =  a/np.sqrt(3) # bond-length

         R1 = [(a/np.sqrt(3)),0]
```

```
R2 = [(-a/(2*np.sqrt(3))),a/2]
R3 = [(-a/(2*np.sqrt(3))),-a/2]

H = np.zeros((Dim,Dim)) # Hamiltonian


Sym_pts = [[0,0],[((2*np.pi)/(np.sqrt(3)*a)),0],[((2*np.pi)/(np.sqrt(3)*a)),((2*np.pi)/
#           G                   M                               K




kx1 = np.linspace(Sym_pts[0][0],Sym_pts[1][0], 100)
ky1 = np.linspace(Sym_pts[0][1],Sym_pts[1][1], 100)

kx2 = np.linspace(Sym_pts[1][0],Sym_pts[2][0], 100)
ky2 = np.linspace(Sym_pts[1][1],Sym_pts[2][1], 100)

kx3 = np.linspace(Sym_pts[2][0],Sym_pts[3][0], 100)
ky3 = np.linspace(Sym_pts[2][1],Sym_pts[3][1], 100)


r1 =  np.linspace(0,(2*np.pi/(np.sqrt(3)*a)),100)
r2 =  np.linspace((2*np.pi/(np.sqrt(3)*a)),((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),1
r3 =  np.linspace(((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),((2*np.pi/(np.sqrt(3)*a))+


def f(R,x,y):
    return (np.exp(1j*np.dot([x,y],[R[0],R[1]])))




for i in range(len(kx1)):
        tAB = t*(f(R1,kx1[i],ky1[i])+f(R2,kx1[i],ky1[i])+f(R3,kx1[i],ky1[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]


        w1, v1 = LA.eig(H)
        wG2M1.append(w1[0])
        wG2M2.append(w1[1])

for i in range(len(kx2)):
        tAB = t*(f(R1,kx2[i],ky2[i])+f(R2,kx2[i],ky2[i])+f(R3,kx2[i],ky2[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]


        w1, v1 = LA.eig(H)
        wM2K1.append(w1[0])
        wM2K2.append(w1[1])
```

```
for i in range(len(kx3)):
        tAB = t*(f(R1,kx3[i],ky3[i])+f(R2,kx3[i],ky3[i])+f(R3,kx3[i],ky3[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]


        w1, v1 = LA.eig(H)
        #print (w1)
        wK2G1.append(w1[0])
        wK2G2.append(w1[1])




for k in range(len(r1)):
        kxG2M.append(r1[k])
        kxM2K.append(r2[k])
        kxK2G.append(r3[k])

plt.plot(kxG2M,wG2M1,'-b', linewidth=2.5)
plt.plot(kxG2M,wG2M2,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K1,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K2,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G1,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G2,'-b', linewidth=2.5)
plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
plt.xlabel('-----K-path----',fontsize=14)
plt.ylabel('-----Energy (eV)----',fontsize=14)
plt.title('Band structure of monolayer Graphene',fontsize=18)

x = np.array([0 , 1.4746336294587137  ,  2.32601375  ,  4.028774])
plt.axvline(x=0.0, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=1.4746336294587137, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=2.32601375, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=4.028774, color='k', linestyle='--', linewidth=0.7)

my_xticks = ['$\Gamma$','M','K','$\Gamma$']
plt.xticks(x, my_xticks,fontsize=18)


plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
```

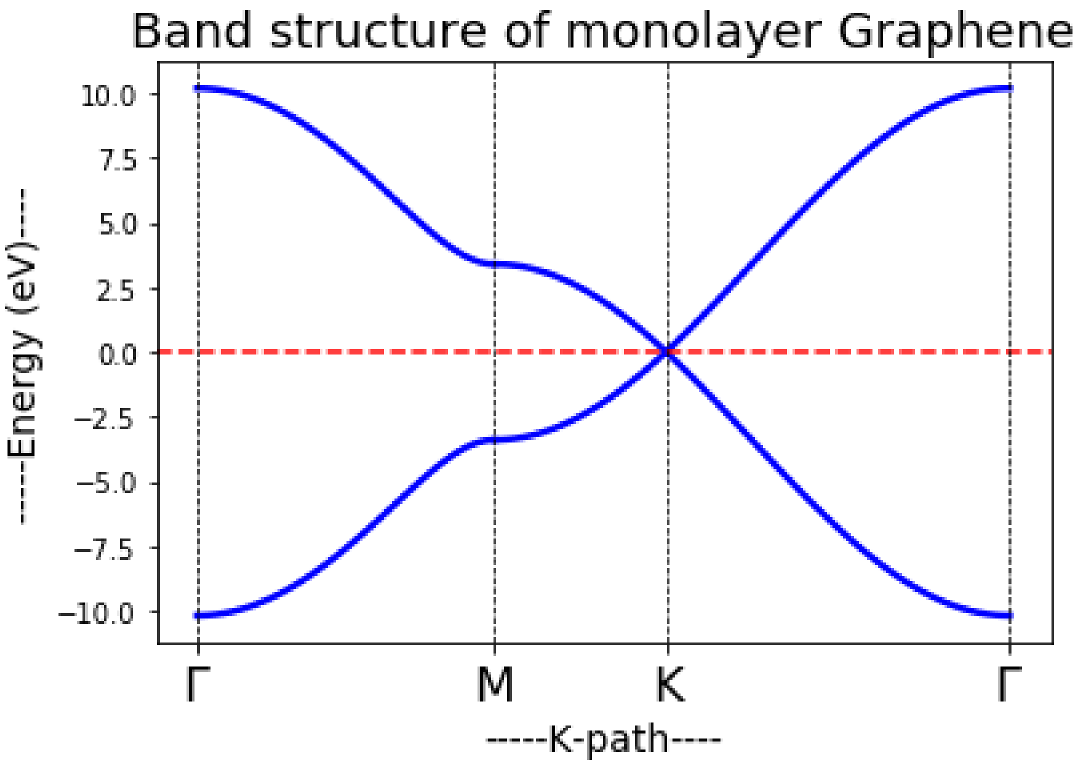


at t = -3.4 eV

```
In [18]: import numpy as np
         from numpy import linalg as LA
         import matplotlib.pyplot as plt

         Dim = 2 # Number of atoms in primitive cell
         VA = 0
         VB = 0
         t = -3.4 #eV  -> nearest neighbor hopping term

         wG2M1 = []
         wG2M2 = []
         kxG2M =[]

         wM2K1 = []
         wM2K2 = []
         kxM2K = []

         wK2G1 = []
         wK2G2 = []
         kxK2G = []
```

```
a = 2.46 # Ao ->lattice parameter
acc =  a/np.sqrt(3) # bond-length

R1 = [(a/np.sqrt(3)),0]
R2 = [(-a/(2*np.sqrt(3))),a/2]
R3 = [(-a/(2*np.sqrt(3))),-a/2]

H = np.zeros((Dim,Dim)) # Hamiltonian


Sym_pts = [[0,0],[((2*np.pi)/(np.sqrt(3)*a)),0],[((2*np.pi)/(np.sqrt(3)*a)),((2*np.pi)/
#           G                    M                              K




kx1 = np.linspace(Sym_pts[0][0],Sym_pts[1][0], 100)
ky1 = np.linspace(Sym_pts[0][1],Sym_pts[1][1], 100)

kx2 = np.linspace(Sym_pts[1][0],Sym_pts[2][0], 100)
ky2 = np.linspace(Sym_pts[1][1],Sym_pts[2][1], 100)

kx3 = np.linspace(Sym_pts[2][0],Sym_pts[3][0], 100)
ky3 = np.linspace(Sym_pts[2][1],Sym_pts[3][1], 100)


r1 =  np.linspace(0,(2*np.pi/(np.sqrt(3)*a)),100)
r2 =  np.linspace((2*np.pi/(np.sqrt(3)*a)),((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),1
r3 =  np.linspace(((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),((2*np.pi/(np.sqrt(3)*a))+


def f(R,x,y):
    return (np.exp(1j*np.dot([x,y],[R[0],R[1]])))




for i in range(len(kx1)):
        tAB = t*(f(R1,kx1[i],ky1[i])+f(R2,kx1[i],ky1[i])+f(R3,kx1[i],ky1[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]



        w1, v1 = LA.eig(H)
        wG2M1.append(w1[0])
        wG2M2.append(w1[1])

for i in range(len(kx2)):
        tAB = t*(f(R1,kx2[i],ky2[i])+f(R2,kx2[i],ky2[i])+f(R3,kx2[i],ky2[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]
        w1, v1 = LA.eig(H)
        wM2K1.append(w1[0])
        wM2K2.append(w1[1])

for i in range(len(kx3)):
        tAB = t*(f(R1,kx3[i],ky3[i])+f(R2,kx3[i],ky3[i])+f(R3,kx3[i],ky3[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]


        w1, v1 = LA.eig(H)
        #print (w1)
        wK2G1.append(w1[0])
        wK2G2.append(w1[1])




for k in range(len(r1)):
        kxG2M.append(r1[k])
        kxM2K.append(r2[k])
        kxK2G.append(r3[k])

plt.plot(kxG2M,wG2M1,'-b', linewidth=2.5)
plt.plot(kxG2M,wG2M2,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K1,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K2,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G1,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G2,'-b', linewidth=2.5)
plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
plt.xlabel('-----K-path----',fontsize=14)
plt.ylabel('-----Energy (eV)----',fontsize=14)
plt.title('Band structure of monolayer Graphene',fontsize=18)

x = np.array([0 , 1.4746336294587137  ,  2.32601375  ,  4.028774])
plt.axvline(x=0.0, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=1.4746336294587137, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=2.32601375, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=4.028774, color='k', linestyle='--', linewidth=0.7)

my_xticks = ['$\Gamma$','M','K','$\Gamma$']
plt.xticks(x, my_xticks,fontsize=18)


plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
```



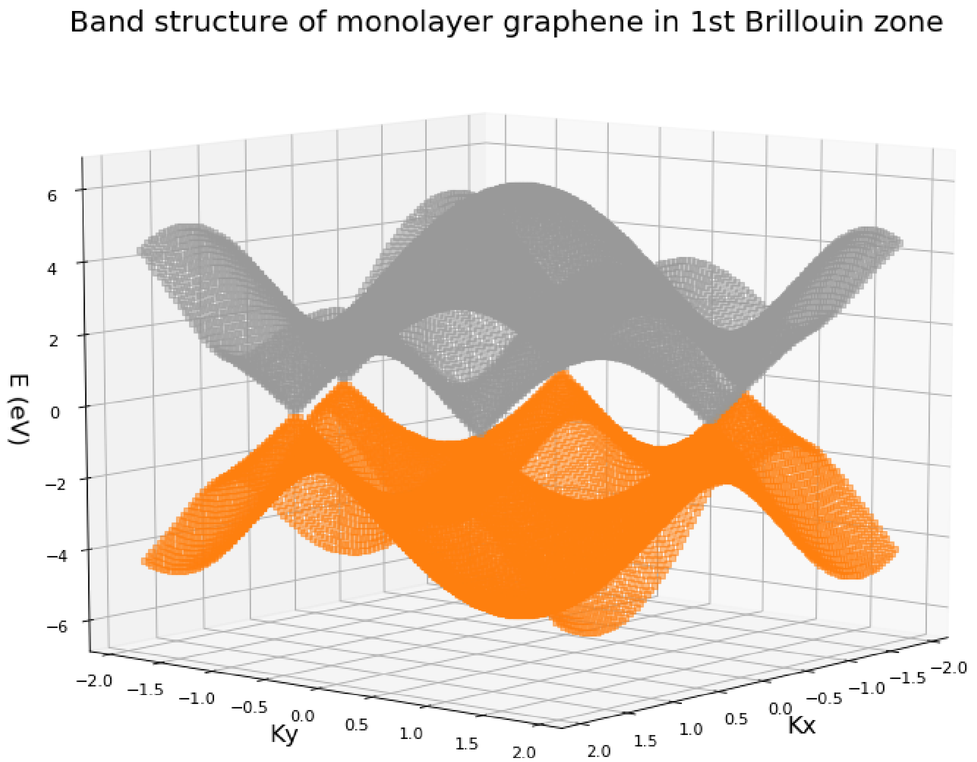
(b) Represent the bands as surface plots in the Brillouin zone.

```
In [4]: from mpl_toolkits.mplot3d import Axes3D
        import matplotlib.pyplot as plt
        from matplotlib import cm
        from matplotlib.ticker import LinearLocator, FormatStrFormatter
        import numpy as np
        from numpy import linalg as LA


        Dim = 2 # Number of atoms in primitive cell
        VA = 0
        VB = 0
        t = -2 #eV  -> nearest neighbor hopping term

        wG2M1 = []
        wG2M2 = []
        kxG2M =[]

        wM2K1 = []
        wM2K2 = []
```

```
kxM2K = []

wK2G1 = []
wK2G2 = []
kxK2G = []

a = 2.46 # Ao ->lattice parameter
acc =  a/np.sqrt(3) # bond-length

R1 = [(a/np.sqrt(3)),0]
R2 = [(-a/(2*np.sqrt(3))),a/2]
R3 = [(-a/(2*np.sqrt(3))),-a/2]

N = 100
Kx = np.linspace(-3*np.pi/(2*a),3*np.pi/(2*a),N)
Ky = np.linspace(-3*np.pi/(2*a),3*np.pi/(2*a),N)
#Kx, Ky = np.meshgrid(Kx, Ky)


H = np.zeros((Dim,Dim)) # Hamiltonian




def f(R,x,y):
    return (np.exp(1j*np.dot([x,y],[R[0],R[1]])))


fig = plt.figure(figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
ax = fig.add_subplot(111, projection='3d')

for i in range(N):
    for j in range(N):
        tAB = t*(f(R1,Kx[i],Ky[j])+f(R2,Kx[i],Ky[j])+f(R3,Kx[i],Ky[j]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]
        w1, v1 = LA.eig(H)

        ax.scatter(Kx[i],Ky[j],w1[0],color='#999999', marker='s', alpha=0.5)
        ax.scatter(Kx[i],Ky[j],w1[1],color='C1', marker='s', alpha=0.5)

ax.view_init(elev=10., azim = 40)
ax.set_xlabel('Kx',fontsize=15)
ax.set_ylabel('Ky',fontsize=15)
ax.set_zlabel('E (eV)',fontsize=15)
ax.set_title('Band structure of monolayer graphene in 1st Brillouin zone',fontsize=18)
plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
```

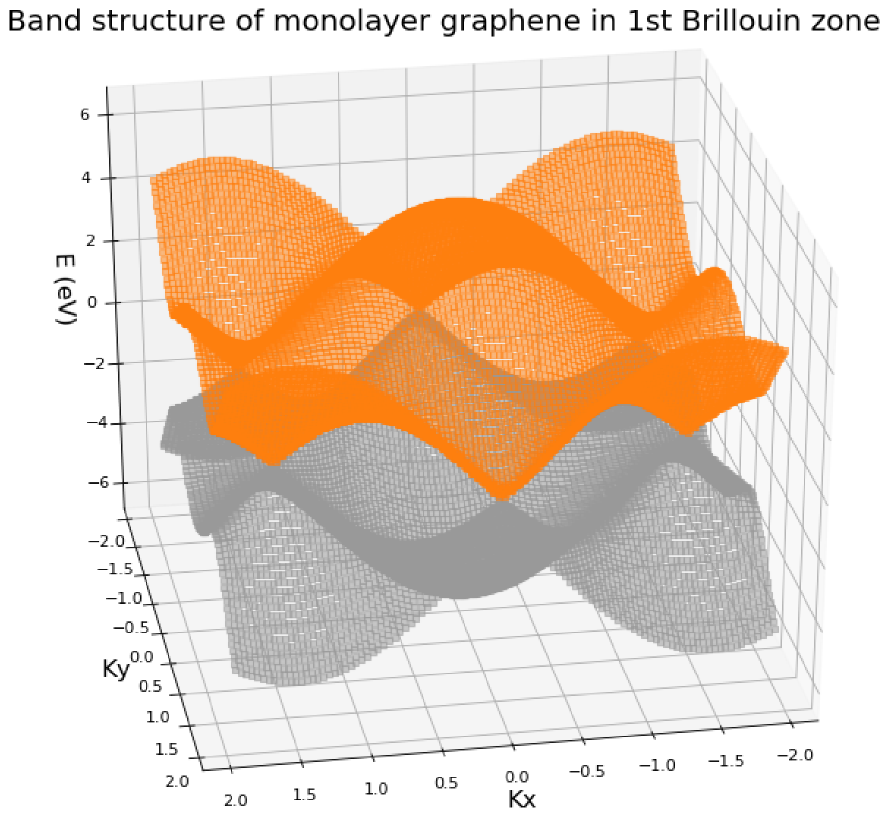
```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:544: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order, subok=True)
```



```
In [5]: fig = plt.figure(figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
        ax = fig.add_subplot(111, projection='3d')

        for i in range(N):
            for j in range(N):
                tAB = t*(f(R1,Kx[i],Ky[j])+f(R2,Kx[i],Ky[j])+f(R3,Kx[i],Ky[j]))
                H = [[VA,tAB],[np.conjugate(tAB),VB]]
                w1, v1 = LA.eig(H)

                ax.scatter(Kx[i],Ky[j],w1[0],color='C1', marker='s', alpha=0.5)
                ax.scatter(Kx[i],Ky[j],w1[1],color='#999999', marker='s', alpha=0.5)

        #ax.view_init(elev=90., azim = 90)
        ax.view_init(elev=30., azim = 80)
        ax.set_xlabel('Kx',fontsize=15)
```

```
         ax.set_ylabel('Ky',fontsize=15)
         ax.set_zlabel('E (eV)',fontsize=15)
         ax.set_title('Band structure of monolayer graphene in 1st Brillouin zone',fontsize=18)
         plt.show()
```

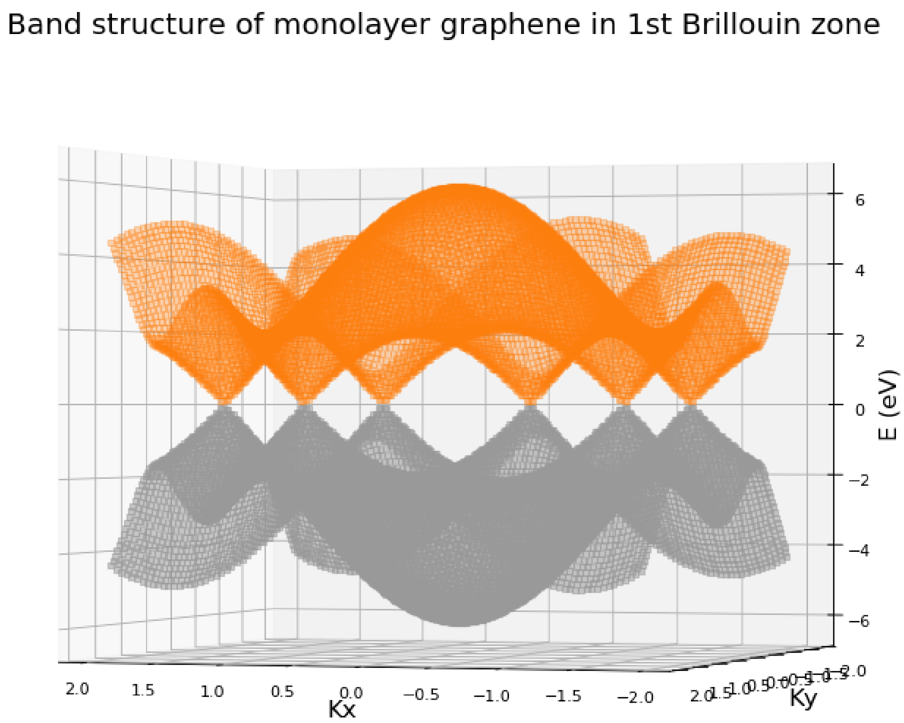
```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:544: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order, subok=True)
```



```
In [14]: fig = plt.figure(figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
         ax = fig.add_subplot(111, projection='3d')

         for i in range(N):
             for j in range(N):
                 tAB = t*(f(R1,Kx[i],Ky[j])+f(R2,Kx[i],Ky[j])+f(R3,Kx[i],Ky[j]))
                 H = [[VA,tAB],[np.conjugate(tAB),VB]]
                 w1, v1 = LA.eig(H)
```

```
                ax.scatter(Kx[i],Ky[j],w1[0],color='C1', marker='s', alpha=0.3)
                ax.scatter(Kx[i],Ky[j],w1[1],color='#999999', marker='s', alpha=0.5)

        ax.view_init(elev=0., azim = 108)
        ax.set_xlabel('Kx',fontsize=15)
        ax.set_ylabel('Ky',fontsize=15)
        ax.set_zlabel('E (eV)',fontsize=15)
        ax.set_title('Band structure of monolayer graphene in 1st Brillouin zone',fontsize=18)
        plt.show()
```

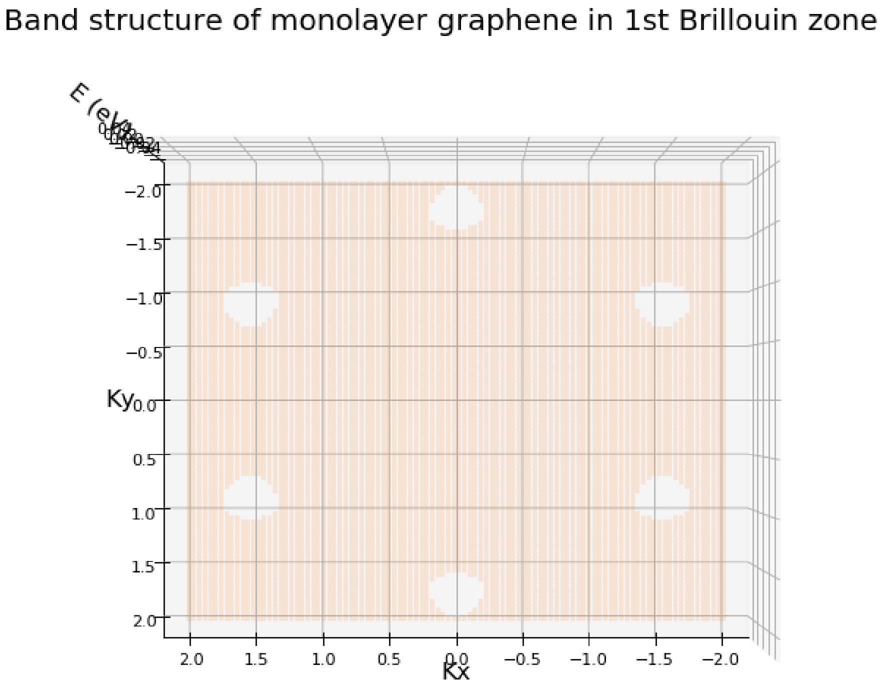
```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:544: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order, subok=True)
```



```
In [16]: fig = plt.figure(figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
         ax = fig.add_subplot(111, projection='3d')
```

```
for i in range(N):
    for j in range(N):
        tAB = t*(f(R1,Kx[i],Ky[j])+f(R2,Kx[i],Ky[j])+f(R3,Kx[i],Ky[j]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]
        w1, v1 = LA.eig(H)

        plt.scatter(Kx[i],Ky[j],w1[0],color='C1', marker='s', alpha=0.1)
        plt.scatter(Kx[i],Ky[j],w1[1],color='#999999', marker='s', alpha=0.5)

ax.view_init(elev=90., azim = 90)
ax.set_xlabel('Kx',fontsize=15)
ax.set_ylabel('Ky',fontsize=15)
ax.set_zlabel('E (eV)',fontsize=15)
ax.set_title('Band structure of monolayer graphene in 1st Brillouin zone',fontsize=18)
plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/matplotlib/collections.py:903: ComplexWarning
  self._transforms[:, 0, 0] = scale
/home/tcsl05/anaconda3/lib/python3.6/site-packages/matplotlib/collections.py:904: ComplexWarning
  self._transforms[:, 1, 1] = scale
```



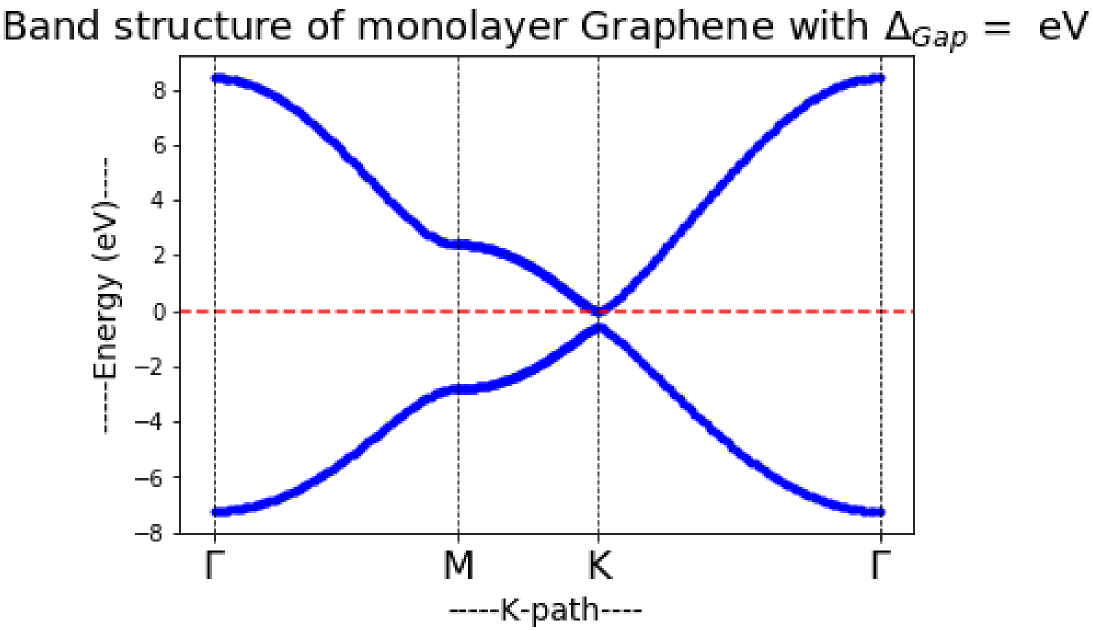
1.0.3 (c) Represent the bands considering an additional second nearest neighbor hopping term t=0.2eV and describe its effects.

In [32]:
```
import numpy as np
from numpy import linalg as LA
import matplotlib.pyplot as plt

Dim = 2 # Number of atoms in primitive cell
VA = 0
VB = 0
t = -2.6 #eV  -> nearest neighbor hopping term
t2 = 0.2 #eV

wG2M1 = []
wG2M2 = []
kxG2M =[]

wM2K1 = []
wM2K2 = []
kxM2K = []

wK2G1 = []
wK2G2 = []
kxK2G = []

a = 2.46 # Ao ->lattice parameter
acc =  a/np.sqrt(3) # bond-length

R1 = [(a/np.sqrt(3)),0]
R2 = [(-a/(2*np.sqrt(3))),a/2]
R3 = [(-a/(2*np.sqrt(3))),-a/2]

R21 = [0,a]
R22 = [0,-a]
R23 = [(np.sqrt(3)*a)/2,a/2]
R24 = [(np.sqrt(3)*a)/2,-a/2]
R25 = [(-np.sqrt(3)*a)/2,a/2]
R26 = [(-np.sqrt(3)*a)/2,-a/2]


H = np.zeros((Dim,Dim)) # Hamiltonian


Sym_pts = [[0,0],[((2*np.pi)/(np.sqrt(3)*a)),0],[((2*np.pi)/(np.sqrt(3)*a)),((2*np.pi)/
#            G                   M                              K
```

```
kx1 = np.linspace(Sym_pts[0][0],Sym_pts[1][0], 100)
ky1 = np.linspace(Sym_pts[0][1],Sym_pts[1][1], 100)

kx2 = np.linspace(Sym_pts[1][0],Sym_pts[2][0], 100)
ky2 = np.linspace(Sym_pts[1][1],Sym_pts[2][1], 100)

kx3 = np.linspace(Sym_pts[2][0],Sym_pts[3][0], 100)
ky3 = np.linspace(Sym_pts[2][1],Sym_pts[3][1], 100)


r1 =  np.linspace(0,(2*np.pi/(np.sqrt(3)*a)),100)
r2 =  np.linspace((2*np.pi/(np.sqrt(3)*a)),((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),1
r3 =  np.linspace(((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),((2*np.pi/(np.sqrt(3)*a))+


def f(R,x,y):
    return (np.exp(1j*np.dot([x,y],[R[0],R[1]])))

def g(x,y):
    return (np.exp(1j*np.dot([x,y],[R21[0],R21[1]]))+np.exp(1j*np.dot([x,y],[R22[0],R22

def gf(x,y):
    return (2*np.cos(y*a)+4*np.cos(x*np.sqrt(3)*a/2)*np.cos(y*a/2))


for i in range(len(kx1)):
        tAB = t*(f(R1,kx1[i],ky1[i])+f(R2,kx1[i],ky1[i])+f(R3,kx1[i],ky1[i]))
        tAA = VA+ t2*(f(R21,kx1[i],ky1[i])+f(R22,kx1[i],ky1[i])+f(R23,kx1[i],ky1[i])+f(
        #tAA = VA+t2*g(kx1[i],ky1[i])
        #tAA = VA+t2*gf(kx1[i],ky1[i])

        H = [[tAA,tAB],[np.conjugate(tAB),VB]]

        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wG2M1.append(w1[0])
        wG2M2.append(w1[1])

for i in range(len(kx2)):
        tAB = t*(f(R1,kx2[i],ky2[i])+f(R2,kx2[i],ky2[i])+f(R3,kx2[i],ky2[i]))
        tAA = VA+ t2*(f(R21,kx2[i],ky2[i])+f(R22,kx2[i],ky2[i])+f(R23,kx2[i],ky2[i])+f(
        #tAA = VA+t2*g(kx2[i],ky2[i])
        #tAA = VA+t2*gf(kx1[i],ky1[i])

        H = [[tAA,tAB],[np.conjugate(tAB),VB]]
        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wM2K1.append(w1[0])
        wM2K2.append(w1[1])

for i in range(len(kx3)):
        tAB = t*(f(R1,kx3[i],ky3[i])+f(R2,kx3[i],ky3[i])+f(R3,kx3[i],ky3[i]))
        tAA = VA+ t2*(f(R21,kx3[i],ky3[i])+f(R22,kx3[i],ky3[i])+f(R23,kx3[i],ky3[i])+f(
        #tAA = VA+t2*g(kx3[i],ky3[i])
        #tAA = VA+t2*gf(kx1[i],ky1[i])

        H = [[tAA,tAB],[np.conjugate(tAB),VB]]

        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wK2G1.append(w1[0])
        wK2G2.append(w1[1])



for k in range(len(r1)):
        kxG2M.append(r1[k])
        kxM2K.append(r2[k])
        kxK2G.append(r3[k])

plt.plot(kxG2M,wG2M1,'.b', linewidth=0.1)
plt.plot(kxG2M,wG2M2,'.b', linewidth=0.1)
plt.plot(kxM2K,wM2K1,'.b', linewidth=0.1)
plt.plot(kxM2K,wM2K2,'.b', linewidth=0.1)
plt.plot(kxK2G,wK2G1,'.b', linewidth=0.1)
plt.plot(kxK2G,wK2G2,'.b', linewidth=0.1)
plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
plt.xlabel('-----K-path----',fontsize=14)
plt.ylabel('-----Energy (eV)----',fontsize=14)
plt.title('Band structure of monolayer Graphene with $\Delta_{Gap}$ = 1.70 eV',fontsize

x = np.array([0 , 1.4746336294587137  ,  2.32601375  ,  4.028774])
plt.axvline(x=0.0, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=1.4746336294587137, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=2.32601375, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=4.028774, color='k', linestyle='--', linewidth=0.7)

my_xticks = ['$\Gamma$','M','K','$\Gamma$']
plt.xticks(x, my_xticks,fontsize=18)
plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
```

```
return array(a, dtype, copy=False, order=order)
```



Second nearest neighbor hopping term t=0.2eV and its effects:

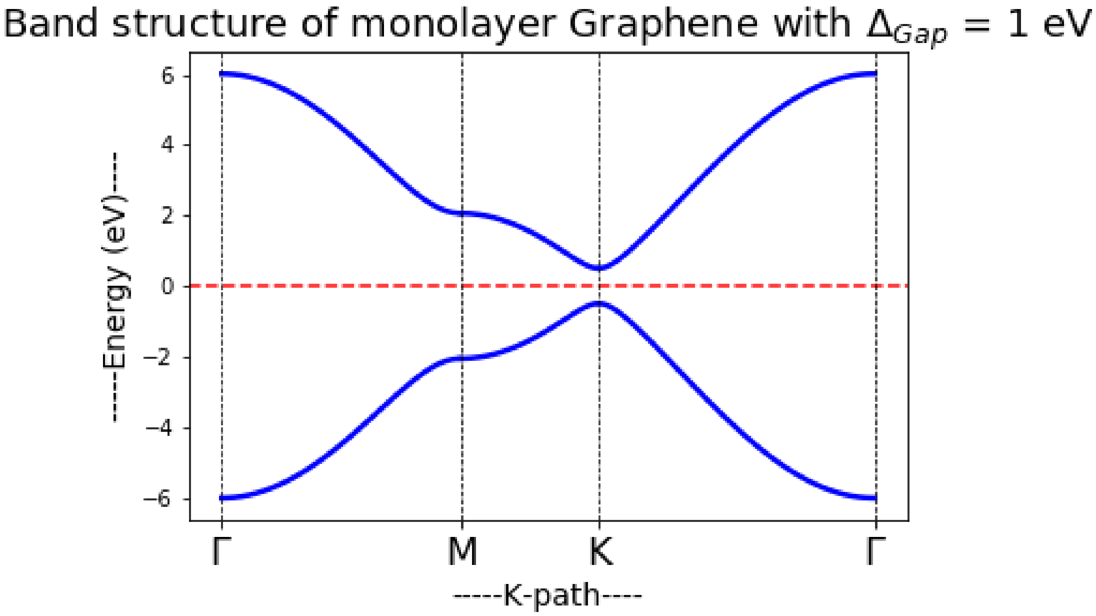
For the atom 'A', the interatomic hopping is with 3 first nearest neighbor 'B' atoms. The second nearest neighbors are 6 for atom 'A' and they belong to same sublattice. By introducing only the second nearest neighbor hopping (positive) , the electron hole symmetry is broken and the conduction band shifted below the fermi level, making the monolayer graphene a metal.

1.0.4 (d) Represent the bands in the presence of VA=0.5 eV and VB=0.5 eV site potential differences.

```
In [20]: import numpy as np
         from numpy import linalg as LA
         import matplotlib.pyplot as plt

         Dim = 2 # Number of atoms in primitive cell
         VA = 0.5
         VB = -0.5
         t = -2 #eV  -> nearest neighbor hopping term

         wG2M1  =  []
         wG2M2  =  []
         kxG2M  =[]

         wM2K1  =  []
         wM2K2  =  []
```

```
kxM2K = []

wK2G1 = []
wK2G2 = []
kxK2G = []

a = 2.46 # Ao ->lattice parameter
acc =  a/np.sqrt(3) # bond-length

R1 = [(a/np.sqrt(3)),0]
R2 = [(-a/(2*np.sqrt(3))),a/2]
R3 = [(-a/(2*np.sqrt(3))),-a/2]

H = np.zeros((Dim,Dim)) # Hamiltonian


Sym_pts = [[0,0],[((2*np.pi)/(np.sqrt(3)*a)),0],[((2*np.pi)/(np.sqrt(3)*a)),((2*np.pi)/
#           G                   M                              K



kx1 = np.linspace(Sym_pts[0][0],Sym_pts[1][0], 100)
ky1 = np.linspace(Sym_pts[0][1],Sym_pts[1][1], 100)

kx2 = np.linspace(Sym_pts[1][0],Sym_pts[2][0], 100)
ky2 = np.linspace(Sym_pts[1][1],Sym_pts[2][1], 100)

kx3 = np.linspace(Sym_pts[2][0],Sym_pts[3][0], 100)
ky3 = np.linspace(Sym_pts[2][1],Sym_pts[3][1], 100)


r1 =  np.linspace(0,(2*np.pi/(np.sqrt(3)*a)),100)
r2 =  np.linspace((2*np.pi/(np.sqrt(3)*a)),((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),1
r3 =  np.linspace(((2*np.pi/(np.sqrt(3)*a))+(2*np.pi/(3*a))),((2*np.pi/(np.sqrt(3)*a))+


def f(R,x,y):
    return (np.exp(1j*np.dot([x,y],[R[0],R[1]])))



for i in range(len(kx1)):
        tAB = t*(f(R1,kx1[i],ky1[i])+f(R2,kx1[i],ky1[i])+f(R3,kx1[i],ky1[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]

        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wG2M1.append(w1[0])
        wG2M2.append(w1[1])

for i in range(len(kx2)):
        tAB = t*(f(R1,kx2[i],ky2[i])+f(R2,kx2[i],ky2[i])+f(R3,kx2[i],ky2[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]

        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wM2K1.append(w1[0])
        wM2K2.append(w1[1])

for i in range(len(kx3)):
        tAB = t*(f(R1,kx3[i],ky3[i])+f(R2,kx3[i],ky3[i])+f(R3,kx3[i],ky3[i]))
        H = [[VA,tAB],[np.conjugate(tAB),VB]]

        w1, v1 = LA.eig(H)
       # w1, v1 = LA.eig(H)
        wK2G1.append(w1[0])
        wK2G2.append(w1[1])



for k in range(len(r1)):
        kxG2M.append(r1[k])
        kxM2K.append(r2[k])
        kxK2G.append(r3[k])

plt.plot(kxG2M,wG2M1,'-b', linewidth=2.5)
plt.plot(kxG2M,wG2M2,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K1,'-b', linewidth=2.5)
plt.plot(kxM2K,wM2K2,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G1,'-b', linewidth=2.5)
plt.plot(kxK2G,wK2G2,'-b', linewidth=2.5)
plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
plt.xlabel('-----K-path----',fontsize=14)
plt.ylabel('-----Energy (eV)----',fontsize=14)
plt.title('Band structure of monolayer Graphene with $\Delta_{Gap}$ = 1 eV',fontsize=18

x = np.array([0 , 1.4746336294587137  ,  2.32601375  ,  4.028774])
plt.axvline(x=0.0, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=1.4746336294587137, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=2.32601375, color='k', linestyle='--', linewidth=0.7)
plt.axvline(x=4.028774, color='k', linestyle='--', linewidth=0.7)

my_xticks = ['$\Gamma$','M','K','$\Gamma$']
plt.xticks(x, my_xticks,fontsize=18)
plt.show()
```

```
/home/tcsl05/anaconda3/lib/python3.6/site-packages/numpy/core/numeric.py:492: ComplexWarning: Ca
  return array(a, dtype, copy=False, order=order)
```



1.0.5 Problem 2. Repeat problem 1 using pybinding, adding appropriate modifications to the following example code provided in the tutorial and 'Additional topics/ Lattice specification' section.

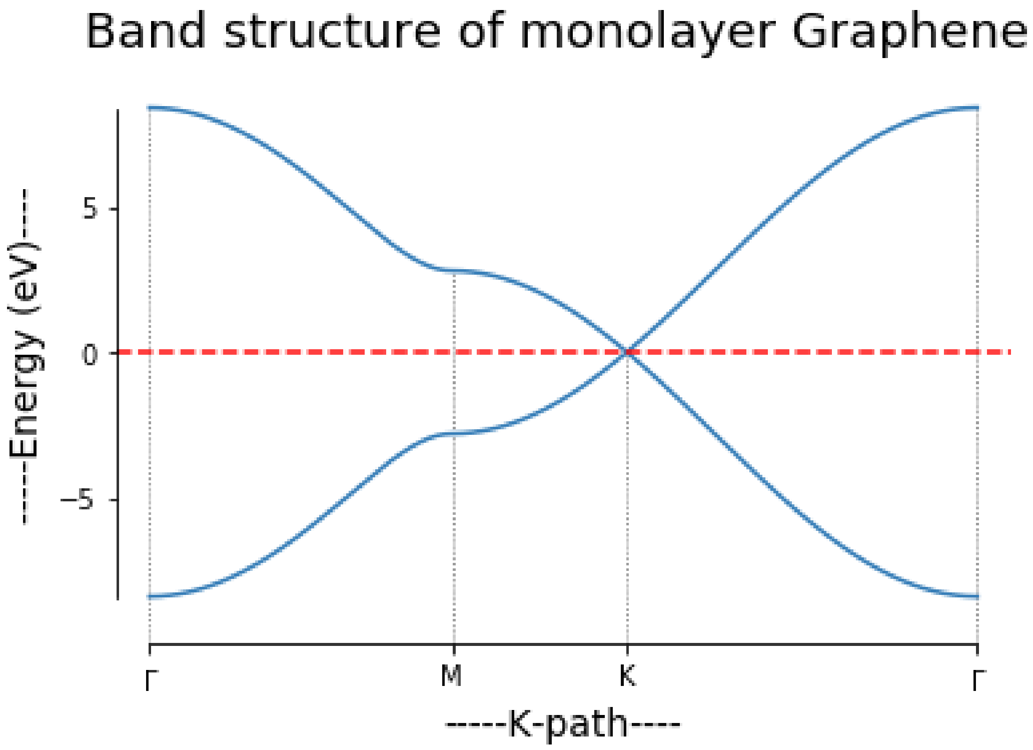
1.0.6 (a) Represent the bands as surface plots and along the symmtry lines in the Brillouin zone.

at t = -2.6 eV

```
In [7]: import pybinding as pb
        import numpy as np
        import matplotlib.pyplot as plt
        from math import sqrt, pi


        # System parameters
        a = 0.24595    # [nm] unit cell length
        a_cc = 0.142   # [nm] carbon-carbon distance
        t = -2.6       # [eV] nearest neighbour hopping


        # Reciprocal lattice symmetry points
        Gamma = [0, 0]
```

```
K1 = [-4*pi / (3*sqrt(3)*a_cc), 0]
M  = [0, 2*pi / (3*a_cc)]
K2 = [2*pi / (3*sqrt(3)*a_cc), 2*pi / (3*a_cc)]

from pybinding.repository import graphene

model = pb.Model(graphene.monolayer(), pb.translational_symmetry())
solver = pb.solver.lapack(model)

a_cc = graphene.a_cc
bands = solver.calc_bands(Gamma, M, K2,Gamma)
bands.plot(point_labels=[r'$\Gamma$', 'M', 'K',r'$\Gamma$'])
plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
plt.xlabel('-----K-path----',fontsize=14)
plt.ylabel('-----Energy (eV)----',fontsize=14)
plt.title('Band structure of monolayer Graphene ',fontsize=18)
```

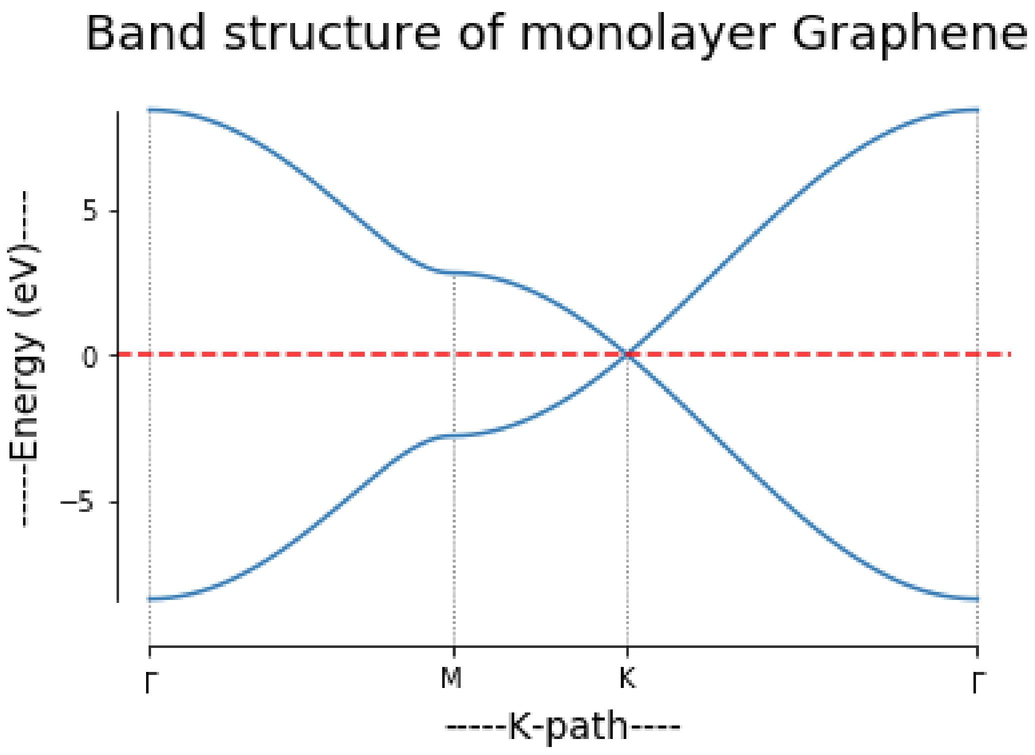
```
Out[7]: Text(0.5,1,'Band structure of monolayer Graphene ')
```



at t = -3.4 eV

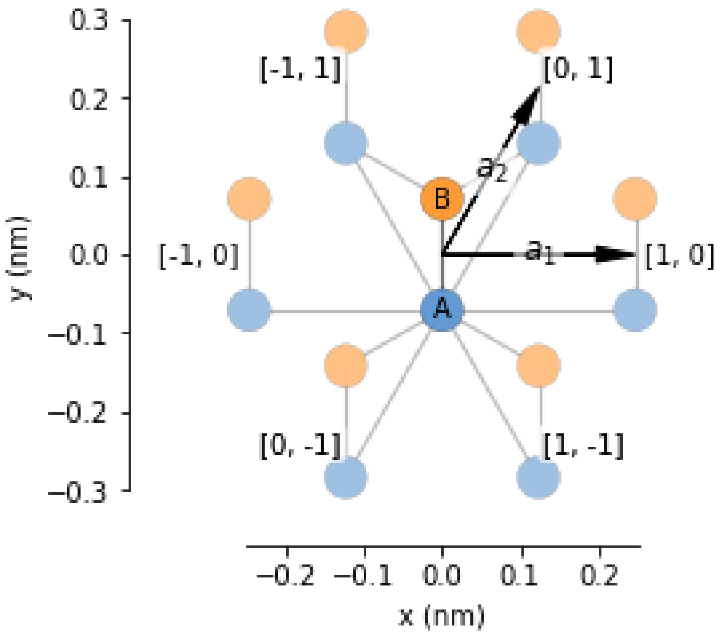
```
In [3]: import pybinding as pb
        import numpy as np
        import matplotlib.pyplot as plt
        from math import sqrt, pi


        # System parameters
        a = 0.24595   # [nm] unit cell length
        a_cc = 0.142  # [nm] carbon-carbon distance
        t = -3.4      # [eV] nearest neighbour hopping


        # Reciprocal lattice symmetry points
        Gamma = [0, 0]
        K1 = [-4*pi / (3*sqrt(3)*a_cc), 0]
        M  = [0, 2*pi / (3*a_cc)]
        K2 = [2*pi / (3*sqrt(3)*a_cc), 2*pi / (3*a_cc)]

        from pybinding.repository import graphene

        model = pb.Model(graphene.monolayer(), pb.translational_symmetry())
        solver = pb.solver.lapack(model)

        a_cc = graphene.a_cc
        bands = solver.calc_bands(Gamma, M, K2,Gamma)
        bands.plot(point_labels=[r'$\Gamma$', 'M', 'K',r'$\Gamma$'])
        plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
        plt.xlabel('-----K-path----',fontsize=14)
        plt.ylabel('-----Energy (eV)----',fontsize=14)
        plt.title('Band structure of monolayer Graphene ',fontsize=18)
```

```
Out[3]: Text(0.5,1,'Band structure of monolayer Graphene ')
```



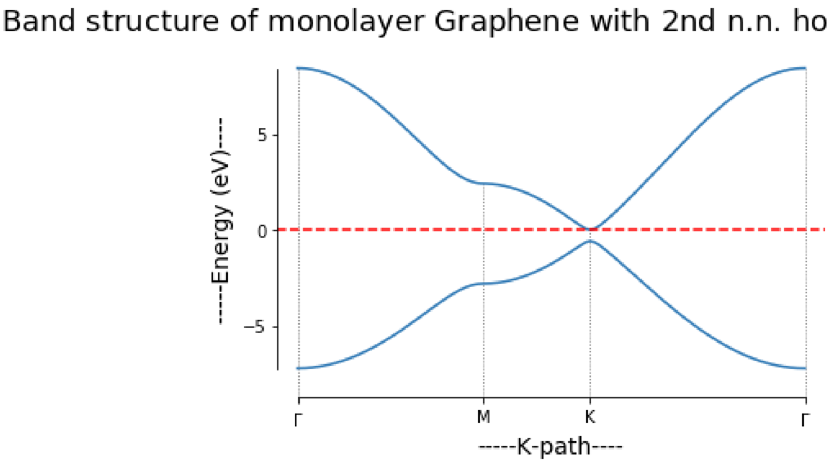
1.0.7 (c) Represent the bands considering an additional second nearest neighbor hopping term t=0.2 and describe its effects.

```
In [33]: import pybinding as pb
         import numpy as np
         import matplotlib.pyplot as plt
         from math import sqrt, pi


         # System parameters
         a = 0.24595    # [nm] unit cell length
         a_cc = 0.142   # [nm] carbon-carbon distance
         t = -2.6       # [eV] nearest neighbour hopping
         t2 = 0.2

         # Reciprocal lattice symmetry points
         Gamma = [0, 0]
         K1 = [-4*pi / (3*sqrt(3)*a_cc), 0]
         M  = [0, 2*pi / (3*a_cc)]
         K2 = [2*pi / (3*sqrt(3)*a_cc), 2*pi / (3*a_cc)]

         def monolayer_graphene(onsite_energy=[0, 0]):
             lat = pb.Lattice(a1=[a, 0], a2=[a/2, a/2 * sqrt(3)])
             lat.add_sublattices(('A', [0, -a_cc/2], onsite_energy[0]),
                                 ('B', [0,  a_cc/2], onsite_energy[1]))
             lat.add_hoppings(([0,  0], 'A', 'B', t),
                              ([1, -1], 'A', 'B', t),
                              ([0, -1], 'A', 'B', t),
                              ([-1, 0], 'A', 'A', t2),
                              ([0, -1], 'A', 'A', t2),
                              ([1, -1], 'A', 'A', t2),
                             )
             return lat

         lattice = monolayer_graphene()
         lattice.plot()
```



```
In [34]: model = pb.Model(
             monolayer_graphene(),  # eV
             pb.translational_symmetry()
         )
         solver = pb.solver.lapack(model)
         bands = solver.calc_bands(Gamma, M, K2,Gamma)
         bands.plot(point_labels=[r'$\Gamma$', 'M', 'K',r'$\Gamma$'])

         plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
         plt.xlabel('-----K-path----',fontsize=14)
         plt.ylabel('-----Energy (eV)----',fontsize=14)
         plt.title('Band structure of monolayer Graphene with 2nd n.n. hopping $t_{AA}$ = 0.2 eV
```

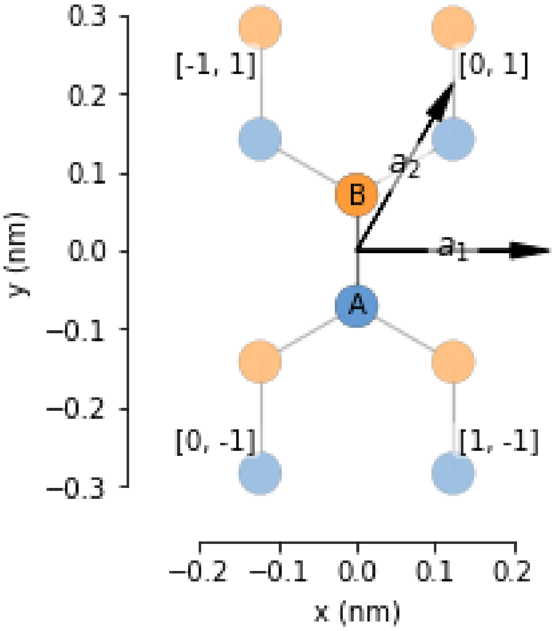
```
Out[34]: Text(0.5,1,'Band structure of monolayer Graphene with 2nd n.n. hopping $t_{AA}$ = 0.2 e
```



1.0.8 (d) Represent the bands in the presence of VA=0.5 eV and VB=0.5 eV site potential differences.

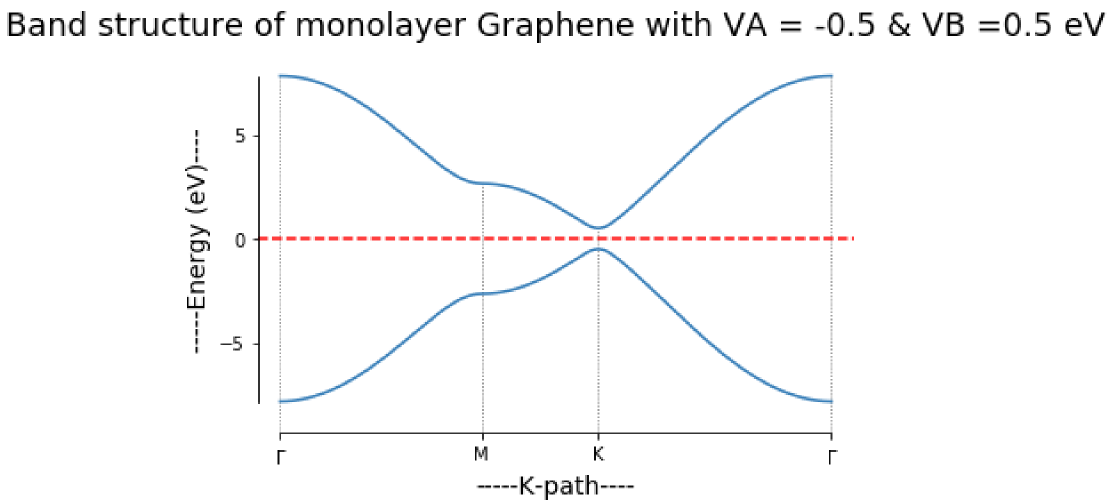
```
In [2]: import pybinding as pb
        import numpy as np
        import matplotlib.pyplot as plt
        from math import sqrt, pi


        # System parameters
        a = 0.24595    # [nm] unit cell length
        a_cc = 0.142   # [nm] carbon-carbon distance
        t = -2.6       # [eV] nearest neighbour hopping
        t2 = 0.2

        # Reciprocal lattice symmetry points
        Gamma = [0, 0]
        K1 = [-4*pi / (3*sqrt(3)*a_cc), 0]
        M  = [0, 2*pi / (3*a_cc)]
        K2 = [2*pi / (3*sqrt(3)*a_cc), 2*pi / (3*a_cc)]


        def monolayer_graphene(onsite_energy=[-0.5, 0.5]):
            lat = pb.Lattice(a1=[a, 0], a2=[a/2, a/2 * sqrt(3)])
            lat.add_sublattices(('A', [0, -a_cc/2], onsite_energy[0]),
                                ('B', [0,  a_cc/2], onsite_energy[1]))
```

```
    lat.add_hoppings(([0,  0], 'A', 'B', t),
                     ([1, -1], 'A', 'B', t),
                     ([0, -1], 'A', 'B', t))
    return lat

lattice = monolayer_graphene()
lattice.plot()
```



```
In [5]: model = pb.Model(
            monolayer_graphene(onsite_energy=[-0.5, 0.5]),  # eV
            pb.translational_symmetry()
        )
        solver = pb.solver.lapack(model)
        bands = solver.calc_bands(Gamma, M, K2,Gamma)
        bands.plot(point_labels=[r'$\Gamma$', 'M', 'K',r'$\Gamma$'])

        plt.axhline(y=0.0, color='r', linestyle='--', linewidth=1.5)
        plt.xlabel('-----K-path----',fontsize=14)
        plt.ylabel('-----Energy (eV)----',fontsize=14)
        plt.title('Band structure of monolayer Graphene with VA = -0.5 & VB =0.5 eV',fontsize=18
```

```
Out[5]: Text(0.5,1,'Band structure of monolayer Graphene with VA = -0.5 & VB =0.5 eV')
```


Band structure of monolayer Graphene with VA = -0.5 & VB =0.5 eV
5
0
−5
----Energy (eV)----
Γ
M
K
Γ
-----K-path----