PRÁCTICA 1 - MECÁNICA CUÁNTICA I

RUIZ MUÑOZ, JUAN MANUEL [redacted]

8 de diciembre del 2021

1 EJERCICIO 1

Se ha definido todos estos módulos. No todos nos harán falta pero para ahorrarnos tener que estar buscándolos en el caso de que los necesitáramos se han llamado.

```
import math
import random
from time import sleep
from os.path import isdir, isfile, join
from sys import argv, stderr
import glob, os.path
from os import listdir, scandir
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import matplotlib.mlab as mlab
from scipy.integrate import odeint
from scipy import signal
import matplotlib.animation as animation
from matplotlib.pylab import *
from mpl_toolkits.axes_grid1 import host_subplot
import numpy as np
import scipy.linalg as spy
import matplotlib.pyplot as plt
from scipy.sparse import dia_matrix
import random as ran
```

Para el primer ejercicio se define una función potencial tal que,

$$V(x,p) = -5x^2 + \frac{p}{4}x^4, \tag{1}$$

que escribiremos en python como,

```
def modelo1(x, p):

    return (-5)*(x**2) + (p/4)*(x**4)
```

A continuación, se busca el valor del parámetro p para el cual el salto de energía entre los dos primeros niveles de energía E_0, E_1 es igual a 608 MeV. Para ello se ha creado la siguiente función:

```
def varE1E0(varE, epsilon, paso):
   i = 0.0
   vectori = []
   dife = []
   V = ham(xmax, N, potfun, i, mass)
   ener, wave = spy.eigh(V)
   diferencia = abs(ener[1] - ener[0])
   while diferencia < varE - epsilon:
        i += paso
        V = ham(xmax, N, potfun, i, mass)
        ener, wave = spy.eigh(V)
```

```
        diferencia = abs(ener[1] - ener[0])
        dife.append(diferencia)
        vectori.append(i)
        #print(ener[1],ener[0],diferencia,i)
    return i
```

En él se define una variable i que hará la función del parámetro p que se define en (1). Para cada valor de i se define la matriz V que se diagonaliza, dándonos los autovalores que corresponden a los valores de energía del Hamiltoniano. Nos quedamos con los dos valores más pequeños, siendo $E_0 < E_1$.

En cada iteración se comprueba que la diferencia entre ambos niveles sea menor que $varE - epsilon$. Dicho valor será un argumento de entrada y significa el valor de diferencia que queremos llegar, es decir, la función no solo está hecha para una diferencia de 608 MeV. El *paso* de la función viene a ser cómo de grandes queremos que sean los incrementos del parámetro i. A más grande el paso, menos precisión tendremos y viceversa.

Por último, se añade una constante de error, que viene siendo útil si podemos observar el valor de energía correspondiente a la i devuelta. En mi caso le di un valor de $epsilon = 0.1\ MeV$. Si descomentamos el print anterior al return podríamos observar para que valores de energía se detiene el bucle y sería de utilidad, para mayor precisión, ajustar *epsilon*.

En cada entrada al while se actualizan las variables i y V. En *ener* tenemos un vector, en orden ascendente, de los autovalores de energía, siendo $E[0]$ y $E[1]$ los correspondientes al estado fundamental E_0 y al primer estado excitado E_1.

Resumiendo, se le da a la función la diferencia a la que queremos llegar entre dichos estados, junto con un error y un paso. La función en cada iteración irá incrementando la variable i tantas veces como el valor de paso. Para cada i se comprueba que las correspondiente energías de los estados correspondientes sea menor que la diferencia que deseamos, menos un error. Si se cumple sigue el bucle, en caso contrario, nos devuelve el valor de i correspondiente, que es el que buscamos. Añadir que como las energías van aumentando a medida que lo hace el parámetro p (pues solo basta con observar que (1) es polinómica) la diferencia entre los estados también lo hace (esto lo he ido observando a medida que hacia pruebas imprimiendo cada iteración, la diferencia siempre aumentaba).

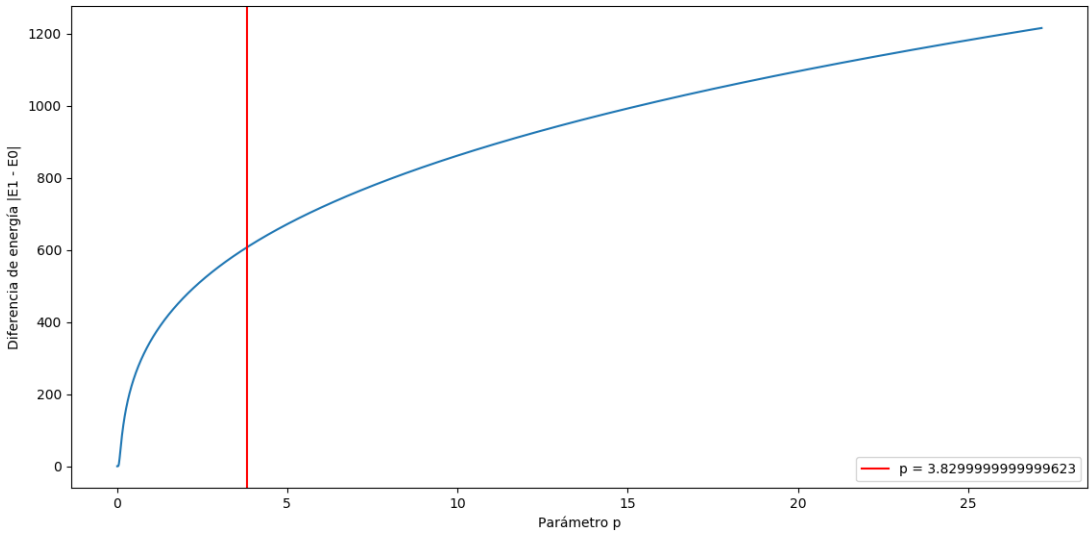


Figura 1: Diferencia entre los estados E_1 y E_0 en función del parámetro p.

Aquí podemos observarlo que acabo de decir. La diferencia de energía va aumentando conforme lo hace p. En principio, este crecimiento no debería de estar acotado pues

$$Lim_{p\to\infty} V(x,p) = \infty,$$

sin embargo, estamos ligados a restricciones físicas.

Las dimensiones de la nanoestructura vienen descritas en el código como,

```
xmax = 100
N = 250
mass = 1  #SUPONEMOS LA MASA DEL ELECTR N COMO LA UNIDAD
potfun = modelo1
```

Esto quiere decir que $x \in [-100, 100]$ y N es únicamente el número de puntos en este intervalo.

Está claro entonces que los valores de x están acotados con lo que si la diferencia de energía se va a infinito debe ser debido al valor del parámetro p. **¿qué significado físico puede tener p?**.

Observamos en la Figura 1 que a medida que este aumenta la diferencia de energía entre ambos estados va aumentando. Esto se debe, evidentemente, a un aumento del potencial que nos provoca que para el electrón cada vez sea más complicado saltar de un estado a otro.

Estamos hablando de una nanoestructura con lo que podríamos pensar que dicho parámetro p puede estar relacionado con el material con el que trabajemos, es decir, en función de los átomos que utilicemos dicha diferencia será diferente pues no sería alocado pensar que el tamaño de la partícula nos afecte a dicha diferencia. La separación entre niveles de energía discretos podrían verse incrementados a medida que el tamaño de la partícula disminuye pues el electrón deberá de realizar un 'salto' mayor. No tendrá el mismo tamaño un átomo de carbono que uno de silicio, por ejemplo.

Dejemos de lado este análisis y continuemos. Para la Figura 1 no he utilizado los vectores que voy cargando en la misma función, pues esos vectores dejan de cargarse una vez se halla el valor de i correspondiente. Con ello, es necesario redifinir la función anterior y darle un valor más amplio al intervalo de iteraciones con el objetivo, no de obtener i, sino de obtener un vector con un mayor rango de diferencias de energías para distintos valores de p.

```
def vartotal(varE, paso):
   i = 0.0
   vectori = []
   dife = []
   V = ham(xmax, N, potfun, i, mass)
   ener, wave = spy.eigh(V)
   diferencia = abs(ener[1] - ener[0])
   while diferencia < 2*varE:
        i += paso
        V = ham(xmax, N, potfun, i, mass)
        ener, wave = spy.eigh(V)
        diferencia = abs(ener[1] - ener[0])
        dife.append(diferencia)
        vectori.append(i)
        #print(ener[1],ener[0],diferencia,i)
   return dife, vectori
```

En el while le he dicho que mientras sea más pequeño que el doble dela diferencia buscada que siga iterando. Con esto pintamos la Figura 1. Con la función $varE1E0$ se pinta el valor i en rojo en la gráfica.

Para pintar las figuras 1 y 2 que se piden se ha escrito en python:

```
A = varE1E0(608, 0.1, 0.01)
B, C = vartotal(608, 0.01)
plt.figure(1)
plt.plot(C, B)
plt.axvline(A, ymin=0, ymax=1, color='red', label = f'p = {A}')
plt.xlabel('Par metro p')
```

```
plt.ylabel('Diferencia de energ a |E1 - E0|')
plt.legend()
plt.show

plt.figure(2)
plotstates(ham(xmax, N, potfun, A, mass), 2, xmax, N, potfun, A)
plt.show()
```

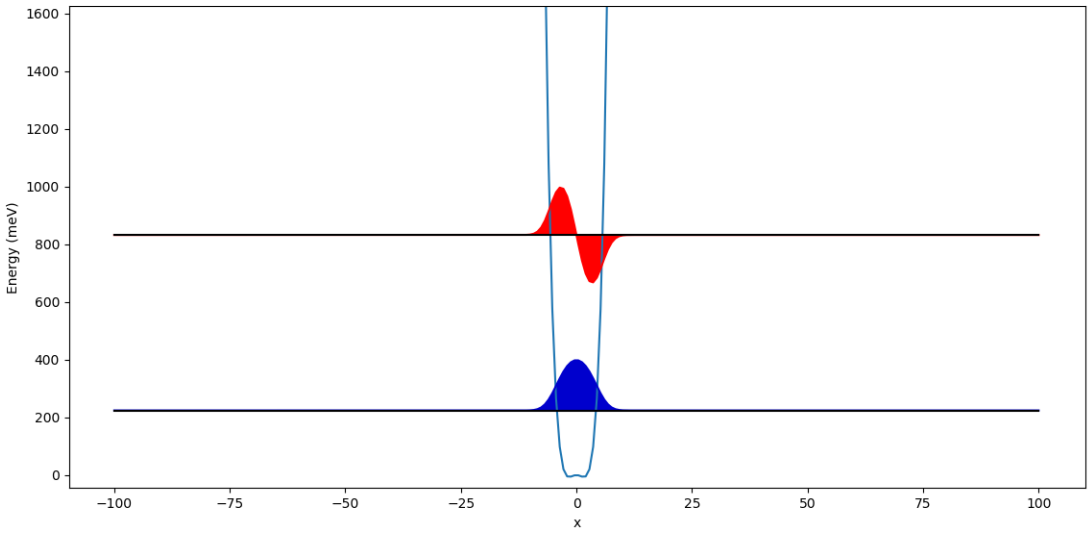


Figura 2: Primeros dos niveles de energía: E_0, E_1.

2 EJERCICIO 2

Se define una doble barrera de potencial de 500 MeV cada una, separadas 30 $\hat{A}$ y con una anchura de 20 $\hat{A}$ cada una. Los parámetros para definir el espacio se dejarán iguales que en el ejercicio anterior.

```
xmax = 100
N = 250
mass = 1 #SUPONEMOS LA MASA DEL ELECTR N COMO LA UNIDAD
potfun = doublebarrier
param =[0,20,30,50,500,500]
```

Para determinar la energía que se nos pide creamos una función parecida a la del anterior ejercicio:

```
def energiacero(p):
    E = 0.0
    T = trans(xmax, N, potfun, param, mass, E)
    while abs(T - p) > 0.001:
        E += 0.1
        T = trans(xmax, N, potfun, param, mass, E)
        #print(E, T, abs(T - p))
    print('Paciencia, queda poco. Estoy buscando el valor de E...')
    return  E
```

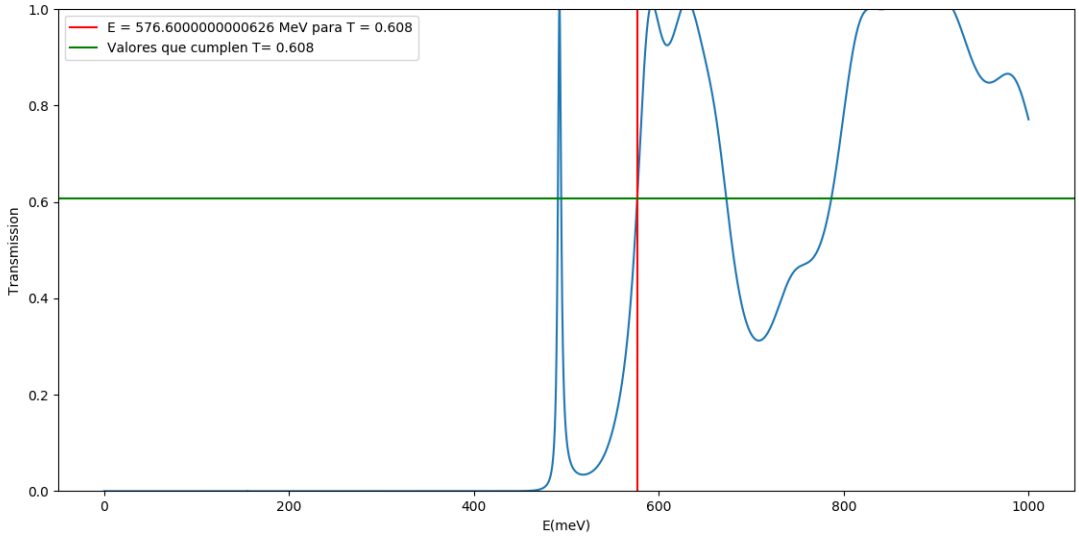
Su funcionamiento es similar al anterior, trabajaremos con un error absoluto. El valor obtenido con esta función es aproximado pues el error es del 0.001.

Introducimos una variable E que irá incrementándose a medida que entramos en el bucle. Si la diferencia absoluta entre la transmitancia teórica con el valor que buscamos p es mayor a 0.001 seguimos iterando. En

caso contrario, cuando la diferencia absoluta entre ambos sea menor a 0.001 nos devuelve el valor de energía correspondiente.

```
#Se evalua la funci n trans para diferentes valores de energ a. Cuando la diferencia
    #absoluta es menor que 0.001 se toma dicho valor de E como v lido
#Tomo como paso de la energ a 0.1 y como error absoluto 0.001.
    #Se obtiene una energ a aprox de 576.6 MeV.
#En las siguientes gr ficas se representa T en funci n de E
    #y se traza una vertical en la energ a que nos da T=0.608 +- 0.001 MeV
L = energiacero(0.608)
plt.figure(3)
plotT(xmax, N, mass, 0.0, 1000, 0.1, potfun, param)
plt.axvline(L, ymin=0, ymax=1, color='red', label=f'E = {L} MeV para T = 0.608')
plt.axhline(y = 0.608, xmin=0,xmax=1, color='green', label=f'Valores que cumplen T= 0.608')
plt.legend()
plt.show()
print('Listo!')
```

Lo que se obtiene es lo siguiente:



Transmitancia en función de la energía.

La linea vertical roja nos representa el valor de energía para el cual se da una transmitancia igual a $T = 0.608$. Este valor, como se observa, no es único pues todos aquellos cortes de la función azul con la linea horizontal verde nos determinan valores de energía con $T = 0.608$.

El valor rojo ha sido hallado con la función que se ha definido arriba, los otros han sido hallados por inspección, es decir, observando la intersección entre la función constante verde y la función azul.