Journal of Cheminformatics

EDUCATIONAL ARTICLE

Open Access

Advantages of two quantum programming platforms in quantum computing and quantum chemistry

Pei-Hua Wang[1,2], Wei-Yeh Wu[3], Che-Yu Lee[4], Jia-Cheng Hong[3] and Yufeng Jane Tseng[3,4,5,6*]

Abstract

Quantum computing is at the forefront of technological advancement and has the potential to revolutionize various fields, including quantum chemistry. Choosing an appropriate quantum programming language becomes critical as quantum education and research increase. In this paper, we comprehensively compare two leading quantum programming languages, Qiskit and PennyLane, focusing on their suitability for teaching and research. We delve into their basic and advanced usage, examine their learning curves, and evaluate their capabilities in quantum computing experiments. We also demonstrate using a quantum programming language to build a half adder and a machine learning model. Our study reveals that each language has distinct advantages. While PennyLane excels in research applications due to its flexibility to adjust parameters in detail and access multiple sources of real quantum devices, Qiskit stands out in education because of its web-based graphical user interface and smaller code size. The codes and the dataset used in the studies are available at https://github.com/wangpeihua1231/quantum-programming-platform.

Scientific contribution

This article reviews key applications of quantum computing within quantum chemistry, including ground state energy calculations, quantum dynamics, and Hamiltonian learning. We present a comprehensive comparison of the PennyLane and Qiskit platforms, examining their respective advantages and limitations to inform their suitability for both educational and research contexts in the rapidly advancing field of quantum computing. Additionally, we demonstrate foundational quantum circuits and introduce quantum machine learning models, encouraging readers to explore interdisciplinary applications that bridge quantum computing with broader scientific inquiry.

Keywords Quantum computing, Quantum programming language, Qiskit, PennyLane, Quantum chemistry

*Correspondence:
Yufeng Jane Tseng
yjtseng@csie.ntu.edu.tw
Full list of author information is available at the end of the article

Introduction

Quantum computing is a cutting-edge field in computer science and physics that leverages the principles of quantum mechanics to perform computations at a scale and speed that traditional classical computers cannot match. It also can potentially solve complex problems intractable for classical computing methods [1]. At the heart of quantum computing lies the quantum bit or qubit, the fundamental unit of quantum information. Unlike classical bits, which can only have states of 0 or 1, a qubit can simultaneously have a state that is superposed of 0 and 1 [2]. This property enables quantum computers to explore many possibilities in parallel [3], making them exceptionally well-suited for solving complex problems that are infeasible for classical computers [4]. The quantum advantage refers to the exceptional capabilities of quantum computers in solving problems that are practically unsolvable with classical computers [5]. Quantum computers have significant long-term potential for widespread applications from logistics optimization to materials science, chemistry, and information security.

Quantum chemistry plays a pivotal role in various scientific domains, particularly in investigating and manipulating molecular systems at the atomic and electronic levels. Its application extends to studying biological macromolecular systems; it can model intricate phenomena such as enzyme catalysis [6], gene duplication, mutation, and drug-receptor interactions. Understanding the molecular and electronic structure of biological molecules at the quantum level is indispensable for designing enzymes, modifying enzyme structures, and synthesizing artificial enzymes. Moreover, quantum chemistry elucidates the mysteries of heredity and mutation [7], offering opportunities to regulate gene replication and mutation and obtain potential benefits for humanity. Additionally, the detailed examination of drug-receptor interactions through quantum chemical methods facilitates the design of new drugs with heightened efficacy and reduced toxicity [8], highlighting quantum chemistry's broad and impactful applications in studying and manipulating biological phenomena.

Researchers and developers use quantum programming languages and frameworks to harness the power of quantum computers. Examples include PennyLane [9], Google's Cirq [10], IBM's Qiskit [11], NVIDIA's cuQuantum [12], and Microsoft's Q# [13] among others. These languages enable the design and simulation of quantum circuits, allowing researchers to develop quantum algorithms and experiment with quantum hardware. They are pivotal in advancing quantum computing research and practical applications.

IBM's Qiskit [11] is an open-source quantum computing development framework that provides tools and libraries for working with quantum computers. It was developed by IBM and is designed to make it easier for researchers, developers, and enthusiasts to explore and experiment with quantum computing. Qiskit enables users to create, simulate, and run quantum programs on IBM's cloud-based quantum computers. Users can also set up their local simulators for testing and development. Qiskit provides libraries and tools for quantum information science, including operations for quantum entanglement, quantum teleportation, and quantum error correction.

PennyLane [9] is an open-source quantum machine learning and computing software library. It was developed by Xanadu [9], a quantum technology company. PennyLane is unique because it focuses on the interface between quantum computing devices and machine learning frameworks, particularly popular libraries such as TensorFlow [14] and PyTorch [15]. There are several key features of PennyLane noted by Xanadu [9]: write-once, run-anywhere code; simulators and hardware in one place; an accompanying global research and coding community; built-in automatic differentiation of quantum circuits; the ability to perform machine learning on quantum hardware; and an "everything included" nature.

Our paper aims to comprehensively analyze and compare two prominent quantum computing and quantum machine learning frameworks, Qiskit and PennyLane, specifically focusing on their roles in education and research. We aim to provide valuable insights for educators, researchers, and quantum enthusiasts by evaluating the strengths and weaknesses of both frameworks in terms of accessibility, usability, educational resources, community support, and research capabilities. By examining their respective contributions to quantum education and their suitability for cutting-edge research applications, we aim to assist readers in making informed decisions about which framework aligns best with their specific needs and objectives in the rapidly evolving field of quantum technology.

Applications of quantum computing in quantum chemistry

Quantum simulation

Quantum computing has a notable advantage in simulating the microscopic physical characteristics involved in quantum mechanics [16], a task that grows in complexity exponentially with system size in traditional computing systems. Simulating quantum systems [17], directly manipulating physical quantum devices in the laboratory to emulate other quantum systems, can conduct accurate modeling of molecular structures and chemical reactions in quantum chemistry, materials science, and drug discovery. The problem size in this context is related to the simulated quantum system's complexity, allowing for the computation of quantum dynamics and adaptability to large-scale systems.

Ground state energy calculation

The lowest energy state, also known as the ground state, is the state of a system with the lowest possible energy. Finding the ground state of a molecule involves solving the electronic Schrödinger equation. From the electronic ground state, we can determine the arrangement of the electrons in the molecular orbital that minimizes the system's overall energy. Various methods, such as the Hartree–Fock [18], density functional theory (DFT) [19], and post-Hartree–Fock methods [20], have been employed for calculating electronic ground-state structures.

The goal of understanding the lowest energy states of molecular systems has driven innovative approaches in quantum chemistry simulations. A recent study introduced quantum HF/DFT-embedding algorithms as a promising avenue for scaling up electronic structure calculations in complex molecular systems [21]. An effective Hamiltonian construction incorporating mean field potentials to describe inactive electrons in an active space was outlined. Quantum HF/DFT-embedding algorithms can provide significant energy corrections to the HF and DFT calculations used as a reference, particularly in the strongly correlated limit and for systems the size of the oxirane molecule.

Quantum dynamics

Quantum dynamics [22] is the quantum counterpart of classical dynamics, which describes particles' motion, energy, and momentum changes according to classical mechanics. In quantum dynamics, particles such as electrons and photons exhibit particle-like and wave-like behaviors. The time-dependent Schrödinger equation illustrates how the quantum state of a system changes over time, and its solution provides insights into the evolution of the quantum state. Quantum dynamics is essential for understanding various phenomena in quantum mechanics, including the behavior of electrons in atoms, the formation and breaking of chemical bonds, and the properties of materials at the quantum level.

Recent advancements in quantum computational algorithms, particularly in adiabatic and nonadiabatic quantum dynamics [23], have addressed the longstanding challenge of simulating molecular dynamics within a comprehensive quantum framework. These innovations aim to address the exponential scaling inherent in the simulation of many-body quantum dynamics, providing valuable insights into the potential of the quantum advantage for applications in quantum chemistry, quantum computing, and quantum information science.

Hamiltonian learning

Hamiltonian learning provides a validation framework for ensuring the accuracy of quantum system descriptions [24]. In an isolated n-qubit system, the system's dynamics are typically governed by a known Hamiltonian [25]. However, when experimental observations exhibit significant deviations from theoretical predictions, this may indicate the presence of unaccounted interactions, such as an unidentified qubit coupling to the system or an unknown energy exchange pathway between the system and its environment. These discrepancies suggest potential inaccuracies in the assumed Hamiltonian, necessitating a systematic approach for validation and refinement. By leveraging expectation values, Hamiltonian learning enables a direct comparison between a theoretically defined Hamiltonian and the one inferred from experimental data, facilitating the identification of errors arising from (1) incorrect parameterization in the defined Hamiltonian or (2) operational imperfections in quantum devices.

Unlike conventional methods that rely on predefined Hamiltonians to predict quantum behavior [25], Hamiltonian learning operates in reverse—deducing the underlying Hamiltonian directly from observational data [26]. One approach to reconstructing the Hamiltonian involves expressing it in the Pauli basis and utilizing measurements on random states to generate a time-series dataset [26]. This approach exemplifies how Hamiltonian learning can uncover deviations from expected system dynamics, making it a crucial tool for validating and optimizing quantum models. By systematically refining theoretical descriptions and identifying errors in quantum devices, Hamiltonian learning contributes to the broader advancement of quantum computing and quantum information science.

Quantum machine learning and optimization

Given datasets and the cost function, quantum computers show potential in solving optimization problems by minimizing the predefined cost functions. Quantum algorithms such as the quantum approximate optimization algorithm (QAOA) [27] aim to find optimal solutions for such problems. Additionally, if the target is to predict the target value or label of a dataset, quantum machine learning algorithms, such as the quantum support vector machine [28], which achieves exponential [29] and quadratic speed-up through Grover's search algorithm [30], contribute to solving linear algebra problems more efficiently. Notably, various quantum algorithms, such as the variational quantum eigensolver (VQE) [31] and

Table 1 Summary of comparisons between Qiskit and PennyLane. Qiskit is more concise when using quantum gates and circuits, whereas PennyLane offers greater clarity

Comparison Point	Qiskit	PennyLane
Quantum Circuits	Treats circuits as objects combined using "compose."	Uses "wires" for qubits, represented by functions
Gate Usage	More concise, e.g., "h" for the Hadamard gate	Clearer naming, e.g., Hadamard() for the Hadamard gate
Measurement Methods	Specifies the output during creation, offering more flexibility in design	Returns measurements via functions, such as probabilities and expected values
Environment setting	Often encounter version and dependency problems	Smoother with a few basic libraries
Tutorials	It may have a steeper learning curve for beginners	The gentler learning curve for those with some quantum knowledge
Visualization	Integrates with Jupyter, a beginner-friendly platform	Focused on quantum machine learning, not as intuitive as Qiskit
Real Devices	IBM superconducting computers open to users for free with a beginner-friendly GUI	Utilizes quantum computers from several providers and the resources of users
Case Study: Half Adder	More intuitive result representation	Uses PauliZ prediction, requires familiarity with PauliZ
Case Study: Machine Learning	Built-in functions, easy to implement	Detailed user-defined functions suitable for research

Concerning the use of measurement and simulators, PennyLane emphasizes computation speed and precision. Regarding visualization, Qiskit easily enables the visualization of the current circuit via Jupyter. In contrast, PennyLane often requires assistance from Qiskit for visualization purposes

quantum adiabatic algorithm (QAA) [32], play roles in addressing different problem domains, contributing to the diverse applications of quantum computing.

Nevertheless, quantum computing faces numerous challenges, particularly in algorithm design, which necessitates fundamentally different approaches from those used in classical computing. The advantages of current quantum computers are mostly demonstrated through specific, carefully chosen problems due to the limited error correction capabilities of quantum computers [33], which correct errors by using additional qubits among the limited number available.

Methods

The code for this study was written in Python using Qiskit (version 1.4.1) and PennyLane (version 0.41.0). The usage examples, half adder and machine learning were adapted from their official tutorials.

To compare the two quantum programming languages quantitatively, we conducted two types of analysis:

1. Line count analysis – measuring the total number of lines required to implement each example.
2. Dependency analysis – counting the number of required libraries.

Both analyses were performed on the two usage examples implemented in both languages, resulting in eight data points ($2\times2\times2=8$) for comparison and discussion. To ensure a fair comparison, we excluded non-essential code related to dataset analysis and visualization, retaining only the core implementation necessary to execute the experiments.

Table 2 Quantitative analysis between usage examples in Qiskit and PennyLane. The line count excludes empty lines and comments

Usage example	Programming language	Line count	Number of imported library
Half adder	Qiskit	14	1
	PennyLane	18 (qiskit format) 12 (expected value)	1
Machine learning	Qiskit	45	6
	PennyLane	48	5

If multiple modules in the same library are separately imported in different lines, we only count as 1 library

Quantum programming languages: Qiskit and PennyLane

We analyze the difference between the two quantum programming languages in several respects: basic usage, visualization, implementation of tasks, available simulators and real devices, local environment construction, and official tutorials. Table 1 summarizes the contents that we introduce in this section. Table 2 lists the line counts and the number of libraries required in the two usage examples described in the following sections.

PennyLane is user-friendly in local environment construction

Building a local environment for basic usage of PennyLane is simple, as we can follow the official instructions and go through tutorials without problems. In contrast, beginners often face compatibility issues when learning Qiskit locally due to the multiple external libraries packaged with Qiskit. Sometimes, the Python

$$\frac{\sqrt{2}}{2}|00\rangle + \frac{\sqrt{2}}{2}|11\rangle$$

Fig. 1 Visualization of a statevector in Qiskit with the LaTeX parameter. The compact math formulation is friendlier to beginners

version must be changed to align with the specific requirements of different packages. Some of these packages might require a higher version of Python, whereas others are compatible only with lower versions. Balancing these requirements often necessitates a middle-ground version of Python that can cater to both needs. This can be a significant difficulty for programming novices, as switching between different Python versions to meet varying compatibility needs may require using specific Python version management tools or implementing virtual environments. The complexity of managing these different versions adds a layer of difficulty to the learning process, especially for those just starting in the field.

PennyLane's tutorial has a gentler learning curve for beginners

The Qiskit Textbook imparts foundational knowledge of quantum computation and programming. This comprehensive introduction covers the theoretical aspects of quantum computation and its practical implementation in Qiskit. For complete beginners, the learning curve might initially be steep. However, with detailed explanations and examples, progress may become smoother once the basic concepts become familiar. In contrast, PennyLane's Codebook layout shows coding challenges on the left and textbook content on the right. After grasping the concepts from the right side, learners can address the challenges on the left. The learning curve might be gentler in this case for those with some quantum computing knowledge. This platform offers a direct introduction if learners are primarily interested in quantum machine learning.

Qiskit has better visualization and a graphical user interface

When used in conjunction with Jupyter [34], Qiskit provides visualization tools that are particularly beginner-friendly. With just a single line of code, users can display features such as quantum circuit diagrams, statevectors (Fig. 1**,** Fig. 2a), and histograms for experimental results, allowing quantum computing newcomers to focus on learning without the distractions of visualization complexities. PennyLane, however, is more focused on quantum machine learning. While the statevector of a circuit in PennyLane can be displayed with a single line of code (Fig. 2b), its visualization capabilities are generally not as straightforward as those of Qiskit. For instance, visualizing a statevector in PennyLane requires additional efforts to use modules such as Matplotlib, unlike Qiskit, which offers a more intuitive visualization with its LaTeX parameter.

IBM Quantum Composer provides a web-based GUI (Fig. 3) for building quantum circuits by dragging gates to the lines representing the quantum circuit. Tutorials are shown on the left, and the current quantum states are visualized at the bottom. In the tutorial for building a Bell state, the visualization updates dynamically with any circuit change. On the right side is the coding area, which shows the OpenQASM or read-only Qiskit code. The circuit can be run on a real quantum device by clicking the "Setup and run" button in the upper-right corner and selecting available quantum devices. For newcomers unfamiliar with programming languages, IBM's Quantum Composer is a user-friendly learning platform for understanding how gates and codes work.

PennyLane accepts various real devices from a broad range of quantum providers

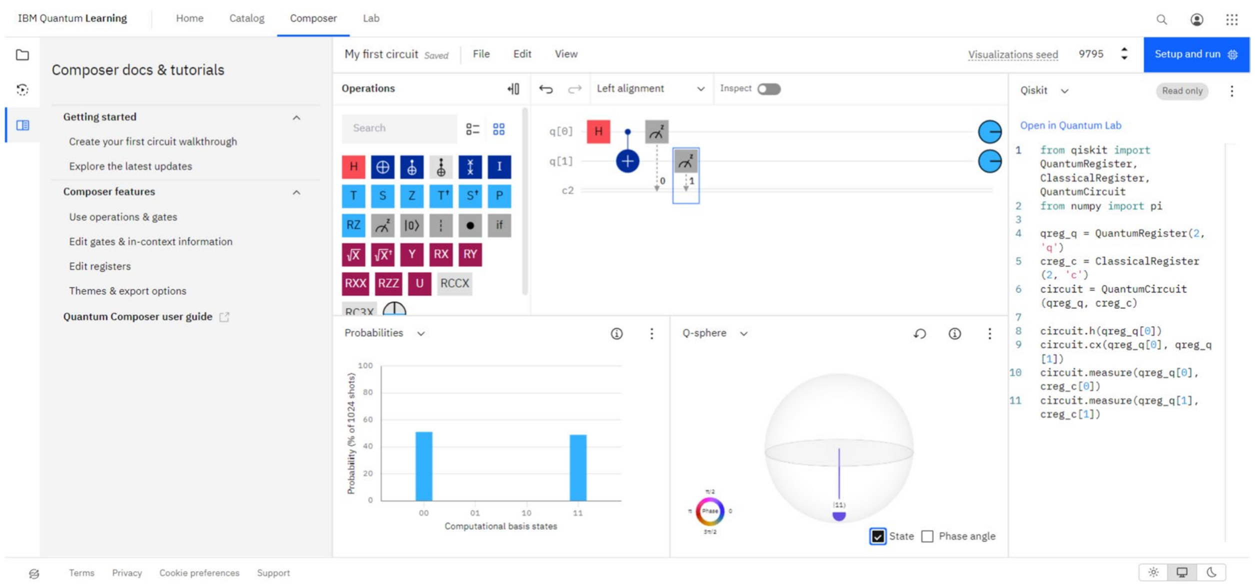
To run Qiskit on real devices, IBM provides several superconducting unitary quantum computers to registered users with a free open plan. One 127-qubit system (ibm_brisbane) and three 7-qubit systems (ibm_perth, ibm_lagos, and ibm_nairobi) are available to free users as of the date of Oct 15, 2023. Free users have up to 10 min of system execution time per month.

Running PennyLane on real devices requires plugins such as Qiskit to gain access to external quantum devices. The PennyLane development team supports popular

a)
```
Statevector([0.70710678+0.j, 0.        +0.j, 0.        +0.j,
             0.70710678+0.j],
            dims=(2, 2))
```

b)
```
array([0.70710678+0.j, 0.        +0.j, 0.        +0.j, 0.70710678+0.j])
```

Fig. 2 Visualization of a statevector with complex numbers. The 2-qubit statevector is shown with four complex numbers in a set. **a** Visualization using Qiskit without parameter. **b** Visualization using PennyLane



```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(2, 'q')
creg_c = ClassicalRegister(2, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.h(qreg_q[0])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.measure(qreg_q[0], creg_c[0])
circuit.measure(qreg_q[1], creg_c[1])
```

Fig. 3 Screenshot of IBM Quantum Composer after completing the tutorial "Create your first circuit walkthrough." Building a quantum circuit with the online graphical user interface is intuitive. The code area on the right side reflects the contents of the quantum circuit and is updated with every change

libraries, including IBM's Qiskit, Amazon Bracket, Google's Cirq, Microsoft QDK, Honeywell, IonQ, and Rigetti Forest. It is even possible to create quantum applications across different quantum platforms via PennyLane. The real quantum resources depend on the user when using PennyLane.

Qiskit is better for education to beginners, and PennyLane has the potential for teaching advanced quantum computing concepts

In Qiskit, while setting up a local environment and understanding the theoretical explanations in tutorials can be challenging, beginners may find it easier to construct quantum circuits using the web-based IBM Quantum Composer, allowing them to intuitively drag and drop gates. Additionally, Qiskit provides high-level functions that simplify circuit implementation and enable execution on real quantum devices through its open-access plan. In contrast, PennyLane explicitly defines nearly every step in the workflow, allowing users to grasp fundamental concepts and understand the detailed operations performed at each computation stage, making it well-suited for research applications.

Usage examples: half adder

This case study examines experiments taught in the Qiskit textbook introduction course and contrasts its implementations in Qiskit and PennyLane.

The introduction course in the Qiskit textbook introduces the "half adder" operation, which adds two single bits and measures the result. This process educates learners on the NOT, CNOT, and Toffoli gates. After designing the quantum circuit and specifying the measurement in Qiskit, a simulator is needed to replicate the behavior of the quantum circuit on a real device. If learners aim to implement the half adder locally, the first obstacle is that simulator-related modules cannot be imported. Since the course does not address this issue, students must independently identify and rectify the problem. Only after installing the correct modules can the operation run smoothly.

In contrast, implementing the half adder in PennyLane allows the quantum function to return the expected value of the PauliZ prediction. This method differs from platforms such as Qiskit, which offer a more intuitive result presentation. Specifically, for an input of (1,1) with the expected output '10', separate measurements on each qubit of the half adder results in the states $|1\rangle$ and $|0\rangle$. The PauliZ measurement represents these states as -1 and 1, respectively. This outcome is attributed to the nature of the PauliZ gate: it keeps the base state $|0\rangle$ unchanged and maps the state $|1\rangle$ to $-|1\rangle$. Therefore, while this measurement method offers precision, understanding the results requires some familiarity with PauliZ, unlike the more straightforward interpretation in Qiskit.

Usage examples: machine learning

To analyze the advantages and limitations of Qiskit and PennyLane in machine learning, we referred to the tutorials [35, 36] and built two kernel-based quantum

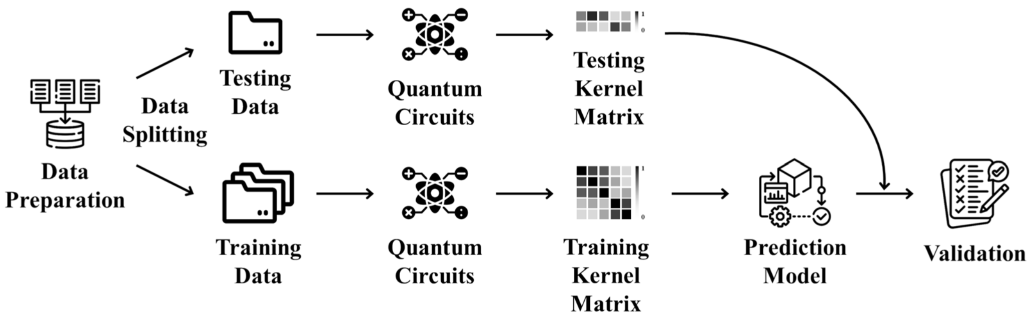
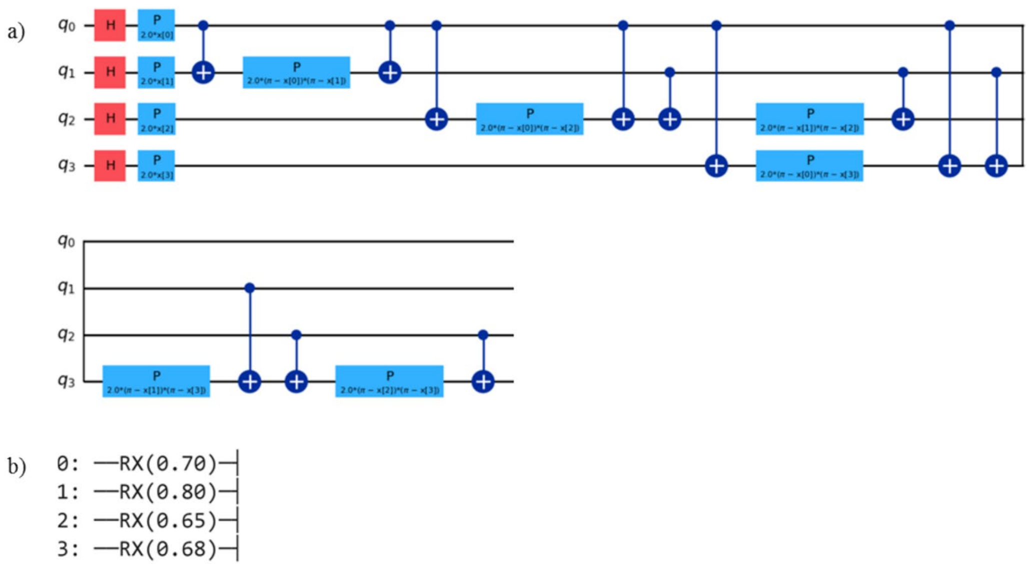


Fig. 4 Workflow for building a quantum prediction model. Quantum circuits are reversible. Applying the inverse of an executed quantum circuit restores the initial quantum state. For each pair of molecules in the training data, one molecule's features are fed into the quantum circuit, while the other molecule's features reverse the quantum circuit. The probability of measuring the initial states represents the similarity between the two molecules. A classical optimizer then uses the resulting kernel matrix to construct the prediction model



```
0: ──RX(0.70)─┤
1: ──RX(0.80)─┤
2: ──RX(0.65)─┤
3: ──RX(0.68)─┤
```

Fig. 5 Visualization of quantum circuits in feature mapping step. **a** Four-qubit ZZ feature map in Qiskit. The x variables stand for the input features of the data. **b** The visualization of four-qubit AngleEmbedding circuits defined in PennyLane. The angles are derived from the first data record used in the usage example

support vector machine (QSVM). The AqSolDB dataset, a widely used curated solubility dataset from Sorkun's study [37], was used in this study. The workflows used to build a quantum regressor on real data are similar in both tutorials, including data preparation, quantum circuit construction, kernel matrix generation, and validation (Fig. 4).

The data preparation process involves scaling the features to a range between 0 and 1, then splitting the dataset into training and testing sets. The AqSolDB dataset contains 9,983 molecules, each with 17 molecular features. The QSVM regressor's goal is to predict an unseen molecule's solubility based on its features and the training data. We used a subset of 300 compounds with 4 features to minimize training time for demonstration purposes.

The first component of the quantum circuit to calculate the kernel matrix is feature preparation, also referred to as feature mapping or state preparation. The scaled feature vectors are input into the quantum circuits. An overview of the feature mapping quantum circuit diagrams for Qiskit and PennyLane is shown in the upper parts of Fig. 5a and b.

Rotational gates, such as RX and RZ, are commonly employed in feature preparation, with the rotation angles determined by the input features. In Qiskit, the feature preparation part (Fig. 5a) employs the ZZ feature map, where the x variables represent the input features of a molecule. The cascades of CNOT gates entangle all qubits. PennyLane's circuit visualization results, generated by the built-in pennylane.draw function, require the input angles from the data preparation step (Fig. 5b).

The complete quantum circuit comprises a feature mapping and its corresponding reversed counterpart. In reversible quantum circuits, applying all gates in reverse to a quantum state restores the initial quantum state. When the features are input into the forward feature mapping circuit, and the reversed feature mapping is applied using another molecule's features, the probability of retrieving the initial state is measured. This process, called ComputeAndUncompute in Qiskit, measures the fidelity between the forward and reversed quantum circuits. Fidelity, ranging from 0 to 1, indicates how closely the forward and reversed circuits match, thereby measuring similarity between the two molecules. The similarity between all pairs of molecules in the training set is aggregated to form the training kernel matrix. The testing kernel matrix is constructed from the fidelity between each testing molecule and all the training molecules.

Classical optimizers, such as support vector machine (SVM), based on the training kernel matrix and target values (solubility) to maximize the training accuracy. Finally, model performance is validated using the testing kernel matrix to assess the classifier's ability to predict the solubility of unseen molecules.

In summary, Qiskit wraps several trivial procedures in quantum machine learning into functions that users can readily call. For beginners, it is much easier to understand the meaning of each line of the code and build a workable classifier in a short time. On the other hand, PennyLane's tutorial describes building almost everything in detail when constructing a quantum regressor. This might benefit experienced users by allowing them to learn all the adjustable parts and customize their prediction models.

Limitations of this study

Quantum programming languages continuously evolve, with new versions addressing existing issues, introducing additional modules, and enhancing usability. For example, PennyLane can utilize the Qiskit plugin to achieve similar statevector and circuit visualizations. Additionally, experienced users may find it relatively easy to switch between platforms as needed. Third-party libraries built on these frameworks may extend functionality or simplify usage by providing higher-level abstractions.

Our analysis is primarily based on official tutorials, which may not fully represent all possible use cases. While we have tried to quantify our comparisons, certain aspects remain qualitative, introducing potential bias. To mitigate this, we included an evaluation by graduate students new to quantum computing, ensuring a fair comparison of the learning experience. However, individual learning styles and prior knowledge variations may influence the results.

As demonstrated in the usage examples of kernel-based quantum machine learning, reducing the number of molecules and features is often necessary to achieve affordable training times for tutorials, which may sacrifice the accuracy of the resulting prediction models. This trade-off arises because simulating quantum circuits on classical computers requires significantly more time and memory than classical machine learning methods. However, this challenge can be addressed with real quantum computing. While current quantum devices may experience errors due to hardware limitations, quantum error mitigation techniques have been developed and can be readily applied using Qiskit and PennyLane. As hardware continues to improve each year, the future of quantum computing in machine learning looks promising.

Conclusion

Comparing these two programming languages, Qiskit is better for quantum education. It wraps numerous intricate procedures into easily callable functions, enabling beginners to quickly comprehend every line of code and construct operational quantum circuits quickly. IBM's Quantum Composer is a noteworthy feature offering a web-based graphical interface. This tool allows students to craft quantum circuits by dragging and dropping gates. For those unfamiliar with programming, it offers a clear view of the workings of various quantum gates and circuits.

In contrast, PennyLane is better for research applications. While Qiskit offers streamlined solutions, PennyLane's quantum machine learning tutorials delve deeply into every detail of building variational quantum classifiers. This explanatory depth allows seasoned researchers in quantum machine learning to fully grasp all tunable aspects of the programming language, empowering them to customize models to their requirements. This profound flexibility is an outstanding tool for researchers interested in in-depth study and model fine-tuning.

Users can decide between these two quantum programming languages based on their needs and proficiency levels. Qiskit and PennyLane are premier choices in today's quantum computing domain.

Abbreviations

DFT	Density functional theory

GUI	Graphical user interface
HF	Hartree–Fock
QAA	Quantum adiabatic algorithm
QAOA	Quantum approximate optimization algorithm
QSVM	Quantum support vector machine
SVM	Support vector machine
VQE	Variational quantum eigensolver

Acknowledgements
Resources at the Laboratory of Computational Molecular Design and Metabolomics, the Department of Computer Science and Information Engineering, and the IBM Q Hub at National Taiwan University were used in performing this study.

Author contributions
Y.J.T. conceived the study. P.H.W., W.Y.W., and C.Y.L. conducted the literature survey and wrote the draft. P.H.W. and W.Y.W compared Qiskit and PennyLane and performed the case studies with usage examples. J.C.H. revised the paragraphs about quantum chemistry. P.H.W. organized and edited the draft. Y.J.T. provided feedback and revised the manuscript.

Funding
This work was financially supported by the "Center for Advanced Computing and Imaging in Biomedicine (NTU-113L900703)" from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan; National Science and Technology Council (NSTC 111–2320-B-002–043-MY2, NSTC 113–2119-M-033 -001).

Availability of data and materials
The dataset and codes supporting the conclusions of this article are available in the GitHub repository, https://github.com/wangpeihua1231/quantum-programming-platform.

Declarations

Competing interests
The authors declare no competing interests.

Author details
[1]Undergraduate Program in Intelligent Computing and Big Data, Chung Yuan Christian University, No. 200, Zhongbei Road, Taoyuan 320314, Taiwan. [2]Quantum Information Center, Chung Yuan Christian University, No. 200, Zhongbei Road, Taoyuan 320314, Taiwan. [3]Department of Computer Science and Information Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan. [4]Graduate Institute of Biomedical Electronics and Bioinformatics, College of Electrical Engineering and Computer Science, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan. [5]Physics Division, National Center for Theoretical Sciences, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan. [6]Center for Quantum Science and Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan.

Received: 10 November 2024 Accepted: 30 April 2025
Published online: 19 May 2025

References
1. Arute F, Arya K, Babbush R, Bacon D, Bardin JC, Barends R, Biswas R, Boixo S, Brandao FG, Buell DA (2019) Quantum supremacy using a programmable superconducting processor. Nature 574(7779):505–510
2. Aaronson S (2013) Quantum computing since Democritus. Cambridge University Press, Cambridge
3. Nielsen MA, Chuang IL (2010) Quantum computation and quantum information. Cambridge University Press, Cambridge
4. Zhong H-S, Wang H, Deng Y-H, Chen M-C, Peng L-C, Luo Y-H, Qin J, Wu D, Ding X, Hu Y (2020) Quantum computational advantage using photons. Science 370(6523):1460–1463
5. Savchuk M, Fesenko A (2019) Quantum computing: survey and analysis. Cybern Syst Anal 55:10–21
6. Ramos MJ, Fernandes PA (2008) Computational enzymatic catalysis. Acc Chem Res 41(6):689–698
7. McFadden J, Al-Khalili J (1999) A quantum mechanical model of adaptive mutation. Biosyst 50(3):203–211
8. Mortier J, Rakers C, Bermudez M, Murgueitio MS, Riniker S, Wolber G (2015) The impact of molecular dynamics on drug design: applications for the characterization of ligand-macromolecule complexes. Drug Discov Today 20(6):686–702
9. Bergholm V, Izaac J, Schuld M, Gogolin C, Ahmed S, Ajith V, Alam MS, Alonso-Linaje G, AkashNarayanan B, Asadi A Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv preprint. 2018.
10. Pattanayak S (2021) Quantum machine learning with python: using Cirq from google research and IBM Qiskit. Apress Berkeley, California
11. Aleksandrowicz G, Alexander T, Barkoutsos P, Bello L, Ben-Haim Y, Bucher D, Cabrera-Hernández FJ, Carballo-Franquis J, Chen A, Chen C-F Qiskit: An open-source framework for quantum computing. https://typeset.io/papers/qiskit-an-open-source-framework-for-quantum-computing-3jygtnm0rj. Accessed 30 Nov 2023
12. Stanwyck S, Bayraktar H, Costa T. cuquantum: Accelerating quantum circuit simulation on GPUs. In: APS March Meeting Abstracts, March 2022. APS Meeting Abstracts. The SAO/NASA Astrophysics Data System, p Q36.002. 2022
13. Hooyberghs J (2022) Q# language overview and the quantum simulator. Introducing microsoft quantum computing for developers using the quantum development kit and Q#. Apress, Berkeley
14. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M TensorFlow: a system for Large-Scale machine learning. In: 12th USENIX symposium on operating systems design and implementation (OSDI 16), Savannah, November 2016. USENIX Association. 2016; 265–283.
15. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L Pytorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems 32. Curran Associates, Inc. 2019
16. Hirthe S, Chalopin T, Bourgund D, Bojović P, Bohrdt A, Demler E, Grusdt F, Bloch I, Hilker TA (2023) Magnetically mediated hole pairing in fermionic ladders of ultracold atoms. Nature 613(7944):463–467
17. Georgescu IM, Ashhab S, Nori F (2014) Quantum simulation. Rev Mod Phys 86(1):153
18. Hartree DR The wave mechanics of an atom with a non-Coulomb central field. Part I. Theory and methods Mathematical Proceedings of the Cambridge Philosophical Society.1928; **24**: 89–110.
19. Parr RG (1983) Density functional theory. Annu Rev Phys Chem 34(1):631–656
20. Bartlett RJ, Stanton JF (1994) Applications of post-hartree-fock methods a tutorial. Rev Comput Chem 1:65–169
21. Rossmannek M, Barkoutsos PK, Ollitrault PJ, Tavernelli I (2021) Quantum HF/DFT-embedding algorithms for electronic structure calculations: Scaling up to complex molecular systems. J Chem Phys 154(11):114105
22. Dirac PAM (1927) The physical interpretation of the quantum dynamics. Proc R Soc Lond A 113:621–641
23. Ollitrault PJ, Miessen A, Tavernelli I (2021) Molecular quantum dynamics: a quantum computing perspective. Acc Chem Res 54(23):4229–4238
24. Bairey E, Arad I, Lindner NH (2019) Learning a local Hamiltonian from local measurements. Phys Rev Lett 122(2):020504
25. Bravyi S, Chowdhury A, Gosset D, Wocjan P (2022) Quantum Hamiltonian complexity in thermal equilibrium. Nat Phys 18(11):1367–1370
26. Gupta R, Selvarajan R, Sajjan M, Levine R, Kais S. Hamiltonian learning from time dynamics using variational algorithms. arXiv. Preprint posted online December 28. 2022
27. Farhi E, Goldstone J, Gutmann S (2014) A quantum approximate optimization algorithm. arXiv preprint https://arxiv.org/abs/1411.4028
28. Anguita D, Ridella S, Rivieccio F, Zunino R (2003) Quantum optimization for training support vector machines. Neural Netw 16(5–6):763–770
29. Rebentrost P, Mohseni M, Lloyd S (2014) Quantum support vector machine for big data classification. Phys Rev Lett 113(13):130503
30. Grover LK. A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. 1996; 212–219

31. Peruzzo A, McClean J, Shadbolt P, Yung M-H, Zhou X-Q, Love PJ, Aspuru-Guzik A, O'brien JL, (2014) A variational eigenvalue solver on a photonic quantum processor. Nat Commun 5(1):4213
32. Farhi E, Goldstone J, Gutmann S, Sipser M (2000) Quantum computation by adiabatic evolution. arXiv preprint https://arxiv.org/abs/quant-ph/0001106
33. Aoki T, Takahashi G, Kajiya T, Yoshikawa J-i, Braunstein SL, van Loock P, Furusawa A (2009) Quantum error correction beyond qubits. Nat Phys 5(8):541–546
34. Kluyver T, Ragan-Kelley B, Pérez F, Granger B, Bussonnier M, Frederic J, Kelley K, Hamrick J, Grout J, Corlay S et al (2016) Jupyter Notebooks-a publishing format for reproducible computational workflows. In: Loizides F, Schmidt B (eds) Positioning and power in academic publishing: players, agents and agendas. IOS Press, Amsterdam
35. IBM (2024) Quantum Kernel Machine Learning. https://qiskit-community.github.io/qiskit-machine-learning/tutorials/03_quantum_kernel.html. Accessed 21 Apr 2025.
36. Schuld M (2024) Kernel-based training of quantum models with scikit-learn. https://pennylane.ai/qml/demos/tutorial_kernel_based_training. Accessed 21 Apr 2025.
37. Sorkun MC, Khetan A, Er S (2019) AqSolDB, a curated reference set of aqueous solubility and 2D descriptors for a diverse set of compounds. Scientific Data 6(1):143

Pei-Hua Wang is a postdoc working in the laboratory of Yufeng Jane Tseng at National Taiwan University.

Wei-Yeh Wu is a graduate student in Yufeng Jane Tseng's laboratory.

Che-Yu Lee is a graduate student in Yufeng Jane Tseng's laboratory.

Jia-Cheng Hong is a graduate student in Yufeng Jane Tseng's laboratory.

Yufeng Jane Tseng is the professor leading the Laboratory of Computational Molecular Design and Metabolomics, the Department of Computer Science and Information Engineering of National Taiwan University.