Modelling a Finite Well Inside an Infinite Square Well

By: Zheng Yi Wang, 20580180 For: CHEM 350, Winter 2019

# Thesis

Infinite wells and finite wells are standard examples of particles in a box, however when you put these two models together and you change the potential of the finite square well to include more bound states you can effectively model both models as one infinite square well.

# Motivation

Although it’s already been proven mathematically, as the finite well’s potential goes to infinity it can be essentially modelled as an infinite well, it is still unclear as to how this system would actually accomplish this. By making a model that emulates a finite well (where only bound states contribute to the energy) inside an infinite well (well all states are bounded) we can get an idea of exactly what happens as the potential goes to infinite in terms of an actual infinite potential well.

# Description and Implementation of Method

**Description**

How I went about this is by defining a function which calculates the energies of all the bound states inside the finite well. It was difficult to conceptualize how the code will work, so it essentially boiled down to: Find Bound States🡪Find Discrete Energies🡪Re-iterate increasing potential or box width.

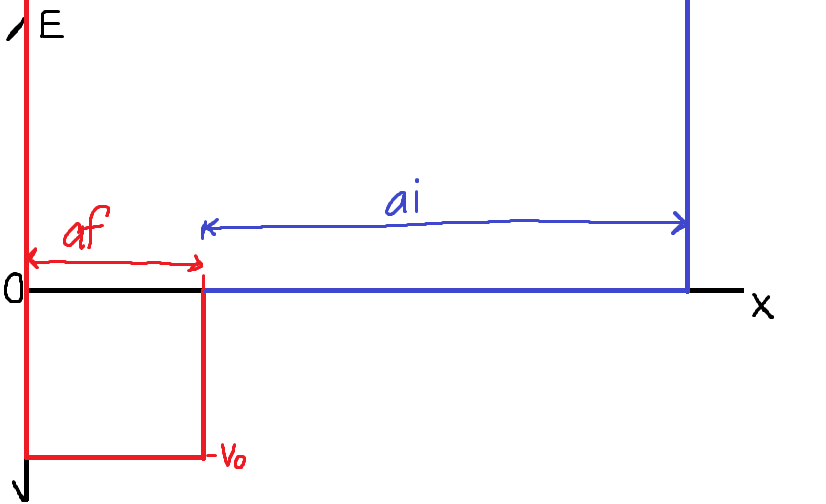


Figure 1: Physical model of finite well (red) inside infinite well (blue of the right)

Most of the work was put into making sure the function to find the energy of the finite well was correct, then I re-iterated the potential and box width through simple for loops and divided the finite well’s energy with the infinite well’s energy. The model of the whole function (the finite well inside the infinite well) should physically look like Figure 1, where the infinite well’s energy is the remainder of the well.

**Implementation**

As stated before, the function which calculates the energies inside the finite well was worked on the most. The theory was taken from “Introduction to Quantum Mechanics Second Edition” by David J. Griffiths, I won’t go over the derivation of the finite well’s energy, but it essentially is taking the boundary conditions and manipulating the fact that the wavefunction is a continuous function.1 The bound states were found by these transcendental equations (line 44 to 53):

(Eq.1)

(Eq.2)

From this we can get values for z, which is directly based off of energy, where:

I plotted the both transcendental equations (line 55 to 62), and found the intercepts by finding when the sign changed when the function with the square root subtracted the trigonometric functions (lines 27 to 30, 37 to 42, and 51 and 52). (Figure 2)

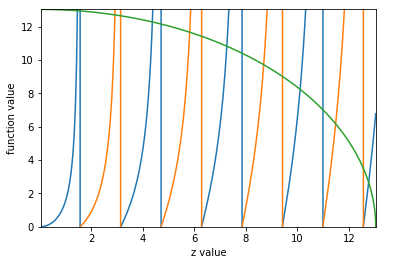


Figure 2: Example plot of equation 1 and equation 2, where the green line would be the square root value. Can be done with mass set to 10.

One problem I had is there would be two intercepts per bound state, so I did some Python trickery (I won’t go into detail because it’s off topic) to get the first intercept (lines 66 to 75). Then I did a sum up of all the bound states’ energies (lines 79 to 86) and the function should return the total energy inside the finite well.

As for constants I set them to very “tame” values so I didn’t get any range errors, however I still did get range errors for low masses where the variables would round to 0 despite there having at least 1 bound state (Figure 3). This only happened for re-iterating a value, so it didn’t really matter, and you’ll see why in the results section.

1Griffiths, D.J. (2017) *Introduction to Quantum Mechanics*. Cambridge University Press

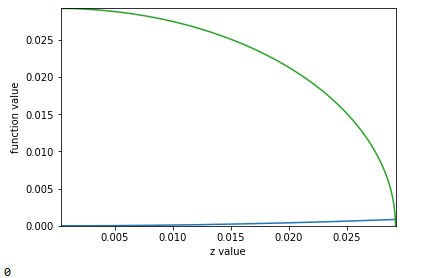


Figure 3: Example of when there wouldn’t be a bound state despite there having one

The last step would be to re-iterate potentials and see what we get. Re-iterating the potentials and the box width itself is pretty simple, just a for loop (lines 90 to 107). I made the potential as a function of the energy of the infinite well so it doesn’t go crazy with insane values that I can’t really use, and I made the box width dependent on each other, so if infinite well potential changes with each iteration as well. I also made a ratio of the infinite well energy to the finite well energy to have an idea of what we’re dealing with.

# Results and Discussion

The results from the tests show that as the potential of the finite well the energy of the well increases (duh), and the energy of the infinite well part stays the same. So far so good, but as the potential approaches the energy value of the infinite well portion of the well, the ratio of finite well energy to infinite well energy gets larger and larger, well above 1. From data with mass set to 1, for potential equal to 99% of the infinite well energy we get a ratio of around 5.6 (Figure 4). This was expected since it was already proven mathematically that as potential goes up the finite well can be modelled as an infinite well, and the results here seem that it’s very “eager” to become an infinite well.

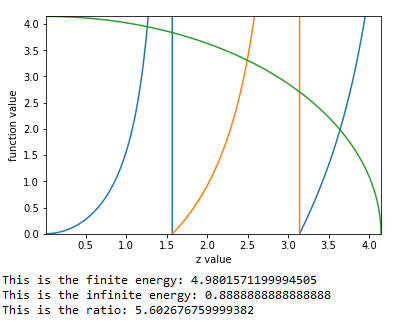


Figure 4: This is with Vo=0.99\*Infinite well energy and mass =1

As for the box width iterations, the ratio mention above actually decreases as the width of the finite well slowly becomes the whole well. This was also expected since the infinite well’s energy would typically increase with smaller box width, so if infinite well’s energy is increasing then the ratio would stagnate or decrease; which is what we saw (Figure 5).

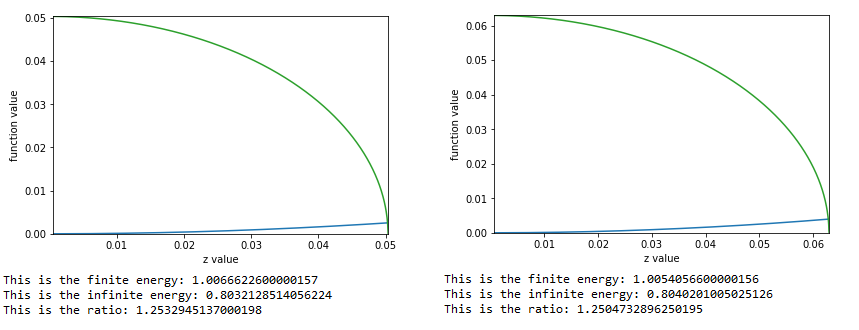


Figure 5: This is with increasing a value from left to right

One concern that I have, which sort of undermines the whole basis of my program, is with the derivation of the equations. The equations are derived on the basis that the unbound particles are free particles, however since it’s in a finite well all states are bound. If I had another week, I would re-derive the transcendental equations to add in another term to accommodate for that extra variable, and see if it changes my program. Other than that concern, I think this is a good place to leave off at.

# Conclusion

Very briefly: I set out to model and investigate changing parameters of a finite well inside an infinite square well, and I can see that it worked out exactly the way I had hypothesized. The finite well energy goes to infinite with increase in potential and doesn’t change with increase in box width.

## Code (for Python)

#Made by Zheng Yi Wang for CHEM 350, February 12th 2019

#Defining some constants and importing some libraries

#we assume that the potential of the infinite square well is 0

import numpy as np

import math

import matplotlib.pyplot as plt

pi=3.1415

hbar=1/pi

m=1

a=10#length of the total well

af=1#length of the finite well

ai=a-af#length of the infinite well

Ei=64/(8\*ai) #energy of the infinite sqaure well

def Energyfinite(Vo,af): #function to find energy of the finite well

######### arrays for the intercepts

symint=[]

asymint=[]

symbound=[]

asymbound=[]

#lists for graphs

sym=[] #for even parity

asym=[] #for odd parity

blist=[]

zlist=[]

#########

qo=1.0 #this is used to find the intercepts

q=1.0

p=1.0

po=1.0

E=-Vo+0.00001 #used to scan over an energy

z=(af/hbar)\*(2\*m\*(Vo+E))\*\*0.5 #refer to discussion

zo=(af/hbar)\*(2\*m\*Vo)\*\*0.5 #taken from Griffiths

######

######This finds all the bound states within the finite well

while zo>=z: #while the particle is bound

if q==-qo: #this is also part of finding when the b value intersects

symint.append(z)

qo=q

elif p==-po:

asymint.append(z)

po=p

################################

z=(af/hbar)\*(2\*m\*(Vo+E))\*\*0.5#calculates the current z value

zo=(af/hbar)\*(2\*m\*Vo)\*\*0.5

b=((zo\*\*2-z\*\*2)\*\*0.5) #calculates the current b value

sym.append(z\*(math.tan(z)))

asym.append(-z/(math.tan(z)))

blist.append(b)

zlist.append(z)

p=(np.sign(b+z/(math.tan(z))))#sees if it intersected or not

q=(np.sign(b-z\*(math.tan(z))))

E=E+0.0001 #this is to make sure we have a nice smooth curve

###################

plt.plot(zlist,sym)

plt.plot(zlist,asym)

plt.plot(zlist,blist)

plt.ylabel("function value")

plt.xlabel("z value")

plt.ylim(0,zo)

plt.xlim(zlist[0],z)

plt.show()

#this is for showing the plot of the finite state

#####################

#this filters all the intercepts so the first one is counted only

for i in range(0,len(symint)):

if i%2==0:

symbound.append(symint[i])

else:

continue

for o in range(0,len(asymint)):

if o%2==0:

asymbound.append(asymint[o])

else:

continue

i=o=0 #resets the variable

##################################

Ef=0 #energy of the finite well

for i in range(0,len(symbound)-1):

Ef=Ef+(2\*hbar\*(symbound[i])\*\*2)/(m\*af\*\*2)

if len(symbound)==1:

Ef=Ef+(2\*hbar\*(symbound[0])\*\*2)/(m\*af\*\*2)

for o in range(0,len(asymbound)-1):

Ef=Ef+(2\*hbar\*(asymbound[o])\*\*2)/(m\*af\*\*2)

if len(asymbound)==1:

Ef=Ef+(2\*hbar\*(asymbound[0])\*\*2)/(m\*af\*\*2)

#sums up all the energy from the states in the finite well

return (Ef)

##########################

for x in range (1,99): ##When I change the potential to match

af=1 ##

ai=a-af ##

Vo=0.01\*x\*Ei ##if the whole system was an infinite square well

Ef=Energyfinite(Vo,af)#

print(Ef) ##

print(Ei) ##

print(Ef/Ei) ##

##########################

for y in range(1,30): ##When I make the well as wide as the whole

Vo=0.1\*Ei ##well

af=y\*0.01 ##

ai=a-af #length of the infinite well

Ei=64/(8\*ai) #energy of the infinite sqaure well ##

Ef=Energyfinite(Vo,af)#

print(Ef) ##

print(Ei) ##

print(Ef/Ei) ##

##########################