Classical Simulation of Quantum Circuits using Tensor Networks: Matrix Product States (MPS) for Quantum Approximate Optimization Algorithms (QAOA)

Annwoy Roy Choudhury

BS-MS Student in Physics (2022-2027)
Indian Institute of Science Education and Research (IISER), Mohali, Punjab, India

Research Internship (May-July 2025) at the
Center for Optical Quantum Technologies, University of Hamburg, Germany

Supervisor: Prof. Dr. Peter Schmelcher (Universität Hamburg)

Local Guide: Prof. Arvind (IISER Mohali)

Abstract

This report summarizes the work undertaken during a research internship focused on the classical simulation of quantum circuits using tensor networks. It delves into the foundational theoretical concepts of Matrix Product States (MPS), detailing their structure, various tensor decompositions (SVD, QR, RQ), and canonical forms which are crucial for their manipulation. The report then explores the application of these MPS techniques to the simulation of Quantum Approximate Optimization Algorithms (QAOA), outlining its origins from Adiabatic Quantum Computation and its problem-mapping methodologies. A significant part of the work involved the implementation of a Python framework to perform these simulations, with a particular emphasis on the efficient application of gates and the critical distinction between exact and inexact simulation strategies. The report presents key preliminary results on entanglement growth within QAOA circuits and the accuracy of infidelity calculations for various compression approaches, highlighting the immense potential of MPS-based methods for achieving scalable analysis of quantum circuits.

Contents

1 Introduction

Quantum computing holds immense promise for solving problems currently intractable for classical machines. However, a fundamental challenge persists in simulating these quantum systems classically: the *exponential growth of the Hilbert space*. A quantum state of N qubits is described by a state vector whose dimension scales as 2^N. This directly implies that the memory required for storage and the computational cost for manipulating such a state also scale exponentially, rendering exact classical simulation quickly intractable for systems exceeding $N > 40-50$ qubits.

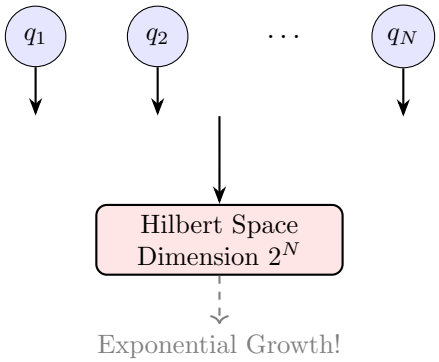

Figure 1: The Hilbert space dimension scales exponentially with the number of qubits, illustrating the "curse of dimensionality."

This inherent limitation motivates the development of efficient approximate methods. *Tensor Networks* (TNs) provide a powerful framework for efficiently representing and manipulating certain classes of quantum states, notably those with limited entanglement. *Matrix Product States* (MPS), a specific type of TN, are particularly effective for 1D quantum systems, leveraging the "Area Law" for entanglement. MPS enable classical simulation of quantum circuits that would otherwise be beyond reach, offering insights into quantum algorithms and serving as a benchmark for future quantum hardware.

The internship focused on the classical simulation of *Quantum Approximate Optimization Algorithms* (QAOA), a leading candidate for near-term quantum advantage. The project involved developing a Python framework, implementing core MPS functionalities, simulating QAOA circuits for optimization problems, and analyzing performance metrics like entanglement growth and fidelity.

2 Theoretical Foundations

2.1 Matrix Product States (MPS): The Quantum State Representation

A quantum state of N qubits is generally described by a high-dimensional coefficient tensor $C_{\sigma_1,\ldots,\sigma_N}$. MPS offer a compressed factorization of this tensor into a product of smaller tensors (matrices). This factorization is particularly efficient for one-dimensional quantum systems where entanglement typically obeys an "area law", scaling with the boundary size rather than the system volume.

$$C_{\sigma_1,\ldots,\sigma_N} \approx A^{\sigma_1}_{\alpha_0,\alpha_1} A^{\sigma_2}_{\alpha_1,\alpha_2} \cdots A^{\sigma_N}_{\alpha_{N-1},\alpha_N}$$

Here, σ_i are *physical indices* representing local states (dimension d, e.g., $d = 2$ for qubits), and α_i are *virtual/bond indices* (dimension χ or D). These virtual bonds carry the quantum entanglement information between different parts of the system.

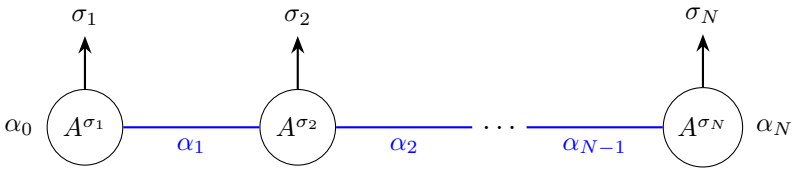

Figure 2: Diagrammatic representation of a Matrix Product State (MPS) chain. Circles are tensors, black lines are physical indices, blue lines are virtual/bond indices.

2.2 Tensor Decompositions: Fundamental Tools

Tensor decomposition is the process of breaking down a large, complex tensor into a product of simpler, structured tensors. This is analogous to matrix factorization and is fundamental to creating, manipulating, and compressing MPS.

The *Singular Value Decomposition (SVD)* ($M = USV^\dagger$) is arguably the most important decomposition in tensor network analysis. Its non-negative real singular values (s_α) directly quantify entanglement (they are the Schmidt coefficients) and enable optimal compression. SVD guarantees the best possible low-rank approximation (Eckart-Young Theorem).

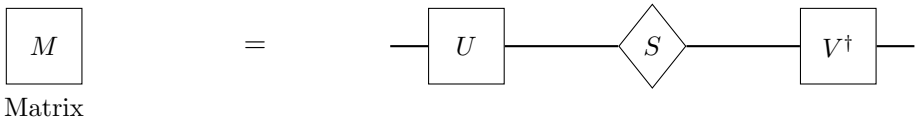

Figure 3: Visual representation of the Singular Value Decomposition (SVD).

QR and RQ decompositions ($M = QR$ or $M = RQ$) are computationally faster alternatives to SVD. They produce an isometric matrix (Q) and a triangular matrix (R). Unlike SVD, they do not provide entanglement information and are primarily used when truncation based on singular values is not required (e.g., in canonicalization sweeps). These decompositions can be generalized to act on specific "modes" (indices) of higher-rank tensors by reshaping the tensor into an effective matrix.

2.3 Working with MPS: Canonical Forms and Contractions

Gauge freedom in MPS means that a given quantum state can be represented by multiple sets of tensors. To manage this ambiguity and facilitate calculations, MPS tensors are transformed into *canonical forms*. These forms impose specific orthogonality properties: *Left-Canonical* ($A^\dagger A = I$) for tensors on the left, and *Right-Canonical* ($BB^\dagger = I$) for tensors on the right. The

Mixed-Canonical form combines these, with an *orthogonality center* (an unnormalized tensor) that holds all local norm and entanglement information. The *Vidal Gauge* is a special canonical form where singular values (Λ) are explicitly placed on the virtual bonds connecting isometric Γ tensors.

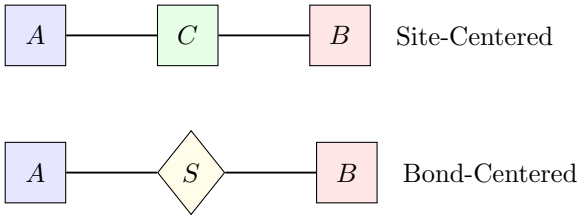

Figure 4: Common types of orthogonality centers in an MPS chain.

Tensor contraction is the fundamental operation for combining tensors by summing over common indices. The efficiency of this process critically depends on the *contraction order*. The *Zipper Algorithm* is an optimized iterative method for contracting entire MPS chains (e.g., for inner products or expectation values). It achieves a polynomial scaling of $O(L \cdot \chi^3 \cdot d)$ total cost, a massive improvement over the exponential cost of full state vector methods.

2.4 MPS for Quantum Circuit Simulation

Constructing an MPS from a quantum state typically uses an iterative SVD procedure, where tensors are "peeled off" one site at a time. Quantum gate application involves applying local unitary operators directly to MPS tensors. Single-qubit gates are simple local contractions that do not change bond dimensions. Two-qubit gates are more complex; they increase entanglement, requiring a multi-step "evolve-then-compress" cycle involving contraction, SVD, and truncation to manage the bond dimension.

2.5 MPS Compression Techniques

Continuous application of gates in quantum circuits causes the bond dimension (χ) to grow exponentially with circuit depth, rendering exact simulation intractable. *Compression* techniques circumvent this by limiting χ to a fixed maximum ($\chi_{\max}$), trading perfect accuracy for computational tractability.

- **SVD Compression:** A fast, direct, and non-iterative method. It performs local SVDs and truncates singular values, providing a locally optimal but globally suboptimal approximation as errors accumulate.
- **Variational Compression:** An iterative, globally optimal approach that minimizes the distance between the exact state and the compressed MPS. It is highly accurate for a fixed bond dimension but is computationally more demanding than SVD compression.

2.6 Quantum Approximate Optimization Algorithm (QAOA): Theory and Problem Mapping

QAOA is inspired by *Adiabatic Quantum Computation* (AQC), which aims to find a problem Hamiltonian's ground state through slow, continuous evolution. QAOA discretizes this process into p layers of alternating Hamiltonian applications ($e^{-i\gamma_k H_C} e^{-i\beta_k H_B}$), where γ_k, β_k are variational parameters optimized by a classical computer. The *Cost Hamiltonian* (H_C) encodes the problem (e.g., Max-Cut), while the *Mixer Hamiltonian* (H_B) drives exploration. Classical optimization problems like Max-Cut are typically mapped to *Quadratic Unconstrained Binary Optimization* (QUBO) forms, which are then transformed into *Ising Hamiltonians* ($H_C =$

$\sum h_i Z_i + \sum J_{ij} Z_i Z_j$) suitable for quantum execution. The full pipeline connects the classical problem to a quantum circuit acting on an MPS.

3 Implementation & Methodology

3.1 Overview of the Python Framework

The implemented Python framework is modular and relies on `NumPy` for efficient tensor operations. It is structured into distinct modules for clear functionality separation.

- **General Tensor Functions:** Core operations like SVD, QR, and RQ decompositions.
- **Basic MPS Functions:** Handles MPS canonicalization, norm/amplitude calculation, and inner products (using the Zipper Algorithm).
- **MPS Gate Applications:** Implements single-qubit and two-qubit gate application workflows on MPS tensors.
- **QAOA Simulation & Analysis:** Orchestrates the full QAOA algorithm, including exact and inexact simulation methods.

The workflow proceeds from input quantum states or problems, through these modules, to output optimization results or simulation analysis.

3.2 Core Building Blocks: From Math to Code

Theoretical concepts are directly translated into Python functions. For instance, `svd_tensor_mode()` and `qr_tensor_mode()` implement tensor decompositions by mode. MPS canonicalization is handled by functions like `make_left_canon()` and `make_mixed_canon()`. Efficient MPS utilities include `inner_product()` (via Zipper Algorithm) and `calculate_expectation_value()`. Compression algorithms are implemented by `svd_compression_fixed_bond()` (greedy) and `variational_compressi` (optimal). `NumPy`'s optimized array manipulation is crucial for the efficiency of these tensor operations.

3.3 Simulating Quantum Circuits with MPS: The Approach

The framework simulates QAOA circuits by applying quantum gates directly to MPS states using functions like `apply_single_qubit_gate()` and `apply_two_qubit_gate()`. The simulation process follows a hybrid QAOA optimization loop: a classical optimizer proposes variational parameters $(\vec{\gamma}, \vec{\beta})$, the MPS simulator executes the layered QAOA circuit with these parameters, computes the expectation value of the Cost Hamiltonian (H_C), and provides feedback to the optimizer for subsequent iterations.

3.4 Exact vs. Inexact Simulation: A Crucial Distinction

Exact classical simulation of quantum circuits (without compression) faces an exponential increase in bond dimension ($\chi \sim 2^p$), making it intractable for large systems or deep circuits. For small systems, the exact state (Ψ_{exact}) can be computed and used as a benchmark to quantify infidelity ($1 - |\langle\Psi_{\text{exact}}|\Psi_{\text{inexact}}\rangle|^2$) of compressed simulations. Inexact simulation, however, enforces a fixed maximum bond dimension ($\chi_{\max}$) and employs compression algorithms (SVD or Variational) after each operation to achieve polynomial scaling, making it a practical approach for larger systems.

3.5 Fidelity Estimation: Quantifying Inexactness

For large inexact simulations, the true exact state (Ψ_{exact}) becomes computationally unreachable, making direct infidelity calculation impossible. To address this, the *Multiplicative Fidelity Law* is used, approximating total fidelity as $F_{\text{total}} \approx \prod_{\delta=1}^{m} f_\delta$. Each partial fidelity f_δ (the fidelity of the compressed state with the uncompressed state at that step) is calculated during the simulation. This provides an accurate, internally consistent estimate of the final state's fidelity, crucial for assessing approximation quality when the exact solution is unknown.

4 Results & Discussion

4.1 Solving Linear Max-Cut with MPS: Initial Setup

Problem: Linear Max-Cut (nearest-neighbor graph). *Method:* Classical MPS simulation with exact gate tensors. *Problem Mapping:* Max-Cut $\rightarrow$ QUBO ($C(\mathbf{x}) = \sum(2x_ix_j - x_i - x_j)) \rightarrow$ Ising Hamiltonian ($H_C = \sum h_iZ_i + \sum J_{ij}Z_iZ_j$). *Initial Purpose:* Confirmed MPS can solve this problem. *Main Focus (next slides):* Investigate bond dimension growth and error accumulation for various simulation approaches.

4.2 Entanglement Growth: Exact Simulation

Observation: In exact (no compression) simulation, maximum bond dimension (χ) shows **exponential growth** with both Number of Nodes (N) and QAOA Layers (p). This confirms the classical simulation bottleneck.

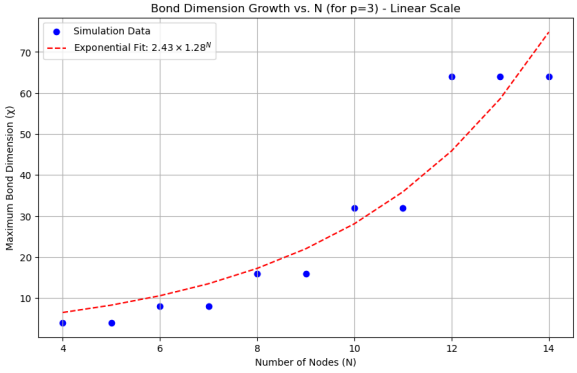

Figure 5: Max Bond Dimension vs. N (for p=3).

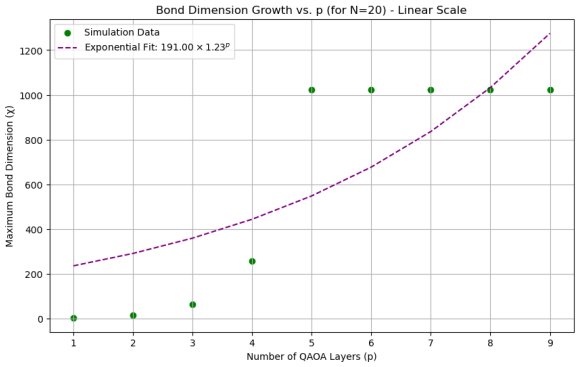

Figure 6: Max Bond Dimension vs. p (for N=15).

4.3 Entanglement Growth: Exact vs. Compressed

Aim: Cap exponential χ growth. *Method:* Fixed $\chi_{\max} = 4$ compression (SVD per layer, Variational per layer). *Observation:* Compression effectively **caps bond dimension** at $\chi_{\max} = 4$ regardless of N or p. *Implication:* Enables **polynomial scaling**, making simulation cost independent of N or p.

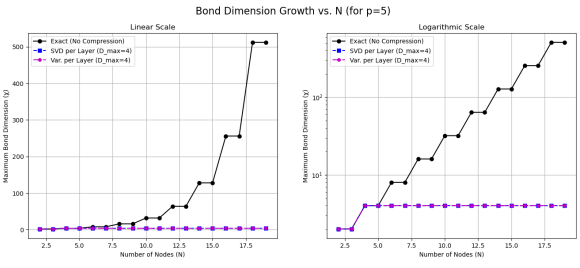

Figure 7: Max Bond Dimension vs. N (p=5).

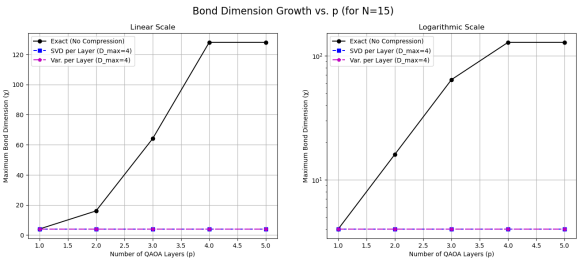

Figure 8: Max Bond Dimension vs. p (N=15).

4.4 Infidelity Analysis: Exact Reference

Goal: Quantify infidelity using Ψ_{exact} (for small N/p). *Strategies compared ($\chi_{max} = 4$):* SVD per Layer, Variational per Layer, SVD at End, Variational at End. *Observation:* Infidelity $\uparrow N$ and p. "Per Layer" $\geq$ "at End" methods. Variational $>$ SVD.

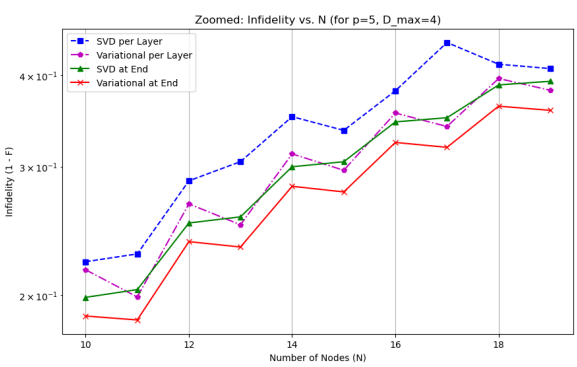

Figure 9: Infidelity vs. N (p=5).

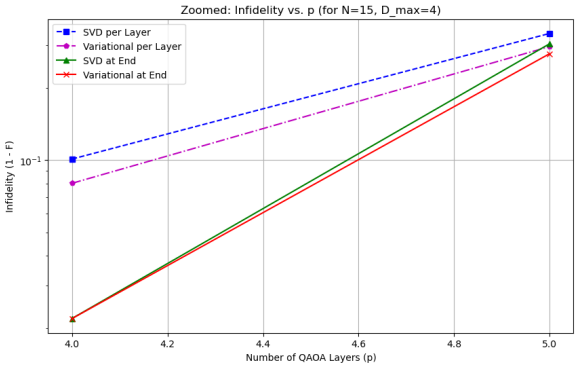

Figure 10: Infidelity vs. p (N=15).

4.5 Validating Inexact Fidelity Estimation

Challenge: Ψ_{exact} unreachable for large systems. *Solution:* Multiplicative Fidelity Law to estimate infidelity. *Validation:* Estimated infidelity (Multiplicative Fidelity Law) closely follows true infidelity (vs. Exact State). *Implication:* Reliable accuracy assessment when exact results are unknown.

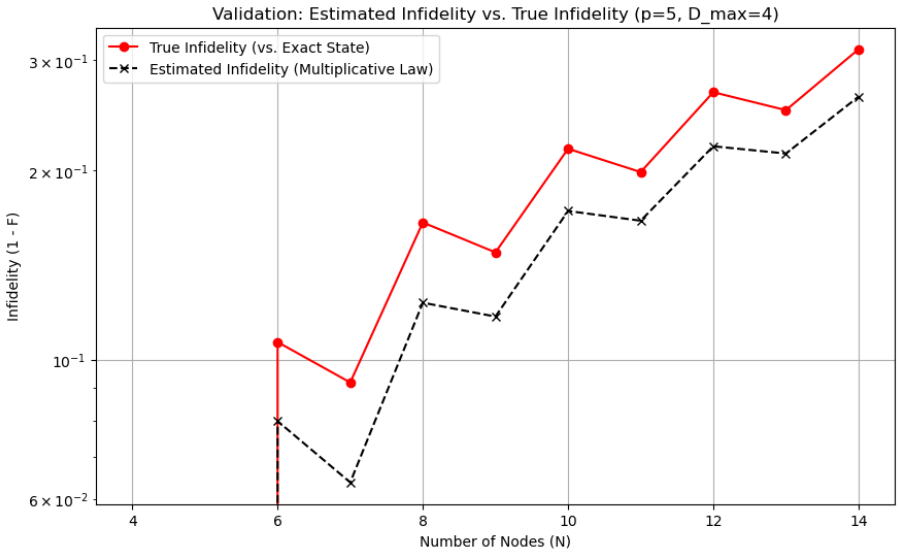

Figure 11: Validation: Estimated Infidelity vs. True Infidelity (p=5, D_max=4).

4.6 Performance of Inexact Simulations

Method: Evaluate performance using estimated infidelity ($\chi_{\max} = 4$). *Observation:* Estimated infidelity shows **saturating trend** as N or p increase. *Implication:* Predictable error profile for scalable quantum circuit analysis.

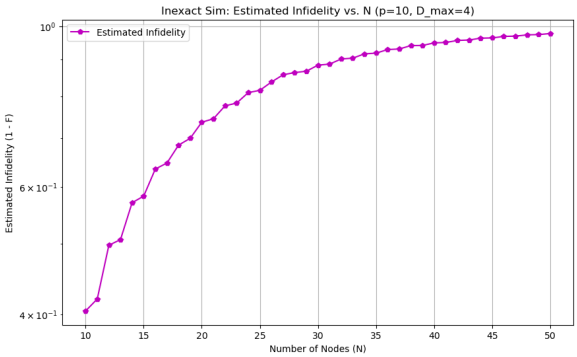

Figure 12: Estimated Infidelity vs. N (p=10).

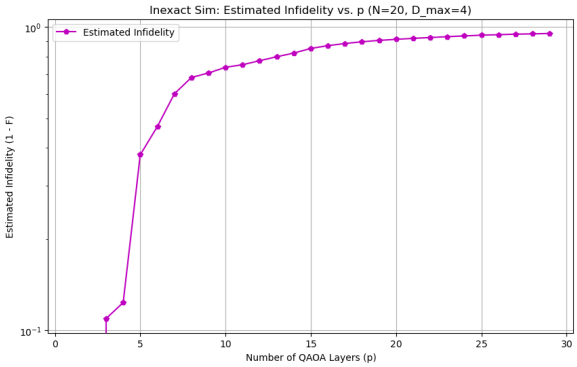

Figure 13: Estimated Infidelity vs. p (N=20).

5 Conclusion & Future Outlook

5.1 Summary of Contributions

Theoretical Understanding: Gained comprehensive understanding of MPS, tensor decompositions, QAOA, and problem mapping. *Python Framework:* Developed a robust classical quantum circuit simulation framework using MPS, implementing core functionalities and exact/inexact compression. *QAOA Performance Analysis:* Established foundation for studying entanglement growth and evaluating approximation accuracy via fidelity estimation.

5.2 Key Principles Revisited

Entanglement: Resource and bottleneck. *Approximation:* Key for tractable solutions. *Hybrid Quantum-Classical:* Synergy enables practical algorithms. *Measurement:* Interface to quantum states. *Simulation as Benchmark:* Advanced classical simulations benchmark "quantum advantage."

5.3 Future Work

Explore to Tree Tensor Networks (TTNs): Explore TTNs for complex circuits, building on existing research. *Advanced QAOA Applications:* Simulate on more complex graphs; investigate mixer Hamiltonian impact. *Performance Optimization:* Optimize Python code (JIT, GPU); integrate with specialized TN libraries.

6 References

- Ayral, T., Louvet, T., Zhou, Y., Lambert, C., Stoudenmire, E. M., & Waintal, X. (2023). *Density-Matrix Renormalization Group Algorithm for Simulating Quantum Circuits with a Finite Fidelity*. PRX Quantum, 4(2), 020304.
- Dupont, M., Didier, N., Hodson, M. J., Moore, J. E., Reagor, M. J. (2022). *Entanglement perspective on the quantum approximate optimization algorithm*. Physical Review A, 106(2), 022423.
- Dubey, A., Zeybek, Z., Schmelcher, P. (2025). *Simulating Quantum Circuits with Tree Tensor Networks using Density-Matrix Renormalization Group Algorithm*. arXiv:2504.16718.
- Schollwöck, U. (2011). *The density-matrix renormalization group in the age of matrix product states*. Annals of Physics, 326(1), 96-192.
- Zhou, Y., Stoudenmire, E. M., & Waintal, X. (2020). *What Limits the Simulation of Quantum Computers?*. Physical Review X, 10(4), 041038.
- Feynman, R. P. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7), 467-488.
- Farhi, E., Goldstone, J., Gutmann, S. (2014). A Quantum Approximate Optimization Algorithm. *arXiv preprint arXiv:1411.4028*.
- Eckart, C., Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3), 211-218.

General Resources:

- TensorNetwork.org: Comprehensive resource.
- Tensors.net: General introduction.

Signatures

Annwoy Roy Choudhury	Prof. Dr. Peter Schmelcher
BS-MS Student, IISER Mohali	Supervisor, Universität Hamburg
Date:	Date: