Fonones y espectroscopía Raman en semiconductores bidimensionales.

casimir

March 3, 2021

Contents

Abstract

Los materiales bidimensionales (2D) como el grafeno son de gran interés tanto por sus propiedades físicas exclusivas como por sus aplicaciones potenciales. El estudio de la dinámica de la red cristalina (fonones) de estos materiales es un requisito previo para entender su estabilidad estructural y propiedades térmicas, así como sus propiedades de transporte y ópticas.

Este Trabajo de Fin de Grado consiste en la computación de los modos vibracionales de materiales semiconductores 2D y su correlación con los observables relevantes para la interpretación de los experimentos de dispersión de luz.

La redacción del TFG todavía se encuentra en una versión muy preliminar, de echo es más un bloc de notas que lo que espero que sea el TFG final.

1 TODO Introduccion

Las vibraciones reticulares están regidas por las fuerzas que experimentan los átomos cuando se desplazan de su posición de equilibrio. La primera hipótesis es que cada átomo tiene una posicion de equilibrio en el cristal, y consideraremos que estos átomos vibran con una amplitud pequeña alrededor de su posición de equilibrio, de manera que el sólido se encuentra en estados que corresponden a lo que macroscópicamente se conoce como *la region de comportamiento elástico lineal*, donde se verifica la ley de Hooke.

Podremos por tanto aproximar la energia potencial de interaccion por el termino armónico de su desarrolo en serie de potencias del desplazamiento, y la fuerza resultante es por tanto una funcion aproximadamente lineal del desplazamiento

2 TODO Desarrollo del trabajo

2.1 TODO Formular la matriz dinámica para el BN monolayer

En el caso del BN monolayer estamos tratando con un cristal bidimensional del base diatómica, cuya celda unidad viene dada por:

$$\vec{a}_1 = a(1,0); \qquad \vec{a}_2 = a\left(-\frac{1}{2}, \frac{\sqrt{3}}{2}\right) \tag{1}$$

```
import numpy as np
from  numpy  import array, sqrt, sort
import pandas as pd
import matplotlib
from matplotlib import pyplot as plt

a=1
a1=np.array([a,0])
a2=np.array([-a/2,sqrt(3)*a/2])
```

Podemos comprobar que efectivamente se trata de una celdilla hexagonal, pues los dos vectores base forman un angulo de $2\pi/3$ rad.

```
from numpy import vdot
from numpy import pi
from numpy import arccos
from numpy.linalg import norm
print(arccos(vdot(a1,a2)/(norm(a1)*norm(a2))))
```

```
2.0943951023931957
```

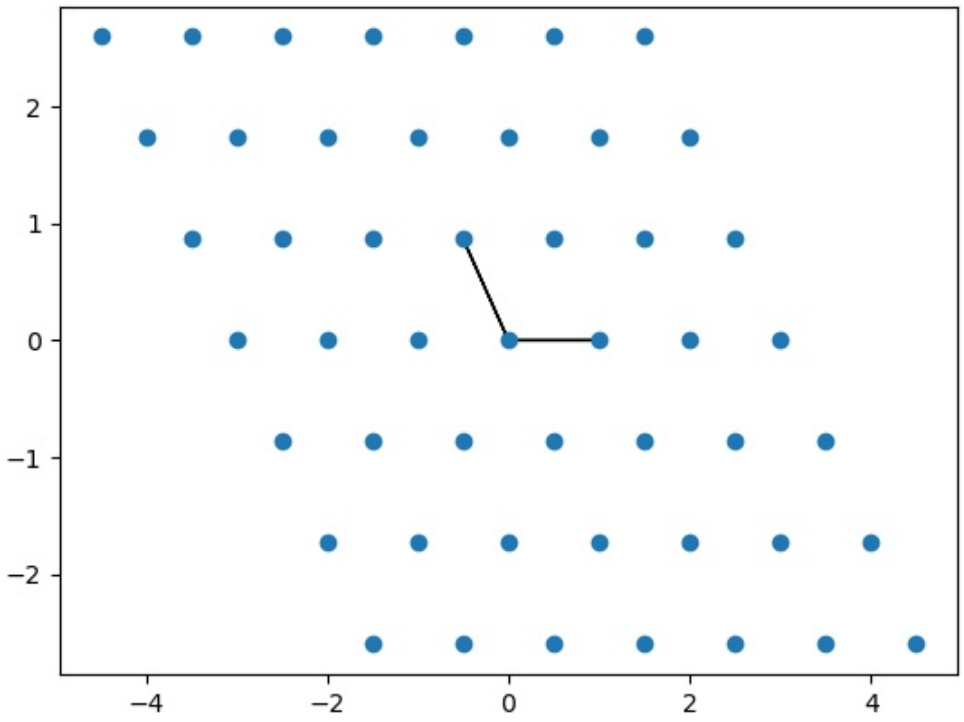
Numeraremos las celdillas unidad con un índice vectorial $\vec{l} = (l_1, l_2)$.

Las posiciones de los nudos son $\vec{R}_{\vec{l}} = l_1\vec{a}_1 + l_2\vec{a}_2$.

Visualizamos una región de la red hexagonal, con los correspondientes nudos (que no átomos), así como la correspondiente celda unidad,

```
def nudo(l1,l2):
 return l1*a1+l2*a2

reddenudos=array([nudo(l1,l2) for l1 in range(-3, 4) for l2 in range(-3,4)])

x = reddenudos[:,0]
y = reddenudos[:,1]
plt.plot(x,y,"o")
ax = plt.axes()
ax.arrow(nudo(0,0)[0],nudo(0,0)[1],nudo(1,0)[0],nudo(1,0)[1])
```

```
ax.arrow(nudo(0,0)[0],nudo(0,0)[1],nudo(0,1)[0],nudo(0,1)[1])
plt.savefig("Graficas/Reddenudos.jpg")
plt.close()
```



Para calcular los modos de vibración por primeros principios debemos determinar primero las posiciones atómicas de equilibrio en la celda unidad **nota: proporcionadas como datos**

Los átomos estan situados en:

$$\begin{aligned} \vec{R}_B &= \frac{1}{3}\vec{a_1} + 2\vec{a_2} \\ \vec{R}_N &= \frac{2}{3}\vec{a_1} + \frac{1}{3}\vec{a_2} \end{aligned} \tag{2}$$

```
R_B=1/3*a1+2/3*a2
R_N=2/3*a1+1/3*a2
```

```
## La gráficas las he echo usando sagemath, las volvere a hacer con matplotlib, o plotly o otra
#  Celda=plot(arrow((0,0),(a1/a),color="red")+
#         arrow((0,0),(a2/a),color="red")+
#         line([(a1/a),(a1/a+a2/a)],linestyle="--",color="red")+
#         line([(a2/a),(a2/a+a1/a)],linestyle="--",color="red")+
#         point(R_B/a, size=140,color="green")+
#         point(R_N/a, size=130,color="red"))
#  Celda.save("Celda.png")
```

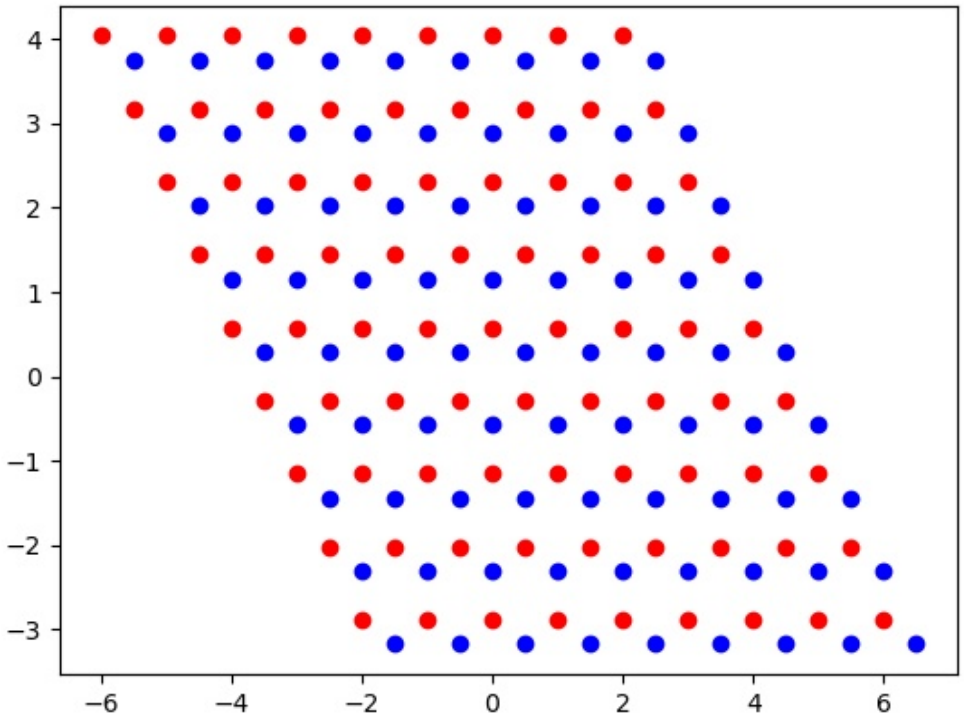
2.1.1 Cristales bidimensionales con base diatómica

Numeraremos las celdillas unidad con un índice vectorial $\vec{l} = (l_1, l_2)$.

Las posiciones de los nudos son $\vec{R}_{\vec{l}} = l_1\vec{a}_1 + l_2\vec{a}_2$. Las posiciones de equilibrio de los átomos de la base respecto de su nudo son $\vec{R}^0_\nu$, con $\nu = 1, 2$, puesto que la base tiene 2 átomos diferente, el 1 hará referencia a los átomos de B y 2 a los de átomos de N. Notemos que aunque los átomos fuesen idénticos, tenemos que especificar a que átomo de la base nos referimos, puesto que no ocupan posiciones equivalentes. Las posiciones de equilibrio de los átomos $\vec{R}_{\nu,\vec{l}} = \vec{R}_{\vec{l}} + \vec{R}^0_\nu$ así como los desplazamientos

atómicos $\vec{u}_{\nu,\vec{l}}$ quedarán por tanto identificados por medio de dos índices. La fuerza que ejerce el átomo $\nu,\vec{l}$ sobre el átomo $\nu',\vec{0}$ tiene aproximádamente la dirección determinada por las posiciones de equilibrio de estos átomos. Esta dirección es la del vector $\vec{R}_{\nu',\nu,\vec{l}} = \vec{R}_{\vec{l}} + \vec{R}^0_{\nu',\nu}$ donde $\vec{R}^0_{\nu',\nu} \equiv \vec{R}^0_{\nu} - \vec{R}^0_{\nu'}$.

```
#Posiciones de equilibrio de los átomos

def R_nu(nu,l1,l2):
  if nu == 1:
    return l1*a1+l2*a2+R_B

  elif nu == 2:
    return l1*a1+l2*a2+R_N

  else:
    print("Error, nu solo puede ser 1 (Boro) o 2 (Nitrogeno)")


AtomosB=array([R_nu(1,l1,l2) for l1 in range(-4, 5) for l2 in range(-4,5)])

AtomosN=array([R_nu(2,l1,l2) for l1 in range(-4, 5) for l2 in range(-4,5)])

xB = AtomosB[:,0]
yB = AtomosB[:,1]
plt.plot(xB,yB,"o",color="red")

xN = AtomosN[:,0]
yN = AtomosN[:,1]
plt.plot(xN,yN,"o",color="blue")

plt.savefig("Graficas/Reddeatomos.jpg")
plt.close()
```



Las dimensiones del cristal son $L_1 = N_1 a_1$ y $L_2 = N_2 a_2$, donde N_i $(i = 1, 2)$ es el número de celdillas en la dirección $\hat{a}_i$. El cristal tiene tiene $N = N_1 N_2$ celdillas unidad primitivas y $2N$ átomos.

La idea básica es que si la base tiene N_ν átomos entonces debemos plantear y resolver las ecuaciones de movimiento de los N_ν átomos de la base de la celdilla $\vec{0}$, por lo tanto en el caso que estamos estudiando debemos resolver 2 ecuaciones vectoriales de movimiento: una para el átomo de B y la otra para el de N.

La fuerza que ejerce el átomo $\nu, \vec{l}$ sobre el átomo $\nu', \vec{0}$ se puede expresar de manera aproximada como:

$$F_{\nu',\vec{0},\nu,\vec{l}} = \alpha_{\nu',\nu,\vec{l}} \left(\hat{R}_{\nu',\nu,\vec{l}} \times \hat{R}_{\nu',\nu,\vec{l}} \right) \cdot \left(\vec{u}_{\nu,\vec{l}} - \vec{u}_{\nu',\vec{0}} \right)$$

donde $\hat{R}_{\nu',\nu,\vec{l}}$ es el vector unitario en la dirección $\vec{R}_{\nu',\nu,\vec{l}}$

La ecuación de movimiento del átomo $\nu', \vec{0}$ es por lo tanto:

$$m_{\nu'} \ddot{\vec{u}}_{\nu',\vec{0}} = \sum_{\nu,\vec{l}} \alpha_{\nu',\nu,\vec{l}} \left(\hat{R}_{\nu',\nu,\vec{l}} \times \hat{R}_{\nu',\nu,\vec{l}} \right) \cdot \left(\vec{u}_{\nu,\vec{l}} - \vec{u}_{\nu',\vec{0}} \right)$$

Buscaremos soluciones de la forma:

$$u_{\nu,\vec{l}} = \vec{A}_\nu e^{i(\vec{q}\cdot\vec{R}_{\vec{l}} - \omega t)}$$

donde $\vec{A}_\nu$ es el *vector de polarización* que determina la amplitud y dirección de vibración de los átomos de tipo ν . Es importante apreciar que se necesitan tantas amplitudes de vibración como átomos tenga la base porque estos no ocupan posiciones equivalentes y describen vibraciones distintas. Se deben cumplir así $N_\nu = 2$ ecuaciones vectoriales del tipo

$$\boxed{-m_{\nu'} \omega^2 \vec{A}_{\nu'} = \sum_{\nu,\vec{l}} \alpha_{\nu',\nu,\vec{l}} \left(\hat{R}_{\nu',\nu,\vec{l}} \times \hat{R}_{\nu',\nu,\vec{l}} \right) \cdot \left(\vec{A}_\nu e^{i\vec{q}\cdot\vec{R}_{\vec{l}}} - \vec{A}_{\nu'} \right)} \tag{3}$$

Como se trata de un sistema de ecuaciones lineales homogéneas, se debe cumplir la correspondiente ecuación secular, es decir, que el determinante de la matriz de dimensión $2N_\nu \times 2N_\nu$ $(2 \cdot 2 \times 2 \cdot 2)$ de los coeficientes $A_{\nu,i}$ en la ecuación 3 sea nula. Esta ecuación tiene $2N_\nu = 4$ soluciones que describen las 4 ramas de la relación de dispersión, es decir, las 4 frecuencias características de los 4 modos normales de vibración de vector de onda $\vec{q}$. Se cumple que el número total de modos normales de vibración coincide con el doble del número total de átomos, es decir, *el número total de modos normales de vibración coincide con el de grados de libertad de movimiento de los átomos*

Fijemonos en el átomo de Boro de la celdilla $\vec{0}$. Este átomo tiene 3 primeros vecinos situados en las celdillas $(0,0), (-1,0), (0,1)$, cuyas posiciones són

```
print([R_nu(2,0,0),R_nu(2,-1,0),R_nu(2,0,1)])
```

```
[array([0.5       , 0.28867513]), array([-0.5       ,  0.28867513]), array([0.        , 1.154700
```

y se encuentran a ua distancia

```
print([norm(R_nu(2,0,0)-R_nu(1,0,0)),norm(R_nu(2,-1,0)-R_nu(1,0,0)),norm(R_nu(2,0,1)-R_nu(1,0,0)
```

```
[0.5773502691896257, 0.5773502691896257, 0.5773502691896257]
```

del átomo de boro de la celdilla $\vec{0}$.

Puesto que vamos a considerar no sólo estos primeros vecinos, sino al menos hasta los cuartos vecinos, mejor genero un array con los datos que voy a necesitar $(\nu, \nu', \hat{R}_{\nu\prime,nu,\vec{l}}, \ldots$ ordenandolos según su distancia a los 2 átomos de la celda $l = \vec{0}$.

```
def propiedades_atomos(l1, l2):
   return [(k, m, i, j, R_nu(m,i,j), R_nu(m,i,j)-R_nu(k,0,0),norm(R_nu(m,i,j)-R_nu(k,0,0))) for

columnas = [r"$\nu$",r"$\nu\prime$",r"$l_1$", r"$l_2$", r"$\vec R_{\nu,\vec l}$",
            r"$\hat R_{\nu\prime,\nu,\vec l}$",'Distancia' ]

def Atomos(l1, l2):
   return pd.DataFrame(propiedades_atomos(l1, l2),columns=columnas)

Atomos(2,2).sort_values(by=['Distancia']).to_latex(escape=False,float_format="{:0.4f}".format,in
```

ν	$\nu\prime$	l_1	l_2	$\vec{R}_{\nu,\vec{l}}$	$\hat{R}_{\nu\prime,\nu,\vec{l}}$	Distancia
2	2	0	0	[0.5, 0.28867513459481287]	[0.0, 0.0]	0.0000
1	1	0	0	[0.0, 0.5773502691896257]	[0.0, 0.0]	0.0000
1	2	-1	0	[-0.5, 0.28867513459481287]	[-0.5, -0.28867513459481287]	0.5774
2	1	1	0	[1.0, 0.5773502691896257]	[0.5, 0.28867513459481287]	0.5774
1	2	0	0	[0.5, 0.28867513459481287]	[0.5, -0.28867513459481287]	0.5774
1	2	0	1	[0.0, 1.1547005383792515]	[0.0, 0.5773502691896257]	0.5774
2	1	0	-1	[0.5, -0.28867513459481287]	[0.0, -0.5773502691896257]	0.5774
2	1	0	0	[0.0, 0.5773502691896257]	[-0.5, 0.28867513459481287]	0.5774
1	1	-1	-1	[-0.5, -0.28867513459481287]	[-0.5, -0.8660254037844386]	1.0000
2	2	0	-1	[1.0, -0.5773502691896257]	[0.5, -0.8660254037844386]	1.0000
1	1	0	-1	[0.5, -0.28867513459481287]	[0.5, -0.8660254037844386]	1.0000
2	2	1	1	[1.0, 1.1547005383792515]	[0.5, 0.8660254037844386]	1.0000
1	1	0	1	[-0.5, 1.4433756729740643]	[-0.5, 0.8660254037844386]	1.0000
2	2	0	1	[0.0, 1.1547005383792515]	[-0.5, 0.8660254037844386]	1.0000
1	1	1	1	[0.5, 1.4433756729740643]	[0.5, 0.8660254037844386]	1.0000
2	2	-1	-1	[0.0, -0.5773502691896257]	[-0.5, -0.8660254037844386]	1.0000
1	1	-1	0	[-1.0, 0.5773502691896257]	[-1.0, 0.0]	1.0000
2	2	1	0	[1.5, 0.28867513459481287]	[1.0, 0.0]	1.0000
2	2	-1	0	[-0.5, 0.28867513459481287]	[-1.0, 0.0]	1.0000
1	1	1	0	[1.0, 0.5773502691896257]	[1.0, 0.0]	1.0000
2	1	-1	-1	[-0.5, -0.28867513459481287]	[-1.0, -0.5773502691896257]	1.1547
1	2	-1	1	[-1.0, 1.1547005383792515]	[-1.0, 0.5773502691896257]	1.1547
1	2	1	1	[1.0, 1.1547005383792515]	[1.0, 0.5773502691896257]	1.1547
2	1	1	1	[0.5, 1.4433756729740643]	[0.0, 1.1547005383792515]	1.1547
1	2	-1	-1	[0.0, -0.5773502691896257]	[0.0, -1.1547005383792515]	1.1547
2	1	1	-1	[1.5, -0.28867513459481287]	[1.0, -0.5773502691896257]	1.1547
1	2	1	2	[0.5, 2.02072594216369]	[0.5, 1.443375672974064]	1.5275
2	1	0	1	[-0.5, 1.4433756729740643]	[-1.0, 1.1547005383792515]	1.5275
1	2	0	-1	[1.0, -0.5773502691896257]	[1.0, -1.1547005383792515]	1.5275
2	1	2	1	[1.5, 1.4433756729740643]	[1.0, 1.1547005383792515]	1.5275
2	1	0	-2	[1.0, -1.1547005383792515]	[0.5, -1.4433756729740643]	1.5275
1	2	0	2	[-0.5, 2.02072594216369]	[-0.5, 1.443375672974064]	1.5275
1	2	-2	-1	[-1.0, -0.5773502691896257]	[-1.0, -1.1547005383792515]	1.5275
2	1	-1	-2	[0.0, -1.1547005383792515]	[-0.5, -1.4433756729740643]	1.5275
2	1	2	0	[2.0, 0.5773502691896257]	[1.5, 0.28867513459481287]	1.5275
2	1	-1	0	[-1.0, 0.5773502691896257]	[-1.5, 0.28867513459481287]	1.5275
1	2	-2	0	[-1.5, 0.28867513459481287]	[-1.5, -0.28867513459481287]	1.5275
1	2	1	0	[1.5, 0.28867513459481287]	[1.5, -0.28867513459481287]	1.5275
2	2	1	2	[0.5, 2.02072594216369]	[0.0, 1.732050807568877]	1.7321
2	2	-2	-1	[-1.0, -0.5773502691896257]	[-1.5, -0.8660254037844386]	1.7321
1	1	-2	-1	[-1.5, -0.28867513459481287]	[-1.5, -0.8660254037844386]	1.7321
2	2	2	1	[2.0, 1.1547005383792515]	[1.5, 0.8660254037844386]	1.7321
1	1	-1	-2	[0.0, -1.1547005383792515]	[0.0, -1.7320508075688772]	1.7321
1	1	-1	1	[-1.5, 1.4433756729740643]	[-1.5, 0.8660254037844386]	1.7321
2	2	1	-1	[2.0, -0.5773502691896257]	[1.5, -0.8660254037844386]	1.7321
1	1	1	-1	[1.5, -0.28867513459481287]	[1.5, -0.8660254037844386]	1.7321
1	1	1	2	[0.0, 2.309401076758503]	[0.0, 1.7320508075688772]	1.7321
2	2	-1	1	[-1.0, 1.1547005383792515]	[-1.5, 0.8660254037844386]	1.7321
1	1	2	1	[1.5, 1.4433756729740643]	[1.5, 0.8660254037844386]	1.7321
2	2	-1	-2	[0.5, -1.4433756729740643]	[0.0, -1.7320508075688772]	1.7321
2	2	0	2	[-0.5, 2.02072594216369]	[-1.0, 1.732050807568877]	2.0000
2	2	2	2	[1.5, 2.02072594216369]	[1.0, 1.732050807568877]	2.0000
2	2	-2	-2	[-0.5, -1.4433756729740643]	[-1.0, -1.7320508075688772]	2.0000
2	2	0	-2	[1.5, -1.4433756729740643]	[1.0, -1.7320508075688772]	2.0000
1	1	-2	-2	[-1.0, -1.1547005383792515]	[-1.0, -1.7320508075688772]	2.0000
1	1	0	-2	[1.0, -1.1547005383792515]	[1.0, -1.7320508075688772]	2.0000
1	1	0	2	[-1.0, 2.309401076758503]	[-1.0, 1.7320508075688772]	2.0000
1	1	2	2	[1.0, 2.309401076758503]	[1.0, 1.7320508075688772]	2.0000