Training Gaussian boson sampling distributions

Leonardo Banchi,[1,2] Nicolás Quesada,[3] and Juan Miguel Arrazola[3]
[1]*Department of Physics and Astronomy, University of Florence, via G. Sansone 1, I-50019 Sesto Fiorentino (FI), Italy*
[2]*INFN Sezione di Firenze, via G.Sansone 1, I-50019 Sesto Fiorentino (FI), Italy*
[3]*Xanadu, Toronto, Ontario, Canada M5G 2C8*

(Received 21 April 2020; accepted 4 June 2020; published 15 July 2020)

Gaussian boson sampling (GBS) is a near-term platform for photonic quantum computing. Applications have been developed that rely on directly programming GBS devices, but the ability to train and optimize circuits has been a key missing ingredient for developing new algorithms. In this work, we derive analytical gradient formulas for the GBS distribution, which can be used for training devices using standard methods based on gradient descent. We introduce a parametrization of the distribution that allows the gradient to be estimated by sampling from the same device that is being optimized. In the case of training using a Kullback-Leibler divergence or log-likelihood cost function, we show that gradients can be computed classically, leading to fast training. We illustrate these results with numerical experiments in stochastic optimization and unsupervised learning. As a particular example, we introduce the variational Ising solver, a hybrid algorithm for training GBS devices to sample ground states of a classical Ising model with high probability.

DOI: 10.1103/PhysRevA.102.012417

I. INTRODUCTION

Gaussian boson sampling (GBS) is a special-purpose platform for photonic quantum computing. It was proposed as a method to build photonic devices capable of performing tasks that are intractable for classical computers [1,2]. Since then, several quantum algorithms based on GBS have been introduced [3], with applications to graph optimization [4–6], graph similarity [7,8], point processes [9], and quantum chemistry [10,11]. These algorithms rely on strategies to carefully program GBS devices, typically by encoding a suitable symmetric matrix into the GBS distribution.

Yet many quantum algorithms rely on the ability to train the parameters of quantum circuits [12], a strategy inspired by the success of neural networks in machine learning. Examples include quantum approximate optimization [13,14], variational quantum eigensolvers [15], quantum feature embeddings [16,17], and quantum classifiers [18]. Training is often performed by evaluating gradients of a cost function with respect to circuit parameters, then employing gradient-based optimization methods [19,20]. Deriving similar methods to train GBS devices is a missing piece for unlocking new algorithms, particularly in machine learning and optimization.

In this work, we derive analytic gradients of the GBS distribution, which can be used to train the device using gradient-based optimization. We derive a general gradient formula that can be evaluated in simulators, but is not always accessible from hardware. We then introduce a specific parametrization of the GBS distribution that expresses the gradient as an expectation value from the same distribution. Such gradients can be evaluated by sampling from the same device that is being optimized. Using this parametrization, we show that for Kullback-Leibler divergence or log-likelihood cost functions, analytical gradients can be evaluated efficiently using classical methods, leading to fast training. We illustrate these results with numerical experiments in stochastic optimization and unsupervised learning.

As a specific application for our training scheme, we introduce the variational Ising solver (VIS). In this algorithm, as in the variational quantum eigensolver [15], a parametric circuit is optimized to approximate the ground state of a Hamiltonian. Similarly to the quantum approximate optimization algorithm [13,14,21], we focus on combinatorial optimization problems where the Hamiltonian can be expressed as a classical Ising model. Both the variational eigensolver and the quantum approximate optimization algorithm are tailored for near-term qubit-based quantum computers, while VIS is tailored for near-term GBS devices. We use a parametric circuit that creates a particular Gaussian state, and iteratively update the Gaussian state using a gradient-based hybrid strategy based on outcomes coming from either photon-number-resolving detectors or threshold detectors.

The paper is organized as follows. In Sec. II, we provide a short review of GBS. In Sec. III, we review the stochastic optimization and unsupervised learning tasks covered in this work. In Sec. IV we present the main theoretical contributions of this paper: we derive the analytical gradient formulas and introduce a suitable parametrizations of the GBS distribution that enables the measurement of the gradient in a quantum device. Finally, in Sec. V, we provide numerical examples demonstrating the ability of VIS to approximate the solution to certain combinatorial optimization problems, and the ability to train GBS distributions using classical gradient formulas. Conclusions are drawn in Sec. VI. A summary of the main results is shown in Table I.

II. GAUSSIAN BOSON SAMPLING

In quantum optics, the systems of interest are optical modes of the quantized electromagnetic field. The quantum

TABLE I. Summary of the main results.

General gradient formula	Eq. (21)
Efficiently measurable gradients	Eq. (29)
Classically computable gradients	Eq. (33)

state of m modes can be specified by its Wigner function $W(\boldsymbol{q},\boldsymbol{p})$, where $\boldsymbol{q},\boldsymbol{p}\in\mathbb{R}^m$ are known, respectively, as the position and momentum quadrature vectors. Gaussian states are characterized by having a Wigner function that is Gaussian. Consequently, Gaussian states can be completely specified by their first and second moments, namely two m-dimensional vectors of means $\bar{\boldsymbol{q}},\bar{\boldsymbol{p}}$ and a covariance matrix Σ. For our purposes, it is more convenient to work with the complex-normal random variable $\boldsymbol{\alpha}=\frac{1}{\sqrt{2\hbar}}(\boldsymbol{q}+i\,\boldsymbol{p})$ that has mean $\bar{\boldsymbol{\alpha}}=\frac{1}{\sqrt{2\hbar}}(\bar{\boldsymbol{q}}+i\,\bar{\boldsymbol{p}})$ and covariance matrix V.

When measuring a Gaussian state ρ in the photon-number basis, the joint probability of observing n_i photons in mode i, is given by $P_{\mathcal{A}}(\bar{n})=\langle\bar{n}|\rho|\bar{n}\rangle$, where $\bar{n}=(n_1,\ldots,n_m)$ is the vector of outcomes, $|\bar{n}\rangle=|n_1,\ldots,n_m\rangle$ is a product of Fock states, and the matrix $\mathcal{A}$, defined below, uniquely identifies the state ρ. As shown in Ref. [1], the above probability can be written as

$$P_{\mathcal{A}}(\bar{n})=\frac{1}{\mathcal{Z}}\frac{\mathrm{Haf}(\mathcal{A}_{\bar{n}\oplus\bar{n}})}{n_1!\cdots n_m!},\tag{1}$$

with the following definitions:

$$\mathcal{A}=X(\mathbb{1}-(V+\mathbb{1}/2)^{-1}),\tag{2}$$

$$X:=\begin{bmatrix}0&\mathbb{1}\\ \mathbb{1}&0\end{bmatrix},\tag{3}$$

$$\frac{1}{\mathcal{Z}}:=\sqrt{\det(\mathbb{1}-X\mathcal{A})}.\tag{4}$$

We now explain the notation used in Eq. (1). For a matrix $\mathcal{B}\in\mathbb{C}^{m\times m}$ the notation $\mathcal{B}_{\bar{n}}$ indicates the matrix constructed from $\mathcal{B}$ as follows. If $n_i=0$, the ith row and column are deleted from $\mathcal{B}$. If $n_i>0$, the ith row and column are repeated n_i times. In the case of $\mathcal{A}\in\mathbb{C}^{2m\times 2m}$ as in Eq. (1), this procedure is performed with the vector $\bar{n}\oplus\bar{n}=(n_1,\ldots,n_m,n_1,\ldots,n_m)$.

The Hafnian of a $2m\times 2m$ matrix $\mathcal{A}$ is defined as [22]

$$\mathrm{Haf}(\mathcal{A})=\sum_{\mu\in\mathrm{PMP}(2m)}\prod_{(i,j)\in\mu}\mathcal{A}_{i,j},\tag{5}$$

where $\mathcal{A}_{i,j}$ is the (i,j) entry of the symmetric matrix $\mathcal{A}=\mathcal{A}^T$ and PMP is the set of perfect matching permutations, the possible ways of partitioning the set $\{1,\ldots,2m\}$ into disjoints subsets of size two. The Hafnian is ♯P-hard to approximate for worst-case instances [23] and the runtime of the best known algorithms for computing Hafnians of arbitrary matrices scales exponentially with m [24]. Using techniques from Ref. [25], it has been argued that sampling from a GBS distribution cannot be done in classical polynomial time unless the polynomial hierarchy collapses to third level [1].

For pure Gaussian states, it holds that $\mathcal{A}=A\oplus A^*$ and $A\in\mathbb{C}^{m\times m}$ is a symmetric matrix that can be decomposed as

$$A=U\,\mathrm{diag}(\lambda_1,\ldots,\lambda_{\mathrm{m}})\mathrm{U}^{\mathrm{T}},\tag{6}$$

where $0\leqslant\lambda_i<1$. The probability distribution is then

$$P_{\mathcal{A}}(\bar{n})=\frac{1}{\mathcal{Z}}\frac{|\mathrm{Haf}(A_{\bar{n}})|^2}{n_1!\cdots n_m!}.\tag{7}$$

The mean photon number is given by

$$\langle n\rangle=\sum_{i=1}^{m}\frac{\lambda_i^2}{1-\lambda_i^2},\tag{8}$$

which can be adjusted by rescaling the matrix $A\to cA$ for an appropriate parameter $c>0$.

III. TRAINING THE GBS DISTRIBUTION

In this section, we briefly review the training tasks considered in this work: stochastic optimization and unsupervised learning. Here and throughout the paper, given a vector of parameters $\theta=(\theta_1,\theta_2,\ldots,\theta_d)$, we use ∂_θ as a shorthand for the gradient $(\frac{\partial}{\partial\theta_1},\frac{\partial}{\partial\theta_2},\ldots,\frac{\partial}{\partial\theta_d})$. Similarly, we employ ∂_{θ_j} to denote $\frac{\partial}{\partial\theta_j}$.

A. Stochastic optimization

It has been recently shown that certain optimization problems in graph theory can be solved by sampling solutions from a properly configured GBS device [4,6]. This was made possible by encoding graphs into the GBS distribution [26] and exploiting the fact that this distribution outputs, with high probability, photon configurations $\bar{n}$ that have a large Hafnian $\mathrm{Haf}(\mathcal{A}_{\bar{n}})$.

We consider the more general problem of optimizing the GBS distribution directly from the samples, without requiring a theoretical scheme to optimally program the device. Consider a function $H(\bar{n})$ that associates a cost to the set of positive integers n_k sampled from the GBS distribution. Fixing the symmetric matrix $\mathcal{A}=\mathcal{A}(\theta)$ where θ is a set of variational parameters, the cost is given by

$$C(\theta)=\mathbb{E}_{\bar{n}\sim P_{\mathcal{A}(\theta)}(\bar{n})}[H(\bar{n},\theta)]\equiv\sum_{\bar{n}}H(\bar{n})P_{\mathcal{A}(\theta)}(\bar{n}).\tag{9}$$

Our goal is to optimize the Gaussian state, encapsulated by the $2M\times 2M$ matrix $\mathcal{A}(\theta)$, in order to minimize the cost function. Suppose that there are certain choices of the parametrization such that the gradient $\partial_\theta C(\theta)$ can be either efficiently computed numerically or estimated via sampling on a physical device. In such cases it is possible to minimize the average cost $C(\theta)$ using the update rule

$$\theta\to\theta-\eta\,\partial_\theta C(\theta),\tag{10}$$

where $\eta>0$ is a learning rate. Alternatively, other gradient-based optimization algorithms can be used [27,28].

We show that, for some parametrizations of the Gaussian state, it is possible to write

$$\partial_\theta C(\theta)=\mathbb{E}_{\bar{n}\sim P_{\mathcal{A}'(\theta)}(\bar{n})}[G(\bar{n})],\tag{11}$$

namely it is possible to write the gradient of $C(\theta)$ as an expectation value of a different function $G(\bar{n})$ with respect to a possibly different GBS distribution $P_{\mathcal{A}'(\theta)}(\bar{n})$. A GBS device can then be used to sample from this new distribution and

obtain an empirical gradient

$$\partial_\theta C(\theta) \approx \frac{1}{T}\sum_{t=1}^{T} G(\bar{n}^{(t)}), \tag{12}$$

from the samples $\{\bar{n}^{(1)}, \ldots, \bar{n}^{(T)}\}$. The parameters are then iteratively updated using the gradient estimate

$$\theta \to \theta - \eta \frac{1}{T}\sum_{t=1}^{T} G(\bar{n}^{(t)}). \tag{13}$$

B. Unsupervised learning

In a standard unsupervised learning scenario, data are assumed to be sampled from an unknown probability distribution $Q(\bar{n})$, and a common goal is to reproduce the statistics of the data. This is done by introducing a parametric approximation $P(\bar{n})$ of the unknown distribution $Q(\bar{n})$, and then iteratively updating the parameters in such a way that the data sequence matches the samples from the model distribution $P(\bar{n})$. Training can be performed by minimizing a suitably chosen cost function, such as the Kullback-Leibler (KL) divergence

$$D_{KL}[Q, P] = \sum_x Q(x) \ln \frac{Q(x)}{P(x)}. \tag{14}$$

Thanks to the results of this paper, we can use the GBS distribution $P_{\mathcal{A}(\theta)}(\bar{n})$ as a model. In this distribution, the parameters are those entering into the covariance matrix of the Gaussian state, and hence into the $\mathcal{A}$ matrix via Eq. (2). We call θ the set of parameters that we are allowed to vary in the Gaussian state. These can be either physical parameters such as squeezing or multimode transmissivities, or can be mathematical parameters that depend on the physical ones, possibly in a complex way. The KL divergence between the unknown data distribution and a GBS distribution with parameters θ is

$$C_{\text{data}}(\theta) = D_{KL}[P_{\text{data}}(\bar{n}), P_{\mathcal{A}(\theta)}(\bar{n})]. \tag{15}$$

Its gradient is given by

$$\begin{aligned}\partial_\theta C_{\text{data}}(\theta) &= -\sum_{\bar{n}} P_{\text{data}}(\bar{n})\, \partial_\theta \ln P_{\mathcal{A}(\theta)}(\bar{n}) \\ &= \mathbb{E}_{\bar{n}\sim P_{\text{data}}}[-\partial_\theta \ln P_{\mathcal{A}(\theta)}(\bar{n})].\end{aligned} \tag{16}$$

In practice, instead of an explicit expression for the data distribution $P_{\text{data}}(\bar{n})$, a training set $\{\bar{n}^{(1)}, \ldots, \bar{n}^{(T)}\}$ is provided. This is interpreted as a collection of samples from the data distribution. Averages are defined with respect to these samples:

$$\mathbb{E}_{\bar{n}\sim P_{\text{data}}}[-\partial_\theta \ln P_{\mathcal{A}(\theta)}(\bar{n})] = -\frac{1}{T}\sum_{t=1}^{T} \partial_\theta \ln P_{\mathcal{A}(\theta)}(\bar{n}^{(t)}). \tag{17}$$

We show that for certain choices of the parametrization, it is possible to compute the derivatives $\partial_\theta \ln P_{\mathcal{A}(\theta)}(\bar{n})$, allowing for an efficient training of the GBS distribution.

IV. ANALYTICAL GRADIENTS

In this section we obtain gradient formulas for the GBS distribution, which represent some of the main results of this paper. We first derive a general formula expressing the gradient for arbitrary parametrizations. Then we proceed by introducing a strategy, the WAW parametrization, which allows gradients for arbitrary cost functions to be computed as expectation values over GBS distributions. Moreover, for specific cost functions, we show that gradients can be efficiently calculated classically. Finally, we extend our gradient formulas to GBS with threshold detectors, and derive other algorithms based on reparametrization or on the projected subgradient method, which have different applicability.

A. General formula

We now derive the gradient of the GBS distribution. Thanks to Eq. (1), $P_{\mathcal{A}}(\bar{n}) = \frac{1}{\mathcal{Z}}\frac{\text{Haf}(\mathcal{A}_{\bar{n}\oplus\bar{n}})}{n_1!\cdots n_m!}$ can be expressed as

$$\partial_\theta P_{\mathcal{A}}(\bar{n}) = \left(\partial_\theta \frac{1}{\mathcal{Z}}\right)\frac{\text{Haf}(\mathcal{A}_{\bar{n}\oplus\bar{n}})}{n_1!\cdots n_m!} + \frac{1}{\mathcal{Z}}\frac{\partial_\theta \text{Haf}(\mathcal{A}_{\bar{n}\oplus\bar{n}})}{n_1!\cdots n_m!}. \tag{18}$$

Note that in this section we avoid writing the explicit dependence of $\mathcal{A}$ on θ to simplify the notation. As we find in Appendix A, the derivatives in Eq. (18) can be calculated analytically and the result is

$$\partial_\theta\left(\frac{1}{\mathcal{Z}}\right) = -\frac{1}{2}\text{Tr}\left[\frac{1}{\mathcal{Z}}\frac{\partial_\theta \mathcal{A}}{\text{X} - \mathcal{A}}\right], \tag{19}$$

$$\partial_\theta \text{Haf}(\mathcal{A}_{\bar{n}\oplus\bar{n}}) = \sum_{i\neq j}^{2N} (\partial_\theta \mathcal{A}_{\bar{n}})_{ij}\, \text{Haf}\left(\mathcal{A}^{[i,j]}_{\bar{n}\oplus\bar{n}}\right), \tag{20}$$

where $2N$, with $N = \sum_k n_k$, is the dimension of the matrix $\mathcal{A}_{\bar{n}\oplus\bar{n}}$. The submatrix $A^{[i,j]}_{\bar{n}\oplus\bar{n}}$ is constructed from $A_{\bar{n}\oplus\bar{n}}$ by removing rows (i, j) and columns (i, j). Combining these results gives a general formula for the gradient of the GBS distribution:

$$\begin{aligned}\partial_\theta P_{\mathcal{A}}(\bar{n}) = &-\frac{1}{2}\text{Tr}\left[\frac{\partial_\theta \mathcal{A}}{X - \mathcal{A}}\right] P_{\mathcal{A}}(\bar{\text{n}}) \\ &+ \frac{1}{\mathcal{Z}}\frac{1}{n_1!\cdots n_m!}\sum_{i\neq j}^{2N} (\partial_\theta \mathcal{A}_{\bar{n}\oplus\bar{n}})_{ij}\, \text{Haf}\left(\mathcal{A}^{[i,j]}_{\bar{n}\oplus\bar{n}}\right).\end{aligned} \tag{21}$$

From the above equation we can also obtain the derivative of the cost function $C(\theta)$ in Eq. (9):

$$\begin{aligned}\partial_\theta C(\theta) &= \sum_{\bar{n}} H(\bar{n}) \partial_\theta P_{\mathcal{A}}(\bar{n}) \\ &= -\frac{1}{2}\mathbb{E}_{\bar{n}\sim P(\bar{n})}\left[\text{Tr}\left(\frac{\text{H}(\bar{\text{n}})}{\text{X} - \mathcal{A}}\partial_\theta \mathcal{A}\right)\right] \\ &\quad + \frac{\mathcal{Z}^{-1}}{n_1!\cdots n_m!}\sum_{\bar{n}} H(\bar{n}) \sum_{i\neq j}^{2N} (\partial_\theta \mathcal{A}_{\bar{n}\oplus\bar{n}})_{ij} \text{Haf}\left(\mathcal{A}^{[i,j]}_{\bar{n}\oplus\bar{n}}\right).\end{aligned} \tag{22}$$

The generalization to a θ-dependent cost function is straightforward.

Equation (21) represents the first main result of this work. Nonetheless, the quantities $\mathrm{Haf}(\mathcal{A}^{[i,j]}_{\bar{n}\oplus\bar{n}})$ are not proportional to probabilities unless $i = j + N$ or $i + N = j$ [29], which makes it challenging to express gradients as expectations over the GBS distribution. It is currently an open question to define a general strategy to estimate those quantities in GBS-like experiments. Nevertheless, as we derive next, it is possible to cast gradients as expectation values for carefully chosen parametrizations of the matrix $\mathcal{A}$.

B. WAW parametrization

We focus on the pure-state case, $\mathcal{A} = A \oplus A^*$, and replace the matrix A with

$$A_W = WAW, \tag{23}$$

where $W_{kj} = \sqrt{w_k}\delta_{kj}$ and $w_k \geqslant 0$. The generalization to mixed states is studied in Appendix B. The symmetric matrix A is kept fixed and the weights w_k of the diagonal weight matrix W are trainable parameters. The matrix A serves as a model for the distribution and W encodes its free parameters. We refer to this strategy as the WAW parametrization, in reference to Eq. (23). Similar parametrizations have been successfully used for training determinantal point processes in machine learning [30]. The WAW parametrization represents a mathematical abstraction to conveniently parametrize the Gaussian state and evaluate its gradient, as we will see. For any Gaussian state parametrized as in Eq. (23), we may get the physical parameters via standard decompositions [31,32].

It is important that when updating parameters, the matrix A_W always corresponds to a physical Gaussian state. As shown in Appendix B, if A is a valid matrix with singular values contained in $[0,1)$, A_W is also valid whenever $0 \leqslant w_k \leqslant 1$. This condition can be enforced via reparametrization. The condition $w_k \geqslant 0$ is necessary to avoid introducing imaginary numbers, while $w_k \leqslant 1$ is sufficient to get a valid Gaussian state. However, enforcing $w_k \leqslant 1$ at each step is too restrictive, as the matrix elements in Eq. (23) can only diminish. As we will show in Sec. IV F, a more general strategy consists in allowing any positive value of w_k, and then projecting the matrix A_W to the closest physical state. With this in mind, one of the strategies we consider is to express $w_k(\theta)$ as

$$w_k(\theta) = \exp(-\theta^T f^{(k)}), \tag{24}$$

where $f^{(k)} = (f_1^{(k)}, f_2^{(k)}, \ldots, f_d^{(k)})$ is a d-dimensional vector, and $\theta = (\theta_1, \theta_2, \ldots, \theta_d)$ is a vector of parameters. The condition $0 \leqslant w_k \leqslant 1$ can be satisfied by enforcing $\theta^T f^{(k)} \geqslant 0$ for all k.

The Hafnian of A_W can be factorized into independent contributions from A and W [23]:

$$\mathrm{Haf}(A_W) = \mathrm{Haf}(A)\det(W). \tag{25}$$

Inserting the above in Eq. (7) gives

$$P_{A,W}(\bar{n}) = \frac{1}{\mathcal{Z}}\,\mathrm{Haf}(A_{\bar{n}})^2 \prod_{i=1}^{m} \frac{w_i^{n_i}}{n_i!}, \tag{26}$$

where the notation $P_{A,W}(\bar{n})$ is used as a reminder that the distribution depends on both A and W. Since the Hafnian is independent of the parameters w_k, it is possible to express the derivative of the distribution in terms of GBS probabilities. Explicit calculations are done in Appendix A and the result is

$$\partial_{w_k} P_{A,W}(\bar{n}) = \frac{n_k - \langle n_k\rangle}{w_k} P_{A,W}(\bar{n}), \tag{27}$$

where $\langle n_k\rangle$ is the average number of photons in mode k, which can be calculated directly from the covariance matrix V:

$$\langle n_k\rangle = \frac{V_{k,k} + V_{k+m,k+m} - 1}{2}. \tag{28}$$

The above can be generalized with a reparametrization of the weights, namely $w_k = w_k(\theta)$, so by the chain rule

$$\partial_\theta P_{A,W}(\bar{n}) = \sum_{k=1}^{m} (n_k - \langle n_k\rangle)\, P_{A,W}(\bar{n})\partial_\theta \ln w_k. \tag{29}$$

Equation (29) represents the second main result of this work. From Eq. (29) it is also possible to calculate the gradient of cost functions

$$\partial_\theta C(\theta) = \mathbb{E}_{\bar{n}\sim P_{A,W}(\bar{n})}\left[\sum_{k=1}^{m} H(\bar{n})\,(n_k - \langle n_k\rangle)\partial_\theta \ln w_k\right]. \tag{30}$$

Therefore, gradients can be obtained by sampling directly from the distribution to estimate this expectation value.

C. Computing gradients classically

We now show that the gradient of the KL divergence is straightforward to compute with the WAW parametrization. Indeed since $\partial_\theta \ln P = \frac{\partial_\theta P}{P}$, from Eq. (29) the gradient can be written as

$$\begin{aligned}\partial_\theta C_{\mathrm{data}}(\theta) &= -\mathbb{E}_{\bar{n}\sim P_{\mathrm{data}}}\left[\sum_{k=1}^{m}(n_k - \langle n_k\rangle)\partial_\theta \ln w_k\right] \\ &= -\sum_{k=1}^{m}(\langle n_k\rangle_{\mathrm{data}} - \langle n_k\rangle_{\mathrm{GBS}})\partial_\theta \ln w_k,\end{aligned} \tag{31}$$

where we introduce the notation $\langle n_k\rangle_{\mathrm{GBS}}$ to distinguish the average photon number of Eq. (28) from the expectation value $\langle n_k\rangle_{\mathrm{data}}$, defined as $\langle n_k\rangle_{\mathrm{data}} = \mathbb{E}_{\bar{n}\sim P_{\mathrm{data}}}[n_k]$, or alternatively as

$$\langle n_k\rangle_{\mathrm{data}} = \frac{1}{T}\sum_{t=1}^{T} n_k^{(t)}, \tag{32}$$

when the data distribution is defined in terms of a given data set $\{\bar{n}^{(1)}, \ldots, \bar{n}^{(T)}\}$. When using the reparametrization of Eq. (24), the gradient is given by

$$\partial_\theta C_{\mathrm{data}}(\theta) = \sum_{k=1}^{m}(\langle n_k\rangle_{\mathrm{GBS}} - \langle n_k\rangle_{\mathrm{data}})f^{(k)}. \tag{33}$$

Equation (33) is the third main result of this work. This expression can be further simplified by defining

$$F_{\mathrm{data}} := \sum_{k=1}^{m}\langle n_k\rangle_{\mathrm{data}} f^{(k)}, \tag{34}$$

which depends only on the data and the choice of vectors f. We then have

$$\partial_\theta C_{\text{data}}(\theta) = \sum_{k=1}^{m} \langle n_k \rangle_{\text{GBS}} f^{(k)} - F_{\text{data}}. \tag{35}$$

Once F_{data} has been calculated, only m terms $\langle n_k \rangle_{\text{GBS}} f^{(k)}$ need to be computed to obtain the gradient. This can be done in $O(m)$ time on a classical computer by using Eq. (28).

Finally, we note that the log-likelihood function

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \ln P_{A,W}(\bar{n}^{(t)}), \tag{36}$$

which is also often used in unsupervised learning [30], is related to the cost function of Eq. (15) by the formula

$$C_{\text{data}}(\theta) = \frac{1}{T} \sum_{t=1}^{T} \ln \frac{1/T}{P_{A,W}(\bar{n}^{(t)})} = -\frac{\mathcal{L}(\theta)}{T} - \ln T, \tag{37}$$

and therefore

$$\partial_\theta \mathcal{L}(\theta) = -T\, \partial_\theta C_{\text{data}}(\theta), \tag{38}$$

meaning that the gradient formula of Eq. (35) can be used to perform training for either of these two cost functions.

D. GBS with threshold detectors

Threshold detectors do not resolve photon number; they click whenever one or more photons are observed. Mathematically, the effect of this detection on the GBS distribution can be described by the bit string $\bar{x} = (x_1, x_2, \ldots, x_m)$, obtained from the output $\bar{n}$ by the mapping

$$x_k(\bar{n}) = \begin{cases} 0 & \text{if } n_k = 0, \\ 1 & \text{if } n_k > 0. \end{cases} \tag{39}$$

The GBS distribution with threshold detectors was found in Ref. [33] as

$$P_{A,W}(\bar{x}) = \frac{1}{\mathcal{Z}} \text{Tor}(X\mathcal{A}_W), \tag{40}$$

where $\mathcal{A}_W = A_W \oplus A_W$ and $\text{Tor}(\cdot)$ is the Torontonian function. This distribution does not factorize under the WAW parametrization as in Eq. (25), which makes it challenging to compute exact gradients. Instead, we note that whenever $\langle n_k \rangle \ll 1$ it holds that

$$\langle x_k \rangle_{\text{GBS}} = \frac{\langle n_k \rangle}{\langle n_k \rangle + 1} \simeq \langle n_k \rangle, \tag{41}$$

where we have implicitly defined $\langle x_k \rangle_{\text{GBS}}$, the probability of detecting at least one photon in mode k. The latter can be computed efficiently as [6]

$$\langle x_k \rangle_{\text{GBS}} = 1 - \frac{1}{\sqrt{\det(Q^{(k)})}}, \tag{42}$$

where $Q = (\mathbb{1} - X\mathcal{A})^{-1}$ and $Q^{(k)}$ is the submatrix obtained by keeping the $(k, k+m)$ rows and columns of Q. Under this approximation, and assuming $\langle x_k \rangle \approx \langle n_k \rangle$, Eqs. (30) and (33) can be updated to obtain

$$\partial_\theta C(\theta) \approx \mathbb{E}_{\bar{x} \sim \text{Tor}} \left[\sum_{k=1}^{m} H(\bar{x})\, \partial_\theta \ln w_k (x_k - \langle x_k \rangle_{\text{GBS}}) \right], \tag{43}$$

$$\partial_\theta C_{\text{data}}(\theta) \approx \sum_{k=1}^{m} [\langle x_k \rangle_{\text{GBS}} - \langle x_k \rangle_{\text{data}}] f^{(k)}, \tag{44}$$

where $\bar{x} \sim \text{Tor}$ is a shorthand notation to say that $\bar{x}$ are sampled from Eq. (40), and expectations $\langle x_k \rangle_{\text{data}}$ are taken with respect to the data distribution. The opposite limit, $\langle n_k \rangle \gg 1$ is studied in Appendix C. A better approximation to the gradient in this limit is given by

$$\partial_\theta C(\theta) \approx \mathbb{E}_{\bar{x} \sim \text{Tor}} \left[H(\bar{x}) \sum_{k=1}^{m} v_k(\bar{x}) \partial_\theta \ln w_k \right], \tag{45}$$

where

$$v_k(\bar{x}) = \max \{ \langle n_k \rangle (x_k - 1), x_k - \langle n_k \rangle \}. \tag{46}$$

As we demonstrate in the Sec. V, these gradient formulas work sufficiently well in practice for training GBS distributions. These approximate formulas are also a biased estimator of the gradient, but it has been shown that convergence is expected even with some biased gradient estimators [34].

E. Quantum reparametrization

In this section we discuss an alternative training mechanism with a fixed Gaussian state. Before considering the application to GBS, we recall the general problem of stochastic optimization, namely to minimize the average value of a quantity that is estimated from sampled data. We assume that the data are distributed with a parametric probability distribution $p_\theta(x)$ and the quantity to minimize is

$$C(\theta) = \mathbb{E}_{x \sim p_\theta(x)}[f(x, \theta)], \tag{47}$$

where $f(x, \theta)$ is an arbitrary function that depends on the samples x and possibly on the parameters θ. The data distribution $p_\theta(x)$ changes if we update the parameters via training, so at each step a certain number of new samples must be obtained. Reparametrization is a common strategy [35] to get an equivalent optimization problem to Eq. (47) with a θ-independent distribution. It was recently employed to train generative models using quantum annealers [36]. As shown in Ref. [35], reparametrization is possible when a mapping $(x, \theta) \to z$ exists such that

$$p_\theta(x)dx = q(z)dz, \tag{48}$$

with a new probability distribution $q(z)$. With the above definition we can write

$$C(\theta) = \mathbb{E}_{z \sim q(z)}[f(x(z, \theta), \theta)], \tag{49}$$

where data comes from a fixed, θ-independent distribution. When the cost can be expressed this way, it is possible to get a fixed number of samples before training and optimize $C(\theta)$ without having to generate new samples after each step. Moreover, gradients obtained from Eq. (49) typically have a lower variance.

We show that this strategy can be applied to the WAW parametrization because of the explicit form of Eq. (26). More general parametrizations are studied in Appendix D. Indeed, the cost function can be written in an alternative form where the weights are shifted away from the distribution as

$$C(\theta) = \sum_{\bar{n}} H(\bar{n}) P_{A,W}(\bar{n}) = \sum_{\bar{n}} H_A(\bar{n}, W) P_A(\bar{n}), \tag{50}$$

where $P_A(\bar{n})$ is just Eq. (26) with $W = \mathbb{1}$ and, from Eq. (26),

$$H_A(\bar{n}, W) = H(\bar{n}) \sqrt{\frac{\det\left(\mathbb{1} - A_W^2\right)}{\det(\mathbb{1} - A^2)}} \prod_j w_j^{n_j}. \tag{51}$$

The extra numerical cost in computing $H_A(\bar{n}, W)$ is small, as determinants and powers can be efficiently computed numerically. Due to the formal analogy between the above equation and Eq. (26) we find

$$\frac{\partial H_A(\bar{n}, W)}{\partial w_k} = H_A(\bar{n}, W) \frac{n_k - \langle n_k \rangle}{w_k}, \tag{52}$$

and, analogously to Eq. (30),

$$\partial_\theta C(\theta) = \mathbb{E}_{\bar{n} \sim P_A(\bar{n})} \left[\sum_{k=1}^{m} H_A(\bar{n}, W)(n_k - \langle n_k \rangle) \partial_\theta \ln w_k \right]. \tag{53}$$

The advantage of the above is that we can always sample from the same reference state. This approach may be used when there is a preferred choice for the $\mathcal{A}$ matrix, or when generating new samples is expensive. The next section discusses the opposite scenario.

F. Projected subgradient method

In the WAW reparametrization, the matrix A is fixed and must be set at the beginning, while the diagonal weight matrix is updated. Here we discuss a more general strategy where A is also updated at each step.

When following the gradient, it is important that the resulting matrix A always corresponds to a physical Gaussian state. As discussed before, a sufficient condition to enforce this constraint is to require that $0 \leqslant w_k \leqslant 1$ for all k, which can be enforced via a convenient parametrization. An alternative is to use the projected subgradient method, commonly employed in constrained optimization problems [37,38]. For a generic parametrized matrix A, the update rule reads

$$A \to \mathcal{P}[A - \eta \partial C], \tag{54}$$

where ∂C is a matrix with elements $(\partial C)_{ij} = \partial_{A_{ij}} C$ and $\mathcal{P}[A]$ is a projection step that projects A to the closest matrix corresponding to a physical Gaussian state. The projection step is formalized explicitly in Appendix E as a semidefinite program. The complexity of performing this projection is comparable to matrix diagonalization.

We now show that we may combine gradient rules in the WAW parametrization with the projected subgradient method to directly update the matrix A during the optimization. As outlined in the following algorithm, the strategy is to initialize weights to $w_k = 1$, update them by gradient descent, then project the new WAW matrix to the closest physical state, leading to a new matrix A'.

Formally, let $A^{(i)}$ be the matrix at step i. From an initial choice $A^{(0)}$, each step performs the following operations.

(i) Set θ such that $w_k(\theta) = 1$ for all k, e.g., set $\theta_k = 0$ for all k when using $w_k(\theta) = \exp(-\theta^T f^{(k)})$.

(ii) At step i in the optimization, update the parameters θ using $\theta \to \theta - \eta\, \partial_\theta C(\theta) =: \theta_{\text{new}}$, where $\partial_\theta C(\theta)$ is computed using the Gaussian state with matrix $WA^{(i)}W$.

(iii) Construct $A_W^{(i+1)} = W(\theta_{\text{new}}) A^{(i)} W(\theta_{\text{new}})$.

(iv) Set the updated matrix $A^{(i+1)}$ as

$$A^{(i+1)} = \mathcal{P}\left[A_W^{(i+1)}\right]. \tag{55}$$

Since in general some of the weights w_k in $W(\theta_{\text{new}})$ will satisfy $w_k > 1$ after updating the θ parameters, the matrix $A_W^{(i+1)}$ does not lead to a physical state, meaning the projection step is nontrivial and the entire A matrix is updated during the optimization. As such, this algorithm may be used when there is no preferred choice for the matrix A, which can be learned through this procedure.

V. APPLICATIONS AND NUMERICAL EXPERIMENTS

Here we apply the results of previous sections to train GBS distributions. As a first example, we show how to identify, via GBS, the ground state of a classical Ising model, which, to the best of our knowledge, represents the first use of a GBS device for such problems. We call the resulting algorithm variational Ising solver, where our gradient formulas and optimization strategies are used to train the GBS distribution to preferentially sample low-energy states. In the second example, we consider an unsupervised learning scenario where data has been generated from a GBS distribution with a known matrix A but unknown weights. We demonstrate in different cases that classical gradient formulas can be employed to train the GBS distribution to reproduce the statistics of the data. In all examples, sampling from the GBS distribution is performed using numerical simulators from the Walrus library [39].

A. Variational Ising solver

We study a classical Ising Hamiltonian

$$H(\bar{x}) = -\sum_i h_i x_i - \sum_{ij} J_{ij} x_i x_j, \tag{56}$$

where $\bar{x} = (x_1, x_2, \ldots, x_m)$ and $x_k = 0, 1$. Finding the ground state of $H(\bar{x})$ is in general NP-hard, and many known NP-hard models have a known Ising formulation [40]. We are interested in finding a model distribution that samples the Ising ground state with high probability. The output of GBS with threshold detectors is a vector $\bar{x}$ of binary variables, which is well suited for Ising problems, so we consider it here. The cost function for training is the average energy

$$E(W) = \sum_{\bar{x}} H(\bar{x}) P_{A,W}(\bar{x}) \equiv \mathbb{E}_{\bar{x} \sim P_{A,W}(\bar{x})}[H(\bar{x})], \tag{57}$$

where $P_{A,W}(\bar{x})$ is the distribution of Eq. (40). The gradient of this cost function with respect to the weights w can be approximated via Eq. (43), when $\langle n_k \rangle \ll 1$, and using

Eq. (45) when $\langle n_k \rangle \gg 1$. The exact gradient of $E(W)$, which requires photon-number-resolving detectors, is introduced in the Appendix F, while the various approximations that lead to Eqs. (43) and (45) are discussed in Appendix C.

As a concrete example, we focus on the Ising formulation of the maximum clique problem [41]. Given a graph $G = (V, E)$ with vertex set V and edge set E, a clique is an induced subgraph such that all of its vertices are connected by an edge. The maximum clique problem consists of finding the clique with the largest number of vertices. The NP-complete decision problem of whether there is a clique of size K in a graph can be rephrased as the minimization of the following Ising model [40]:

$$H_K(\bar{x}) = c_V H_V(\bar{x}) + c_E H_E(\bar{x}), \tag{58}$$

where c_V, c_E are positive constants and

$$H_V(\bar{x}) = \left(K - \sum_{v \in V} x_v\right)^2, \tag{59}$$

$$H_E(\bar{x}) = \frac{K(K-1)}{2} - \sum_{(u,v)\in E} x_u x_v, \tag{60}$$

with binary variables $x_v = \{0, 1\}$. The above Hamiltonian has ground-state energy $E = 0$ if and only if there is a clique of size K; otherwise $E > 0$. The corresponding NP-hard problem of actually finding the maximum clique can also be written as an Ising model, though the corresponding Hamiltonian H is more complicated [40].

Although finding solutions to NP-hard problems requires exponential time in a worst-case setting, we show that the training of a GBS distribution, with A fixed as the graph's adjacency matrix, leads to a distribution that samples Ising ground states with high probability. The adjacency matrix provides a starting guess, while the weights are variationally updated to get closer to the actual solution.

In Figs. 1 and 2 we study the empirical success probability of sampling the bit string $\bar{x}_{\rm gs}$ that corresponds to the ground state of an Ising Hamiltonian with $c_V = 2K$ and $c_E = 1$. The success probability is defined as the number of times that we get $\bar{x}_{\rm gs}$ in 1000 samples. To simplify the numerical calculations, the sampling algorithm is configured to output a bit string with $\sum_k \langle x_k \rangle = K$, as explained below. However, since the condition $\sum_k x_k = K$ is not exactly enforced (e.g., via postprocessing), the final success probability is unlikely to be exactly 1. Training is done using an estimation of the gradient as in Eq. (45), obtained with 1000 samples per step. At each step, the physicality of the state is enforced by first mapping negative weights to zero, then normalizing the weights so that they sum to one, and finally optimizing a coefficient c in such a way that a Gaussian state with A matrix $c(WAW)$ has $\sum_k \langle x_k \rangle = K$. Note that the weights are not reparametrized: they are directly optimized. The above operations take just a few ms per operation, thanks to Eq. (42), and effectively implement a projection step as in Sec. IV F.

In Fig. 1(a) we study a graph with eight vertices and a single clique of $K = 5$ vertices, for which a sufficiently large number of samples can be generated in a reasonable time. The probability of sampling the ground state of the Ising model is low, roughly 1.5%, when sampling from an untrained distribution with A equal to the adjacency matrix of the graph. However, using the WAW parametrization and updating the parameters via the momentum optimizer [42], we observe that the probability of sampling the ground state steadily increases and is above 85% after a few steps.

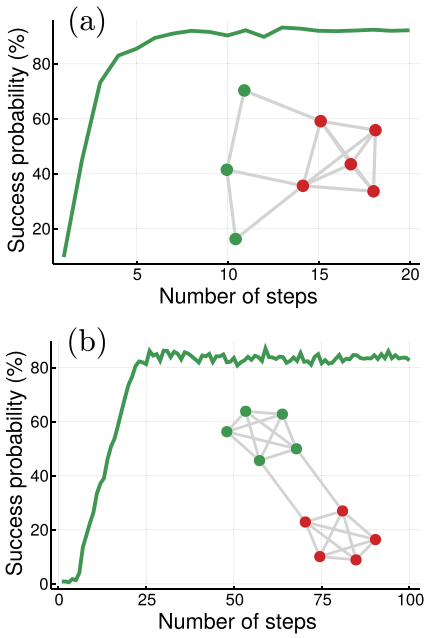

FIG. 1. Success probability, namely the probability of sampling the bit string corresponding to the ground state of the Ising model (58), as a function of the number of steps, for the displayed graph. The clique of size $K = 5$ is shown in red. In (a) there is a single clique, while in (b) there are two degenerate cliques. Training is done with 1000 samples per step.

In Fig. 1(b) we study a more challenging example: a graph with ten vertices and two largest cliques of size $K = 5$, for which the ground state of the corresponding Ising model is degenerate. Nonetheless, we observe that the training algorithm works almost as efficiently as with the simpler case of Fig. 1. During training, one of the two ground states is randomly selected and the algorithm keeps maximizing the sampling probability of that bit string without jumping to the other degenerate configuration. Running the algorithm multiple times we observe that upon convergence, both degenerate configurations can be obtained with essentially equal probability.

In Fig. 2 we switch to random graphs. The top row illustrates the effect of training for random Barabási-Albert graphs, which are built starting from a clique of size $K = 5$. These graphs are more complex than those of Fig. 1 because they contain many cliques of size three and four. We observe that training allows jumping from an initially low success probability to one higher than 80% for sampling the ground-state configuration. The bottom row shows results obtained with random Erdős-Rènyi graphs with ten vertices, constructed by adding an edge with probability $p = 0.5$. The graph in Fig. 1(d) has $K = 5$, while the graphs in Figs. 1(e) and 1(f) have $K = 4$. In all cases, the training procedure increases the probability of sampling the ground-state

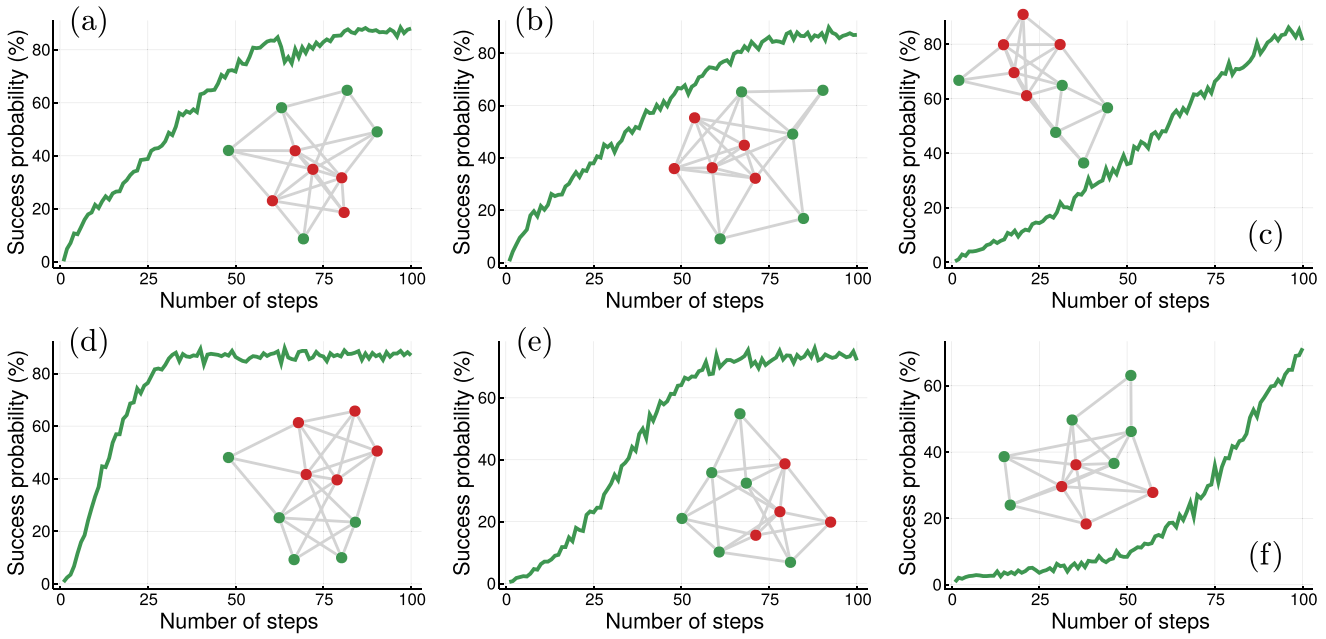

FIG. 2. Success probability as a function of the number of steps, as in Fig. 1, for the displayed graph. Graphs (a), (b), (c) are random Barabási-Albert graphs with ten vertices, built starting from a clique of five vertices and attaching new vertices, each connected to three random nodes. Graphs (d), (e), (f) are random Erdős-Rènyi graphs with ten vertices and probability $p = 0.5$ of adding an edge between pairs of vertices. Clique sizes are either four or five.

configuration, from initial values close to 0% to probabilities larger than 65% after 100 steps.

B. Unsupervised learning

In unsupervised learning, data is unlabeled and the goal is to train a model that can sample from a distribution induced by the data. Here, data is generated by sampling from a GBS simulator with threshold detectors that has been programmed according to a matrix $A_W = WAW$, where A is the adjacency matrix of a graph, and a W is a weight matrix. The data consists of 1000 samples from the distribution. For training, the weight matrix is assumed to be unknown, and the goal is to train a GBS distribution with the same A to recover the weights that were used to generate the data.

We consider three examples. The first two cases explore circulant graphs, with linearly increasing and decreasing weights, respectively. These are configurations with a high degree of symmetry. The final example is a random Erdős-Rènyi graph with randomly chosen weights, hence a less structured model. All graphs have 16 nodes.

In each case, 1000 samples are generated as the training data, with a mean photon number $\langle n \rangle = 3$. For training, we employ the parametrization $w_k(\theta) = \exp(-\theta^T f^{(k)})$, where the vectors $f^{(k)}$ and parameter vectors θ are set to dimension $d = 16$, equal to the number of vertices in the graph. The vectors are chosen to satisfy $f_l^{(k)} = \delta_{kl}$ such that $w_k(\theta) = \exp(-\theta_k)$. The cost function is the KL divergence, and we employ the approximate gradient formula of Eq. (44). We set a constant learning rate $\eta = 0.1$ and find good results when initializing all weights to be small, so in all examples we set $\theta_k = 5$ for all k.

As shown in Fig. 3, optimization based on the gradient formula of Eq. (44) works well for all examples. The weights of the model steadily and smoothly approach the data weights, until the weights at the end of training closely resemble those used to generate the training data. The entire training takes only a few seconds when running on a standard desktop computer. For our numerical calculations we use a simple stochastic gradient descent algorithm, but a better performance may be obtained with more advanced stochastic optimization techniques [43].

VI. CONCLUSIONS

We have derived a general formula for the gradient of the GBS distribution and have shown that, for specific parametrizations of the Gaussian state, the gradients of relevant cost functions take simple forms that can generally be efficiently estimated through sampling, or for specific situations, computed classically. A summary of the main theoretical results is shown in Table I. Moreover, we have showcased this framework for training GBS distributions by applying it to problems in stochastic optimization and unsupervised machine learning.

In stochastic optimization, we have introduced the variational Ising solver (VIS), a hybrid quantum-classical variational algorithm where the GBS device is used to generate samples that can be mapped to a set of binary variables. We have shown how to use the gradient formulas to train the GBS device in order to maximize the probability of sampling configurations that correspond to the ground state of a classical Ising model. Many questions still remain open, especially in order to compare VIS with alternative algorithms, such as VQE or QAOA, for qubit-based computers. For instance, it would be interesting to study how to select the fixed A matrix in the WAW parametrization, depending on the Ising

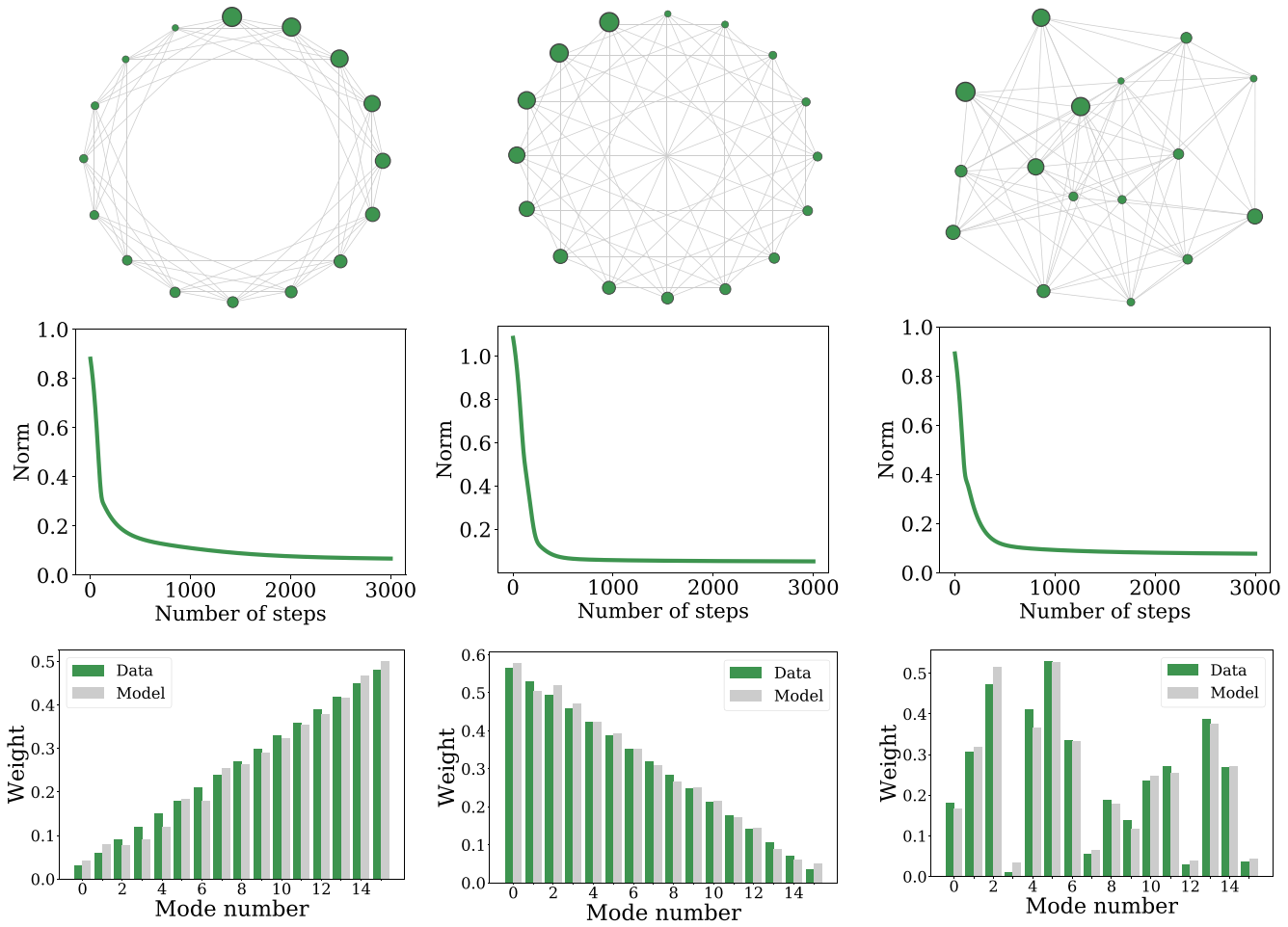

FIG. 3. Results of training a GBS distribution in an unsupervised learning scenario. (Top row): The graphs whose adjacency matrix A is used to generate the training data from a GBS simulator. The first two graphs are circulant graphs, and the third is a random Erdős-Rènyi graph with edge probability 2/3. The weights for the first graph are linearly increasing, they decrease linearly for the second graph, and for the random graph, they are chosen uniformly at random in the interval [0,1). The size of the vertices is proportional to the weights of the W matrix. The goal of training is to recover these weights. (Middle row): The norm $\|W - W_{\text{model}}\|_2$ as a function of the number of steps in the optimization. Here W is the weight matrix used to generate the data and W_{model} is the weight matrix of the model. (Bottom row): Bar graph of the weights used to generate the data versus the weights of the trained model.

Hamiltonian. Moreover, it remains to be proven if VIS can offer provable computational advantages against purely classical strategies, or whether any advantage is impossible.

In unsupervised learning, we have shown that for a specific parametrization, the gradient of the Kullback-Leibler divergence between an unknown data distribution and the GBS distribution depends only the difference between the average photon numbers $\langle n_k \rangle$ of the two distributions. These averages can be computed classically, leading to fast training, which we show can be used to retrieve GBS parameters directly from data. To the best of our knowledge, our results represent the first algorithms to variationally use near-term GBS devices to tackle optimization problems in combinatorial optimization and machine learning.

ACKNOWLEDGMENTS

The authors thank N. Killoran and T. R. Bromley for valuable discussions and comments on the manuscript. L.B. acknowledges support by the program "Rita Levi Montalcini" for young researchers.

APPENDIX A: GRADIENT DERIVATIONS

We first focus on derivatives of Hafnians and show the following result.

Proposition. The derivative of $\partial_\theta \text{Haf}(A(\theta))$ for a matrix A that depends on a certain parameter θ is given by

$$\partial_\theta \text{Haf}(A) = \frac{1}{2} \sum_{j,ki} \sum_{i \neq j} (\partial_\theta A)_{ij} \text{Haf}(A_{-j-i}), \tag{A1}$$

where A_{-j-i} is the submatrix of A where rows (i, j) and columns (i, j) have been removed.

Proof. We follow Ref. [44]: given a set of non-negative integers n_k, where $N = \sum_{j=1}^{m} n_k$ is an even number, it holds that

$$\text{Haf}(A_{\bar{n}}) = \int \prod_{j=1}^{m} dx_j \, \frac{e^{-\frac{1}{2} x^T A^{-1} x}}{\det(2\pi A)^{1/2}} \, x_1^{n_1} \ldots x_m^{n_m}, \tag{A2}$$

where A is an $m \times m$ matrix, and $A_{\bar{n}}$ is constructed by repeating rows and columns of A as discussed in Sec. IV.

Assume that the matrix $A = A(\theta)$ is parametrized by θ. To calculate the derivative of the Hafnian, we use Jacobi's formula

$$\partial_\theta \det(A) = \det(A)\mathrm{Tr}[\mathrm{A}^{-1}\partial_\theta \mathrm{A}], \tag{A3}$$

so from the chain rule

$$\partial_\theta \det(A)^{-1/2} = -\tfrac{1}{2}\det(A)^{-1/2}\mathrm{Tr}[\mathrm{A}^{-1}\partial_\theta \mathrm{A}]. \tag{A4}$$

Moreover,

$$\begin{aligned}\partial_\theta e^{-\frac{1}{2}x^T A^{-1}x} &= -\frac{1}{2}e^{-\frac{1}{2}x^T A^{-1}x}(x^T\partial_\theta A^{-1}x)\\ &= \frac{1}{2}e^{-\frac{1}{2}x^T A^{-1}x}(x^T A^{-1}\partial_\theta A A^{-1}x)\\ &= \frac{1}{2}\sum_{k,\ell} e^{-\frac{1}{2}x^T A^{-1}x}x_k x_\ell (A^{-1}\partial_\theta A A^{-1})_{k\ell},\end{aligned}$$

where we used $\partial_\theta(A^{-1}) = -A^{-1}\partial_\theta A A^{-1}$. Inserting the above equation in (A2) we get

$$\begin{aligned}\partial_\theta \mathrm{Haf}(A_{\bar{n}}) = &\frac{1}{2}\sum_{k,\ell}(A^{-1}(\partial_\theta A)A^{-1})_{k\ell}\mathrm{Haf}\left(A_{\bar{n}+\bar{e}_k+\bar{e}_\ell}\right)\\ &-\frac{1}{2}\mathrm{Tr}[\mathrm{A}^{-1}\partial_\theta \mathrm{A}]\,\mathrm{Haf}(\mathrm{A}_{\bar{\mathrm{n}}}),\end{aligned} \tag{A5}$$

where $\bar{e}_k$ is the vector with elements $(\bar{e}_k)_i = \delta_{ki}$. However, the above formula is not manifestly gauge invariant: since the Hafnian does not depend on diagonal elements of the matrix, neither should its derivative. Below we show how the gauge symmetry can be explicitly restored. Without loss of generality, consider a matrix $A_{\bar{n}}$ with all $n_k = 1$ that we simply call A. The extended matrix $A_{\bar{e}_k+\bar{e}_\ell} \equiv A_{\bar{n}+\bar{e}_k+\bar{e}_\ell}$ in (A5) takes the block form

$$A_{\bar{e}_k+\bar{e}_\ell} = \left(\begin{array}{ccc|cc} A_{11} & \dots & A_{1M} & A_{1k} & A_{1\ell}\\ \vdots & \ddots & \vdots & \vdots & \vdots\\ A_{M1} & \dots & A_{MM} & A_{Mk} & A_{M\ell}\\ \hline A_{k1} & \dots & A_{kM} & A_{kk} & A_{k\ell}\\ A_{\ell 1} & \dots & A_{\ell M} & A_{\ell k} & A_{\ell\ell}\end{array}\right). \tag{A6}$$

Note that the above matrix has the elements A_{kk} and $A_{\ell\ell}$ in off-diagonal positions, so they contribute to its Hafnian. Now we employ the Laplace-like expansion for the Hafnian [45]

$$\mathrm{Haf}(A) = \sum_{j\neq c} A_{jc}\mathrm{Haf}(A_{-j-c}), \tag{A7}$$

valid for any fixed c, where A_{-j-c} is matrix A with rows (j, c) and columns (j, c) removed. Using the expansion (A7) for $\mathrm{Haf}(A_{+\bar{e}_k+\bar{e}_\ell})$ when c is the added column $\bar{e}_\ell$ [namely the $(M+2)$th column] we get

$$\mathrm{Haf}\left(A_{+\bar{e}_k+\bar{e}_\ell}\right) = A_{k\ell}\mathrm{Haf}(A) + \sum_{j=1}^{M} A_{j\ell}\mathrm{Haf}\left(A_{\bar{e}_k-j}\right), \tag{A8}$$

where we used the fact that the index j in (A7) takes $M+1$ values, as it runs from 1 to M and to the copy of the k's column. Inserting this equation into Eq. (A5) we get

$$\partial_\theta \mathrm{Haf}(A) = \frac{1}{2}\sum_{k,j=1}^{M}\left((\partial_\theta A)A^{-1}\right)_{jk}\mathrm{Haf}\left(A_{\bar{e}_k-j}\right). \tag{A9}$$

Using again Eq. (A7) with c equal to the added column $\bar{e}_k$ we get

$$\mathrm{Haf}\left(A_{+\bar{e}_k-j}\right) = \sum_{i\neq j} A_{ik}\mathrm{Haf}(A_{-i-j}), \tag{A10}$$

Inserting the above in Eq. (A9) we get

$$\partial_\theta \mathrm{Haf}(A) = \frac{1}{2}\sum_{j,ki}\sum_{i\neq j}((\partial_\theta A)A^{-1})_{jk}A_{ik}\mathrm{Haf}\left(A_{-\bar{e}_j-\bar{e}_i}\right), \tag{A11}$$

and the proposition follows. The above final form is independent of the diagonal elements of A, as desired. ■

We now focus on the gradient of the GBS distribution in Eq. (18). Using (A1) with the matrix $A_{\bar{n}}$, we get

$$\partial_\theta \mathrm{Haf}(A_{\bar{n}}) = \frac{1}{2}\sum_{i\neq j}(\partial_\theta A_{\bar{n}})_{ij}\ \mathrm{Haf}(A_{\bar{n}-\bar{e}_j-\bar{e}_i}). \tag{A12}$$

Finally to get $\partial_\theta \frac{1}{\mathcal{Z}} = \partial_\theta\sqrt{\det(\mathbb{1}-X\mathcal{A})}$ we can use (A3) to write

$$\partial_\theta \det(\mathcal{B})^{1/2} = \tfrac{1}{2}\det(\mathcal{B})^{1/2}\mathrm{Tr}[\mathcal{B}^{-1}\partial_\theta\mathcal{B}]. \tag{A13}$$

Calling $\mathcal{B} = \mathbb{1} - X\mathcal{A}$, we have

$$\begin{aligned}\mathrm{Tr}[\mathcal{B}^{-1}\partial_\theta\mathcal{B}] &= -\mathrm{Tr}[\mathcal{B}^{-1}\mathrm{X}\partial_\theta\mathcal{A}]\\ &= -\mathrm{Tr}[(\mathcal{B}\mathrm{X})^{-1}\partial_\theta\mathcal{A}]\\ &= -\mathrm{Tr}\left[\frac{1}{\mathrm{X}-\mathcal{A}}\partial_\theta\mathcal{A}\right],\end{aligned} \tag{A14}$$

since $X = X^{-1}$. The above formula, together with (A13) proves the resulting Eq. (19).

For a pure state $\mathcal{A} = A \oplus A$ so we get

$$P_A^{\mathrm{pure}}(\bar{n}) = \frac{\sqrt{\det(\mathbb{1}-A^2)}}{\bar{n}!}\mathrm{Haf}(A_{\bar{n}})^2, \tag{A15}$$

and

$$\frac{\partial_\theta P_A^{\mathrm{pure}}(\bar{n})}{P_A^{\mathrm{pure}}(\bar{n})} = -\frac{1}{2}\mathrm{Tr}\left[\frac{2\mathrm{A}}{\mathbb{1}-\mathrm{A}^2}\partial_\theta\mathrm{A}\right] + 2\frac{\partial_\theta\mathrm{Haf}(\mathrm{A}_{\bar{\mathrm{n}}})}{\mathrm{Haf}(\mathrm{A}_{\bar{\mathrm{n}}})}. \tag{A16}$$

Finally, we note that the formula (A1) for evaluating gradients of the Hafnian function allows us to compute also the gradient of matrix permanents. Indeed, from Ref. [45] we have

$$\mathrm{per}(A) = \mathrm{Haf}\begin{pmatrix}0 & A\\ A^T & 0\end{pmatrix}, \tag{A17}$$

so we can use Eqs. (A1) and (A12) to get the gradient of the matrix permanent.

Gradients in the WAW parametrization

Recall the GBS probability distribution in the WAW parametrization

$$P_{A,W}(\bar{n}) = \sqrt{\det\left(\mathbb{1}-A_W^2\right)}\ \mathrm{Haf}(A_{\bar{n}})^2\prod_j \frac{w_j^{n_j}}{n_j!}. \tag{A18}$$

To write the gradient of the above distribution, we see that

$$\frac{\partial_{w_k}\prod_j w_j^{n_j}}{\prod_j w_j^{n_j}} = \begin{cases} \frac{n_k}{w_k} & \text{if } n_k > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{A19}$$

Then we get

$$\partial_{w_k} P_{A,W}(\bar{n}) = \frac{n_k}{w_k} P_{A,W}(\bar{n}) - \frac{1}{2} P_{A,W}(\bar{n}) \mathrm{Tr}\left[\frac{2\mathrm{A_W}}{\mathbb{1} - \mathrm{A_W^2}} \partial_{\mathrm{w_k}} \mathrm{WAW}\right]. \tag{A20}$$

By explicit calculations

$$\begin{aligned} \partial_{w_k} WAW &= \tfrac{1}{2} w_k^{\frac{1}{2}-1} (|k\rangle\langle k|AW + WA|k\rangle\langle k|) \\ &= \tfrac{1}{2} w_k^{-1} (|k\rangle\langle k|WAW + WAW|k\rangle\langle k|) \\ &= \tfrac{1}{2} w_k^{-1} (|k\rangle\langle k|A_W + A_W|k\rangle\langle k|), \end{aligned} \tag{A21}$$

we then obtain

$$\begin{aligned} \partial_{w_k} P_{A,W}(\bar{n}) &= \left(\frac{n_k}{w_k} - \frac{1}{w_k}\langle k|\left[\frac{A_W^2}{\mathbb{1} - A_W^2}\right]|k\rangle\right) P_{A,W}(\bar{n}) \\ &= \frac{n_k - \langle n_k\rangle}{w_k} P_{A,W}(\bar{n}), \end{aligned} \tag{A22}$$

where $\langle n_k\rangle$ is the average number of photons in mode k.

APPENDIX B: WEIGHT UPDATING

1. Spectral properties

When A has spectrum in $[-1, 1]$ we show that, under some conditions, even the matrix A_W has the same property. This corresponds to the requirement that

$$|\langle x|A_W|x\rangle| \leqslant \langle x|x\rangle \quad \text{for each } |x\rangle. \tag{B1}$$

Let $|y\rangle = W^{1/2}|x\rangle$ then

$$|\langle x|A_W|x\rangle| = |\langle y|A|y\rangle| \leqslant \langle y|y\rangle| = |\langle x|W|x\rangle| \leqslant \langle x|x\rangle, \tag{B2}$$

where we used the fact that the eigenvalues of A are smaller than one, while the last equality is true if

$$0 \leqslant w_k \leqslant 1. \tag{B3}$$

So if A was a valid parametrization for a pure-state GBS distribution, then so is A_W, provided that the weights satisfy the above inequality. The conditions (B3) provide a sufficient condition for having a valid A_W matrix, which in general is not necessary.

2. Generalization to mixed states

A sensible generalization of the update rule in Eq. (23) is the following:

$$\mathcal{A} \to \mathcal{A}_{\mathcal{W}} = \mathcal{W}^{1/2} \mathcal{A} \mathcal{W}^{1/2}, \tag{B4}$$

where $\mathcal{W} = W \oplus W$. In the case where $\mathcal{A}$ is block diagonal then this rule indeed reduces to Eq. (23), which is of course the desired limit behavior.

Now we would like to argue that the transformation in Eq. (B4) also maps a valid $\mathcal{A}$ matrix corresponding to a Gaussian state to another $\mathcal{A}_{\mathcal{W}}$ that corresponds to a Gaussian state. Recall that the covariance matrix V of the Gaussian state is related to the $\mathcal{A}$ matrix as [recall Eq. (2)]

$$\mathcal{A} = X(\mathbb{1} - [V + \mathbb{1}/2]^{-1}). \tag{B5}$$

For V to be a valid quantum covariance matrix it needs to satisfy the uncertainty relation

$$V + \frac{Z}{2} \geqslant 0, \tag{B6}$$

where $Z = \sigma^z \otimes \mathbb{1}_m$. The update equation for $\mathcal{A}$ matrices can be written in terms of the covariance matrix as

$$\begin{aligned} &V \to V_{\mathcal{W}}, \\ &= -\frac{\mathbb{1}_{2m}}{2} + \left[\mathbb{1}_{2m} - \mathcal{W} + \mathcal{W}^{1/2}\left(V + \frac{\mathbb{1}_{2m}}{2}\right)^{-1} \mathcal{W}^{1/2}\right]^{-1}. \end{aligned} \tag{B7}$$

One would like to show that the matrix $V_{\mathcal{W}}$ is a valid quantum covariance matrix if V is a valid quantum covariance matrix, i.e., that it satisfies $V_{\mathcal{W}} + \frac{1}{2}Z \geqslant 0$. A simple way to show this is to first define the matrix $V^\epsilon = V + \epsilon \mathbb{1}_{2m}$, which is always a valid quantum covariance matrix if V is also in this set. Then defining $V^\epsilon_{\mathcal{W}}$ to be the matrix obtained by letting $V \to V^\epsilon$ in Eq. (B7) one can easily show the following inequality:

$$V^\epsilon_{\mathcal{W}} + \frac{\mathbb{1}_{2m}}{2} \geqslant \begin{bmatrix} (\mathbb{1}_m - W + \epsilon^{-1}W)^{-1} & 0 \\ 0 & (\mathbb{1}_m - W + (1+\epsilon)^{-1}W)^{-1} \end{bmatrix}, \tag{B8}$$

assuming Eq. (B6) holds. In the limit $\epsilon \to 0$, one has $V^\epsilon \to V$, $V^\epsilon_{\mathcal{W}} \to V_{\mathcal{W}}$ and

$$\begin{bmatrix} (\mathbb{1}_m - W + \epsilon^{-1}W)^{-1} & 0 \\ 0 & (\mathbb{1}_m - W + (1+\epsilon)^{-1}W)^{-1} \end{bmatrix} \to \frac{\mathbb{1}_{2m}}{2} - \frac{Z}{2}, \tag{B9}$$

thus showing that indeed $V_{\mathcal{W}} + Z/2 \geqslant 0$ and $V_{\mathcal{W}}$ is a valid covariance matrix.

APPENDIX C: VARIATIONAL ISING SIMULATION WITH THRESHOLD DETECTORS

Numerical simulation of GBS is very complicated even for small-scale problems, as the range of possible integer values n_k is possibly unbounded. Moreover, from the experimental point of view, GBS requires NRDs, which are more complex and less efficient than threshold detectors. GBS with threshold detectors was introduced in Ref. [33] and it was proven that the resulting sampling is still ♯P-hard. The use of threshold detector formally results in the mapping (39), namely the kth detector clicks only when $n_k > 0$. We write $x_k = 1$ in that case, and $x_k = 0$ otherwise. The outcome is then a collection of binary variables $\bar{x}$, which are related to the number

distribution via (39). As threshold detectors output a binary variable, they are well suited for Ising model formulation. In Appendix F we show that, when using number-resolving detectors, exact gradients of the average energy can be obtained via an extension of the Ising model $H(\bar{x}) = H(\bar{n})$, where all numbers n_k are mapped to $x_k = 0$ if $n_k = 0$ and $x_k = 1$ if $n_k \geqslant 1$. When using threshold detectors, this extension not required, as the output of the detectors is the desired binary variable x_k. However, we also need to consider the other n-dependent terms in Eq. (F3).

Let $B_{\bar{x}} = \{\bar{n} : \bar{x}(\bar{n}) = \bar{x}\}$ be the set of all possible integer sequences that produce the same binary string $\bar{x}$ via Eq. (39). Clearly, for fixed x, the set $B_{\bar{x}}$ contains infinitely many sequences $\bar{n}$. The probability

$$p_{\mathrm{Tor},W,A}(\bar{x}) = \sum_{\bar{n}\in B_{\bar{x}}} p_{A,W}(\bar{n}), \tag{C1}$$

is the GBS probability with threshold detectors. On the other hand, with these definitions, the energy gradient can be decomposed as

$$\frac{\partial E(w)}{\partial w_k} = \sum_{\bar{x}} H(\bar{x}) \sum_{\bar{n}\in B_{\bar{x}}} \frac{n_k - \langle n_k\rangle}{w_k}\, p_{A,W}(\bar{n}).$$

The aim is to separate the second sum for using (C1). Indeed, we may write

$$\sum_{\bar{n}\in B_{\bar{x}}} n_k p_{A,W}(\bar{n}) = \tilde{n}_k(\bar{x})\, p_{\mathrm{Tor},A,W}(\bar{x}), \tag{C2}$$

where

$$\tilde{n}_k(\bar{x}) = \begin{cases} \sum_{n_k} n_k p_{A,W}(n_k|\bar{x}, x_k{=}1) & \text{if } x_k = 1, \\ 0 & \text{if } x_k = 0, \end{cases} \tag{C3}$$

and $p_{A,W}(n_k|\bar{x}, x_k{=}1)$ is the conditional probability of having n_k photons given that the kth detector clicked and that the other detectors produced the outcome $\bar{x}$. With these definitions we finally get

$$\frac{\partial E(w)}{\partial w_k} = \mathbb{E}_{\bar{x}\sim\mathrm{Tor}}\left[H(\bar{x})\, \frac{\tilde{n}_k(\bar{x}) - \langle n_k\rangle}{w_k}\right], \tag{C4}$$

where $\bar{x} \sim \mathrm{Tor}$ is a shorthand notation to write that $\bar{x}$ is sampled from (C1). The above gradient is still exact, as no approximations have been made so far. The expectation value $\langle n_k\rangle$ is simple to get in a closed form from the Gaussian covariance matrix, whereas the quantity $\tilde{n}_k(\bar{x})$ is hard to estimate. Nonetheless, we can use the fact that $n_k \geqslant 1$ when $x_k = 1$ to write $\tilde{n}_k(\bar{x}) \geqslant x_k$. The above implies

$$\begin{aligned}\frac{\partial E(w)}{\partial w_k} &\geqslant \sum_{\bar{x}} H(\bar{x})\, \frac{x_k - \langle n_k\rangle}{w_k}\, p_{\mathrm{Tor},A,W}(\bar{x}) \\ &= \mathbb{E}_{\bar{x}\sim\mathrm{Tor}}\left[H(\bar{x})\, \frac{x_k - \langle n_k\rangle}{w_k}\right],\end{aligned} \tag{C5}$$

namely the exact gradient is lower bounded by a quantity that can be estimated with via GBS with threshold detectors. An alternative estimation of the gradient is via the approximation $\tilde{n}_k(\bar{x}) \approx \max\{\langle n_k\rangle, 1\}x_k$, so

$$\frac{\partial E(w)}{\partial w_k} \approx \mathbb{E}_{\bar{x}\sim\mathrm{Tor}}\left[H(\bar{x})\, \frac{\max\{\langle n_k\rangle(x_k - 1), x_k - \langle n_k\rangle\}}{w_k}\right]. \tag{C6}$$

While Eq. (C5) is always a lower bound to the exact gradient, Eq. (C6) is just an approximation. However, we found that in numerical experiments it performs very well.

For GBS with number-resolving detectors, Eq. (F2) provides an unbiased estimator of the gradient, so convergence can be exactly proven for stochastic gradient descent algorithms. On the other hand, Eqs. (C6) and (C5) represent a biased estimator. Nonetheless, it has been shown that convergence is expected even with some biased gradient estimators [34].

APPENDIX D: GENERAL CONSIDERATIONS ON THE QUANTUM REPARAMETRIZATION TRICK

To study a general form of the quantum reparametrization trick for GBS, we write the cost function (9) as

$$C(\theta) = \sum_{\bar{n}} H(\bar{n}) P_{\mathcal{A}(\theta)}(\bar{n}), \tag{D1}$$

where $\mathcal{A}(\theta)$ is the θ-dependent $\mathcal{A}$ matrix of a Gaussian state and $\bar{n}$ is a vector of numbers, where n_i is the number of detected photons in mode i. The above cost function can be written using quantum operators as

$$C(\theta) = \mathrm{Tr}[\mathrm{H}\rho(\theta)], \tag{D2}$$

where $\rho(\theta)$ is a quantum state (in general, not necessarily Gaussian) and

$$H = \sum_{\bar{n}} H(\bar{n})|\bar{n}\rangle\langle\bar{n}|. \tag{D3}$$

If we expand the trace in the Fock basis, then for a Gaussian state with $\mathcal{A}$ matrix $\mathcal{A}(\theta)$ we get (D1). Now assume that

$$\rho(\theta) = \mathcal{R}_\theta[\rho_0], \tag{D4}$$

where $\mathcal{R}_\theta$ is a quantum channel, namely a completely positive trace preserving linear map, and ρ_0 is a reference state that does not depend on θ. Using the dual channel $\mathcal{R}^*_\theta$ we find

$$C(\theta) = \mathrm{Tr}[\mathcal{R}^*_\theta(\mathrm{H})\rho_0], \tag{D5}$$

and

$$\partial_\theta C(\theta) = \mathrm{Tr}[\rho_0\, \partial_\theta \mathcal{R}^*_\theta(\mathrm{H})]. \tag{D6}$$

In (D2) the observable is θ independent, but the state $\rho(\theta)$ changes at each step. On the other hand, in Eq. (D5) the quantum state is always the same and the observable is changed.

GBS can be used for estimating the gradient in at least two cases:

(i) When $\mathcal{R}_\theta$ maps diagonal states (in the Fock basis) to diagonal states. In that case

$$\mathcal{R}^*_\theta(H) = \sum_{\bar{n}} H_{\mathcal{R}}(\bar{n}, \theta)|\bar{n}\rangle\langle\bar{n}|, \tag{D7}$$

for some $H_{\mathcal{R}}(\bar{n}|\theta)$ that depends on $\mathcal{R}$. Calling $\mathcal{A}_0$ the $\mathcal{A}$ matrix of ρ_0 we find

$$C(\theta) = \sum_{\bar{n}} H_{\mathcal{R}}(\bar{n}, \theta) p(\theta|\mathcal{A}_0), \tag{D8}$$

and

$$\partial_\theta C(\theta) = \sum_{\bar{n}} \partial_\theta H_{\mathcal{R}}(\bar{n}, \theta)\, p(\bar{n}|\mathcal{A}_0) \tag{D9}$$

$$= \mathbb{E}_{\bar{n}\sim p(\bar{n}|\mathcal{A}_0)}[\partial_\theta H_{\mathcal{R}}(\bar{n}, \theta)]. \tag{D10}$$

Therefore, we can always sample from a reference state ρ_0 to get the gradient.

(ii) When $\partial_\theta \mathcal{R}^*_\theta(H)$ can be put in a diagonal Fock basis by a symplectic transformation $S(\theta)$, possibly dependent on θ. Namely, if

$$\partial_\theta \mathcal{R}^*_\theta(H) = \sum_{\bar{n}} h'(\bar{n}, \theta)\, S(\theta)|\bar{n}\rangle\langle\bar{n}|S(\theta)^\dagger, \tag{D11}$$

then

$$\partial_\theta C(\theta) = \sum_{\bar{n}} h'(\bar{n}, \theta)\, p(\bar{n}|\mathcal{A}_{S(\theta)}) \tag{D12}$$

$$= \mathbb{E}_{\bar{n}\sim p(\bar{n}|\mathcal{A}_{S(\theta)})}[h'(\bar{n}, \theta)], \tag{D13}$$

where $\mathcal{A}_{S(\theta)}$ is the $\mathcal{A}$ matrix of the state $S(\theta)^\dagger \rho_0 S(\theta)$. Therefore, for each θ we can run a θ-dependent GBS to estimate the gradient.

APPENDIX E: PROJECTION TO THE CLOSEST GAUSSIAN STATE

We discuss the case of a pure Gaussian state with $\mathcal{A} = A \oplus A$ and $A^* = A$. In that case, a physical state is defined by the requirement that $A = A^T$ and that its spectrum lies in $[-1, 1]$. The latter condition can be enforced by requiring that $A \pm \mathbb{1}$ are positive semidefinite operators, so the projection step $\mathcal{P}[X]$ can be computed via semidefinite programming as

$$\text{minimize } \|X - A\|, \tag{E1}$$

$$\text{such that } A = A^T, A \pm \mathbb{1} \geqslant 0, \tag{E2}$$

for a suitable norm $\|\cdot\|$. Using the projected subgradients we can then update the parameters via (13) and (30), and then finding the closest Gaussian state via the projection.

APPENDIX F: VARIATIONAL ISING SIMULATION WITH NUMBER RESOLVING DETECTORS

The main difference between the configuration space $\bar{x}$ of an Ising problem and the possible outputs $\bar{n}$ of GBS is that $\bar{x}$ is a vector of binary variables while $\bar{n}$ is made of arbitrary positive integers. There are many ways of defining a binary variable out of an integer. Here, we focus on the mapping (39), as it is naturally implemented experimentally by threshold detectors. By reversing that mapping we may extend the Ising model to arbitrary integer sequences via $H(\bar{n}) = H[x(\bar{n})]$. With these definitions, the goal is then to minimize the average energy

$$E(w) = \sum_{\bar{n}} H(\bar{n}) p_{A,W}(\bar{n}) \equiv \mathbb{E}_{\bar{n}\sim p_{A,W}(\bar{n})}[H(\bar{n})]. \tag{F1}$$

The gradient of the above energy cost function easily follows from Eq. (27) [extension to the more general (29) is trivial], and we find

$$\frac{\partial E(w)}{\partial w_k} = \mathbb{E}_{\bar{n}\sim p_{A,W}(\bar{n})}[G_k(\bar{n}, w)], \tag{F2}$$

$$G_k(\bar{n}, w) = H(\bar{n})\, \frac{n_k - \langle n_k\rangle}{w_k}. \tag{F3}$$

Therefore, we can estimate the gradient by sampling from the GBS devices, without calculating classically hard quantities like the Hafnians. Indeed, from many sampled integer strings $\bar{n}$ we can easily calculate $G_k(\bar{n}|w)$ and update the weights following the stochastic estimation of the gradient.

[1] C. S. Hamilton, R. Kruse, L. Sansoni, S. Barkhofen, C. Silberhorn, and I. Jex, Gaussian Boson Sampling, Phys. Rev. Lett. **119**, 170501 (2017).

[2] R. Kruse, C. S. Hamilton, L. Sansoni, S. Barkhofen, C. Silberhorn, and I. Jex, Detailed study of Gaussian boson sampling, Phys. Rev. A **100**, 032326 (2019).

[3] T. R. Bromley, J. M. Arrazola, S. Jahangiri, J. Izaac, N. Quesada, A. D. Gran, M. Schuld, J. Swinarton, Z. Zabaneh, and N. Killoran, Applications of near-term photonic quantum computers: Software and algorithms, Quantum Sci. Technol. **5**, 034010 (2020).

[4] J. M. Arrazola and T. R. Bromley, Using Gaussian Boson Sampling to Find Dense Subgraphs, Phys. Rev. Lett. **121**, 030503 (2018).

[5] J. M. Arrazola, T. R. Bromley, and P. Rebentrost, Quantum approximate optimization with Gaussian boson sampling, Phys. Rev. A **98**, 012322 (2018).

[6] L. Banchi, M. Fingerhuth, T. Babej, C. Ing, and J. M. Arrazola, Molecular docking with Gaussian boson sampling, Sci. Adv. **6**, eaax1950 (2020).

[7] K. Bradler, S. Friedland, J. Izaac, N. Killoran, and D. Su, Graph isomorphism and Gaussian boson sampling, arXiv:1810.10644.

[8] M. Schuld, K. Brádler, R. Israel, D. Su, and B. Gupt, A quantum hardware-induced graph kernel based on Gaussian Boson Sampling, Phys. Rev. A **101**, 032314 (2020).

[9] S. Jahangiri, J. M. Arrazola, N. Quesada, and N. Killoran, Point processes with Gaussian boson sampling, Phys. Rev. E **101**, 022134 (2020).

[10] J. Huh, G. G. Guerreschi, B. Peropadre, J. R. McClean, and A. Aspuru-Guzik, Boson sampling for molecular vibronic spectra, Nature Photonics **9**, 615 (2015).

[11] J. Huh and M.-H. Yung, Vibronic boson sampling: Generalized Gaussian boson sampling for molecular vibronic spectra at finite temperature, Sci. Rep. **7**, 7462 (2017).

[12] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New J. Phys. **18**, 023023 (2016).

[13] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[14] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance,

mechanism, and implementation on near-term devices, Phys. Rev. X **10**, 021067 (2020).
[15] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Qi. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nature Commun. **5**, 4213 (2014).
[16] M. Schuld and N. Killoran, Quantum Machine Learning in Feature Hilbert Spaces, Phys. Rev. Lett. **122**, 040504 (2019).
[17] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature (London) **567**, 209 (2019).
[18] M. Schuld, A. Bocharov, K. Svore, and N. Wiebe, Circuit-centric quantum classifiers, Phys. Rev. A **101**, 032308 (2020).
[19] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri, K. McKiernan, J. J. Meyer, Z. Niu, A. Száva, and N. Killoran, PennyLane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968.
[20] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, Phys. Rev. A **99**, 032331 (2019).
[21] L. Gentini, A. Cuccoli, S. Pirandola, P. Verrucchi, and L. Banchi, Noise-assisted variational hybrid quantum-classical optimization, arXiv:1912.06744.
[22] E. R. Caianiello, On quantum field theory: Explicit solution of Dyson's equation in electrodynamics without use of Feynman graphs, Il Nuovo Cimento (1943–1954) **10**, 1634 (1953).
[23] A. Barvinok, *Combinatorics and Complexity of Partition Functions* (Springer, Berlin, 2016), Vol. 276.
[24] A. Björklund, B. Gupt, and N. Quesada, A faster Hafnian formula for complex matrices and its benchmarking on a supercomputer, J. Exper. Algorithm. (JEA) **24**, 11 (2019).
[25] S. Aaronson and A. Arkhipov, The computational complexity of linear optics, Theor. Comput. **9**, 143 (2013).
[26] K. Brádler, P.-Luc Dallaire-Demers, P. Rebentrost, D. Su, and C. Weedbrook, Gaussian boson sampling for perfect matchings of arbitrary graphs, Phys. Rev. A **98**, 032310 (2018).
[27] S. Bubeck *et al.*, Convex optimization: Algorithms and complexity, Found. Trends Mach. Learn. **8**, 231 (2015).
[28] J. C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control* (John Wiley & Sons, New York, 2005), Vol. 65.
[29] N. Quesada, L. G. Helt, J. Izaac, J. M. Arrazola, R. Shahrokhshahi, C. R. Myers, and K. K. Sabapathy, Simulating realistic non-Gaussian state preparation, Phys. Rev. A **100**, 022341 (2019).
[30] A. Kulesza and B. Taskar, Learning determinantal point processes, in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2011, pp. 419–427.
[31] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, Experimental Realization of Any Discrete Unitary Operator, Phys. Rev. Lett. **73**, 58 (1994).
[32] W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, and I. A. Walmsley, Optimal design for universal multiport interferometers, Optica **3**, 1460 (2016).
[33] N. Quesada, J. M. Arrazola, and N. Killoran, Gaussian boson sampling using threshold detectors, Phys. Rev. A **98**, 062322 (2018).
[34] J. Chen and R. Luss, Stochastic gradient descent with biased but consistent gradient estimators, arXiv:1807.11880.
[35] D. P. Kingma and M. Welling, Auto-encoding variational Bayes, arXiv:1312.6114.
[36] W. Vinci, L. Buffoni, H. Sadeghi, A. Khoshaman, E. Andriyash, and M. H. Amin, A path towards quantum advantage in training deep generative models with quantum annealers, arXiv:1912.02119.
[37] S. Boyd, L. Xiao, and A. Mutapcic, Subgradient methods, in *Lecture notes of EE392O*, Autumn Quarter (Stanford University, 2003), pp. 2004–2005.
[38] L. Banchi, J. Pereira, S. Lloyd, and S. Pirandola, Convex optimization of programmable quantum computers, npj Quantum Information **6**, 42 (2020).
[39] B. Gupt, J. Izaac, and N. Quesada, The Walrus: a library for the calculation of Hafnians, hermite polynomials and Gaussian boson sampling, J. Open Source Software **4**, 1705 (2019).
[40] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, 5 (2014).
[41] Q. Wu and J.-K. Hao, A review on algorithms for maximum clique problems, Euro. J. Oper. Res. **242**, 693 (2015).
[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature (London) **323**, 533 (1986).
[43] D. P. Kingma and J. Ba, A method for stochastic optimization, arXiv:1412.6980.
[44] R. Kan, From moments of sum to moments of product, J. Multivariate Anal. **99**, 542 (2008).
[45] A. Barvinok, Approximating permanents and Hafnians, Discrete Analysis **2**, 34 (2017).