Quantum Particle-in-a-Sandbox: A video game that explores the time-dependent wave function for any arbitrary one-dimensional potential

Dhabih V. Chulhai*

Department of Chemistry, University of Indianapolis. 1400 East Hanna Ave., Indianapolis, IN 46227, USA.

E-mail: chulhaid@uindy.edu

Abstract

We present a tool, one that is both a stand-alone video game and a Python package, designed for students to explore a particle's wave function on one-dimensional potential surfaces. The tool relies on a basis set formalism and can therefore explore any one-dimensional potential surface imaginable. This tool also lets students interact with the wave function and is the first of its kind to explore concepts such as the superposition principle and wave function collapse; its time-dependent nature shows how any wave function evolves over time allowing for exploration of other concepts such as the uncertainty principle. The video game's ease of use makes these concepts accessible to anyone, without prior chemistry or programming background—though we expect that it would be most useful for undergraduate physical chemistry students and instructors.

Keywords

Upper-Division Undergraduate, Physical Chemistry, Quantum Chemistry, Computer-Based Learning, Humor/Puzzles/Games, Computational Chemistry

Introduction

Solving the Schrödinger equation and analyzing those solutions for one-dimensional systems is a core part of undergraduate physical chemistry courses, and the infamous particle-in-a-box case is almost always a student's first quantum mechanical system.[1–4] This and other one-dimensional systems help develop students' understanding of core concepts—such as how the quantum mechanical wave function relates to the underlying potential surface, the uncertainty principle, the superposition principle, and the like—before they move on to more complex two- and three-dimensional systems.

Only a handful of potentials have analytical solutions, and therefore, many educational tools have been developed that numerically solve the Schrödinger equation for various potentials.[5–15] Almost all of the tools that we have found focus on solutions to a subset of well-defined one-dimensional potentials, such as the particle-in-a-box for various types of boxes, the double-well, harmonic oscillator, Morse, Lennard-Jones, and others. As an exception, Beddard described a method in which the eigenfunctions of the particle-in-a-box potential may be used as a basis set to numerically solve the Schrödinger equation for any one-dimensional potential, though that work focused only on the anharmonic double-well potential.[13] In addition, none of the tools we found allow students to explore concepts such as superposition or wave function collapse.

In this report, we present a tool, called Quantum Particle-in-a-Sandbox (QPiaS),[16] designed for students to explore the time-dependent dynamics of one-dimensional wave functions on any imaginable one-dimensional potential surface. We achieve this by utilizing the basis set method described by Beddard,[13] coupled with a time-dependent visualization of the

wave function. This tool also lets students interact with the wave function, for example, by "collapsing" the particle to a position or momentum wave function described as a superposition of energy eigenstates, and observing how this new wave function evolves in time. QPiaS is written in Python3[17] and PyGame,[18] but its usage does not require any programming skills beyond what is required to install the software; we designed QPiaS to primarily be a video game. Video games as educational tools can reach larger audiences and have yielded improved learning outcomes in the sciences.[19] As such, our intention is that QPiaS may be used by anyone, not just physical chemistry students, to build intuition and understanding of quantum mechanical concepts. QPiaS may also be used as a Python package by instructors and students who would like to explore any other one-dimensional potential. QPiaS is a free and open source software that can be found at https://github.com/dchulhai/QPiaS.

Theoretical Framework and Software Requirements

Summary of the Theoretical Framework

A full description of the theoretical framework of QPiaS may be found in the Supporting Information. The "sandbox" nature of QPiaS lets students explore the evolution of the time-dependent, one-dimensional wave function $\Psi(x,t)$ on any time-independent potential surface $V(x)$. To allow for any potential surface, we coded QPiaS like any first principles computational chemistry software:

(1) We define a basis set $\{\chi_n\}$, comprised of the $n_{\max}$ lowest energy eigenfunctions from the one-dimensional particle-in-a-box model:[13]

$$\chi_n(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right); \quad n = 1, 2, \cdots, n_{\max} \tag{1}$$

where L is the length of the box and $n_{\max}$ sets the size of the basis set; in our code we use $n_{\max} = 40$.

(2) A potential energy matrix $\mathbf{V}$ is calculated from the potential surface $V(x)$ using numerical integration:

$$\mathbf{V}_{nm} = \int_0^L \chi_n^*(x) V(x) \chi_m(x) dx \tag{2}$$

From this, we generate and diagonalize the Hamiltonian $\mathbf{H}$ in the $\{\chi_n\}$ basis, which gives us a set of stationary state energy eigenfunctions $\phi_a(x,t)$:

$$|\phi_a(x,t)\rangle = \sum_n c_n |\chi_n(x)\rangle \cdot e^{-i\epsilon_a t} \tag{3}$$

where ϵ_a is the energy of the stationary state ϕ_a.

(3) Any time-dependent wave function $\Psi(x,t)$ is expressed as a superposition of the time-dependent energy eigenfunctions $\phi_a(x,t)$:

$$|\Psi(x,t)\rangle = \sum_a C_a |\phi_a(x,t)\rangle \tag{4}$$

(4) The real and imaginary parts of Ψ, as well as the square modulus $|\Psi|^2$, are continually displayed to the user. Properties of the wave function, such as the energy, energy expectation, position expectation, position uncertainty, momentum expectation, and momentum uncertainty are all continually analytically calculated and displayed to the user.

(5) A student can "collapse" the wave function into a single energy eigenfunction—by either absorbing or emitting a photon—or into a position (or momentum) wave function (the exact functional forms of these collapsed wave functions are given in the Supporting Information). These collapsed position/momentum wave functions are, in turn, expressed as a superposition of energy eigenfunctions and allowed to evolve in time.

Software Requirements and Installation

A complete installation guide can be found in the Supporting Information. QPiaS is written in Python (version 3.7+)[17] and runs on any Windows, macOS, or Linux/Unix platforms. It requires the PyGame engine (version 2.0+),[18] matplotlib (version 3.1+)[20] for generating graphs, and NumPy (version 1.17+)[21] and SciPy (version 1.3+)[22] for linear algebra operations. To run QPiaS as a Python package, we recommend the IPython[23] environment.

Software Features and Learning Outcomes

QPiaS Game Controls

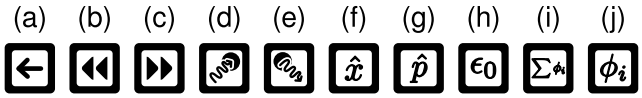

Figure 1: QPiaS game controls are displayed at the bottom of the game screen when applicable. Students may click on the icon or press the keyboard shortcut to manipulate the wave function: (a) Returns to the previous menu or screen [keyboard shortcut: escape key]; (b) Reduces the wave function's animation speed [left arrow]; (c) Increases the animation speed [right arrow]; (d) Causes the particle to absorb a photon and go to the next highest energy eigenstate [up arrow]; (e) Causes the particle to emit a photon and go to the next lowest energy eigenstate [down arrow]; (f) Finds the position of the particle by collapsing the wave function [X key]; (g) Finds the momentum of the particle [P key]; (h) Returns the particle to the ground state [G key]; (i) Toggles between showing the current wave function as a superposition of energy eigenstates or as a single wave function [S key]; (j) Toggles between showing all the calculated energy eigenstates or the current wave function [E key].

QPiaS may be used either as a stand-alone video game or as a Python package. In both cases, students are allowed to manipulate the particle's wave function by either clicking on a displayed icon or pressing the corresponding keyboard shortcut; these icons and a description of how they affect the wave function are shown in Figure 1.

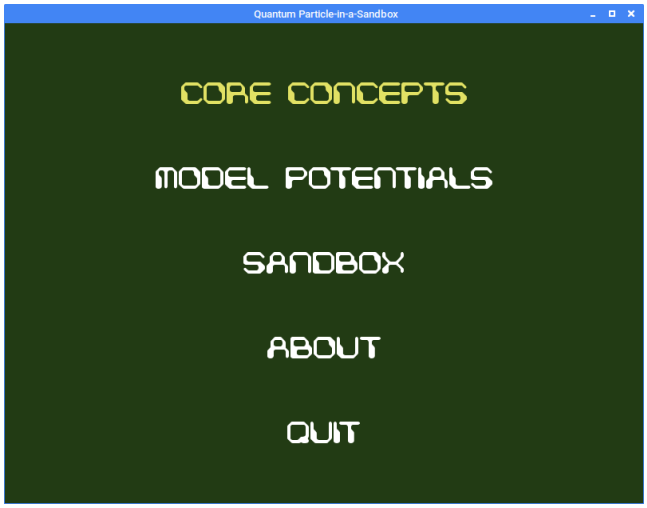

Figure 2: QPiaS main menu showing available game modes: Core Concepts; Model Potentials; and Sandbox.

As a Stand-Alone Video Game

As a stand-alone game, QPiaS presents a barrier-less entry for anyone to gain insights into core quantum mechanical concepts. Our primary objective, however, was to create a tool to aid undergraduate physical chemistry students to visualize the evolution of a particle's time-dependent wave function without any required coding or distracting equations. There are three game modes (see Figure 2 for the game's main menu) that were designed with specific learning outcomes in mind: "Core Concepts"; "Model Potentials"; and "Sandbox".

Core Concepts

The "Core Concepts" game mode contains five game levels (an example of a game level is shown in Figure 3) that help build intuition and understanding of quantum mechanical principles, while also teaching the basic controls for the game. Successfully completing a game level unlocks the subsequent level. The expected learning outcomes for each of the five game levels in the Core Concepts mode are (with QPiaS game control learning outcomes italicized):

1. Quanta:

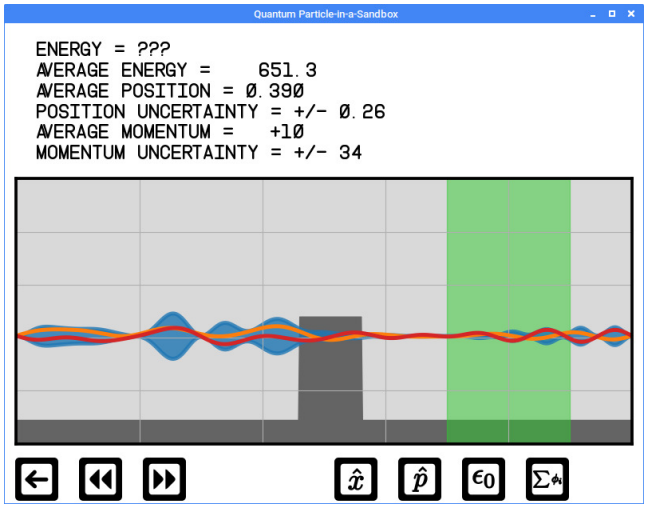

Figure 3: A snapshot of a Core Concept game level exploring quantum tunneling. The student explores how the height of the potential barrier (dark gray) affects the probability of finding the particle on the right side (green region). The real (red) and imaginary (orange) time-dependent wave functions and the square module of the wave function (shaded blue) are displayed. Properties of the wave function, such as its energy (if not a superposition), average energy, and average position, are continually calculated and shown. All units are arbitrary.

- that energy is discrete.
- that energy changes occur when electrons absorb or emit a photon.
- *how to speed up/slow down the animation of the game.*

2. Box length:

- that energy values and energy level differences change with box length.
- *how to reset the wave function to the ground state.*

3. Schrödinger's cat:

- that a position wave function identifies the current position of the particle.
- that a wave function can be written as a linear combination of other functions (superposition principle).
- that when you collapse a wave function, it randomly "chooses" (depending on the probabilities) one of the functions to collapse into.

- *how to collapse into a position wave function.*
- *how to show the wave function as a superposition of energy eigenfunctions.*

4. Uncertainty principle:
    - that a momentum wave function identifies the current direction and speed of the particle.
    - that you cannot know, with absolute certainty, both the position and momentum of a particle at the same time.
    - *how to collapse into a momentum wave function.*
5. Tunneling:
    - that wave functions have some probability of being inside barriers with low potentials.
    - that electrons have a small (non-zero) probability of tunneling across barriers.
    - that this probability decreases with increasing barrier height or width.

Model Potentials

In the "Model Potentials" game mode, users can explore how the quantum wave function evolves on the following common potential surfaces:

- Particle-in-a-box (in an infinite potential well)
- Harmonic oscillator potential
- Morse potential
- One-dimensional Coulombic potential
- Particle-in-a-box with a finite potential barrier

If students wish to explore other potentials not included here, they may use the Sandbox mode to draw their potentials or use QPiaS as a Python package; both of these are described below.

Sandbox Mode

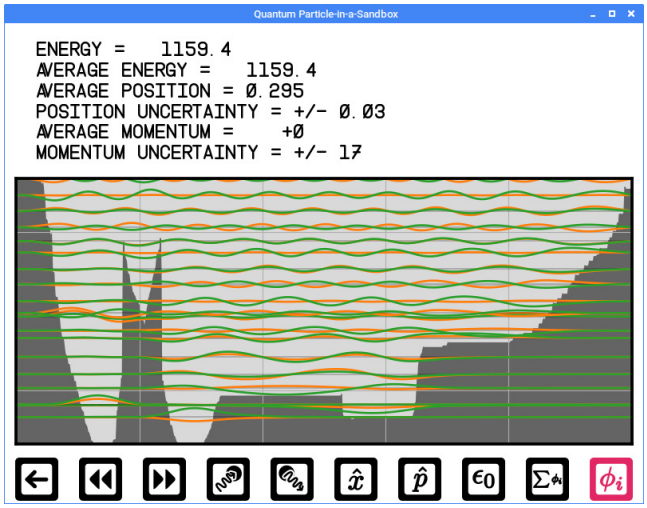

Figure 4: Eigenvectors for some arbitrarily drawn potential surface (shaded gray) generated using the Sandbox game mode. Both the real (green) and imaginary (orange) parts of the time-dependent energy eigenfunctions are displayed.

The "Sandbox" game mode allows the user to draw any one-dimensional potential and explore the evolution of the particle's wave function on that potential; the student is limited only by their imagination and their ability to draw with a mouse. An example of the types of potential surfaces that could be explored is shown in Figure 4.

As a Python Package

The open source Python package also includes modular codes that allow instructors and students to explore the quantum mechanical wave function using their own one-dimensional potential or even create entire game levels. An example of how to use the QPiaS Python package to simulate a double square-well potential is shown in Figure 5. Other usages of QPiaS as a Python package can be found in the Supporting Information.

a)

```
import numpy as np
import qpias

# create the 1D 'box'
x = np.linspace(0,10,1000)

# create double well potential
V = np.zeros_like(x)
V[np.where((x>2) & (x<4))] = -2500
V[np.where((x>6) & (x<8))] = -3000

# initialize the game
game = qpias.Game()

# create the stage with the potential
stage = qpias.Stage(game, V)
stage.run()
```

b)

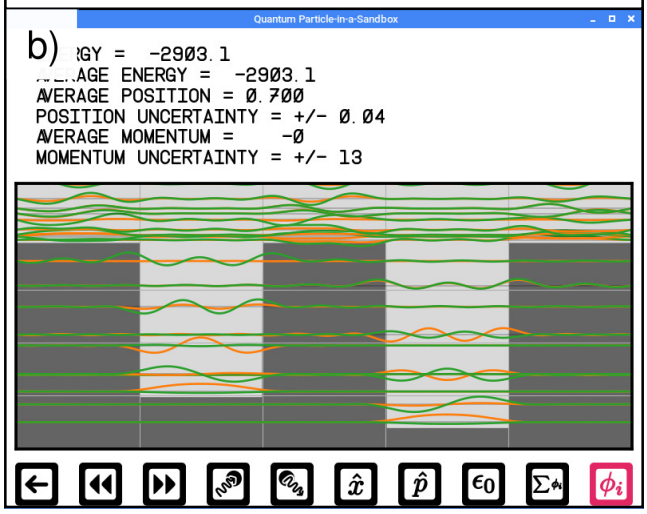

Figure 5: Using QPiaS as a Python package to simulate wave functions. (a) The Python code used to create a double square-well potential. (b) The resulting energy eigenfunctions.

Conclusion

We presented a learning tool, QPiaS, that solves the Schrödinger equation for any imaginable one-dimensional potential surface. This tool also allows the student to interact with the particle's wave function, and is the first tool—as far as we are aware—that allows students to explore concepts like wave function collapse and the superposition principle. Since it is written as a video game first and a Python package second, QPiaS may be used by anyone to gain insights into key quantum mechanical concepts, regardless of chemistry or programming background—though its primary audience is undergraduate physical chemistry students and instructors.

Supporting Information Available

Details of the theoretical framework; Installation instructions for Windows, MacOS, and Linux/Unix systems; How to use the QPiaS Python package.

Quantum Particle-in-a-Sandbox source code: https://github.com/dchulhai/QPiaS.

Acknowledgement

The author thanks Jasmine E. Chulhai for testing the game and proofreading this report, and his physical chemistry students for their feedback and encouragement.

References

(1) McQuarrie, D. A.; Simon, J. D. *Physical Chemistry: A Molecular Approach*, 1st ed.; University Science Books, 1997.

(2) Engel, T.; Reid, P. *Physical Chemistry (3rd Edition)*, 3rd ed.; Pearson, 2012.

(3) Ball, D. W. *Physical Chemistry*, 2nd ed.; Cengage Learning, 2014.

(4) Atkins, P.; de Paula, J.; Keeler, J. *Atkins' Physical Chemistry 11e*, 11th ed.; Oxford University Press, 2018.

(5) Tellinghuisen, J. Accurate numerical solutions of the one-dimensional Schrödinger equation. *J. Chem. Educ.* **1989**, *66*, 51, DOI: 10.1021/ed066p51.

(6) Rioux, F. Quantum mechanics using Mathcad 3.0. *J. Chem. Educ.* **1992**, *69*, A240, DOI: 10.1021/ed069pA240.

(7) Hansen, J. C. Schröedinger.m: A Mathematica Package for Solving the Time-Independent Schrödinger Equation. *J. Chem. Educ.* **1996**, *73*, 924, DOI: 10.1021/ed073p924.1.

(8) Lang, P. L.; Towns, M. H. Visualization of Wavefunctions using Mathematica. *J. Chem. Educ.* **1998**, *75*, 506, DOI: 10.1021/ed075p506.

(9) Cedillo, A. Quantum Mechanical Tunneling through Barriers: A Spreadsheet Approach. *J. Chem. Educ.* **2000**, *77*, 528, DOI: 10.1021/ed077p528.

(10) Francis, T. A.; Miles, D. G. A Graphical Approach to the Angular Momentum Schrödinger Equation. *J. Chem. Educ.* **2001**, *78*, 405, DOI: 10.1021/ed078p405.

(11) Johnson, J. L. Visualization of Wavefunctions of the Ionized Hydrogen Molecule. *J. Chem. Educ.* **2004**, *81*, 1535, DOI: 10.1021/ed081p1535.1.

(12) Poiares, J. P. M.; Rodrigues, S. P. J.; Marques, J. M. C. A Quantum Mechanics Toolkit: Useful Internet Toolkit to Teach Fundamental Concepts of Quantum Mechanics. *J. Chem. Educ.* **2008**, *85*, 591, DOI: 10.1021/ed085p591.

(13) Beddard, G. S. Solution of the Schrödinger Equation for One-Dimensional Anharmonic Potentials: An Undergraduate Computational Experiment. *J. Chem. Educ.* **2011**, *88*, 929–931, DOI: 10.1021/ed1000137.

(14) Ge, Y.; Rittenhouse, R. C.; Buchanan, J. C.; Livingston, B. Using a Spreadsheet To Solve the Schrödinger Equations for the Energies of the Ground Electronic State and the Two Lowest Excited States of H2. *J. Chem. Educ.* **2014**, *91*, 853–859, DOI: 10.1021/ed400693p.

(15) Srnec, M. N.; Upadhyay, S.; Madura, J. D. A Python Program for Solving Schrödinger's Equation in Undergraduate Physical Chemistry. *J. Chem. Educ.* **2017**, *94*, 813–815, DOI: 10.1021/acs.jchemed.7b00003.

(16) Chulhai, D. V. QPiaS: Quantum Particle-in-a-Sandbox. https://github.com/dchulhai/QPiaS, 2021; Accessed: 2022-7-6.

(17) Python Software Foundation. Python Language Reference, version 3.7. https://www.python.org/, Accessed: 2022-7-6.

(18) PyGame Development team, PyGame - Python Game Development. https://www.pygame.org, 2011; Accessed: 2022-7-6.

(19) Mayo, M. J. Video games: a route to large-scale STEM education? *Science* **2009**, *323*, 79–82, DOI: 10.1126/science.1166900.

(20) Hunter, J. D. Matplotlib: A 2D Graphics Environment. *Comput. Sci. Eng.* **2007**, *9*, 90–95, DOI: 10.1109/MCSE.2007.55.

(21) Harris, C. R. et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362, DOI: 10.1038/s41586-020-2649-2.

(22) Jones, E.; Oliphant, T.; Peterson, P.; Others, SciPy: Open source scientific tools for Python. 2001.

(23) Perez, F.; Granger, B. E. IPython: A System for Interactive Scientific Computing. *Comput. Sci. Eng.* **2007**, *9*, 21–29, DOI: 10.1109/MCSE.2007.53.

Graphical TOC Entry