Universiteit Leiden

Master Computer Science

Investigating the application of genetic algorithms to design parameterised quantum circuits to mitigate barren plateau problems

Name:	Richard Middelkoop
Student ID:	s2388715
Date:	5th July 2023
Specialisation:	Artificial Intelligence
1st supervisor:	Evert van Nieuwenburg
2nd supervisor:	Vedran Dunjko

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Parameterised quantum circuits (PQCs) have shown promise in quantum machine learning and optimisation problems. However, a significant obstacle to their effective training is the presence of barren plateaus. These are regions in the parameter landscape where the gradient of the cost function diminishes exponentially with the number of qubits. This phenomenon leads to vanishing gradients, hindering the search for optimal parameter values.

This research aims to address the barren plateau problem by exploring the application of a genetic algorithm to evolve PQCs to find designs with higher gradients that exhibit reduced barren plateau issues. By leveraging the principles of genetic algorithms, which mimic the process of natural evolution, this study seeks to improve the search capability within the parameter landscape. The genetic algorithm will be employed to iteratively generate and evolve PQCs, considering factors such as circuit depth, qubit counts and expressivity.

The potential of a genetic algorithm will be investigated in terms of gradient improvement, circuit error reduction and expressivity maximisation, potentially leading to the discovery of PQCs with more trainable and comprehensive search space landscapes. By mitigating the barren plateau issue, the study aims to enable more effective training and optimisation of PQCs for quantum machine learning tasks. The findings of this research will contribute to a better understanding of the challenges associated with parameterised quantum circuits and provide insights into potential solutions for improving their performance in practical applications. Furthermore, with the insights gained from this thesis, numerous suggestions will be made that could produce interesting results and will extend the insights gained into the application of genetic algorithms for parameterised quantum circuits.

Acknowledgements

First, I would like to thank Vedran Dunjko for giving me the opportunity to conduct my master's thesis with the Applied Quantum Algorithms group. Second, I would like to extend my thanks to the condensed ai group whose weekly meetings provided me with inspiration to pursue my research. Finally, I would like to thank Evert van Nieuwenburg for his excellent guidance and support.

Contents

Chapter 1

Introduction

Introduction

In the current era of noisy intermediate-scale quantum computing (NISQ), quantum computations are made using circuits. These circuits can be parameterised, and parameters can then be optimised. Optimisation can be done through the gradient. However, optimisation using gradients faces a significant challenge known as the barren plateau problem, wherein the gradients of parameterised quantum circuits (PQCs) vanish as the number of qubits increases. This limitation hinders the effective application of PQCs, making it challenging to achieve PQCs with both high gradients and low error rates.

To address this issue, this thesis aims to explore the potential of a genetic algorithm in discovering circuits that exhibit desirable properties, such as high gradients and low error rates. The purpose of identifying and optimising circuits with these characteristics is to enable a more manageable application of PQCs. To measure the success of the algorithm, a popular application of PQCs called variational quantum eigensolver (VQE) will be employed. VQE utilises a PQC as an ansatz to determine the ground state of a given physical system. The circuit parameters are optimised classically to find the parameter set that approximates the ground state as closely as possible. However, if the search space of the parameters exhibits symptoms of a barren plateau, the optimum ground state may not be reached.

Quantum circuits are powerful tools for performing complex computations, but they come with their energy consumption considerations. The energy consumption of quantum circuits primarily stems from the physical operations required to manipulate qubits and perform quantum gates. In traditional quantum computing architectures, such as superconducting qubits or trapped ions, these operations often involve cooling systems and precise control mechanisms that demand significant energy resources.

However, advancements in quantum computing techniques have shown promising prospects for reducing energy consumption and discuss the possibility of a "green" quantum advantage.[1, 2] One key factor is the optimisation of gradient calculations, which play a crucial role in training quantum machine learning models. Gradients provide information about the direction and magnitude of changes needed to improve the model's performance during optimisation. When gradients can be computed accurately and efficiently, it leads to better convergence and fewer iterations, ultimately reducing computational time and, consequently, energy consumption.

Recently, numerous research papers have been published exploring the barren plateau problem. This thesis will specifically focus on generating circuits with a high average gradient across their search space. These circuits will then undergo classical optimisation using a gradient-based optimisation method. The resulting minima will be evaluated based on their ability to approximate the ground state of an example quantum system, providing insights into the effectiveness of the proposed approach. The two main aspects that will be assessed are the gradient and the expressivity of a PQC. The high gradient should allow the circuit to learn within the search space of the circuits, and the expressivity should enable the search space to overlap with the ground state. By using two iterations of experiments in this thesis, using a non-heuristic genetic algorithm, experiments will investigate the potential of genetic algorithms to evolve circuits with improved gradients and expressivity. By shedding light on the generation of circuits with these properties, this research aims to contribute to the design of PQCs and their practical applications in quantum computing. More so, the performance of designed circuits is dependent on the quantum device on which they are executed. The architecture of each device consists of a layout describing the connectivity between qubits and the noise sources that occur when interacting with the device. This thesis will consider the IBM hardware devices and map the generated circuits onto the layout of any given device. The mapped circuits will be evaluated on their error rate that occurs when the circuit is operated, enabling the possibility of the genetic algorithm to discover circuits that are well suited to operate on a given device.

Chapter 2

Preliminaries

2.1 Quantum fundamentals

This section explains several basic quantum concepts. However, if you want to gain more in-depth knowledge about quantum mechanics, the work of Nielsen and Chuang: Quantum Computation and Quantum Information [3], is highly recommended.

2.1.1 Quantum Computing

Quantum bits (qubits) are the quantum variation of classical bits which can be either a zero or a one. The distinction of qubits is that they do not have to be either a zero or a one but the qubits, can be described as a linear combination of zero and one. The state of such a combination is usually written in Dirac-notation. The notation uses the *ket* which is shown as $|\psi\rangle$ to represent the state of a qubit. The two basic states of a qubit are $|0\rangle$ and $|1\rangle$. Any single qubit can be written as a linear combination of the two basis states.

$$|\psi\rangle = \begin{bmatrix}\alpha_0\\ \alpha_1\end{bmatrix} = \alpha_0|0\rangle + \alpha_1|1\rangle \tag{2.1}$$

α_0 and α_1 are called the amplitudes of the quantum state and can be used to measure the probability of either $|0\rangle$ or $|1\rangle$ occurring by calculating $|\alpha_x|^2$, with $\alpha_0, \alpha_1 \in \mathbb{C}$, $|\alpha_0|^2+|\alpha_1|^2 = 1$.
Quantum gates, quantum computation revolves around changing quantum states. The operations that cause these changes are created on a quantum circuit containing wires and quantum gates. In a quantum circuit, the state can change over time, in each step unitary linear operations are applied to the current state. Unitary is a property of a square matrix which holds if $UU^{-1} = I$ where I is the identity matrix.

$$|\psi'\rangle = U|\psi\rangle \tag{2.2}$$

Each quantum gate consists of a square matrix of varying sizes, depending on the number of qubits it acts on. A single-qubit operation is often used and can be represented as a 2x2 matrix. Moreover, multi-qubit operations act on multiple qubits like the controlled-not gate($CNOT$) which is a two-qubit operation that flips the state of the qubit on the second wire of a quantum system if the state of the qubit on the first wire is $|1\rangle$. The computation of such a two-qubit system with a $CNOT$ gate is shown in equations 3-5 in a circuit, in Dirac notation and matrix notation.

$$\begin{array}{l}|1\rangle \\ |0\rangle\end{array} \quad \begin{array}{c}\text{—}\bullet\text{—} \\ \text{—}\oplus\text{—}\end{array} \quad \begin{array}{l}|1\rangle \\ |1\rangle\end{array} \tag{2.3}$$

$$CNOT(|1\rangle \otimes |0\rangle) = CNOT|10\rangle = |11\rangle \tag{2.4}$$

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix} \left(\begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix}\right) = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix} \begin{bmatrix}0\\0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\end{bmatrix} \tag{2.5}$$

Eigenvectors and eigenvalues. like all common matrices, the linear operations on a quantum circuit have an eigenvector such that $U|v\rangle = v|v\rangle$, where v is the eigenvalue of operation U. It's important to note that an operation can have multiple eigenvectors, each associated with its eigenvalue. In other words, for a given quantum operation U, there may be several vectors that, after the application of U, remain in the same direction, with each vector being multiplied by a different scalar factor.

The set of all eigenvectors corresponding to the same eigenvalue forms what is known as an eigenspace. This eigenspace consists of all the vectors that share the same eigenvalue under the given operation. The eigenspace associated with a particular eigenvalue can have various dimensions depending on the multiplicities of that eigenvalue.

Eigenvectors and eigenvalues have practical implications in quantum computing, as they provide insights into the behaviour of quantum operations and can be utilised in various algorithms and simulations. By understanding the eigenvectors and eigenvalues of quantum processes, researchers and practitioners can analyse the behaviour of quantum systems, identify specific properties, and make predictions about the evolution of quantum states.

2.1.2 Quantum Circuits

In quantum computing, a quantum circuit is a sequence of quantum gates applied to a set of qubits. Each gate has a specific effect on the qubits, such as flipping their state, creating entanglement, or introducing phase

shifts. Quantum circuits are constructed by arranging these gates in a specific order to perform calculations or transformations on the qubits. After applying all the gates, the final state of the qubits represents the result of the computation.

Quantum chips. Quantum chips, also known as quantum processors or quantum computing hardware, are the fundamental building blocks of quantum computers. These chips are designed to manipulate and control qubits. The properties of quantum chips vary between each chip and require management using a combination of hardware design, error correction techniques, precise calibration, cooling, and software interfaces to harness the potential of quantum computing. For the scope of this thesis, the properties taken into consideration by the design of circuits are gate fidelity and connectivity. Gate fidelity relates to the accuracy of quantum operations on gates. A high gate fidelity ensures that minimal errors occur during quantum computations. The connectivity of quantum chips relates to the specific layout of the qubits on the chip. The connectivity then refers to the ability to establish interactions or entanglement between qubits. The connectivity architecture can affect the efficiency of quantum algorithms and the types of problems that can be solved.

Variational Quantum Eigensolvers (VQE). Variational quantum Eigensolvers (VQE) is an algorithm used to find the lowest energy state (ground state) of a quantum system. It is beneficial for solving problems in quantum chemistry, where the goal is to determine the electronic structure and properties of the molecules.

VQE combines classical and quantum computation. It uses a classical optimiser to find the optimal parameters for a quantum circuit that minimises the energy of the system being studied. The quantum circuit is referred to as the "ansatz". The ansatz is a parameterised quantum circuit that prepares a trial state for the system. It typically consists of a series of gates acting on the qubits, with the gate parameters being varied during the optimisation process. The quantum circuit prepares a trial state, and that state's energy is calculated using a quantum computer.

The classical optimiser adjusts the parameters of the quantum circuit iteratively, evaluating the energy of each trial state produced by the ansatz. The optimiser aims to find the parameters that minimise the energy, corresponding to the best approximation of the ground state.

By optimising the parameters of the ansatz quantum circuit through successive iterations, the VQE algorithm converges to an approximation of the ground state energy and its corresponding quantum state. This information is used to analyse the properties of the system under investigation.

A well-known application of the variational quantum eigensolver is the transverse Ising field model. It is used to study the behaviour of interacting quantum spins and has applications in condensed matter physics, quantum magnetism, and optimisation problems.

2.1.3 Optimisation

Parameterised quantum circuits (PQCs) play a crucial role in the field of quantum computing. They are quantum circuits that contain adjustable parameters, allowing them to represent a wide range of quantum states and operations. The optimisation of parameterised quantum circuits is a fundamental task in quantum machine learning and quantum optimisation algorithms like the VQE mentioned above. It involves finding the optimal values for the circuit parameters that minimise or maximise a given objective function. There are several approaches to optimise parameterised quantum circuits, depending on the specific problem and available resources. Here are some commonly used techniques:

Gradient-based optimisation. This approach involves computing the gradient of the objective function concerning the circuit parameters and updating the parameters in the direction of the steepest descent or ascent. The most commonly used gradient-based optimisation algorithm is known as the parameter-shift rule, which leverages finite-difference approximations to estimate gradients efficiently.

Variational algorithms. Variational algorithms combine classical optimisation techniques with parameterised quantum circuits. They iteratively optimise the circuit parameters by evaluating the objective function on a quantum computer and using classical optimisation methods to update the parameters. Variational Quantum Eigensolver and Quantum Approximate Optimisation Algorithm (QAOA) are popular variational algorithms used for solving quantum chemistry problems and combinatorial optimisation, respectively [4].

Stochastic optimisation. Stochastic optimisation methods use random sampling or noise injection to estimate the objective function. Techniques like quantum Monte Carlo (QMC) methods, Bayesian optimisation, and reinforcement learning can be adapted to optimise parameterised quantum circuits.

Quantum natural gradient. The natural gradient is a modification of the classical gradient that takes into account the underlying manifold structure of the parameter space. Quantum natural gradient methods adapt the classical natural gradient to the quantum setting, where the parameterised quantum circuit acts as the model.

It is important to note that optimising parameterised quantum circuits can be challenging due to noise,

limited qubit connectivity, and the presence of local minima/maxima in the landscape of the objective function. Moreover, the high dimensionality of the parameter space can pose additional difficulties. Therefore, researchers are continually exploring new optimisation techniques and heuristics to overcome these challenges and improve the efficiency and effectiveness of optimising parameterised quantum circuits.

2.1.4 Barren Plateaus

A barren plateau refers to a region in the optimisation landscape where the gradient signal becomes vanishingly small, making it difficult for classical optimisation algorithms to find the optimal set of parameters.

Several factors can contribute to the occurrence of a barren plateau:

Circuit depth and width: The depth of a quantum circuit refers to the number of layers or steps involved, while the width refers to the number of qubits or quantum components used. As the depth and width increase, the optimisation landscape becomes more complex, and the gradient can diminish significantly, leading to barren plateaus [5].

Choice of parameterised gates: The specific set of gates chosen for the parameterised quantum circuit can affect the likelihood of encountering a barren plateau. Certain gate choices may lead to more pronounced barren plateaus due to their inherent properties and the resulting structure of the optimisation landscape [6].

Random initialisation: Parameterised quantum circuits often require initialisation of the parameters before optimisation. Random initialisation is commonly used but can lead to instances where the circuit is initially far from an optimal configuration. In such cases, the optimisation process may struggle to make progress, resulting in a barren plateau.

Inadequate step size: The step size determines the size at which the circuit's parameters are updated during optimisation. If the step size is too large, the steps may be too drastic, causing the circuit to overshoot optimal solutions and struggle to converge.

Measurement noise: Real-world quantum computers are prone to noise and imperfections. When executing parameterised quantum circuits on these devices, the noise can introduce additional difficulties in optimising the circuit parameters. The presence of measurement noise can amplify the occurrence of barren plateaus by making the optimisation landscape even more challenging to navigate [6].

There exist several strategies that have been proposed to deal with barren plateaus in PQCs, including Circuit ansatz design, problem-tailored initialisation, noise mitigation and gradient rescaling.

2.2 Genetic algorithm fundamentals

Genetic algorithms are a class of optimisation algorithms inspired by the principles of natural selection and genetics. They are used to solve complex optimisation problems by mimicking the process of evolution. At their core, genetic algorithms work with a population of potential solutions and iteratively improve them over generations. Each solution is represented as a chromosome, typically encoded as a string of binary digits (genes). The collection of chromosomes forms the initial population.

The algorithms begin by evaluating the fitness of each chromosome in the population. Fitness is a measure of how well a chromosome solves a given problem and is determined by an objective function of the fitness function. The fitter a chromosome is, the more likely it is to be selected for reproduction.

The process of reproduction involves selecting parents from the population based on their fitness and creating offspring through crossover and mutation operations. Crossover involves exchanging genetic material(genes) between parent chromosomes to create new combinations. Mutation introduces random changes to the genes of offspring chromosomes, introducing exploration in the search space.

The offspring from the next generation, replace some or all of the less fit individuals in the current population. This process continues for several generations, with the hope that the population converges towards an optimal solution. Genetic algorithms leverage the principles of natural selection, where the fittest individuals have a higher probability of passing their genetic material to the next generation, and random genetic variations provide diversity and allow exploration of different parts of the solution space. The algorithm terminates either after a certain number of generations or when a satisfactory solution is found, the basis on which the algorithm can be terminated is called the stopping criteria. The final output is the best individual (or multiple individuals) from the last generation, representing an approximate or optimal solution to the problem.

Overall, genetic algorithms provide a powerful and flexible approach to solving optimisation problems by harnessing the principles of evolution and genetics. If some specific parts of the terminology remain unclear, the work of Busetti [7] explains each subsection of GA's in detail. To get a deeper overall understanding of the origins of genetic algorithms the works of Holland are more suitable [8].

2.2.1 Types of genetic algorithms

As mentioned in section 2.2 the primary components of a genetic algorithm are the population, chromosome, and gene. The population is the pool of the entire population, the chromosome represents a single individual, and the gene is a property of the individual. The algorithm passes through five general steps to complete a single evolution phase. These phases are the initialisation, evaluation, selection, crossover and mutation. In the initialisation, the initial pool of individuals is created, which is evaluated for their fitness in the evaluation phase. The fitness scores are used to select individuals to fit for reproduction during the selection, and the genes of two or more individuals combine during the mutation phase. Finally, the mutation phase occurs when the properties of a gene are altered. The exact functionality of each step can be varied and multiple variations exists all with different strengths and weaknesses in comparison to different implementations. In this section a selection of different genetic algorithms as well as different phases will be highlighted, note that there are many more versions which aren't covered in the thesis.
Here are some notable variations of genetic algorithms:

Canonical Genetic algorithm. This is the basic form of a genetic algorithm, as described above, it uses selection, crossover, and mutation operations to evolve the population.

Steady-State Genetic Algorithm. In this variation, only a few individuals(parents) are selected from the population to produce offspring in each generation. These offspring replace the least fit individuals in the current population. The advantage of the steady-state approach is that it often converges faster, especially when the population size is limited and should be a consideration when applying a GA whilst having limited computational resources.[9]

Adaptive Genetic Algorithm. This variation incorporates mechanisms to dynamically adjust parameters, such as mutation rate or selection pressure, during the optimisation process. It allows the algorithm to adapt its behaviour based on the progress and characteristics of the population. Adaptive genetic algorithms can enhance exploration in the early stages and exploitation of promising regions in later stages. This can be very useful when the problem characteristics or constraints change dynamically, requiring adjustment of algorithm parameters or when the canonical genetic algorithm seems to be having trouble with balancing the exploration and exploitation [10].

Parallel Genetic Algorithm. This approach utilises parallelism to speed up the search process. It involves dividing the population into sub-populations that evolve independently, possibly on separate processors or computing nodes, this is called the island model [11]. Communication and information exchange between sub-

populations can occur periodically to share useful genetic material. Parallel genetic algorithms are useful for large-scale optimisation problems that require extensive computational resources.

Niching Genetic Algorithm. This variation seeks multiple diverse solutions and is useful for multi-modal function optimisation and simulation of complex and adaptive systems. The Niching Genetic Algorithm aims to find multiple solutions in different regions of the search space, avoiding premature convergence to a single optimal solution. Techniques such as fitness sharing or crowding distance are used to preserve diverse solutions [12].

It is important to experiment and tune the algorithm parameters for the specific problem domain to find the most effective variant. Additionally, hybrid approaches that combine different variations or integrate genetic algorithms with other optimisation techniques can also be explored based on the problem requirements.

2.2.2 Variations of the GA phases

The phases of a GA can also be varied we will look at each phase individually a highlight a few notable variations.

Initialisation. The two most common methods are random initialisation and heuristic initialisation. Random initialisation randomly generates each individual with random values for the problem variables within the defined search space for the initial population. Heuristic initialisation is initialisation that requires prior knowledge about the problem that is being investigated which can be for example starting the initial population with existing solutions to the problem at hand [7].

Evaluation. An individual's evaluation is completely dependent on the fitness function related to the problem being investigated. This does not however mean that once a fitting fitness function has been created that it shouldn't be altered. Fitness functions can be complex to design and often after analysing the results of an experiment some tweaks and/or changes to components within the function may be required.

Selection. The two common variations of selection are proportional selection, often revered to as roulette wheel selection and tournament selection. The proportional selection gives each individual a probability of being selected based on their fitness value. Individuals with higher fitness have a greater chance of being selected for reproduction. Tournament selection chooses a fixed number of individuals for the population and the individual with the most increased fitness is chosen for reproduction. This process is repeated multiple times until the new population is filled. It's worth noting that the extent of the proportion increase for high fitness and the number of individuals in a tournament play a big role in the trade-off of exploration versus exploitation [8].

Crossover. There are many variations for the combinations of two parents in the crossover step. Three common methods are single-point crossover, two-point crossover and uniform crossover. Single-point cross selects a random point along the chromosome, and the genetic material is exchanged between the parents at that point to create offspring. Two-point crossover selects two random points along the chromosome, and the genetic material between the two points is exchanged between the parents to generate offspring. Uniform crossover selects for each bit or gene of the offspring randomly from one of each parent, allowing for a more diverse combination of genetic material [7].

Mutation: The mutation phase is a particularly important choice as to what variation is used and how the parameters of that variation are set. Because the mutations introduce random changes that can disrupt good solutions and should be monitored closely to control the balance between exploration and exploitation in the search process. Bit flip mutation is the simplest form of mutation and the most commonly used operator for binary-encoded problems. In this type of mutation, a random bit or gene in an individual's chromosome is flipped from 0 to 1 or vice versa. There are several different mutation variations but their applicability heavily depends on the type of encoding used for the individual.

In general, it can be difficult to manage local optimums, global optimums, exploration and exploitation. Therefore it is important to think of what each change in hyperparameter or variation in methods will change. The final result of your algorithms will often not tell the whole story, so it could be wise to keep track of your data during the training process to identify what kind of changes should be tried and to what parameters.

2.3 Related work

In recent years, there has been growing interest in the application of genetic algorithms within the quantum research field [13, 14, 15, 16]. For our avenue of applying genetic algorithms to parameterised quantum systems, there have been several works that provide context.

One notable study in this field is the work by Wakaura et al. [17] who proposed a variational quantum eigensolver method with GA on hydrogen molecules as the target. Wakaura et al. demonstrated that the optimisation of parameters using genetic algorithms should not be used without the combination of other more localised search methods.

The work of Yabuki et al. [18] proposed a method to apply genetic algorithms to the quantum circuits' design. The GA was able to evolve circuits that performed quantum teleportation correctly and improved upon the earlier work by William et al. [19]. Massey et al. [20] evolved several existing quantum algorithms using a genetic algorithm and found some different circuits with the resembling costs as the known circuits. These results show that GAs can effectively explore the high-dimensional search space of quantum circuits, leading to improved circuit structures.

Variational Quantum Circuits play a central role in several applications in the current quantum era due to their robustness to the device's noise. Therefore, it seems natural that genetic algorithms are applied to a significant issue with VQCs, which are barren plateaus. Acampora et al. tried precisely this. Their paper proposed to apply GAs to train VQCs used as quantum classifiers. They found that their experiments could obtain accurate solutions in fewer queries than in gradient descent.

Furthermore, Cerezo et al. and Arrasmith et al. [21, 22] studied the effects of barren plateaus on parameterised quantum circuits and the impact of barren plateaus on non-gradient optimisation methods. These studies show the presence of barren plateaus in PQCs and show that even non-gradient-based techniques are impacted by the existence of these plateaus.

In summary, genetic algorithms have shown potential in recent research when applied to quantum circuits/systems. Furthermore, the barren plateau problem is a prominent issue which is still present in state-of-the-art ansatzes.

Chapter 3

Optimising the gradient

3.1 Problem statement

Parameterised quantum circuits (PQCs) have emerged as a promising approach for quantum machine learning and optimisation problems. These circuits consist of a fixed set of gates, where the parameters of these gates are varied to perform different computations. The effectiveness of PQCs relies on the ability to find optimal parameter values that minimise the cost function associated with the given task.

However, recent studies have revealed the presence of a phenomenon known as "barren plateaus" in parameterised quantum circuits [6]. Barren plateaus refer to regions in the parameter landscape where the gradient of the cost function approaches zero exponentially with the number of qubits. This poses a significant challenge for training PQCs, as it leads to vanishing gradients and impedes the ability to find meaningful solutions.

The barren plateau problem arises due to the accumulation of two factors: circuit depth and the presence of random initialisation of parameters. As the number of qubits and circuit depth increase, the parameter landscape becomes increasingly flat, resulting in a lack of gradient information for optimisation algorithms to effectively traverse the landscape and find optimal parameter values.

The main objective of this research is to investigate the application of a genetic algorithm to evolve PQCs that have less of a barren plateau issue. The specific objectives include:

- Analysing the relationship between circuit depth, qubit count, and the prevalence of barren plateaus.
- Studying the ability of genetic algorithms to overcome barren plateaus.
- Developing insight into the capability of genetic algorithms to evolve PQCs to optimise for multiple objectives.
- Evaluating the expressivity of the evolved PQCs.
- Comparing the performance of the proposed algorithm with a benchmark based on the fitness of the evolved circuits.

The research aims to provide a comprehensive investigation of the application of genetic algorithms to the barren plateau problem in parameterised quantum circuits. The expected outcomes of this research include:

- Identification of circuit and algorithmic characteristics that lead to the occurrence of barren plateaus.
- Insights into the ability of genetic algorithms to overcome barren plateaus and their limitations.
- Empirical evidence demonstrating the effectiveness of the proposed genetic algorithm through experimental evaluation in comparison to benchmarks on a benchmark task.
- Insight into the relationship between the gradient and expressivity
- Recommendations for practitioners and researchers on selecting suitable genetic algorithm methods for training PQCs under barren plateau conditions.

Addressing the barren plateau problem in parameterised quantum circuits is crucial for advancing the field of quantum machine learning and optimisation. Overcoming the limitations imposed by barren plateaus will enable more effective training of PQCs, leading to enhanced performance in a wide range of applications, including quantum chemistry, machine learning, and optimisation. Additionally, understanding the causes and characteristics of barren plateaus can contribute to the development of new (genetic) optimisation algorithms and techniques that are specifically tailored for quantum systems.

3.2 New development

3.2.1 Theory

The purpose of the development made by this thesis is to introduce a way to escape the barren plateau issue that many ansaetze struggles with. The idea is to genetically evolve circuits to find ones that have a promising average gradient in the search space set by the parameters of a PQC. As the existence of a barren plateau is indicated by vanishing gradients, the genetic algorithm should have a component that tracks the gradient of the circuit in the fitness function. There are several methods of getting an estimation of the circuits' gradient. The work of Schuld et al. [23] contains a numerical formula for a point-wise evaluation of the gradient within the search space. The work of Perez-Salinas et al.[24] makes a more general statement about the search space by investigating the landscape through the lens of information content, which is a measure of the variability between points in parameter space. Furthermore, for the thesis, we will use the point-wise evaluation of Schuld et al. as an average across all parameters. The point-wise evaluation can, however, have difficulties extracting an accurate estimation of the quantum device on which the PQC is run if it has high error rates on the used parts of its layout. Therefore, we will also map the generated circuit onto an IBM device and measure the error rates of the qubits and connections that are used by the circuit.

The encoding of the PQC for the GA should encode the structure of the circuit in a manner that allows for the selection and crossover phases to be applied to the individual without causing a computational strain, as these operations will have to be performed for a large number of iterations. This means that the encoding should ideally be binary or something equally accommodating.

As the best-performing setup in terms of the variation of a genetic algorithm will vary depending on the type of problem that the PQC is being applied to, the implementation should allow for easy exchange between methods and easy change of the parameters of these methods.

3.2.2 Implementation

In the implementation[25] of the genetic algorithm, multiple variations have been created which can be switched between varying parameters in a parameter file. In the implementation section, we will divide the created algorithm into five different sections and will discuss the implementation of each section independently, the sections are: Individuals, Selection, Combination, Mutation and Fitness. The overall structure of the algorithm is based on the algorithm Xenakis [26] built by Molecular Quantum Solutions [27].

Individual

In each generation of the algorithm a population is created filled with individuals, this section will explain what is stored in each individual. The individual has two features: fitness and the chromosome. The fitness is a floating point value which states how "good" an individual is, the fitness section will provide more details of how we measure the "goodness" of an individual. The chromosome of the individual consists of a bit-string which encodes a PQC[26]. The string is divided into the gate and qubit identifiers as can be seen in figure 3.1, the bit-value of the gate identifier simply refers to an entry in the list of possible gates. The qubit identifier is then used to identify the qubits on which the gate will operate. This is done by reordering the ordered list of all possible qubits, using the qubit identifier as the seed for the reordering. A single qubit gate will then only use the first entry to create the operation and a multi-qubit gate will use the first n entries based if the gate requires n qubits. For example, gate identifier and qubit identifier $\begin{bmatrix}010100 & 0101\end{bmatrix}$ will be decoded as follows: The gate identifier refers to the CNOT gate as the 20th gate in the gate list. $\begin{bmatrix}010100\end{bmatrix} \rightarrow$ 

The list of possible qubits will be reordered using the qubit section, $\begin{bmatrix}0101\end{bmatrix} \rightarrow$ random seed 5. The random seed will reorder the list of qubits $\begin{bmatrix}0 & 1 & 2 & 3\end{bmatrix} \rightarrow$ permutation seed $5 \rightarrow \begin{bmatrix}1 & 2 & 0 & 3\end{bmatrix}$. Because the gate identifier is the CNOT gate the operation will use the leading two qubits for its operation resulting in the following operation:

(3.1)

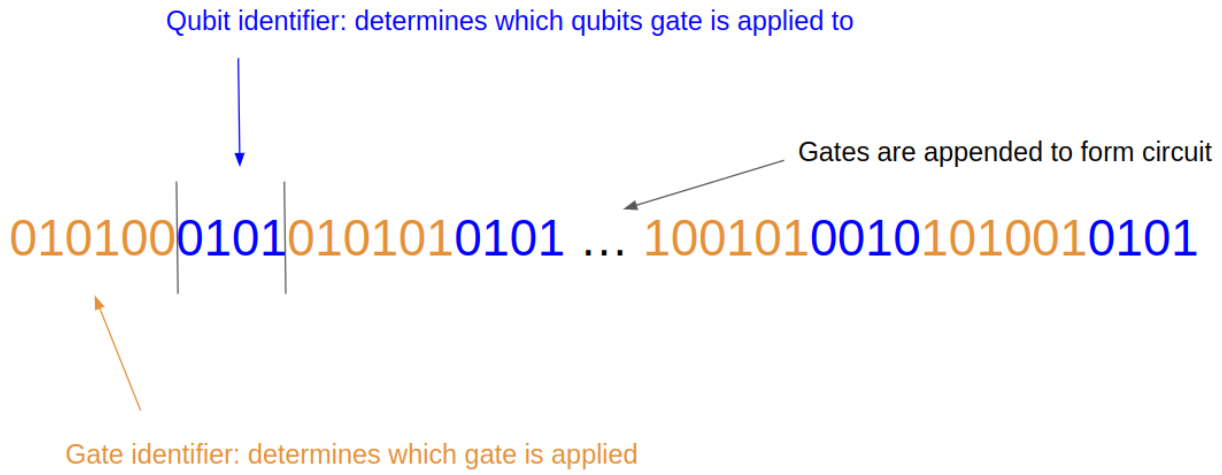


Figure 3.1: The structure of the chromosome based on Xenakis[26]

Selection

The selection step is made using the Elitism principle which means that the best x percentage of the population, based on their fitness, is selected for the next generation without any combination or mutation. The percentage can be varied in the implementation but in the literature generally, a percentage between ten to twenty percent is used. For our implementation, we use ten percent. Note that the selection phase combined with the mutation phase are the two main tuning tools for balancing exploitation and exploration.

Combination

The combination step combines the chromosomes' uniform crossover. This means that based on some odds, like a coin flip, a bit is either swapped between two parents or not. In the implementation two different variations of uniform crossover are used, the first variation is a vanilla uniform crossover with a 50% chance of swapping. The second variation is a variant of modified uniform crossover[28]. Modified uniform crossover has a weighted swap rate depending on some parameter. For our implementation, the weight depends on the crowd score of the population. The crowd score of the population is modelled on the crowd score used by NSGA-II[29] and is computed by calculating how closely related the parent chromosomes are to the other individuals in the parent population, the higher the crowd score the higher the chance of swapping. This variation is added to create more diversity in the population and this is something that can be used to improve the exploration for the global optimum, but it will be very detrimental for convergence.

Mutation

The mutation step is identical to Xenakis[26] and simply replaces each bit with a random bit in the individuals' chromosome of the population with a given mutation chance of this occurring.

Fitness

The base version of the algorithm calculates for each individual in the population a gradient estimation at initialisation and the cumulative error of the circuit from the individuals' chromosomes as can be seen in algorithm 1. The gradient is estimated by taking the central difference of each parameter used by the operations of the circuit, this is known as the parameter shift rule. The average gradient is then computed by computing the total size of all the gradients and dividing by the number of parameters as shown in algorithm 2. The error is found by transpiling the circuit to the quantum layout of a given IBM chip and adding each 2-bit gate error that occurs together. The formula of algorithm 2 is based on the work of Schuld et al. which outlines the computation of quantum gradients using numerical differentiation as a means of estimating the gradient through black-box evaluations.[23] The average gradient score that is used is simply an extension of the formula by averaging the estimated gradients of all parameters in the given circuit. The random values that the parameters are initialised

with can be changed to start closer to zero as a heuristic.

Algorithm 1: Fitness function

Result: Population sorted in descending order of fitness
initialisation;
for *individual* ∈ *population* **do**
  circuit ← genome_to_circuit(individual.chromosome);
  gradient ← compute_gradient(circuit);
  configured_circuit ← configure_circuit_to_backend(circuit);
  circuit_error ← get_circuit_properties(configured_circuit);
  individual.fitness = $\frac{1}{1+circuit\_error} * gradient$;
end

Algorithm 2: Compute gradient

Input: $Circuit, nr\_of\_parameters$
Result: The estimated average gradient for a given parameterised quantum circuit
total_gradient = 0;
$\epsilon = 10^{-5}$;
Let parameters[0...nr_of_parameters-1] be a new array;
for *i* ∈ *range(0,nr_of_parameters)* **do**
  parameters[i] ← (random number between [0,2π]);
end
for *i* ∈ *range(0,nr_of_parameters)* **do**
  parameters_forwards = parameters;
  parameters_forwards[i] = parameters[i] + $\frac{\epsilon}{2}$;
  parameters_backwards = parameters;
  parameters_backwards[i] = parameters[i] - $\frac{\epsilon}{2}$;
  partial_gradient ← $\frac{energy\_from\_circuit(parameters\_forwards) - energy\_from\_circuit(parameters\_backwards)}{\epsilon}$;
  total_gradient = total_gradient + partial_gradient^2;
end
`// if nr_of_parameters=0,output=0 to prevent division by 0`
Output: $\frac{\sqrt{total\_gradient}}{nr\_of\_parameters}$;

Parameters

Table 3.1: List of parameters

	Parameter name	Parameter type
1	Random seed	None or positive integer
2	Population size	positive integer
3	Max generations	positive integer
4	Mutation rate	float between 0 and 1
5	Elitism rate	float between 0 and 1
6	Qubit sectioning length	positive integer
7	Chip backend	real or fake IBM backend chip
8	Chip backend simulator	real or fake IBM backend chip
9	Number of qubits	positive integer
10	Number of ising qubits	positive integer
11	Number of gates	positive integer
12	Number of shots	positive integer
13	Improvement criteria	float between 0 and 1
14	Modified uniform crossover	True or False
15	Parameter initialisation lower bound	float between 0 and 1
16	Parameter initialisation upper bound	float between 0 and 1

The current iteration of the algorithm allows for tuning of several algorithms as can be seen in table 3.1.

1) Random seeds can be used to fix the individuals created at the initialisation. Note that this will not create identical results due to the randomness within the genetic process itself.

2) Population size is the number of individuals used in the population pool.

3) The max generation sets the maximum number of generations set as stopping criteria for the algorithm.

4) The mutation rate is the chance that a bit in the encoding will be replaced by a new random bit, and is measured separately for each bit.

5) The Elitism rate is the partition size that is used for the selection phase and will decide the number of well-performing individuals that are kept between generations.

6) The qubit sectioning length is the length of the qubit identifier as seen in figure 3.1 and can be increased to allow for more ordering possibilities of the qubits.

7,8) The chip backend and chip backend simulator is used for the transpilation and simulation of the individual.

9) The number of qubits is the encoded number of qubits in the individual and will set the size of the individual's circuit.

10) The number of Ising qubits is the problem that is the template function that is being used for the experiments and can be used to set the size of the problem.

11) The number of gates is the number of operations that will be encoded in the chromosome.

12) The number of shots is the repetitions used when measuring the circuit. Increasing the number of shots will lead to longer running times but improved accuracy while decreasing the number of shots will have the opposite effect.

13) The improvement criteria is a parameter used as stopping criteria and will finish the experiment if the fitness function hasn't changed the set percentage with a fitness's moving average for 50 generations.

14) Modified uniform crossover is a variation of crossover which will be used if set to true. The crossover variation will otherwise be a generic uniform crossover. 15,16) The parameter initialisation parameters are the bounds used in the gradient computation that are by default set to 0 and 2π.

3.3 Experiments and results

3.3.1 Experimental setup

As the aim of this thesis revolves around the possibility of using genetic algorithms to design circuits that have less of a barren plateau issue, the genetic algorithm will be tested on 4,6,8 and 10 qubits and will be compared to a benchmark on each qubit. The benchmark is created by generating random chromosomes each generation instead of the crossover and using a mutation rate of zero. Each experiment will be evaluated on three aspects

- the fitness of the best individual in the population
- the error rate of the best individual in the population
- the overall crowd score of the population

Furthermore, the secondary purpose of this thesis is to evaluate the structures of the circuits that have been found. For that reason in the experiments the modified uniform crossover based on the crowd score has also been included, this will ensure that the final population will have some different structured circuits to be analysed, instead of some slight variations around the global optimum.
The experimental protocol consists of the following steps:

1. Initialise the population with random individuals and sort the population based on their fitness score.
2. Execute the selection, combination and mutation steps and sort the population based on their fitness score.
3. Measure the fitness of the best individual in the population and the average fitness of the family, the elitism pool, of the population.
4. Measure the error rate of the best individual in the population and the average error rate of the family of the population.
5. Every 50 steps save the average fitness, average error rate and average crowd score of both the individuals and families.
6. Repeat steps 2-5 iteratively until convergence or a predefined stopping criterion is met.
7. Record the final population.
8. Repeat the above steps for 4, 6, 8 and 10 qubits to perform a comprehensive analysis of the implementation.

3.3.2 Data analysis

The experimental data collected from the experiments are analysed to determine the performance of the genetic algorithm and to determine the presence of barren plateaus. Additionally, the evolution of the various metrics spread across the different qubit numbers is analysed in comparison to the benchmarks.

3.3.3 Limitations

It is important to acknowledge the limitations of the experimental setup. Due to the current constraints of quantum hardware, we may encounter noise and errors during the execution of quantum circuits, which can affect the observed phenomena. Another constraint is the variability of the genetic algorithms can result in varying results and rerunning the experiments will result in different results. Furthermore, the computational resources and time constraints may limit the maximum circuit depth and qubit count that can be investigated in this study.

3.4 Results

3.4.1 GA performance

The performance of the algorithm as compared to the benchmark on the benchmark problem, which is the 1-d transverse ising field model can be seen in figure 3.2. The results show that the algorithm can outperform the benchmark across all qubit counts. The extent to which the algorithm improves on the benchmark is close to double the fitness value and the benchmark, logically, shows no sign of improvement which our algorithm does show. Additionally, it can be seen that the QGA does not keep its upward trajectory but can sometimes lose good-performing individuals. Partly, this is because the estimation value within the fitness function can vary between measurements but a fine-tuned genetic algorithm should normally be able to handle such measurement noise as it is a very robust type of algorithm, and it does regain its lost information, eventually. Nevertheless, the algorithm that is used for the experiment is a very crude variation of a genetic algorithm without any heuristics applied to any of the phases in the algorithm and with some more clever selection, which will depend on the problem, the information loss can probably be prevented. Possible avenues of improvement will be explored in chapter 5, the outlook.

In terms of the barren plateau, it can be stated that in figure 3.2 the algorithm is still able to find areas with a high gradient within the search space at this level of qubit scaling.

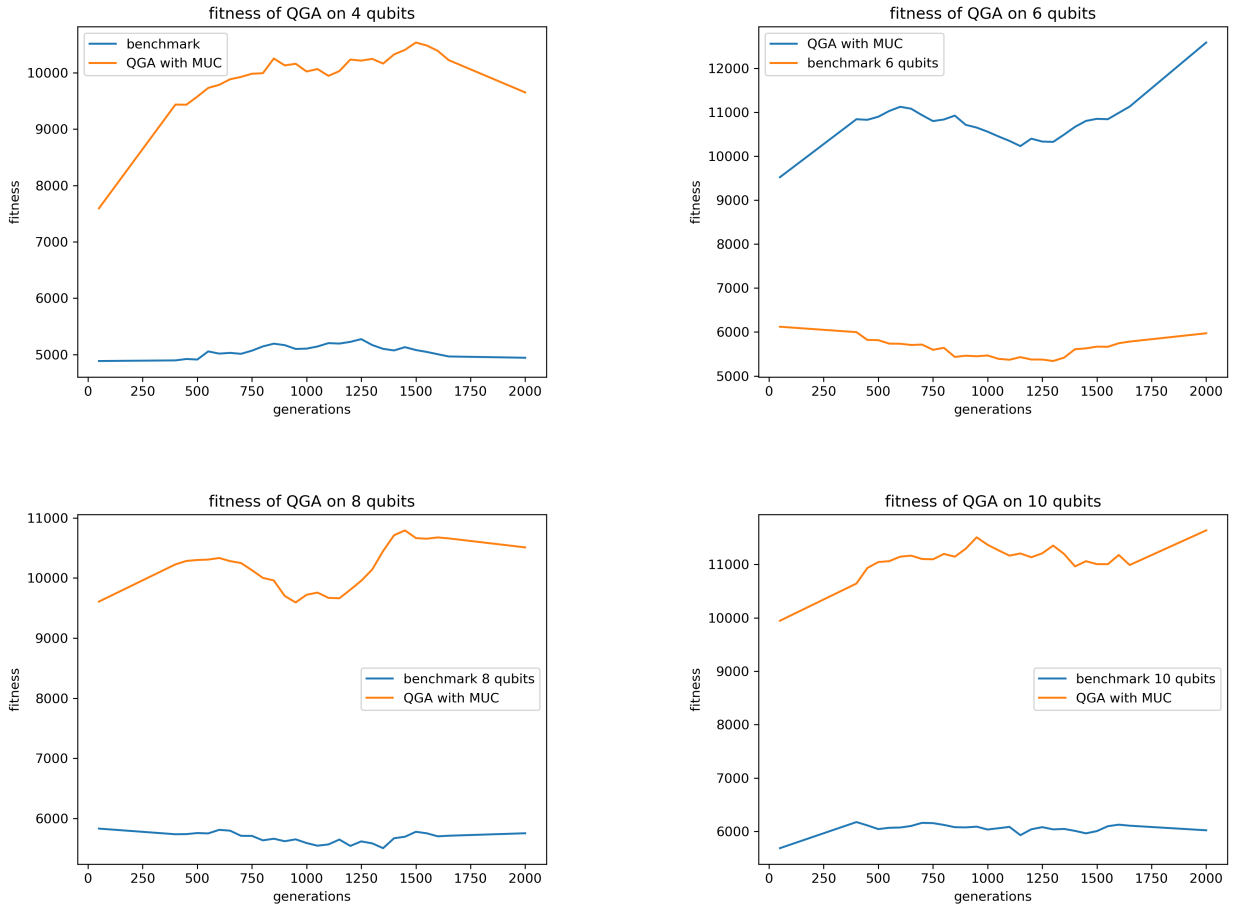


Figure 3.2: the fitness score of the QGA on 4, 6, 8 and 10 qubits on an equal size 1-d ising model problem with mutation rate set to 0.01 and simulated on the fake Toronto backend over 2000 generations.

3.4.2 Evolution of individuals

The evolution of the number of gates within the experiments depended on the type of gates present within the individuals' chromosomes. This is because the individuals have a fixed length and only through the transpilation of the unconfigured circuit to the IBM backend gate set can the number of gates be varied. In figure 3.3 it can

be seen that the number of gates oscillates around a certain point that relates to the chromosome length for both the total and the number of controlled gates. This indicates that the actual number of gates does not play an essential factor in the search for a high gradient as much as the location or combination of gates on specific qubits.

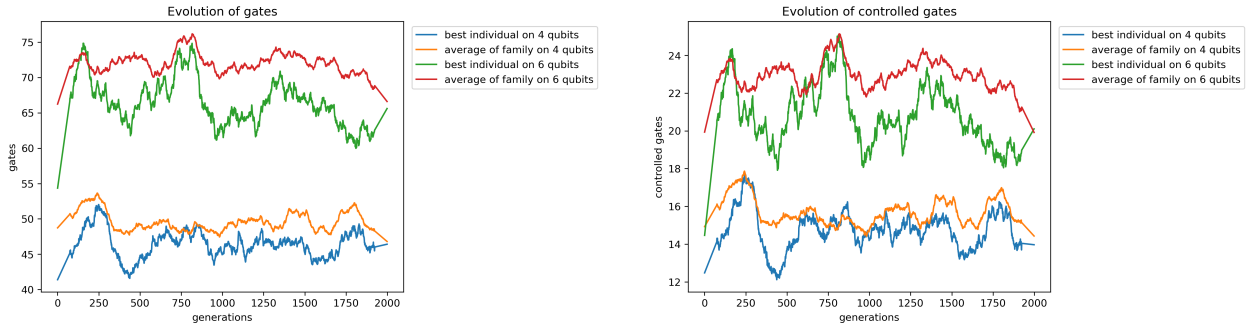


Figure 3.3: the evolution of the number of gates by QGA on 4 and 6 qubits on an equal size 1-d ising model problem with mutation rate set to 0.01 and simulated on the fake Toronto backend over 2000 generations.

The evolution of the error rate across the qubit counts as seen in figure 3.4 shows that the benchmark is consistently worse than the QGA which means that the algorithm can optimise not only for the gradient but is also able to simultaneously optimise for the error rate. Furthermore, note that when the qubit count increases the total error rate of the circuits reduces. This is likely because the increased qubit count increases the number of options for the transpiler to create circuit layouts.

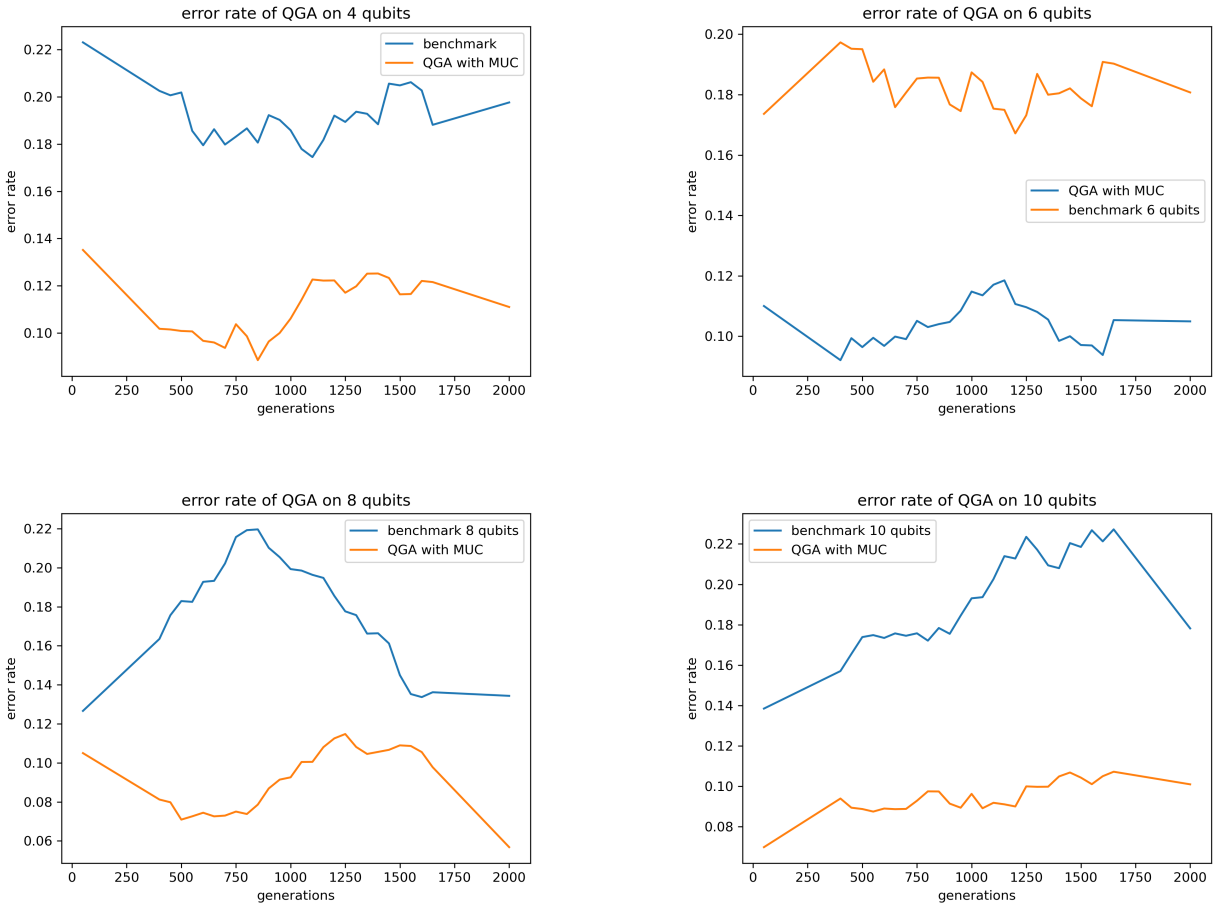


Figure 3.4: The evolution of the error rate by QGA on 4, 6, 8 and 10 qubits on an equal size 1-d ising model problem with mutation rate set to 0.01 and simulated on the fake Toronto backend.

The evolution of the crowd score across the qubit counts as seen in figure 3.5 shows that the modified uniform crossover can consistently keep a diversity of around 33% similarity across the entire evolution process. Furthermore, the crowd score of the benchmark shows that without the crossover and mutation phases, no improvements can be made and cannot improve beyond the random chromosomes.

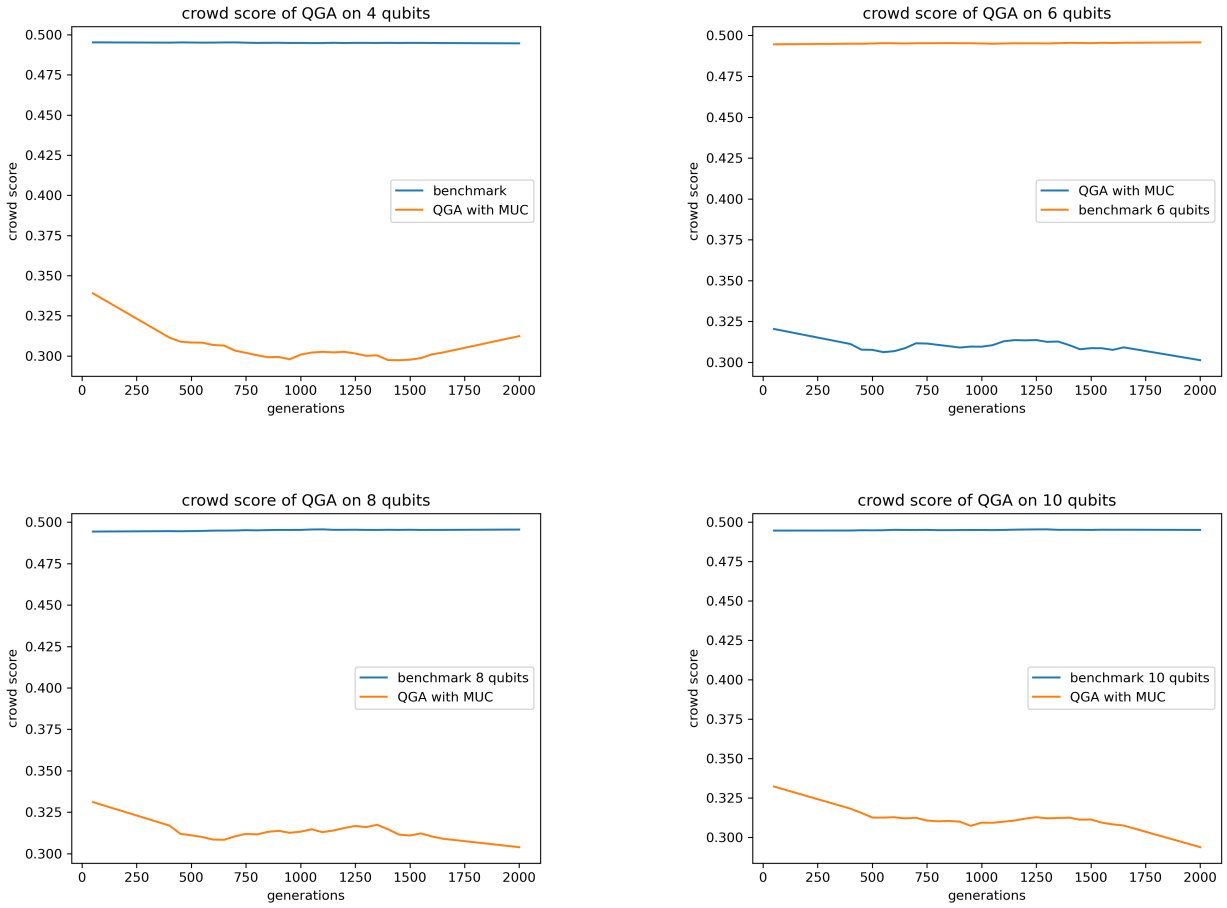


Figure 3.5: The evolution of crowd score by QGA on 4, 6, 8 and 10 qubits on an equal size 1-d ising model problem with mutation rate set to 0.01 and simulated on the fake Toronto backend.

Chapter 4

Gradient versus expressivity

4.1 Gradient and expressivity requirements

In the previous chapter, we optimised circuits for their gradient at initialisation. In order for these circuits to be any good they have to uphold the following requirements:

- During the optimisation steps no barren plateau must be found that prevents the optimum from being reached.
- The PQC search space needs to overlap with the global optimum

The first point refers to finding a barren plateau in later iterations of the optimisation, which makes it extremely challenging for optimization algorithms to make progress. To ensure that a circuit is good for optimisation, it is crucial to avoid encountering barren plateaus during the optimization process. Finding oneself trapped in a barren plateau can significantly hinder the search for an optimal solution. If the gradients become too small, the optimization algorithm may struggle to differentiate between promising and unpromising directions, leading to slow convergence or even convergence to sub-optimal solutions.

The second point specifies the requirement that for effective circuit optimisation, the PQC search space must overlap with the global optimum. The search space refers to the set of all possible configurations of the circuit's parameters, and the global optimum represents the best possible solution to the optimisation problem. If the PQC search space does not overlap with the global optimum, the algorithm will be restricted to exploring only a limited subset of the total search space, missing out on potentially superior solutions. To ensure that the PQC search space overlaps with the global optimum, one must carefully consider the circuit's architecture and fitness function in the design of the GA and its heuristics.

In conclusion, avoiding barren plateaus and ensuring that the PQC search space overlaps with the global optimum are critical requirements for good circuit optimisation. By addressing these requirements, one can improve the efficiency and effectiveness of the optimization process, leading to better solutions for the given problem.

4.1.1 Requirement evaluation

In the analysis of tables 4.1 and 4.3, we can assess whether the benchmark and our algorithm satisfy the gradient and expressivity requirements. From the results presented, it is evident that neither the current crude version of the GA nor the benchmark is capable of reaching the global optimum consistently. The tables demonstrate that the optimised energies obtained by both algorithms differ from the ground state energy by a significant margin, as indicated by the delta energy values. This indicates that the parameterised quantum circuits (PQCs) utilised in these algorithms do not overlap with the global optimum for the majority of cases.

One possible reason for this outcome is that the Quantum Genetic Algorithm (QGA) favours relatively small search spaces of PQCs. The QGA's search space might cause these constraints due to a combination of factors, including the error rate component of the fitness function and the basic genetic phases that have been used. If the circuit only spans a small region but exhibits a very high gradient within that confined area, the fitness of individuals in the population will consistently remain high across generations as opposed to a circuit with higher expressivity with a varying degree of gradients within the search space. As a result, the QGA favours individuals that become trapped in localised regions with a sub-optimal solution, rather than optimising for more expressive PQCs with a broader search space which would more likely find the global optimum.

Table 4.1: Delta energy of the benchmarks

	4 qubits	6 qubits	8 qubits	10 qubits
Average of the final population	2.51	5.61	11.19	10.11
Average of the final family	2.87	6.26	11.88	10.51
Best final individual	1.58	4.36	12.99	10.51
Individuals that reach the ground state	24%	7%	0%	0%

Table 4.2: Delta energy of the QGA

	4 qubits	6 qubits	8 qubits	10 qubits
Average of the final population	3.09	7.01	8.06	12.18
Average of the final family	3.64	7.24	8.53	12.76
Best final individual	4.50	8.0	10.49	13.28
Individuals that reach the ground state	16%	1%	1%	0%

4.2 Genetic algorithm updates

In order to improve the expressivity of the GAs results, the development has to receive some updates. Preferably a slight adaptation to maintain a crude GA which can act as a base layer for future developments for a wide array of problem instances. Therefore only a change to the fitness function is made by adding an energy component with a small weight in comparison to the gradient component. The implementation can be seen in the updated pseudo-code 3 and 4, **the updated lines of code are added in bold**.

Algorithm 3: Updated fitness function

Result: Population sorted in descending order of fitness
initialisation;
for *individual* ∈ *population* **do**
  circuit ← genome_to_circuit(individual.chromosome);
  gradient, **energy** ← compute_gradient(circuit);
  configured_circuit ← configure_circuit_to_backend(circuit);
  circuit_error ← get_circuit_properties(configured_circuit);
  individual.fitness = $\frac{1}{1+circuit\_error} * gradient * -$**energy**;
end

Algorithm 4: Updated compute gradient

Input: $Circuit, nr\_of\_parameters$
Result: The estimated average gradient for a given parameterised quantum circuit
total_gradient = 0;
$\epsilon = 10^{-5}$;
Let parameters[0...nr_of_parameters-1] be a new array;
for *i* ∈ *range(0,nr_of_parameters)* **do**
  parameters[i] ← **(random number between [0,0.01])**;
end
energy ← 100 minimise steps using COBYLA;
for *i* ∈ *range(0,nr_of_parameters)* **do**
  parameters_forwards = parameters;
  parameters_forwards[i] = parameters[i] + $\frac{\epsilon}{2}$;
  parameters_backwards = parameters;
  parameters_backwards[i] = parameters[i] - $\frac{\epsilon}{2}$;
  partial_gradient ← $\frac{energy\_from\_circuit(parameters\_forwards) - energy\_from\_circuit(parameters\_backwards)}{\epsilon}$;
  total_gradient = total_gradient + partial_gradient^2;
end
`// if nr_of_parameters=0,output=0 to prevent division by 0`
Output: $\frac{\sqrt{total\_gradient}}{nr\_of\_parameters}$, **energy**;

The two main changes are to the random initialisation and the new component. The initialisation now has a much tighter window to select from which will mean that the algorithm will on be initialised closer to identity. This reduces the chance that a well-performing individual will be discarded over a randomly great initialised individual. Secondly, this should to some extent identify individuals that reach a barren plateau after several optimisation steps. Because the combination of the close-to-identity initialisation with the energy component will produce a worse result if a barren plateau is found within those first one hundred steps. Furthermore, even if there is no barren plateau preventing optimisation, if the individual just cannot reach the global optimum due to a lack of expressivity then the energy component will also have a worse value than if the global optimum can be reached. Note that if the energy found is exceptionally bad then the fitness function will become negative, so even though the energy component has a relatively low weight in comparison to the gradient component, it does put a lower bound on the performance of the individual.

The expectation of the results is that there is a significant improvement in the delta energy and the percentage of individuals that reach the ground state. However, likely, there will still be some individuals with large gradients that will come close to the ground state but will not be able to reach it.

4.3 Expressivity Results

Upon analysing tables 4.3-4.6, it becomes evident that the updated Genetic Algorithm (GA) exhibits significant improvements compared to the initial results. Notably, when considering systems with 4 and 6 qubits, the updated algorithm demonstrates that even a relatively small energy component is sufficient to discover individuals with high gradients, capable of reaching the ground state. Although the enhancement may not be strong enough yet to evolve individuals on 8 and 10 qubits that can reach the ground state, there is a substantial improvement observed in the delta energy results compared to the initial algorithm. It is important to note that the tables also track the highest-scoring individuals across all generations. A new individual is only added to this pool if it outperforms the current best-scoring individual. With the expectation that the updated algorithm performs as intended, the new initialisation process should ensure that the elitism section of the final population performs equally well with the collection of best-scoring individuals. Consequently, this implies that there is no longer a loss of information from one generation to the next, a conclusion supported by the observed results. The promising outcome from the updated GA underscores the effectiveness of the modifications introduced. Even with only a small energy component, individuals that balance gradients and expressiveness can be generated, thereby increasing the likelihood of reaching the ground state in systems with fewer qubits within a reduced number of optimisation steps. Although further enhancements are required to achieve similar success on larger systems, the significant improvements observed in the delta energy results are indicative of the algorithm's progress.

In summary, the analysis of tables 4.3-4.6 highlights the considerable advancements achieved by the updated GA. The algorithm's ability to generate individuals with high gradients and improve upon the initial delta energy results substantiates its effectiveness. Furthermore, the updated initialisation ensures the preservation of valuable information throughout the evolutionary process. These insights lend support to the notion that the algorithm's improvements have successfully mitigated the loss of information from one generation to the next, as confirmed by the obtained results and improved the expressivity of the circuits.

Table 4.3: Delta energy of the QGA with minimise update on 4 qubits on 500 generations

	final population	Best individuals during the process
Average of the total population	2.35	0.002
Average of the best 10 individuals	0.001	0.002
Best individual	0	0
Individuals that reach the ground state	37%	100%

Table 4.4: Delta energy of the QGA with minimise update on 6 qubits on 500 generations

	final population	Best individuals during the process
Average of the total population	1.51	0.38
Average of the best 10 individuals	0.36	0.20
Best individual	0.0	0.0
Individuals that reach the ground state	40%	55.6%

Table 4.5: Delta energy of the QGA with minimise update on 8 qubits on 500 generations

	final population	Best individuals during the process
Average of the total population	5.33	2.17
Average of the best 10 individuals	2.03	2.02
Best individual	2.01	2.01
Individuals that reach the ground state	0%	0%

Table 4.6: Delta energy of the QGA with minimise update on 10 qubits on 500 generations

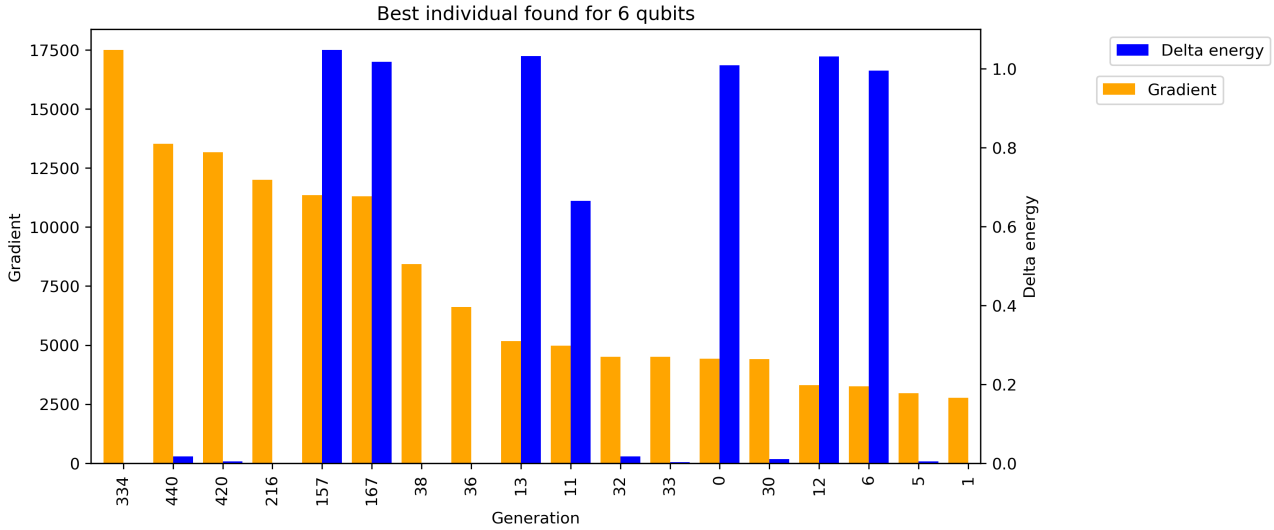
	final population	Best individuals during the process
Average of the total population	9.53	4.84
Average of the best 10 individuals	4.82	4.94
Best individual	4.01	4.03
Individuals that reach the ground state	1%	0%

4.4 Expressivity versus gradient results

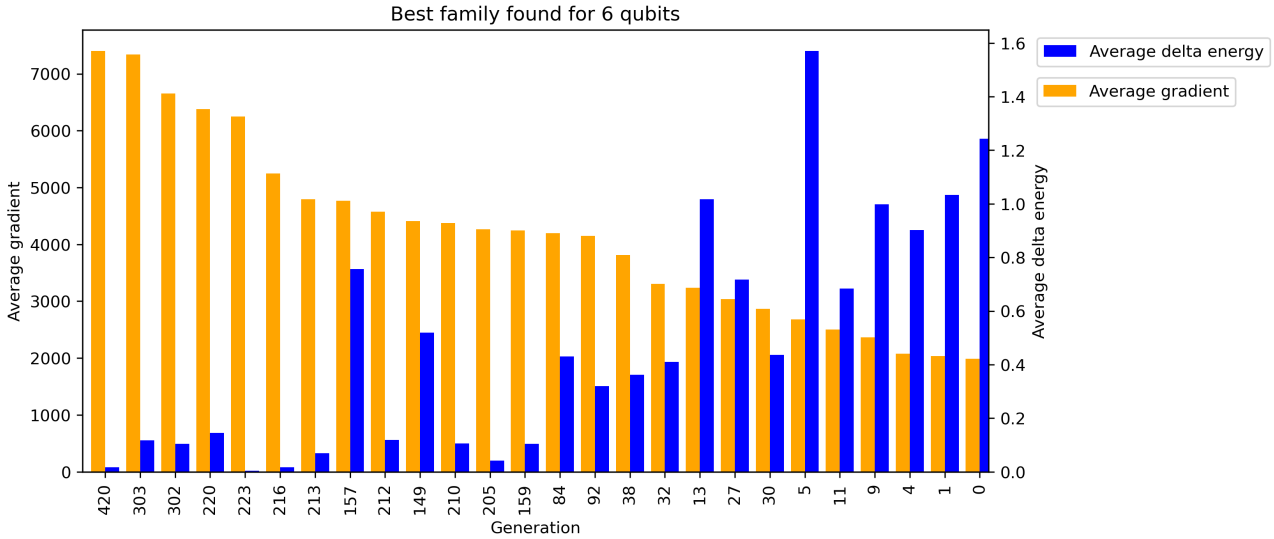
In Appendix 5.3 a large collection of plots can be seen that show the trade-off between expressivity and the gradient. In this section, we will highlight a few of those plots as examples of possible pitfalls or other insights into the managing of this trade-off.

The figures 4.1 are two insightful examples from the collection in Appendix 5.3. Figure 4.1a shows an unexpected occurrence that roughly a third of the best individuals improve upon the current-best fitness with an individual that does not meet the expressivity requirements. Therefore, the updated fitness function cannot distinguish between individuals that meet the requirements and individuals that do not reach the requirements as a more prominent gradient can overrule the lack of expressivity. Another insight is that in generations 334 and 440, the two leftmost bars, the performance decreases both in the gradient and the expressivity. This seems to contradict the earlier statement made in section 4.3 and indicates a loss of information from generation to generation. However, a distinction is that the optimisation of these graphs was not limited to only 100 steps and instead received no limitation in the number of steps. The number of steps might not have been enough steps to pass through a barren plateau that hinders optimisation but does prevent convergence entirely and could be a reason why figure 4.1a has this variance in expressivity. It could also simply be an anomaly caused by the gradient sampling variance and a worse error rate. Therefore in figures 5.5 and 5.6, a comparison will be made between the individuals in terms of the number of optimisation steps it takes before the PQCs optimum is found.

Figure 4.1 shows the evolution of the current-best families during the experiment. Because the results consist of the average values of the elitism section of the population the results show a much more consistent and robust evolution with signs of convergence yet, raising the suggestion that if the experiment would be continued for another couple of hundred generations the family would continue to improve. More so the anomaly seen for the best individuals in generations 334 and 440, evolve normally within a family perspective so measuring the data to gain insight into the evolution process should either be measured from a family perspective or averaged across multiple consecutive generations of the best individuals to anomalies within the data distorting the data and consequently raising false flags. Figure 5.6 show that the optimisation of the PQCs results in a gradual reduction of optimisation steps with the increase of fitness as expected. Following this, there is still improvement to be gained after the ground state has been reached by a single individual. However, the improvement is minimal. Evaluating the number of optimisation steps as a metric for the stopping criteria of the algorithm can be considered as this is a relevant measure of the effectiveness of the gradient.



(a) The gradient versus delta energy plot sorted on gradient descending for the best individuals in the GA using 500 generations and using COBYLA optimisation with a tolerance of 0.001



(b) The gradient versus delta energy plot sorted on gradient descending for the best families in the GA using 500 generations and using COBYLA optimisation with a tolerance of 0.001

Figure 4.1: The evolution of the best individuals and families showing the gradient versus the delta energy(expressivity)

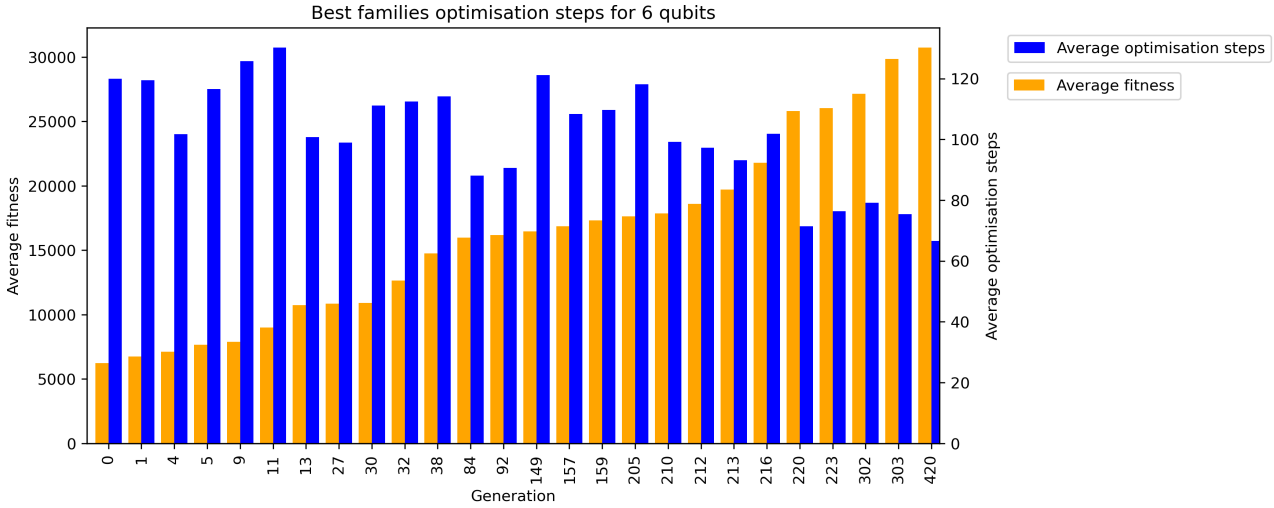


Figure 4.2: The evolution of the best families showing the number of optimisation steps compared to the fitness

Chapter 5

Outlook

5.1 Discussion

This thesis investigates the optimisation of parameterised quantum circuits by looking at two crucial factors: the gradient and the circuits' expressivity. Circuit optimisation is vital across domains such as machine learning and quantum chemistry. Vanishing gradients are detrimental to optimisation efficiency, while expressivity captures a circuit's ability to represent areas in the global search space. Understanding how gradient and expressivity relate is essential, as optimising one will impact the other. The choice of optimisation techniques influences expressivity and solution quality. Exploring gradient landscapes reveals convergence challenges and the influence of local and global optima. The fitness function of the algorithm plays a pivotal point in the optimisation process through the evaluation of individuals. Two variations were tested in the experiments showing that insights into a problem instance can easily be tested in the fitness function and can result in significant improvements. However, in chapter 4 it could be seen that managing multiple components is complex and often will result in a component overpowering the other components. Therefore, in the design of the algorithm, one should take this into account and consider implementing a minor component, like the error rate used in this thesis, in a genetic operator in the form of a heuristic. Furthermore, the gradient estimation component certainly found structures with improved gradients at initialisation. This implementation did, however, cause the evolution to solely focus on the gradient at initialisation. Possible solutions that might make a more general statement of the PQCs search space are an averaged gradient measured at multiple points during the optimisation steps and/or the use of a different metric for evaluating the gradient, like some form of random walk.

The expressivity requirements summarise the most common pitfalls for this type of genetic algorithm and should be the main focus regarding what data to gather during the GA process. Occurrences like family pools losing information and individuals reaching similar fitness values with dissimilar component values are signs that require investigation. As mentioned maintaining a balance between gradient and expressivity is crucial to reaching the global optimum. This does not mean that improving one worsens the other, instead, it refers to the fact that without its counterpart a high gradient does nothing and vice versa. Therefore, in the design of the genetic algorithm, it is not necessary to have both metrics in the fitness function and instead could be distributed across different genetic operators.

The goal of an experiment can also require different designs. For example, the modified uniform crossover used in this thesis enforced more diversity in structures but hindered convergence. For some problems, certain structures might be needed so specific heuristics about symmetry or a different gate set. Overall, the investigation has shown that genetic algorithms can be a suitable application for designing structures when the right set of circumstances are created in the design of the genetic algorithm.

5.2 Recommendations for future works

For future work in the field of evolving parameterised quantum circuits with the use of genetic algorithms, there are many yet-to-be-explored avenues. In this section, a list of possible interesting research topics will be listed based on the insights gained by working on this thesis as well as several intriguing avenues that could enhance the performance and applicability of this thesis' QGA. The following talking points outline some key areas worth investigating:

1. Exploring variations of Genetic Algorithms:
   To explore applications of evolving circuits using a GA, it is essential to experiment with more intricate variations of genetic algorithms. This could involve incorporating advanced genetic operators, such as crossover techniques inspired by biological phenomena like gene duplication or gene translocation. Additionally, investigating innovative selection strategies, like tournament selection or rank-based selection, could lead to improved convergence and diversity maintenance in the population. More so, the decision of what type of genetic operator to use should be heavily dependent on the heuristics used and implemented specifically for the application. For example, in this thesis, a modified uniform crossover was used to enforce structural diversity within the population pool. The implementation would have been less intuitive than if another genetic operator would was used.

2. Initialisation of individuals with preselected circuit structures:
   To target specific properties or characteristics in circuit design, one potential approach is to initialise individuals in the GA with preselected circuit structures. This allows for a more directed exploration of the search space, focusing on desired properties from the beginning. By leveraging prior knowledge and expertise, the GA can be tailored to tackle circuit design challenges efficiently and effectively. When trying this approach consider traits like symmetry and entanglement.

3. Utilising genetic algorithms to evolve PQCs in quantum reinforcement learning:
   An intriguing application of PQCs is their integration with reinforcement learning, this application integrates quantum technology in the form of PQCs within classical reinforcement learning. Dragan et al. compared the performance of 19 different PQCs layouts in a frozen lake environment with success[30]. In their study, the authors noted that it was difficult to establish metrics that could directly link to the performance of the PQCs. However, the metrics used in the paper, in combination with some tuning based on their reflection on those metrics, could be used in a fitness function of a genetic algorithm to try and evolve the PQCs. This would offer an exciting opportunity to explore the potential of evolving PQCs in solving reinforcement learning problems.

4. Parallel genetic algorithm:
   To address the computational demands associated with complex optimization problems, parallelising the genetic algorithm can significantly boost efficiency. Exploring techniques such as parallel evaluation of fitness functions or parallel population evolution can lead to substantial speed-ups. Additionally, investigating different variations of parallelised genetic algorithms, such as island models or cellular automata-based approaches, may offer further insights into optimising the computational process.

5. Landscape analysis of circuit structures and experimental data comparison:
   To validate the effectiveness of components used in the fitness function of the genetic algorithm in approximating the actual metrics of a PQC, a comprehensive landscape analysis can be conducted. By comparing the GA-generated circuits with experimental data, it becomes possible to assess the correlation between approximations and actual values. This analysis helps evaluate the reliability and accuracy of the GA's outputs, providing valuable feedback for further improvements and fine-tuning.

6. Optimisation for multiple objective areas in the global search space:
   The global search space often concentrates on a singular important area that requires distinct strategies and heuristics for optimisation. Employing the genetic algorithm to optimise for multiple objective areas in the search space simultaneously can be a promising avenue for future research. By incorporating multi-objective optimisation techniques like Pareto-based approaches or fitness assignment algorithms, it becomes possible to navigate and converge upon the diverse and complex landscape of the global search space.

7. Mid-circuit measurement:
   The use of non-linear methods like mid-circuit measurement could improve ansaetze as suggested by Rao et al.[31]. Therefore, it adding such methods to the gate set might allow for the generation of interesting circuits.

8. NEAT[32] for PQCs:
   Neuroevolution of Augmenting Topologies (NEAT) by Stanley and Miikkulainen is a method that evolves neural networks. What makes this method special is that it simultaneously both optimises and *complexifies* solutions simultaneously. It means that the structure of the network gradually increases in size and complexity during its evolution. The methods of NEAT could also be applied to PQCs with relatively little change as neural networks and PQCs hold numerous similarities. Furthermore, the arguments used by Stanley et al. as to why such a method would be effective hold true for the application of PQCs.

Exploring these recommendations in future work will contribute to the advancement and refinement of the research field, enabling their application in increasingly complex problem domains and enhancing their effectiveness in solving real-world challenges.

5.3 Conclusion

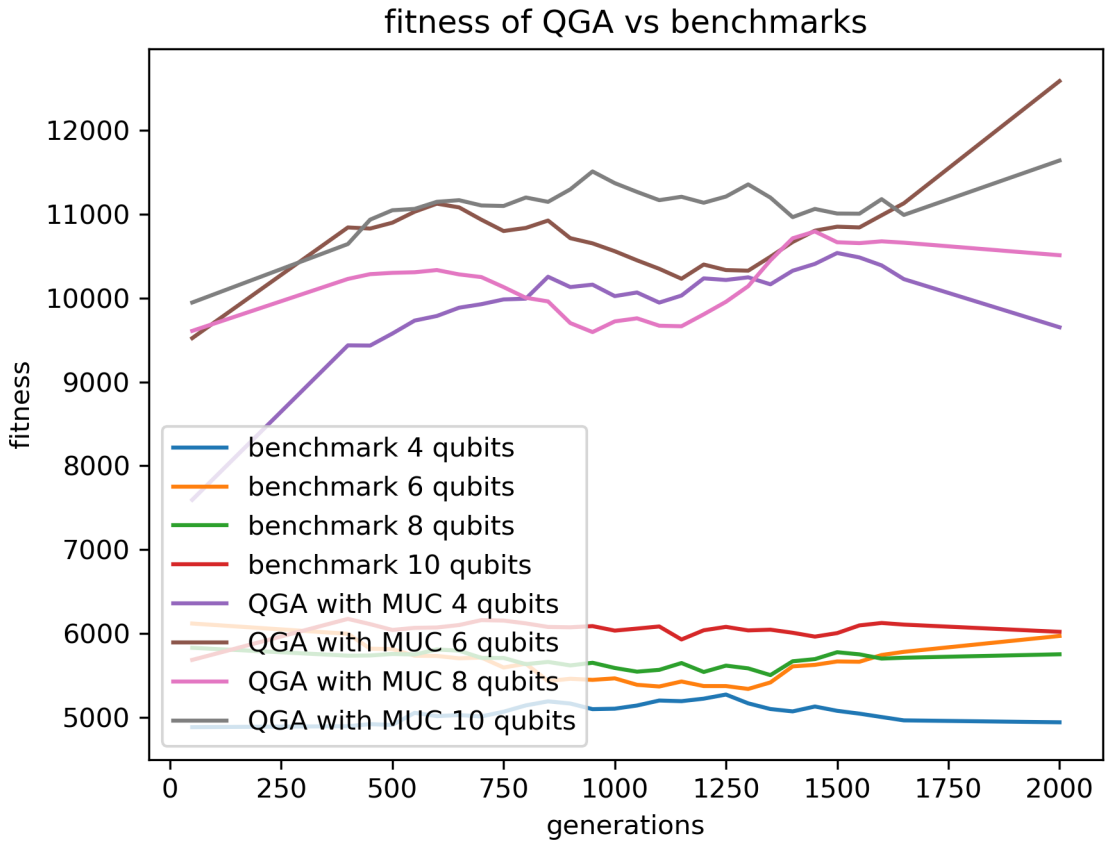
Barren plateaus are a prevalent issue for parameterised quantum circuits that occur when the number of qubits is scaled, imposing limitations on the effective training of PQCs. It is important to gain insight into ways to design circuits that reduce and/or avoid the impact of barren plateaus. In this thesis, genetic algorithms are investigated as a possible method of designing circuits that are less barren plateau prone. The results of the investigation, by constructing a GA to optimise the gradient of circuits, are two-fold. First, genetic algorithms are capable of evolving PQCs into circuits with improved gradients. However, the performance of a GA for our purpose depends on the problem task which requires specific heuristics depending on the task's goals. Nevertheless, genetic algorithms showcased the capability of producing successful circuits with just standard genetic operators. This indicates that even without specific knowledge of the optimisation landscape, GAs can be

used to a lesser extent. More so, our initial experiment showed that by using a vanilla genetic algorithm without applying any knowledge of the problem instance the algorithm was able to find an improvement. However, by considering expressivity the algorithm was able to find the ground state more consistently. Further knowledge of the optimisation landscape will likely result in additional improvements.

Second, when optimising for the gradient the expressivity requirements have to be taken into account for the resulting circuits to be able to reach the global optimum. When considering both statements the application of genetic algorithms to parameterised quantum circuits can play a vital role in the construction of quantum circuits. This would, however, require further research into improving the potency of GA through either change in the current structure like an improved and more sophisticated fitness function or a complete overhaul of the algorithm structure to, for example, a NEAT-like method.

A Appendix

A.1 Gradient performance



A.2 Expressivity plots

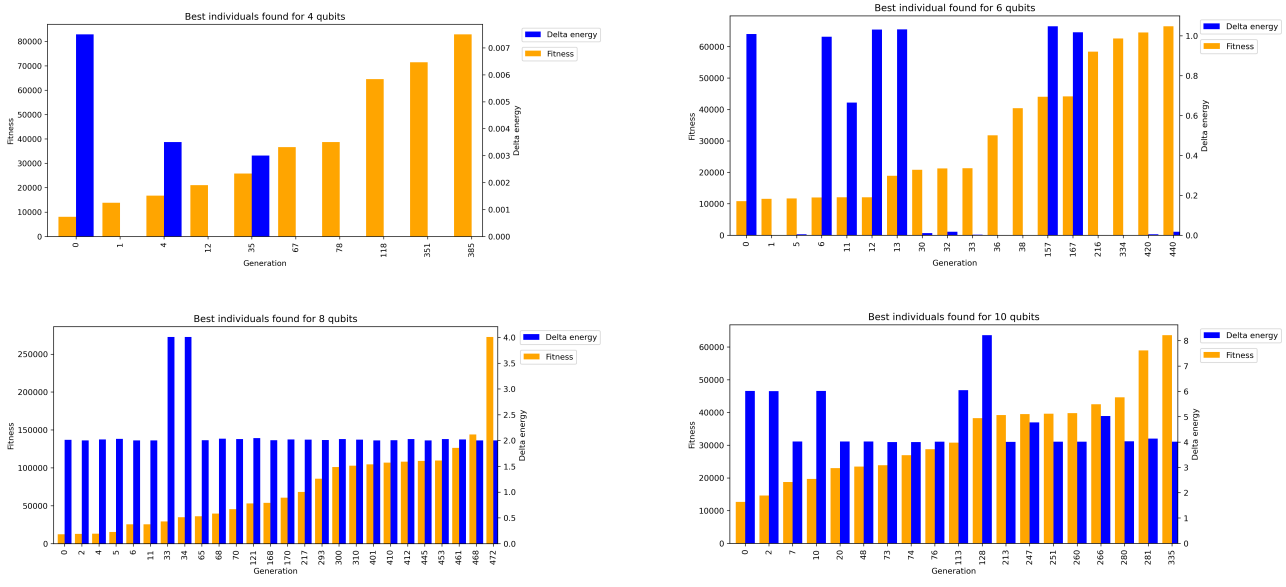


Figure 5.1: The delta energy between the ground state energy and the optimised energy of the best individuals found across the generations during the experiment in comparison to the fitness of the individuals.

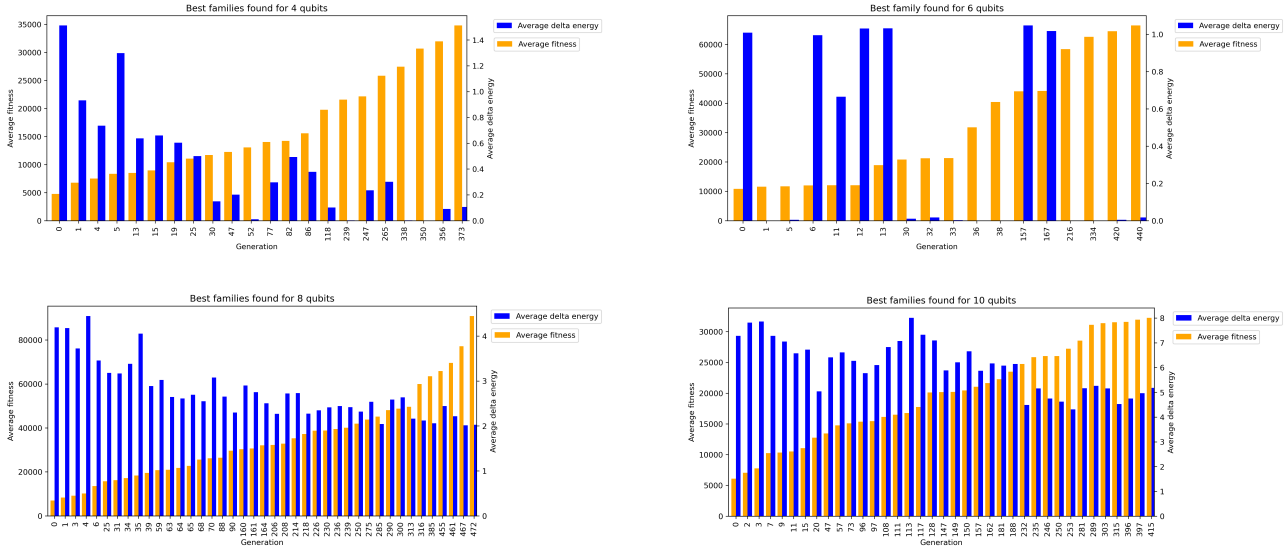


Figure 5.2: The average delta energy between the ground state energy and the optimised energy of the best families found across the generations during the experiment in comparison to the average fitness of the families.

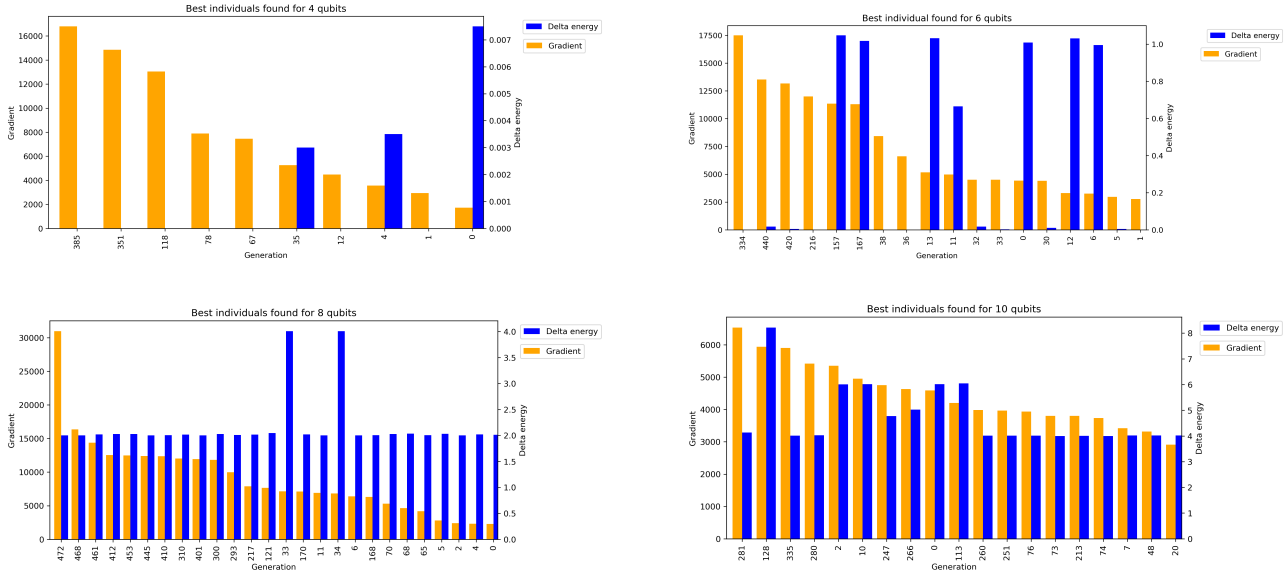


Figure 5.3: The average delta energy between the ground state energy and the optimised energy of the best individuals found across the generations during the experiment in comparison to the gradient of the individuals.

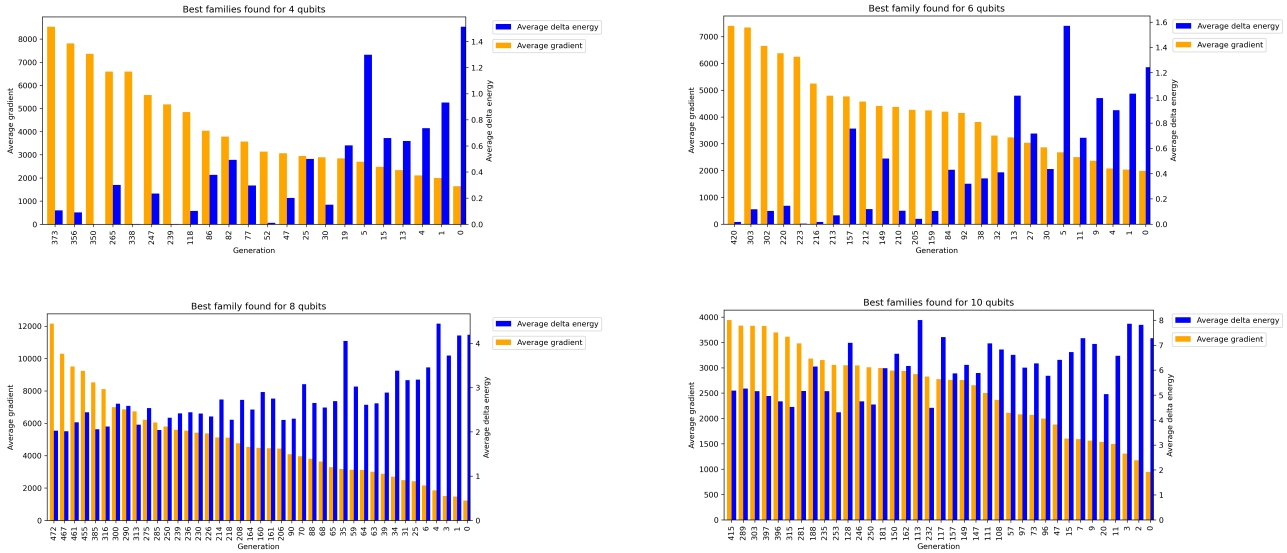


Figure 5.4: The average delta energy between the ground state energy and the optimised energy of the best families found across the generations during the experiment in comparison to the average gradient of the families.

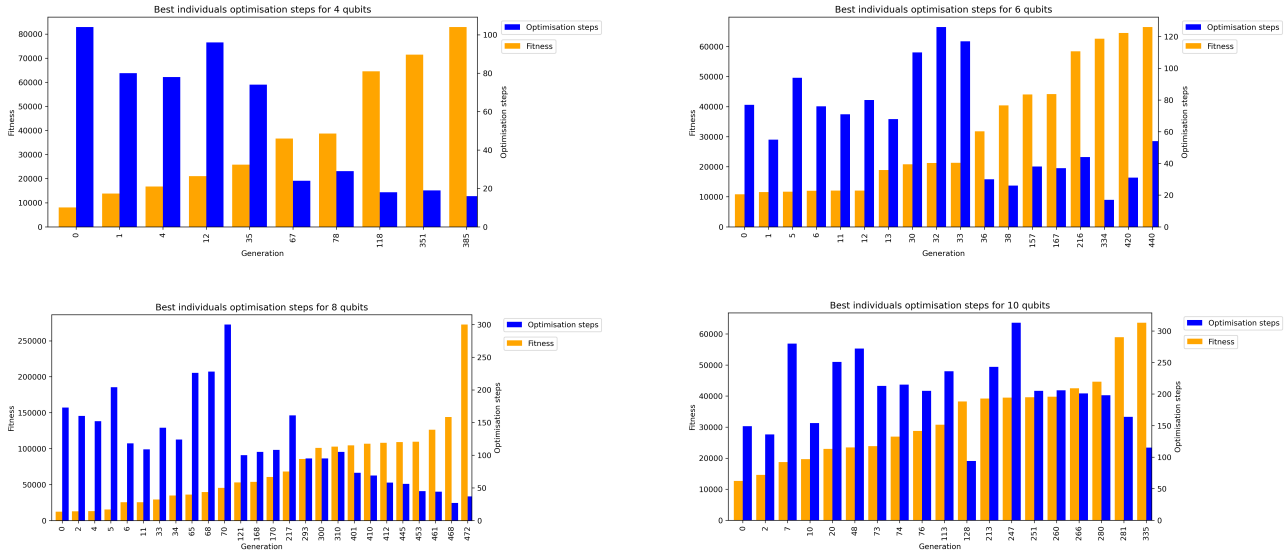


Figure 5.5: The number of optimisation steps is required before the optimum is reached in comparison to the fitness of the individuals.

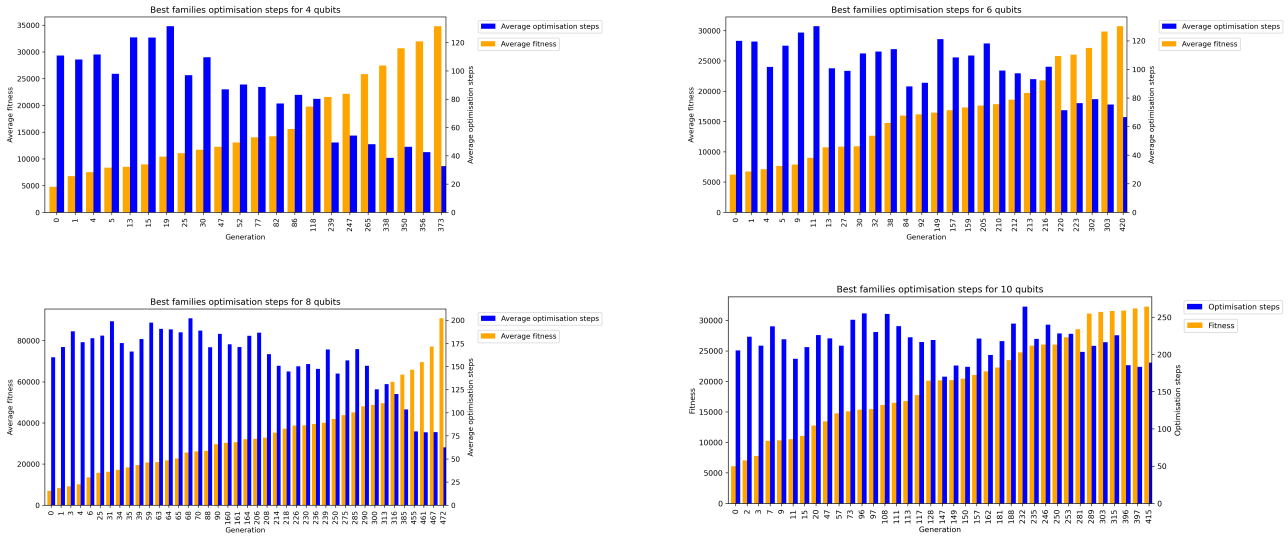


Figure 5.6: The average number of optimisation steps is required before the optimum is reached in comparison to the average fitness of the families.

B References

[1] D. Jaschke and S. Montangero, "Is quantum computing green? An estimate for an energy-efficiency quantum advantage," *Quantum Science and Technology*, vol. 8, p. 025001, Apr. 2023. arXiv:2205.12092 [quant-ph].

[2] "Quantum computing just might save the planet | McKinsey."

[3] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge ; New York: Cambridge University Press, 10th anniversary ed ed., 2010.

[4] X. Bonet-Monroig, "aQa-Lecture-3-vqe1.pdf."

[5] B. T. Kiani, S. Lloyd, and R. Maity, "Learning Unitaries by Gradient Descent," Feb. 2020. arXiv:2001.11897 [math-ph, physics:quant-ph].

[6] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, p. 4812, Nov. 2018.

[7] F. Busetti, "Genetic algorithms overview," *AIinfinance*, 2007.

[8] J. H. Holland, "Genetic Algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992. Publisher: Scientific American, a division of Nature America, Inc.

[9] G. Syswerda, "A Study of Reproduction in Generational and Steady-State Genetic Algorithms," in *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), vol. 1, pp. 94–101, Elsevier, Jan. 1991.

[10] D. Jakobović and M. Golub, "Adaptive Genetic Algorithm," *Journal of computing and information technology*, vol. 7, pp. 229–235, Sept. 1999. Publisher: Fakultet elektrotehnike i računarstva Sveučilišta u Zagrebu.

[11] J. Stender, *Parallel Genetic Algorithms: Theory and Applications*. IOS Press, 1993. Google-Books-ID: lkEXdxiArLsC.

[12] S. W. Mahfoud, *Niching methods for genetic algorithms*. Ph.D., University of Illinois at Urbana-Champaign, United States – Illinois, 1995. ISBN: 9798209041603.

[13] D. Tandeitnik and T. Guerreiro, "Evolving Quantum Circuits," Oct. 2022. arXiv:2210.05058 [quant-ph].

[14] R. Nichols, L. Mineh, J. Rubio, J. C. F. Matthews, and P. A. Knott, "Designing quantum experiments with a genetic algorithm," *Quantum Science and Technology*, vol. 4, p. 045012, Oct. 2019.

[15] D. Ribas Tandeitnik, *EVOLVING QUANTUM ERROR CORRECTION CODES*. MESTRE EM FÍSICA, PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO, Rio de Janeiro, Brazil, May 2022.

[16] K. Rambhatla, S. E. D'Aurelio, M. Valeri, E. Polino, N. Spagnolo, and F. Sciarrino, "Adaptive phase estimation through a genetic algorithm," *Physical Review Research*, vol. 2, p. 033078, July 2020.

[17] H. Wakaura, T. Tomono, and S. Yasuda, "Evaluation on Genetic Algorithms as an optimizer of Variational Quantum Eigensolver(VQE) method," Oct. 2021. arXiv:2110.07441 [quant-ph].

[18] T. Yabuki and H. Iba, "Genetic Algorithms for Quantum Circuit Design –Evolving a Simpler Teleportation Circuit–," *2000*, 2000.

[19] C. P. Williams and A. G. Gray, "Automated Design of Quantum Circuits," *1999*, 1999.

[20] P. Massey, J. A. Clark, and S. Stepney, "Evolving Quantum Circuits and Programs Through Genetic Programming," in *Genetic and Evolutionary Computation – GECCO 2004* (K. Deb, ed.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 569–580, Springer, 2004.

[21] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, "Cost function dependent barren plateaus in shallow parametrized quantum circuits," *Nature Communications*, vol. 12, p. 1791, Mar. 2021. Number: 1 Publisher: Nature Publishing Group.

[22] A. Arrasmith, M. Cerezo, P. Czarnik, L. Cincio, and P. J. Coles, "Effect of barren plateaus on gradient-free optimization – Quantum," Oct. 2021.

[23] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," *Physical Review A*, vol. 99, p. 032331, Mar. 2019.

[24] A. Pérez-Salinas, H. Wang, and X. Bonet-Monroig, "Analyzing variational quantum landscapes with information content," Apr. 2023. Number: arXiv:2303.16893 arXiv:2303.16893 [quant-ph].

[25] RichardMiddelkoop, "GCQS," May 2023. original-date: 2023-01-23T13:46:24Z.

[26] "Xenakis - The Automated Quantum Circuit Composer," Feb. 2023. original-date: 2022-07-31T10:45:53Z.

[27] "Molecular Quantum Solutions."

[28] E. Semenkin and M. Semenkina, "Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator," in *Advances in Swarm Intelligence* (Y. Tan, Y. Shi, and Z. Ji, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 414–421, Springer, 2012.

[29] K. Deb, A. Anand, and D. Joshi, "A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization," *Evolutionary Computation*, vol. 10, pp. 371–395, Dec. 2002.

[30] T.-A. Drăgan, M. Monnet, C. B. Mendl, and J. M. Lorenz, "Quantum Reinforcement Learning for Solving a Stochastic Frozen Lake Environment and the Impact of Quantum Architecture Choices," Dec. 2022. arXiv:2212.07932 [quant-ph].

[31] A. Rao, D. Madan, A. Ray, D. Vinayagamurthy, and M. S. Santhanam, "Learning hard distributions with quantum-enhanced Variational Autoencoders," May 2023. arXiv:2305.01592 [quant-ph].

[32] K. O. Stanley and R. Miikkulainen, "Evolving Neural Networks through Augmenting Topologies," *Evolutionary Computation*, vol. 10, pp. 99–127, June 2002. Conference Name: Evolutionary Computation.