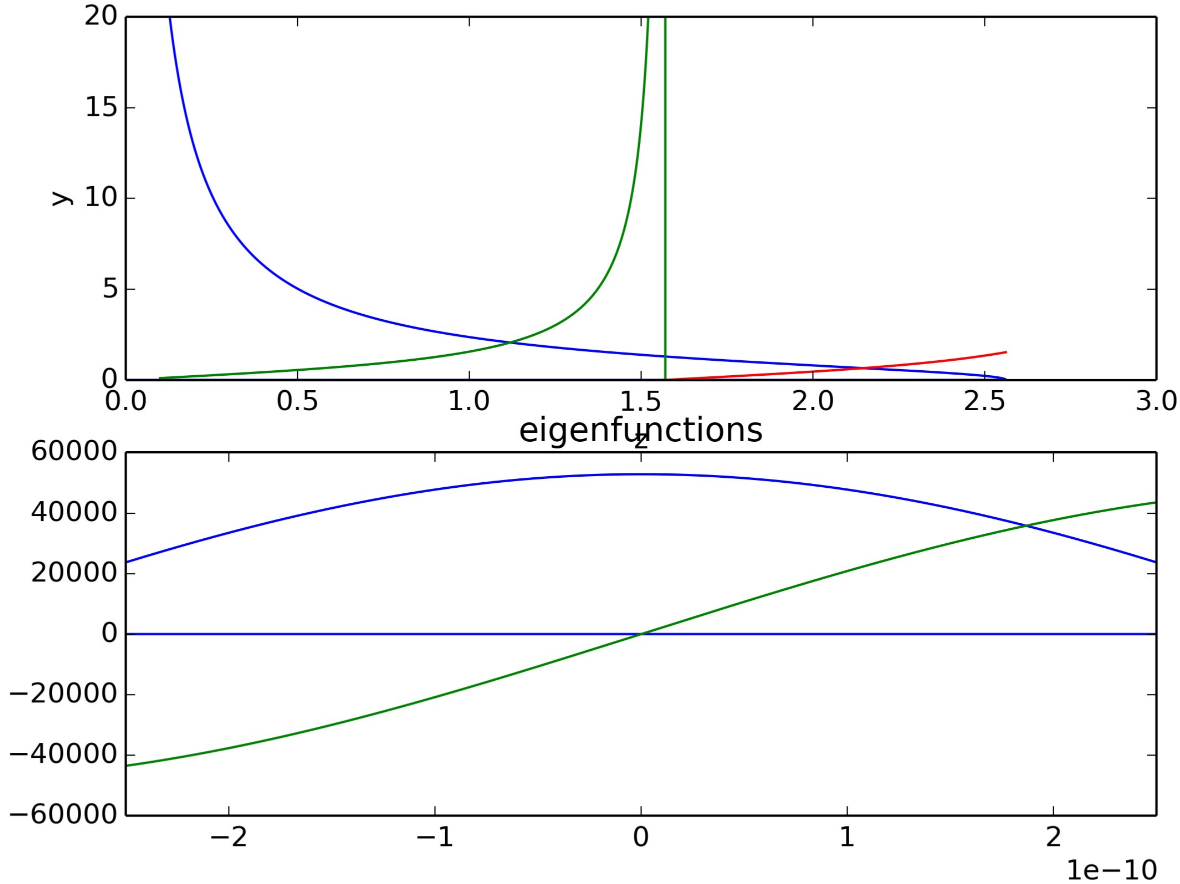
Ting-Jui, Hsu(Matr-Nr. 351218)

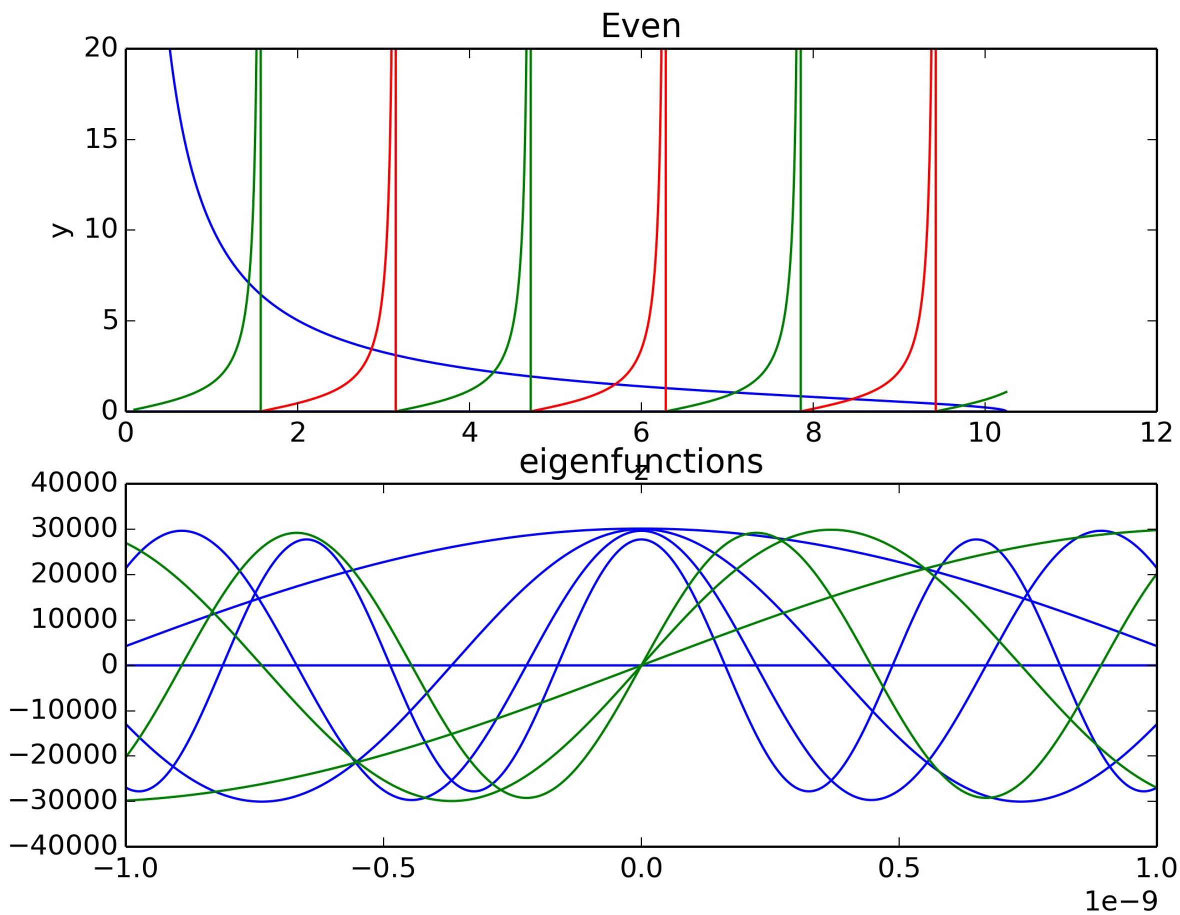
Exercise Sheet 3

1.

When the length of the well is 5Å, the eigenenergies, E+V0, of all bound states are 2.98*10^-29 Joule and 1.08*10^-28 Joule. And the plot of the normalized eigenfunctions is plotted in below. The ratio of E1 in infinite potential well and the finite potential well is 1.24*10^-10, and The ratio for the E2 is 1.12*10^-10.



When the length of the well is 20Å, the eigenenergies, E+V0, of all bound states are
1.25*10^-29 Joule,
4.96*10^-29 Joule,
1.11*10^-28 Joule,
1.96*10^-28 Joule,
3.03*10^-28 Joule,
4.30*10^-28 Joule,
5.71*10^-28 Joule.
And the plot of the normalized eigenfunctions is plotted in below.
The ratio of each E for the infinite potential well and finite potential well is listed below
For E1: 5.18700878723e-11
For E2: 5.14551271694e-11
For E3: 5.11784867007e-11
For E4: 5.08326861148e-11
For E5: 5.0293237201e-11
For E6: 4.95647506335e-11
For E7: 4.83556247755e-11

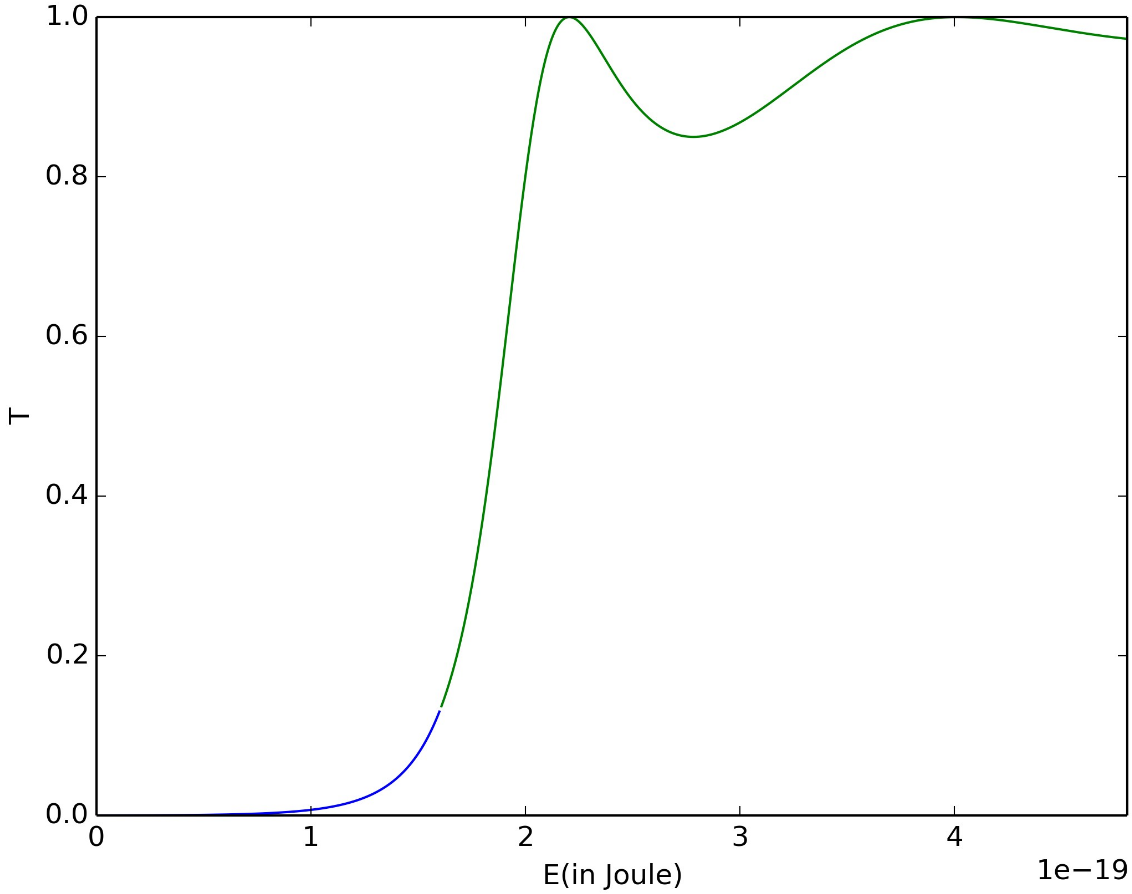


2.

i.

When E<V0: $T^{-1} = : + \frac{V_0^2}{4E(V_0 - E)} \sinh^2 \left(\frac{2a}{\hbar} \sqrt{2m(V_0 - E)} \right)$

When E>V0: $T^{-1} = 1 + \frac{V_0^2}{4E(E - V_0)} \sin^2 \left(\frac{2a}{\hbar} \sqrt{2m(E - V_0)} \right).$



ii.

iii.

For the energies where transmission is largest, the wave-length of the electron are

1.05nm and 0.775nm

in this case.

From the solution we derived in i., we can find that when T=1, when $\left(\frac{2a}{\hbar} \sqrt{2m(E - V_0)} \right)$ =nπ, which means E-V is precisely the allowed energies for the infinite square well.

Source code for 1.

```
import matplotlib.pyplot as plt
import numpy as np
from scipy import constants as const

#set the given parameter
hbar     = const.hbar  #in J*s
L        = 2.5e-10#in meter
M        = const.m_e
V0       = 4* 1.602176565e-19 #ev into J

interval = 5000
z0 = (L / hbar)*((2*M*V0)**0.5)


#set the plot
z1 = np.linspace(0,z0, interval)
x1 = z1*0.
z2 = np.linspace(0.1,z0, interval)
x2 = (((z0/z2)**2)-1)**0.5
z3 = np.linspace(0.1,z0, interval)
x3 = np.tan(z3)
z4 = np.linspace(0.1,z0, interval)
x4 = -(1./np.tan(z4))

#finding eigenvalue
t=-1
u=-1
p=0
q=0

z_even= []
z_odd = []
z_even_count =-1
z_odd_count =-1

for i in z2:

        if  abs((((z0/i)**2)-1)**0.5 - np.tan(i)) < 10**-1 and i-t>0.5*np.pi:
                t=i
                print "When z=", i,",E0+V0 = ",(hbar**2)*(i**2)/(2*M*L),"Joule."
                print (((z0/i)**2)-1)**0.5 - np.tan(i)
                z_even=z_even+[i]
                z_even_count = z_even_count+1
        if  abs((((z0/i)**2)-1)**0.5 + (1./np.tan(i))) < 10**-1 and i-u>0.5*np.pi:
                u=i
                print "When z=", i,",E0+V0 = ",(hbar**2)*(i**2)/(2*M*L),"Joule."
```

```
        print (((z0/i)**2)-1)**0.5 +(1./np.tan(i))
        z_odd=z_odd+[i]
        z_odd_count = z_odd_count+1




#plot setting
plt.subplot(211)
plt.plot(z1,x1)
plt.plot(z2,x2,"b")
plt.plot(z3,x3,"g")
plt.plot(z4,x4,"r")
plt.xlabel("z")
plt.ylabel("y")
plt.ylim(0,20)

plt.subplot(212)
z1 = np.linspace(-L,L, interval)
x1 = z1*0.
plt.plot(z1,x1)
kappa=(((z0**2)-(z_even[0]**2))**0.5)/L
z5= np.linspace(-L, L, interval)
x5= (1/((L+(1/kappa))**0.5)*np.cos((z_even[0])*z5/L))
plt.plot(z5,x5,"b")

kappa=(((z0**2)-(z_odd[0]**2))**0.5)/L
z7= np.linspace(-L, L, interval)
x7= (1/((L+(1/kappa))**0.5)*np.sin((z_even[0])*z7/L))
plt.plot(z7,x7,"g")

plt.title("eigenfunctions")


plt.xlim(-L,L)


plt.show()
#plt.savefig("hw3-1-5A.png",dpi=300,format="png")
```

Source code for 3.i.

```
import matplotlib.pyplot as plt
import numpy as np
from scipy import constants as const

#set the given parameter
hbar     = const.hbar  #in J*s
L        = 5e-10#in meter
M        = const.m_e
V0       = 1* 1.602176565e-19 #ev into J
Emax     = 3* 1.602176565e-19 #ev into J


interval = 500
E1 = np.linspace(10e-39,V0, interval)
T1 = 1.0/(1.+((V0**2)/(4*E1*(V0-E1)))*((np.sinh(((2*L)/hbar)*((2*M*(V0-E1))**0.5)))**2))

E3 = np.linspace(V0,Emax, interval)
T3 = 1.0/(1.+((V0**2)/(4*E3*(E3-V0)))*((np.sin(((2*L)/hbar)*((2*M*(E3-V0))**0.5)))**2))

# calculate the wave-length
print const.h/((2*M*2.206e-19)**0.5)
print const.h/((2*M*4.010e-19)**0.5)

#plot setting
plt.plot(E1,T1)

plt.plot(E3,T3)
plt.ylabel("T")
plt.xlabel("E(in Joule)")
plt.xlim(0,Emax)
plt.show()
plt.savefig("hw3-2-barrier_test.png",dpi=300,format="png")
```
