estados_ligados

Dario Mitnik

May 19, 2015

Part I

Ecuación de Schrödinger: Estados Ligados

In [1]:
```
import numpy as np
import scipy as sp
from scipy.integrate import odeint
import matplotlib.pyplot as plt
```

1 Ecuación de Schrödinger:

$$
\begin{aligned}
-\frac{1}{2}\frac{\partial^2\varphi(x)}{\partial x} + V(x)\,\varphi(x) &= E\,\varphi(x) \\
\frac{\partial^2\varphi(x)}{\partial x^2} - 2\left[V(x) - E\right]\,\varphi(x) &= 0 \\
\Rightarrow \frac{\partial^2\varphi(x)}{\partial x^2} &= 2\left[V(x) - E\right]\,\varphi(x)
\end{aligned}
$$

$$
\mathbf{y} \equiv [y_0, y_1] = \left[\varphi(x), \frac{\partial\varphi(x)}{\partial x}\right]
$$

$$
\begin{aligned}
\frac{\partial\mathbf{y}}{\partial x} &= \frac{\partial}{\partial x}\left[\varphi(x), \frac{\partial\varphi(x)}{\partial x}\right] = \left[\frac{\partial\varphi(x)}{\partial x}, \frac{\partial^2\varphi(x)}{\partial x^2}\right] = \\
&= \left[y_1, 2\left(V(x) - E\right)y_0\right] \equiv \left[y_1, g(x)y_0\right]
\end{aligned}
$$

In [2]:
```
# Definición del Potencial

def Vpot(x):
    return ( (x-5)**2 )/ 2.0
```

2 Solución de la Ecuación

$$\frac{\partial \mathbf{y}}{\partial x} \quad = \quad [y_1, 2\,(V(x) - E)\,y_0]$$

In [3]:
```
def g(y, x, E):
    return [y[1],2*(Vpot(x)-E)*y[0]]
```

In [4]:
```
# Valores iniciales de phi(x) y phi'(x)
initialY = 0.0,0.0005

# Valor tentativo de E
E = 0.5

x = np.linspace(0,10,1000)

# Solucion ecuación diferencial
sol = odeint(g,initialY,x,(E,))

# Ploteo de solución
plt.plot(x, sol[:,0], color='b')
plt.axis([0, 10, 0,15])
plt.plot(x,Vpot(x),color='k')
plt.show()
```

3 Ejercicio: Encontrar el 1er Estado Excitado

In [5]:
```
#Solución:

init = 0.0,0.005
E = 0.5
x = np.linspace(0,10,1000)
sol2 = odeint(g,init,x,(E,))

plt.plot(x, sol2[:,0], color='b')
plt.axis([0, 10, -20,20])
plt.plot(x,Vpot(x),color='k')
plt.show()
```

4 Otros Potenciales

In [6]:
```
# Definición del Potencial Wood-Saxon

def Vpot(x):
    U = 7
    a = 2
    pot = -U / ( np.exp((x-a)**2) + 1 )
    return pot
```

In [7]:
```
init = 0.0,0.5
E = -2.61
x = np.linspace(0,6,1000)
sol = odeint(g,init,x,(E,))

plt.plot(x, sol[:,0], color='b')
plt.plot(x,Vpot(x),color='k')
plt.axis([0, 6, -5,5])
plt.show()
```

4.1 Solución Interactiva

In [8]:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider, Button, RadioButtons
import scipy as sp
from scipy.integrate import odeint
```

In [9]:

```
fig, ax = plt.subplots()
plt.subplots_adjust(left=0.25, bottom=0.25)
xmin = 0
xmax = 5
npts = 1000

x = np.linspace(xmin, xmax,npts)

# Valores iniciales de phi(x) y phi'(x)
initialY = 0.0,0.5

# Valor tentativo de E
Ener = -3.5

def function(Ener,x):
# Solucion ecuación diferencial
    y = odeint(g,initialY,x,(Ener,))
    return y

s = function(Ener,x)


l, = plt.plot(x,s[:,0], lw=2, color='red')
plt.plot(x,Vpot(x),color='k')
plt.axis([xmin,xmax, -4, 4]);
```

In [10]:

```
axcolor = 'lightgoldenrodyellow'
axenergy = plt.axes([0.25, 0.1, 0.65, 0.03], axisbg=axcolor)

senergy = Slider(axenergy, 'Energy', -3.5,0.5,valinit=Ener)
```

In [11]:

```
def update(val):
    Ener = senergy.val
    y = function(Ener,x)
    l.set_ydata(y[:,0])
    fig.canvas.draw_idle()

senergy.on_changed(update);
```

In [12]:

```
plt.show()
```

5 Ejercicios:

- Encontrar Estados Excitados (si existen)
- Repetir el ejercicio centrando el potencial en $a = 2$
- Encontrar los 4 primeros estados ligados de un pozo finito

In [ ]:

```

```