A Unique, Project-based, Microcourse to Teach the Fundamental Concepts of Quantum Mechanics

Elijah Begin
Ouachita Baptist University

SENIOR THESIS APPROVAL

This Honors thesis entitled

"A Unique, Project Based, Microcourse to Teach the Fundamental Concepts of Quantum Mechanics"

written by

Elijah Begin

and submitted in partial fulfillment of
the requirements for completion of
the Carl Goodson Honors Program
meets the criteria for acceptance
and has been approved by the undersigned readers.

____
Dr. Joseph Bradshaw, thesis director

____
Dr. Sharon Hamilton, second reader

____
Dr. Benjamin Utter, third reader

____
Dr. Barbara Pemberton, Honors Program director

Date:

April 16, 2021

Table of Contents:

Table of Figures:

Outline:

Abstract:

Spyder, a Scientific Python Development Environment, provides an easy-to-use software that can be used to generate data from quantum mechanical systems. This project proposes and explores a microcourse which takes advantage of this utility to teach undergraduates the fundamentals of quantum mechanics. The ease of use allows students with little coding experience the ability to study and explore the most important aspects of quantum mechanics projects. The concepts that this microcourse will teach include the basics of Python programming, the uncertainty principle, wave functions, quantum operators, and Dirac notation. Assignments in the microcourse include solving the harmonic oscillator problem and simulating the absorbance of a molecular aggregate. The main project allows students to build the energy matrix of a molecular aggregate which will be used to produce absorbance graphs. The project includes slides and worked examples for the users as well as discussion on teaching methods. The project-based approach allows students to interact with data and challenge their own learning. This project can be applied to short directed studies for students with an inclination toward physics and/or chemistry. This project provides an opportunity for those seeking an introduction to quantum mechanics to gain an understanding of the fundamentals without the intense rigor of an upper-level undergraduate course.

Introduction:

Personal introduction of the microcourse from the author:

Quantum mechanics is a daunting course for many students. In my personal experience, the first time I learned many of the concepts introduced in the undergraduate course, I had difficulty connecting the topics to meaningful phenomena. In classical mechanics, I could easily imagine the velocity and potential energy in terms of everyday objects like cars and roller coasters. For me, having a visual component to the concepts that I was learning helped to make the mathematical application of the concepts intuitive and reasonable. However, quantum mechanical concepts such as wave functions, uncertainty, or raising operators have little to no real world counterpart. This made application difficult and made learning slow. The solution for me was being able to use all the concepts that I had been taught in one larger project where I could draw meaningful connections and solidify my understanding of the material. I hope that by bringing many of the fundamental concepts together in this microcourse, students may be able to make more meaningful connections between concepts and be able to better retain the material. My hope for this project is to create a course which will make this exciting material of quantum mechanics more engaging for driven students like me who learn through application and visual representation.

Since the inception of quantum mechanics in the 1920-s, quantum mechanics research and application have continuously grown. A knowledge of quantum mechanics is required to be able to effectively study electronics, chemistry, or physics in depth. Quantum mechanics is also applicable in emerging technologies such as quantum computing, nanotechnology, and solar energy conversion. Because of the applicability of quantum mechanics, knowledge of the fundamentals of quantum mechanics is a requirement for most undergraduate physics and chemistry programs. As quantum mechanics is a growing field in both academia and industry, the need for workers in this field will grow as well.

Due to the rigor and uniqueness of quantum mechanics, it is one of the most intellectually challenging courses in an undergraduate science curriculum. In fear that the mathematical rigor may inhibit the students' understanding of the concepts of quantum mechanics, many instructors have studied quantum mechanics course structures that focus only on conceptual learning rather than mathematical application of quantum mechanics.[2-5] One study found that high-schoolers and undergraduate students may benefit from using a “visual–conceptual approach along with real-life applications”.[3] Another study found that “hands-on experience greatly reinforces what has been covered with the lecture demonstrations”.[2] Many of these types of studies provide in-class lecture reinforcement through hands-on activities. However, these studies do not provide insight into how programming could be an effective tool for teaching.

This paper seeks to provide a method for an introduction to quantum mechanics outside of a lecture classroom environment. The microcourse described here provides a brief introduction to the fundamentals of quantum mechanics for students. The microcourse is intended to be implemented in a directed study environment between proctor and student(s). The course should take between three and four weeks. The goal of this course is not to provide a comprehensive understanding of any of the concepts found in quantum mechanics (that would require a full semester course), but rather to lay a foundational understanding of some of the fundamentals of quantum mechanics that can be built on through more study.

This course takes advantage of Spyder, a scientific development environment, which can be learned quickly and used to solve quantum mechanical problems. A few of the problems this course will cover are also covered in a typical quantum mechanics class such as particle-in-a-box and quantum harmonic oscillator problems. The main project is unique because it teaches students to build the Hamiltonian matrix of an approximate molecular aggregate using a one-particle approximation. The main project incorporates many foundational concepts of quantum mechanics (Hamiltonian operators, Hamiltonian matrices, Bra-Ket notation, etc.) The incorporation of many concepts of quantum mechanics into one hands-on project may provide students a way to grasp the applied concepts in a more memorable way. It also gives the students a chance to apply the concepts they learn to a physical problem. Encompassing the concepts in this way may enhance the learning experience of the students before they enter into a more mathematically challenging environment.

Future educational research into the strengths and weaknesses of this approach will need to be done before large scale implementation. This microcourse may be able to help prepare students to begin a regular undergraduate quantum mechanics course. This microcourse may also be used to help undergraduates decide what subject they would like to pursue. However, there are a few prerequisites for students seeking to begin this microcourse. These prerequisites include a high-school level understanding of physics, mathematical concepts such as matrices and vectors, fundamental concepts of chemistry and programming, as well as a college-level understanding of derivative calculus.

Regarding programming, students should be familiar with for-loops, if-else statements, and arrays. It is recommended that individuals have some programming experience to begin this microcourse.

Main Project Description (Simulating Absorbance of Aggregates):

This project mimics the work done by Morgan Sosa in the Wong lab at the University of Oregon.[1] (If you are unfamiliar with quantum mechanics, read this section at the end of the project.)

The goal of the project is to use Spyder to generate an absorption spectrum of a linear aggregate with adjustable parameters. Sosa's work does this and more using a Hamiltonian operator[1] which includes electronic coupling between nearest neighboring molecules/particles. This method is powerful and accurate, however, for educational purposes the operator will be simplified by removing the electronic coupling term; this simplification is called the "one-particle approximation" as it does not account for coupling between neighboring molecules. The simplified operator can be found in Figure 1. An operator is used that does not include electronic coupling in order to reduce complexity and allow for more effective learning. The results will not be as accurate but are still able to illustrate concepts important in quantum mechanics and chemistry. A more in-depth review of this topic can be seen in Sosa's work.[1]

The project focuses on building a Hamiltonian matrix using the Hamiltonian operator shown in Figure 1 for the case in which no electronic coupling is considered between particles.

The discussion below will include a detailed description of the steps that must be followed to complete the mathematical calculation of energy levels which can be used to determine absorption. To see a discussion of the application of this project in Spyder, visit the “Course Schedule and Overview: Week 3”.

The most significant part of any quantum mechanical problem is the Hamiltonian operator that will be used. The operator in Figure 1 has a few parts that are explained as follows: The first term ($\epsilon\, a^{\dagger} a$) extracts the electronic energy of the each wavefunction, the second ($\omega\, b^{\dagger} b$) extracts the vibrational energy of each wavefunction, and the third term ($\omega\, a^{\dagger} a[\lambda\,(b^{\dagger} + b) + \lambda^{2}$) extracts the coupling energy from the electronic and vibrational energy of each state.

Figure 1: Simplified Hamiltonian operator

$$\hat{H} = \epsilon\, a^{\dagger} a + \omega\, b^{\dagger} b + \omega\, a^{\dagger} a[\lambda\,(b^{\dagger} + b) + \lambda^{2}],$$

The operator above is the Hamiltonian where,

ϵ represents the electronic energy,

$a^{\dagger}$represents the electronic raising operator,

a represents the electronic lowering operator,

ωrepresents the vibrational energy,

$b^{\dagger}$ represents the vibrational lowering operator,

b represents the vibrational raising operator,

and λ represents the Huang-Rhys factor.

Figure 2: Dirac notation and formatting

Wavefunction | *A , B* > where "*A*" is the reference number for the electronically excited molecule in the aggregate (0 if the aggregate doesn't contain any electronically excited molecules) and "*B*" is the vibrational energy level on the molecule "*A*".

To begin this project, a list of possible wavefunctions must be generated. To keep track of these states, the Dirac notation shown in Figure 2 is used. This notation consists of only two variables. This notation requires only two variables because a few assumptions about this aggregate are made. The approximation is called the one-particle approximation because of the three assumptions that follow:

1. Only 1 particle in an aggregate will be excited (electronically and/or vibrationally) at a time.

2. The electronic nature of a particle will be that it only exists in two distinct states. These states are either not excited or excited to the first energy level. There will be no higher electronic energy levels on a particle beyond the first excited energy level.

3. The vibrational nature of each particle will be that it only exists in a non-excited/ground state or one of a quantized number of vibrationally excited states. The excited states will consist of integer levels from 1 to "V_{max}", an adjustable parameter, which can be decided based on the aggregate that is being simulated and the computational power that you have.

A list of all possible states can be stored in a 2-dimensional array with each row designating one of all possible states. Once an accurate and complete list of states (ground and excited) is made for the aggregate, these can be used to generate the Hamiltonian matrix. The matrix will be square with the sides of the matrix having the same length as the length of the list of possible states. The matrix should also be symmetric so that the value in entry [row n, column m] is equivalent to the entry in [row m, column n]. The entries of the Hamiltonian matrix will follow the formula found in Figure 3.

Figure 3: Hamiltonian matrix formula

Hamiltonian matrix [row n, column m] = $< \Psi_n | \hat{H} | \Psi_m >$ where Ψ_n is the wavefunction in row n of the array of states, $\hat{H}$ is the Simplified Hamiltonian operator for aggregates found in Figure 1, and Ψ_m is the wavefunction in row m of the array of states. Solving this formula will require following the common rules of operators and to work the equation from right to left. Images and examples of this can be found in the "Lecture Slides" section of Appendix A.

By building the Hamiltonian matrix, one is extracting the energy from every combination of possible wavefunctions and mapping them in that matrix. The properties of the Hamiltonian matrix mimic the properties of a regular Hamiltonian operator so that the eigenvectors of the matrix will be the wavefunctions in vector form and the eigenvalues will be the energy levels.

To determine which transitions are possible through the absorption of light, a dipole moment matrix must be built in a similar fashion to the Hamiltonian matrix. The dipole moment matrix should also be square and symmetric with the side length being equal to the length of the list of states. The entries will obey the formula in Figure 4.

Figure 4: Dipole moment matrix formula

$$\textit{dipole matrix}\,[n, m] = \langle \Psi_n \,|\, \hat{u} \,|\, \Psi_m \rangle \quad \text{where} \quad \hat{u} = a^{\dagger} + a$$

With the wavefunctions in vector form that were gathered from the Hamiltonian as well as the dipole moment matrix, the absorbance of the aggregate can be found and used to build an absorbance graph. The formula for determining the absorbance is found in Figure 5.

Figure 5: Absorbance vector formula

Absorbance = $| \langle G \,|\, \textit{dipole matrix} \,|\, \Psi \rangle |^2$ where $\langle G |$ is the ground state eigenvector/wavefunction and $| \Psi \rangle$ is a two dimensional array with each of the columns of the array as the eigenvectors/wavefunctions found from the Hamiltonian matrix.

When calculated correctly, this should generate a vector containing the proportional absorbance for the allowable transitions from the ground state. The last step is to simply plot the eigenvalues of the Hamiltonian matrix versus the absorbance (preferably using a distribution curve).

Course Schedule and Overview:

Overview:

The project has a recommended total of four lectures and seven assignments that will take roughly three-four weeks depending on the speed of the student and the availability of the proctor. A recommended timeline for the project is shown in Figure 6. Following the timeline is a detailed description of each section of the project.

Figure 6: Project timeline

Week 1: An Introduction to Python:

a. Assignment 1: Downloading Spyder software though Anaconda
b. Lecture 1: A quick walkthrough of Python syntax, numpy, matplotlib, and pandas
c. Assignment 2: Using numpy and pandas with excel file data and graphing in matplotlib
d. Assignment 3: Building numpy data sheets from the kinematic equations

Week 2: The Harmonic Oscillator Project

e. Assignment 4: Watch a 5-minute YouTube video for an introduction to quantum mechanics
f. Lecture 2: An overview of basic physics, chemistry, math, and quantum mechanics concepts for students
g. Assignment 5: Creating a list of all possible wavefunctions for an aggregate
h. Lecture 3: Harmonic oscillator in classical and quantum mechanics using Bra-Ket notation

Figure 6 continued:

i. Assignment 6: Building the Hamilton matrix of the quantum harmonic oscillator

Week 3: The One-Particle Approximation

j. Lecture 4: A review of what students have learned thus far and the introduction of a Hamiltonian for a one-particle approximation

k. Assignment 7: Building the Hamiltonian matrix for a molecular aggregate

l. Lecture 5: Proctor adds in code to make absorbance plots; Discussion of results

Week 4: Additional Projects of higher difficulty (if desired)

Week 1: An Introduction to Python

The goal of week 1 is to familiarize the students with the Python language, the numpy functions list, the matplotlib functions list, and the Spyder software. This, along with the prerequisite knowledge of for-loops, if-else statements, and arrays, will be all the foundational knowledge students require for the programming aspect of the project. Discussion during week one should be slow and simple so that students feel comfortable and confident enough to move forward to more challenging programming tasks. For this reason, the first week contains only a simple walkthrough of Python with three easy tasks that should be accomplished in less than an hour each. If the student struggles too much with these assignments, they may not currently have the adequate programming ability to move forward in the project at this time.

Assignment 1: Downloading Spyder software through Anaconda

Assignment: Download Anaconda, open Spyder, and configure the graphic settings to "Automatic"

Anaconda Individual Edition is a free downloadable software that contains many useful Python compilers, one of which is Spyder. Spyder is specifically for scientific data analysis which makes it simple and easy to use for the purpose of this microcourse. To download follow the steps found in Figure 7.

Figure 7: How to install and configure Spyder

1. *Visit https://www.anaconda.com/products/individual and download for your system*
2. *Follow on screen prompts and use recommended settings while downloading*
3. *Open Anaconda on your computer, then launch Spyder*
4. *Click Tools>Preferences>IPython Console>Graphics*
5. *On the menu, under Graphics Backend select Automatic then click Apply*
6. *Close then reopen Spyder and Anaconda tabs to update settings*

If the steps were correctly followed, Spyder should now be ready to operate for use in this project. If the proctor or student is unfamiliar with Python or Spyder software, they can choose to look at the tutorial that is offered when downloading. However, the fundamentals of Python and Spyder will also be taught in the next section.

Lecture 1: A quick walkthrough of Python syntax, numpy, matplotlib, and pandas

Learning Objective: Following this lecture, the students should understand the structure and function of for-loops, arrays, plots, while-loops, and useful functions from numpy, matplotlib, and pandas.

In this section the course facilitator will describe the basic programming skills needed to be successful in this project with the assumption that the reader already has an understanding of for-loops, while-loops, and arrays. The proctor should walk through the basics of Python found in this section with the student. Students can use the resources in Figure 8 through Figure 10 and/or any others that are readily available on the internet that the students may find helpful. The proctor should be sure to discuss each function below. Some programming details are not covered here. If the students run into any issues, they may need to consult Stack Overflow or other internet resources to assist in their programming.

Unlike other programming languages, Python has been simplified so that variable types don't need to be defined. This simplifies code writing significantly. For teaching purposes, this can reduce the amount of time spent editing code. An example of how this works in comparison to C# is illustrated below in Figure 8.

Figure 8: Variable comparison of C# versus Python

In C#:

```
decimal A = 5;
```

In Python:

```
A=5
```

Python will suggest the type of variable based on the context so that students will not have to manually select the variable. Also, instead of bracetting, Python uses indentation to separate loops. An example of how this works as well as the basic structure of loops and other logic in Python is shown in Figure 9.

Figure 9: Basic loop structure in C# and Python

In C#:

```
for (int i = 0; i < 5; i ++)
{
//code block to be executed
}
```

In Python:

```
block = [0,1,2,3]

for i in block:
        ///Code block to be executed
Code not to be executed in loop
```

or

```
if(///conditional statement):
        ///Code block to be executed
Code not to be executed
```

or

```
while(///conditional statement):
        ///Code block to be executed
Code not to be executed in loop
```

or

```
def CODE_NAME(var1, var2):
        ///Code block to be executed
        return()
Code not defined is CODE_NAME
```

As one can see, in Python the code block in the for-loop is indented ahead of the for-loop itself. To end a for-loop, one continues coding on the indentation level of the indentation. This is the same procedure for *while-loops* and *conditional statements*.

Numpy, matplotlib, and pandas are importable function sets that make programming in Python easier. Numpy contains functions dealing with arrays. Matplotlib is useful for plotting. Pandas is useful for importing data from the computer. Listed in Figure 10 are functions that the student may find helpful in completing this project. The student will use these functions in all upcoming assignments. Also, the HELP section of Spyder has a search function which adds more detail on the specifics of these functions. The final section of Figure 10 labeled "scipy.linalg" is useful later on to diagonalize a matrix and find eigenvectors/eigenvalues of that matrix.

Figure 10: List of helpful functions

1. a=matrix[r,c] - extracts elements from row r column c of matrix
2. matrix[r,c] = a - sets the element in row r and column c of matrix equal to a
3. a=len(matrix) - extracts the length of a matrix/vector
4. max(a,b) - returns the larger of each number

import numpy

1. numpy.arange(x, y, z) - creates a vector from x to (y-1) with step size of z (this is useful in building for loops)
2. numpy.zeros([x,y]) - creates an array of size (x,y) full of zeros

Figure 10 continued:

import matplotlib

1. matplotlib.pyplot.scatter(x,y, s=2) - plots the point/points or arrays x and y with x in the x-axis and y in the y-axis; s is the size of the dots

2. matplotlib.pyplot.plot(x, y, color=”z”) - plots the point/points or arrays x and y with x in the x-axis and y in the y-axis and connects the points with lines

3. matplotlib.pyplot.xlabel(‘x’) - changes the x-label on a plot to x

4. matplotlib.pyplot.ylabel(‘x’) - changes the y-label on a plot to x

import pandas

1. Data = pandas.read_csv(r”filename”) - extracts a csv file in the students computer in the location filename as long as you have given anaconda the correct permissions. (You can find the filename of a program in the file explorer of your computer)

2. Data = pd.read_excell(r”filename”) - extracts a csv file in the computer in the location filename as long as you have given anaconda the correct permissions. (One can find the filename of a program in the file explorer of the computer)

3. Data_entry = data.loc(“row_name”,”column_name”) - extracts an entry from a panda data file in row “row_name” and column “column_name

import scipy.linalg

1. Results = scipy.linalg.eig(array) - returns the eigenvalues and eigenvectors of array

2. Eigenvectors = results[1].real

3. Eigenvalues = results[0].real

To run a program, students will select the green button in the top right corner of the window. If the code runs in an unexpected way, the student can use the variable explorer tab in the top right corner of the window as a debugging tool. Debugging can be done by modifying the student's code and checking the values of the variables. If the student was expecting a plot and the plot does not show up in a new window, the student can check that they completed the set-up steps found in Figure 9 correctly.

Assignment 2: Using numpy and pandas with excel file data and graphing in matplotlib

Assignment: The student should extract data from an excel sheet on their computer, and plot the data using 2-dimensional numpy arrays, matplotlib, and for-loops.

The next assignment is useful in demonstrating coding proficiency with for-loops, plotting, and accessing data in arrays in Python before beginning larger projects. This assignment can be done many ways; however, the student should complete the task in a way that incorporates for-loops. Below in Figure 11, one can see an example of how this can be done.

In this assignment, a table of weight and horsepower of cars in comma separated values(csv) format is used. Any data set could be used but one can readily find this type of data related to cars online. Proctors can give the programming example to the students but emphasize that the for-loop should input data into a 2-dimensional array instead. By doing this, the student must make adjustments to the code that will test their understanding of programming and their Python ability.

As this is the first assignment that requires coding, the proctor should be available and willing to guide the student through the process and answer any questions that could potentially arise.

Figure 11: Example of Spyder graphing code

```
import pandas
import numpy as np
import matplotlib.pyplot as mp

Data = pd.read_csv(r"file_name.csv")
Weights = np.zeros([32,1])
Horsepower = np.zeros([32,1])

for i in np.arange(0,32):
    Weights[i] = Data.loc[i,"wt"]
    Horsepower[i] = Data.loc[i, "hp"]
mp.scatter(Weights, Horsepower, color = "blue", s=20)
mp.ylabel("horsepower")
mp.xlabel("weight")
```

Assignment 3: Building 2-dimensional numpy data sheets using the kinematic equations

Assignment: The student should build a 2-dimensional data sheet containing the time, x-axis, and y-axis coordinates of a baseball in motion. This assignment assumes that there is no air resistance. Students will use a 30 second time interval as the independent variable and allow the kinematic equations below to produce x and y coordinates from each of the time coordinates. They will then plot Y as a function of X.

*X = (initial_velocity_in_x-axis) * time + (initial_position_in_x-axis)*

*Y = (initial_velocity_in_y-axis) * time + (initial_position_in_y-axis) + (½) * (acceleration_of_gravity) * time * time*

The purpose of this assignment is to give a small amount of context to quantum mechanics by analyzing a classical physics scenario. Also, this assignment is a more challenging assignment than the last assignment, this will help give the student additional experience in Python before beginning additional coding. The looping utilized in this project is also not as straightforward as the last project which increases the difficulty. A segment of code that can satisfy the requirements of this assignment is given in the

"Resources for Proctors" section in Appendix B under "Baseball Solution". The solution is not located in the "Resources for Students" section so that students trying to complete this assignment on their own can attempt to complete the assignment without any other guidance. **As this assignment is more challenging the proctor should be available and able to assist the student when any difficulty that may arise.**

If a student has more than a few questions or challenges that arise, this may be a point at which to reconsider the prerequisite knowledge of the student. The following assignments will grow in difficulty and if these first assignments pose a significant challenge, the student should continue practicing basic programming before continuing or revisit the project when they have more experience in programming.

While giving the instructions of the assignment, the proctor should explain to the student that the equations used in this assignment fall in the realm of classical mechanics. The proctor should also discuss the deterministic nature of classical mechanics and the lack of uncertainty there is when you are given the necessary data. These two concepts are the major differences between the material introduced in classical physics courses such as Physics 1 and more in-depth quantum mechanics courses.

Week 2: The Harmonic Oscillator Project

As the name implies, the goal of week 2 is to complete a simple program that will find the energy of any energy-level for the quantum harmonic oscillator. The majority of this assignment will be spent covering basic concepts of quantum mechanics that would not be covered in the main project specifically so that the students develop a holistic understanding of the field of quantum mechanics.

Assignment 4: Watch a 5-minute YouTube video for an introduction to quantum mechanics

Students are to watch an online video to become familiar with principles of quantum mechanics. Students will view the video in the link in Figure 12 below. The video provides a visual resource to help students realize how different the field of quantum mechanics is from others that may be familiar. The purpose of students viewing the video is to help students begin to build questions so they can prepare for the upcoming material. If the video causes more confusion than helps, the assignment can be replaced or skipped entirely. Many introductory videos on quantum mechanics exist on YouTube and are great explanations of fundamental concepts.

Figure 12: Video option for fundamentals of quantum mechanics

https://www.youtube.com/watch?v=7u_UQG1La1o

While the purpose of the video should be to introduce fundamental concepts, the videos will likely leave the students with many questions and an incomplete understanding of quantum mechanics. The following lectures and assignments should answer many of their questions and give them hands-on experience in solving quantum mechanical problems.

Lecture 2: An overview of basic physics, chemistry, math, and quantum mechanics concepts for students

Learning objective: The students should review the following concepts: atomic structure, molecular structure, light as a wave, light carrying energy, and light absorption/electron excitation. The students should be comfortable with the concepts of vibrational excitations, molecular aggregation, Schrödinger's equations, the wave nature of matter, the significance of the wave function, the particle-in-a box solutions, operators, and eigenvalue problems.

The second lecture will have two sections which can be done sequentially or separately depending on which the proctor feels would be the most effective for the students. This first lecture section is a quick review and overview of knowledge that students should have already acquired from prerequisite courses and the knowledge will be required later in the project. The second lecture section contains a few fundamentals of quantum mechanics. The purpose of the second lecture section is to prepare the student for additional concepts that will be taught in future assignments, as well as cover necessary material the coding project itself will not cover on its own. These two sections can be taught together or separately, however both should be completed before continuing in the project.

Appendix B contains a link to a shared Google Drive folder which will allow access to all prepared presentations for the proctor as a tool to teach these concepts. The proctor should look through the slides and be familiar with the structure of the presentation before teaching. Each slide contains a subject with a bulleted list of properties or laws. As every student in the course is unique, it is the role of the proctor to determine how best to explain each concept, law, or property. Some teaching examples will be given in the slides, but creativity in teaching will be rewarded with a better understanding in the students. When going through the materials, the proctor should stop after each slide to ensure that the student is comfortable with each concept.

As this process could be difficult for the proctor, practice is recommended and a thorough understanding of the material is required before beginning the lecture. The lecture presentation was not created to develop a complex and complete understanding of quantum mechanics, however the projects that follow should allow the student to practice and apply their learning of these fundamental concepts. This background will also deepen and add complexity to their understanding of quantum mechanics. The goal of this project is to allow for fun and creativity on the part of the student. If this creativity is not occurring, the proctor should reassess the teaching process. However, this project is not intended to be easy and it will be a challenge on the part of the student to use logic and reasoning to develop their own understanding of the material as well.

Following this lecture, Assignment 5 should be given. The next section addresses how to implement this assignment. More information about Assignment 5 can be found on the "Lecture 2 Slides" in Appendix B.

Assignment 5: Creating a list of all possible wavefunctions for an aggregate

Assignment: Write a code which will have the inputs (aggregate size, max vibrational state) and the output a 2-dimensional matrix whose rows are all possible wavefunction states of an aggregate. The first column should designate excited particle number, and second column would designate vibrational excitation on that particle number.

In simpler terms, the assignment is for students to write a section of code that will create a list of all possible excited-states and ground-states in which a molecular aggregate could exist. This code will be used as written later in the project so it is of extreme importance that the student fully understands what this list is used for and the function of each part. The student should be given access to previous presentations so that they can review when necessary.

An atom can exist in a ground-state and many excited-states. This is also true for an aggregate which can exist in a ground-state or one of many excited-states. An aggregate can flip between states by absorbing energy so it is important to know how many different states in which an aggregate can exist. To limit this number considerably, three important assumptions are made. These three assumptions are listed below:

1. Only one particle in an aggregate will be excited electronically and/or vibrationally at one time.
2. The electronic nature of a particle is that it only exists as either excited or not excited. There will be no higher electronic excitation on a particle beyond the first excited level.
3. The vibrational nature of each particle will be that the particle only exists in a non-excited/ground state or one of a limited number of vibrationally excited states. The excited states will consist of integer levels from 1 to a V_{max}, or vibrational maximum, parameter which can be selected for the aggregate.

The students code should be flexible enough to take any reasonable maximum vibrational parameter and output a correct list that obeys all assumptions without listing any state more than once. All states should be stored in a 2-dimensional array. For more information on the notation used to store the states, see the “Main Project Description” section of this paper. Each row of the array should contain a vector that represents one of all possible states. In a physical sense it is not the wavefunction, but the vector that stands in place of the wavefunction as a way to classify each state. Figure 13 contains a few examples and common errors.

Figure 13: Wavefunction notation examples

- The row vector [1,2] represents an excited aggregate state with an electronic excitation on molecule/particle 1. That same molecule/particle is vibrationally excited to the second excited state represented by the 2.
- The row vector [3,1] represents an excited aggregate state with an electronic excitation on molecule/particle 3. That same molecule/particle is vibrationally excited to the first excited state.

A few common errors:

- The list should not have any more than 2 columns.
- The list should have the row [0,0] once and only once. This is the ground state.
- Any row that has a 0 in the first column should have a zero in the second. This is because you can only have vibrational excitation if there is already an electronic excitation according to the assumptions. (As examples, [0,1], [0,2], etc. are not allowed)
- There should not be two rows that are the same.
- This code can be created quite easily using a nested for-loop and if it becomes too complicated, the students should rethink and rework.

This assignment may not be self-explanatory for the student. If the student is confused at this stage, something that may help them is to have them draw diagrams of all the states that can possibly exist for a given V_{max} and particle number (begin with $V_{max} = 2$ for a 3-particle aggregate.) After this, have the students use formatting rules to assign a row vector to each state. With this list, students should hopefully see a pattern that is simple enough to code.

For students to check to ensure their output is correct, see the “Wavefunction list example code” section of Appendix B to compare.

Lecture 3: Harmonic oscillator in classical and quantum mechanics using Bra-Ket notation

Learning objectives: Students will understand the energy expression and application of the classical harmonic oscillator, the energy expression of the quantum harmonic oscillator and its use in the Schrödinger's equation, Bra-Ket notation, and the mathematical rules that apply. Students will understand the effects and application of raising and lowering operators, the process of filling in a Hamiltonian matrix, and the process of extracting data from the Hamiltonian matrix.

Lecture 3 and Assignment 6 on the quantum harmonic oscillator work together as the last phase of preparation before allowing students to complete their final project. This lecture and assignment gives students an understanding of molecular vibrations as well as the Hamiltonian matrix before applying these concepts to a larger project.

Lecture 3 will follow the format of Lecture 2 with a given presentation that is accessible in Appendix B. The proctor should proceed through the presentation, after some individual preparation, addressing each section of each slide in detail and giving insight where necessary. The proctor should also allow for questions as this lecture may be more challenging than the last. Again, the purpose of this lecture presentation is to familiarize the students with the concepts in the learning objectives that they can practice in the assignment. However, if the student has meaningful questions that impact their understanding, the proctor should answer the questions or help search for the answer if it is not apparent or known.

Assignment 6: Building the Hamiltonian matrix of the quantum harmonic oscillator

Assignment: Write code in Spyder which will build a correct Hamiltonian matrix and extract the energy of different energy levels (simple or more complex).

A worked example is in Appendix B under the name “Hamiltonian matrix of a quantum harmonic oscillator (Worked Example)”. Figures to help explain the project can be found in Appendix B under the name “Lecture 3 slides”.

Week 3: The One-Particle Approximation

Week 3 includes the discussion and instructions for completion of the final project. The goal of the final project is to enhance students' learning of quantum mechanics fundamentals through the completion of a more challenging assignment. The student will use the fundamental coding they have learned thus far, resources they've been given, and the Hamiltonian of the molecular aggregate/one-particle approximation to build the Hamiltonian matrix, identify energy levels, and plot the absorbance spectra of aggregates. The final project should help students to better understand the major concepts of quantum mechanics and may require a large amount of commitment from both the proctor and the student as many questions may need to be asked by the students.

Lecture 4: A review of what students have learned thus far and the introduction of a Hamiltonian for a one-particle approximation

Learning Outcomes: The students will be reminded of familiar concepts of quantum mechanics they have learned thus far. The students will be familiar with and ready to utilize the Hamiltonian operator for aggregates. The students will understand the significance of each term in the Hamiltonian operator for aggregates.

The purpose of Lecture 4 is to remind the students of all they have learned thus far in the project so that they may begin the next section well. Again, this lecture will follow the pattern of the previous lectures. A lecture presentation, under the name “Lecture 4 presentation”, can be found in Appendix B. A description of Assignment 7 can be found at the end of the “Lecture 4 presentation” that may assist proctors in the explanation of the assignment.

Assignment 7: Building the Hamiltonian matrix for a molecular aggregate

Assignment: Students will build a function which should have the inputs for aggregate size, electronic energy parameter, vibrational energy parameter, V_{max}, and Huang-Rhys factor. The code should output the correct Hamiltonian matrix.

This final assignment will assist the student to practice how to use quantum operators, wavefunctions, Hamiltonian matrices, and pattern recognition. The final assignment should also teach the student to build more intricate code in Python. This assignment will likely require the code to be analyzed and edited multiple times before a working code is complete. Because of this, the proctor should be ready to assist students when needed.

The proctor should be sure to allow students to practice the mathematical techniques to solve individual pieces of the Hamiltonian matrix so the student can identify each step in filling in the entire Hamiltonian matrix. Proctors should be sure to have the student try to fill in a small matrix on paper by hand so they can look for patterns that can be used in their code. This step should help the student to know what conditional statements the code should use to fill in the Hamiltonian matrix. If the student is having difficulty understanding the pattern presented, the proctor should demonstrate on paper what pattern students should look for. A description of the patterns can be found in "Main Project Description" (page 11). Also, a worked example can be found in the "Main Project example code" section of Appendix B for comparing results.

Lecture 5: Proctor adds in code to make absorbance plots; Discussion of results

Learning Objective: The student will understand the effects of changes in particle size, electronic energy, vibrational energy, vibrational max energy, and Huang-Rhys factor on the absorbance of molecular aggregates.

The proctor should obtain the working code from the student and add in a plotting function that will graph an absorbance plot for a Hamiltonian matrix. The function to be added, "Plotting Function", can be found in Appendix B. The function will determine the eigenvalues and eigenvectors of the matrix, generate a dipole moment matrix of the possible states, use matrix multiplication to solve for intensities, and plot this information in Spyder.

The proctor should meet with the student to return the working program and make a few different plots that will illustrate the effects of changing parameters. The parameters needed for these plots and examples of what they should look like can be found in the "Plotting Examples and Parameters" section of Appendix B. To have an absorbance in the visible light range, try these parameters ϵ = 10,000 cm^{-1}, ω= 1,000 cm^{-1}, and λ = 1 cm^{-1}.

Week 4: Additional Projects of higher difficulty

If needed, a fourth week could be used as an additional week for students to complete the assignment. This fourth week could also be used for students who have completed the project to rewrite the Plotting Function on their own. By doing so students will gain understanding of the dipole moment matrix and of the eigenfunctions and eigenvalues of the Hamiltonian matrix.

Conclusions:

As emerging fields such as quantum computing, nanotechnology, and solar energy conversion are ever expanding, the need for students understanding and researching in quantum mechanics grows. Undergraduate quantum mechanics courses provide the fundamental knowledge required of students in both physics and chemistry. Quantum mechanics is also one of the most difficult courses that these students will take in their undergraduate studies due to the mathematical challenges in quantum mechanics as well as the uniqueness of quantum mechanical concepts. To overcome this difficulty, many instructors have sought a hands-on approach to quantum mechanics.[2-5]

This micro course has been created for driven students with the following prerequisites:

- a high-school level understanding of physics, matrices, vectors, chemistry, and programming (for-loops, if/else statements, and arrays)
- a college-level understanding of derivative calculus

This microcourse may help prepare students for the challenges they will face in their undergraduate quantum mechanics course and help students to code more effectively. Additionally, this microcourse may also help students better choose which field of science they would like to pursue. This microcourse is unique in that it provides an out-of-classroom learning experience where students can proceed on their own time with the assistance of a proctor.

The microcourse utilizes Spyder, a scientific development environment, which can be learned quickly and can be used to solve more-complex quantum mechanical problems. Using this environment, students can build the Hamiltonian matrix of a quantum harmonic oscillator and of an approximate molecular aggregate. This microcourse allows students to use concepts they've learned and see the application of their understanding in a real-world problem.

This microcourse may help students conceptually grasp the fundamentals of quantum mechanics in a unique and hands-on project before trying to apply these concepts mathematically in a more rigorous course. As a result, the students may be better equipped to apply these concepts to other courses as well as outside of the classroom.

Appendix A: Resources for the Student

All lecture slides, data sets, and some worked examples of assignments can be found at the link in Figure 14 to a shared Google Drive folder under “Student Resources”.

Figure 14: Link for the shared Google Drive folder

https://drive.google.com/drive/folders/1YTsTEQZAHqsmP4A3Y7mlFaYuqLYMyarV?usp=sharing

Appendix B: Resources for the Proctor

All lecture slides, data sets, and worked assignments can be found at the link in Figure 14 under “Proctor Resources”.

Figure 14: Link for the shared Google Drive folder

https://drive.google.com/drive/folders/1YTsTEQZAHqsmP4A3Y7mlFaYuqLYMyarV?usp=sharing

References:

1. Morgan L. Sosa and Cathy Y. Wong , “Revealing the evolving mixture of molecular aggregates during organic film formation using simulations of in situ absorbance” , The Journal of Chemical Physics 153, 21 (2020)

2. G. L. Breneman and O. J. Parker, “Teaching quantum mechanics with Theorist”, Journal of Chemical Education 69, 1 (1992)

3. Yehudit Judy Dori, Vered Dangur, Shirly Avargil, and Uri Peskin, “Assessing Advanced High School and Undergraduate Students’ Thinking Skills: The Chemistry—From the Nanoscale to Microelectronics Module”, Journal of Chemical Education 91, 9 (2014)

4. Ross D. Hoehn, Nick Mack, and Sabre Kais, “Using Quantum Games To Teach Quantum Mechanics, Part 1”, Journal of Chemical Education 91, 3 (2014)

5. Erin Brown and Lisandro Hernández de la Peña, “A Simplified Pöschl–Teller Potential: An Instructive Exercise for Introductory Quantum Mechanics”, Journal of Chemical Education 95, 11 (2018)

Acknowledgements:

Special Thanks to my advisers:

Dr. Joe Bradshaw from Ouachita Baptist University Department of Chemistry,

Dr. Cathy Wong from University of Oregon Department of Chemistry and Biochemistry,

Morgan Sosa, and other members of the Wong lab at University of Oregon.

This material is based upon work supported by the National Science Foundation REU under Grant No. CHE-1659346