Enhanced Digitized Adiabatic Quantum Factorization Algorithm Using Null-Space Encoding

Universidad Internacional Menéndez Pelayo

ENHANCED DIGITIZED ADIABATIC QUANTUM FACTORIZATION ALGORITHM USING NULL-SPACE ENCODING

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof. dr. Jeroen J.G. Geurts,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de Faculteit der Bètawetenschappen
op [defenseDay] [DATE] om [TIME xx.xx] uur
in [de aula / het auditorium] van de universiteit,
De Boelelaan 1105

by

Felip Pellicer Benedicto

geboren te [Place, Country (if not NL) of Birth]

This dissertation was approved by

Promoter:

Prof. dr. promoter 1	Vrije Universiteit Amsterdam, The Netherlands

Copromoter:

Prof. dr. promoter 2	Vrije Universiteit Amsterdam, The Netherlands

Dissertation Committee

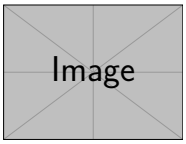
Chairman:

Rector Magnificus, Prof. dr. Jeroen J.G. Geurts	Vrije Universiteit Amsterdam, The Netherlands

Committee:

Prof. dr. committee member 1	Vrije Universiteit Amsterdam, The Netherlands
Prof. dr. committee member 2	Affiliation, City, Country
Prof. dr. committee member 3	Affiliation, City, Country
Prof. dr. committee member 4	Affiliation, City, Country
Prof. dr. committee member 5	Affiliation, City, Country

VU VRIJE UNIVERSITEIT AMSTERDAM



NWO

Acknowledgments

Without a doubt, the acknowledgments are the most widely and most eagerly read part of any thesis.

Laurens

Amsterdam, January 2021

Dedicatory of this thesis...

Author

Contents

Abstract

Your abstract.

1

INTRODUCTION

Quantum computing has emerged as the most promising paradigm for solving problems that are classically intractable. One of the most celebrated achievements is the discovery of the Shor's algorithm, which demonstrated that the integer factorization problem can be solved exponentially faster on a quantum computer than with the best-known classical algorithm. However, the full realization of this algorithm is currently out of reach due to quantum hardware limitations, known as the Noisy Intermediate-Scale Quantum (NISQ) era.

The constraints imposed by the NISQ era have motivated alternative methods to attempt to solve the same problems by using fewer quantum resources. One such approach is the Adiabatic Quantum Computation (AQC), which takes advantage of the robustness of adiabatic evolution to bring the system from a known state to a final state that encodes the solution of a problem. Another approach is the Variational Quantum Algorithm (VQA), which is a hybrid algorithm that uses quantum hardware for state evolution, and classical routines for optimization purposes.

In this work, we propose and analyze a version of the Digitized Adiabatic Quantum Factorization (DAQF) algorithm that incorporates a variation in the cost Hamiltonian. By this, we aim to provide a more resource-efficient and scalable method for integer factorization.

1.1 CIRCUIT MODEL OF QUANTUM COMPUTATION

The circuit or gate-based model of quantum computation is the most widely studied framework and the foundation of many quantum algorithms, including Shor's and Grover's algorithms. In this model, computation proceeds through the application of a sequence of unitary gates, which evolve the state of the qubits in a discrete, step-wise fashion.

Mathematically, a quantum circuit implements a unitary transformation U on the initial quantum state, typically chosen as $|0\rangle^{\otimes n}$. This transformation is decomposed into a series of quantum gates from a universal set of gates. Measurement in the computational basis is performed at the end of the circuit to extract classical information from the quantum circuit.

Universality is a crucial property of this model, as it ensures that any unitary transformation –and therefore any quantum algorithm– can be implemented using only a finite set of gates. This makes the circuit model a general-purpose framework, capable of simulating any other model of quantum computation.

Gate-based quantum computation aligns well with digital control paradigms, and most existing quantum hardware platforms, such as superconducting qubits and trapped ions, are designed to implement this model. However, the depth and width of circuits that can be reliably executed on near-term devices are limited by noise, decoherence, and imperfect gate fidelities.

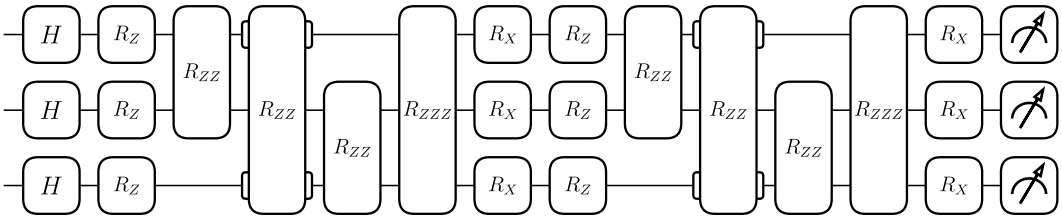


Figure 1.1: Example of a quantum circuit, part of a QAOA protocol that implements integer factorization for a small number.

1.2 Adiabatic Quantum Computation

ALAN: Talk here about adiabatic theorem and evolutions. Then, introduce the concept/definition of adiabatic quantum computation, where we define the problem Hamiltonians and etc. To this end, we can get results by Sarandy [https://doi.org/10.1007/s11128-004-7712-7] and references therein, and some discussion done in [Rev. Mod. Phys. 90, 015002 (2018)] to make our life easier.

The adiabatic theorem of quantum mechanics provides the foundation for an alternative model of quantum computation based on continuous-time evolution. Consider a time-dependent Hamiltonian $H(t)$ with a discrete and non-degenerate spectrum. Thus we can define its instantaneous eigenstates and eigenenergies by

$$H(t)\,|n(t)\rangle = E_n(t)\,|n(t)\rangle\,. \tag{1.1}$$

The adiabatic theorem states that a quantum system initially prepared in an eigenstate $|n(0)\rangle$ of $H(0)$, will remain in its instantaneous eigenstate $|n(t)\rangle$ throughout the evolution, provided that the spectrum remains gapped and the change is sufficiently slow [1, 2]. This last condition is generally formulated as follows:

$$\max_{0\le t\le T}\left|\frac{\langle k|\,\dot{H}\,|n\rangle}{g_{nk}}\right| \ll \min_{0\le t\le T}|g_{nk}|, \tag{1.2}$$

where T is the total evolution time and g_{nk} represents the energy gap between levels n and k:

$$g_{nk}(t) \equiv E_n(t) - E_k(t) \tag{1.3}$$

Despite the adiabatic theorem works for any eigenstate, in practice it is customary to focus on ground state adiabatic passages. This is because ground states are typically more robust against decoherence and thermal excitations, as they are energetically isolated from higher-energy levels. Moreover, for many computational problems, particularly those related to optimization problems, it is possible to construct a problem Hamiltonian H_P whose ground state encodes the solution to the instance under consideration.

In the adiabatic quantum computation model, the Hamiltonian is interpolated between an initial Hamiltonian H_0, with a known and easily preparable ground state, and the problem Hamiltonian H_P, according to a schedule [2]:

$$H(t) = \big[1 - s(t)\big]H_0 + s(t)H_P, \quad s(t) \in [0,1], \tag{1.4}$$

where $s(t)$ is a smooth, monotonic function such that $s(0) = 0$ and $s(T) = 1$. The problem Hamiltonian H_P is designed so that its ground state corresponds to the solution of the computational task.

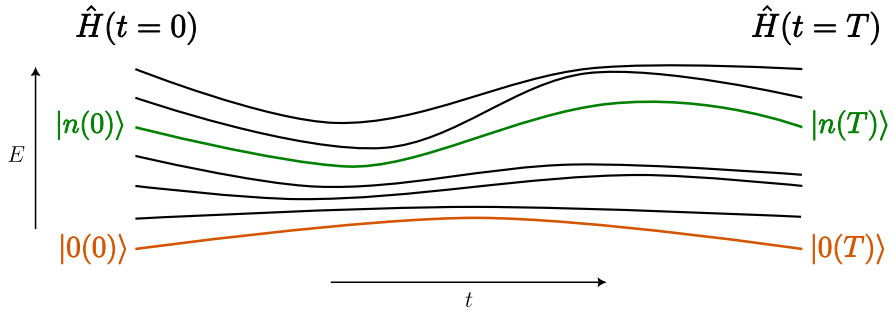


Figure 1.2: Schematic of an adiabatic passage. The orange line represents the Hamiltonian's ground state, while the green a Hamiltonian's eigenstate different from the ground state.

(Maybe talk about AQC can be efficiently simulated in the circuit model).

1.2.1 Adiabatic Quantum Annealers

ALAN: Here we discuss about the adiabatic quantum annealers, where the problem Hamiltonian is the Ising Hamiltonian (which is the focus of our applications). To this end, maybe we can use the review paper to get some key discussions [Rev. Mod. Phys. 90, 015002 (2018)].

An important and widely studied application of adiabatic evolution in quantum devices is Adiabatic Quantum Annealing (AQA), where the goal is to find low-energy configurations of a classical cost function mapped onto a quantum Hamiltonian. AQA is most often applied to Quadratic Unconstrained Binary Optimization (QUBO) problems, which can be exactly reformulated as a 2-body interaction Ising Hamiltonian of the form

$$H_{\mathrm{P}} = \sum_i h_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \,, \tag{1.5}$$

where h_i are local fields, J_{ij} represent coupling strengths between qubits, and σ_i^z are Pauli-Z operators acting on the i-th qubit. The task is to drive the system from an initial, easily preparable ground state towards the ground state of H_{P}, which encodes the optimal solution to the problem at hand.

In a typical AQA protocol, the evolution begins with an initial transverse-field Hamiltonian

$$H_0 = \sum_i \sigma_i^x \,. \tag{1.6}$$

whose ground state is straightforward to prepare. The system is then evolved according to the interpolation scheme in equation (1.4). The aim is to adiabatically steer the system from the easily preparable ground state H_0 to the ground state of H_{P}, which encodes the solution to the problem of interest.

This procedure is straightforward for problems that can be expressed in the QUBO form, as they can be mapped to an Ising Hamiltonian with only two-body interactions. However, the factorization problem presented in chapter 2 does not directly fall into the QUBO class, as its Hamiltonian contains three- and four-body interaction terms. As will be discussed later, this limitation can be addressed by adopting an alternative formulation of the problem Hamiltonian that avoids these high-order terms while preserving the encoded solution.

1

1.3 QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical algorithm introduced by Farhi et al. [3] to tackle combinatorial optimization problems. Its goal is to approximate the ground state of a cost Hamiltonian whose minimum encodes the optimal solution to the problem. QAOA is particularly suited to near-term quantum devices due to its shallow circuit depth and iterative variational structure, which offloads part of the computational workload to a classical optimizer.

At its core, QAOA constructs a parametrized quantum state (ansatz) by sequentially applying two alternating types of unitaries derived from two Hamiltonians:

- The cost Hamiltonian H_C, which encodes the objective function of the optimization problem. This is typically written in the form of an Ising Hamiltonian.
- The mixing Hamiltonian H_M, which introduces transitions between computational basis states to enable exploration of the solution space. A common choice is:

$$H_M = \sum_i \sigma_i^x, \tag{1.7}$$

  where σ_i^x is the Pauli-X operator acting on qubit i.

The algorithm starts with the initial state $|\psi_0\rangle = |+\rangle^{\otimes n}$, which is the ground state of H_M and a uniform superposition over all computational basis states. The QAOA ansatz with p layers is constructed as:

$$|\psi_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle = e^{-i\beta_p H_M} e^{-i\gamma_p H_C} \cdots e^{-i\beta_1 H_M} e^{-i\gamma_1 H_C} |+\rangle^{\otimes n}, \tag{1.8}$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ are real variational parameters to be optimized.

The performance of a QAOA instance is assessed by evaluating the expected value of the cost Hamiltonian in the ansatz state:

$$F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle \psi_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) | H_C | \psi_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) \rangle . \tag{1.9}$$

This expectation value serves as the cost function for the classical optimizer. The parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ are iteratively updated to minimize F_p, with quantum circuits being re-evaluated at each step until convergence.

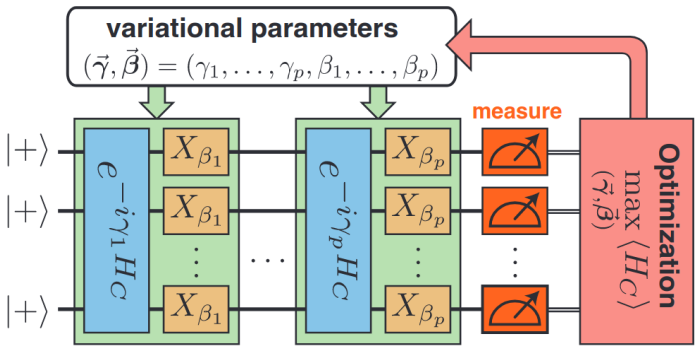


Figure 1.3: Diagram of a p-layer QAOA circuit. Starting from the initial state $|+\rangle^{\otimes n}$, the circuit alternates between applying the unitaries $e^{-i\gamma_i H_C}$ and $e^{-i\beta_i H_M}$ for $i = 1$ to p. The final state is measured to estimate the expectation value $\langle H_C \rangle$, which is passed to a classical optimizer. This process is repeated until convergence.

Source: Adapted from Zhou et al., 2019.

The QAOA stands out as one of the most compelling candidates for demonstrating quantum advantage on near-term quantum devices. From a complexity-theoretic standpoint, Farhi et al. [4] argue that sampling from the output distribution of even the lowest-depth version of QAOA (i.e., $p = 1$) could be classically intractable. Specifically, they show that there exist problem instances and choices of parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ for which classical algorithms cannot feasibly reproduce the output of a quantum device running QAOA, strengthening the case for using it as a path to quantum supremacy.

Beyond theoretical hardness arguments, QAOA also exhibits practical advantages in terms of implementability and flexibility. Ho and Hsieh [5] introduce a closely related variational framework (VQCS), which shares the QAOA structure of alternating unitaries generated by simple Hamiltonians. They demonstrate that such protocols can efficiently prepare complex, non-trivial quantum states and are compatible with current quantum hardware platforms such as trapped ions and superconducting qubits. The minimal requirement of time evolution under simple, local Hamiltonians makes QAOA especially attractive for near-term experimental implementation.

Additionally, the adaptability of QAOA makes it suitable for a wide variety of problem domains, including factoring. In their work on Variational Quantum Factoring (VQF), Anschuetz et al. [6] apply a QAOA-like approach to integer factorization, showing that hybrid quantum-classical heuristics can offer a path toward solving classically hard problems on noisy intermediate-scale quantum (NISQ) devices. While challenges remain particularly related to scalability and robustness against noise the framework's modularity and reliance on classical feedback make it well-suited for real-world NISQ conditions.

2

Factorization Algorithms

In this chapter, we introduce the factorization algorithms that are most relevant to the work presented in this thesis. The integer factorization problem consists of decomposing a given semiprime number N into its prime constituents. While seemingly simple, no known classical algorithm can factor large integers efficiently, and the best classical methods –such as the general number field sieve– scale superpolynomially with input size [7]. This computational asymmetry forms the foundation of widely used cryptographic protocols, most notably RSA, which relies on the practical difficulty of factoring large semiprimes to ensure security.

The emergence of quantum algorithms has dramatically shifted the landscape of computational complexity associated with factorization. In particular, Shor's algorithm demonstrated that quantum computers can solve the problem in polynomial time, meaning a direct threat to RSA-based cryptography. However, implementing Shor's algorithm requires fault-tolerant quantum, hardware which remains out of reach in the current NISQ era.

This chapter explores two quantum approaches to the factorization problem. First, we review Shor's algorithm and its quantum Fourier transform-based structure. Then, we introduce alternative strategies based on adiabatic quantum computation, which may offer more viable paths towards factorization on near-term quantum devices.

2.1 Shor's Factorization Algorithm

ALAN: First we have to briefly introduce the Shor's algorithm. We do not need a very detailed explanation, but it would be nice to provide a didactic discussion about the algorithm and its relevance. To this end, we can use the book by Nielsen and Chuang. In this section, we make clear that it is a gate-based way to do factorization.

In computer science, an algorithm is considered to be efficient when the number of steps of the algorithm grows as a polynomial in the input size. For the problem of integer factorization, the input is a semiprime number N, and the input size is measured as the number of bits required to represent it, i.e., $\log N$. The best known classical algorithm for factoring scales superpolynomially with input size, making the problem computationally hard for large N.

$$\mathcal{O}\left(\exp\left(c(\log N)^{1/3}(\log\log N)^{2/3}\right)\right). \tag{2.1}$$

In 1994, Peter Shor introduced a quantum algorithm that factors integers in polynomial time, marking a landmark result in quantum computing. Shor's algorithm runs in time

$$\mathcal{O}\left((\log N)^3\right), \tag{2.2}$$

dramatically outperforming the best classical method [8]. The algorithm relies on the quantum circuit model, making it a gate-based approach to factorization and one of the strongest motivations for the development of quantum computers.

The core idea of Shor's algorithm is to reduce factorization to the problem of order finding: given a number a coprime to N, find the smallest integer, r such that

$$a^r = 1 \mod N. \tag{2.3}$$

Once the order r is known, under certain conditions, one can recover a nontrivial factor of N using elementary number-theoretic arguments [8].

The quantum part of the algorithm is used to efficiently find this order r using period-finding techniques. This is achieved by preparing a quantum superpositioin and applying the quantum Fourier transform (QFT) to extract information about the period of the function $f(x) = a^x \mod N$. The classical post-processing step then uses continued fractions to extract r and attempt to derive the prime factors of N.

(FELIP: Add some schematic about the algorithm)

Despite its theoretical significance, implementing Shor's algorithm at scale requires a large number of qubits, long coherence times, and fault-tolerant error correction, which remain beyond the reach of current quantum hardware. As such, while Shor's algorithm remains the most efficient known method for factoring in the long-term quantum regime, alternative approaches –such as adiabatic and variational algorithms– are being explored for use in the NISQ era.

2.2 Adiabatic Factorization Algorithm

The problem of integer factorization can be formulated as a constrained search over pairs of natural numbers p and q such that

$$N = p \times q. \tag{2.4}$$

In the context of adiabatic quantum computation we aim to encode this constraint into the ground state of a problem Hamiltonian, allowing the solution to emerge through adiabatic evolution.

To encode candidate solutions p and q, we adopt a binary representation of natural numbers. Any natural number N can be expressed as:

$$N = \sum_{j=0}^{n_{\text{bits}}-1} 2^j x_j, \tag{2.5}$$

where each $x_j \in \{0,1\}$ and the bit string $\mathbf{x} = x_{n_{\text{bits}}-1} \ldots x_0$ represents the binary encoding of N. In our approach, we exploit the fact that the factors p and q of an odd composite number can be rewritten as:

$$\begin{cases} p = 2p' + 1 \\ q = 2q' + 1 \end{cases}. \tag{2.6}$$

It can be proved that the n_p and n_q are an upper bound on the number of qubits required to represent p'

and q', respectively:

$$\begin{cases} n_p = m\left(\lfloor\sqrt{N}\rfloor_o\right) - 1 \\ n_q = m\left(\left\lfloor\frac{N}{3}\right\rfloor\right) - 1 \end{cases}, \tag{2.7}$$

where $\lfloor a\rfloor\left(\lfloor a\rfloor_o\right)$ denotes the largest (odd) integer not larger than a, while $m(b)$ denotes the smallest number of bits required for representing b [9]. Then, the adiabatic factorization algorithm will make use of $n = n_p + n_q$ qubits. The full quantum state

$$|\Psi\rangle = |\Psi_{p'}\rangle \otimes |\Psi_{q'}\rangle, \tag{2.8}$$

where $|\Psi_{p'}\rangle = |\psi_1\rangle \otimes \cdots \otimes |\psi_{n_p}\rangle$ and $|\Psi_{q'}\rangle = |\psi_{n_p+1}\rangle \otimes \cdots \otimes |\psi_{n_p+n_q}\rangle$.

To encode the factorization constraint into the adiabatic quantum framework, we follow the approach of Ref. [10] and define the objective function:

$$f(p,q) = \left(N - p \times q\right)^2, \tag{2.9}$$

such that its global minimum $f(p,q) = 0$ corresponds to valid factors. Translating this into a Hamiltonian acting on the computational basis, we obtain the quadratic problem Hamiltonian:

$$\hat{H}_{\mathrm{QP}} = \left[N\mathbb{1} - \left(\sum_{\ell=1}^{n_p} 2^\ell \hat{x}_\ell + \mathbb{1}\right)\left(\sum_{m=1}^{n_q} 2^m \hat{y}_m + \mathbb{1}\right)\right]^2, \tag{2.10}$$

where $\hat{x}_\ell = \dfrac{\mathbb{1} - \hat{\sigma}_\ell^z}{2}$ and $\hat{y}_m = \dfrac{\mathbb{1} - \hat{\sigma}_m^z}{2}$ are the number operators acting on the qubits encoding p' and q', respectively. The solution to the factorization problem is encoded in the ground state of $\hat{H}_{\mathrm{QP}}$.

The initial state of the system is prepared as:

$$|\psi(0)\rangle = |+\rangle^{\otimes n}, \tag{2.11}$$

where $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ is the eigenstate of $\hat{\sigma}^x$, corresponding to the ground state of the initial Hamiltonian $\hat{H}_0$ defined in Eq. 1.6. Following the adiabatic theorem, if the evolution from $\hat{H}_0$ to $\hat{H}_{\mathrm{QP}}$ is slow enough, the system will remain in the instantaneous ground state, reaching the ground state of $\hat{H}_{\mathrm{QP}}$ at the end of the protocol. This final state encodes the solution to the factorization problem.

Despite its conceptual clarity, the Hamiltonian $\hat{H}_{\mathrm{QP}}$ includes high-order multiqubit interactions, such as three- and four-body terms of the form $\hat{\sigma}_\ell^z \hat{\sigma}_m^z \hat{\sigma}_k^z$ and $\hat{\sigma}_\ell^z \hat{\sigma}_m^z \hat{\sigma}_k^z \hat{\sigma}_n^z$. These many-body interactions are difficult to implement, since they require one to bring all involved qubits together and make them interact in a controlled way, resulting in prone-to-error processes. The need to avoid those terms is of pivotal interest to provide efficient quantum algorithms.

(FELIP: maybe put the results that these research groups obtained when applying AQC for factorization)

2.2.1 Digitized Adiabatic Quantum Factorization

Although the adiabatic model is often formulated in terms of continuous time evolution, it can be simulated efficiently using gate-based quantum computation (ADD REFERENCES) through a process known as digitization.

In the digitized approach, the continuous adiabatic evolution governed by a time-dependent Hamiltonian as of Eq. 1.4 is approximated by a sequence of quantum gates through trotterization, breaking the total evolution into small time slices. Each slice is implemented as a layer in a quantum circuit, simulating the adiabatic trajectory step by step.

This method was successfully demonstrated in Ref. [10], where the authors implemented a digitized adiabatic factorization algorithm on superconducting hardware.

2.2.2 QAOA Applied to Factorization (Standard Protocol)

The Quantum Approximate Optimization Algorithm can be viewed as a shortcut to adiabaticity. Rather than performing a slow, continuous evolution, QAOA uses a fixed-depth quantum circuit composed of alternating unitaries derived from the mixing and problem Hamiltonians. The parameters of these unitaries are optimized variationally to prepare a state that approximates the solution.

As shown in Ref. [11], QAOA and quantum annealing share universal structural properties, and there is compelling evidence that smooth annealing paths can be constructed from QAOA optimal parameters. This supports the interpretation of QAOA as a digitized and variationally optimized version of an adiabatic process.

In summary, the ingredients to build and execute the QAOA algorithm applied to integer factorization –or what we will call the "standard protocol"– are:

- A Hamiltonian $\hat{H}_{\mathrm{QP}}$ (Eq. 2.10) that encodes the solution to the factorization problem in its ground state.
- A mixing Hamiltonian $\hat{H}_{\mathrm{M}}$ that does not commute with $\hat{H}_{\mathrm{QP}}$, e.g., Eq. 1.7.
- A quantum circuit that implements the state evolution given by Eq. 1.8.
- A classical optimizer to find the circuit's optimal parameters.
- A cost function given by $F_p(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})|\,\hat{H}_{\mathrm{QP}}\,|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle$.

Due to the form of Eq. 2.10, the problem Hamiltonian can be expressed as:

$$\hat{H}_{\mathrm{QP}} = n\mathbb{1} + \sum_i a_i\hat{\sigma}_i^z + \sum_{ij} b_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z + \sum_{ijk} c_{ijk}\hat{\sigma}_i^z\hat{\sigma}_j^z\hat{\sigma}_k^z + \sum_{ijk\ell} d_{ijk\ell}\hat{\sigma}_i^z\hat{\sigma}_j^z\hat{\sigma}_k^z\hat{\sigma}_\ell^z \tag{2.12}$$

Then, the part of $e^{-i\gamma\hat{H}_{\mathrm{QP}}}$ corresponding to the term $a_i\hat{\sigma}_i^z$ is

$$e^{-i\gamma a_i\hat{\sigma}_i^z} = e^{-i\gamma a_i}\,|0\rangle\langle 0| + e^{i\gamma a_i}\,|1\rangle\langle 1| = R_Z(2\gamma a_i)\,, \tag{2.13}$$

the part corresponding to $b_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z$ is

$$\begin{aligned} & e^{-i\gamma b_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z} \\ & \quad = e^{-i\gamma b_{ij}}|00\rangle\langle 00| + e^{i\gamma b_{ij}}|10\rangle\langle 10| + e^{i\gamma b_{ij}}|01\rangle\langle 01| + e^{-i\gamma b_{ij}}|11\rangle\langle 11| \\ & \quad = R_{ZZ}(2\gamma b_{ij}), \end{aligned} \tag{2.14}$$

and so on for the higher-order terms. Similarly, the mixing Hamiltonian $\hat{H}_{\mathrm{M}}$ gives rise to individual X-rotations at the end of each QAOA layer. With all this, one constructs the QAOA circuit, as shown in Fig. 2.1.

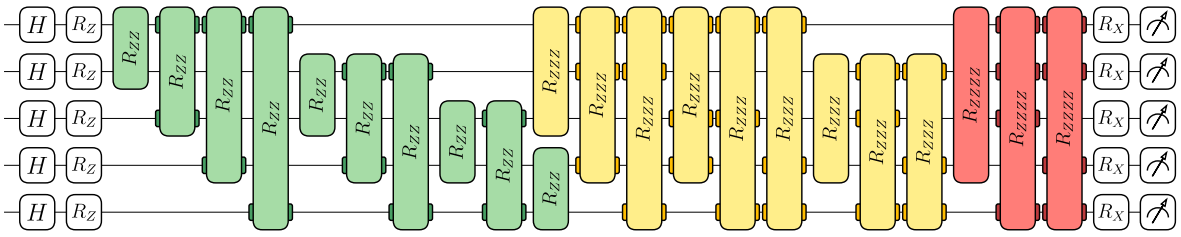


Figure 2.1: One-layer circuit for factorizing the number $N = 35$ using the standard QAOA protocol. Notice the presence of three- and four-qubit gates, highlighted in yellow and red, respectively. Rotation angles are omitted for simplicity.

(FELIP: I found these few articles on QAOA factorization. The problem is that they use simplification techniques and ad-hoc analysis to run factorization using much less qubits. I was not able to find articles that use the standard basic approach.)

In a study that makes use of a similar approach, plus further pre-processing and simplifications, Anschuetz et al. factorize numbers up to 291311. Using additional classical pre-processing heuristics, Karamlou et al. were able to factorize 1099551473989, 3127, and 6557 using only 3, 4, and 5 qubits, respectively. However, our aim is not to compete with those approaches, since we focus on the general standard approach for factorization, without making use of ad-hoc analysis.

2.2.3 OUR PROPOSAL (LINEARIZED PROTOCOL)

As mentioned in the introduction to Section 2.2, the adiabatic factorization algorithm needs to deal with the difficulty of implementing three- and four-body interaction terms. In the digitized version of this problem, like QAOA, this difficulty is transformed into three- and four-qubit gates (Fig. 2.1), which need to be decomposed into multiple two-qubit gates (Fig. 2.2), leading to lower fidelities at the end of the quantum circuit.

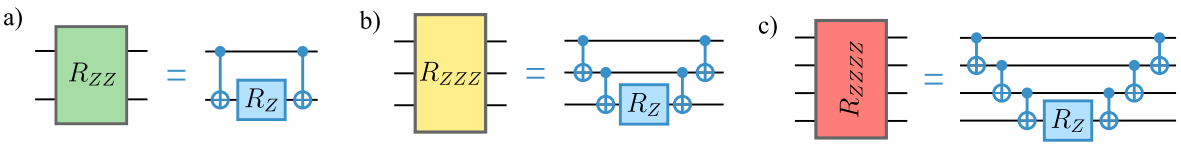


Figure 2.2: Decomposition of (a) two-, (b) three-, and (c) four-qubit Z-rotation gates in CNOTs and single-qubit Z-rotations.

To mitigate this issue, we propose a linearized problem Hamiltonian inspired by the same factorization condition, defined as:

$$\hat{H}_{\mathrm{LP}} = N\mathbb{1} - \left(\sum_{\ell=1}^{n_p} 2^\ell \hat{x}_\ell + \mathbb{1}\right)\left(\sum_{m=1}^{n_q} 2^m \hat{y}_m + \mathbb{1}\right). \tag{2.15}$$

Unlike the original Hamiltonian $\hat{H}_{\mathrm{QP}}$, whose ground state encodes the solution, $\hat{H}_{\mathrm{LP}}$ contains only two-body terms and is therefore easier to implement, but the factorization solution corresponds to an eigenstate with eigenvalue zero rather than the ground state. Since the adiabatic theorem is not restricted to ground states but applies to any non-degenerate eigenstate, we hypothesize that it is possible to target this eigenstate through a suitable adiabatic or variational process. Based on this idea, our proposal is to replace the original Hamiltonian in the QAOA layers with $\hat{H}_{\mathrm{LP}}$, leveraging the possibility of starting from an eigenstate near the middle of the initial Hamiltonian's spectrum and evolving under $\hat{H}_{\mathrm{LP}}$ to preserve this eigenstate structure, ultimately reaching the correct factorization solution, which also lies near the center of the spectrum.

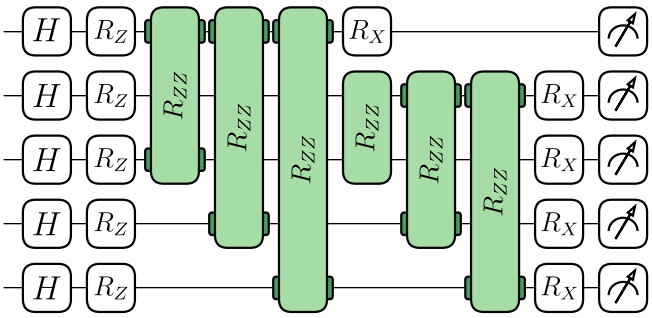


Figure 2.3: One-layer circuit for factorizing the number $N = 35$ using our protocol, evolving the state with the linear Hamiltonian H_{LP}. Notice the simplification with respect to Fig. 2.1 due to the absence of three- and four-qubit gates.

(I am not sure if adding this paragraph or not, since at the end the cost function is just a classical computation and maybe this explanation adds confusion.) There is another difference with respect to the standard QAOA protocol. In the latter, the same Hamiltonian is typically used for both quantum state evolution and classical cost evaluation. In contrast, our proposal explicitly decouples these two roles: we use the linearized Hamiltonian to drive the quantum dynamics (problem Hamiltonian), and a separate cost Hamiltonian, whose ground state encodes the correct factors to steer the classical optimizer.

3

METHODOLOGY

The Quantum Approximate Optimization Algorithm has been extensively investigated since its introduction, leading to a broad landscape of theoretical analyses, practical implementations, and performance-enhancement techniques. Numerous variants have been proposed, ranging from modified ansatze to parameter initialization heuristics, and adaptive layer-scaling procedures. This diversity reflects both the flexibility of the QAOA framework and the complexity of identifying implementations that perform well accross different problem instances.

Given this wide range of possible approaches, it is essential to precisely specify the algorithmic configuration adopted in this work. In the following sections, we describe the chosen setup, selected to ensure that the comparison between our proposed protocol and the standard QAOA implementation is both meaningful and fair. The configuration is based on the best-performing practices for the standard protocol, so that any observed differences in performance can be attributed to the change in the problem Hamiltonian rather than to auxiliary implementation details.

Although several software frameworks for quantum circuit simulation exist, such as PennyLane and Qiskit, this work employs a direct state-vector simulation using explicit matrix multiplications implemented in Numpy. This approach avoids the overhead of gate-by-gate simulation, providing a more efficient and transparent means of evaluations QAOA circuits at the scale considered. For completeness and reproducibility, the full implementation is available in a public GitHub repository[1], where all details of the numerical procedures and data analysis can be examined.

3.1 INCREMENTAL APPROACH

In multi-layer QAOA, increasing the number of layers also increases the number of parameters to be optimized. Finding global optima is considerably simpler for $p = 1$ (i.e., two parameters) than for circuits with many layers. Moreover, when a new layer is added, the previously optimized solution remains valid within the enlarged parameter space: the parameters corresponding to the first p layers can be initialized with their previously optimized values, while the newly introduced parameters can be set to zero. Consequently, adding an additional layer ($p \to p+1$) can only maintain or improve performance relative to the previous iteration.

This observation motivates a progressive training strategy, often referred to as an *incremental* approach, which proceeds as follows:

1. Analyze the cost function landscape for $p = 1$ to select initial parameters γ_1 and β_1 close to the optimal region (Fig. 3.1).
2. Optimize QAOA with $p = 1$ to obtain the parameters $\widetilde{\gamma}_1$ and $\widetilde{\beta}_1$.

[1] `https://github.com/fpllcr/master-thesis`

3. Optimize QAOA with $p = 2$, initializing $\gamma_1 = \widetilde{\gamma}_1$ and $\beta_1 = \widetilde{\beta}_1$, and selecting suitable initial values for γ_2 and β_2 using an appropriate heuristic (Sec. 3.2). This yields the updated parameters $\widetilde{\gamma}_1, \widetilde{\gamma}_2, \widetilde{\beta}_1, \widetilde{\beta}_2$.
4. Repeat this procedure iteratively up to the desired depth p, each time initializing with the optimized parameters from the previous iteration and applying a strategy to set the new layer's parameters.

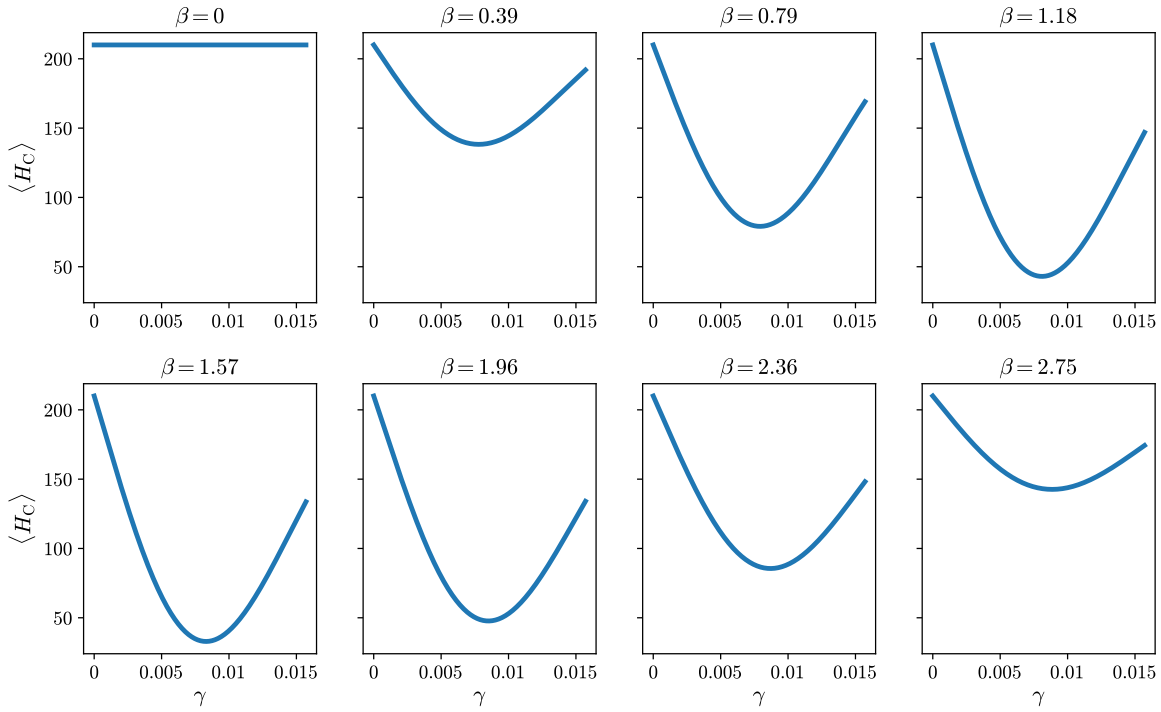


Figure 3.1: Cost function profile for the quadratic Hamiltonian when $N = 21$. After this analysis, $\gamma_0 = 0.0075$ and $\beta_0 = 1.57$ are used as initial parameters for $p = 1$. Maximum γ is determined as the value that causes angle folding for the maximum energy eigenvalue, i.e. $e^{-i\gamma_{\max} E_{\max}} = e^{-2\pi i}$.

3.2 PARAMETER INITIALIZATION

As discussed in Sec. 3.1, adding a new QAOA layer requires a suitable heuristic to initialize the additional variational parameters. Beyond random initialization, two specific strategies have been considered in this work:

- **Parameter interpolation.** This method, adopted by Zhou et al. [12], linearly interpolates the trajectory traced by the optimized parameters at depth p to generate an initial guess for depth $p+1$. Such interpolation is particularly effective for problems like MaxCut and QUBO, where the optimal parameters often evolve monotonically in an adiabatic-like fashion.
- **Alternative heuristic.** This approach, which achieved superior performance for both the standard and linearized protocols in our experiments, initializes the new parameters as

$$\begin{cases} \gamma_{p+1} \leftarrow \widetilde{\gamma}_p, \\ \beta_{p+1} \leftarrow 0. \end{cases} \tag{3.1}$$

Two factors motivated the adoption of this alternative strategy. First, the interpolation heuristic performs poorly for the linearized protocol, frequently leading to trapping in local minima. Second, integer factorization differs fundamentally from MaxCut and QUBO problems: there is no evidence that optimal parameters follow adiabatic-like trajectories, and our results confirm that they do not (see Fig. 3.2 and Fig. 3.3). The fact that the alternative heuristic also outperforms interpolation for the standard protocol ensures a fair basis for comparison, as both protocols are evaluated under the most favorable optimization setup for the reference (standard) implementation.

3.3 CLASSICAL OPTIMIZER

A wide variety of classical optimizers have been employed within QAOA, as reviewed in the literature [13]. In this work, we focused on optimizers available in the SciPy library and tested representative classes: bounded vs. unbounded and gradient-free vs. gradient-based.

Gradient-free methods such as Nelder–Mead showed promising performance for small problems but scale poorly with the number of parameters, and were therefore discarded. Cobyla exhibited poor convergence and frequent trapping in local minima, even for few-qubit cases, making it unsuitable for this study. Among gradient-based methods, the best performance was obtained with BFGS and its bounded variant L-BFGS-B:

- **L-BFGS-B.** This bounded optimizer is well-suited for QUBO and MaxCut problems, where parameter symmetries justify restricting γ and β to specific intervals. Under such constraints, L-BFGS-B often produces adiabatic-like parameter trajectories, with γ starting near zero and increasing smoothly, while β decreases monotonically toward zero.
- **BFGS.** As an unbounded optimizer, BFGS allows variational parameters to take any real value. In all tested cases up to 8 qubits, BFGS consistently outperformed L-BFGS-B, likely because its unbounded nature helps it escape local minima that trap the bounded optimizer (see Fig. 3.2).

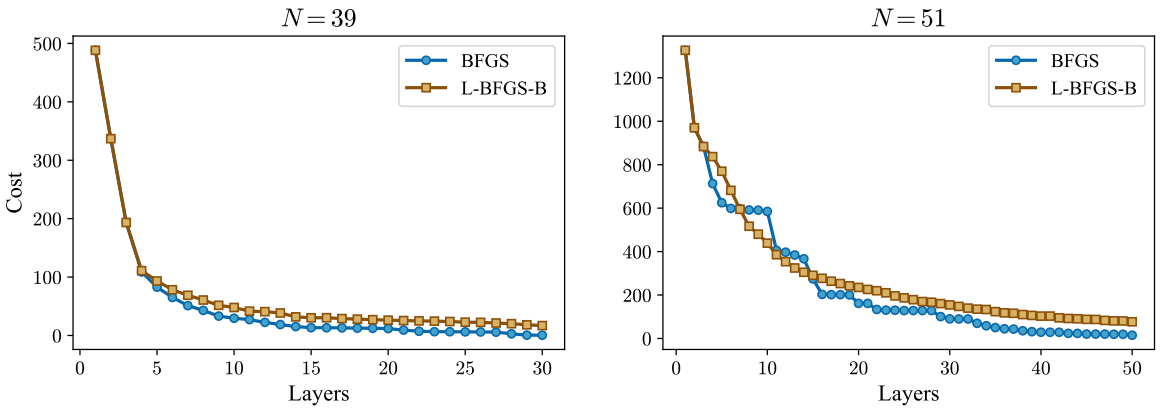


Figure 3.2: Cost evolution comparison between BFGS and L-BFGS-B for factorizing $N = 39$ and $N = 51$ using the standard protocol. The performance gap was observed consistently up to 8-qubit problems.

Based on these findings, **BFGS is selected as the classical optimizer for all experiments**. This choice complements the other elements of our QAOA setup, ensuring a consistent and high-performance framework for comparing the standard protocol and the proposed linearized protocol.

3.4 QAOA Setup Summary

The methodological choices adopted throughout this chapter can be summarized as follows:

- **Layer-by-layer training:** QAOA parameters are optimized incrementally, starting from $p = 1$ and using previously optimized parameters as initialization for deeper circuits.
- **Parameter initialization:** New layers are initialized using the heuristic $\gamma_{p+1} \leftarrow \widetilde{\gamma}_p$, $\beta_{p+1} \leftarrow 0$, which outperforms interpolation strategies.
- **Classical optimizer:** The gradient-based, unbounded BFGS method is employed, providing superior convergence compared to bounded alternatives.

Together, these decisions define a robust and fair QAOA implementation, enabling meaningful comparisons between different problem Hamiltonians and protocols.

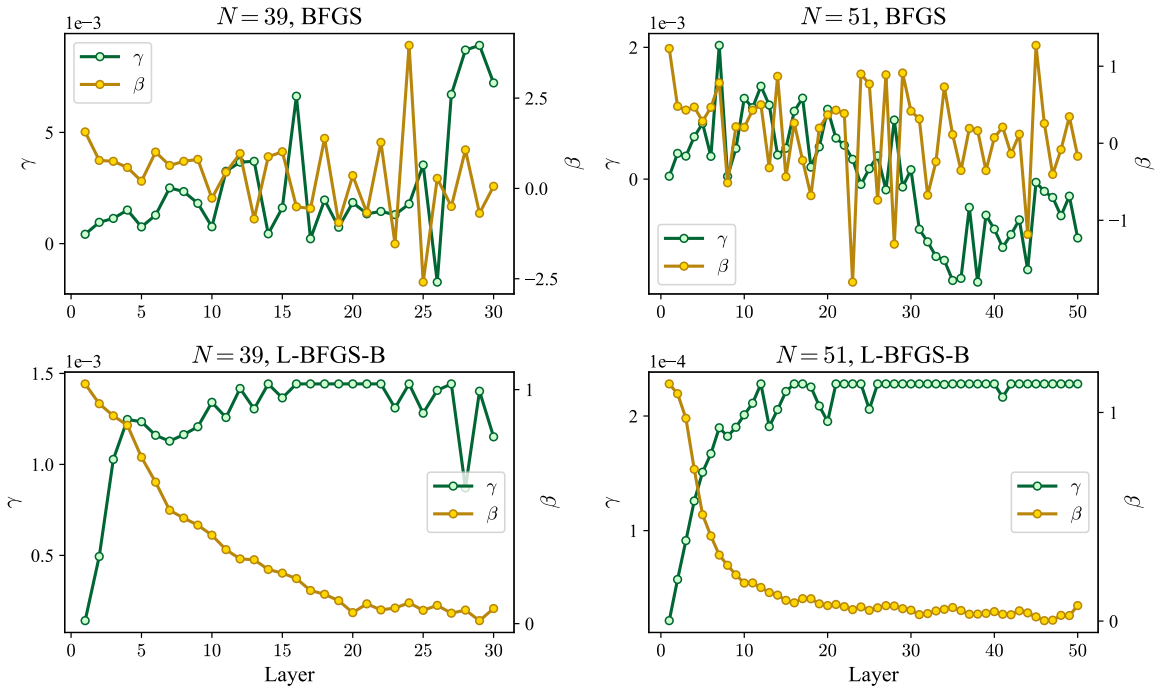


Figure 3.3: Variational parameter evolution for $N = 39$ and $N = 51$ using BFGS and L-BFGS-B. An adiabatic-like trajectory emerges under L-BFGS-B due to parameter bounds.

4

RESULTS (WORK IN PROGRESS)

This chapter we introduce our results!!! To this end, we already can state here the problem of the many-body terms presented in the previous section. Then, we can be more clear in the next section.

4.1 BENCHMARKING PROCESS

To ensure a fair comparison between our proposed protocol and the standard QAOA approach, it is necessary to establish consistent guidelines for the classical optimization process. For benchmarking, we adopt the configuration that yields the best performance for the standard protocol. Specifically,

- We employ the BFGS algorithm as the classical optimizer, which resulted in better performance than other optimizers such as L-BFGS-B (Fig. 3.2) or Cobyla. Nelder-Mead has been discarded due to its poor scalability with the number of parametes.
- The QAOA parameters are trained using an incremental (layer-by-layer) approach: optimization begins with a single QAOA layer, and the optimal parameters obtained at each step are used to initialize the training of the next layer. (ADD REFERENCES)
- In this initialization strategy, the parameters from the previous iteration are reused for the corresponding layers of the deeper circuit, while the new layer is initialized with its γ parameter set equal to the last optimized γ value and its β parameter set to zero. This procedure improves convergence and reduces the likelihood of the optimizer becoming trapped in poor local minima, thereby providing a consistent reference for evaluating the performance of our protocol. (ADD REFERENCES)

4.2 RESULTS

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

5 Conclusion

The conclusions of your thesis.

5.1 This is a section title

Hello have a table.

What? A booktabs table?

5.1.1 This is a subsection

Hi again!

This is a subsubsection header

Bye this time.

Table 5.1: A nice table.

Column name	Explanation
last_modified	Kaas
version	Baas
db_schema_version	Haas

Table 5.2: A nice table.

Column name	Explanation
last_modified	Kaas
version	Baas
db_schema_version	Haas

BIBLIOGRAPHY

REFERENCES

[1] M. S. Sarandy, L.-A. Wu, and D. A. Lidar. Consistency of the Adiabatic Theorem. *Quantum Information Processing*, 3(6):331–349, December 2004.

[2] Tameem Albash and Daniel A. Lidar. Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002, January 2018.

[3] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm, November 2014. arXiv:1411.4028 [quant-ph].

[4] Edward Farhi and Aram W. Harrow. Quantum Supremacy through the Quantum Approximate Optimization Algorithm, October 2019. arXiv:1602.07674 [quant-ph].

[5] Wen Wei Ho and Timothy H. Hsieh. Efficient variational simulation of non-trivial quantum states. *SciPost Physics*, 6(3):029, March 2019. arXiv:1803.00026 [cond-mat].

[6] Eric R. Anschuetz, Jonathan P. Olson, Alán Aspuru-Guzik, and Yudong Cao. Variational Quantum Factoring, August 2018. arXiv:1808.08927 [quant-ph].

[7] P.L. Montgomery. A survey of modern integer factorization algorithms. *CWI Quarterly*, 7(4):337—366, December 1994.

[8] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[9] Xinhua Peng, Zeyang Liao, Nanyang Xu, Gan Qin, Xianyi Zhou, Dieter Suter, and Jiangfeng Du. A Quantum Adiabatic Algorithm for Factorization and Its Experimental Implementation. *Physical Review Letters*, 101(22):220405, November 2008. arXiv:0808.1935 [quant-ph].

[10] Narendra N. Hegade, Koushik Paul, Francisco Albarrán-Arriagada, Xi Chen, and Enrique Solano. Digitized Adiabatic Quantum Factorization. *Physical Review A*, 104(5):L050403, November 2021. arXiv:2105.09480 [quant-ph].

[11] Pablo Díez-Valle, Fernando J. Gómez-Ruiz, Diego Porras, and Juan José García-Ripoll. Universal Resources for QAOA and Quantum Annealing, June 2025. arXiv:2506.03241 [quant-ph].

[12] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices. *Physical Review X*, 10(2):021067, June 2020.

[13] Kostas Blekos, Dean Brand, Andrea Ceschini, Chiao-Hui Chou, Rui-Hao Li, Komal Pandya, and Alessandro Summer. A Review on Quantum Approximate Optimization Algorithm and its Variants. *Physics Reports*, 1068:1–66, June 2024. arXiv:2306.09198 [quant-ph].