Roll no :- Gaurav
2020PHY1122

Ans 1)

a) The time-independent Schrodinger eqn :-

$$-\frac{\hbar^2}{2m}\nabla^2\psi + V(r)\psi = E\psi$$

$$H\psi = E\psi$$

In spherical coordinates

~~$\nabla^2 = \frac{1}{r^2}\frac{\partial}{}$~~

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

So our time-independent Schrodinger eqn can be written as :-

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

b) let $\psi(r,\theta,\phi) = R(r)Y(\theta,\phi)$

$$-\frac{\hbar^2}{2m}\left[\frac{Y}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{dR}{dr}\right) + \frac{R}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial Y}{\partial\theta}\right) + \frac{R}{r^2\sin^2\theta}\frac{\partial^2 Y}{\partial\phi^2}\right] = RY(E-V)$$

Soumyaa

$$\cancel{\frac{-\hbar^2}{2m}} \quad \cancel{\frac{-\hbar^2}{2m_r^2}\left[\frac{1}{R}\frac{\partial}{\partial r}\left(r^2\frac{dR}{dr}\right)+\frac{1}{\varphi\sin\theta}\frac{\partial}{\partial\theta}\left(\right.\right.}$$

$$\frac{1}{R}\frac{\partial}{\partial r}\left(r^2\frac{dR}{dr}\right)+\frac{1}{\varphi\sin^2\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\varphi}{\partial\theta}\right)+\frac{1}{\varphi\sin\theta}\frac{\partial^2\varphi}{\partial\phi^2}$$

$$= -\frac{2mr^2}{\hbar^2}(E-V)$$

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

so;

$$\frac{1}{R}\frac{\partial}{\partial r}\left(r^2\frac{dR}{dr}\right)-\frac{2mr^2}{\hbar^2}\left[V(r)-E\right] = l(l+1)$$

or

$$\frac{\partial}{\partial r}\left(r^2\frac{dR}{dr}\right)-\frac{2mr^2}{\hbar^2}\left[V(r)-E\right]R = l(l+1)R$$

or if we have taken;

$\psi_{nlm}(r,\theta,\phi) = R_{nl}(r)\,Y_{lm}(\theta,\phi)$

then;

eqn would be;

$$\frac{\partial}{\partial r}\left(r^2\frac{dR_{nl}(r)}{dr}\right)-\frac{2mr^2}{\hbar^2}\left[V(r)-E\right]R_{nl}(r) = l(l+1)R_{nl}(r)$$

c) we originally have S.E in polar coordinates, as for radial part:

$$\left[\frac{-\hbar^2}{2m}\frac{d^2}{dr^2} + \frac{\hbar^2}{2mr^2} l(l+1) - \frac{Ke^2}{r}\right] \cancel{X}_{nl}(r) = E_{nl} \cancel{X}_{nl}(r)$$

$$\underbrace{V_{eff} = \frac{-Ke^2}{r} + \frac{\hbar^2}{2mr^2} l(l+1)}_{\text{effective potential}}$$

~~let~~ $R_{nl}(r) = \frac{K_{nl}(r)}{r}$

let $\rho = \frac{r}{a_0}$ $\qquad a_0 = \text{bohr's radius} = \frac{\hbar^2}{mKe^2}$

$$\left[\frac{-\hbar^2}{2m} \times \frac{1}{a_0^2}\frac{d^2}{d\rho^2} + \frac{\hbar^2 l(l+1)}{2m a_0^2 \rho^2} - \frac{Ke^2}{a_0\rho} - E_{nl}\right] X_{nl}(a_0\rho) = 0$$

let $\vartheta_{nl}(\rho) = X_{nl}(a_0\rho)$

$$\frac{-\hbar^2}{2ma_0^2}\left[\frac{d^2}{d\rho^2} - \frac{l(l+1)}{\rho^2} + \frac{2ma_0Ke^2}{\hbar^2\rho} + \frac{2ma_0^2 E_{nl}}{\hbar^2}\right]\vartheta_{nl}(\rho) = 0$$

$$\left[\frac{d^2}{d\rho^2} - \frac{l(l+1)}{\rho^2} + \frac{2}{\rho} + e_{nl}\right]\vartheta_{nl}(\rho) = 0$$

here, $\boxed{e_{nl} = \frac{2ma_0^2 E_{nl}}{\hbar^2}}$

$\frac{1}{4\pi\varepsilon_0}$

$$\Rightarrow \vartheta''_{nl}(\rho) - V_{eff}\,\vartheta_{nl}(\rho) = -e_{nl}\,\vartheta_{nl}(\rho)$$

$$V_{eff} = \frac{l(l+1)}{\rho^2} - \frac{2}{\rho}, \quad V(r) = \frac{-2}{\rho}$$

Soumyaa

d) $V_{eff}(r) = \frac{\hbar^2}{2m} \frac{l(l+1)}{r^2} - \frac{Ke^2}{r}$

first term :- It is the Centripetal potential related to $\frac{1}{r^2}$

Second term :- it is the Coulombic potential, related to $\frac{-1}{r}$

when $l \neq 0$ (i.e not S orbital), Centripetal potential is dominant, i.e.
electrons of P, d, f orbitals are far from nucleus then electrons in S-orbitals ($l=0$)

Soumyaa

```python
#name : Gaurav
#rollno : 2020PHY1122

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.linalg import eigh
from scipy.special import assoc_laguerre as al


def V(r):
    return -2/r

def V_eff(l, r):
    return l*(l+1)/(r**2) + V(r)

def fin_diff(X,l):
    #h = (b-a)/(n-1)  #n is number of grid points
    n = len(X)
    K,V = np.zeros((n,n)),np.zeros((n,n))
    h = X[1]-X[0]
    #X = np.linspace(a,b,n)

    v = V_eff(l, X)

    K[0,0] = -2;K[0,1] = 1
    K[n-1,n-1] = -2;K[n-1,n-2] = 1

    for i in range(n):
        V[i,i] = v[i]
    for i in range(1,n-1):
```

```python
    for i in range(1,n-1):
        K[i,i]=-2
        K[i,i-1]=1
        K[i,i+1]=1

    H = (-1*K)/(h**2) + V

    U = eigh(H)[1]
    e = eigh(H)[0]

    return [e,U]

def MySimp(x,y):  #x here is the array of independent variable  and y for dependent variable
    # calculating step size
    h = abs((x[-1] - x[0]) / len(x))

    simpint = y[0] + y[-1]

    for i in range(1,len(x)):

        if i%2 == 0:
            simpint = simpint + 2 * y[i]
        else:
            simpint = simpint + 4 * y[i]

    # multiply h/2 with the obtained integration to get Simpson integration
    simpint =simpint * h/3

    return simpint

def normalize(wavefx,wavefy,int_method = MySimp):  #this function returns list including normalisation constant and
                                                   #normalised eigen function
    I = int_method(wavefx,wavefy**2)
    A = (I)**(-1/2)

    return [A,A*wavefy]
```

```python

def plots(x,y1,y2,title,color = None): #num defines if there would be only one plot or more


    for i in range(len(y1)):
        plt.plot(x, y2[i], label='analytical l = '+ str(i),c = color[i][0])
        plt.scatter(x, y1[i],s=5, label='computed l = '+ str(i),c=color[i][1])
    plt.grid()
    plt.xlabel('x')
    plt.ylabel('u**2')
    plt.xlim(0,10)
    plt.title(title)
    plt.legend()
    plt.show()

def analytical_sol(x,n,l):
    anal = np.exp(-x/n)*(2*x/n)**l*al(2*x/n,n-l-1 ,2*l+1)
    norm_anal = normalize(x, anal)[1]
    return norm_anal



#PROGRAMMING

#part a_i

r = np.linspace(10**(-14), 150, 1000)

for i in range(1, 4):  # l= 1,2,3
    plt.plot(r, V_eff(i, r),label = 'V_eff for l='+str(i))

plt.plot(r, V(r),label = "V(r)",c = 'y')
plt.title("plot of potential vs x")
plt.xlabel("r")
plt.ylabel("v(r)")
plt.grid()
plt.legend()
plt.show()

```

```python
l0,l1,l2 = 0,1,2

sol = fin_diff(r,l0)
print("for l  =0")
print("THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 0 ARE : ")

anal_e = []

for i in range(1,11):
    anal_e.append(-1*(i)**-2)

print(pd.DataFrame({'COMPUTED e':sol[0][1:11],'ANALYTICAL e':anal_e}))


for i in range(1,5):

    u = sol[1][:, i]

    norm_u = normalize(r, u)[1]  #normalised wave using normalise function

    anal = analytical_sol(r, i, l0)
    sign = [1,-1,1,-1]
    plt.scatter(r,norm_u,s=5,label = 'computed',c = 'r')
    plt.plot(r,sign[i-1]*anal,label = 'analytical')
    plt.xlabel('X')
    plt.xlim(0,50)
    plt.ylabel('U')
    plt.title("PLOT FOR U VS X FOR N = "+str(i))
    plt.grid();plt.legend()
    plt.show()

print('')
print("for l = 1")
sol_1 = fin_diff(r[1:],1)

print("THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 1 ARE : ")
```

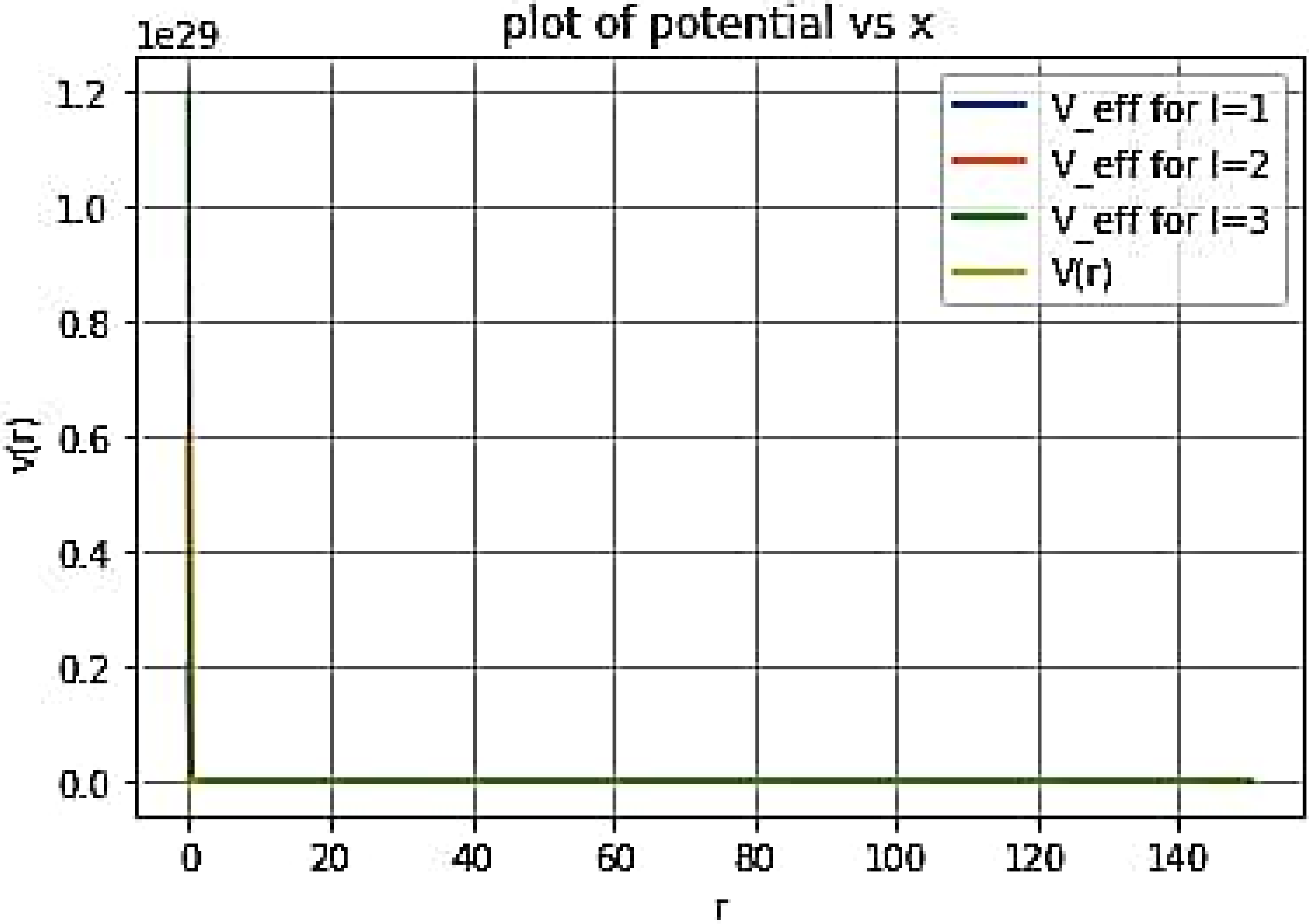
```
anal_e = []

for i in range(2,12):
    anal_e.append(-1*(i)**-2)
print(pd.DataFrame({'COMPUTED e':sol_1[0][:10],'ANALYTICAL e':anal_e}))
print('')
print("for l = 2")
sol_2 = fin_diff(r[1:],2)

print("THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 2 ARE : ")

anal_e = []

for i in range(3,13):
    anal_e.append(-1*(i)**-2)
print(pd.DataFrame({'COMPUTED e':sol_2[0][0:10],'ANALYTICAL e':anal_e}))


#("plot for probability density")
#part_i  for n=l=0
u = sol[1][:,1] #n+1 th eigen vectors
u_norm = normalize(r, u)[1]
anal = analytical_sol(r,1,l0)

plots(r, [u_norm**2], [anal**2], title = "probability density plot for n=0",color = [['b','g']])


#part_ii  for n=1 l = 0,1
u_1 = sol[1][:,2][1:]
u_2 = sol_1[1][:,2]
u_norm_1 = normalize(r[1:], u_1)[1]
u_norm_2 = normalize(r[1:], u_2)[1]
anal_1 = analytical_sol(r[1:],2,l0)
anal_2 = analytical_sol(r[1:],2,l1)

plots(r[1:], [u_norm_1**2,u_norm_2**2], [anal_1**2,anal_2**2], title = "probability density plot for n=1",color = [['b','g']
```

```python

for i in range(3,13):
    anal_e.append(-1*(i)**-2)
print(pd.DataFrame({'COMPUTED e':sol_2[0][0:10],'ANALYTICAL e':anal_e}))


#("plot for probability density")
#part_i  for n=l=0
u = sol[1][:,1] #n+1 th eigen vectors
u_norm = normalize(r, u)[1]
anal = analytical_sol(r,1,l0)

plots(r, [u_norm**2], [anal**2], title = "probability density plot for n=0",color = [['b','g']])


#part_ii  for n=1 l = 0,1
u_1 = sol[1][:,2][1:]
u_2 = sol_1[1][:,2]
u_norm_1 = normalize(r[1:], u_1)[1]
u_norm_2 = normalize(r[1:], u_2)[1]
anal_1 = analytical_sol(r[1:],2,l0)
anal_2 = analytical_sol(r[1:],2,l1)

plots(r[1:], [u_norm_1**2,u_norm_2**2], [anal_1**2,anal_2**2], title = "probability density plot for n=1",color = [['b','g'],['r','y']])


#part_iii  for n=2 l = 0,1,2
u_1 = sol[1][:,3][1:]
u_2 = sol_1[1][:,3]
u_3 = sol_2[1][:,3]
u_norm_1 = normalize(r[1:], u_1)[1]
u_norm_2 = normalize(r[1:], u_2)[1]
u_norm_3 = normalize(r[1:],u_3)[1]
anal_1 = analytical_sol(r[1:],3,l0)
anal_2 = analytical_sol(r[1:],3,l1)
anal_3 = analytical_sol(r[1:],3,l2)
plots(r[1:], [u_norm_1**2,u_norm_2**2,u_norm_3**2], [anal_1**2,anal_2**2,anal_3**2], title = "probability density plot for n=2",color = [

```

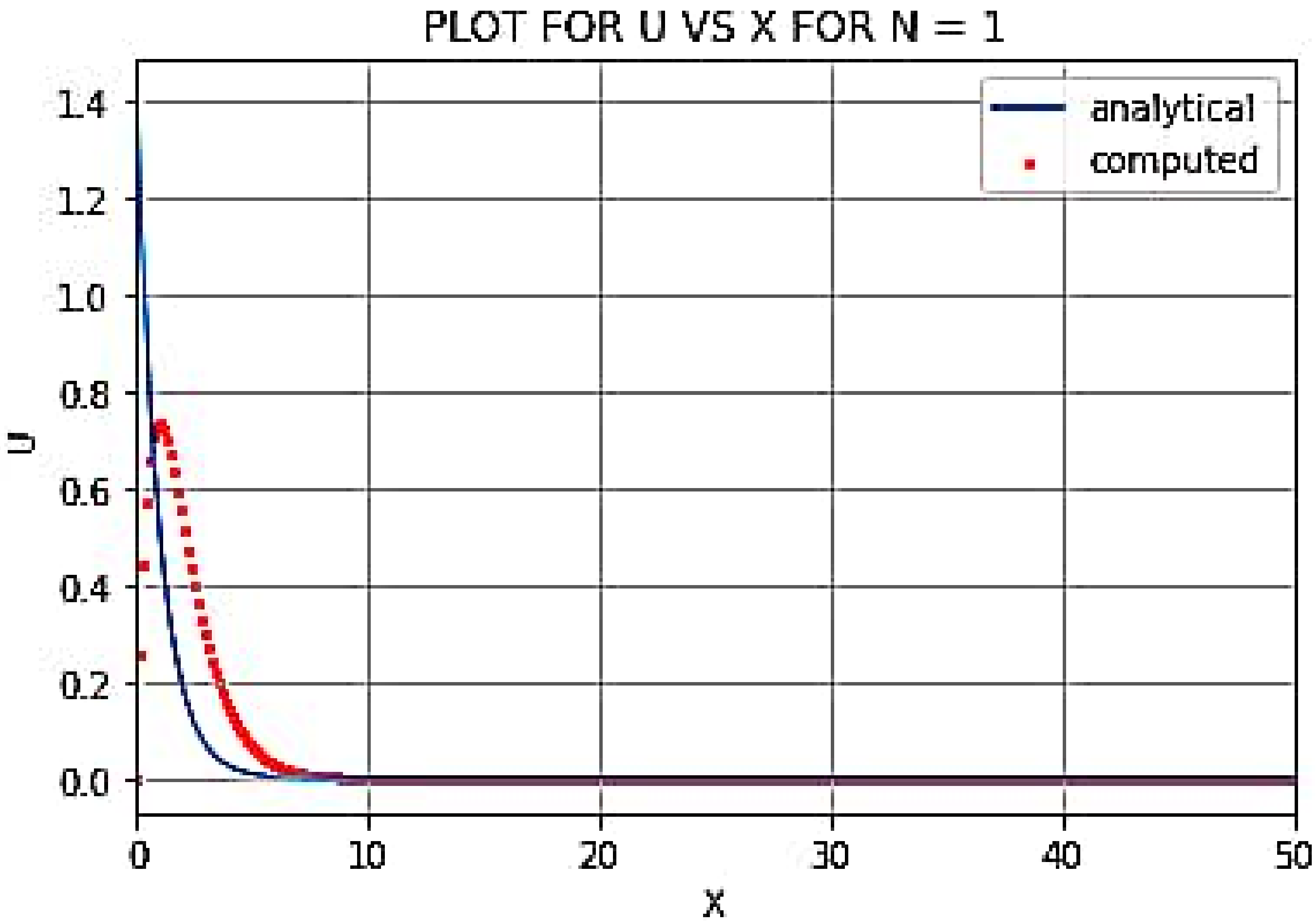
```
for l  =0
THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 0 ARE :
   COMPUTED e  ANALYTICAL e
0   -0.994426     -1.000000
1   -0.249649     -0.250000
2   -0.111042     -0.111111
3   -0.062478     -0.062500
4   -0.039991     -0.040000
5   -0.027773     -0.027778
6   -0.020404     -0.020408
7   -0.015488     -0.015625
8   -0.010963     -0.012346
9   -0.005321     -0.010000
```

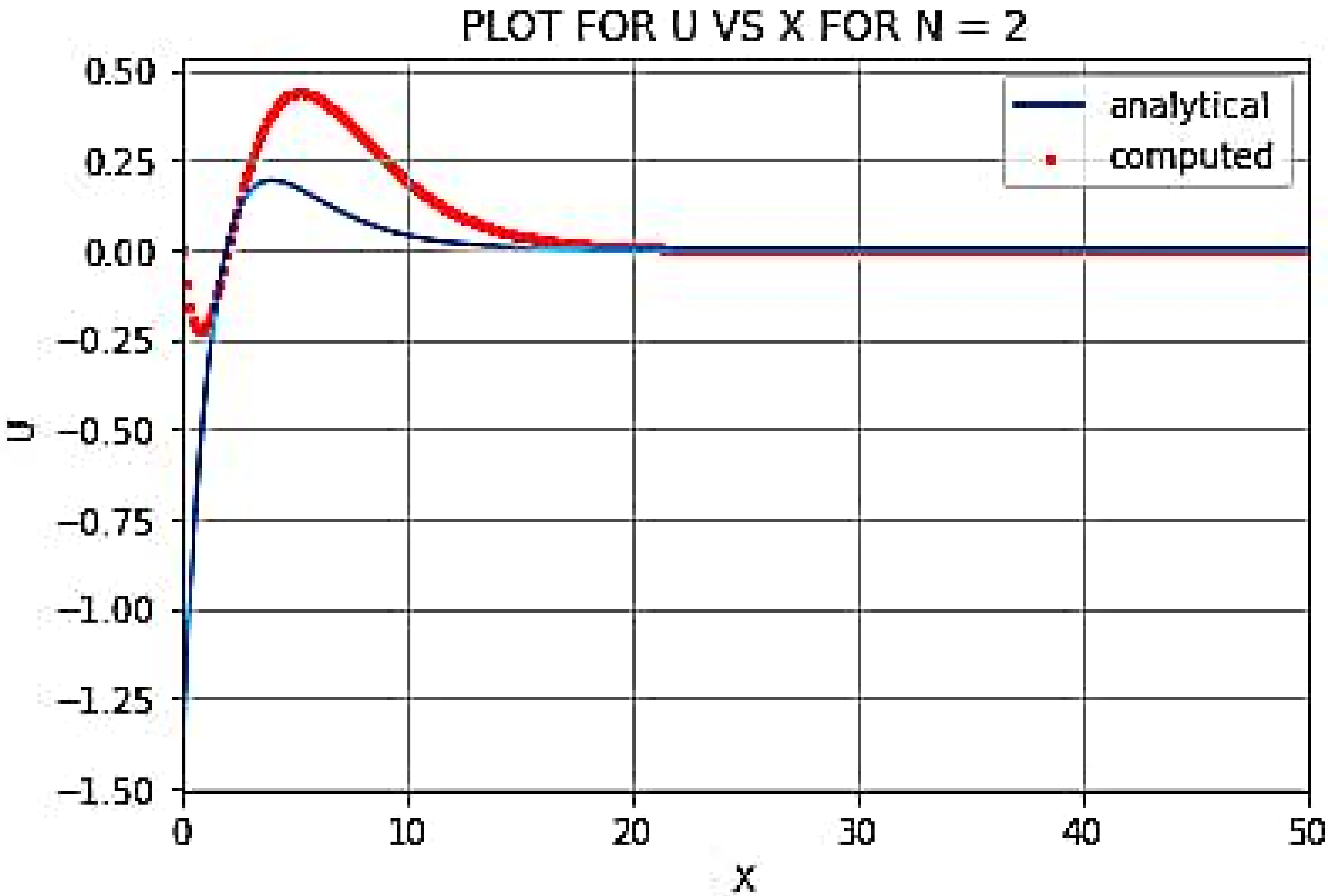
```
for l = 1
THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 1 ARE :
   COMPUTED e  ANALYTICAL e
0   -0.250118     -0.250000
1   -0.111165     -0.111111
2   -0.062528     -0.062500
3   -0.040016     -0.040000
4   -0.027788     -0.027778
5   -0.020413     -0.020408
6   -0.015508     -0.015625
7   -0.011049     -0.012346
8   -0.005507     -0.010000
9    0.001283     -0.008264
```

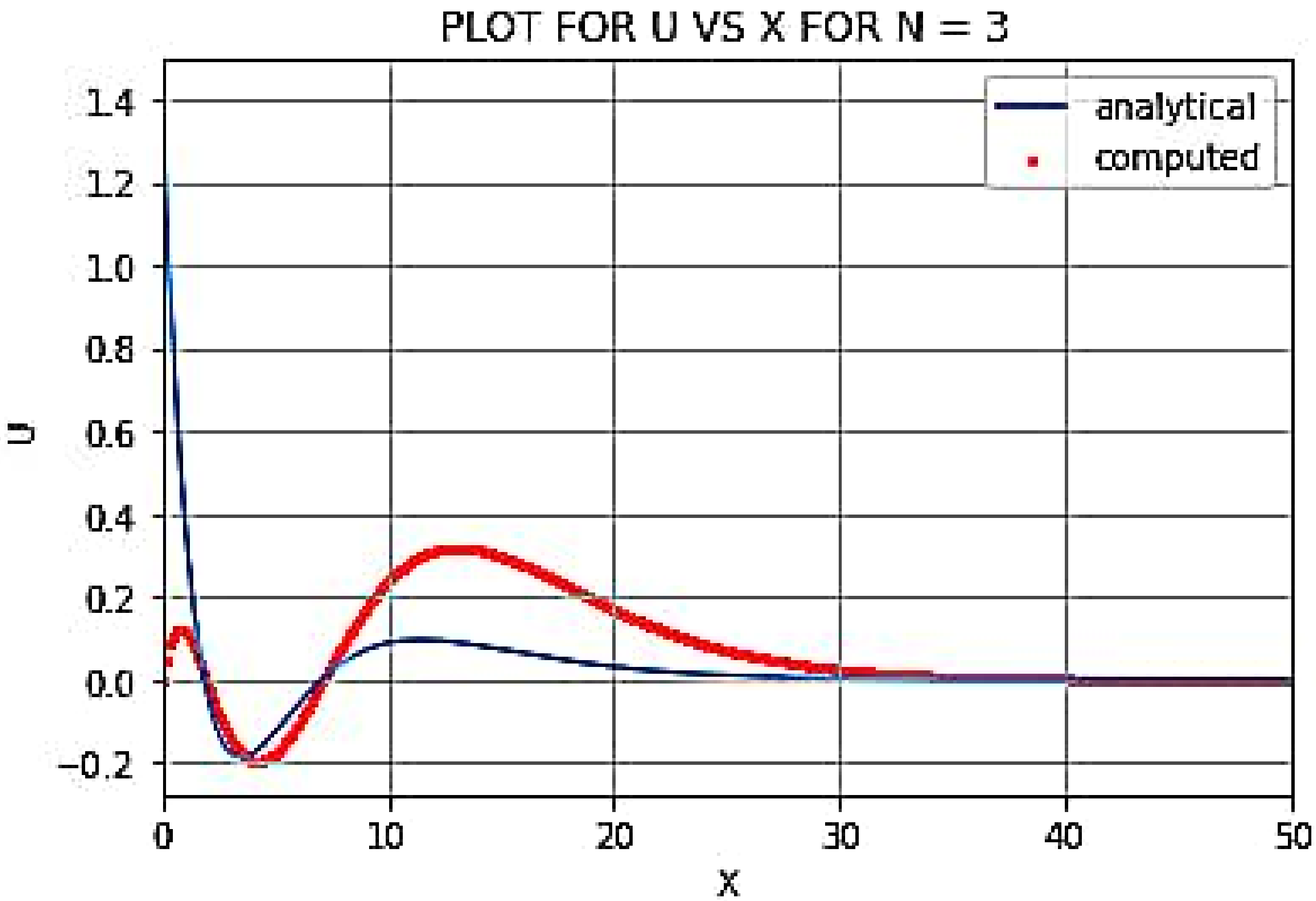
```
for l = 2
THE FIRST 10 EIGEN VALUES COMPUTED USING FINITE DIFFERENCE METHOD FOR L = 2 ARE :
   COMPUTED e  ANALYTICAL e
0   -0.111116     -0.111111
1   -0.062504     -0.062500
2   -0.040003     -0.040000
3   -0.027780     -0.027778
4   -0.020409     -0.020408
5   -0.015528     -0.015625
6   -0.011193     -0.012346
7   -0.005842     -0.010000
8    0.000735     -0.008264
9    0.008438     -0.006944
```

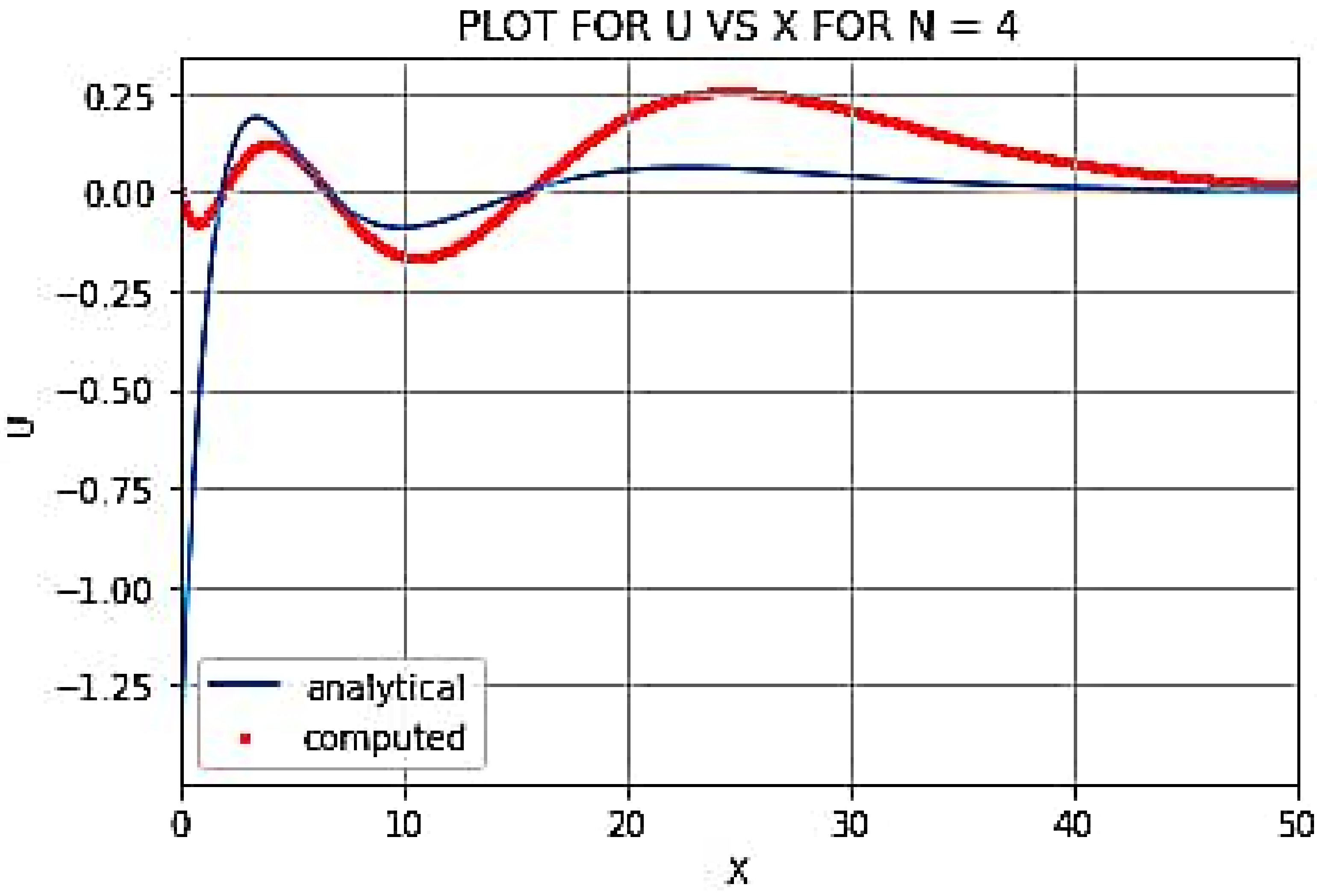


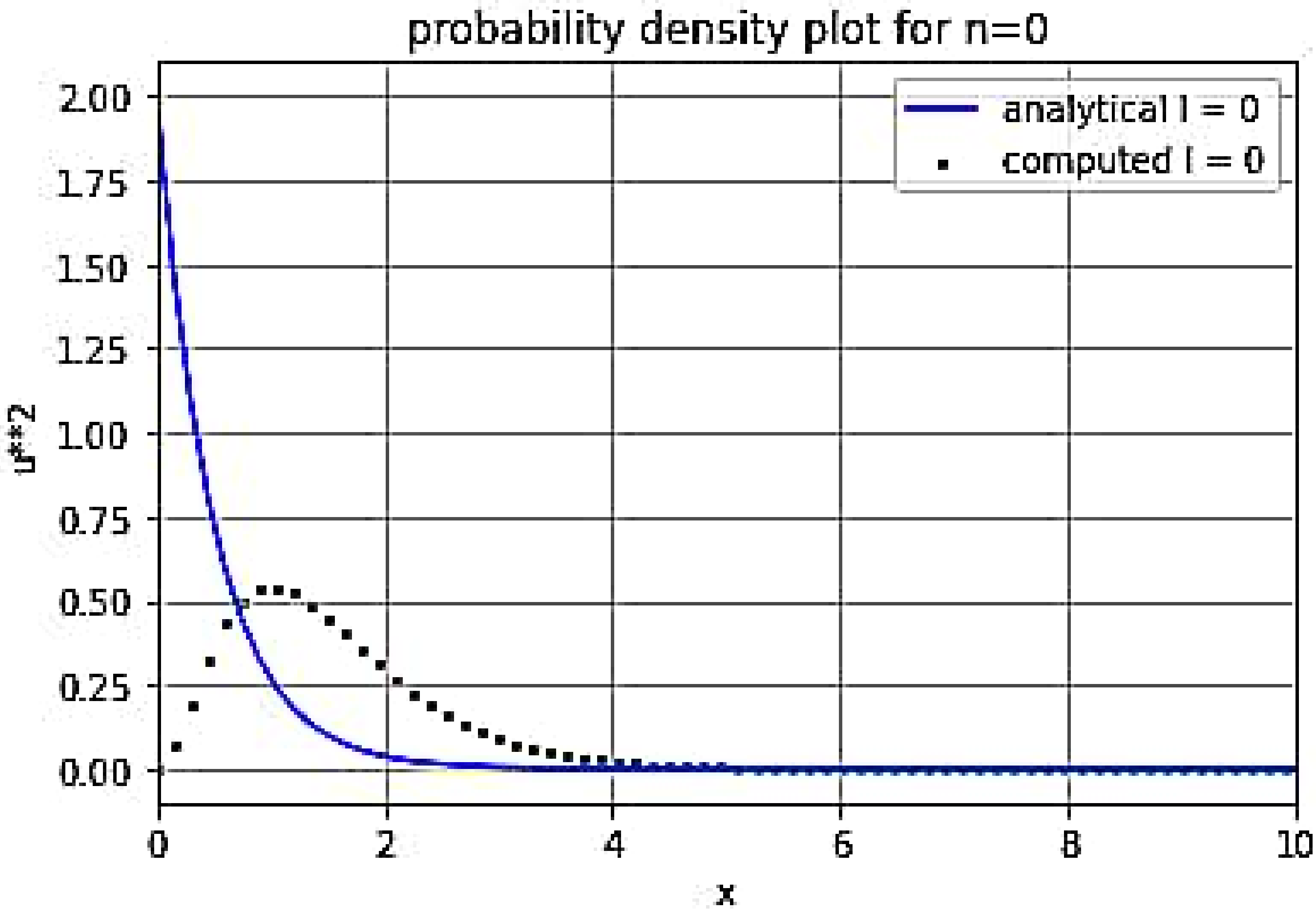
plot of potential vs x
1e29
1.2
1.0
0.8
0.6
0.4
0.2
0.0
v(r)
0
20
40
60
80
100
120
140
r
V_eff for l=1
V_eff for l=2
V_eff for l=3
V(r)

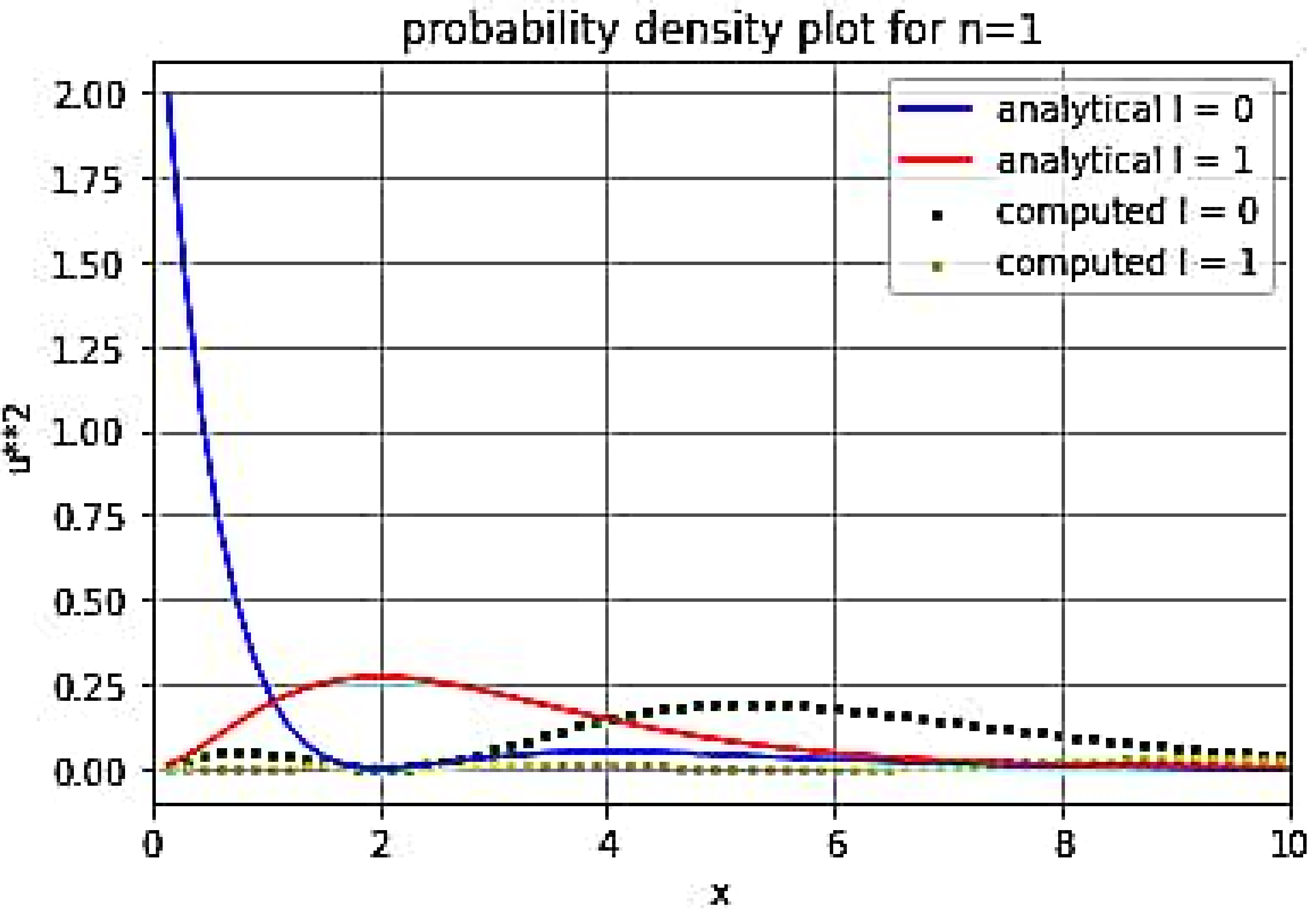

PLOT FOR U VS X FOR N = 1
analytical
computed
U
X
0
10
20
30
40
50
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4

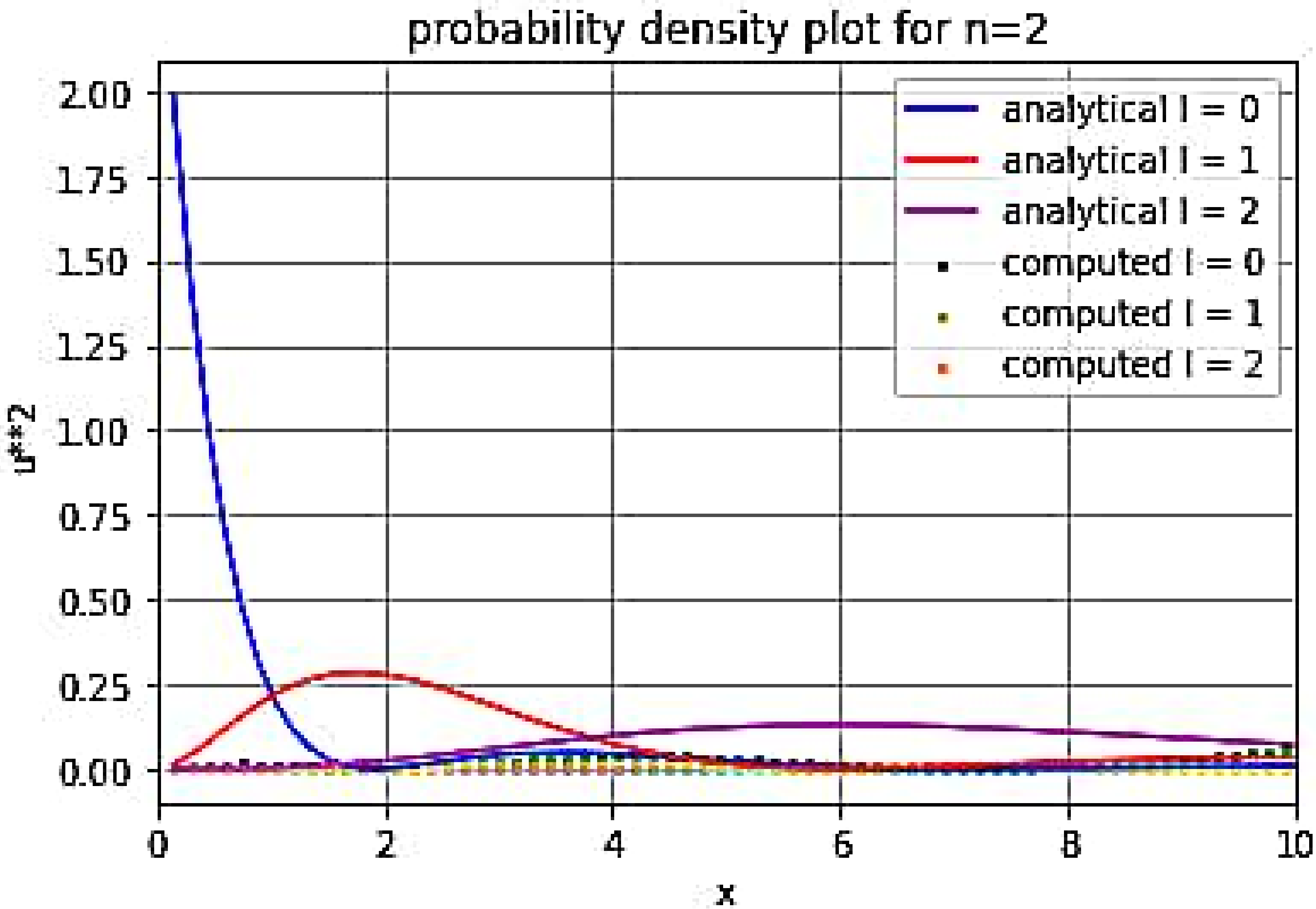

PLOT FOR U VS X FOR N = 2
analytical
computed
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
U
0
10
20
30
40
50
X


PLOT FOR U VS X FOR N = 3
analytical
computed
U
X
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
10
20
30
40
50


PLOT FOR U VS X FOR N = 4
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
U
analytical
computed
0
10
20
30
40
50
X


probability density plot for n=0
analytical l = 0
computed l = 0
u**2
x
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10


probability density plot for n=1
analytical l = 0
analytical l = 1
computed l = 0
computed l = 1
u**2
x
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0
2
4
6
8
10


probability density plot for n=2
analytical l = 0
analytical l = 1
analytical l = 2
computed l = 0
computed l = 1
computed l = 2
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
u**2
0
2
4
6
8
10
x