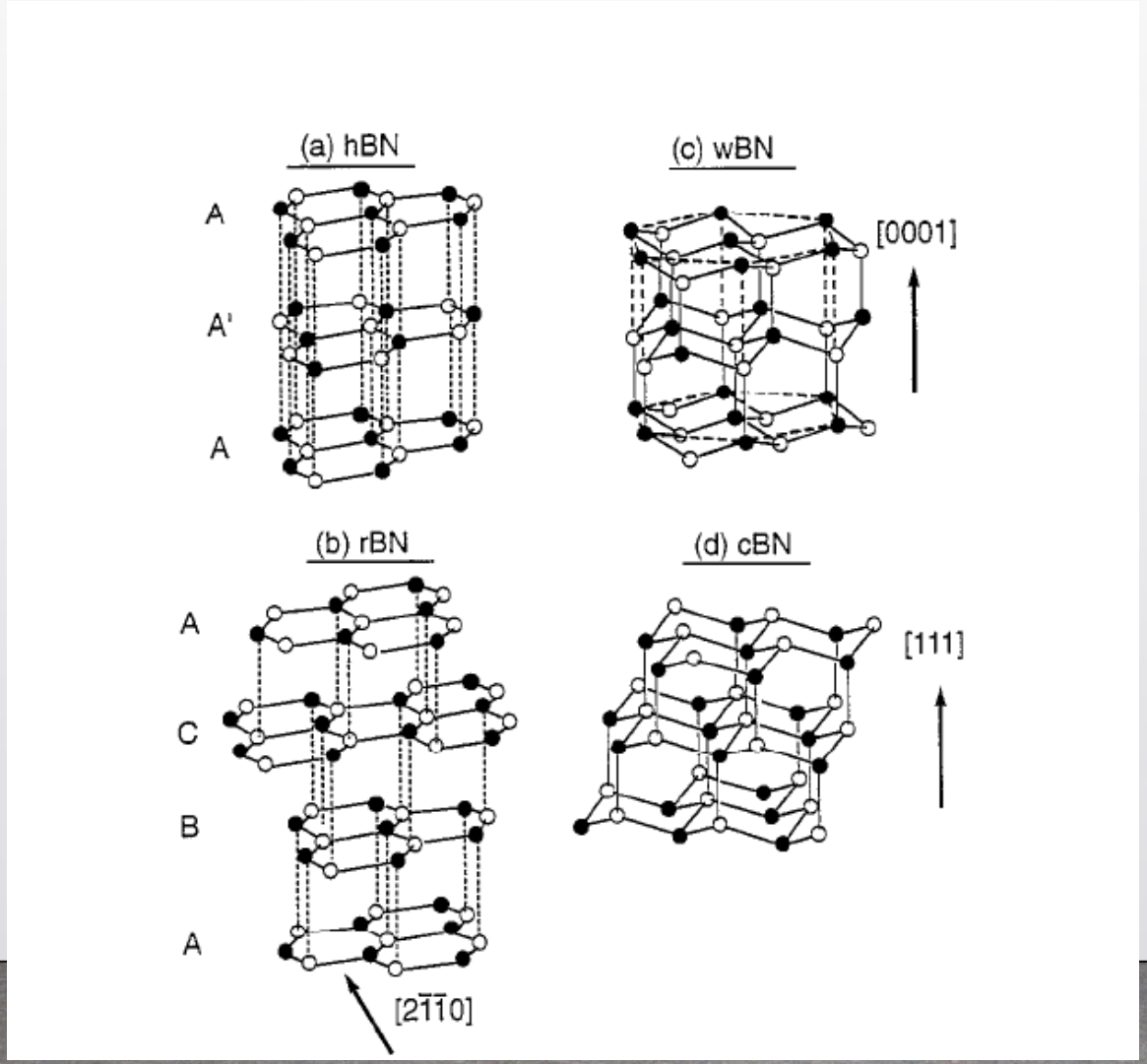
Nitruro de Boro h-BN

Arroyo Miranda, Omar Everardo

De León Piña, Alejandro

Estructuras cristalinas del nitruro de boro



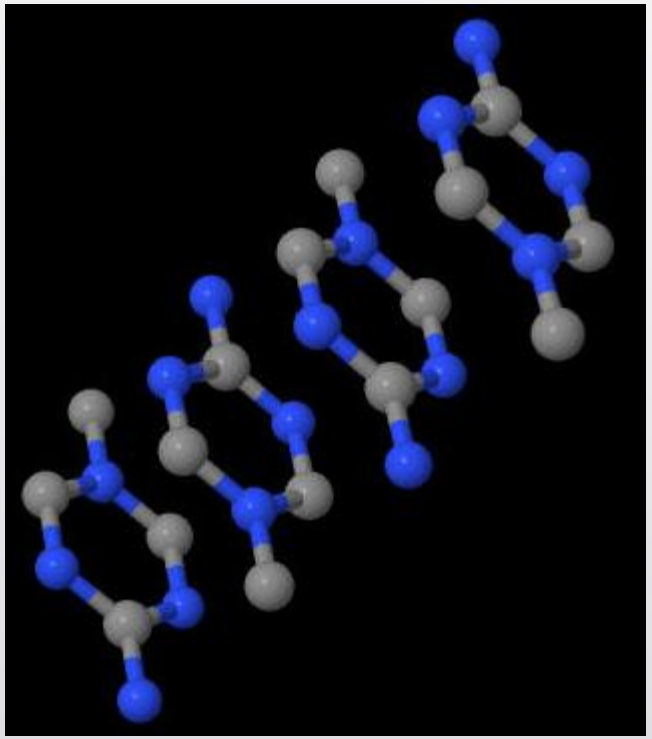
Películas delgadas de nitruro de boro.
https://www.researchgate.net/publication/277197056

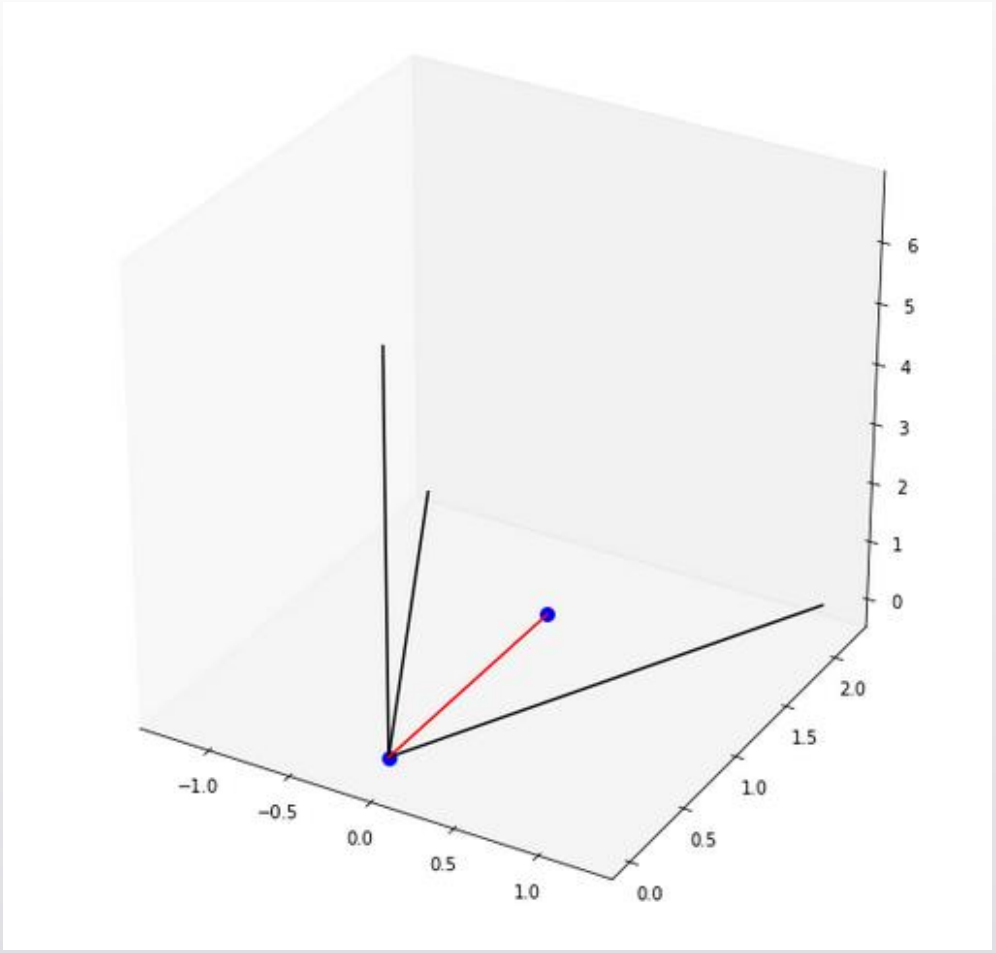
Propiedades del cristal

- Vectores de traslación: $t_1 = a\left(\frac{1}{2}, \frac{\sqrt{3}}{2}, 0\right)$, $t_2 = a\left(\frac{-1}{2}, \frac{\sqrt{3}}{2}, 0\right)$, $t_3 = c(0,0,1)$

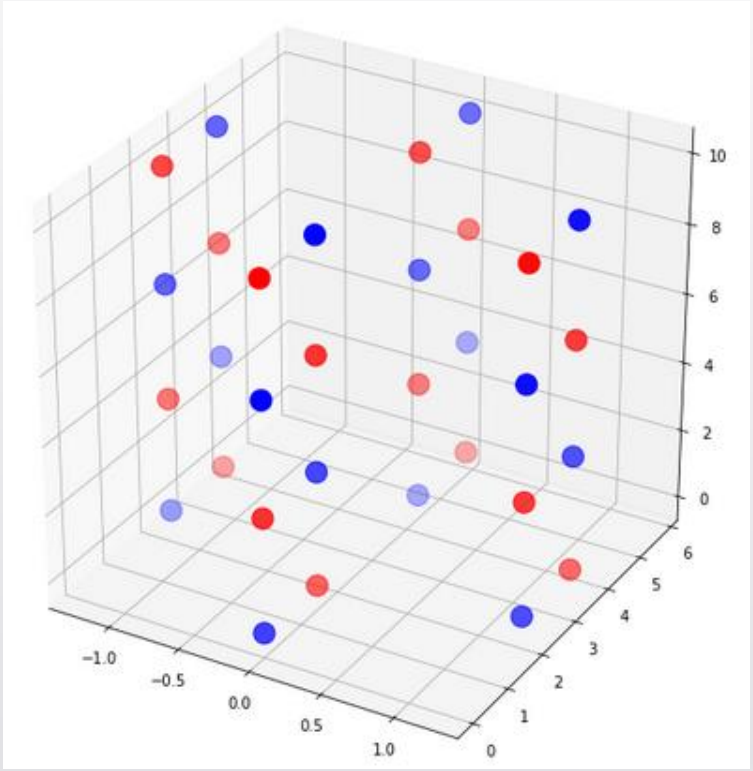
- Vectores de la Base:$d_1 = (0,0,0)$, $d_2 = a\left(0, \frac{\sqrt{3}}{3}, 0\right)$, $d_3 = c\left(0,0, \frac{1}{2}\right)$, $d_4 = (0, \frac{\sqrt{3}}{3}a, \frac{c}{2})$
- Parámetros de red $a = 2.5043A$, $c = 6.656\ A$

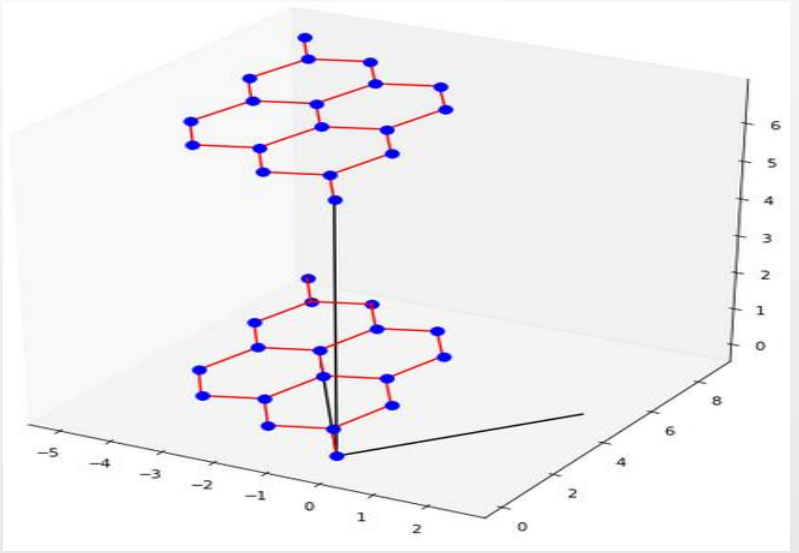
Red y Base Atómica





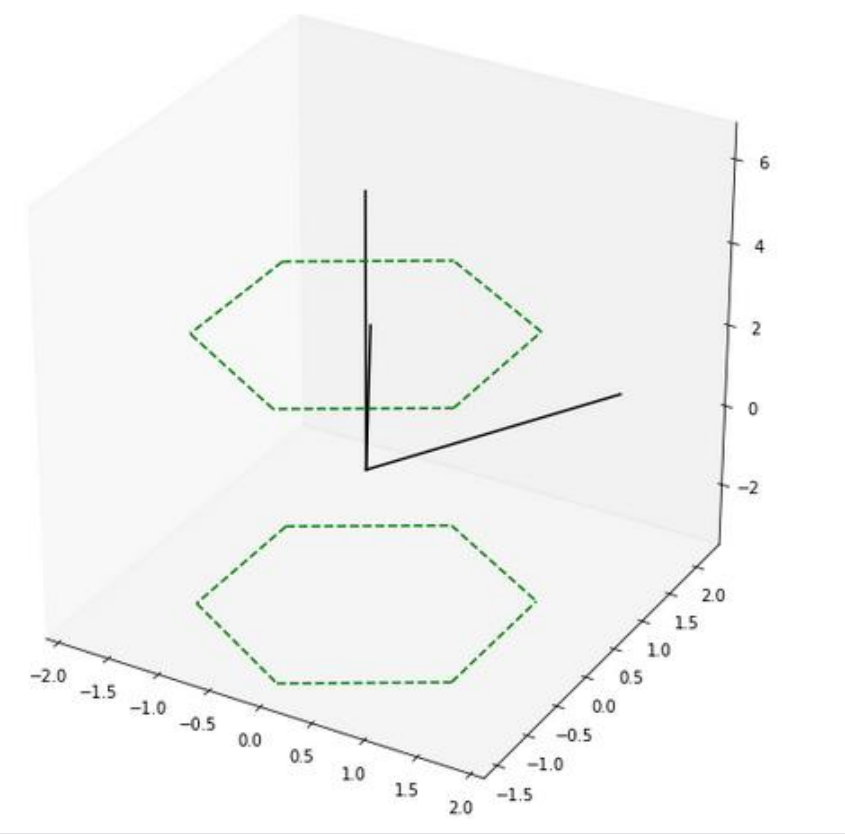
Supercelda y traslación



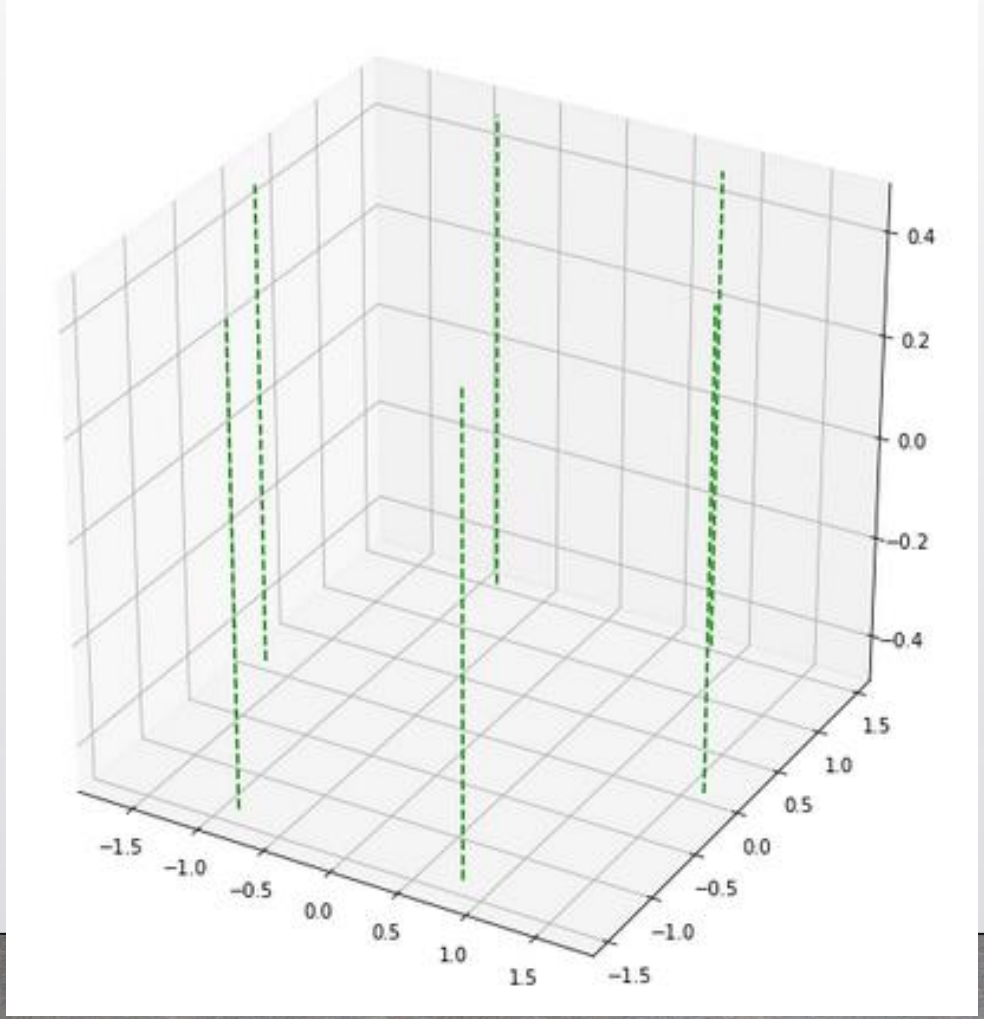


In [17]:
```
class CrystalNB(object):

    def __init__(self, t1, t2 ,t3,d1,d2, d3,d4,n):

        #Para vizualizar el cristal, crear superceldas y aplicar las operaciones de traslación

        #se generan los posiciones de Boro para un plano para superceldas de nxn
        positionsB1=[]
        for i in range(n):
            for k in range(n):
                for j in range(n):
                    positionsB1 += [(d1)+k*(t1)+i*(t2)+j*(t3)]
        #se generan las posiciones de Boro para el plano subsiguiente para superceldas
```

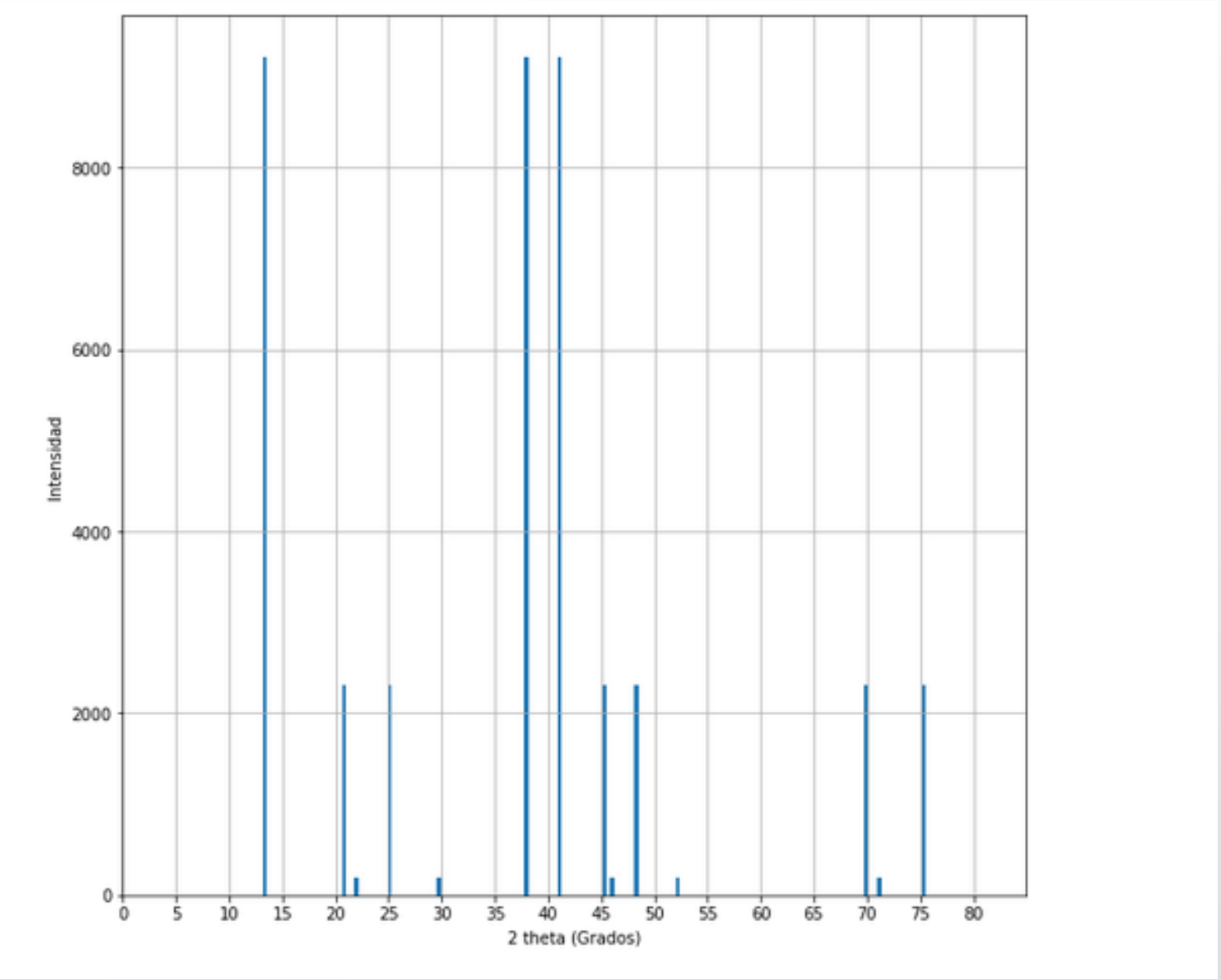
Celda de Wigner-Seitz



Primera zona de Brillouin

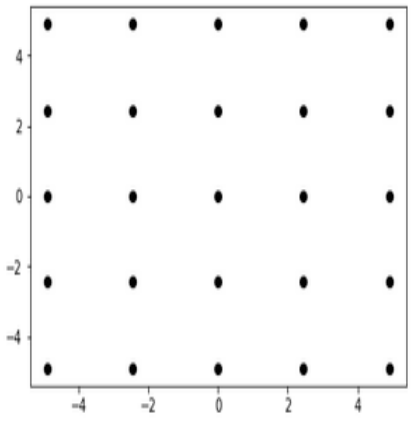


Patrón de difracción



Fonones red 2D Monoatómica

```
In [35]: plt.plot(N[:,0],N[:,1], 'o', color='black'); #vizualización del cristal
```



Se construye la matriz dinámica para la primera trayectoria, de Gamma a X

```
Dxx=[]
Dyy=[]
Dxy=[]
Dyx=[]
Dxx1=[]
Dyy1=[]
Dxy1=[]
Dyx1=[]
kx=np.linspace(0,np.pi/a,num=20)  #puntos k en dirección x
ky= np.linspace(0,np.pi/a,num=20)
for k in range(0,len(kx)):   #primera trayectoria horizontal.
    for i in range(9):
        Dxx += [(1/Masa)*np.concatenate((xxp,xxs),axis=0)[i]*(np.exp(-(np.dot(np.concatenate((P1,S),axis=0)[i],
for k in range(0,len(kx)):
    for i in range(9):    #se hace el cálculo de la matriz dinámica
        Dyy += [(1/Masa)*np.concatenate((yyp,yys),axis=0)[i]*(np.exp(-(np.dot(np.concatenate((P1,S),axis=0)[i],
for k in range(0,len(kx)):
    for i in range(9):    #se hace el cálculo de la matriz dinámica
        Dxy += [(1/Masa)*np.concatenate((xyp,xys),axis=0)[i]*(np.exp(-(np.dot(np.concatenate((P1,S),axis=0)[i],
for k in range(0,len(kx)):
    for i in range(9):    #se hace el cálculo de la matriz dinámica
        Dyx += [(1/Masa)*np.concatenate((yxp,yxs),axis=0)[i]*(np.exp(-(np.dot(np.concatenate((P1,S),axis=0)[i],
#se hacen las sumas siguientes para obtener los elementos de la matriz
for j in range(1,len(Dxx),9):
         Dxx1.append(sum(Dxx[j:j+20]))
         np.asarray(Dxx1)
for j in range(1,len(Dyy),9):
         Dyy1.append(sum(Dyy[j:j+20]))
         np.asarray(Dyy1)
for j in range(1,len(Dxy),9):
         Dxy1.append(sum(Dxy[j:j+20]))
         np.asarray(Dxy1)
for j in range(1,len(Dyx),9):
         Dyx1.append(sum(Dyx[j:j+20]))
         np.asarray(Dyx1)
#diagonalización de la matriz
```