NAME: ANURAG DAS

ROLL NO.: 2020PHY1116

QUANTUM MECHANICS

(LAB)

SEMESTER - V

$\boxed{\text{Ans 1}}$ (a) The time-independent S.E.:

$$i\hbar\frac{\partial \Psi}{\partial t} = -\frac{\hbar^2}{2m}\nabla^2\Psi + V\Psi$$
$$= H\Psi$$

In spherical coordinates;

$$\nabla^2 = \frac{1}{\rho^2}\frac{\partial}{\partial \rho}\left(\rho^2\frac{\partial}{\partial \rho}\right) + \frac{1}{\rho^2 \sin\theta}\frac{\partial}{\partial \theta}\left(\sin\theta\frac{\partial}{\partial \theta}\right) + \frac{1}{\rho^2\sin^2\theta}\frac{\partial^2}{\partial \phi^2}$$

So our Time Independent SE can be written as:

$$\frac{\hbar^2}{2m}\left[\frac{1}{\rho^2}\frac{\partial}{\partial \rho}\left(\rho^2\frac{\partial \Psi}{\partial \rho}\right) + \frac{1}{\rho^2\sin\theta}\frac{\partial}{\partial \theta}\left(\sin\theta\frac{\partial \Psi}{\partial \theta}\right) + \frac{1}{\rho^2\sin^2\theta}\frac{\partial^2\Psi}{\partial \phi^2}\right]$$
$$= [E - V(\rho)]\Psi$$

ut $\quad \Psi(\rho,\theta,\phi) = R(\rho)Y(\theta,\phi)$

$$-\frac{\hbar^2}{2m}\left[\frac{Y}{\rho^2}\frac{d}{d\rho}\left(\rho^2\frac{dR}{d\rho}\right)+\frac{R}{\rho^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial Y}{\partial\theta}\right)+\frac{R}{\rho^2\sin^2\theta}\frac{\partial^2 Y}{\partial\theta^2}\right]$$

$$= RY\left[E-V(\rho)\right]\psi$$

$$\frac{1}{R}\frac{d}{d\rho}\left(\rho^2\frac{dR}{d\rho}\right)-\frac{2m\rho^2}{\hbar^2}\left[V(\rho)-E\right] = -\frac{1}{Y}LY$$

where $L = \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right)+\frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\phi^2}$.

Seperating the radial & angular part of equation:

$$\frac{d}{d\rho}\left(\rho^2\frac{dR}{d\rho}\right)-\frac{2m\rho^2}{\hbar^2}\left[V(\rho)-E\right]R = l(l+1)R$$

$$\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial Y}{\partial\theta}\right)+\frac{1}{\sin^2\theta}\frac{\partial^2 Y}{\partial\phi^2} = -l(l+1)Y$$

Ans 1 (b) For radial part:

$$\left[-\frac{\hbar^2}{2m}\frac{d^2}{dr^2} + \frac{\hbar^2}{2mr^2} l(l+1) - \frac{b\, e^{-r/a} e^2}{r}\right] X_{nl}(r) = E_{nl} X_{nl}(r)$$

$\cancel{V_{eff}(r) = \frac{e^2}{r}}$

Let $\xi a_0 = r$

$$\therefore \left[-\frac{\hbar^2}{2m}\frac{1}{a^2}\frac{d^2}{d\xi^2} + \frac{\hbar^2 l(l+1)}{2m a_0^2 \xi^2} - \frac{e^{-\xi a_0/a} e^2}{a_0 \xi} - E_{nl}\right] X_{nl}(a_0\xi) = 0$$

Let $u_{nl}(\xi) = X_{nl}(a_0\xi)$

$$\left[\frac{d^2}{d\xi^2} - \frac{l(l+1)}{\xi^2} + \frac{e^{-\xi a_0/a}\, 2\mu e^2 a_0}{\hbar^2 \xi} + \frac{2\mu a_0^2 E_{nl}}{\hbar^2}\right] u_{nl}(\xi) = 0$$

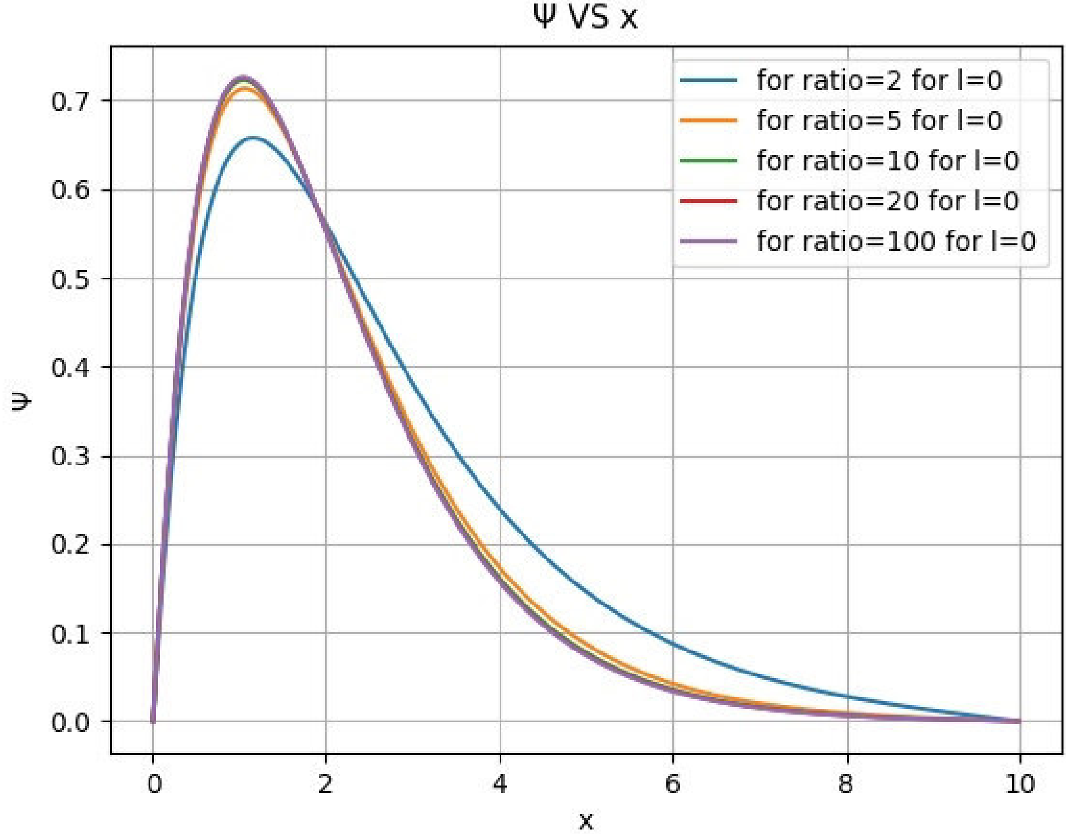
$$\left[\frac{d^2}{d\xi^2} - \frac{l(l+1)}{\xi^2} + \frac{2}{\xi}e^{-\frac{a_0\xi}{a}} - d_{nl}^2\right] u_{nl}(\xi) = 0$$

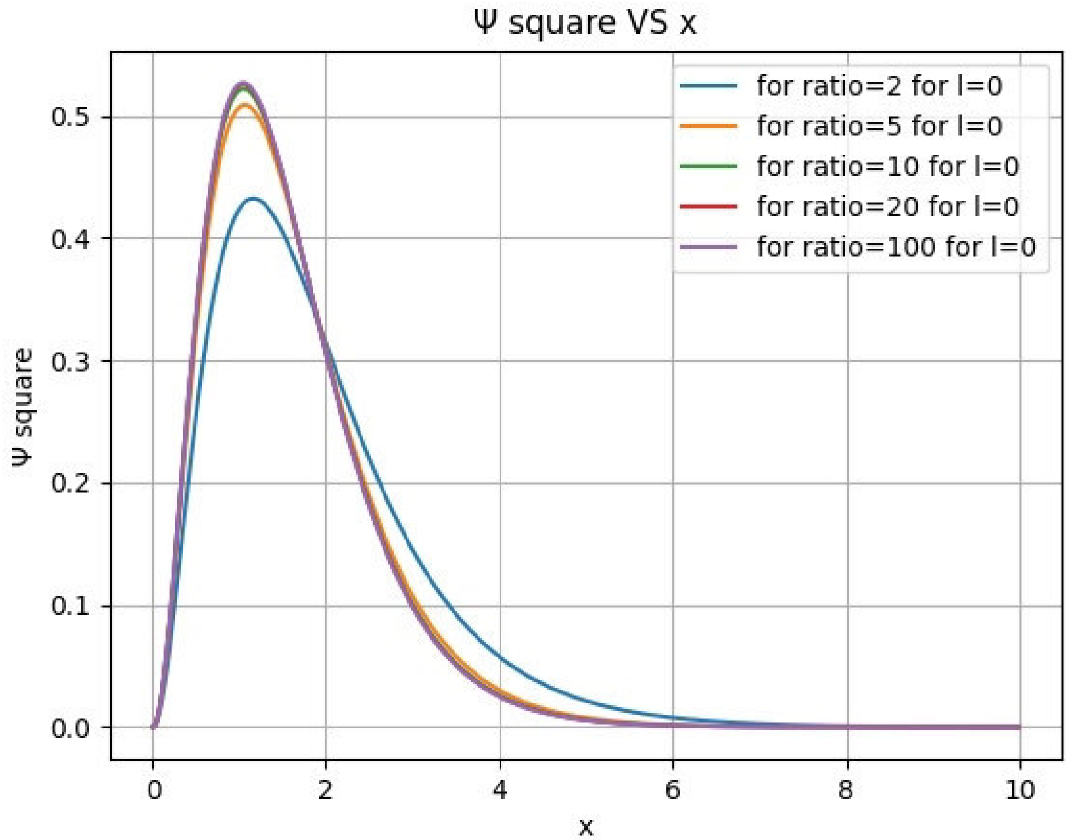
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import eigh
import scipy.integrate as integrate


def matrix(a, b, n, l,ratio):
    x = np.arange(a, b, n)
    # print(x)
    h = x[1] - x[0]
    u = np.zeros(shape=(len(x), len(x)))
    V = np.zeros(shape=(len(x), len(x)))
    for i in range(1, len(x) - 1):
        for j in range(1, len(x)):
            if i == j:
                u[i][j] = (2 / h ** 2)
                V[i][j] = (l * (l + 1) / 2*((x[i] ** 2))) - (
2 / x[i])*np.exp(-x[i]/ratio)
            elif i == j + 1:
                u[i][j] = -1 / h ** 2
            elif i == j - 1:
                u[i][j] = -1 / h ** 2
    return u + V, x

def plot(i, l, power,ratio):
    H, x = matrix(0.01, 10, 0.01, l,ratio)
    u = eigh(H)[1][:, i]
    # print(u)
    # NORMALIZATION
    c = integrate.simps(u ** 2, x)
    N = u / np.sqrt(c)
    if power==2:
        plt.title("Ψ square VS x")
        plt.ylabel("Ψ square")
    else:
        plt.title("Ψ VS x")
```

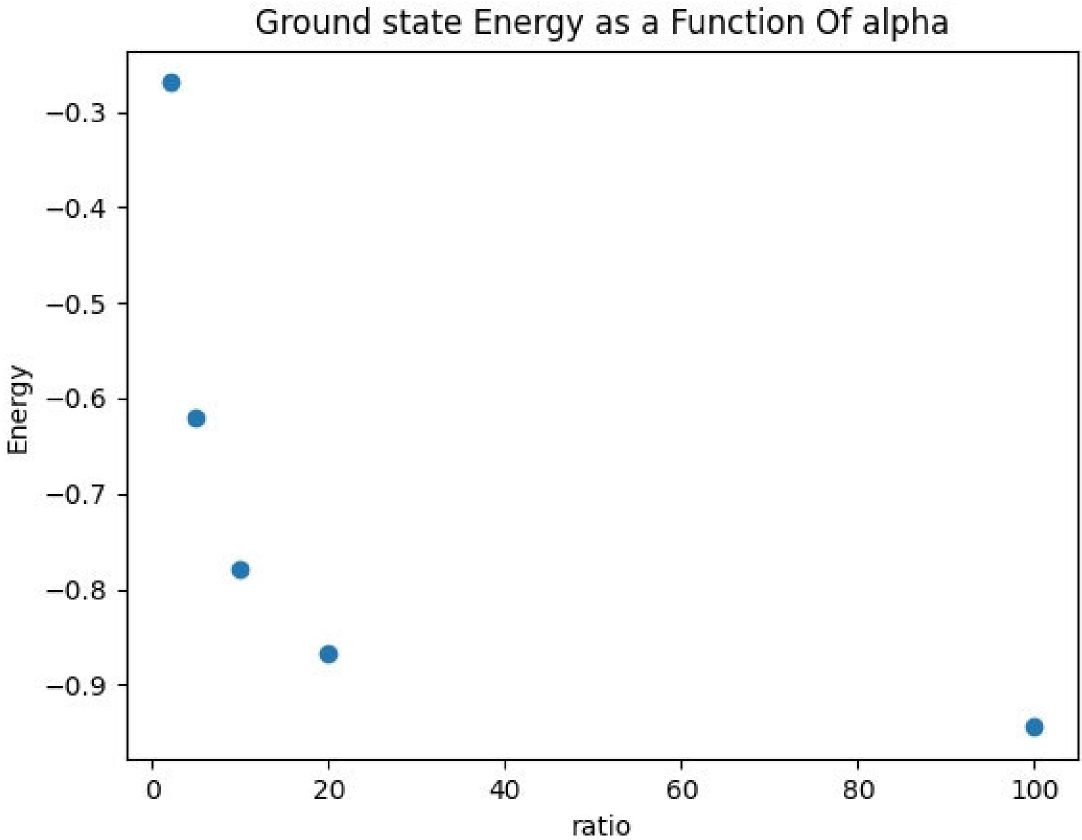
```
        plt.ylabel("Ψ")
    plt.xlabel('x')
    plt.plot(x, N ** power, label='for ratio='+str(
ratio)+' for l='+str(l))
    plt.legend()

def Veff(x, l, ratio):
    Vef = (l * (l + 1) / (x ** 2)) - (2 / x)*np.exp(-x/
ratio)
    V = -2 / (x)
    return Vef, V

def eigen(a, b, h, l,ratio, i):
    H, x = matrix(a, b, h, l,ratio)
    u = eigh(H)[0][i]
    v = eigh(H)[1][:, i]
    return u

# A,B

for j in range(0,2):
    print("For n=", j + 1)
    for i in [2,5,10,20,100]:
        energy=eigen(0.01, 5*(j+1), 0.01, 0, i, j)
        if energy<0:
            print("bound state energy eigen value
exists for alpha=",i,": ",energy)

#C,D

E=[]
ratio=[2,5,10,20,100]
for i in ratio:
    plot(0,0,1,i)
    energy = eigen(0.01, 10, 0.01, 0, i, 0)
    E.append(energy)
```

```
plt.grid()
#plot(0,0,2,0.00001)
plt.savefig('plot1.jpg')
plt.show()

for i in [2,5,10,20,100]:
    plot(0,0,2,i)
plt.grid()
#plot(0,0,2,0.00001)
plt.savefig('plot2.jpg')
plt.show()

#E

plt.scatter(ratio,E)
plt.xlabel('ratio')
plt.ylabel('Energy')
plt.title('Ground state Energy as a Function Of
alpha')
plt.savefig('plot3.jpg')
plt.grid()
plt.show()
```

```
C:\Users\anura\AppData\Local\Programs\Python\
Python38\python.exe E:/SEM-5/A12/
1116_Anurag_qmLabA12.py
For n= 1
bound state energy eigen value exists for alpha= 2
 :  -0.24539631379986598
bound state energy eigen value exists for alpha= 5
 :  -0.6087348863725712
bound state energy eigen value exists for alpha= 10
 :  -0.7696501575954996
bound state energy eigen value exists for alpha= 20
 :  -0.8592611636168581
bound state energy eigen value exists for alpha=
100 :  -0.9358098729469347
For n= 2
bound state energy eigen value exists for alpha= 10
 :  -0.06169922842762132
bound state energy eigen value exists for alpha= 20
 :  -0.13063482377068447
bound state energy eigen value exists for alpha=
100 :  -0.19955575721751584

Process finished with exit code 0
```


Ψ VS x
for ratio=2 for l=0
for ratio=5 for l=0
for ratio=10 for l=0
for ratio=20 for l=0
for ratio=100 for l=0
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Ψ
0
2
4
6
8
10
x


Ψ square VS x
for ratio=2 for l=0
for ratio=5 for l=0
for ratio=10 for l=0
for ratio=20 for l=0
for ratio=100 for l=0
Ψ square
x
0.0
0.1
0.2
0.3
0.4
0.5
0
2
4
6
8
10


Ground state Energy as a Function Of alpha
Energy
ratio
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
−0.9
0
20
40
60
80
100