FeynmanPAQS: a graphical interface program for photonic analog quantum computing

Hao Tang[a,*] **Xiao-Jun Xu**[b,*] **Yan-Yan Zhu**[a] **Jun Gao**[a] **Xuan Chen**[a] **Marcus Lee**[a] **Peng-Cheng Lai**[a] **and Xian-Min Jin**[a,b,*]

[a]Shanghai Jiao Tong University, Center for Integrated Quantum Information Technologies, School of Physics and Astronomy and State Key Laboratory of Advanced Optical Communication Systems and Networks, Shanghai, China

[b]TuringQ Co., Ltd., Shanghai, China

Abstract. We present a user-friendly software for photonic analog quantum computing with a graphical user interface (GUI) that allows for convenient operation without requiring programming skills. Hamiltonians can be flexibly set by either importing the waveguide position files or manually plotting the configuration on the interactive board of the GUI. Our software provides a powerful approach to theoretical studies of two-dimensional quantum walks, quantum stochastic walks, multiparticle quantum walks, and boson sampling, which may all be feasibly implemented in the physical experimental system on photonic chips, and it will inspire a rich diversity of applications for photonic quantum computing and quantum simulation. We have improved algorithms to ensure the efficiency of permanent calculation and provided case studies on educational uses, which bring users easier access to the studies of photonic quantum simulation.
 [DOI: 10.1117/1.OE.61.8.081804]

Keywords: quantum sciences and technologies; quantum software; quantum simulation; quantum walk; photonic lattice.

Paper 20220101SS received Feb. 3, 2022; accepted for publication Apr. 12, 2022; published online May 10, 2022.

1 Introduction

Quantum simulation, which has been a main sector of quantum information sciences since Feynman raised the concept of quantum computing,[1] uses the Hamiltonian of a quantum system to simulate the Hamiltonian of the target system in various subjects. The mapping does not need to be strict but only needs to be able to produce some expected features of the target system. Even some qualitative results instead of full quantitative details are valuable.[2,3] In the two major genres of quantum computing, universal (or digital) and analog, the former is more prone to the influence of errors and relies more heavily on error corrections. On the other hand, analog quantum computing has the advantages of the lower resource requirements and a higher tolerance level to imperfections of the quantum system. Together with the aforementioned wide demand and loose requirement for simulating target systems, these facts have made analog quantum computing an important tool for quantum simulation. The analog quantum simulation now has a rich diversity of applications in condensed-matter physics, high-energy physics, atomic physics, quantum chemistry, biology, etc.[2–5]

Among a variety of physical systems for quantum computing, including solid-state devices, atomic or nuclear spin systems, and superconducting devices, photons have many inherent advantages due to their fast speed and a lack of interaction with the environment.[6,7] In particular, photonic systems are very suitable for analog quantum computing and quantum simulation[8,9] because the integrated photonic lattices enable flexible and precise constructions of the target Hamiltonians. Some representative examples of photonic analog quantum information include

*Address all correspondence to Hao Tang, htang2015@sjtu.edu.cn; Xiao-Jun Xu, xuxiaojun@turingq.com; Xian-Min Jin, xianmin.jin@sjtu.edu.cn

quantum walks and boson sampling that evolves continuously in the well-designed photonic arrays. Quantum walks are regarded as a highly versatile approach for quantum simulation.[10,11] A quantum walk with a real two-dimensional evolution space was recently demonstrated on a photonic chip.[10] Boson sampling, a scheme that requires only a passive linear optics interferometer, single-photon source, and photodetection, may set a key milestone in the quantum computing field called quantum supremacy[12] and has inspired many experimental explorations.[13–18]

Similar to the impressive advances of quantum information processing hardware, quantum software has been rapidly developed in recent years.[19–21] A layered software architecture for quantum computing design tools was proposed in 2006; it defined the four-phase design flow to connect the front end to the physical quantum devices through quantum intermediate representation, quantum assembly language, and quantum physical operations language.[22] This layered structure inspired a series of software in different classical program languages for universal quantum computing in the past few years, including L|QUi|⟩[23] and Q# by Microsoft;[24,25] Qiskit by IBM;[26] Forest by Regetti,[27,28] a start-up company; ProjectQ by the Swiss Federal Institute of Technology in Zürich;[29] and $Q|\mathrm{SI}\rangle$ by University of Technology Sydney.[30] Meanwhile, a growing number of quantum clouds[31,32] on different quantum physical systems have been launched by companies such as IBM, Rigetti, and IonQ. These clouds and software make pioneering attempts with a focus on universal quantum computing.

Software for analog quantum computing and simulation is another major genre of quantum software. OpenFermion is open-source software for analog quantum simulation developed by Google,[33] with a specialization in simulating fermionic systems and quantum chemistry. In addition, as open quantum systems are always involved in quantum simulation, the Lindblad master equation solvers for quantum stochastic walks in open quantum systems have emerged; these include QuTiP, a Python package;[34,35] QSWalk, a Mathematica package useful for Hamiltonian based on graphs;[36] and a few improved Lindblad master equation solvers using Julia, TensorFlow, or massively parallel computers.[37–39] These analog quantum packages do not clearly correspond to a real quantum physical system. An exception is Strawberry Fields, a software for simulating photonic quantum computing, but it focuses on continuous-variable quantum computing.[40] Overall, software that provides practical instruction for analog quantum computing in photonic lattices is lacking.

Therefore, we launch FeynmanPAQS, a software for photonic analog quantum computing and simulation.[41] The name is a salute to Feynman for his pioneering ideas on quantum computers and insightful emphasis on quantum simulation.[1] PAQS is an acronym for photonic analog quantum simulation that suggest two key points of the software: the focus on analog quantum computing and the corresponding physical platform on the photonic system. FeynmanPAQS is compatible with various integrated photonic quantum circuits regardless of their fabrication method, such as silicon-on-insulator,[42] silica-on-silicon,[43] UV writing,[13] and femtosecond laser writing.[44–47] The advantages of this software can be summarized as follows:

- It is easy to operate using the user-friendly graphical user interface (GUI).[48–50]
- It supports flexible design of the photonic lattice and the corresponding Hamiltonian. In particular, the interactive board is enabled so that the user can manually locate the waveguide in any position just by clicking the mouse.
- It provides a practical way of realizing open quantum systems on the photonic lattices. It inspires analog quantum simulation for various applications on the photonic lattices.
- It uses optimization algorithms to speed up the calculation for multiparticle quantum walks and boson sampling. Multiple panels are also provided to view data from different perspectives.

The structure of this paper is as follows. In Secs. 2–5, we introduce the four modules of this software, namely, for two-dimensional quantum walks with abitrary Hamiltonian design (QW), for quantum stochastic walks on the photonic chip (QSW), for multiparticle quantum walks (MultiParticle), and for boson sampling (BosonSampling), as shown in Fig. 1. In each section, we set three sections. We begin with a brief overview of the theoretical models that support the related module in this section. We then introduce the GUI and detailed functions for that module.

Fig. 1 Framework of FeynmanPAQS. The software contains four modules, namely, the two-dimensional QW with arbitrary Hamiltonian design, QSW on the photonic chip, multiparticle quantum walks (MultiParticle), and boson sampling (BosonSampling).

We further discuss the educational uses for that module and give some case studies. For the module of MultiParticle and BosonSampling, we also compare our performance using the "Ryser + gray and Glynn + gray" mixed algorithm with those using other or no optimization algorithms. A brief discussion of this software is given in Sec. 6.

2 Module for Two-Dimensional Quantum Walks with Arbitrary Hamiltonian Design (QW)

2.1 *Quantum Walks with Arbitrary Hamiltonian Design*

Quantum walks, the quantum analog of classical random walks,[51,52] have unique features of interference and superposition, which bring quantum walks remarkably different behaviors and normally faster transport performances compared with classical random walks. Therefore, quantum walks have become a powerful approach to quantum information algorithms,[53–55] and quantum simulation for various systems.[5,8,56] To apply quantum walks to real applications, a two-dimensional evolution space of a large scale and a flexible design of the Hamiltonian for quantum walks are two preliminary requirements. A few examples of such large-scale two-dimensional quantum walks have now been experimentally realized on the integrated photonic chip.[10,11] This module of QW provides a platform to allow for more Hamiltonian designs to help researchers explore a rich diversity of quantum walk applications.

As concerned in this module (QW), single photons are injected into the photonic lattice consisting of straight waveguides to form two-dimensional quantum walks. Photons propagating through these coupled waveguides are described by the Hamiltonian

$$H = \sum_{i}^{N} \beta_i a_i^{\dagger} a_i + \sum_{i \neq j}^{N} C_{i,j} a_i^{\dagger} a_j, \tag{1}$$

where β_i is the propagating constant in waveguide i and $C_{i,j}$ is the coupling strength between waveguide i and j. $C_{i,j}$ that mainly depends on waveguide spacing can be experimentally obtained via a coupled mode approach.[45] In the module, we use the empirically fitted relationship between the coupling coefficient C (unit: cm^{-1}) and the waveguide spacing d (unit: μm) as follows: $C = 41.42 \times \exp(-d/4.616)$. Hence once the waveguide spacing is set, the coupling coefficient in the Hamiltonian is obtained (see Fig. 2).

For a quantum walk that evolves along the waveguide, the propagation length z is proportional to the propagation time by $z = c_w t$, where c_w is the speed of light in the waveguide, so we use z instead of t for simplicity. The wavefunction that evolves from an initial wavefunction satisfies

$$|\Psi(z)\rangle = e^{-iHz}|\Psi(0)\rangle, \tag{2}$$

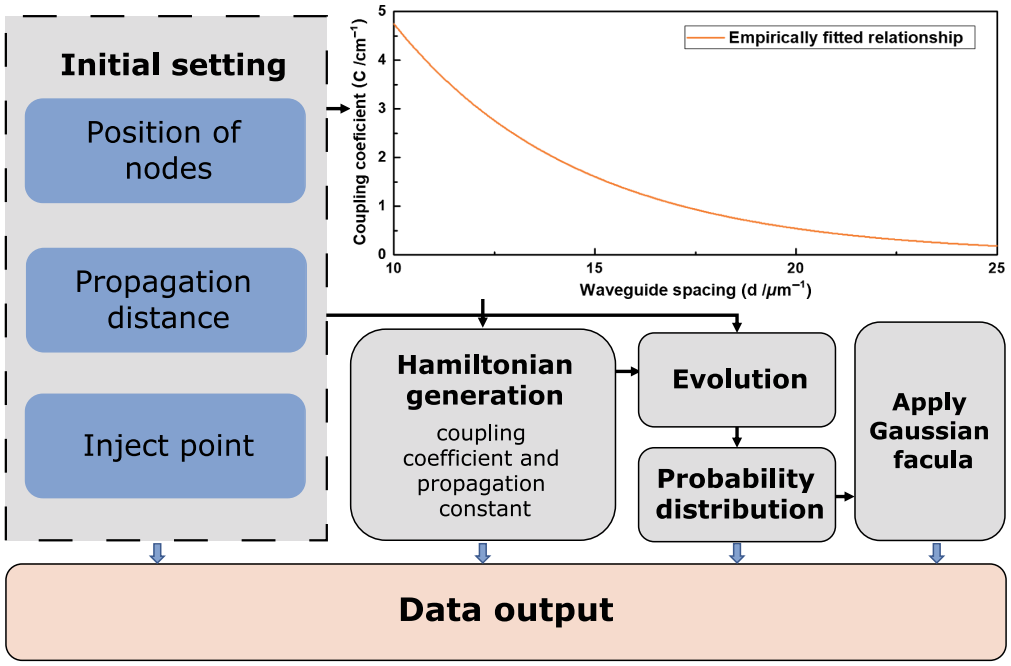


Fig. 2 Theoretical calculation for quantum walks on a photonic chip. The coupling coefficient C as a function of the center-to-center waveguide spacing and the flowchart for the procedures of calculating the evolution result for quantum walks on a photonic chip.

where $|\Psi(z)\rangle = \sum_j a_j(z)|j\rangle$ and $|a_j(z)|^2$ equals $P_j(z)$, the probability of the walker being found at waveguide j at propagation length z.

In the experiment, we observe such probability distribution by injecting a laser beam or single-photon source into one waveguide and measuring the evolution patterns using an intensified charge-coupled device camera. The normalized light intensity of each waveguide corresponds to the probability at this waveguide. As the facula at each waveguide is normally in the shape of the two-dimensional Gaussian distribution, we also plot the Gaussian-shaped facula for presenting the theoretically obtained probability distribution in the software modules.

2.2 *GUI and Functions for the Module of QW*

The workflow in this module for theoretically calculating quantum walks on a photonic chip includes the following four steps: generate Hamiltonian, evolve (exponentiate Hamiltonian), obtain probability distribution, and apply Gaussian facula (See Fig. 2). A GUI of this module is further given in Fig. 3, which shows a case of a two-dimensional quantum walk with an arbitrary Hamiltonian design from the panel "design your own chip pattern." Users can then define the propagation distance and input nodes on the GUI. The Hamiltonian matrix corresponding to the pattern and the calculated probability distribution are displayed and can be exported as image and data files.

2.3 *Educational Use for the Module of QW*

The highlight of this module is the flexible setting of Hamiltonian, which is fun and straightforwardly shows the possible Hamiltonian manipulation on the photonic lattice. Teachers can then go deeper by showing that the specifically designed Hamiltonians can indeed be applied to different quantum simulation tasks. For instance, the Hamiltonians that are required in many topological photonics models such as the Aubry–Andŕe–Harper model[57] and Su–Schrieffer–Heeger model[58] can be implemented on the photonic lattice.

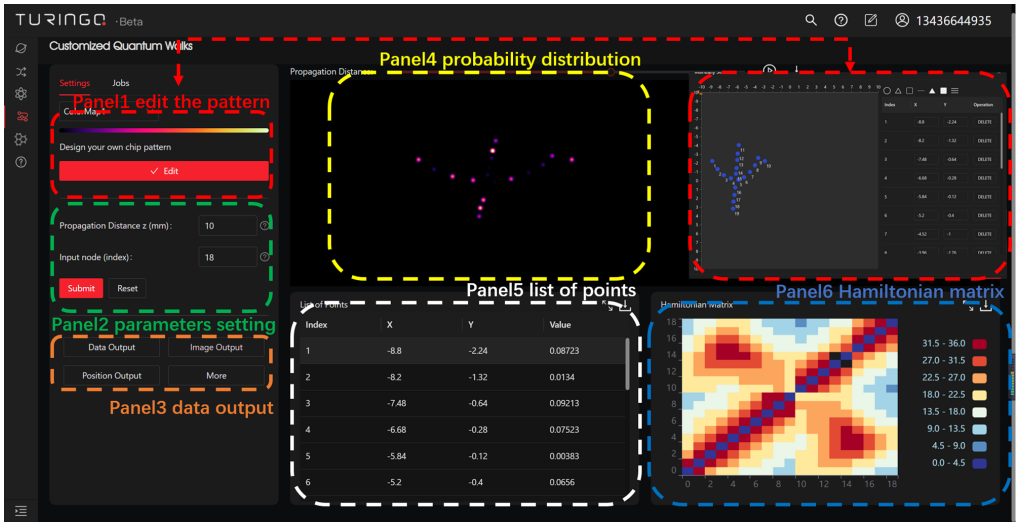


Fig. 3 The GUI for the module of QW. An array forming the shape of Ψ has been set by manually plotting the waveguide in the interactive board, and its evolution pattern with Gaussian-shaped facula for a certain propagation distance and input node is plotted. Different panels on the GUI are marked in the figure, and the functions of all buttons in these panels are explained in Table 1.

Table 1 Functions of all panels for the module of QW.

Panel ID	Button	Function
Panel 1	Edit	Open the board for manually setting the waveguide positions
Panel 2	Propagation distance z	Input the value of z
	Input node (index)	Define the initial quantum state
	Submit	Submit the parameters and compute the result
	Reset	Reset the parameters by default
Panel 3	Data output	Export all data in a json file
	Image output	Export the waveguide intensity in a png file
	Position output	Export the positions and intensities of each waveguide in a csv file
Panel 4	The scroll bar and play	Show the dynamic waveguide intensity pattern
Panel 5	N.A.	Show the position and intensity of each waveguide
Panel 6	N.A.	Show the Hamiltonian matrix

Note: N.A. means that there is no specific button on this panel, so we explain the function of the whole panel.

2.3.1 *Case study*

In a course that we deliver on analog quantum computing,[59] we use this module to demonstrate a case of quantum fast hitting. This is a representative algorithm to show the advantages of quantum walks, originally proposed by Childs et al.,[52] and it was recently adapted to a photonic experiment.[11] Students are asked to plot a hexagonal glued tree structure (see Fig. 4, which is reproduced from Ref. 11) and input light in an entry site, the left-most node as the vertex of one tree. Students then observe the hitting efficiency, the probability at the exit site, and the

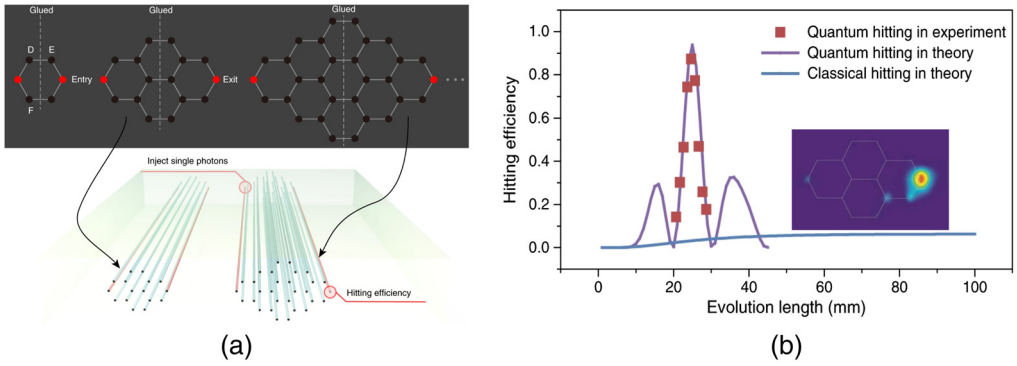


Fig. 4 Quantum fast-hitting on hexagonal graphs. (a) Mapping the hexagonal glued trees onto the photonic chip. (b) The hitting efficiencies for quantum and classical hitting. Inset of (b) is the experimental pattern with the highest hitting efficiency. The figures are reproduced from Ref. 11.

right-most node as the vertex of another tree. They see that the probability changes dynamically over time, which reveals the dynamic nature of quantum walks, and they see that the hitting efficiency can be very high, usually above 90%. The teacher demonstrates running the hitting task with classical random walks, which only gets a low hitting efficiency. Through this example, students see quantum advantages in fast-hitting tasks.[52]

3 Module for Quantum Stochastic Walks on the Photonic Chip

3.1 *Quantum Stochastic Walks on the Photonic Chip*

In real systems, environmental noise easily causes some dephasing and decoherence of the original quantum system. This leads to the issue of an open quantum system,[3–5] which can be addressed with a useful approach named the quantum stochastic walk.[60]

The Lindblad master equation, the most common formulation of quantum stochastic walks, is a differential equation that incorporates both quantum and classical walks in the continuous-time evolution:[36,60]

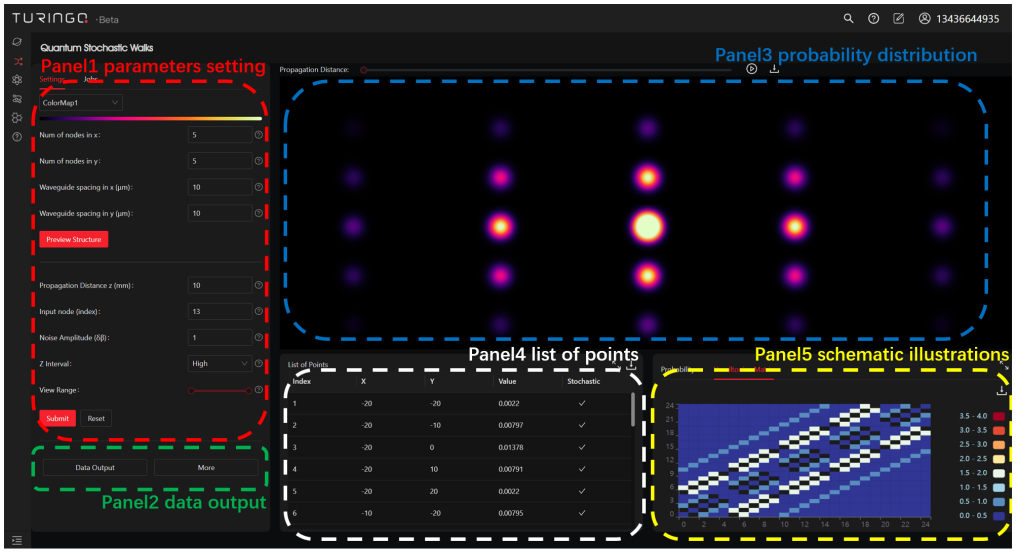
$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -(1-\omega)i[H,\rho] + \omega\sum_{ij}\left(L_{ij}\rho L_{ij}^{\dagger} - \frac{1}{2}\{L_{ij}^{\dagger}L_{ij},\rho\}\right). \tag{3}$$

The first part right to Eq. (3) represents the quantum walk evolution, where H is the Hamiltonian operator. The second part, which contains the Lindblad operators L_{ij}, describes the classical random walk evolution. The parameter ω interpolates the weight of quantum walk and classical walk, that is, a higher value of ω suggests a larger portion of the classical walk in the mixture of the quantum stochastic walk.

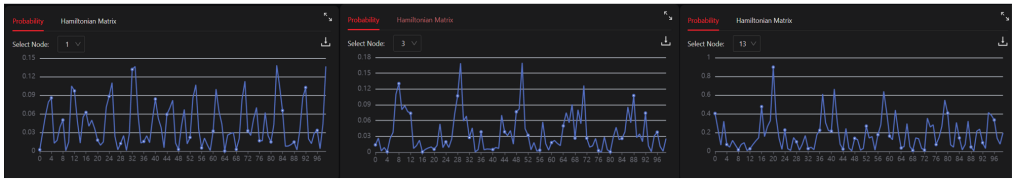
There are a few solvers for Lindblad equations, such as Qutip[34,35] and QSWalk.[36] However, the parameters in the Lindblad equation do not have a clear correspondence to the physical parameters of the photonic quantum systems. In the photonic chip, a $\Delta\beta$ photonic model was raised[61] instead to achieve a continuous-time quantum stochastic walk

$$i\frac{\mathrm{d}\psi_n}{\mathrm{d}z} = \Delta\beta_n(z)\psi_n + \sum_{m\neq n} C_{mn}\psi_m, \tag{4}$$

where ψ_n is the wave function at waveguide n and $\Delta\beta_n(z)$ is the change to the propagation constant $\beta_n(z)$. The $\beta_n(z)$s are originally a constant if the photonic chip fabrication parameters are kept stable, and the photonic array makes pure quantum walks. However, when $\Delta\beta_n(z)$s are added along the evolution path, which can be physically realized by randomly tuning certain fabrication parameters along the waveguide, the noise is introduced into the photonic quantum system. The strength of the noise is positively related to the amplitude of the random fluctuations



(a)



(b)

Fig. 5 The GUI for the module of QSW. (a) Different panels on the GUI are marked in the figure, and the functions of all buttons in these panels are explained in Table 2. (b) In panel 5, users can also view the probability of each specified waveguide.

by $\Delta\beta_n(z)$s. While the experimental $\Delta\beta$ approach is not a strict mapping of the Lindblad master equations, it manages to introduce dephasing terms that make a quantum stochastic walk.

3.2 *GUI and Functions for the Module of QSW*

The GUI for this module of QSW is shown in Fig. 5. It is similar to that for the module of QW, in that they both follow the same four-step workflow to get the theoretical result. Still, QSW has its special panels to facilitate the study of the $\Delta\beta$ approach for quantum stochastic walks. Users can define how frequently $\Delta\beta$ varies along the waveguide (by filling "z interval"), and the strength $\Delta\beta$ variation (by filling "noise amplitude $\Delta\beta$"). In panel 5, users can view the evolution probability against the evolution length for each specified waveguide [see Fig. 5(b)].

3.3 *Educational Use for the Module of QSW*

By adding various $\Delta\beta$ detunings into the photonic lattice, users are able to create quantum stochastic walks that meet wider scenarios in quantum simulation. Via this module, users will learn the rich capabilities of photonic lattice and explore the difference between quantum stochastic walks and quantum walks.

3.3.1 *Case study*

Students can try an exercise using quantum stochastic walks to form the Haar measure, as has been recently conducted in an experiment[62] (see Fig. 6, which is reproduced from Ref. 62). By averaging the evolution probabilities for many quantum stochastic walk patterns, students observe that the average probabilities converge to a uniform distribution, which meets the criteria for a Haar measure.[62,63] They see that this feasible and scalable experimental approach can make

Table 2 Functions of all panels for the module of QSW.

Panel ID	Button	Function
Panel 1	Number of nodes in x, y	Input the number of nodes in the x or y direction of a 2D lattice
	Waveguide spacing in x, y (μm)	Input the spacing in the x or y direction of a 2D lattice
	Preview structure	Preview the lattice with the above parameters
	Propagation distance z	Input the value of the propagation distance
	Input node (index)	Define the input quantum state
	Noise amplitude ($\Delta\beta$)	Input the scale of $\Delta\beta$ detuning
	Z interval	Define how frequently we change $\Delta\beta$
	View range	Define the range of z to view
	Submit	Submit the parameters and compute the result
	Reset	Reset the parameters by default
Panel 2	Data output	Export all data in a json file
Panel 3	N.A.	Show the waveguide intensity pattern
Panel 4	N.A.	Show the list of position and intensity of each waveguide
Panel 5	Possibility	Show the possibility of the selected node with respect to the propagation distance
	Hamiltonian matrix	Show the Hamiltonian matrix

Note: N.A. means that there is no specific button on this panel, so we explain the function of the whole panel.

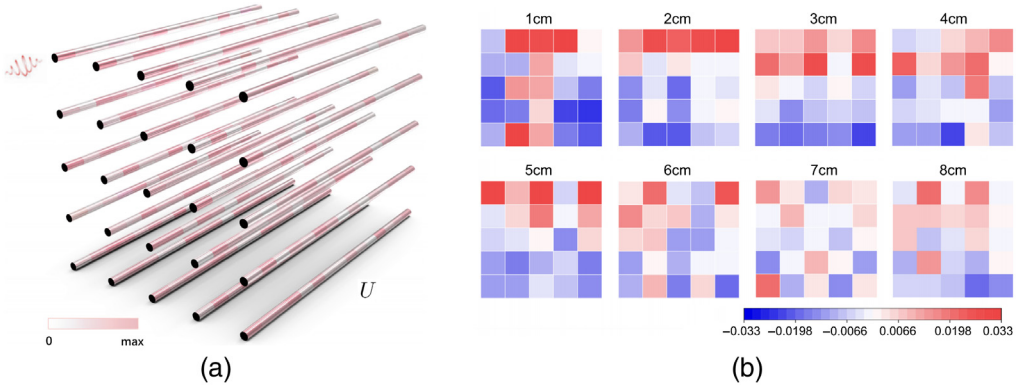


Fig. 6 Generating Haar randomness via quantum stochastic walks on photonic chips. (a) Illustration of generating quantum stochastic walks via $\Delta\beta$ detunings on the photonic lattice. (b) The average of experimental results converges to uniform distribution as the propagation length increases. The figures are reproduced from Ref. 62.

the Haar randomness useful as it can be applied to many quantum information processing tasks. Students are then asked to turn the "noise amplitude $\Delta\beta$" into zero, which reduces the model to a pure quantum walk, and they see that the average probabilities always change dynamically and never converge. This exercise shows the difference between quantum walks and quantum stochastic walks.

4 Module for Multiparticle Quantum Walks (MultiParticle)

4.1 *Multiparticle Quantum Walks*

In the above two modules, QW and QSW, we have focused on a single particle evolving in a two-dimensional quantum system. It is natural to wonder about the case of injecting more than one particle into the system, i.e., two or more indistinguishable particles. In this case, two main features of quantum optics, quantum interference and quantum correlation play central roles. The genuine quantum interference [also known as Hong–Ou–Mandel (HOM) effect[64]], which cannot be classically simulated, gives rise to nonclassical quantum correlations.[65] With two particles populating a two-dimensional lattice, the corresponding state space can be easily extended to a greater dimension.[66] Such high-dimensional graphs demonstrate quantum speedup for the continuous-time search algorithm.[55]

Another striking departure of quantum mechanics from classical physics is that particle statistics, either bosonic or fermionic, can influence the quantum correlation, exhibiting either bunching or antibunching behaviors. These phenomena can be physically simulated with bipartite entanglement in the form of

$$|\psi\rangle = \frac{1}{\sqrt{2}}(a_i^\dagger b_j^\dagger + e^{i\phi} a_j^\dagger b_i^\dagger)|00\rangle. \tag{5}$$

Phase ϕ controls the states' symmetry, thus determining the particle statistics.

With an even greater number of N particles involved in the evolution through a system of M waveguides, the process can be regarded as a generalized HOM effect and a photon scattering model. The outcome also depends on whether the particles are distinguishable or indistinguishable and if they are indistinguishable, whether they are bosons or fermions. This distinguishability depends on whether the particles are identical, e.g., whether they have temporal delay or the same frequency when they enter the waveguide array.

If the particles are indistinguishable bosons, then given a Hamiltonian H and its corresponding unitary evolution $U = e^{-\frac{iHt}{\hbar}}$, we can calculate the probabilities of various states occurring. If we define a configuration S_i to be the number of photons injected in waveguide i and T_j to be the number of photons exiting waveguide j, then the probability of an initial state $S = |S_1 \cdots S_M\rangle$ evolving into a final state $T = |T_1 \cdots T_M\rangle$ is given as[15]

$$P(T|S) = \frac{|\mathrm{Per}(U^{(S,T)})|^2}{\prod_{i=1}^M S_i! \prod_{j=1}^M T_j!}, \tag{6}$$

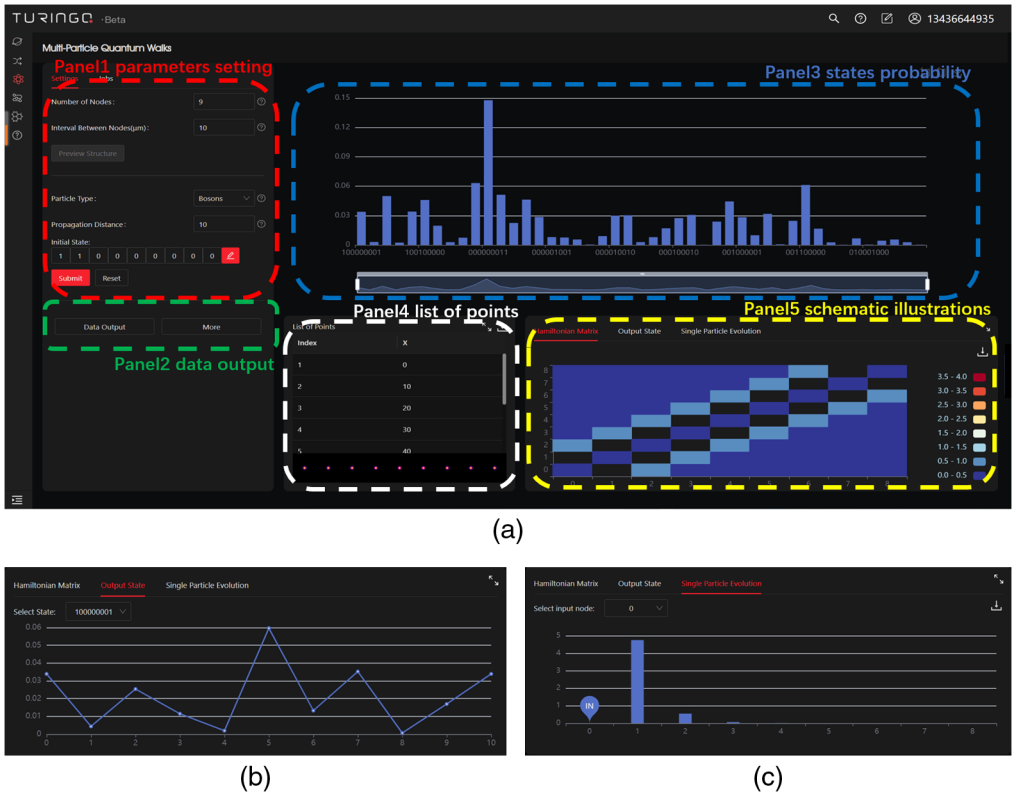
where Per is the permanent of a matrix and $U^{(S,T)}$ is a submatrix of U that is constructed by taking S_i copies of the i'th column of U and taking T_j copies of the j'th row of U. Since $\sum_{i=1}^M S_i = \sum_{j=1}^M T_j = N$, $U^{(S,T)}$ has dimensions $N \times N$. Fermions have a similar equation

$$P(T|S) = \frac{|\mathrm{Det}(U^{(S,T)})|^2}{\prod_{i=1}^M S_i! \prod_{j=1}^M T_j!}, \tag{7}$$

where Det is the determinant of a matrix. As for distinguishable particles, they evolve independently (classically). Hence cross terms can be eliminated by taking a linear combination of the determinant and permanent

$$P(T|S) = \frac{1}{2}\frac{|\mathrm{Per}(U^{(S,T)})|^2 + |\mathrm{Det}(U^{(S,T)})|^2}{\prod_{i=1}^M S_i! \prod_{j=1}^M T_j!}. \tag{8}$$

With these formulas incorporated into the module of the multiparticle, we are able to use this software to calculate the multiparticle evolution.



(a)

(b)

(c)

Fig. 7 The GUI for the module of MultiParticle. (a) Different panels on the GUI are marked in the figure, and the functions of all buttons in these panels are explained in Table 3. (b) The probability of a specified quantum state at different propagation lengths and (c) the probability of a specified waveguide at different propagation lengths are also provided in panel 5 of the GUI.

4.2 *GUI and Functions for the Module of MultiParticle*

The GUI for multiparticle is shown in Fig. 7. Users can define the "number of nodes" and "interval between nodes" for a one-dimensional array. They then choose input nodes for the "initial state." In particular, users can choose the "particle type," which can be bosons, fermions, or distinguishable. The above panel shows the breakdown of probability distribution of all states, and a scroll bar can be used for the very long list. Users can specify an output state and observe its probability at different evolution lengths. They can also view the evolution dynamics for inputting only a single particle in a specified input node.

4.3 *Educational Use for the Module of MultiParticle*

This module can be a good tool for teaching the bunching and antibunching effects of different types of particles.[67,68] This is a key issue in quantum optics and can be very naturally linked to the up-to-date research in quantum optics and quantum information processing.

4.3.1 *Case study*

Students are asked to generate any initial input state and then pick an output state, e.g., $|00002000\rangle$, where two photons output from the same waveguide. Students try the three different particle types and write down the output probability of that output state (e.g., $|00002000\rangle$) for each particle type. They see that the probability goes to zero for fermions because of the antibunching effects, while the value for bosons doubles the value for distinguishable particles because of photon bunching. Students are then impressed to see the same effect in real experiments. In a work of topological protection of two-photon quantum correlation on a photonic chip,[69] the authors show the measured two-photon coincidence count from the output end of

Table 3 Functions of all panels for the module of MultiParticle.

Panel ID	Button	Function
Panel 1	Number of nodes	Input the number of nodes in 1D array
	Interval between nodes(μm)	Input the interval between nodes
	Preview structure	Preview the array with the parameters above
	Propagation distance	Input the propagation length
	Particle type	Choose a particle type from "bosonic," "fermionic," and "distinguishable"
	Initial state	Define the initial state
	Submit	Submit the parameters and compute the result
	Reset	Reset the parameters by default
Panel 2	Data output	Export all data in a json file
Panel 3	N.A.	Show the waveguide intensity pattern
Panel 4	N.A.	Show the list of the position of each waveguide
Panel 5	Hamiltonian matrix	Show the Hamiltonian matrix
	Output state	Show the probability of a specified quantum state at different propagation lengths
	Single particle evolution	Show the probability of a specified waveguide at different propagation lengths

Note: N.A. means that there is no specific button on this panel, so we explain the function of the whole panel.

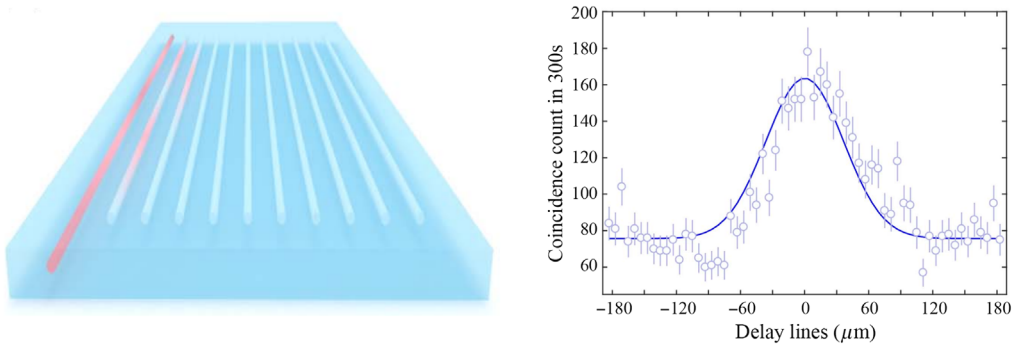


Fig. 8 The photon bunching effect in the experiment. (a) A one-dimensional photonic lattice with photons injected into the boundary waveguide. (b) The two-photon coincidence counts against different delay lines between two photons, measured from the output end of the boundary waveguide. The figures are reproduced from Ref. 69.

a waveguide in the photonic lattice. The count has a clear peak that doubles the bottom value when the delay between the two photons reduces to zero (see Fig. 8, which is reproduced from Ref. 69). This module of multiparticle can well explain those experimental phenomena.

4.4 *Optimization Algorithms for the Permanent Calculation*

It is also worth mentioning that we manage to use effective optimization algorithms for the key part of the theoretical model of this module: the permanent calculation. Computing the

permanent has been proven to be of #P complexity, even harder than NP-complete, and is widely believed to be classically intractable. The permanent of an $n \times n$ matrix per M is defined as

$$\mathrm{Per}\, M = \sum_{\sigma \in S} \prod_{i=1}^{n} M_{i,\sigma(i)}, \tag{9}$$

where σ is a permutation of the set $\{1, \ldots, n\}$ and S is the group of all such permutations. If we try to compute the permanent based on this formula alone, we obtain an algorithm with time complexity $O(n!n)$, which is very inefficient.

Two better algorithms have been discovered, the first of which is due to Ryser[70]

$$\mathrm{Per}\, M = (-1)^n \sum_{\epsilon \in S} (-1)^{\sum_i \epsilon_i} \prod_{k=1}^{n} \sum_{j=1}^{n} \epsilon_j M_{jk}, \tag{10}$$

where $\epsilon = (\epsilon_1, \ldots, \epsilon_n) \in S = \{0, 1\}^n$ (i.e., ϵ is a n-dimensional vector in which all elements are 0 or 1, and S is the set of all 2^n possible ϵ). The second algorithm is due to Glynn[71]

$$\mathrm{Per}\, M = 2^{1-n} \sum_{\delta \in P} \left(\prod_{i=1}^{n} \delta_i \right) \prod_{k=1}^{n} \sum_{j=1}^{n} \delta_j M_{jk}, \tag{11}$$

where $\delta = (1, \delta_2, \ldots, \epsilon_n) \in P = \{\pm 1\}^n$ and there are 2^{n-1} possible δ, since we fix $\delta_1 = 1$.

These two formulas are both related to polarisation identities of symmetric tensors and in fact part of a larger family of permanent algorithms.[72]

These formulas give algorithms of complexity $O(2^n n^2)$ and can be further reduced to $O(2^n n)$ using a gray code.[73] We explain this using Ryser's formula, and the same logic applies to Glynn's. If we say that $\lambda_k = \sum_{j=1}^{n} \epsilon_j M_{jk}$, then if ϵ and ϵ' differ by a single ϵ_m (i.e., the Hamming distance is 1), then $\lambda'_m = \lambda_m + (\epsilon'_m - \epsilon_m) M_{mk}$, and we only need to calculate one element of the sum instead of all n elements.

Comparing the two formulas, it seems that Glynn's formula is faster than Ryser's by a factor of 2, as the number of possible δ is half the number of possible ϵ. This is indeed true as $n \to \infty$, but Ryser's formula seems to be better optimized for small matrices where $N < 6$ in our implementations (see Fig. 9).

Therefore, in this software, we use "Ryser + gray and Glynn + gray" mixed algorithm to calculate the permanent, i.e., "Ryser + gray" for $N < 6$ and "Glynn + gray" for large N.

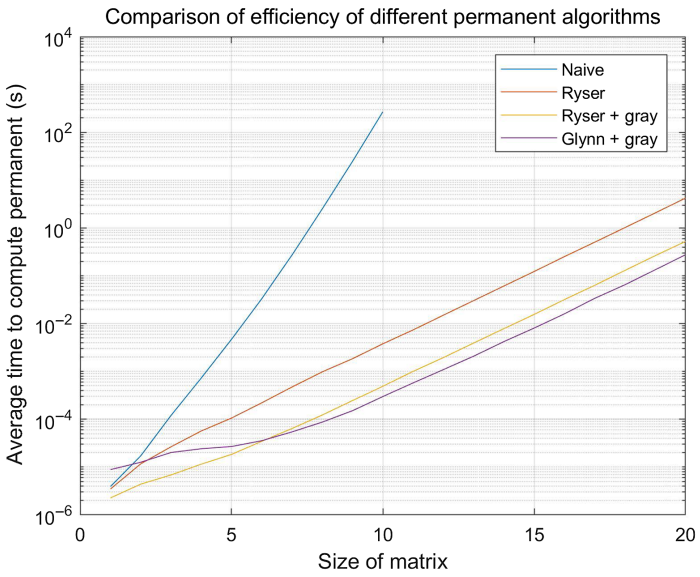


Fig. 9 Comparison of the calculation time using different permanent algorithms.

This ensures that we always have an optimized permanent calculation efficiency for different scenarios of photon counts.

5 Module for Boson Sampling (BosonSampling)

5.1 *Boson Sampling*

Boson sampling, an analog quantum computing model first raised by Aaronson and Arkhipov,[74] has been an emerging field in the quantum research community. Unlike universal quantum computation scheme that usually requires active elements, boson sampling schemes only require a passive linear optics interferometer, single-photon source, and photodetection (photon number resolving is not necessary). Results show that these experimentally friendly devices may offer the first evidence that refutes the extended church-turing (ECT) thesis and set a key milestone in the quantum computing field called quantum supremacy.[12]

The boson sampling problem can be modeled as a multiphoton scattering process. We first prepare an input state consisting of N single photons in an M modes passive linear optics interferometer. The process is very similar to that in the multiparticle quantum walks in which N single photons are injected into M waveguides, but a key difference lies in there not being a unitary matrix in boson sampling schemes that can be effectively described by the Hamiltonian, similar to $U = e^{-\frac{iHt}{\hbar}}$ in the case of multiphoton quantum walks. In boson sampling, one normally defines U, a unitary map on the creation operators, directly. The injected photons in the boson sampling scheme interfere and scatter in the linear optics interferometers. The interferometer implements a Haar-random M-mode transformation U on these N indistinguishable bosons:

$$Ua_i^\dagger U^\dagger = \sum_{j=1}^{M} U_{i,j} a_j^\dagger, \tag{12}$$

where i and j denote the i'th and j'th modes of the interferometer. After the transformation, an input configuration, which is denoted as S with $\sum_{i=1}^{M} S_i = N$, becomes an output state, which is a superposition of different configurations T for the output modes, denoted as $|\psi_{\text{out}}\rangle = \sum_T \gamma_{S,T} |n_1^T n_2^T \cdots n_m^T\rangle$, where n_i^T is the corresponding photon number n in the i'th mode.

The hardness of this problem is rooted in evaluating the value of $P(T|S)$, related to the permanent of the scattering amplitudes. The calculation can follow the same equation as Eq. (6).

5.2 *GUI and Functions for the Module of BosonSampling*

In the GUI of BosonSampling, users need to import an $M \times M$ unitary scattering matrix as U and are informed of an error if the imported matrix does not satisfy the requirement for a unitary matrix. The users also need to design their interferometers. Generally, there are two types of decomposition, namely, the Reck's type[75] and the Clements's type[76] (see Fig. 10).

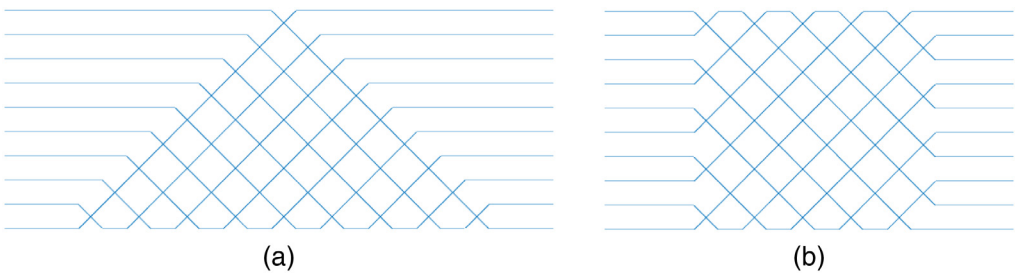


Fig. 10 Schematic diagram of the decomposition. An example of (a) 10 modes in the Reck decomposition and (b) 10 modes in the Clements decomposition.

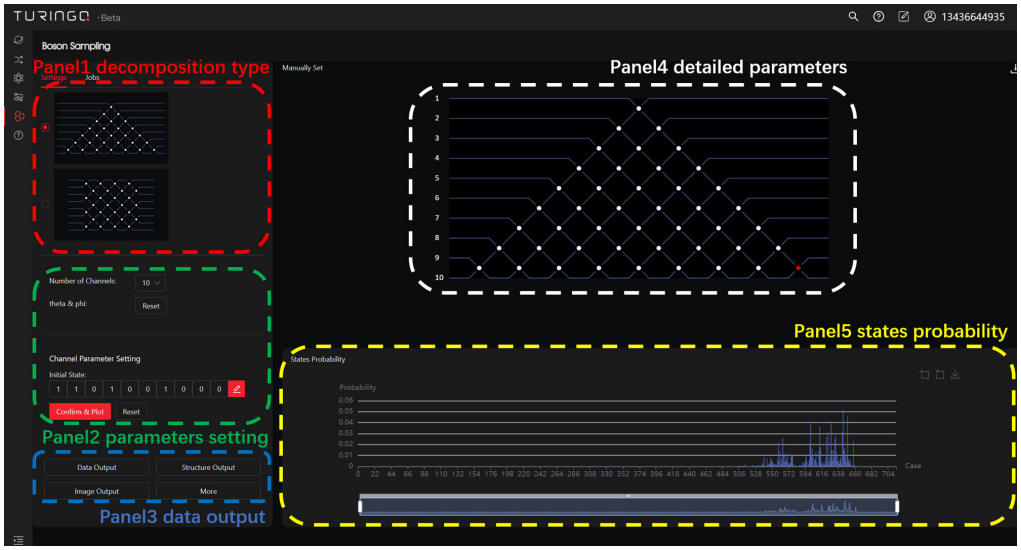


Fig. 11 The GUI for the module of BosonSampling. Different panels on the GUI are marked in the figure, and the functions of all buttons in these panels are explained in Table 4.

After choosing the unitary scattering matrix U and the decomposition style, users need to set the parameters for beam splitters and phase shifts that accomplish pairwise transformations between channels by satisfying $U = D(\prod T_{m,n})$, where D is a diagonal matrix and $T_{m,n}$ is the transformation from channel m to n ($m = n - 1$) through a lossless beam splitter with the reflectivity $\cos\theta$ and through a phase shift ϕ at the input m side.[76] $T_{m,n}(\theta, \phi)$ reads as

$$T_{m,n}(\theta,\phi) = \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & & & & & \iddots & \vdots \\ \vdots & & \ddots & & & \iddots & & \vdots \\ \vdots & & & e^{i\phi}\cos\theta & -\sin\theta & & & \vdots \\ \vdots & & & e^{i\phi}\sin\theta & \cos\theta & & & \vdots \\ \vdots & & \iddots & & & \ddots & & \vdots \\ \vdots & \iddots & & & & & 1 & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 1 \end{pmatrix}. \tag{13}$$

Users can manually set the parameters for all $T_{m,n}$s. They can also ask the software to randomly set the parameters by choosing "reset" for θ and ϕ. A GUI for BosonSampling is shown in Fig. 11, where the decomposition with the defined type, number of modes, and injected photons is shown in the interactive board. The interferometer parameters can be shown by placing the mouse on the corresponding white dot that represents an interferometer. The obtained probability distribution data can be exported to facilitate further analysis for boson sampling.

5.3 *Educational Use for the Module of BosonSampling*

Users can get familiar with the framework of boson sampling via this module. When they increase the injected photon numbers and mode numbers, they experience the increase of complexity of the output states and the computational time on classical computers. This deepens their understanding of the quantum advantage for boson sampling in real quantum hardware.

5.3.1 *Case study*

In class, students may try to reproduce previously published works on boson sampling.[13–16] In their experiments, they characterize their unitary matrix for their photonic chip. Students are

Table 4 Functions of all panels for the module of BosonSampling.

Panel ID	Button	Function
Panel 1	N.A.	Choose decomposition type (Reck or Clements)
Panel 2	Number of channels	Define the number of modes
	Theta and phi:Reset	Reset θ and ϕ values randomly
	Initial state	Define the initial state
	Confirm and plot	Submit the parameters and compute the result
	Reset	Reset the parameters by default
Panel 3	Data output	Export all data in a json file
	Structure output	Export the recent structure of the interferometers
	Image output	Export the recent schematic diagram in panel 4 in a png file
Panel 4	N.A.	Show the θ and ϕ values of a particular interferometer when putting the mouse on it
Panel 5	N.A.	Show the probability of each state

Note: N.A. means that there is no specific button on this panel, so we explain the function of the whole panel.

asked to input those data and get the theoretical sampling result, which can be compared with the theoretical and experimental results in their papers.

6 Discussion

Photonic lattices are a promising physical system for analog quantum computing and quantum simulation. So far, increasing efforts are being made for their experimental demonstration. We launched FeynmanPAQS with a user-friendly GUI to facilitate education and research on quantum information sciences based on integrated photonic lattices, without heavy prerequisite knowledge of quantum physics and computer sciences. We have incorporated several versatile tools of analog quantum computing in FeynmanPAQS, covering two-dimensional quantum walks, quantum stochastic walks, multiparticle quantum walks, and boson sampling, and all of these tools can be feasibly implemented on photonic chips in the experiment. FeynmanPAQS allows for flexible Hamiltonian designs and engineering, as well as various settings of input photons, particle types, mode number, interferometer parameters, etc., which inspire brainstorming for potential applications for real problems. The version of the software described in this paper is FeynmanPAQS 1.0. We will continue to update with additional case studies and functions in the near future.

Acknowledgments

This research was supported by the National Key R&D Program of China (2019YFA0706302, 2019YFA0308700, and 2017YFA0303700), the National Natural Science Foundation of China (61734005, 11761141014, 11690033, and 11904229), the Science and Technology Commission of Shanghai Municipality (STCSM) (21ZR1432800, 20JC1416300, and 2019SHZDZX01), and the Shanghai Municipal Education Commission (SMEC) (2017-01-07-00-02-E00049). X.-M. J. acknowledges additional support from a Shanghai talent program. H. T. and X.-M. J. thank Zhiyuan College at Shanghai Jiao Tong University for supporting the development of a course on quantum computing (Grant No. WF620160207/001/008), in which the students have used FeynmanPAQS to facilitate studies on analog quantum computing.

References

1. R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.* **21**, 467–488 (1982).
2. I. Buluta and F. Nori, "Quantum simulators," *Science* **326**, 108–111 (2009).
3. I. M. Georgescu, S. Ashhab, and F. Nori, "Quantum simulation," *Rev. Mod. Phys.* **86**, 153–185 (2014).
4. J. Argüelloluengo et al., "Analogue quantum chemistry simulation," *Nature* **574**, 215–218 (2019).
5. N. Lambert et al., "Quantum biology," *Nat. Phys.* **9**, 10–18 (2013).
6. F. Flamini, N. Spagnolo, and F. Sciarrino, "Photonic quantum information processing: a review," *Rep. Prog. Phys.* **82**, 016001 (2019).
7. J. L. O'Brien, A. Furusawa, and J. Vučković, "Photonic quantum technologies," *Nat. Photonics* **3**, 687–695 (2009).
8. A. Aspuru-Guzik and P. Walther, "Photonic quantum simulators," *Nat. Phys.* **8**, 285–291 (2012).
9. H. Tang and X. M. Jin, "Birefringence aids photonic lattice simulations," *Nat. Photonics* **16**, 178–179 (2022).
10. H. Tang et al., "Experimental two-dimensional quantum walk on a photonic chip," *Sci. Adv.* **4**, eaat3174 (2018).
11. H. Tang et al., "Experimental quantum fast hitting on hexagonal graphs," *Nat. Photonics* **12**, 754–758 (2018).
12. A. W. Harrow and A. Montanaro, "Quantum computational supremacy," *Nature* **549**, 203–209 (2017).
13. J. B. Spring et al., "Boson sampling on a photonic chip," *Science* **339**, 798–801 (2013).
14. M. A. Broome et al., "Photonic boson sampling in a tunable circuit," *Science* **339**, 794–798 (2013).
15. M. Tillmann et al., "Experimental boson sampling," *Nat. Photonics* **7**, 540–544 (2013).
16. N. Spagnolo et al., "Experimental validation of photonic boson sampling," *Nat. Photonics* **8**, 615–620 (2014).
17. H. Wang et al., "High-efficiency multiphoton boson sampling," *Nat. Photonics* **11**, 361–365 (2017).
18. J. Gao et al., "Quantum advantage with membosonsampling," *Chip* **1**, 100007 (2022).
19. D. Finke, "Quantum computing report," 2018, https://quantumcomputingreport.com/resources/education/.
20. "Quantiki: list of QC simulators," 2018, https://www.quantiki.org/wiki/list-qc-simulators.
21. R. LaRose, "Overview and comparison of gate level quantum software platforms," *Quantum* **3**, 130 (2019).
22. K. M. Svore et al., "A layered software architecture for quantum computing design tools," *Computer* **39**, 74–83 (2006).
23. D. Wecker and K. M. Svore, "L|QUi|>: a software design architecture and domain-specific language for quantum computing," arXiv:1402.4467 (2014).
24. Microsoft, "Microsoft quantum development kit," 2017, https://microsoft.com/quantum.
25. K. Svore et al., "Q#: enabling scalable quantum computing and development with a high-level DSL," in *Proc. Real World Domain Specific Languages Workshop*, Vol. 7, pp. 1–10 (2018).
26. IBM, "Quantum information science kit," 2018, https://qiskit.org/.
27. C. Rigetti et al., "Forest: an API for quantum computing in the cloud," 2018, https://www.rigetti.com/index.php/forest.
28. A. Polloreno and W. Zeng, "Pyquil license," 2018, github.com/rigetticomputing/pyquil/blob/master/LICENSE#L204.
29. D. S. Steiger, T. Häner, and M. Troyer, "ProjectQ: an open source software framework for quantum computing," *Quantum* **2**, 49 (2018).
30. S. S. Liu et al., "Q|SI⟩: a quantum programming environment," arXiv:1710.09500 (2017).
31. IBM, "IBM Q experience," 2017, https://quantumexperience.ng.bluemix.net/qx/experience

32. T. Xin et al., "Nmrcloudq: a quantum cloud experience on a nuclear magnetic resonance quantum computer," *Sci. Bull.* **63**, 17–23 (2018).
33. J. R. McClean et al., "OpenFermion: the electronic structure package for quantum computers," *Quantum Sci. Technol.* **5**, 034014 (2020).
34. J. R. Johansson, P. D. Nation, and F. Nori, "QuTiP: an open-source Python framework for the dynamics of open quantum systems," *Comput. Phys. Commun.* **183**, 1760–1772 (2012).
35. J. R. Johansson, P. D. Nation, and F. Nori, "QuTiP 2: a Python framework for the dynamics of open quantum systems," *Comput. Phys. Commun.* **184**, 1234 (2013).
36. P. Falloon, J. Rodriguez, and J. Wang, "Qswalk: a mathematica, package for quantum stochastic walks on arbitrary graphs," *Comput. Phys. Commun.* **217**, 162–170 (2017).
37. A. Glos, J. A. Miszczak, and M. Ostaszewski, "QSWalk.jl: Julia package for quantum stochastic walks analysis," *Comput. Phys. Commun.* **235**, 414–421 (2019).
38. E. Matwiejew and J. Wang, "QSW_MPI: a framework for parallel simulation of quantum stochastic walks," *Comput. Phys. Commun.* **260**, 107724 (2021).
39. H. Tang et al., "TensorFlow solver for quantum PageRank in large-scale networks," *Sci. Bull.* **66**, 120–126 (2021).
40. N. Killoran et al., "Strawberry fields: a software platform for photonic quantum computing," *Quantum* **3**, 129 (2019).
41. FeynmanPAQS, "A software for photonic analog quantum simulation provided by TuringQ Co., Ltd.," https://feynmanpaqs.turingq.com/.
42. J. Wang et al., "Gallium arsenide (GaAs) quantum photonic waveguide circuits," *Opt. Commun.* **327**, 49–55 (2014).
43. A. Politi et al., "Silica-on-silicon waveguide quantum circuits," *Science* **320**, 646–649 (2008).
44. Z. Feng et al., "Invisibility cloak printed on a photonic chip," *Sci. Rep.* **6**, 28527 (2016).
45. A. Szameit et al., "Control of directional evanescent coupling in fs laser written waveguides," *Opt. Express* **15**, 1579–1587 (2007).
46. A. Crespi et al., "Integrated multimode interferometers with arbitrary designs for photonic boson sampling," *Nat. Photonics* **7**, 545–549 (2013).
47. Z. Chaboyer et al., "Tunable quantum interference in a 3D integrated circuit," *Sci. Rep.* **5**, 9601 (2015).
48. T. Giorgino, A. Laio, and A. Rodriguez, "Metagui 3: a graphical user interface for choosing the collective variables in molecular dynamics simulations," *Comput. Phys. Commun.* **217**, 204–209 (2017).
49. M. Umansky and D. Weihs, "Novel algorithm and matlab-based program for automated power law analysis of single particle, time-dependent mean-square displacement," *Comput. Phys. Commun.* **183**, 1783–1792 (2012).
50. S. Vergaraperez and M. Marucho, "MPBEC, a matlab program for biomolecular electrostatic calculations," *Comput. Phys. Commun.* **198**, 179–194 (2016).
51. Y. Aharonov, L. Davidovich, and N. Zagury, "Quantum random walks," *Phys. Rev. A* **48**, 1687 (1993).
52. A. M. Childs, E. Farhi, and S. Gutmann, "An example of the difference between quantum and classical random walks," *Quantum Inf. Process.* **1**, 35–43 (2002).
53. A. Ambainis, "Quantum walks and their algorithmic applications," *Int. J. Quantum Inf.* **1**, 507–518 (2003).
54. N. Shenvi, J. Kempe, and K. B. Whaley, "Quantum random-walk search algorithm," *Phys. Rev. A* **67**, 052307 (2003).
55. A. M. Childs and J. Goldstone, "Spatial search by quantum walk," *Phys. Rev. A* **70**, 022314 (2004).
56. O. Mülken and A. Blumen, "Continuous-time quantum walks: models for coherent transport on complex networks," *Phys. Rep.* **502**, 37–87 (2011).
57. Y. Wang et al., "Quantum topological boundary states in quasi-crystal," *Adv. Mater.* **31**, 1905624 (2019).
58. Y. Wang et al., "Direct observation of topology from single-photon dynamics on a photonic chip," *Phys. Rev. Lett.* **122**, 193903 (2019).

59. H. Tang et al., "Teaching quantum information technologies and a practical module for online and offline undergraduate students," arXiv: 2112.06548 (2021).
60. J. D. Whitfield, C. A. Rodríguez-Rosario, and A. Aspuru-Guzik, "Quantum stochastic walks: a generalization of classical random walks and quantum walks," *Phys. Rev. A* **81**, 022323 (2010).
61. F. Caruso et al., "Fast escape of a quantum walker from an integrated photonic maze," *Nat. Commun.* **7**, 11682 (2016).
62. H. Tang et al., "Generating Haar-uniform randomness using stochastic quantum walks on a photonic chip," *Phys. Rev. Lett.* **128**, 050503 (2022).
63. L. Banchi, D. Burgarth, and M. J. Kastoryano, "Driven quantum dynamics: will it blend?" *Phys. Rev. X* **7**, 041015 (2017).
64. C. K. Hong, Z. Y. Ou, and L. Mandel, "Measurement of subpicosecond time intervals between two photons by interference," *Phys. Rev. Lett.* **59**, 2044–2046 (1987).
65. A. Peruzzo et al., "Quantum walks of correlated photons," *Science* **329**, 1500–1503 (2010).
66. J. Gao et al., "Non-classical photon correlation in a two-dimensional photonic lattice," *Opt. Express* **24**, 12607–12616 (2016).
67. L. Sansoni et al., "Two-particle Bosonic-Fermionic quantum walk via integrated photonics," *Phys. Rev. Lett.* **108**, 010502 (2012).
68. J. C. Matthews et al., "Observing fermionic statistics with photons in arbitrary processes," *Sci. Rep.* **3**, 1719 (2013).
69. Y. Wang et al., "Topological protection of two-photon quantum correlation on a photonic chip," *Optica* **6**, 955–960 (2019).
70. H. J. Ryser, *Combinatorial Mathematics (Carus Mathematical Monographs)*, Vol. **14**, Mathematical Association of America, New York (1963).
71. D. G. Glynn, "The permanent of a square matrix," *Eur. J. Comb.* **31**, 1887–1891 (2010).
72. D. G. Glynn, "Permanent formulae from the Veronesean," *Des. Codes Cryptogr.* **68**, 39–47 (2013).
73. A. Nijenhuis and H. S. Wilf, *Combinatorial Algorithms: For Computers and Calculators*, 2nd ed., Academic Press, New York (1978).
74. S. Aaronson and A. Arkhipov, "The computational complexity of linear optics," *Theory Comput.* **9**, 143–252 (2013).
75. M. Reck et al., "Experimental realization of any discrete unitary operator," *Phys. Rev. Lett.* **73**, 58–61 (1994).
76. W. R. Clements et al., "Optimal design for universal multiport interferometers," *Optica* **3**, 1460–1465 (2016).

Hao Tang is currently an associate professor at the School of Physics and Astronomy, Shanghai Jiao Tong University. She focuses on the experimental and theoretical exploration of quantum computing and quantum simulation on integrated photonics, as well as the applications of quantum computing for machine learning, finance, and optimization tasks. She teaches undergraduate and postgraduate courses, including quantum optics and quantum information technologies.

Xiao-Jun Xu works as a quantum algorithm scientist at TuringQ. He received his master's degree on physics from Imperial College London in UK. He led the project of implementing FeynmanPAQS on the cloud platform. Current research interest lies in quantum natural language processing algorithms.

Yanyan Zhu was a visiting student at Shanghai Jiao Tong University, where he worked with Hao Tang for implementing the initial version of FeynmanPAQS on MATLAB. He is currently a PhD student in physics at the University of California, Los Angeles.

Jun Gao was a PhD student at Shanghai Jiao Tong University, where he contributed to the construction of the modules related to multiple quantum walks and boson sampling in FeynmanPAQS. He is currently a postdoctoral researcher at KTH Royal Institute of Technology in Stockholm.

Xuan Chen is a master's student at Shanghai Jiao Tong University. He contributed to the data analysis and manuscript writing for this work.

Markus Lee was a visiting student to Shanghai Jiao Tong University, where he worked with Hao Tang for improving the optimization algorithms on permanent calculation for FeynmanPAQS. He is a graduate student at Cambridge University.

Pengcheng Lai was an undergraduate student at Shanghai Jiao Tong University, where he contributed to the data analysis and manuscript preparation for this work. He is currently a PhD student on quantum information at Tsinghua University.

Xian-Min Jin is a professor at Shanghai Jiao Tong University (SJTU) and the director of the Center for Integrated Quantum Information Technologies (IQIT). His interests cover a broad spectrum ranging from quantum computing, quantum communication, and quantum metrology with special focus on the subject of building large-scale quantum systems via integrated photonics and quantum memory.