Computational Physics
Chapter 3 Problems

Cristopher Cerda Puga

Problem 3.5

Consider the potential $v(y) = -\dfrac{1}{1+(y/10)^2}$.

- How many bound states are there? Use the shooting method to calculate the energies;

Listing 1: Method to solve numerically Schrödinger Equation.

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

# define schr dinger equation
def sch(y, x, beta, epsilon):
    phi, dphi = y
    dydx = [dphi, -1./(1. + (beta*x/10)**2)*phi - epsilon*phi]
    return dydx

# evolve the schr dinger equation
def evolve(L, beta, epsilon):
    phi0 = [1.0,0.0] #initial condition for phi(y)
    x = np.linspace(0,L,501)
    from scipy.integrate import odeint
    sol = odeint(sch, phi0, x, args=(beta, epsilon))
    val = sol[:, 0][[500]]
    return val[0]

L=20
beta=1
epsmin = -1.0
epsmax = 0.0
nmax = 10**3
val1 = []
val2 = []
for j in range(nmax):
    epsilon = epsmin + (epsmax-epsmin)*j/(nmax-1)
    val2.append(np.log(abs(evolve(L,beta,epsilon))))
    val1.append(epsilon)

plt.plot(val1,val2)
plt.xlabel('$\epsilon$')
plt.ylabel('$\log(|\phi(20)|)$')
plt.show()
```

There are 5 *possible* bound states, as we can see in the next figure:

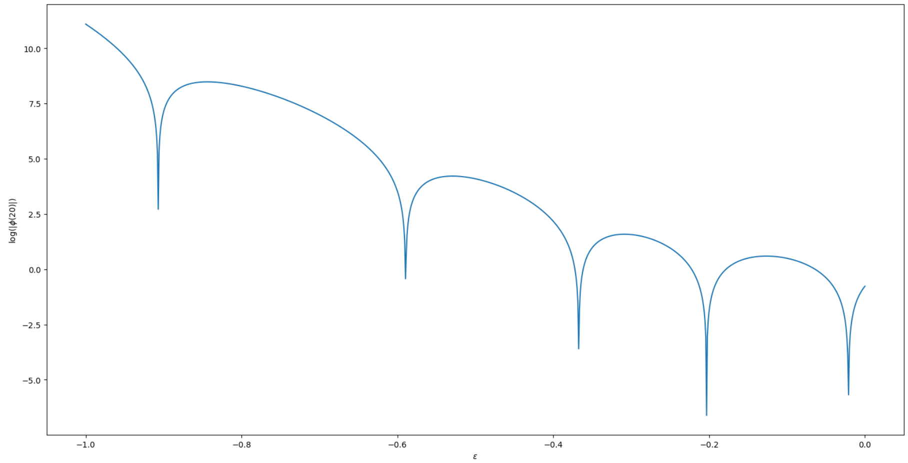

Figure 1: Bound states for the potential $v(y) = -\dfrac{1}{1 + (y/10)^2}$

Now, using the shooting method to calculate their energies:

- From the left to the right, the eigenvalues ϵ are:

$$[-0.906983436504379, -0.5893558163661508, -0.3672235061178116].$$

Listing 2: Method to find the eigenvalues.

```
#defines the function
def f(eguess):
    f = evolve(L,beta,eguess)
    return f

#implements the bisection method
def bisection(x0,xf,nmax,tol):
    xm = 0
    for i in range(nmax):
        f0 = f(x0)
        xm = (x0 + xf) / 2.0
        fm = f(xm)
        if f(xf)*f(x0) > 0:
            return None
        if fm == 0 or abs(x0 - xf) < tol:
            break
        if f0*fm < 0:
            xf = xm
        else:
            x0 = xm
    return xm

tol = 1e-10
xl = -1.0
xr = -0.8

root = bisection(xl,xr,1000,tol)
print(root)
```

- Calculate (and plot) the wave functions of each bound state:

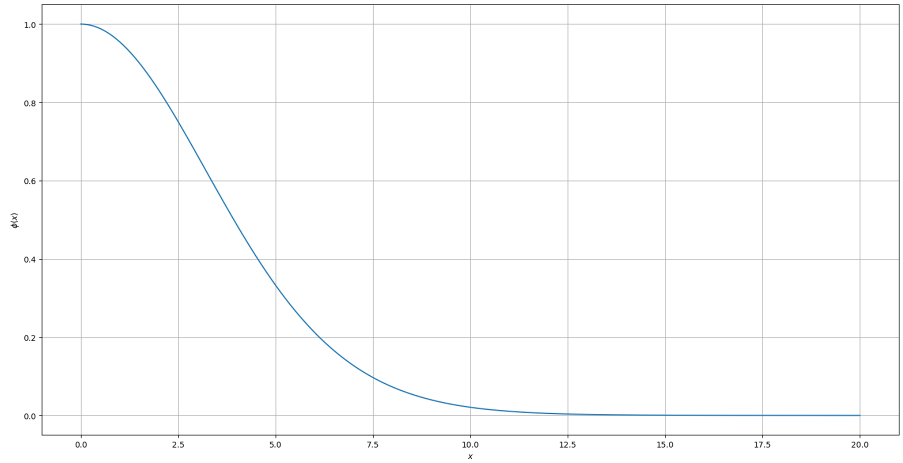

Figure 2: $\phi(y)$ for the potential $v(y) = -1/(1+y/10)^2$ corresponding to the eigenvalue $\epsilon = -0.906983436504379$

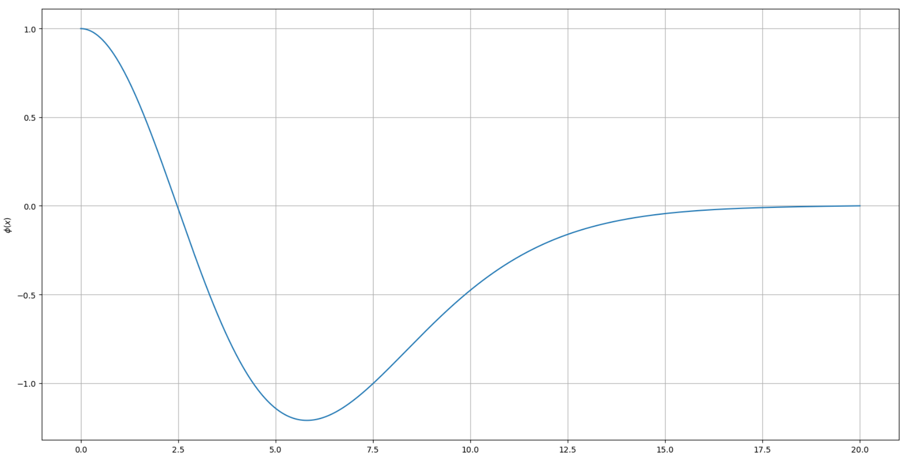

Figure 3: $\phi(y)$ for the potential $v(y) = -1/(1+y/10)^2$ corresponding to the eigenvalue $\epsilon = -0.5893558163661508$

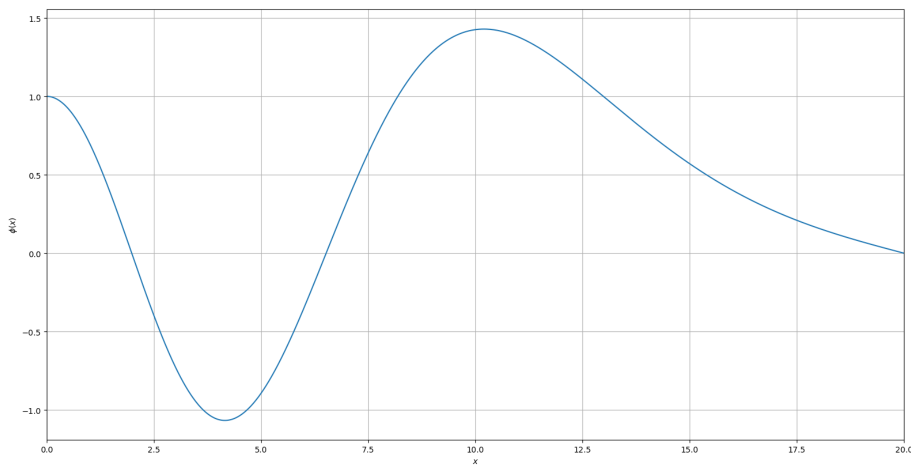

Figure 4: $\phi(y)$ for the potential $v(y) = -1/(1+y/10)^2$ corresponding to the eigenvalue $\epsilon = -0.3672235061178116$

- Modify the programs to calculate the odd bound states; how many bound states are there?

  We only need to modify the initial condition; instead of (1,0) we should use (0,1). Doing so, we find the next *possible* bound states:

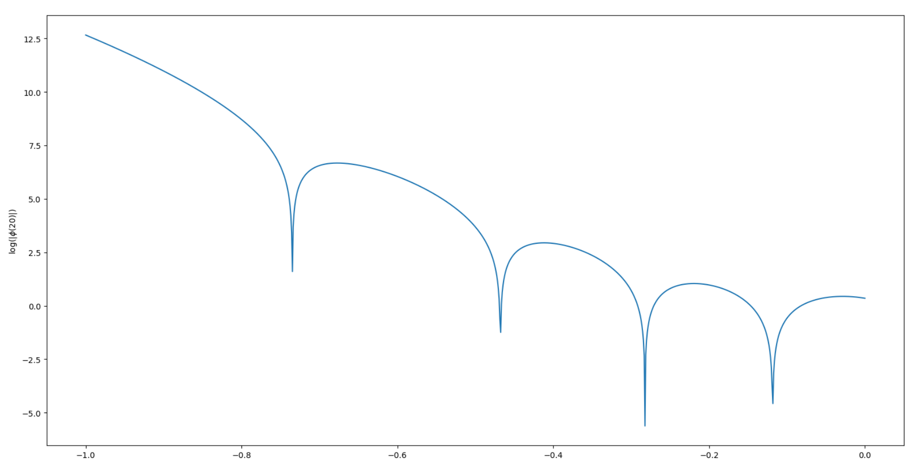

Figure 5: Possible odd bound states for the potential $v(y) = -\dfrac{1}{(1+y/10)^2}$.

  Now, using the shooting method to calculate their energies:

  – From the left to the right, the eigenvalues ϵ are:

$$[-0.7348752335762829, -0.4678218032831767].$$

- Calculate (and plot) the wave functions of the odd bound states;

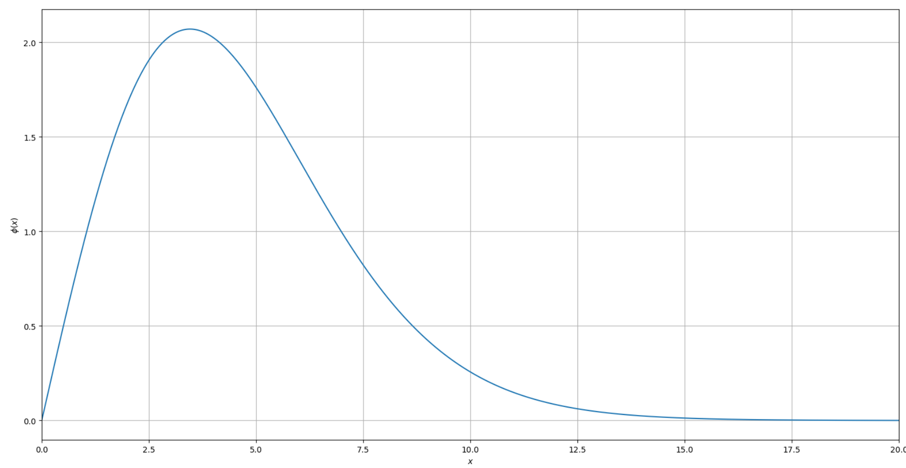

Figure 6: $\phi(y)$ for the potential $v(y) = -1/(1 + y/10)^2$ corresponding to the eigenvalue $\epsilon = -0.7348752335762829$

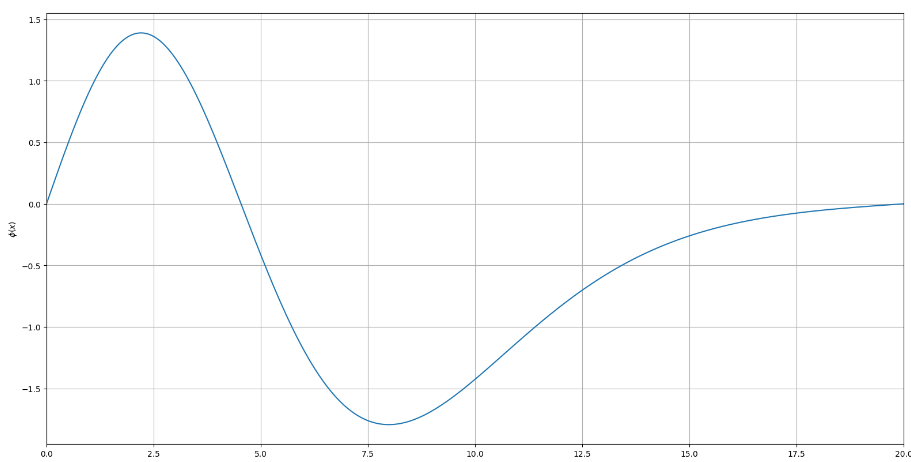

Figure 7: $\phi(y)$ for the potential $v(y) = -1/(1 + y/10)^2$ corresponding to the eigenvalue $\epsilon = -0.4678218032831767$

Exercises 3.6

Consider the Scrhödinger equation for the simple harmonic oscillator:

$$-\frac{\hbar^2}{2m}\frac{\mathrm{d}^2\psi}{\mathrm{d}x^2} + \frac{1}{2}m\omega^2 x^2 \psi = E\psi$$

Notice that in this case there is an infinite number of bound states.

- Convert this equation to a dimensionless form;
  Let us write the differential equation in the next equivalent way:

$$\frac{\mathrm{d}^2\psi}{\mathrm{d}x^2} - \left(\frac{m\omega x}{\hbar}\right)^2 \psi = -\frac{2mE}{\hbar^2}\psi$$

We define a dimensionless variable $\xi = \sqrt{\frac{m\omega}{\hbar}}x$, so that

$$\frac{\mathrm{d}}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}\xi}\frac{\mathrm{d}\xi}{\mathrm{d}x} = \sqrt{\frac{m\omega}{\hbar}}\frac{\mathrm{d}}{\mathrm{d}\xi} \implies \frac{\mathrm{d}^2}{\mathrm{d}x^2} = \frac{m\omega}{\hbar}\frac{\mathrm{d}^2}{\mathrm{d}\xi^2}$$

Therefore, in terms of ξ the Schrödinger equation reads:

$$\frac{\mathrm{d}^2\psi}{\mathrm{d}\xi^2} - \frac{m\omega}{\hbar}x^2\psi = -\frac{2E}{\hbar\omega}\psi$$

Finally, introducing our dimensionless variable:

$$\frac{\mathrm{d}^2\psi}{\mathrm{d}\xi^2} = (\xi^2 - K)\psi$$

where $K = 2E/\hbar\omega$.

- Use the shooting method to estimate the energies of the first few even and odd bound states and make a plot where on the horizontal you put the number of bound state (starting from 0) and on the vertical axis you put the energy; can you guess the functional form?

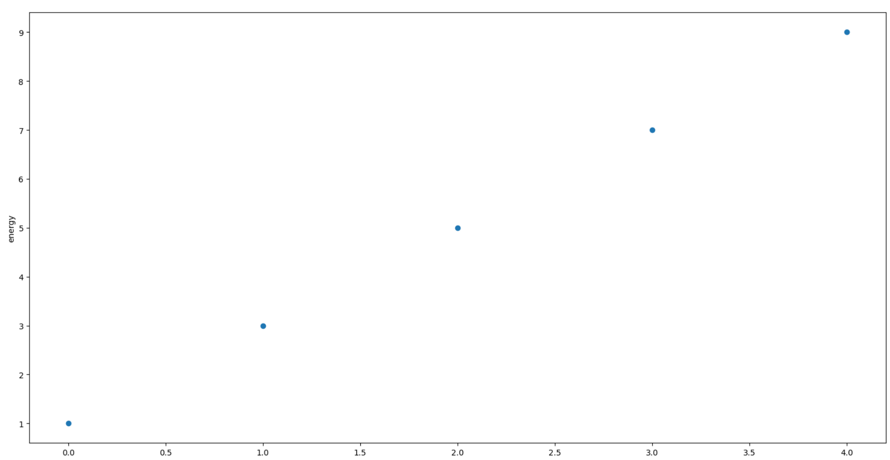

Figure 8: Energies and bound states

So that we can see that the energy corresponding to the bound states has the form $K_n = (2n + 1)$, where n is the number of bound state. And since $K_n = 2E_n/\hbar\omega$, then $E_n = \left(n + \frac{1}{2}\right)$ as expected for the quantum harmonic oscillator ($\hbar = 1, \omega = 1$).

- Plot the first few even and odd wave functions;

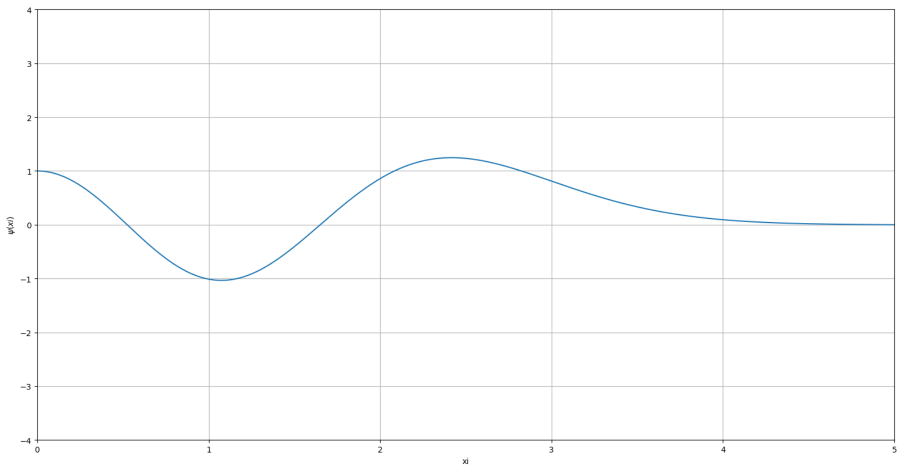

Figure 9: Wave function corresponding to $K = 9.000025350542273$

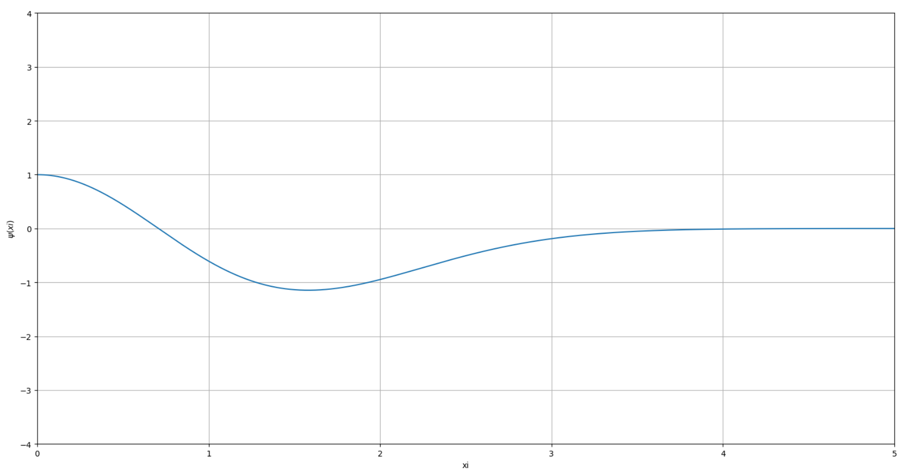

Figure 10: Wave function corresponding to $K = 5.000000213331077$

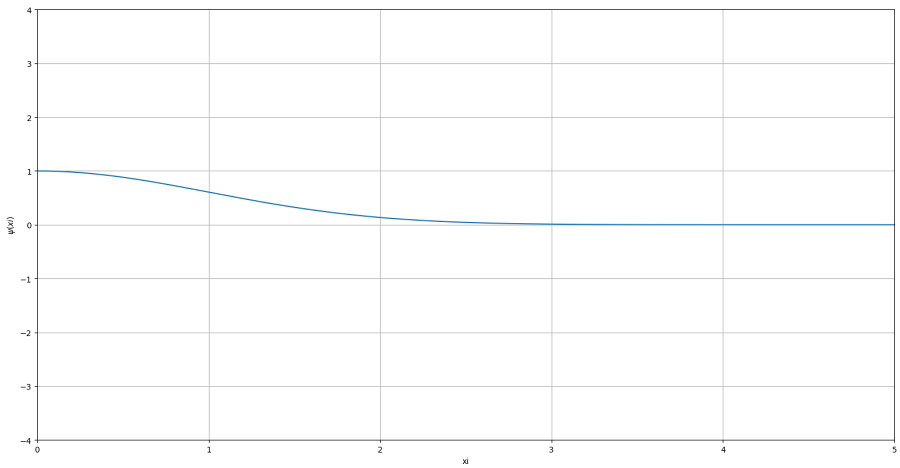

Figure 11: Wave function corresponding to $K = 0.999999959662091$

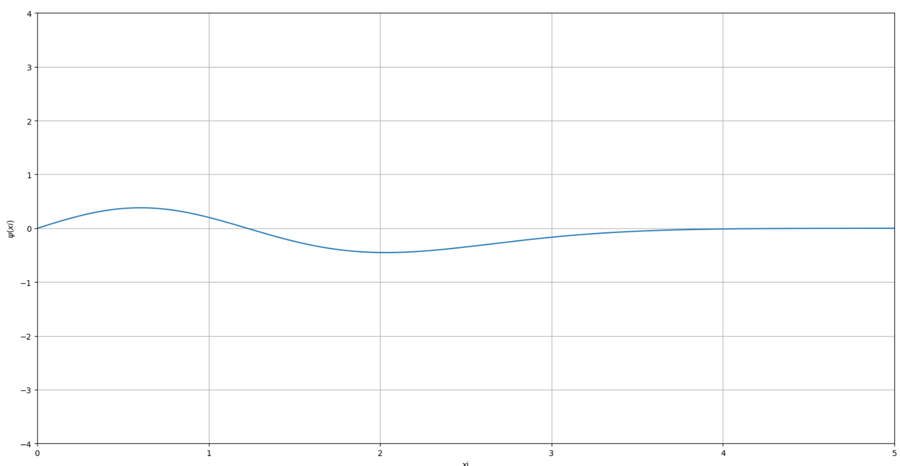

Figure 12: Wave function corresponding to $K = 7.000002433254849$

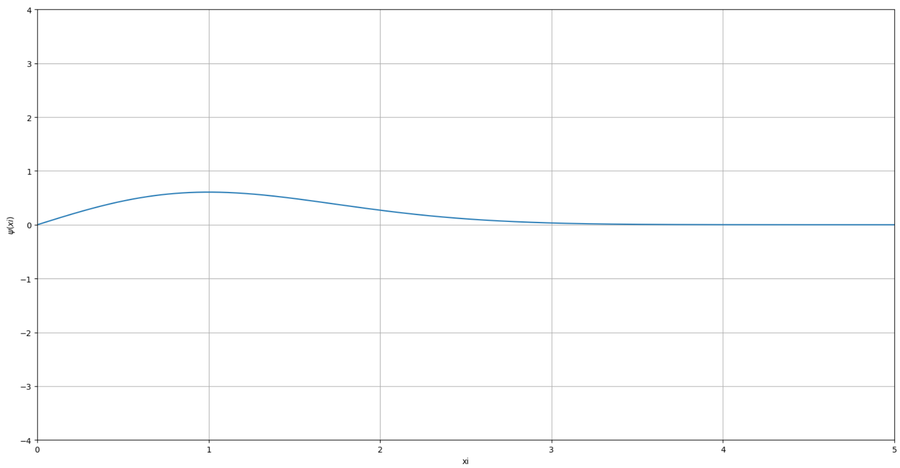

Figure 13: Wave function corresponding to $K = 2.9999999884166755$

Listing 3: Finding the bound states.

```
import numpy as np
import matplotlib.pyplot as plt


#define schrodinger equation:

def sch(y,xi,k):
    phi,dphi = y
    dydx = [dphi, (xi**2-k)*phi]
    return dydx

def evolve(L,k):
    phi0 = [0.0,1.0]
    xi = np.linspace(0,L,501)
    from scipy.integrate import odeint
    sol = odeint(sch,phi0,xi,args = (k,))
    val = sol[:,0][[500]]
    return val[0]


L=5
kmin = 0.0
kmax = 10.0
nmax = 10**3
val1 = []
val2 = []
for j in range(nmax):
    k = kmin + (kmax-kmin)*j/(nmax-1)
    val2.append(np.log(abs(evolve(L,k))))
    val1.append(k)


def f(eguess):
    f = evolve(L,eguess)
    return f

#bisection method to find root
```

```
def bisection(f,xl,xr,tol):
    while (xr - xl) / 2.0 > tol:
        xm = (xl + xr) / 2.0
        if f(xm) == 0:
            return xm
        elif f(xl)*f(xm) < 0:
            xr = xm
        else:
            xl = xm
    return (xl + xr) / 2.0

xl = # guess
xr = # guess
tol = 1e-10
root = bisection(f,xl,xr,tol)
print(root)

plt.plot(val1,val2)
plt.show()
```

Listing 4: Plotting the wave function for each bound state.

```
import numpy as np
import matplotlib.pyplot as plt

def sch(y,xi,k):
    phi,dphi = y
    dydx = [dphi, (xi**2-k)*phi]
    return dydx

def evolve(L,k):
    phi0 = [0.0,1.0]
    xi = np.linspace(0,L,1000)
    from scipy.integrate import odeint
    sol = odeint(sch,phi0,xi,args = (k,))
    return sol

L= 5
#k =  # 9.000025350542273 # 5.000000213331077  # 0.999999959662091 # even bound_states
#k = # 7.000002433254849 # 2.9999999884166755  # odd bound_states
sol = evolve(L,k)
x = np.linspace(0,L,1000)

plt.ylim(-4,4)
plt.xlim(0,L)
plt.xlabel('xi')
plt.ylabel('$\psi(xi)$')
plt.grid()
plt.plot(x,sol[:,0])
plt.show()
```