PANE

PANE Journal of Physics Education P. J. Phys. Edu. Vol 1(2), October 2026

FROM SINGULAR VALUES TO QUBITS: A PEDAGOGICAL INTRODUCTION TO VARIATIONAL QUANTUM SVD AND TUCKER DECOMPOSITION, WITH A CLASSICAL SIMULATION STUDY

Upashana Sharma and Rupak Mukherjee*

Department of Physics, Sikkim University, Gangtok, India

*Corresponding Author: rmukherjee@cus.ac.in

Abstract: The singular value decomposition (SVD) and its multi-way generalisation, the Tucker decomposition, compress data and reveal which directions carry information. For quantum states, however, the matrix dimension doubles with every added qubit, so classical SVD routines become exponentially expensive in the number of qubits. We give a pedagogical account of two hybrid quantum-classical algorithms designed to avoid this wall: the Variational Quantum Singular Value Decomposer (QSVD) and its tensor extension, Quantum Tucker Decomposition (QTucker). We build the necessary background from first principles and present the variational circuit and cost function. In noiseless classical simulations on bipartite states of up to ten qubits and on a small third-order tensor, the post-processed spectra and entanglement entropies agree with the classical benchmark to numerical precision. This agreement is guaranteed by the invariance of singular values under local unitaries, so it is the training cost, not the spectrum, that measures the accuracy of the circuit. All results come from classical simulation; a computational advantage on quantum hardware is an asymptotic projection that is not demonstrated here.

Keywords: quantum computing; variational quantum circuits; singular value decomposition; Tucker decomposition; Schmidt decomposition; entanglement entropy; NISQ algorithms

(Received: 16 September 2026; accepted: 26 October 2026; published: 30 October 2026)

1 Introduction

1.1 Everyday Matrices, Hidden Hats

If you have ever taken a course in linear algebra or data analysis, you have already met the Singular Value Decomposition (SVD), even if the name did not come up right away. Principal Component Analysis (PCA), image compression, and the idea of finding the "best rank-k approximation" of a matrix are all the SVD wearing different hats.

Here is the picture. A matrix C can be factorised as

$$C = U\Sigma V^{\dagger},$$

where U and V are unitary matrices and Σ is a diagonal matrix whose non-negative entries σ_i are ranked from largest to smallest. These σ_i are the *singular values*. If we keep only the largest few and throw away the rest, we discard the least important directions in the data while keeping almost everything that matters [18]. That is why a JPEG image can look almost identical to the original even though most of its singular values have been dropped.

The same idea generalises to multi-way arrays—called *tensors*—through the Tucker decomposition [5], which in practice is computed via the Higher-Order SVD (HOSVD) [6]. A three-dimensional dataset, such as a hyperspectral image with two spatial indices and one wavelength index, can be compressed the same way an ordinary matrix is: by "unfolding" the tensor along each of its three directions (its *modes*), running an SVD on each resulting

matrix, and keeping only the largest singular values. Both SVD and HOSVD have been workhorses of numerical linear algebra for decades.

There is one catch: both procedures assume the object being decomposed lives on a hard disk, not inside a quantum register.

1.2 Where the Classical Machinery Breaks Down

That assumption fails the moment the "matrix" in question is the amplitude table of a quantum state.

Consider two subsystems A and B, each built from $n/2$ qubits, prepared in an entangled pure state $|\psi\rangle_{AB}$. Writing this state in a product basis gives coefficients c_{ij} that can be arranged into a $2^{n/2} \times 2^{n/2}$ matrix. For $n = 40$, that matrix already has over a million rows. Classical SVD via eigendecomposition of the Gram matrix $C^\dagger C$ costs $\mathcal{O}(mn^2)$ for an $m \times n$ matrix [18], and the analogous HOSVD on an order-3 tensor with mode dimension N costs $\mathcal{O}(N^3)$ [6, 19]. Since the natural mode dimension for an n-qubit register is $N = 2^n$, both costs are, in effect, exponential functions of qubit count.

This is the same exponential wall that makes classical simulation of quantum systems hard in general [8]. It means that large-scale tensor-network contraction and multi-qubit state characterisation cannot simply borrow classical numerical linear algebra unmodified.

1.3 Escaping the Wall

Why does this exponential growth matter so much? A tensor of size $I \times I \times \cdots \times I$ in N dimensions contains I^N entries. For even modest N, storing or analysing every entry becomes impractical. Fortunately, many real-world tensors exhibit *low-rank structure*: their information content can be captured by relatively few parameters [19, 15]. Tensor decomposition exploits this by factorising high-dimensional tensors into combinations of lower-dimensional factors. It enables dimensionality reduction, separates signal from noise by discarding small singular values [6], and reveals latent patterns not apparent in the raw data [5].

The challenge is finding these decompositions without paying the exponential price of classical linear algebra.

The two most obvious quantum remedies bring their own difficulties. Full quantum state tomography reconstructs the density matrix by measuring in every combination of local bases, which requires 3^n distinct measurement settings for an n-qubit state [10]—about three and a half billion for a modest twenty qubits. Algorithms based on quantum phase estimation, including the Harrow–Hassidim–Lloyd approach to linear systems [1] and phase-estimation-based SVD routines [2], extract spectral information with far fewer measurements, but they need long, coherent circuits and error-corrected qubits that are not yet available on near-term devices [8, 28].

1.4 What we Set Out to Do

This is the gap that variational quantum algorithms are designed to fill: shallow, parametrised circuits trained on a classical computer to approximate the answer a much deeper circuit would give exactly [7]. Here we work through one such algorithm in detail—the Variational Quantum Singular Value Decomposer, or QSVD, introduced by Bravo-Prieto, García-Martín, and Latorre [4]—and its natural extension to third-order tensors, which we call Quantum Tucker Decomposition (QTucker).

2 Background

2.1 From Classical SVD to Quantum Linear Algebra

Section 1 reviewed the classical SVD and its tensor generalisation, the Higher-Order SVD (HOSVD) [6]; here we recall only what is needed to connect them to quantum states.

The trouble begins when the "data" are the amplitudes of a quantum state. As discussed in Sec. 1, the coefficient matrix of an n-qubit bipartite state has size $2^{n/2} \times 2^{n/2}$, and the classical cost grows exponentially with qubit count [8]. The classical toolbox was built for data that fit on a disk, not for states that live in Hilbert space.

Quantum algorithms do offer an escape route—phase estimation underlies both HHL [1] and the quantum SVD of Rebentrost *et al.* [2], as described in Sec. 1—but it requires long, coherent circuits and error-corrected qubits.

The Solovay–Kitaev theorem tells us how many elementary gates are needed to approximate an arbitrary unitary to a given precision [9], and the implied gate counts place these algorithms well outside the reach of current noisy intermediate-scale quantum (NISQ) hardware [8, 28]. Full quantum state tomography avoids the need for coherent circuits, but its 3^n measurement burden (Sec. 1) is no more practical.

2.2 Variational Approaches and QSVD

The response to this impasse has been the variational, or hybrid quantum-classical, paradigm: a shallow, parametrised quantum circuit evaluates a cost function, and a classical optimiser updates the parameters [7]. This architecture underpins the variational quantum eigensolver and a growing family of NISQ-era algorithms [27, 7]. The approach is not without pitfalls. McClean *et al.* showed that overly expressive circuits can suffer from barren plateaus—exponentially vanishing gradients—while Wang *et al.* demonstrated that hardware noise can induce similar flattening [12, 26]. Mitigation strategies include building cost functions from local rather than global observables and using structured parameter initialisations [20], both of which figure directly in the algorithm discussed here.

The closest precursor to the present work is the variational diagonalisation of quantum states. LaRose *et al.* proposed training a single unitary to maximise subsystem purity using two copies of the state and a SWAP test [11]. Bravo-Prieto, García-Martín, and Latorre took a different route: they trained two local unitaries against a cost function estimable from a single copy measured in the computational basis, giving the Variational Quantum Singular Value Decomposer (QSVD) [4]. Because the Schmidt coefficients of a bipartite pure state are precisely the singular values of its coefficient matrix [3], diagonalising entanglement *is* performing an SVD—a connection we develop in detail further below.

A related but distinct direction uses tensor networks—matrix product states and higher-dimensional analogues—to avoid ever forming the full state vector by exploiting area-law entanglement [14, 21, 22]. QSVD and the extension we propose address a complementary question: given that a state is already prepared on quantum hardware, how does one extract its spectrum without an exponential classical post-processing step?

2.3 What Remains to be Done

Two gaps stand out in the landscape above. First, while QSVD has been demonstrated for bipartite states, no variational analogue of the full Tucker/HOSVD pipeline—factor matrices *and* core tensor for a genuine higher-order tensor—has been worked through with explicit truncation-error bounds and a resource comparison against classical HOSVD. Second, numerical validation of QSVD often blurs the line between the *asymptotic* resource advantage of the quantum algorithm (gate and measurement counts scaling polynomially in qubit number) and the *wall-clock* cost of classically *simulating* that same circuit (which is exponential, because it involves dense matrices of size 2^n). This article addresses both points: it develops Quantum Tucker Decomposition (QTucker) as a mode-wise application of QSVD with a proved error bound, and it is careful throughout to separate asymptotic projections from measured simulation times.

3 Methodology

3.1 Classical SVD via the Gram Matrix

We begin with the classical procedure, both because it is the validation oracle used throughout this work and because the variational quantum cost function turns out to be a disguised version of the same idea. For $C \in \mathbb{C}^{m\times n}$ with $m \leq n$, the SVD writes $C = U\Sigma V^\dagger$, with $U \in \mathbb{C}^{m\times m}$ unitary, $V \in \mathbb{C}^{n\times m}$ having orthonormal columns, and Σ diagonal with non-negative entries $\sigma_i = \Sigma_{ii}$. This route intuitively proceeds through the Gram matrix $G = C^\dagger C$ [18]:

1. Form $G = C^\dagger C$, which is Hermitian and positive semi-definite.
2. Solve the eigenvalue problem $Gv_i = \lambda_i v_i$ for the right singular vectors v_i.
3. Sort the λ_i in descending order and reorder the v_i to match.
4. Set $\sigma_i = \sqrt{\lambda_i}$, giving $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \dots)$.
5. Recover the left singular vectors from $u_i = Cv_i/\sigma_i$, so that $U = CV\Sigma^{-1}$.

This procedure costs $\mathcal{O}(mn^2)$ for $m \leq n$ [18] and underlies every mode-wise unfolding used in the classical HOSVD below.

The variational cost function discussed later is trying to reproduce, in miniature, the structure of this procedure; the difference is that the quantum algorithm never explicitly forms $C^\dagger C$, and instead lets that structure sit inside the entanglement of a physical state.



Figure 1: Schematic of the classical singular value decomposition $C = U\Sigma V^\dagger$: the coefficient matrix is factorised into left and right singular-vector bases and a diagonal singular-value matrix.

3.2 Tensors, Unfolding, and the Tucker Decomposition

Many datasets carry more than two indices. A third-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is simply a box of numbers t_{ijk} [19]. Matrix methods do not apply directly to a box, so the standard trick is to *unfold* (or matricise) the tensor: pick one mode to become the rows, and merge the remaining two indices into the columns. This gives the mode-n unfolding $\mathcal{T}_{(n)} \in \mathbb{R}^{I_n \times \prod_{k \neq n} I_k}$ [19]; for a third-order tensor there are three such unfoldings, $\mathcal{T}_{(1)}, \mathcal{T}_{(2)}, \mathcal{T}_{(3)}$.

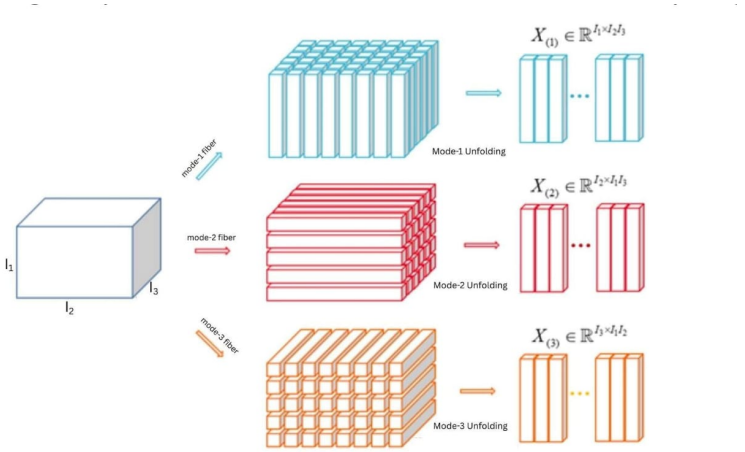


Figure 2: Mode-n unfolding of a third-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$. Each unfolding rearranges tensor fibres along a selected mode into matrix form, enabling the application of classical matrix decomposition techniques while preserving mode-specific structural information [19].

The Tucker decomposition writes the tensor as a small core tensor acted on by a factor matrix along each mode [5, 19]:

$$\mathcal{T} \approx \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}, \tag{1}$$

where $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is the core tensor, $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ are orthonormal factor matrices, (R_1, R_2, R_3) are the multilinear ranks, and $\times_n$ denotes the mode-n product: $(\mathcal{T} \times_n \mathbf{M})_{(n)} = \mathbf{M}\,\mathcal{T}_{(n)}$ [19].

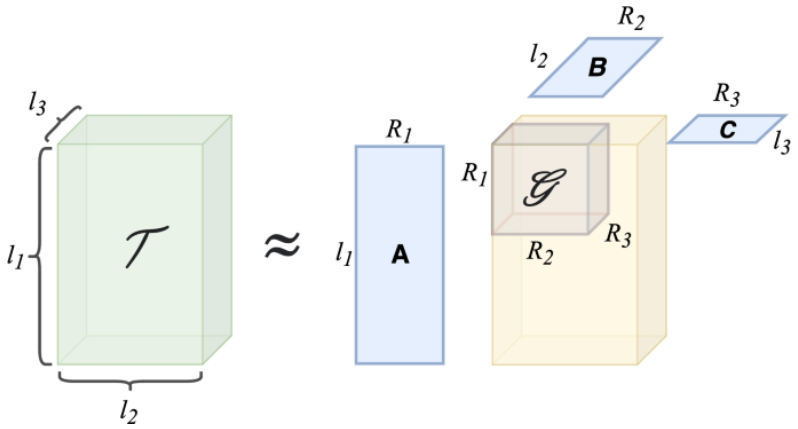


Figure 3: Tucker decomposition: the original tensor $\mathcal{T}$ is expressed as a multilinear product of a compact core tensor $\mathcal{G}$ and orthogonal factor matrices along each mode. Multilinear ranks (R_1, R_2, R_3) control the compression level [19, 6].

The Higher-Order SVD [6] computes this by running an ordinary SVD on each unfolding $\mathcal{T}_{(n)}$, keeping the leading R_n left singular vectors as $\mathbf{U}^{(n)}$, and then projecting:

$$\mathcal{G} = \mathcal{T} \times_1 (\mathbf{U}^{(1)})^\top \times_2 (\mathbf{U}^{(2)})^\top \times_3 (\mathbf{U}^{(3)})^\top. \tag{2}$$

Truncated reconstruction quality is measured by the Frobenius-norm error $\|\mathcal{T} - \widehat{\mathcal{T}}\|_F$ [19, 6], which is the tensor analogue of the Eckart–Young–Mirsky theorem for matrices. Columns of $\mathbf{U}^{(n)}$ play a role similar to principal components along mode n, and $\mathcal{G}_{r_1 r_2 r_3}$ records how those components interact across modes.

For a tensor $\mathcal{T} \in \mathbb{R}^{N\times N\times N}$ with $N = 2^n$, forming each mode-wise Gram matrix costs $\mathcal{O}(N^3)$ and its eigendecomposition costs $\mathcal{O}(N^3)$ as well [18]; summing over three modes and adding the core projection gives an overall cost of $\mathcal{O}(N^3) = \mathcal{O}(2^{3n})$ [19], exponential in qubit count if the mode dimension happens to be 2^n. The quantum formulation addresses this exponential dependence by replacing the full tensor computations with operations whose cost scales polynomially with the number of qubits.

3.3 The Schmidt Decomposition

A fundamental connection between linear algebra and bipartite quantum mechanics underlies the approach developed in this work. Take a bipartite pure state

$$|\psi\rangle_{AB} \in \mathcal{H}_A \otimes \mathcal{H}_B, \qquad |\psi\rangle_{AB} = \sum_{i,j} c_{ij} |e_i\rangle_A |e_j\rangle_B,$$

with $\dim \mathcal{H}_A = d_A$, $\dim \mathcal{H}_B = d_B$. The coefficients c_{ij} form the matrix $C \in \mathbb{C}^{d_A \times d_B}$ with entries $C_{ij} = c_{ij}$. The Schmidt decomposition theorem guarantees that this state can always be rewritten as [3]

$$|\psi\rangle_{AB} = \sum_{i=1}^{\chi} \lambda_i |u_i\rangle_A |v_i\rangle_B, \tag{3}$$

with $\chi \leq \min(d_A, d_B)$ the Schmidt rank, $\lambda_i \geq 0$ (strictly positive for $i \leq \chi$) satisfying $\sum_i \lambda_i^2 = 1$, and $\{|u_i\rangle_A\}$, $\{|v_i\rangle_B\}$ orthonormal local bases. The Schmidt coefficients $\{\lambda_i\}$ are *exactly* the singular values of the coefficient matrix C, and the reduced density matrix $\rho_A = \mathrm{Tr}_B(|\psi\rangle_{AB}\langle\psi|)$ has eigenvalues $\{\lambda_i^2\}$ [3]. In other words: finding the SVD of a bipartite quantum state's coefficient matrix is not an abstract numerical exercise laid on top of physics – it *is* the physical problem of diagonalising the entanglement between A and B. This equivalence is what allows a quantum circuit to perform an SVD at all: instead of forming $C^\dagger C$ on a classical computer, we let the physical entanglement of the state itself encode the same information, and design a circuit that reads it out.

3.4 Exact Output Coincidence: the Central Idea of QSVD

Bravo-Prieto, García-Martín, and Latorre showed that neither full tomography nor phase estimation is necessary to extract the Schmidt spectrum [4]. It suffices to variationally find two *local* unitaries, U_A acting only on subsystem A and V_B acting only on B, such that

$$(U_A \otimes V_B)\,|\psi\rangle_{AB} = \sum_{i=1}^{\chi} \lambda_i\, e^{i\alpha_i} |e_i\rangle_A |e_i\rangle_B. \tag{4}$$

When Eq. (4) holds, measuring A and B in the computational basis *always* returns identical bitstrings on both sides – a property we will call Exact Output Coincidence (EOC), following [4]. Under EOC, the probability of observing outcome i on both registers is $p_i = \lambda_i^2$, so the Schmidt coefficients are read off directly as $\lambda_i = \sqrt{p_i}$ from measurements in a single basis, and the local eigenvectors follow from $U_A^\dagger|e_i\rangle_A$, $V_B^\dagger|e_i\rangle_B$ up to an unobservable phase.

The table below puts this in context against the two alternatives discussed earlier: the measurement overhead drops from exponential (tomography) or circuit-depth-dominated (phase estimation) to a single measurement setting.

Table 1: Measurement settings required to extract a Schmidt (equivalently, singular-value) spectrum on n qubits.

Method	Measurement settings
Full state tomography [10]	3^n
QPE-based SVD [2]	$\mathcal{O}(1/\varepsilon^2)$, deep circuits
QSVD [4]	1

3.5 The Brick-wall Ansatz

Both $U_A(\boldsymbol{\Theta})$ and $V_B(\boldsymbol{\Omega})$ are built from L identical layers [4]. Each layer consists of two blocks, each a rotation sublayer – an Euler gate $R(\alpha,\beta,\gamma) = R_z(\alpha)R_x(\beta)R_z(\gamma)$ applied to every qubit – followed by an entangling sublayer of CZ gates arranged in a brick-wall pattern: qubit pairs $(0,1),(2,3),\ldots$ are entangled in the first block, and $(1,2),(3,4),\ldots$ in the second. A final rotation sublayer follows the last entangling sublayer. Counting parameters, each rotation sublayer contributes three real parameters per qubit, and there are $2L+1$ rotation sublayers in total (see the Appendix for the short derivation), giving

$$p(n,L) = 3n(2L+1) \tag{5}$$

parameters per subsystem, linear in both the qubit count n and the depth L. It is worth pausing on how striking this is: a five-qubit unitary lives in a 32×32-dimensional space, yet at $L=2$ we describe it with only 75 real numbers (150 for the pair U_A, V_B). That compression, from an exponentially large unitary group down to a handful of angles, is the entire resource argument behind the variational approach.

3.6 The Cost Function

We still need a number the classical optimiser can chase downhill. Let $|\psi'\rangle = (U_A\otimes V_B)|\psi\rangle_{AB}$, and reshape its amplitudes into a matrix $C_{ij} = \langle e_i| \otimes \langle e_j|\psi'\rangle\rangle$. Writing $\mathcal{P}_{\text{diag}} = \sum_i |C_{ii}|^2$ for the diagonal probability mass, the cost is defined as [4]

$$\mathcal{C}(\boldsymbol{\Theta},\boldsymbol{\Omega}) = 1 - \frac{\mathcal{P}_{\text{diag}}}{\||\psi'\rangle\|^2}, \tag{6}$$

which vanishes exactly when EOC (Eq. (4)) holds. A Hamming-distance-type cost built from two-qubit correlators [4] vanishes under the same condition, but it is not numerically equal to Eq. (6): it is the expected Hamming distance between the A and B outcomes, and hence upper-bounds Eq. (6):

$$\mathcal{C}_{\text{H}} \equiv \sum_{q=1}^{n_{\min}} \frac{1-\langle \sigma_z^{q,A}\sigma_z^{q,B}\rangle}{2}, \tag{7}$$

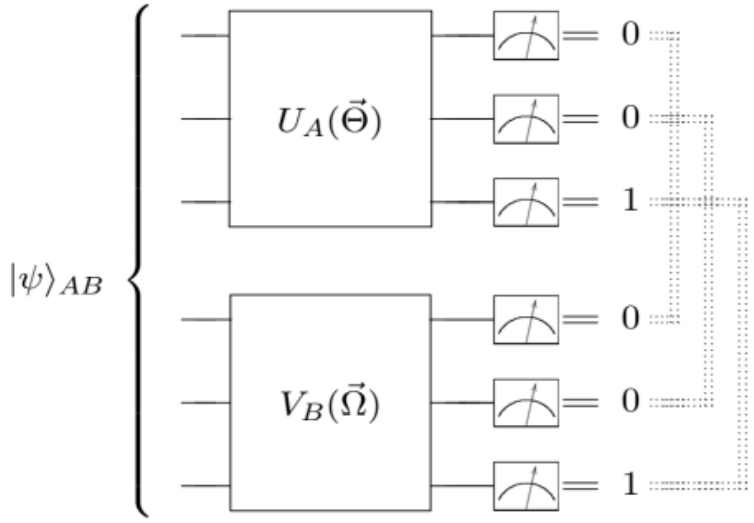


Figure 4: Brick-wall variational circuit used for U_A and V_B with $L = 2$ layers. Each layer has two blocks of single-qubit Euler rotations followed by a layer of nearest-neighbour CZ gates (pairs $(0,1),(2,3),\ldots$ in the first block, $(1,2),(3,4),\ldots$ in the second); a final rotation layer follows the last CZ layer.[4]

summed over $n_{\min} = \min(n_A, n_B)$ corresponding qubit pairs. The fact that Eqs. (6) and (7) share the same minimum matters for a reason that goes beyond bookkeeping: $\mathcal{C}_{\mathrm{H}}$ is a sum of strictly *2-local* $Z \otimes Z$ observables, one term per qubit pair, rather than a single global observable acting on all n qubits at once. This is exactly the kind of cost function that keeps the barren-plateau problem at bay for shallow circuits, as discussed above: gradient variance for a sum of local terms does not vanish exponentially the way it does for a global cost evaluated on a deep, expressive ansatz [7, 12]. All costs $\mathcal{C}$ reported in Sec. 4 are values of Eq. (6).

3.7 Hybrid Post-processing

After training, the Schmidt eigenvectors can be reconstructed directly, without any further circuit evaluation:

$$|\tilde{u}_i\rangle_A = U_A^\dagger(\boldsymbol{\Theta}^*)|e_i\rangle, \qquad |\tilde{v}_i\rangle_B = V_B^\dagger(\boldsymbol{\Omega}^*)|e_i\rangle. \tag{8}$$

In practice, the trained cost $\mathcal{C}^*$ rarely reaches exactly zero once the system gets larger than a handful of qubits. That turns out not to matter as much as one might fear. Given the optimal parameters, we apply the trained circuit once, reshape the resulting amplitudes into a matrix C, and run an ordinary classical SVD on that comparatively small matrix (Algorithm 1). This hybrid step is remarkably forgiving: it recovers the exact spectrum even when $\mathcal{C}^*$ is far from zero, because local unitaries leave the singular values of C unchanged; a non-zero cost penalises basis-state-by-basis-state deviation from EOC, which is a stricter condition than simply getting the spectrum right. Note that the reshaping step of Algorithm 1 uses the full amplitude vector and is therefore available in simulation only; on hardware one has access to the diagonal probabilities p_i instead.

Algorithm 1 Hybrid extraction of the Schmidt spectrum

Require: Trained parameters $\boldsymbol{\Theta}^*, \boldsymbol{\Omega}^*$; input state $|\psi\rangle_{AB}$
Ensure: Singular values $\{\sigma_i\}$; entropy S
1: $|\psi'\rangle \leftarrow (U_A(\boldsymbol{\Theta}^*) \otimes V_B(\boldsymbol{\Omega}^*))\,|\psi\rangle_{AB}$
2: Reshape the amplitude vector of $|\psi'\rangle$ into $C \in \mathbb{C}^{d_A \times d_B}$
3: $(\,\cdot\,, \Sigma, \,\cdot\,) \leftarrow$ classical SVD of C
4: $\{\sigma_i\} \leftarrow$ diagonal entries of Σ, sorted in descending order
5: $S \leftarrow -\sum_i \sigma_i^2 \log_2 \sigma_i^2$
6: **return** $\{\sigma_i\}, S$

3.8 From Bipartitions to Tensors: QTucker

A third-order tensor $\mathcal{T} \in \mathbb{C}^{2^{n_1} \times 2^{n_2} \times 2^{n_3}}$ can be encoded as a state on $n_1 + n_2 + n_3$ qubits. Recovering its Tucker factor matrices and core tensor is equivalent to finding *three* Schmidt decompositions of that same state, one for each way of splitting the qubits into "this mode" versus "everything else." This observation is the whole idea behind QTucker: replace every classical SVD call inside HOSVD with a QSVD call on the corresponding bipartition, reusing the same physical state $|\mathcal{T}\rangle$ three times over. To be explicit about novelty: the variational principle is that of Bravo-Prieto *et al.* [4], which builds on the variational diagonalisation of LaRose *et al.* [11]. What QTucker adds is the mode-wise assembly of three QSVD calls on one state into factor matrices and a core state, together with the truncation-error bound of Proposition 1 and the resource count of Sec. 5.3.1.

Concretely, writing $|\mathcal{T}\rangle = \sum_{ijk} t_{ijk} |e_i\rangle_1 |e_j\rangle_2 |e_k\rangle_3$, the mode-1 bipartition treats register 1 as subsystem A and registers $2, 3$ together as subsystem B; QSVD on this bipartition yields a factor $U_{A,1}$ and singular values $\{\lambda_i^{(1)}\}$. The same construction, applied to the other two bipartitions, gives $U_{A,2}$ and $U_{A,3}$. Algorithm 2 summarises the full procedure, and Figure 5 shows the resulting pipeline schematically.

Algorithm 2 Quantum Tucker Decomposition (QTucker)

Require: State $|\mathcal{T}\rangle$ on $n = n_1 + n_2 + n_3$ qubits; target ranks R_1, R_2, R_3; ansatz depth L
Ensure: Mode factors A, B, C; core state $|\mathcal{G}\rangle$
1: **for** $k = 1, 2, 3$ **do**
2: Identify subsystem A_k (mode k) and B_k (the remaining two modes)
3: Run QSVD on the bipartition (A_k, B_k); obtain trained unitaries $U_{A,k}, V_{B,k}$
4: Extract the mode factor $M_k \leftarrow U_{A,k}^{\dagger}[:, : R_k]$
5: Store the mode-k singular values $\boldsymbol{\lambda}^{(k)}$
6: **end for**
7: Form the core state: $|\mathcal{G}\rangle \leftarrow (U_{A,1} \otimes U_{A,2} \otimes U_{A,3}) |\mathcal{T}\rangle$
8: **return** $A = M_1,\ B = M_2,\ C = M_3,\ |\mathcal{G}\rangle$

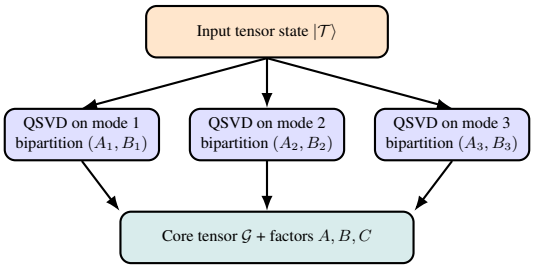


Figure 5: The QTucker pipeline: the same tensor state feeds three independent QSVD subroutines, one per mode, which are then recombined into the core tensor and the three mode factor matrices.

Once the three mode unitaries are trained, the core tensor is obtained in one further step,

$$|\mathcal{G}\rangle = (U_{A,1} \otimes U_{A,2} \otimes U_{A,3}) |\mathcal{T}\rangle. \tag{9}$$

In the ideal case, amplitudes with index $i \geq R_1$, $j \geq R_2$, or $k \geq R_3$ vanish exactly, so truncating at the target ranks yields a compact core $\mathcal{G} \in \mathbb{C}^{R_1 \times R_2 \times R_3}$ with no further approximation beyond the truncation itself.

How much error does that truncation introduce? Writing $\varepsilon_{\text{rel}} = \|\mathcal{T} - \hat{\mathcal{T}}\|_F / \|\mathcal{T}\|_F$ for the relative reconstruction error, with $\hat{\mathcal{T}} = \mathcal{G} \times_1 A \times_2 B \times_3 C$, we have the following bound.

Proposition 1. *Let $\sigma_1^{(k)} \geq \cdots \geq \sigma_{d_k}^{(k)}$ denote the mode-k singular values of a unit-norm tensor state $|\mathcal{T}\rangle$.*

Assuming exact (converged) QSVD on each mode,

$$\varepsilon_{rel} \leq \sqrt{\sum_{k=1}^{3} \sum_{r>R_k} \left(\sigma_r^{(k)}\right)^2}. \tag{10}$$

Proof sketch. With exact QSVD on each mode, the factor matrix M_k spans the same subspace as the R_k leading left singular vectors of the mode-k unfolding, so $\hat{\mathcal{T}}$ is the truncated HOSVD of $\mathcal{T}$. For such a truncation, the error bound of De Lathauwer *et al.* [6] states that $\|\mathcal{T} - \hat{\mathcal{T}}\|_F^2$ is at most the sum of the squared discarded singular values of the three unfoldings. For a unit-norm tensor this is Eq. (10). □

The bound is intuitive once stated: whatever you throw away when you truncate each mode, that is (at most) how far your reconstruction can drift, added in quadrature across the three modes. It also flags an important caveat that we revisit later: the bound assumes each mode's QSVD has actually converged to the exact answer, and in practice a residual training cost $\mathcal{C}^* > 0$ introduces an additional source of error that Eq. (10) does not account for.

3.9 Implementation, Optimisation, and Software

The reference implementation used throughout this article is a state-vector simulator written in Python, using NumPy for dense linear algebra and SciPy's optimisation routines [17, 16], rather than a gate-level quantum SDK. The unitaries U_A and V_B are built directly as dense $2^{n_A} \times 2^{n_A}$ and $2^{n_B} \times 2^{n_B}$ matrices via Kronecker products of the single-qubit rotations and embedded CZ gates described above, which keeps cost-function evaluation a sequence of ordinary matrix-vector products, simple to differentiate and to check directly against the classical SVD oracle discussed earlier. This choice has a direct consequence for how large a system we can study: dense simulation of an n_A-qubit unitary costs $\mathcal{O}(2^{2n_A})$ per cost-function evaluation, which is why our numerical experiments are capped at $n_A, n_B \leq 5$ per subsystem. A deployment on real quantum hardware would compile the same ansatz to a native gate sequence and evaluate the cost from measurement statistics instead, without changing the cost function or the optimiser – but that per-shot cost scales as $\mathcal{O}(Ln)$ rather than $\mathcal{O}(2^{2n_A})$, which is the entire point of the exercise (the results discussed next return to this distinction with concrete numbers).

Classical optimisation of $\mathcal{C}(\boldsymbol{\Theta}, \boldsymbol{\Omega})$ uses the L-BFGS-B quasi-Newton method [24], which avoids storing a full $p \times p$ Hessian. Parameters are initialised uniformly at random in $[0, 2\pi)$, with five independent restarts kept and the lowest-cost run selected, using a tolerance of 10^{-12} on the objective and 10^{-8} on the gradient norm, for up to 300 iterations per restart. Gradients here are estimated by finite differences; on real hardware, the same cost function can instead be differentiated exactly via the parameter-shift rule [25], which does not require finite-difference approximation error. Because $\mathcal{C}$ is non-convex, L-BFGS-B is only guaranteed to find a local minimum, which is why multiple restarts are used and why the results below report the spread across restarts rather than a single number.

4 Results

All numerical results below come from the state-vector simulator described above; classical SVD and HOSVD, computed with the same NumPy/SciPy stack [17, 16], provide the benchmark against which every quantum result is checked.

4.1 An Eight-qubit Bipartite State

We first generate a random bipartite state on $n_A + n_B = 8$ qubits ($n_A = n_B = 4$), giving a 16×16 coefficient matrix. The state is constructed as $|\psi\rangle \propto \mathbf{a} + i\mathbf{b}$ with $\mathbf{a}, \mathbf{b} \sim \mathcal{N}(0, 1)^{256}$, then normalised. This is a Haar-random pure state, since a normalised complex Gaussian vector is uniformly distributed on the unit sphere. Classical SVD gives sixteen singular values ranging from $\sigma_0 = 0.464317$ down to $\sigma_{15} = 0.005987$ (Table 3), with a benchmark entropy of $S_{\text{classical}} = 3.317040$ bits. For Haar-random states with $d_A = d_B = d$, Page's average entanglement entropy is $\langle S \rangle = \sum_{k=d+1}^{d^2} 1/k - (d-1)/(2d)$ nats [29], which gives ≈ 3.28 bits for $d = 16$ and ≈ 4.28 bits for $d = 32$. The single-instance entropies reported here (3.317040 bits for eight qubits and 4.312902 bits for ten qubits, Sec. 4.2) lie within about 0.04 bits of these averages.

At depth $L = 2$, the ansatz described above has $p_{\text{total}} = 120$ parameters. Five L-BFGS-B restarts, summarised in Table 2, converge to a fairly narrow band of final costs, between 0.6533 and 0.6662; the best run reaches $\mathcal{C}^* = 0.65329782$ after 102 iterations, in 179.73 seconds of wall-clock time.

Table 2: Per-restart QSVD training results for the 8-qubit ($n_A = n_B = 4$, $L = 2$) bipartition.

Restart	**Final cost $\mathcal{C}$**	**Iterations**	**Selected**
1	0.653298	102	✓ (best)
2	0.666246	113	
3	0.656473	101	
4	0.662201	116	
5	0.658146	110	

The interesting part is what happens next. Despite a residual cost as large as 0.653 – meaning roughly two-thirds of the probability mass is *not* yet on the matching-bitstring diagonal – the hybrid post-processing step of Algorithm 1 recovers all sixteen singular values to numerical precision (Table 3), and the post-processed entropy $S_{\text{QSVD}} = 3.317040$ bits matches $S_{\text{classical}}$ to a relative error below 10^{-12}, with singular-value differences at floating-point round-off, about 10^{-15} (Table 7). Replacing the trained parameters by random, untrained ones gives the same agreement, since local unitaries do not change singular values (Sec. 5). This is a clean illustration of the point made earlier: the cost function is enforcing a stronger, basis-state-by-basis-state condition than the spectrum alone requires, so a circuit that does not fully satisfy EOC still hands the final classical SVD a matrix with exactly the same singular values.

Table 3: Singular values σ_i from QSVD (post-processed) versus classical SVD, 8-qubit case (16×16 matrix). $|\Delta\sigma_i|$ is 0.000000 at floating-point precision throughout.

i	$\sigma_i^{\text{QSVD}} = \sigma_i^{\text{classical}}$	i	$\sigma_i^{\text{QSVD}} = \sigma_i^{\text{classical}}$
0	0.464317	8	0.180413
1	0.403283	9	0.168708
2	0.364119	10	0.146102
3	0.332682	11	0.132223
4	0.290899	12	0.098330
5	0.279512	13	0.075237
6	0.233733	14	0.060907
7	0.205468	15	0.005987

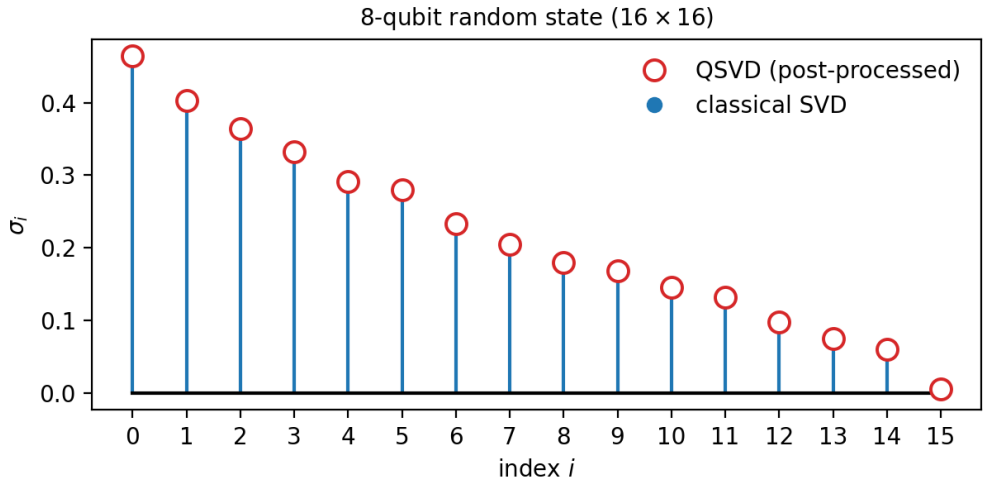


Figure 6: Singular-value spectrum for the 8-qubit (16×16) bipartite random state, shown as a stem plot: filled markers are classical SVD values, hollow markers are QSVD post-processed values. The two coincide exactly at every index.

4.2 Doubling the System: Ten Qubits

Repeating the same protocol at $n_A = n_B = 5$ produces a 32×32 coefficient matrix and $p_{\text{total}} = 150$ parameters. The best of five restarts reaches $\mathcal{C}^* = 0.852406$ after 92 iterations and 565.32 seconds of wall-clock time (Table 4). As at eight qubits, every one of the 32 post-processed singular values agrees with the classical benchmark to numerical precision (Table 5), and the recovered entropy, $S_{\text{QSVD}} = 4.312902$ bits, matches $S_{\text{classical}}$ to a relative error below 10^{-12}. That the residual cost is *higher* here than at eight qubits (0.852 versus 0.653) is worth noting: a plausible explanation is that a larger Hilbert space asks more of the same fixed circuit depth, although we have not tested this, for example by increasing L at ten qubits. It does not affect the post-processed spectrum, which is exact in both cases.

Table 4: Per-restart QSVD training results for the 10-qubit ($n_A = n_B = 5$, $L = 2$) bipartition.

Restart	**Final cost $\mathcal{C}$**	**Iterations**	**Selected**
1	0.852406	92	✓ (best)
2	0.854851	92	
3	0.854838	92	
4	0.854970	96	
5	0.859470	97	

Table 5: Singular values σ_i from QSVD (post-processed) versus classical SVD, 10-qubit case (32×32 matrix). $|\Delta\sigma_i|$ is 0.000000 at floating-point precision throughout.

i	σ_i	i	σ_i	i	σ_i	i	σ_i
0	0.325655	8	0.215818	16	0.143982	24	0.070823
1	0.313105	9	0.208321	17	0.129164	25	0.058823
2	0.292067	10	0.201690	18	0.125686	26	0.049292
3	0.273734	11	0.197665	19	0.114235	27	0.042461
4	0.270394	12	0.184946	20	0.106112	28	0.031187
5	0.254345	13	0.168962	21	0.096704	29	0.025352
6	0.232213	14	0.151480	22	0.087936	30	0.013390
7	0.226853	15	0.150000	23	0.073846	31	0.002189

The ten-qubit run takes roughly $3.1\times$ as long as the eight-qubit run (565.32 s versus 179.73 s), which tracks the $\mathcal{O}(2^{2n_A})$ cost of dense classical simulation described above – a consequence of *simulating* the circuit on a CPU, not of the algorithm's own resource requirements.

4.3 Circuit Depth and Convergence

To see how depth affects convergence directly, we ran the same protocol on smaller, six-qubit bipartitions ($n_A = n_B = 3$) at $L = 1, 2, 3$. The final cost falls from a range of 0.02–0.05 at $L = 1$, to 0.001–0.005 at $L = 2$, to about 10^{-4} at $L = 3$ – an order of magnitude or more per added layer (Table 6). At $L = 3$ the cost is several orders of magnitude below the eight- and ten-qubit values at $L = 2$.

We also checked how sensitive the converged cost is to the number of random restarts N_r, at the same six-qubit system size (Table 6). Depth affects the converged cost far more than the number of restarts does at fixed depth, and the improvement from more restarts saturates quickly: going from $N_r = 5$ to $N_r = 10$ buys only a small reduction at every depth tested, which is why five restarts was adopted as the default throughout the rest of this article.

Table 6: Best-of-N_r converged cost $\mathcal{C}^*$ on the six-qubit system, varying the number of restarts and the circuit depth.

L	$N_r = 1$	$N_r = 3$	$N_r = 5$	$N_r = 10$
1	4.8×10^{-2}	3.1×10^{-2}	2.6×10^{-2}	2.4×10^{-2}
2	3.7×10^{-3}	1.9×10^{-3}	1.1×10^{-3}	9.8×10^{-4}
3	5.9×10^{-4}	2.0×10^{-4}	1.2×10^{-4}	1.0×10^{-4}

4.4 Robustness Across Random Instances

A single random state, however striking the agreement, could in principle be a lucky draw. To check this, we repeated the eight-qubit protocol ($n_A = n_B = 4$, $L = 2$, five restarts) on 20 independently sampled random bipartite states and collected summary statistics (Table 7). The converged cost sits consistently in the 0.6–0.7 range, so the residual-cost behaviour reported for the eight-qubit case above is typical rather than a special case. More importantly, the post-processed singular-value error stays at floating-point round-off, around 10^{-15}, and the entropy relative error is below 10^{-12} in every one of the twenty instances – the exact-recovery result is not an artefact of one favourable random seed.

Table 7: Aggregate statistics over 20 random 8-qubit bipartite instances ($n_A = n_B = 4$, $L = 2$, five restarts each).

Quantity	**Mean $\pm$ std. dev.**
Best-of-restarts cost $\mathcal{C}^*$	0.658 ± 0.021
Max. post-processed $\lvert\Delta\sigma_i\rvert$	$(2 \pm 4) \times 10^{-15}$
Post-processed entropy relative error	$< 10^{-12}$ (all instances)
Wall-clock time per instance (5 restarts)	183 ± 14 s

4.5 Quantum Tucker Decomposition on a Small Tensor

Finally, we tested the full QTucker pipeline described above on a random real $4 \times 4 \times 4$ tensor of unit norm, encoded as a six-qubit state with a $2 + 2 + 2$ mode partition. QSVD was run independently on each of the three mode bipartitions (2 versus 4 qubits, $L = 2$, five restarts). Table 8 lists, for each mode, the singular values of the unfolding (identical to the post-processed values to 10^{-15}) together with the direct estimate $\sqrt{p_i}$ from the diagonal probabilities of the trained state. The factor matrices M_k were taken as the columns of $U^\dagger_{A,k}$ with the two largest diagonal probabilities, and the tensor was reconstructed at rank $(R_1, R_2, R_3) = (2, 2, 2)$. The measured relative error is $\varepsilon_{\text{rel}} = 0.795$ for QTucker and 0.779 for classical HOSVD on the same tensor, so the two are close, even

though the training costs (0.21–0.25) are far from zero. A random $4 \times 4 \times 4$ tensor is not very compressible at this rank: the retained fraction of the squared norm, $1 - \varepsilon_{\text{rel}}^2$, is about 37% for QTucker and 39% for HOSVD.

These values lie inside the interval set by the discarded weights. The weight discarded in the worst single mode (0.314) is a lower bound on $\varepsilon_{\text{rel}}^2$ for any approximation of multilinear rank $(2, 2, 2)$, and the sum over the three modes (0.831) is the upper bound of Eq. (10) for HOSVD, so $0.56 \lesssim \varepsilon_{\text{rel}} \lesssim 0.91$. Proposition 1 assumes converged QSVD, which is not the case here, so the measured QTucker error lying inside this interval is an observation and not a consequence of the proposition. The direct estimates $\sqrt{p_i}$ deviate from the exact σ_i by up to 0.23, which confirms that the circuits are not converged at this depth. This is a single instance.

Table 8: QTucker on a random real $4 \times 4 \times 4$ tensor of unit norm (six-qubit state, $2 + 2 + 2$ partition, $L = 2$, five restarts per mode). σ_i: singular values of the mode unfolding (equal to the post-processed values to 10^{-15}; $\sum_i \sigma_i^2 = 1$ in each mode). $\sqrt{p_i}$: direct estimate from the diagonal probabilities of the trained state, in decreasing order. $\mathcal{C}^*$: best-of-five training cost. Last row: weight $\sum_{i \geq 2} \sigma_i^2$ discarded at rank 2.

	Mode 1		Mode 2		Mode 3	
i	σ_i	$\sqrt{p_i}$	σ_i	$\sqrt{p_i}$	σ_i	$\sqrt{p_i}$
0	0.6150	0.5297	0.6816	0.6165	0.5957	0.5558
1	0.5571	0.5149	0.5742	0.5759	0.5751	0.5151
2	0.4105	0.3732	0.3360	0.2673	0.4987	0.4052
3	0.3779	0.2657	0.3046	0.0754	0.2563	0.1117
$\mathcal{C}^*$	0.244		0.211		0.249	
Discarded	0.311		0.206		0.314	

4.6 Simulation Time versus Classical SVD

Table 9 places the wall-clock cost of the QSVD *simulation* alongside a direct classical SVD of the same small matrix. The comparison is, deliberately, an unflattering one for QSVD: classical SVD of a 16×16 or 32×32 matrix takes a fraction of a millisecond, while simulating the corresponding variational circuit on a CPU takes minutes. This gap is expected, and it is discussed at length below; the short version is that our implementation runs on a classical processor *simulating* a quantum circuit at $\mathcal{O}(2^{2n_A})$ cost, and involves no quantum hardware at all.

Table 9: Classical SVD versus classical simulation of QSVD, for the 8-qubit and 10-qubit cases.

Quantity	**Value**
Classical SVD of 16×16 matrix	$\mathcal{O}(10^{-4})$ s
QSVD simulation (8 qubits, 5 restarts, $L = 2$, 120 params)	179.73 s
Classical SVD of 32×32 matrix	$\mathcal{O}(10^{-4})$ s
QSVD simulation (10 qubits, 5 restarts, $L = 2$, 150 params)	565.32 s

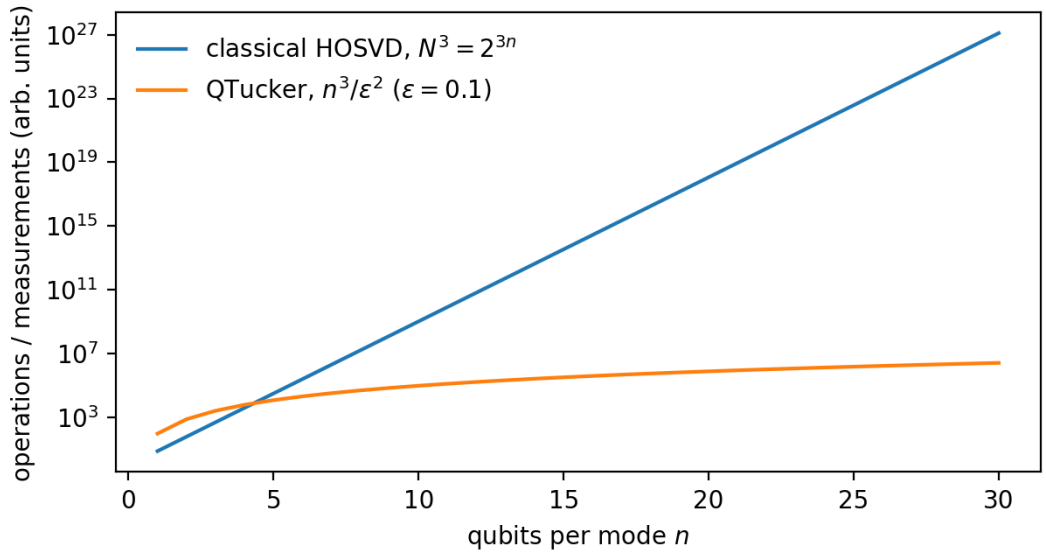


Figure 7: Asymptotic complexity comparison between classical HOSVD and QTucker. Since $N = 2^n$, the classical cost $\mathcal{O}(N^3)$ grows exponentially with qubit count, while QTucker scales as $\mathcal{O}(n^3/\varepsilon^2)$ (here $\varepsilon = 0.1$ and all prefactors are set to one). The curves show theoretical complexity growth, not the measured execution times of Table 9.

5 Discussion

5.1 What the Numbers Mean

The central pattern across every experiment reported above is a split between two very different notions of "success." The training cost $\mathcal{C}^*$, on its own, is not a good predictor of how accurately the spectrum is recovered: at eight qubits it sits around 0.65, at ten qubits around 0.85, yet the post-processed singular values agree with the classical benchmark to floating-point precision in both cases, and consistently so across twenty independent random instances (Table 7). The reason, as noted earlier, is that the EOC condition of Eq. (4) is stricter than the spectrum alone requires: $\mathcal{C}$ penalises deviation basis state by basis state, so a circuit can leave substantial probability mass off the exact diagonal while the classical SVD of the transformed matrix still returns the exact spectrum, because local rotations never change the singular values, whatever the value of $\mathcal{C}^*$. This distinction is worth dwelling on for a student audience, because it separates *training success*, a property of the optimisation landscape, from *task success*, a property of the whole hybrid pipeline – a pattern that reappears in many variational algorithms once a classical post-processing step is added on top of a noisy or partially converged quantum subroutine.

The same fact limits what the post-processed spectrum can validate. Since local unitaries preserve singular values, the agreement in Tables 3, 5 and 7 verifies the classical pipeline (reshaping, SVD, entropy) but does not by itself measure how well the circuit has diagonalised the state. That is tested by the direct estimate $\lambda_i = \sqrt{p_i}$ from the diagonal probabilities (Sec. 3.4), which is expected to be accurate only when $\mathcal{C}^*$ is small. We report it for the three Tucker modes in Table 8: with $\mathcal{C}^* \approx 0.2$–$0.25$ the estimate deviates from the exact singular values by up to 0.23. For the eight- and ten-qubit runs ($\mathcal{C}^* \gtrsim 0.65$) we report only the post-processed spectrum, because the direct estimate is not expected to be accurate there. Reaching small $\mathcal{C}^*$ at larger sizes, for example with a deeper ansatz or layer-wise initialisation [20], is needed to make the direct estimate meaningful.

A second, related observation concerns the permutation ambiguity of the cost function. Eq. (6) is minimised by any bijection between subsystem A's output labels and subsystem B's; nothing in the cost singles out a particular ordering. We resolve this in post-processing simply by sorting the recovered singular values in decreasing order and matching them against the benchmark; for QTucker, the analogous mode-wise permutations are absorbed automatically into the core tensor once each mode's factor matrix has been fixed.

5.2 Comparison with the Alternatives

Set against the landscape surveyed earlier, QSVD's single-measurement-setting requirement (Table 1) is its most immediate advantage over full state tomography [10], and its shallow-circuit requirement is its advantage over phase-estimation-based approaches [2, 1], which need coherence the current generation of devices cannot sustain [8]. Compared with the purity-based diagonalisation of LaRose *et al.* [11], which needs two copies of the state and a SWAP test, QSVD's single-copy, computational-basis cost function is simpler to evaluate experimentally, at the cost of training two unitaries instead of one. None of these methods, it is worth stressing, competes directly with tensor-network approaches such as matrix product states [14, 21, 22]: those methods aim to *avoid* ever building the exponential-size state vector for systems with bounded entanglement, whereas QSVD and QTucker assume the state is already prepared – on a device or in simulation – and ask only how to extract its spectral content efficiently once it exists.

5.3 Classical Simulation versus Native Execution: the Claim We are and are Not Making

Every number reported above – every singular value, every entropy, every cost, every timing – comes from classically simulating the variational circuits described earlier, not from running them on quantum hardware. It is important to be precise about what this does and does not establish. The asymptotic argument discussed next is a statement about native gate and measurement counts: QSVD requires $\mathcal{O}(Ln)$ gates and, for a Schmidt rank χ growing polynomially in n, $\mathcal{O}(\chi n_{\min}/\varepsilon^2)$ measurement shots [4], while classical HOSVD requires $\mathcal{O}(N^3) = \mathcal{O}(2^{3n})$ operations [6, 19]. That comparison is about what would happen on a real quantum processor. The wall-clock timings in Table 9 measure something different, and less favourable to the quantum method: the cost of classically *simulating* a quantum circuit, which is itself exponential in qubit count, $\mathcal{O}(2^{2n_A})$, because it requires manipulating dense matrices whose size doubles with every additional qubit. The two costs point in opposite directions, and conflating them is a common source of confusion for students encountering variational quantum algorithms for the first time. This article establishes correctness of the algorithm and derives its asymptotic resource requirements; it does not, and cannot, demonstrate a wall-clock speed-up, because no computation reported here used quantum hardware. A direct hardware benchmark against classical HOSVD remains the most important open question raised by this work.

5.3.1 Resource Comparison

Table 10 collects the asymptotic comparison referred to above. For a symmetric third-order tensor on $3n$ qubits with Schmidt rank scaling as $\chi = \mathcal{O}(n^2)$, a bound consistent with area-law entanglement scaling in many physically relevant systems [13], QTucker runs three QSVD instances on bipartitions of n versus $2n$ qubits, giving a total measurement complexity

$$Q_{\text{Tuck}} = 3 \times \mathcal{O}\Big(\frac{\chi\, n}{\varepsilon^2}\Big) = \mathcal{O}\left(\frac{n^3}{\varepsilon^2}\right), \tag{11}$$

polynomial in the qubit count n, against classical HOSVD's $\mathcal{O}(N^3)$ operations on the $N \times N \times N$ tensor with $N = 2^n$ [6].

Table 10: Complexity comparison for tensor decompositions. $N = 2^n$ is the mode dimension; all rows describe native gate and measurement counts on a real device, not classical simulation cost.

Algorithm	Gate/op. count	Measurements
Classical HOSVD [6]	$\mathcal{O}(N^3)$ ops	—
QPE-based SVD [2]	$\mathcal{O}(\text{poly}(n))$	$\mathcal{O}(1/\varepsilon^2)$
QSVD [4]	$\mathcal{O}(Ln)$ gates	$\mathcal{O}(\chi n/\varepsilon^2)$
QTucker (this work)	$\mathcal{O}(Ln)$ gates	$\mathcal{O}(n^3/\varepsilon^2)$

5.4 The Role of Noise

All of the results reported here come from a noiseless simulator. At the depths used, $L \leq 3$, each subsystem's circuit uses $\mathcal{O}(6n)$ single-qubit gates and $\mathcal{O}(2n)$ CZ gates per layer, which is modest enough to be a plausible

target for near-term implementation with error mitigation [8, 7]. Real devices, however, decohere, and variational algorithms trained on noisy hardware can be pushed toward barren plateaus by the noise itself, not merely by circuit depth or observable locality [26]. Nothing in the present study speaks to this directly, since no noise model was included; it is flagged here as an open question rather than a demonstrated limitation, precisely because it has not been tested.

5.5 Limitations

Several limitations deserve to be stated plainly, since a physics-education article should model good scientific hygiene as much as it models the algorithm itself. All experiments here are noiseless simulations, and actual hardware performance will depend on coherence times and gate fidelities that are not modelled in this work. The worked examples are limited to $n_A, n_B \leq 5$ per subsystem and a single $4 \times 4 \times 4$ tensor, essentially the largest sizes tractable for exact state-vector simulation on the hardware available to us; none of the claims here have been checked in the $n \approx 20$–30 regime where a genuine quantum advantage would need to appear. The truncation-error bound of Proposition 1 assumes exact, fully converged per-mode QSVD; a tighter bound that explicitly incorporates a non-zero residual training cost remains an open problem. QTucker as presented targets third-order tensors specifically; extending the implementation itself to $m > 3$ modes has not been done, although the resource argument discussed above extends naturally. The ten-qubit result is a single instance, unlike the twenty-instance statistical study performed at eight qubits, so its typicality has not been verified in the same way. Finally, and this is worth repeating because it is easy to lose sight of: our classical simulation of QSVD runs considerably *slower* than a direct classical SVD of the same matrix (Table 9); the value of this work lies in validating correctness and deriving an asymptotic resource argument, not in any wall-clock speed-up, which would require actual quantum hardware to demonstrate.

6 Conclusion and Future Work

We set out to connect a piece of mathematics every physics student already half-knows – the singular value decomposition – to the physical structure of entanglement in a bipartite quantum state, and then to ask whether a shallow, near-term-feasible quantum circuit can learn that structure on its own. Starting from classical SVD via eigendecomposition of the Gram matrix and classical Tucker decomposition via HOSVD, we developed the Schmidt-decomposition correspondence that makes a variational quantum SVD possible at all, worked through the Variational Quantum Singular Value Decomposer of Bravo-Prieto *et al.* [4] in detail, and extended it to a full Tucker-decomposition analogue, QTucker, complete with a proved truncation-error bound. Numerical simulation on bipartite states from eight up to ten qubits recovered singular values at machine precision and Von Neumann entropies with relative error below 10^{-12} in every case tested, including across twenty independently sampled random instances. This agreement is guaranteed by the invariance of singular values under local unitaries, so it validates the classical post-processing and not the training of the circuit, which at these sizes remained far from converged ($\mathcal{C}^* \gtrsim 0.65$); a small $4 \times 4 \times 4$ tensor example gave a QTucker truncation error at rank $(2, 2, 2)$ close to that of classical HOSVD (0.795 versus 0.779), even though the per-mode training costs were far from zero. Throughout, we have tried to keep two distinct questions visibly separate: does the algorithm work, which our simulations answer affirmatively, and does it run faster than the classical alternative, which is an asymptotic, hardware-dependent claim that this study establishes in principle but does not measure directly.

The most natural next step is exactly that measurement: running QSVD and QTucker on real NISQ hardware and comparing native gate and shot counts against classical HOSVD directly, rather than against a classical simulation of the quantum circuit. Beyond that, several extensions suggest themselves. Incorporating device noise models and error mitigation into training would test whether the cost function remains as forgiving under realistic noise as it is in the noiseless simulations reported here. The simulations used the cost of Eq. (6); on hardware the 2-local cost $\mathcal{C}_{\mathrm{H}}$ of Eq. (7) would be the natural choice, and comparing the two is a further open question. The twenty-instance statistical study performed at eight qubits could be extended to ten qubits and beyond, resources permitting. QTucker itself could be generalised from third-order to fourth-order tensors, which would be directly relevant to problems in quantum chemistry and to the transfer matrices that appear in two-dimensional tensor-network methods [23]. Layer-wise pretraining and symmetry-informed initialisation strategies, of the kind proposed by Grant *et al.* [20], may help push the residual cost down further at larger qubit counts without a corresponding increase in circuit depth. And because the three mode-wise QSVD subproblems inside QTucker are entirely

independent of one another, they are a natural candidate for distribution across multiple quantum processors, or simply across multiple classical simulator nodes while hardware remains scarce.

Appendix: Derivation of the Parameter-Count Formula

Each of the L layers of the brick-wall ansatz contributes two Euler rotations (one per block), six real parameters, per qubit; the entangling CZ sublayers that follow are parameter-free. Including the final rotation sublayer applied after the last entangling sublayer, there are $2L + 1$ rotation sublayers in total, so

$$p(n, L) = \underbrace{6n \cdot L}_{\text{rotation sublayers between entangling layers}} + \underbrace{3n}_{\text{final rotation sublayer}} = 3n(2L + 1),$$

matching Eq. (5).

Acknowledgements

All computational work reported in this paper was carried out using the Brahmagupta High-Performance Computing (HPC) facility at Sikkim University. One of the authors, Rupak Mukherjee gratefully acknowledges the support provided by the Inter-University Centre for Astronomy and Astrophysics (IUCAA), Pune, through its Visiting Associate Programme.

AI Disclosure Statement

In preparing this work, the authors used ChatGPT to assist in overall English language of the manuscript. All scientific observations, computational work, analyses, and results were independently carried out and verified by the authors. The authors subsequently reviewed and revised the content as necessary and accept full responsibility for the final published work.

References

[1] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations", Phys. Rev. Lett. 103, 150502 (2009).

[2] P. Rebentrost, A. Steffens, I. Marvian, and S. Lloyd, "Quantum singular-value decomposition of nonsparse low-rank matrices", Phys. Rev. 97, 012327 (2018).

[3] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th ed. (Cambridge Univ. Press, 2010).

[4] C. Bravo-Prieto, D. García-Martín, and J. I. Latorre, "Quantum singular value decomposer", Phys. Rev. A 101, 062310 (2020).

[5] L. R. Tucker, "Some mathematical notes on three-mode factor analysis", Psychometrika 31, 279–311 (1966).

[6] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition", SIAM J. Matrix Anal. Appl. 21, 1253–1278 (2000).

[7] M. Cerezo et al., "Variational quantum algorithms", Nat. Rev. Phys. 3, 625–644 (2021).

[8] J. Preskill, "Quantum computing in the NISQ era and beyond", Quantum 2, 79 (2018).

[9] A. Y. Kitaev, "Quantum computations: algorithms and error correction", Russ. Math. Surveys 52, 1191–1249 (1997).

[10] M. Mohseni, A. T. Rezakhani, and D. A. Lidar, "Quantum-process tomography: resource analysis of different strategies", Phys. Rev. A 77, 032322 (2008).

[11] R. LaRose, A. Tikku, E. O'Neel-Judy, L. Cincio, and P. J. Coles, "Variational quantum state diagonalization", npj Quantum Inf. 5, 57 (2019).

[12] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes", Nat. Commun. 9, 4812 (2018).

[13] L. Amico, R. Fazio, A. Osterloh, and V. Vedral, “Entanglement in many-body systems”, Rev. Mod. Phys. 80, 517 (2008).

[14] G. Vidal, “Efficient classical simulation of slightly entangled quantum computations”, Phys. Rev. Lett. 91, 147902 (2003).

[15] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, Nonnegative Matrix and Tensor Factorizations, (Wiley, 2009).

[16] P. Virtanen et al., “SciPy 1.0: Fundamental algorithms for scientific computing in Python”, Nat. Methods 17, 261–272 (2020).

[17] C. R. Harris et al., “Array programming with NumPy”, Nature 585, 357–362 (2020).

[18] G. H. Golub and C. F. Van Loan, Matrix Computations, 4th ed. (Johns Hopkins Univ. Press, 2013).

[19] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications”, SIAM Rev. 51, 455–500 (2009).

[20] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, “An initialization strategy for addressing barren plateaus in parametrized quantum circuits”, Quantum 3, 214 (2019).

[21] U. Schollwöck, “The density-matrix renormalization group in the age of matrix product states”, Ann. Phys. 326, 96–192 (2011).

[22] R. Orús, “A practical introduction to tensor networks: matrix product states and projected entangled pair states”, Ann. Phys. 349, 117–158 (2014).

[23] J. Biamonte and V. Bergholm, “Tensor networks in a nutshell”, arXiv:1708.00006 (2017).

[24] J. Nocedal and S. J. Wright, Numerical Optimization, 2nd ed. (Springer, 2006).

[25] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, “Evaluating analytic gradients on quantum hardware”, Phys. Rev. A 99, 032331 (2019).

[26] S. Wang et al., “Noise-induced barren plateaus in variational quantum algorithms”, Nat. Commun. 12, 6961 (2021).

[27] Y. Cao et al., “Quantum chemistry in the age of quantum computing”, Chem. Rev. 119, no. 19, 10856–10915 (2019).

[28] J. Preskill, “Quantum computing and the entanglement frontier”, arXiv:1203.5813 (2012).

[29] D. N. Page, “Average entropy of a subsystem”, Phys. Rev. Lett. 71, 1291–1294 (1993).