Approximated quantum-state preparation with entanglement dependent complexity

Israel F. Araujo,1, * Carsten Blank,2, † Ismael Cesar,1, ‡ and Adenilton J. da Silva1, §

1Centro de Informática, Universidade Federal de Pernambuco, 50740-560, Recife, Pernambuco, Brazil
2data cybernetics, 86899, Landsberg am Lech, Germany

Ubiquitous in quantum computing is the step to encode data into a quantum state. This process is called quantum state preparation, and its complexity for non-structured data is exponential on the number of qubits. Several works address this problem, for instance, by using variational methods that train a fixed depth circuit with manageable complexity. These methods have their limitations as the lack of a back-propagation technique and barren plateaus. This work proposes an algorithm to reduce state preparation circuit depth by offloading computational complexity to a classical computer. The initialized quantum state can be exact or an approximation, and we show that the approximation is better on today’s quantum processors than the initialization of the original state. We verified through experimental evaluation that the proposed method allows the initialization of distributions in a quantum state and approximate loading images for quantum machine learning tasks. Indeed, our experiments on IBMQ devices show that they are likely to be aiding experimenters of quantum computing throughout the NISQ era.

INTRODUCTION

Quantum devices can execute information processing tasks that classical computers cannot perform efficiently [1]. The initialization of a $n$-qubit quantum state (state preparation) is an important step in quantum information processing and requires quantum circuits with $O(2^n)$ controlled-NOT (CNOT) gates [2–4]. Several works focus on the development of algorithms that supposes data-efficient initialization. For instance, this supposition leads to exponential advantages in the solution of systems of linear equations [5] and principal component analysis [6].

There are several quantum state preparation algorithms [7–10] with a lower bound of $O(2^n)$ CNOT gates to prepare an arbitrary quantum state with $n$ qubits. Attempts to prepare quantum states more efficiently include a divide-and-conquer strategy that exchanges circuit depth by circuit width [10, 11], probabilistic approaches [12, 13], and strategies to initialize approximated quantum states [14–16]. Most recently, there are works that focus on specific classes of quantum states. For instance, how to prepare uniform [17], sparse [9] or probability distribution [14] states. However, there is no clear understanding of which classes of quantum states can be created efficiently.

Entanglement is one of the quantum resources that allow the development of more efficient algorithms and protocols. Its relationship to the algorithmic complexity of a quantum state has been demonstrated in previous work [18, 19] and plays a key role in many quantum applications, such as quantum communication, quantum error correction, and quantum secret sharing [20–25]. However, the circuit complexity of current algorithms to create a quantum state do not consider the amount of entanglement. In fact, a circuit of only single-qubit operations will have a depth of one and an entanglement of zero. In the context of state preparation algorithms, is there a property to exploit, when we know that the entanglement is low? The main goal of this article is to define the complexity of state preparation as a function of an entanglement measure, instead of only the number of qubits.

The proposed algorithm can accelerate quantum applications that require the initialization of quantum states on noisy devices. The result we find is that the error introduced by the approximation is smaller than the error to encode the original state. The fundamental cause of this behavior is mostly due to the difference in the number of noisy operations necessary between the circuits to encode the original versus the approximate states.

The principal mechanism of the algorithm explores the level of entanglement of the input vector, and leverages the ability to prepare a low-rank approximation [26] of a quantum state, using the Schmidt decomposition. One can use a well-known algorithm, by Plesch et al. [8], as an algorithmic vehicle to drive the low-rank state preparation towards concrete quantum operations. It is possible to completely disentangle a bipartition and recursively apply the analysis of the two resulting vectors. This principle can be applied by a search algorithm with a given maximal error allowed. This bounded approximation error algorithm (BAA) can be applied to any quantum state of interest, including, for example, vector encoding for inverting matrices using HHL [6], loading data into a quantum machine learning model [4, 27] or using quantum simulation of stochastic processes [28].

In order to compute the Schmidt decomposition, classically, a singular-value-decomposition (SVD) must be applied. This algorithm is expensive computationally, scaling cubical with the number of input variables. Even though it is possible to gain some advantage on a quan-
tum computer by applying the BAA, the number of operations as pre-processing scale also exponentially with respect to the number of qubits. However, the ability to off-load computational complexity from the quantum device to a classical device is an important consequence of this observation. There is an error introduced by this off-loading procedure, and this error turns out to never be worse than the geometric entanglement of a quantum state [29]. In a sense, a low-entanglement quantum state is more classical, so off-loading complexity to a classical device will of course not introduce a strong error on a quantum device.

While it is clear that entanglement is an important resource in quantum computation [30], being able to encode low-entanglement states, such as common probability density functions, is still useful as algorithms can use low-entanglement building blocks to reach high-entanglement structures. One such example can be seen in stochastic process simulation [28, 31, 32] using amplitude estimation. On the other hand, even though this method introduces higher approximation errors on highly entangled states, the algorithm still finds good approximations, therefore not limiting its applicability in this regime.

RESULTS

Basic Framework

Entanglement is an important concept in the idea of this work, so it is in order to briefly discuss it, but referring to literature for a more detailed treatment of the subject. The central concept here is the Schmidt decomposition, which describes the entanglement correlations for pure bipartite states [33–36]. A numerical measure of entanglement between the bipartite state subsystems can be based on the squares of the Schmidt coefficients, $|\sigma_i|^2$ since unitary operations performed locally on the subsystems do not change these quantities [37].

Another simple alternative to quantify entanglement is the Schmidt rank (the number of non-zero $\sigma_i$ amplitudes) and is defined in units of e-bits by $E_S(\langle \psi \rangle) = \log_2(\text{rank})$. The amount of entanglement contained in a maximally entangled two-qubit state defines the unit of bipartite entanglement e-bit. For example, the Bell states have rank two, so $E_S(\langle \text{bell} \rangle) = 1$ e-bit.

While this measure of entanglement is of high theoretical interest, we connect it with a concrete algorithm for loading data into quantum devices. Indeed, there is a well-known state preparation algorithm that connects to the Schmidt-decomposition by Plesch et al. [8]. A quick review of the algorithm is schematically shown in Fig. 5d and its immediate application to a low-rank state preparation.

One can observe how entanglement affects the complexity of the circuit. Such a circuit is a direct application of the Schmidt decomposition, and it is modified as follows to achieve low-rank state preparation (LRSP).

Low-rank state preparation

Let $|\psi\rangle$ be a $n$-qubit quantum state with Schmidt decomposition

$$|\psi\rangle = \sum_{i=0}^{k} \sigma_i |i_A\rangle |i_B\rangle$$

where subsystem $\mathcal{H}_A$ has $n_A$ qubits ($1 \leq n_A \leq \lfloor n/2 \rfloor$), subsystem $\mathcal{H}_B$ has $n_B = n - n_A$ qubits and $k$ is the Schmidt rank (also known as Schmidt number). The modification of the algorithm by Plesch et al. [8] together with work on isometries by Iten et al. [38] give a recipe to leverage Schmidt decompositions.

**Theorem 1** (Low-Rank State Preparation). Given $\text{Eqn. (1)}$ with the Schmidt measure $m = \lceil \log_2(k) \rceil$ the low-rank state preparation has a complexity of

| Condition | CNOT count |
|-----------|------------|
| $0 \leq m < n_A$ | $O(2^{m+n_B})$ |
| $m = n_A$ | $O(2^n)$ |

Theorem 1 has a concrete proof that leads to the low-rank state preparation algorithm. The proof is given in the Methods sections. A more detailed description and analysis can be found in the supplementary information.

The LRSP algorithm also allows a low-rank approximation in exchange for an error. Limiting the Schmidt rank imposes an entanglement reduction, as this action reduces both purity and e-bits. The fidelity loss can be used to quantify the loss by the approximation.

**Definition 1.** Given the low-rank parameter $r$, the approximated state is denoted as

$$|\psi^{(r)}\rangle = \sum_{i=0}^{r} \sigma_i |i_A\rangle |i_B\rangle$$

with coefficients for $1 \leq r \leq k$, i.e. $\sigma_j = 0$, $r < j \leq k$. We define the fidelity loss as

$$l(r, |\psi\rangle) := (1 - |\langle \psi, \psi^{(r)} \rangle|^2).$$

**Proposition 2.** Let $1 \leq r \leq k$ be the low-rank approximation parameter, then the fidelity loss is given by

$$l(r, |\psi\rangle) = \sum_{i=r+1}^{k} |\sigma_i|^2$$

Bounded Approximation Algorithm

The approach of the LRSP-algorithm, in particular its approximation, can be generalized by a recursive algorithm that provides an approximation up to a given error bound $l_{\text{max}}$ – if possible – and thus saves the use of
CNOT gates while providing a guarantee of proximity to the original state. It is a bounded approximation error state preparation algorithm (BAA) that potentially has a classical exponential run-time with respect to the number of qubits of the state. As it is a branch-and-bound algorithm using breadth-first search the complexity usually converges faster [39, 40]. The full set of pseudocode which describes the algorithm is printed in the supplementary information. In what follows, we want to outline the core principles of the algorithm.

The approximate state preparation algorithm works as follows. Any (pure) \( n \)-qubit quantum state \( |\psi\rangle \) will allow for a total of

\[
B(n) = \sum_{t=1}^{\lfloor n/2 \rfloor} \binom{n}{t}
\]

bipartitions (also called branches), each is written as \( b \in [1 \ldots B(n)] \) with the Schmidt number bounded by \( k(b) = \min\{\dim(H_1^b), \dim(H_2^b)\} \). By letting the low-rank parameter \( 1 \leq r \leq k(b) \) be \( r = 1 \), each bipartition creates a disentangled pair of smaller states. Combining this pair by the tensor-product returns an approximation of the original state. Its fidelity loss will be calculated as given by Proposition 2. Additionally, it is possible by the complexity analysis of Theorem 1 to predict the number of CNOT gates saved.

Starting from \( |\psi\rangle \in \mathcal{H} \), the algorithm is branching into all possible bipartitions, \((H_1, H_2^b)\). Recursively, each partition \( H_i^b \) can in turn be branched once again, as long as \( H_i^b \) \((i = 1, 2) \) is not one single qubit or the total fidelity loss has not exceeded the error bound \( l_{\text{max}} \). In order for the search-algorithm to work, each node needs to be able to know the total fidelity loss, the total saved number of CNOT gates and its partitioned Hilbert spaces. Due to this structure, the algorithm terminates in the worst case with an exponential number of steps. These mentioned properties are summarized.

**Lemma 1.** The fidelity and the saved CNOT gates of each branch can be recursively calculated.

**Proposition 3.** The BAA-Algorithm’s search tree has \( n \) levels, and therefore terminates. The time and space complexity is exponential in \( n \) in its worst case.

The proof is given in the Methods section. A tree of an example three-qubit state is shown in Fig. 1. With an \( l_{\text{max}} = 0.1 \) only one bipartition is possible within this error bound, and three CNOT gates are saved.

Albeit the fact that the BAA algorithm uses low-rank approximations with \( r = 1 \) to fully disentangle the bipartitions (Fig. 5 and Fig. 1), it is possible to include low-rank approximations with \( r > 1 \) when complete disentanglement is no longer within \( l_{\text{max}} \). This allows fine-tuning the partial disentanglement of states, as each value of the parameter \( r \) may achieve reductions in the number of CNOTs that would not be possible with the original approach, producing additional branches from the tree nodes. Therefore, this advantage comes with a larger search space, and increases the cost of the algorithm. For a fully detailed description of this algorithm, and mathematical details, we refer to the supplementary information.

While the proposed approximate state preparation algorithm reduces the quantum circuit complexity, the BAA has an exponential preprocessing cost. The naive (“brute-force” breadth-first) algorithm is already a solution for those researchers who need to work on small-qubit experiments and currently have no alternatives to an efficient quantum initialization on noisy devices. But a scalable workaround is necessary. One way is to use a greedy approach [41, 42] that seeks to reduce the computational cost, making BAA a useful solution for general problems.

The greedy strategy proposes that branching from a node is limited to a qubit-by-qubit analysis, selecting only one representative of the partitions of size \( k \) where \( 1 \leq k \leq \lfloor n/2 \rfloor \). The increment in the partition size is done by choosing the qubit with the lowest fidelity loss when removed from the remaining entangled subsystem. As an example, on a seven qubit state, the best 1 vs. 6, 2 vs. 5 and 3 vs. 4 partition are attempted, and the best child of each is then propagated in the recursion. This approach implies an exponential reduction in the number of bipartition combinations, of which one is chosen, producing only one branch from each node. Although this method will have no guarantee to end up in the optimal approximation, these methods are efficient and usually good enough for applications. A numerical survey on multiple seven qubit random circuits generated quantum states [43] shows that low \( l_{\text{max}} \) (presumably \( \leq 0.2 \)) result into less CNOT-gates saved. In the methods section, we explore this with data.

**On Product State Approximations**

In view of treating quantum state approximation, one could be naturally inclined to ask for the best product state approximation. This is indeed a measure for entanglement, the so-called geometric measure of entanglement. It was first introduced by Shimony [29] and refined by Barnum & Linden [44]. The definition for this measure of entanglement is given by

\[
E_g(|\psi\rangle) = \min_{|\phi\rangle} \| |\psi\rangle - |\phi\rangle \| \tag{5}
\]

where the minimization is over all states \(|\phi\rangle\) that are product states [45], i.e., \(|\phi\rangle = \otimes_{\ell=1}^{n} |\phi^{(\ell)}\rangle\) with each \(|\phi^{(\ell)}\rangle\) being a local state. That this value is a “useful” measure of entanglement was shown by Wei et al. [46], it is an entanglement monotone.

The geometric measure of entanglement is related to the bounded approximate algorithm (BAA) in the following way. Theoretically, the greatest fidelity loss that any quantum state can experience under the BAA should
The geometric measure of entanglement gives an indication, how bad an approximation can get. This shows that low-entanglement states can be encoded with little loss as a product state. However, looking at the BAA algorithm, the sequence of nested SVDs is not optimal, and there can always be resulting product states of the BAA with a higher fidelity loss.

FIG. 1: An example of a three-qubit state that is disentangled into product states (leaves). It starts with one vector in the root and then all $B(3) = 3$ bipartitions are branched off, creating each two states. From there, the larger subspace has two qubits, so it should have $B(2) = 2$ branches. But in this case, the second branch is equal to the complement of the first, so it is redundant and omitted. Each node has a fidelity loss (red column) and the number of CNOT gates saved (blue column) using the proposed algorithm. The sub-indices of the vectors are the qubit partitions associated with each state. Adding the weights from root to any leaf gives the total fidelity loss of $\approx 0.307$, meaning that the reduced product state to its original state have an overlap of $\approx 0.693$, but with a saving of 4 CNOT gates. The adaptive approximation algorithm goes to ask if we can save some CNOT gates with an acceptable fidelity loss. If $t_{\text{max}} = 0.1$, there is the second bipartition from the root node (node 2) that introduces only a fidelity loss of $\approx 0.058$ by simultaneously saving three CNOT gates.

Never be higher than it. Therefore, the geometric entanglement gives an indication, how bad an approximation can get. This shows that low-entanglement states can be encoded with little loss as a product state. However, looking at the BAA algorithm, the sequence of nested SVDs is not optimal, and there can always be resulting product states of the BAA with a higher fidelity loss.

Naturally, one is interested in computing the product state that maximizes the fidelity in the attempt to calculate the geometric measure of entanglement. The first such attempt published in scientific literature calculates the measure with the help of a non-linear eigenvalue problem [46]. More efficient algorithms have been devised in the meantime. As was pointed out by Teng [45] one can use both methods, Tensor rank decomposition and Tucker decomposition [47].

Yet, calculating the geometric measure of entanglement can efficiently and with high probability be achieved by applying the Tucker decomposition of a quantum state, $|\psi\rangle$ interpreting it as a tensor of each mode being 2-dimensional with rank one. The so-called “core” [48] is then simply a complex number $\lambda$ and the geometric measure of entanglement is $E(\psi) = 1 - |\lambda|^2$ [45]. Each factor is a vector, and they constitute the single qubit states that can be prepared together as a product state. The state created in such a way is then called $|\psi\rangle$, and its overlap with the original state is $\langle \psi | \psi \rangle = \lambda$.

The particular case where the BAA partitions are split in half and only one branch selected is equivalent to the Hierarchical Tucker Format (HTF) with a binary dimension tree (called Canonical Dimension Tree) [49]. The HTF can be used to estimate the maximum possible fidelity loss with an error less or equal to $\sqrt{2n - 3} |A - A^{\text{best}}||$, where $A^{\text{best}}$ is the best approximation of the tensor $A$, and $n$ is the dimensionality of the problem. The number of bipartitions performed to estimate the maximum fidelity loss is linear in the number of qubits. Since BAA performs the SVD from already truncated tensors, the cost of HTF is given by Lemma 26 of Ref. [49].

The Loading Problem on NISQ Devices

The Bounded Approximation Error Algorithm was designed to solve a class of problems on NISQ devices, namely loading data. To solve (or avoid) the loading problem [4], we create an approximation of the quantum state with the goal of using a less entangled state. In our experimental evaluation – refer to the supplementary section – we achieved an end-to-end improvement in examples from stochastics and machine learning. In the following, we will demonstrate such an end-to-end performance improvement on actual quantum devices and noisy simulators.
The BAA proves to be efficient when working with specific classes of quantum states found in the quantum finance \cite{9, 14, 17}. Indeed, encoding probability distributions on the amplitudes of a 7-qubit quantum state show that even a modest fidelity loss ($\leq 0.02$) results in a significant reduction in the number of CNOTs. The experimental setup is as follows. Four types of probability distributions are used: normal, log-normal, Laplace, and semicircular. Preparing the exact states, that is, no approximation and $l_{\text{max}} = 0.0$, each of these distributions need circuits with 120, 151, 28, and 143 CNOTs, respectively. On the other hand, if a maximum fidelity loss of $l_{\text{max}} = 0.02$ is allowed, the effective fidelity loss achieved by the BAA to encode the approximate state for each distribution is the closest possible to $l_{\text{max}}$, i.e., 0.0089, 0.02, 0.0081, and 0.0111. With such approximation, the BAA can encode all distributions using only seven CNOTs. This relatively low introduced error compared to the extreme reduction of entangling operations results into better results in terms of mean-absolute-error (MAE). Indeed, when executing an experiment on the ibmq_jakarta, the high number of CNOTs to initialize the exact state obscure the probabilities whereas the BAA approximation keep distribution specific features closer to the actual distribution, see Figures 2(a)–2(h).

In quantum machine learning, one of the best-known applications is to train a parametrized quantum circuit in order to learn a decision hyperplane \cite{50} and thus be able to classify binary problems. In this setting, the input data, usually $x \in \mathbb{R}^N$ is real-valued, is encoded by a feature map into a quantum state. Havlicek et al. \cite{50} have hypothesized that their feature map possesses quantum advantage properties, yet any feature map, including a regular state preparation \cite{8, 51, 52}, is a valid option. We investigate, experimentally, the feasibility to use the BAA algorithm to partially disentangle states and reduce their complexity. The classification accuracy of the quantum variational classifier (VQC) \cite{50, 53} is used as a metric to evaluate the effectiveness of state preparations. Ideally, the accuracy should be at a nominal level, which can be established with a noiseless simulation. The experiment on a NISQ device, however, will undoubtedly introduce noise and will reduce the classification accuracy. Still, the BAA is expected to produce better results on a noisy device due to the lower circuit complexity. Indeed, the reduction of entangling gates, shows that even
...is necessary but beyond the scope of this work. In the supplementary information we describe our thought arbitrary quantum state. But how complex the approx-
devices for algorithms that rely on loading data into an
number of noisy operations. A low-rank approximation
tion in the complexity of the circuits, which decreases the
close as possible to the original. This leads to a reduc-
tion in the complexity of the circuits, which decreases the
number of noisy operations. A low-rank approximation
of the state can make the final circuit feasible on noisy
devices for algorithms that rely on loading data into an
arbitrary quantum state. But how complex the approx-

**DISCUSSION**

Previous state preparation algorithms require $O(2^n)$
CNOTs to initialize an arbitrary quantum state. However,
these algorithms do not consider the amount of en-
tanglement. This work shows that the complexity of state
preparation is related to entanglement through numeri-
cal measures of pure state entanglement based on the
square of Schmidt coefficients. These measures are di-
rectly related to the number of CNOTs required for state
initialization. The relation between entanglement and state preparation gives a better understanding of the ini-
itialization complexity of classes of states.

The second result of this work is the low-rank approx-
imation for the initialization of quantum states. Experi-
ments with seven qubits quantum devices show that mea-
surements of the state generated by the low-rank approx-
imation circuits are closer to the expected results than the
exact initialization algorithm case. It occurs because the
algorithm finds a state with reduced entanglement as
close as possible to the original. This leads to a reduc-
tion in the complexity of the circuits, which decreases the
number of noisy operations. A low-rank approximation
of the state can make the final circuit feasible on noisy
devices for algorithms that rely on loading data into an
arbitrary quantum state. But how complex the approx-
imate state must be to achieve the best result depends
on the characteristics of the quantum device itself. From
this perspective, our approach has the potential to char-
acterize quantum devices as to their ability to handle the
complexity of state preparation. A visual example of this
is given in the suppl. information section 2.2 where we
compare a zebra image and its approximations on noise
simulations.

A variant of this approach permits a bounded approx-
imation error state preparation algorithm (BAA) which
is useful if a certain approximation error must be ob-
served. Low entanglement states can usually be created
by product states with a low approximation error, while
high-entanglement state remain hard to create, but still
for medium entanglement there are considerable improve-
ments in terms of saved CNOT gates. The VQC exam-
ple application shows that BAA allows adjustment of the
input data according to the device's capability to man-
age entanglement, improving the classifier’s performance.
Additionally, VQC applications have the characteristic of
loading the same input vector many times, which amor-
tizes the computational time of the BAA preprocessing.

Recent approaches for approximating quantum states
use variational quantum circuits in which parametrized
single qubits and entangling operations are intertwined
according to a gradient descent method while minimizing/maximizing a loss function, usually
taken from the measurement output of an experiment. In
particular noteworthy is the qGAN method [14, 56, 57]
and the AAS method [15] which have been proposed as a
viable alternative to standard state preparation in or-
der to achieve good results in the NISQ era. We found
in a preliminary comparison that the BAA is superior
for several reasons, however, we note that an exhaustive
study is necessary but beyond the scope of this work. In
the supplementary information we describe our thought

![Diagram](image-url)
FIG. 4: Application of the BAA to machine learning problems helps to increase classification accuracy. (a) The effect of BAA on the input data (digits) for different maximum fidelity loss parameters. (b) The classifier’s feature map $U_\phi$ encodes the data in the amplitudes of a quantum state using BAA. (c) The CNOT-gate complexity and circuit depth are compared against the multiplexor algorithm [52] and the BAA. At the maximum fidelity loss of 0.5 the classifier delivers the best accuracy.

The process in detail, to summarize, the BAA is based on purely classical algorithms being very efficient in today’s hardware with standard methods to apply for approximations (instead of a full search, we have greedy and Monte-Carlo as options), the method does not suffer from initial conditions failures (such as barren plateaus) and finally it is a mathematically moored method, unlike the variational methods, which are heuristics. We note that variational circuits seem to have worsening performance the bigger the Hilbert space grows, and we see this too with the BAA, as the memory consumption increases exponentially and thus the application of the SVD get more and more infeasible. Yet, looking at the results and the existing literature, we speculate that a direct comparison of qGAN and BAA as approximate amplitude encoding algorithms may not be useful. Rather, it may be hypothesized that the BAA relates as a meta-algorithm that supports variational quantum circuits in finding the best performing ansatz for the variational circuit design, thus being complementary. In the following way, we see this unfolding as an advantage: currently we use a state preparation algorithm as “kernel” on smaller subspaces. If this “kernel” algorithm can now be replaced through a variational approach instead, we hypothesize that the downsides of variational methods can be circumvented as the Hilbert space could stay in a manageable size, thus combining advantages of both approaches. Of course, there is still a limit we expect to hit at about 30 - 35 qubits at which point the memory consumption is beyond today’s most powerful HPC clusters. We conclude that the BAA is here to support the NISQ era. Indeed, this discussion warrants in depth studies, which should be considered for future work.

This work also leaves some open questions. How do quantum machine learning, solving systems of linear equations, principal component analysis, and systems of differential equations perform with low-rank state preparation? As a future investigation, one could verify if the quality of a quantum algorithm reduces when it operates on an approximated quantum state and if this could allow an advantage in solving problems that have state preparation as a bottleneck. One possible result related to the product state approximation is establishing the error bound on estimating the geometric measure of entanglement for a non-binary dimension tree. It would be the error of the BAA using brute force (without partial disentanglement). Partial disentanglement seems to be a novel feature.

Another very promising application is enhancing quantum adiabatic state preparation [58–61] by preparing a low-entanglement bounded approximation error state and find its Hamiltonian for which it is an eigenvalue and then apply the adiabatic theorem. The goal would be to potentially skip spectral gap bottlenecks and therefore make either the state preparation possible or reduce the adiabatic time significantly. In fact, a possible way forward could be to reach high-entanglement states by this approach. It remains unclear how this could actually be achieved, but the applications would be obviously beneficial.
METHODS

Proof of Theorem 1

Theorem 1 establishes the CNOT gates count needed when a low-rank representation of a state can be found using the Schmidt decomposition, as well as approximating the state by truncating the Schmidt coefficients. The number of CNOT gates needed is a direct consequence of the Plesch algorithm.

Proof. If the Schmidt measure is \( m = \lceil \log_2(k) \rceil < n_A \), the operator \( S \) (Fig. 5a) initializes a state with \( m \) qubits in phase 1 (Fig. 5b and Fig 5c), instead of \( n_A \) qubits (Fig. 5d). The second phase requires \( m \) CNOT gates, as they are not needed where the control qubit is \( |0\rangle \).

In this circuit configuration, \( m \) is the Schmidt measure and quantifies the entanglement between subsystems \( \mathcal{H}_A \) and \( \mathcal{H}_B \). Finally, in phases 3 and 4, LRSP is made possible by replacing the unitary decomposition with isometry decomposition [38] (Fig. 5c). The number of columns in each isometry is equal to the Schmidt rank.

In Plesch’s original algorithm [8], the number of CNOT gates is \( O(2^n) \) and considers only the amount of qubits \( n \). The LRSP algorithm (Suppl. Algorithm 2) complexity is defined considering the number of qubits \( n \) and the Schmidt measure of entanglement \( m \) in units of e-bits. The matrices \( U \) and \( V^T \) are isometries \( 2^m \)-to-\( 2^{n_A} \) and \( 2^m \)-to-\( 2^{n_B} \) which require \( O(2^{n_A}) \) and \( O(2^{n_B}) \) CNOTs. The number of CNOTs of the complete LRSP circuit is \( O(2^{n_B}) \) because the cost of the isometry \( V^T \) dominates the cost of the algorithm \( (n_B \geq n_A) \). The best-case occurs when the bipartition is not entangled \( |\psi\rangle = |\psi_A\rangle |\psi_B\rangle \). Therefore, if rank \( = 1 \) there are no operations in phase 1 (to encode singular values), no entanglement between the two subsystems in phase 2 (since there is 0 e-bits between the subsystems), and there are one \( 1 \)-to-\( 2^{n_A} \) and one \( 1 \)-to-\( 2^{n_B} \) isometries in phases 3 and 4 (Fig. 5b) which is equivalent to two parallel sub-state preparations in \( n_A \) and \( n_B \) qubits. If all qubits of the state are separable, applying recursively the same algorithm to prepare the sub-states in phases 3 and 4 generates a circuit without CNOT gates. If the state is initially separable but not all sub-states of the recurrence (some qubits are not separable), the cost of the state preparation is \( O(2^n) \), where \( n_e \) is the number of qubits of the higher entangled subsystem. The worst case occurs when \( m = n_A \), recovering the \( O(2^n) \) complexity of the original Plesch circuit (Fig. 5). The number of CNOT gates is discussed in the Supplementary Information in greater detail.

Proof of Proposition 2

How big an error will the truncation of Schmidt numbers introduce to the fidelity of a state? Proposition 2, that establishes this fact, is proven here.

Proof. Given the low-rank parameter \( r \), the main idea of Suppl. Algorithm 2 is to truncate the coefficients for \( 1 \leq r \leq k \), i.e. \( \sigma_j = 0 \), \( r < j \leq k \). Then a renormalization is necessary, which means that

\[
Z = \sqrt{\frac{r}{\sum_{j=1}^{r} |\sigma_j|^2}} = \sqrt{1 - \frac{k}{\sum_{j=r+1}^{k} |\sigma_j|^2}} \tag{6}
\]

with \( \sigma_i^j = \frac{\sigma_i}{2}, i = 1, \ldots, r \) and the state that emerges from this description is written as \( |\psi^{(r)}\rangle = \sum_i \sigma_i^j |\alpha_i\rangle \otimes |\beta_i\rangle \). The squared overlap (fidelity) between \( \psi \) and \( \psi' \) is then

\[
|\langle \psi, \psi^{(r)} \rangle |^2 = \frac{1}{Z^4} \left( \sum_{i=1}^{r} |\sigma_i|^2 \right)^2 = \frac{2^4}{Z^4} = Z^2. \tag{7}
\]

Then the quantity we seek can be directly be computed:

\[
l(r, |\psi\rangle) = (1 - |\langle \psi, \psi^{(r)} \rangle |^2)
\]

\[
= (1 - Z^2) \tag{8}
\]

which we wanted to show.

Proof of Lemma 1

A small result with an impact is the fact that the fidelity/fidelity loss can be recursively calculated, as well, and trivially, the CNOT gate savings from one branch to another. The lemma is important for the computation of the search tree in the BAA algorithm, as it is possible to parallelize it.

Proof. The fidelity loss on the bipartition/branch \( b \) will be calculated as given by Proposition 2. However, one must show that the total fidelity loss over several steps in the tree can be recursively calculated. Let \( l_{b_i} \) and \( l_b \) be the fidelity loss of the parent and child node, respectively. Then the fidelity of both is \( (1 - l_{b_i})(1 - l_b) \), hence \( l_{tot} = 1 - (1 - l_{b_i})(1 - l_b) \). More generally, \( l_{tot} = 1 - \Pi_{i=1}^k (1 - l_b_i) \) for a path on the tree \( b_1, \ldots, b_t \).

Proof of Proposition 3

When spanning the search tree (see Suppl. Information Pseudocode 1), each child of a parent node will be created by choosing exactly one state and doing exactly one bipartition. The bipartitions of our focus are the one qubit vs. the rest ones. Say, among the \( k \) states of the parent node, \#1 is selected to be partitioned, with \( \dim \mathcal{H}_1 = n_1 \). The split of one qubit vs. rest is done, and in this manner \( n_1 - 1 \) levels will be traversed until this state has been partitioned to a tensor product of single
FIG. 5: Schematics of the low-rank approximation algorithm ($n = 4$ and subsystem $\mathcal{H}_A = \{0,1\}$). (a) Block diagram circuit. Operator $S$ is responsible for encoding $2^{n_A}$ ($n_A = n/2$) SVD singular values. In this example, operator $S$ encodes a maximum of four amplitudes (two qubits). Operator $T$ is a sequence of CNOTs, controlled by the first half of qubits and targeting the second half (one by one). Operators $U$ and $V$ are the two SVD unitaries. For full-rank ($m = n_A$, in units of e-bits), $U$ and $V^T$ are encoded as unitaries of dimension $2^{n_A} \times 2^{n_A}$. For a lower rank ($m < n_A$), the operators are encoded as isometries of dimension $2^{n_A} \times 2^m$. When $m = 0$ (rank = 1), the isometries of dimension $2^{n_A} \times 1$ are equivalent to amplitude encoding with input vectors of length $2^{n_A}$. Plesch’s original work describes the four individual phases. (b)(c)(d) Detailed views of (a) for rank = 1 ($m = 0$), rank = 2 ($m = 1$) and rank = 4 (full-rank, $m = 2$). In this example, phase 1 uses Möttönen’s state preparation but could use any amplitude encoding algorithm, including the low-rank state preparation algorithm.

qubit states. Meanwhile, all $k - 1$ original states were kept as they were. Now the partition #2 is selected, and in turn takes $n_2 - 1$ steps to a tensor product of single qubit states. This goes on with all the remaining $k - 2$ partitions. So in total, there are

$$\sum_{i=1}^k (n_i - 1) = n - k$$  \hspace{1cm} (9)

steps until all $k$ partitions have been partitioned into tensor products of single qubit Hilbert spaces. As this is the longest path possible, the BAA algorithm terminates. Now, we show the runtime complexity of the BAA algorithm.

Lemma 2. Given a $k$-fold partitioning of $\mathcal{H}$, denotes as $(\mathcal{H}_1, \ldots, \mathcal{H}_k)$, the number of children of this node is

$$B_k(n) = \sum_{i=1}^k B(\dim \mathcal{H}_i)$$ \hspace{1cm} (10)

As a special case, we see $k = 1$, which is directly $B(\dim \mathcal{H})$.

Proof. Given $k$ already partitioned subspaces, the search algorithm selects one of the partitions $(\mathcal{H}_i, i = 1, \ldots, k)$ and carries out all $B(\dim \mathcal{H}_i)$ branches, while leaving all the other $k - 1$ original partitions intact. This is then repeated, so the total number of children this node has as claimed. 

After each step, the number of partitions thus changes from $k \mapsto k + 1$ as one partition is created. Of course, the sum of all dimensions must be equal to the dimension...
of the original Hilbert space $n$. Thus, each level of the search tree is identified by the number $k$ of partitions. The maximum number of levels is therefore always $n$. The total number of nodes is then computed as follows.

The first level has exactly 1 node. The second level has $B_1(n)$ nodes. The third level has $\sum_{b_1=1}^{B_1(n)} B_2(n; b_1)$ and the fourth level $\sum_{b_1=1}^{B_1(n)} \sum_{b_2=1}^{B_2(n; b_1)} B_3(n; b_2)$ up until the $n^{th}$ level, we have,

$$\sum_{b_1=1}^{B_1(n)} \cdots \sum_{b_{n-1}=1}^{B_{n-1}(n; b_{n-2})} B_n(n; b_{n-1})$$

(11)

This is clearly exponential in $n$.

**Greedy Algorithm’s Performance**

The greedy algorithm reduces the search-problem exponentially, yet this will necessarily reduce its ability to find a good CNOT reduction. The advantage over the breadth-first (brute-force) strategy is that it will never exceed the approximation configured, but the downside is that the CNOT complexity to attain this approximation could be far from optimal. We use seven qubit random circuits [43] of linear depth to generate 1000 quantum states and apply different strategies and circuits [43] of linear depth to generate 1000 quantum states and apply different strategies and breadth-first (brute-force) strategy is that it will never find a good CNOT reduction. The advantage over the breadth-first strategy performs less well for low $l_{max}$, see Fig. 6.

Regarding the approximation quality, both the greedy and brute-force approaches find approximations that to each other scatter quite evenly, if $l_{max}$ is high, this scattering is less pronounced. The search of the best node will always reject a solution that exceeds the $l_{max}$ value and it is always helpful to reduce the fidelity loss, but the cost of CNOT gates will outgrow the impact on NISQ devices. We can conclude on this survey that the greedy algorithm will miss optimality for low $l_{max}$. To improve this, we have several suggestions as future research.

**Experiments on Probability Density Functions**

To encode a continuous probability density function (PDF) into the amplitudes of a quantum state it is necessary to construct its discrete analog. In general, this construction is based on preserving one or more properties of the continuous distribution [62].

There are several methods by which a discrete random variable can be constructed from a continuous one. Here we employ Methodology-V described in Ref. [62] where the cumulative distribution function (CDF) of a discrete random variable $Y$ maintains the form of the CDF of a continuous random variable $X$. The probability mass function (PMF) of $Y$ is built from the CDF of $X$ $F_X(x) = Pr(X \leq x)$ and is given by

$$P(Y = k) = F_X(k + \delta) - F_X(k - [1 - \delta])$$

(12)

where $0 < \delta < 1$ and $k = \{0, 1, 2, \ldots \}$.

In the experiments the interval $0 \leq x \leq 20$ is divided into $2^7$ discretization points. Therefore, the distance between consecutive points is $d = 20/(2^7 - 1)$. By choosing $\delta = d/2$, the parameters of Eq. (12) are set to

$$P(Y = k) = F_X(k + d/2) - F_X(k - d/2)$$

(13)

for $k = \{0, d, 2d, \ldots, (2^7 - 1)d\}$.

Equation (13) is used to generate the discrete probability distributions analogous to the continuous ones char-
acterized by the following PDFs:

\[ f(x) = \frac{1}{4} \exp\left(-\frac{|x-10|}{2}\right) \]

\[ f(x) = \frac{\exp\left(-\frac{(x-10)^2}{2}\right)}{2\sqrt{2\pi}} \]

\[ f(x) = \frac{1}{x\sqrt{2\pi}} \exp\left(-\frac{\log^2\left(\frac{x}{2}\right)}{2}\right) \]

\[ f(x) = \frac{1}{4\pi} \sqrt{1 - \left(\frac{x-10}{8}\right)^2} \]

Respectively Laplace, Normal, Log-Normal, and Semicircular.

| distribution     | mean     | variance | skewness | kurtosis |
|------------------|----------|----------|----------|----------|
| Log-Normal       | 3.297    | 18.68    | 6.185    | 110.9    |
| Laplace          | 10.0     | 8.0      | 0.0      | 3.0      |
| Normal           | 10.0     | 4.0      | 0.0      | 0.0      |
| Semicircular     | 10.0     | 16.0     | 0.0      | -1.0     |

**TABLE I:** First four moments of the probability distributions.

Each point of the discretization corresponds to an amplitude of the state vector. Note that, given the conservation of probability, each amplitude must be the square root of the PMF at these points. The result of the discretization is employed to construct Figures 2 and 3.

The state vectors of the distributions were encoded via BAA and estimated by measurements on actual quantum devices. All experiments make use of the breadth-first search (brute force) strategy and partial disentanglement of states enabled. Each result is an average of 10 runs with 8192 shots each. The experiments leading to Figure 2 were performed at points \( l_{\text{max}} = \{0.0, 0.002\} \) using device ibmq_perth, while those leading to Figure 3 were performed at points \( l_{\text{max}} = \{0.0, 0.005, 0.01, \ldots, 0.055\} \) using devices ibmq_jakarta and ibmq_casablanca.

**Experiments on VQC**

These experiments evaluate the BAA using brute-force strategy and partial disentanglement of states as a quantum feature map [50] to encode data in the amplitudes of an entangled state [8, 51, 52]. The accuracy of a quantum variational classifier (VQC) [50, 53] is used as a metric to evaluate the state preparation.

The classifier model is a hardware efficient parameterized circuit divided into two layers that form a block [50, 63-65]. The first layer contains parameterized single qubit rotation gates, followed by the second with two-qubit entangling gates. Usually, the block is repeated to achieve better results [53]. The circuit is completed with an additional layer of rotations. Multiple measurements are carried on any qubit to approximate an expectation value. The circuit model TwoLocal from the Qiskit circuit library was chosen for this work. It is a fully entangled model composed of \( R_y \), \( R_z \), and controlled-Z gates with no block repetition.

The Optical Recognition of Handwritten Digits Data Set was used [55]. A binary dataset with 100 samples was extracted from the original dataset. The two classes of the new dataset are 0 and 1, each with 50 samples of 64 attributes. Mean test accuracy is computed on a random initialization for each feature map configuration.

Qiskit’s VQC implementation was modified to accept the BAA encoding as a feature map and used to carry on the classical-quantum hybrid approach, in which the optimization procedure is processed on a classical computer to determine a set of parameters for the parameterized quantum circuit [64, 66, 67]. Simulations of the hybrid classification algorithms were performed using Qiskit’s Aer simulator with approximately 4500 circuit executions for each experiment configuration – 1024 shots per execution. For the experiments with noisy simulation, the noisy model and the coupling map were imported from ibmq_jakarta device. Ten samples of each class of the binary dataset were used as a test set and the remaining 80 samples (40 for each class) as a training set. As preparation for amplitude encoding, each data vector element of the dataset was standardized for a Gaussian around zero with unit variance, rescaled within the range \([0, 1]\), and normalized. Our simulation employs the Simultaneous Perturbation Stochastic Approximation (SPSA) [63, 68] for the optimization process with a mini-batch size of \(1/10\) of the training set size.

**ACKNOWLEDGMENTS**

This research is supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (Grant No. 406854/2021-1, 312354/2021-5, and 162052/2021-9), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Finance Code 001 and Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco - FACEPE (Grant No. IBPG-0834-1.03/19 and APQ-1229-1.03/21). We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

**AUTHOR CONTRIBUTIONS**

A.J.S. conceived the project, A.J.S., C.B. and I.F.A. developed the algorithms, C.B. and I.F.A. developed the theory, I.F.A and I.C.S.A. performed the experiments with the quantum generative neural network while I.F.A. performed all other experiments. C.B. and I.F.A. created the illustrations. All authors discussed the results and contributed to the manuscript.
COMPETING INTERESTS

C.B. is managing director and stakeholder at data cybernetics ssc GmbH. The other authors declare no competing interests.

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, et al., Quantum supremacy using a programmable superconducting processor, Nature 574, 505 (2019).
[2] S. Aaronson, Read the fine print, Nature Physics 11, 291 (2015).
[3] F. Leymann and J. Barzen, The bitter truth about gate-based quantum algorithms in the nisq era, Quantum Science and Technology (2020).
[4] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature 549, 195 (2017).
[5] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Physical review letters 103, 150502 (2009).
[6] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nature Physics 10, 631 (2014).
[7] V. Bergholm, J. J. Vartiainen, M. Möttönen, and M. M. Salomaa, Quantum circuits with uniformly controlled one-qubit gates, Physical Review A 71, 052330 (2005).
[8] M. Plesch and Č. Brukner, Quantum-state preparation with universal gate decompositions, Physical Review A 83, 032302 (2011).
[9] E. Malvetti, R. Iten, and R. Colbeck, Quantum circuits for sparse isometries, Quantum 5, 412 (2021).
[10] I. F. Araujo, D. K. Park, F. Petruccione, and A. J. da Silva, A divide-and-conquer algorithm for quantum state preparation, Scientific Reports 11, 6329 (2021).
[11] I. F. Araujo, D. K. Park, T. B. Ludermir, W. R. Oliveira, F. Petruccione, and A. J. da Silva, Configurable sublinear circuits for quantum state preparation, arXiv preprint arXiv:2108.10182 (2021).
[12] X.-M. Zhang, M.-H. Yung, and X. Yuan, Low-depth quantum state preparation, Phys. Rev. Research 3, 043200 (2021).
[13] D. K. Park, F. Petruccione, and J.-K. K. Rhee, Circuit-based quantum random access memory for classical data, Scientific Reports 9, 3949 (2019).
[14] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, npj Quantum Information 5, 103 (2019).
[15] K. Nakaji, S. Uno, Y. Suzuki, R. Raymond, T. Onodera, T. Tanaka, H. Tzukia, N. Mitsuda, and N. Yamamoto, Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicator (2021).
[16] G. Marin-Sanchez, J. Gonzalez-Conde, and M. Sanz, Quantum algorithms for approximate function loading (2021).
[17] F. Mozafari, H. Riemen, M. Soeken, and G. De Micheli, Efficient boolean methods for preparing uniform quantum states, IEEE Transactions on Quantum Engineering 2, 1 (2021).
[18] C. E. Mora and H. J. Briegel, Algorithmic complexity of quantum states, International Journal of Quantum Information 04, 715 (2006).
[19] C. E. Mora and H. J. Briegel, Algorithmic complexity and entanglement of quantum states, Phys. Rev. Lett. 95, 200503 (2005).
[20] R. Raussendorf and H. J. Briegel, A one-way quantum computer, Phys. Rev. Lett. 86, 5188 (2001).
[21] J. T. Barreiro, P. Schindler, O. Gühne, T. Monz, M. Chwalla, C. F. Roos, M. Hennrich, and R. Blatt, Experimental multiparticle entanglement dynamics induced by decoherence, Nature Physics 6, 943 (2010).
[22] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A 54, 1098 (1996).
[23] A. Steane, Multiple-particle interference and quantum error correction, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences 452, 2551 (1996).
[24] R. Cleve, D. Gottesman, and H.-K. Lo, How to share a quantum secret, Phys. Rev. Lett. 83, 648 (1999).
[25] W. Dürr and J. I. Cirac, Multiparty teleportation, Journal of Modern Optics 47, 247 (2000).
[26] C. Eckart and G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1, 211 (1936).
[27] M. Schuld and F. Petruccione, Supervised Learning with Quantum Computers, 1st ed. (Springer Publishing Company, Incorporated, 2018).
[28] C. Blank, D. K. Park, and F. Petruccione, Quantum-enhanced analysis of discrete stochastic processes, npj Quantum Information 7, 1 (2021).
[29] A. Shimony, Degree of Entanglement, Annals of the New York Academy of Sciences 755, 675 (1995).
[30] R. Josza and N. Linden, On the role of entanglement in quantum-computational speed-up, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences 459, 2011 (2003).
[31] N. Stamatopoulos, D. J. Egger, Y. Sun, C. Zoufal, R. Iten, N. Shen, and S. Woerner, Option Pricing using Quantum Computers, Quantum 4, 291 (2020).

DATA AVAILABILITY

The site https://www.cin.ufpe.br/~ajsilva/qclib contains all the data and software generated during the current study.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to Carsten Blank.
1 BAA and qGAN comparison

In the main text, we performed experiments to load probability distributions into a quantum state. One can also load an approximate probability distribution of a dataset \( D \) into a quantum state with Quantum Generative Adversarial Networks (qGANs) \[^2\] (with some restrictions, as pointed out in Ref. \[^2\]). In this section, we present a conceptual background of classical and quantum Generative Adversarial Networks and perform an experiment comparing the states loaded by a qGAN and BAA methods.

A Generative Adversarial Network \[^2\] is a machine learning model composed of a neural network pair often referred to as generator and discriminator. The discriminator output \( D(x) \in \{0, 1\} \) distinguishes real and synthetic data. The generator \( \mathcal{G} \) generates random synthetic data \( \hat{x} = \mathcal{G}(z) \), from any uniform noise \( z \). To sample synthetic data closer to the real dataset, one trains the generator to maximize the discriminator error. In the case of qGANs, this means optimizing the generator’s quantum circuit parameters so it can load the intrinsic distribution of the data into a quantum state \[^2\].

Discriminator and generator networks are trained competitively. The generator is trained to maximize \( \mathcal{L}_g \) as \( D(\hat{x}) \to 0 \), thus decreasing the performance of \( D \) for telling synthetic data samples apart from the real ones. The discriminator is trained as to minimize \( \mathcal{L}_d \) as \( D(x) \to 1 \) and \( D(\hat{x}) \to 0 \), thus increasing its capacity to distinguish samples from real and synthetic \[^2\]. Equation (1) details the loss functions for both \( D \) and \( \mathcal{G} \) given a mini-batch of size \( m \).

\[
\begin{align*}
\mathcal{L}_g &= -\frac{1}{m} \sum_{0 \leq i < m} \log(D(\hat{x}_i)) \\
\mathcal{L}_d &= \frac{1}{m} \sum_{0 \leq i < m} \log(D(x_i)) + \log(1 - D(\hat{x}_i))
\end{align*}
\]

(1)

The quantum analog for the generative adversarial networks (qGANs) is defined with different configurations concerning the problem being approached \[^2\]. With both generator and discriminator being either classical or quantum. Thus, by defining the hybrid quantum-classical model configuration for a qGAN, with a quantum generator and a classical discriminator, it is possible to load the intrinsic distribution of the data into a quantum state \[^2\].

In order to load the probability distribution of some dataset \( D \), it is necessary to optimize the parameters of the generator. During training, the generator quantum circuit is measured several times producing a set of frequencies corresponding to \( n \)-points of the discretized and truncated distribution, for a \( n \)-qubits quantum circuit. The discriminator then tries to distinguish between the synthetic frequencies sampled by...
the generator and the frequencies of real data. Unlike classical GANs, where the generators use stochastic input to generate new samples [7], the generator relies on quantum measurements for the stochastic sampling of synthetic data [8]. However, one can define an input state for the generator’s quantum circuit. For the experiments presented in this section, the input state of the generator was initialized in a uniform distribution.

A series of three experiments were performed to compare qGAN and BAA methods loading the log-normal distribution. Choosing the log-normal distribution combined with the generator’s input state initialized according to a uniform distribution ensures that the experiment will not suffer from the limitation described in Ref. [7] regarding the sign of the data components. The moments of the distributions varying the scale parameter are detailed in Table 1. The qGAN was trained over 1000 epochs using the ibmq_statevector_simulator backend. The dataset has 20000 uni-dimensional samples generated by Scipy [7] pseudorandom procedure. The generator’s circuit is a 7-qubit full entangled quantum circuit model with only one repetition of the alternating entanglement and rotation block. Such a model is quadratic on the number of qubits and requires 21 CNOTs for 7-qubits. The discriminator’s architecture is a simple multilayer perceptron, implemented with the Pytorch [9] library. The parameters of both models were initialized randomly. Before training, the data was truncated and discretized into a grid of $2^7$ evenly spaced points within the interval $[0, 20]$. Figure 1 details the frequencies generated by the BAA and the trained qGAN. For the sake of comparison, the BAA number of CNOTs was also kept to 21.

The numerical experiment shows that while qGAN can load the data distribution, the frequencies returned by the generator are not as approximate to the target discretized distribution frequencies as the ones produced using the BAA method in terms of state fidelity. The qGAN fidelity loss shown in figures 1d–1f - 0.144, 0.299, and 0.325 - is even larger than the geometric measure of each distribution - 0.052, 0.051, and 0.045 - i.e. the product states generated by BAA with zero CNOTs (Fig. 2) are closer to the original distributions.
### SUPPLEMENTARY Table 1: Approximate moments of the Log-normal distribution according to the parameter scale ∈ {0.5; 1; 2}

| scale | Mean   | Variance | Skewness | Kurtosis |
|-------|--------|----------|----------|----------|
| 0.5   | 0.8243 | 1.1676   | 6.1848   | 110.9363 |
| 1     | 1.6487 | 4.6707   | 6.1848   | 110.9363 |
| 2     | 3.2974 | 18.6830  | 6.1848   | 110.9363 |

SUPPLEMENTARY Fig. 2: BAA frequencies to produce the approximation by a product state.

than the states generated by qGAN with 21 CNOTs. Furthermore, we also attempted to apply the qGANs to other distributions – normal, Laplace, and semicircular (Fig. 3) – and find that those approximations are deviating substantially from the expectation even though we need to note at this point that each qGAN learning protocol takes about 4.5 hours on a GPU based qiskit aer noiseless simulation executed with Google’s Colaboratory. On top of this, we observe the known problem of barren plateaus for given initial conditions. In the future, as better quantum devices emerge, it will be clear whether noiseless classical simulations are at all a benchmark for the full hybrid-approach or if the quantum processing does speed up the overall performance.

SUPPLEMENTARY Fig. 3: Comparison between the qGAN frequencies and the exact frequencies for normal, Laplace and semicircular distributions.

In the example of log-normal distribution, the BAA performs significantly better than the qGAN method with the same number of entangling gates. It is interesting to note that the log-normal distribution has a good product state approximation already (Fig. 2). This good low-rank approximation may explain why the BAA performs better than the qGAN approach because the ansatz with 21 CNOT gates may fail structurally
to gain insight into the problem.

A possible future work is to use the BAA’s entanglement structure analysis and define the qGAN ansatz accordingly. When we know that a product state is a good approximation, we could remove or reduce the entangling gates from the ansatz. When the BAA indicates disentanglement of sub-spaces, we may use this information for the qGAN ansatz and try to reduce the total number of CNOT gates by combining the BAA and the qGAN method.

2 Experiments

2.1 Random State Approximation

Two sets of experiments were performed to evaluate the LRSP and BAA algorithms. The first set evaluates the required number of CNOTs to initialize a quantum state with the number of qubits $n$ from 11 to 14 and Schmidt measure $m = \lceil \log_2(\text{rank}) \rceil$ from 0 to $\lceil n/2 \rceil$. The quantum states are initialized randomly from a uniform distribution. After the Schmidt decomposition, only the first $2^m$ Schmidt coefficients are considered. Figure 4 shows the reduction in circuit depth and number of CNOTs with smaller values of $m$. Numerical results confirm that the number of CNOTs and the circuit depth are an exponential function concerning the Schmidt measure of entanglement $m$.

### Table 2: Mean Absolute Error between probabilities from measurements of an approximate 7-qubit state generated by the low-rank state preparation (LRSP) algorithm and the expected probabilities for the original (exact) state. Each result is an average of 10 runs with 8192 shots. The values in parentheses are the standard deviations. The variable $m=3$ indicates full rank, i.e., no approximation is employed.

| $m$ | CNOTs | depth | fidelity | ibmq_qasm_simulator | ibmq_casablanca | ibmq_jakarta | ibmq_perth |
|-----|--------|-------|----------|----------------------|-----------------|--------------|------------|
| 0   | 13     | 11    | 0.79869  | 0.003730 (0.000080)  | 0.004384 (0.000280) | 0.004394 (0.000186) | 0.004894 (0.000467) |
| 1   | 36     | 51    | 0.85460  | 0.003226 (0.000067)  | 0.004557 (0.000160) | 0.004848 (0.000286) | 0.004666 (0.000170) |
| 2   | 84     | 115   | 0.95016  | 0.002068 (0.000070)  | 0.004564 (0.000173) | 0.004523 (0.000192) | 0.004581 (0.000274) |
| 3   | 150    | 243   | 1.00000  | 0.000696 (0.000049)  | 0.006293 (0.001588) | 0.005272 (0.001090) | 0.005353 (0.000780) |

Supplementary Table 3: Mean Absolute Error between probabilities from measurements of an approximate 7-qubit state generated by the bounded approximation algorithm (BAA) and the expected probabilities for the original (exact) state. Each result is an average of 10 runs with 8192 shots. The values in parentheses are the standard deviations. At maximum fidelity loss $l = 0.26$, the state is completely disentangled (CNOTs=0 and depth=1).

| $l$  | CNOTs | depth | fidelity | ibmq_qasm_simulator | ibmq_casablanca | ibmq_jakarta | ibmq_perth |
|------|-------|-------|----------|----------------------|-----------------|--------------|------------|
| 0.00 | 150   | 243   | 1.00000  | 0.000006 (0.000064)  | 0.005556 (0.000164) | 0.005175 (0.000841) | 0.005487 (0.000848) |
| 0.12 | 54    | 57    | 0.88742  | 0.003046 (0.000073)  | 0.005051 (0.000309) | 0.004834 (0.000212) | 0.005297 (0.000347) |
| 0.18 | 31    | 53    | 0.82141  | 0.003527 (0.000101)  | 0.004653 (0.000166) | 0.004584 (0.000159) | 0.005165 (0.000507) |
| 0.19 | 30    | 53    | 0.81630  | 0.003369 (0.000045)  | 0.004630 (0.000132) | 0.004801 (0.000208) | 0.005172 (0.000538) |
| 0.22 | 10    | 11    | 0.78536  | 0.003748 (0.000069)  | 0.004532 (0.000174) | 0.004016 (0.000219) | 0.004805 (0.000333) |
| 0.23 | 5     | 9     | 0.77064  | 0.003822 (0.000065)  | 0.003976 (0.000084) | 0.003939 (0.000070) | 0.004292 (0.000290) |
| 0.24 | 4     | 9     | 0.76403  | 0.004060 (0.000074)  | 0.004054 (0.000106) | 0.004126 (0.000081) | 0.004313 (0.000174) |
| 0.25 | 1     | 3     | 0.75907  | 0.004043 (0.000090)  | 0.004214 (0.000105) | 0.004056 (0.000079) | 0.004005 (0.000068) |
| 0.26 | 0     | 1     | 0.74502  | 0.004168 (0.000059)  | 0.004369 (0.000086) | 0.004223 (0.000055) | 0.004280 (0.000085) |

In the second set of experiments, the Mean Absolute Error (MAE) between the squared amplitudes of a random state and measurements of the state generated with noisy quantum devices were evaluated. For each configuration, a 7-qubits random state was initialized ten times. Table 2 shows the average MAE with $m = \{0, 1, 2, 3\}$ and Table 3 shows the average MAE with $l = \{0.0, 0.01, \ldots, 0.3\}$ (Table 3 shows only the
SUPPLEMENTARY Fig. 4: (a)-(d) Circuit depth and number of CNOTs to load a random $2^n$-dimensional complex vector into a quantum computer by adjusting parameter $m$. The decreasing number of CNOTs and lower depths is the consequence of exchanging computational complexity for an approximated quantum state. It is achieved by reducing the entanglement between the qubits. (e)-(h) Fidelity between the original and the approximate state vector.

values of $l$ where there is a change). For noisy devices, the smallest MAE occurs when the entanglement is smaller than that of the original state. It indicates that, in noisy devices, it is better to use approximate states with smaller entanglement because the loss of fidelity introduced by the approximation is smaller than the loss caused by the noise of more complex circuits. The nonparametric Wilcoxon paired signed-rank statistical test \cite{?} with $\alpha = 0.05$ was employed for each device to check if there is a significant difference between the smallest MAE - obtained with the approximate state - and the MAE achieved with the original state. It was determined that such values are statistically distinct.

Figure 5 shows experimental results with the initialization of a 7-qubit state on two noisy devices. Again, quantum approximate state preparation has advantages. The approximation using BAA achieves the lower MAE at $l = 0.22$ for ibmq_casablanca and $l = 0.23$ for ibmq_jakarta. A probable reason for this difference is that ibmq_casablanca has a quantum volume of 32 versus 16 of ibmq_jakarta. Across the interval highlighted by the red area, both lines - actual device and ideal simulation - converge, which indicates that the approximate state becomes better handled by the devices as the parameter $l$ value increases. This observation leads to the possibility of using BAA to characterize quantum devices regarding their capability to harness quantum entanglement.
SUPPLEMENTARY Fig. 5: BAA experimental results with a 7-qubit complex state on two IBM quantum devices. The orange and blue lines show the results from the ibmq_qasm_simulator and the actual device, respectively. The red area indicates the range in which the device is unable to handle entanglement completely.
2.2 Loading images

SUPPLEMENTARY Fig. 6: (a) Approximations of the original image produced by the BAA using the greedy strategy. (b) Reconstruction of the images made from measurements using an ideal quantum simulator. (c) Reconstruction of the images made from measurements using a noisy quantum simulator.

To evaluate the feasibility of encoding a real-world image on current quantum devices, we experimented using a black and white 8bpp image with a resolution of 128x128 pixels (original image from Ref. [?]). Such an image has 16384 pixels that can be encoded in the amplitudes of a 14-qubit quantum state. Figure 6a shows the input vectors approximated by the BAA using the greedy strategy. The number above each image (which represents the vectors) is the maximum fidelity loss $l_{\text{max}}$ used in the approximation. The images in Figure 6b were constructed from measurements using the ibmq_qasm_simulator ideal simulator. A total of 20 runs with 8192 shots were employed for each input vector to achieve this result.

Finally, in Figure 6c, we have a similar experiment as in the previous figure but using the noise model and coupling map of the ibmq.mumbai device. It can be seen that the noise eliminated the ability to recover the images through the measurements. The exception is the image with approximation $l_{\text{max}} = 0.1$.

Given this configuration, the encoding circuit has only 8 CNOTs (Table 4), but such a reduction makes the image recognizable with only but the most significant features. In some applications, i.e. machine learning tasks, this may indeed be sufficient but it is clear that the poor performance displayed here will limit the applications.

| $l_{\text{max}}$ | 0.0 | 0.01 | 0.02 | 0.03 | 0.04 | 0.05 | 0.06 | 0.07 | 0.08 | 0.09 | 0.1 |
|------------------|-----|------|------|------|------|------|------|------|------|------|-----|
| CNOTs            | 18175 | 11292 | 4473 | 2767 | 2767 | 1087 | 658  | 251  | 148  | 54   | 8   |

SUPPLEMENTARY Table 4: The number of CNOTs to initialize quantum states representing different approximations of the zebra image using the BAA state preparation.

The experiment shows that current devices are only capable of handling simple images, such as the dataset of the VQC experiment. The BAA can adjust the images according to the capability of the device used. Our confidence is that future NISQ devices will soon become capable of real-world imaging applications.

3 The Open Source Quantum Computing Library qclib

The Low Rank State Preparation (LRSP) and the Bounded Approximation Error Algorithm (BAA) are both implemented from a publicly available open source repository at github.com. The library is called qclib [?]
SUPPLEMENTARY Fig. 7: Column-by-column isometry decomposition. Quantum circuit for the decomposition of the $2^m$ columns from the isometry $V$. Operator $G_k$ is a modification of Mottõnen's algorithm ($G_0$ is exactly Mottõnen’s state preparation). Operator $D$ clears the global phases.

and encompasses most code used in this paper, but also has many other state preparation and isometry routines that are cited in this work independently available. Also, the VQC machine learning application is available from this library. It is based on qiskit [?] and thus widely applicable to many simulators and quantum devices.

**SUPPLEMENTARY Algorithm 1:** Construct an operator $G_k$ that perform the isometry column $V_k$ decomposition.

**input:** An isometry $V$, number of qubits $n$, and isometry column index $k$.

**output:** Operator $G_k$.

1. Create a quantum circuit with $n$ qubits to construct the operator $G_k$.
2. Set $i = 0$ and $k_{bin}$ as a binary representation of $k$.
3. Set $target = n - i - 1$, $control = \{0, \ldots, target - 1\}$, and $ancilla = \{target + 1, \ldots, n - 1\}$.
4. If $k_{bin}\{i\} = 0$ and $\text{int}(k_{bin}\{0, \ldots, i - 1\}) \neq 0$ continue, otherwise goto Step 7.
5. Perform a multicontrolled gate $MCG$ with unitary $U$, controlled by $control + ancilla$ and acting on $target$.
6. Update isometry $V = MCG \times V$.
7. Perform a uniformly controlled gate $UCG$ with unitaries $U_l$, controlled by $control$ and acting on $target$.
8. Update isometry $V = UCG \times V$.
9. If $i < n - 1$, set $i = i + 1$ and return to Step 3, otherwise output the operator $G_k$.

In addition to the methods described in the main paper (LRSP and BAA), the implementation of efficient algorithms for the construction of unitaries [?] and isometries [?] is of fundamental importance to achieving the final result of this paper. The use of isometries is what unlocked the reduction of entanglement in Plesch’s state preparation [?], replacing unitaries and allowing the construction of low-rank approximation circuits.

This work uses the column-by-column decomposition method to construct the isometries [?]. Each column $k$ of the isometry $V$ is decomposed into a $G_k$ operator (Fig. 7 and Alg. 1). Such a decomposition is a variation of the amplitude encoding by Mottõnen et al. [?]. The variation (lines 4–6) aims to make the decomposition of each new column not affect the previous ones.

Figure 7 and Algorithm 1 summarize the method. The unitaries of rows 5 and 7 are constructed according to Lemma 2 and equations (A7a), (A7b), and (A10) of Ref. [?]. It should be noted that the columns are prepared up to a diagonal gate, which means that a global phase correction is required. This is done by operator $D$ of Fig. 7.

Knowing that upon completion of Algorithm 1 the square matrix $V_{2^m \times 2^m}$ ($2^m$ is the number of columns of the isometry $V$) must equal to the identity, any phase present on the diagonal of the matrix indicates a global phase that needs to be corrected using operator $D$. Therefore, it is constructed as a diagonal gate where the phases are those present in $V_{2^m \times 2^m}$.
4 Low-rank state preparation

4.1 Complexity

When acting on $k$ qubits, a quantum state preparation requires $2^k - k - 1$ CNOTs [7], a unitary operator $\frac{23}{48}2^{2k} - \frac{3}{2}2^k + \frac{1}{3}$ [?] and an isometry $2^{m+k} - \frac{1}{24}2^k + O(k^2)2^m$ [7]. Let $\mathcal{H}_A$ be a subsystem with $n_A$ qubits ($1 \leq n_A \leq n/2$ for $n$ even and $1 \leq n_A \leq (n-1)/2$ for $n$ odd) and $n_B = n - n_A$ the subsystem $\mathcal{H}_B$ number of qubits ($n_B$ is greater than or equal to $n_A$). Considering the complete low-rank state preparation (LRSP) circuit and summing up contributions from all four phases, the overall number of CNOT gates is represented by

$$m < n_A:$$

$$2^m - m - 1 + m + 2^{m+n_A} - \frac{1}{24}2^{n_A} + 2^{m+n_B} - \frac{1}{24}2^{n_B}$$

(2)

$m = n_A$ and $n_A < n_B$:

$$2^{n_A} - n_A - 1 + n_A + \frac{23}{48}2^{n_A} - \frac{3}{2}2^{n_A} + \frac{4}{3} + 2^n - \frac{1}{24}2^{n_B}$$

(3)

$m = n_A$ and $n_A = n_B$:

$$2^{n_A} - n_A - 1 + n_A + 2\left(\frac{23}{48}2^n - \frac{3}{2}2^{n_A} + \frac{4}{3}\right)$$

(4)

The phases brackets indicate the contribution from each phase of the LRSP procedure to the number of CNOTs. Phase 1 is a state preparation, phase 2 a sequence of CNOT gates, phases 3 and 4 are isometry or unitary decompositions. The numbers are bounded by the results presented in Table 5.

| condition | CNOT count |
|-----------|------------|
| $0 \leq m < n_A$ | $O(2^{m+n_B})$ |
| $m = n_A$ | $n_A < n_B$ | $O(2^n)$ |
| $n_A = n_B$ | $O(2^n)$ |

SUPPLEMENTARY Table 5: CNOT gates to leading order required by LRSP.

The above results depend on the choice of unitary and isometry decompositions. The methods used are Quantum Shannon Decomposition (QSD) for the unitaries [?] and Column–by–Column Decomposition (CCD) for the isometries [7]. It is possible to reduce the number of CNOTs of the isometries to $\frac{23}{48}2^{2k}$ when $m = k - 1$ using the Cosine-Sine Decomposition (CSD) [?]. Table 5 shows that the number of CNOTs reduces as $m$ decreases.

4.2 Algorithm

To achieve low-rank approximation, the corresponding input parameter $r$ is introduced to Algorithm 2 and can assume values between 1 and $2^{n_A}$. The following conditional statement is introduced between lines 2 and 3: If $r < \text{rank}$ then $\text{rank} = r$. Setting $r = 1$ completely disentangles subsystems $\mathcal{H}_A$ and $\mathcal{H}_B$, which implies a quadratic reduction in the number of CNOT gates in the circuit to prepare a state when $n_A = \lfloor n/2 \rfloor$. Additionally, the circuit depth is also reduced.
SUPPLEMENTARY Algorithm 2: Low-rank state preparation algorithm

input: A state vector of $2^n$ amplitudes.
input: A partition of $n_A$ qubits ($1 \leq n_A \leq \lfloor n/2 \rfloor$).
output: Encoding quantum circuit.

1. Decompose the state vector using SVD and partition (store unitaries $u$ and $vh$, and vector $s$).
2. Set rank equal to the number of non-zero elements of $s$ \( (\text{rank} = \text{count}(s \neq 0)) \)
3. Set Schmidt measure $m = \lceil \log_2(\text{rank}) \rceil$
4. If $m < n_A$, set the number of columns of $u$, $vh^T$, and $s$ to $2^m$, otherwise continue.
5. Create a quantum circuit with $n$ qubits \( (n = \log_2(\text{length(state vector)})) \)
6. If $m > 0$, encode the normalized vector $s$ on the first $m$ qubits $q \in$ partition using an amplitude encoding state preparation algorithm. This step is named Phase 1.
7. Perform $m$ CNOT gates between control qubit $q_c \in$ partition and target $q_t \notin$ partition. This step is named Phase 2.
8. Encode $u$ on qubits $q \in$ partition using an algorithm according to its dimension (vector, isometry, and unitary). This step is named Phase 3.
9. Encode $vh^T$ on qubits $q \notin$ partition using an algorithm according to its dimension (vector, isometry or unitary). This step is named Phase 4.
10. Output the encoding quantum circuit.

5 Bounded Approximation Algorithm

5.1 Pseudocode

Pseudocode 1 constructs the approximation search tree as described in the main document. Pseudocode 2 employs Pseudocode 1 and expresses the bounded approximation algorithm (BAA). Pseudocode 3 builds quantum circuits using the BAA approach for encoding a complex input vector into the amplitudes of a quantum state.

Lines 3 of Pseudocode 1 generates all combinations of state partitions from size 1 to $\lfloor n/2 \rfloor$. Line 5 separates the indicated partition and creates two new states with a fidelity loss relative to the original state. Line 7 configures the recurrence so that, if the partition’s total fidelity loss is less or equal than the maximal acceptable fidelity loss, a new child node is created. Compared to the previous node, the new one has the original state replaced by the two separated states.

Line 2 of Pseudocode 2 employs the geometric measure to estimate the maximum possible fidelity loss. If it is less than the maximal acceptable fidelity loss, the search routine is avoided and the product state used.

Line 3 of Pseudocode 3 initializes each of the state vectors associated to the selected node. Any initialization algorithm can be used, including the LRSP described in Section 4. The qubits used in each initialization must follow the previous BAA partitioning. Line 9 of Pseudocode 2 CNOT count must take into account the algorithm choice.
SUPPLEMENTARY Pseudocode 1: Generate a search tree.

```plaintext
build_search_tree (node, max_fidelity_loss):
    input : Tree node (node) to apply the next search step.
    input : The maximal acceptable fidelity loss (max_fidelity_loss) for the approximation.
    // Ignore single-qubit states.
    for s ← 0 to length(node.states) − 1 do
        // Combinations taken \( k \in [1, \lfloor n/2 \rfloor] \) at a time without repetition.
        // Combinations equal to the complement of previous ones can be omit, as they are redundant.
        partitions = combinations(node.states[s], k ← 1 to \( \lfloor n/2 \rfloor \))
        for p ← 0 to length(partitions) − 1 do
            subsystem1, subsystem2, fidelity_loss = separate_partition(node.states[s], partitions[p])
            total_fidelity_loss = 1 - (1 - fidelity_loss) × (node.total_fidelity_loss)
            if total_fidelity_loss ≤ max_fidelity_loss then
                new_node.total_fidelity_loss = total_fidelity_loss
                new_node.states = node.states
                new_node.states.remove(s)
                new_node.states.add(subsystem1)
                new_node.states.add(subsystem2)
                node.children.add(new_node)
            end if
        end for
    end for
end
```

SUPPLEMENTARY Pseudocode 2: Find the node that will be used to perform the state preparation.

```plaintext
baa (state, max_fidelity_loss):
    input : A state vector (state) to be approximate.
    input : The maximal acceptable fidelity loss (max_fidelity_loss) for the approximation.
    output: Node with the best approximation.
    // Estimate the maximal fidelity loss.
    entanglement = geometric_measure(state)
    if entanglement ≤ max_fidelity_loss then
        node.vectors = product_state(state)
        return node
    end if
    // Start with the complete state.
    root_node.vectors = state
    build_search_tree (root_node, max_fidelity_loss)
    // Leaf nodes are the ones closest to max_fidelity_loss for each branch.
    leaves = list_leaves(root_node)
    // Search the node with the greatest reduction in the number of CNOTs.
    // The CNOT count depends on the algorithm chosen to encode the states.
    best_node = search_best(leaves)
    return best_node
```

11
SUPPLEMENTARY Pseudocode 3: Construct a circuit that performs an approximate state preparation for the input vector state.

**input** : A state vector (state) to be encoded.
**input** : The maximal acceptable fidelity loss (max_fidelity_loss) for the baa approximation.
**output**: Circuit with the encoded quantum state.

```plaintext
1 node = baa (state, max_fidelity_loss)
2 for s ← 0 to length(node.states) − 1 do
    // Use any amplitude encoding algorithm.
    3 circuit.initialize(node.states[s])
4 return circuit
```