Imperial College
London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Approximate Parameter Shift Rules for Variational Circuits

Author:
Adithya Sireesh

Supervisor:
Dr. Paul Bilokon

Submitted in partial fulfillment of the requirements for the MSc degree in MSc. Advanced Computing of Imperial College London

Abstract

In this work, we tackle one of the fundamental areas in Quantum Machine Learning: gradient based optimization of variational quantum circuits using the parameter shift rules. We go through the existing literature on the shift rules and its variants (Two-term [Mitarai et al. (2018)], Four-term, Gate Decomposition [Crooks (2019)], Stochastic [Banchi and Crooks (2021)] and General Parameter Shift Rules [Wierichs et al. (2022)]) which are used based on the specific forms of the Unitary Generators that make up the parametrised gates in the ansatz. We also show proofs for the two term and general parameter shift rules. Finally, to tackle the computational costs of the shift rules, we propose and run three toy experiments, and propose new methods to approximate the analytic gradients (from the shift rules), and even find that in two of the three cases our approximations converge to the solutions much faster than the analytic gradients from the shift rules.

Acknowledgments

I would first like to thank Dr. Paul Bilokon for his support throughout this project. The frequent meetings helped me stay on track and focus on bigger questions that could actually help further research in Quantum Machine Learning. Hence, in this report, we ended up tackling more fundamental questions in QML. This was very a hard task as we were both relatively new to quantum computing and quantum machine learning, and spent a good deal of time surveying what the current state of the field is, before settling on the topic of the parameter shift rules.

I would also like to thank my family. Without their constant support and dedication to my education, I wouldn't have been able to make it this far.

Also, I would also like to thank my friends and the members of Imperial's Quantum Tech Society. I have been very fortunate to be a part of this newly created society and its mission to bridge the gap between academia and industry at Imperial in the field of Quantum Computing. I was even given the opportunity to deliver their first workshop on Quantum Machine Learning, and it was a great experience.

I would also like to thank Michaela Brezinova for taking the time to read and reread my report and for giving her valuable feedback. Couldn't have done it without her!

Notation

Linear Algebra	
Symbol	**Description**
$\dagger$	Adjoint
$*$	Complex Conjugate
$\otimes$	Tensor Product
$\langle *\|*\rangle$	Inner Product
$[A, B]$	$= AB - BA$; Commutator of two matrices (which could be gates) A and B
δ_{ij}	Kronecker Delta;1 if $i = j$ else 0
$CNOT$	Controlled-NOT gate
H	Energy Hamiltonian of a quantum system
H	Hadamard gate
$H^{\otimes n}$	Hadamard gate on n qubits
$\mathbb{I}$	Identity gate
$\mathcal{G}(\mu)$	Parametrised unitary with parameter μ
R_{XY}, R_{YZ}, R_{ZX}	2 qubit Pauli rotation gates
$\mathcal{G}(\mu) = e^{-i\mu G}$	Parametrised Unitary with generator matrix$G$;
$\|\psi\rangle$	State of quantum system ψ
ψ_i	Component of state $\|\psi\rangle$ in the i^{th} eigenvector direction of an eigenbasis
$\|0\rangle^{\otimes n}$	n qubit quantum system/state in the 0 state
$A \otimes B$	Tensor product between 2 operators/unitaries
$\hat{Q}, \hat{B}$	A quantum observable
$\langle\hat{Q}\rangle$	Expectation value of an observable
σ_Z	Pauli Z Unitary or Observable
$Ansatz$	A parametrised quantum circuit
$E(\theta), f(\theta)$	Expectation value/cost function of an ansatz dependant of parameters θ
λ_{g_i}	i^{th} Eigenvalue of a unitary gate $\mathcal{G}$

Contents

Chapter 1

Introduction

Quantum Computing has been around for decades since 3 of the 4 fundamental quantum algorithms [Nielsen and Chuang (2002)] were proposed in the 90s. One of the first people to propose the idea of building a computer based on the laws of quantum mechanics was Richard Feynman [Nielsen and Chuang (2002)]. He was thinking about the problem of simulating quantum systems, and how the simulations got exponentially complex when being run on classical system (on general purpose computers that we see around us). His conclusion was that a much more efficient and accurate way to simulate these quantum systems would be by using another quantum system also known as. Quantum Computers. Since then, a lot of research was conducted to bring formalisms that would make it easier to think about what logical and physical components one would need to go about building these computers and algorithms associated with these computers.

Now, this area has started to get more widespread appeal and traction, partly due to improvements in hardware (such as GPUs) to simulate theoretically proposed quantum algorithms, and partly due to increased funding by Venture Capitalists and governments to build physical quantum computers. One of the main promises of Quantum computers is the ability to solve problems that may not have tractable solutions on classical computers (laptops, supercomputers, GPUs etc. that are built on the principles of classical mechanics). There are other conjectured benefits such as more energy and cost effective computations.

A part of the hype is also due to the well know Shor's algorithm (proposed around the mid 1990s) to factor large numbers in polynomial time: thus giving an exponential speedup over the best known classical algorithm. This plus the general misinformation and conspiracy theories that spread through social media has lead people to believe that we are very close to having quantum computers that are capable breaking systems built on cryptographic encryption schemes such as RSA and Diffie Helmann that are inturn built on the assumption that prime factoring is a problem in the complexity class NP. However, we are far from it. We are in an era known as the Noisy Intermediate Scale Quantum aka NISQ era. Quantum computers, and ways to manipulate the quantum bits (qubits) on these quantum computers using quantum gates (similar to the AND, OR, NOT etc. gates we see on classical computers) exist, but it is still not easy to keep these systems stable enough to perform quantum gate operations without large errors. But, we are getting there! The end goal is to reach full Fault Tolerence where these qubits are stable enough and error corrected so that we can perform useful computations and get the exact output we expect at the end of the circuit execution. This doesn't mean that there are no uses of quantum computers where we currently stand. Companies are already starting to use quantum computers to optimize their processes. Volkswagen tested out D-Wave's quantum annealer to find approximately efficient routes for their delivery routes (need to double check this). Quantum Computing Inc. is providing business solutions for constrained optimization problems. Startups such as Menten AI and Multiverse Computing are building algorithms for protein design and financial application (Eg: options pricing) respectively. There are hundreds of companies focusing on various levels of the quantum technology stack from the hardware/compiler side to the application/algorithms side, and now is as good a time as ever to get into this area as

there are so many interesting problems yet to be solved.

One of the up and coming areas of research in QC, and also the main focus of this report focus on how quantum computing can extend our current machine learning methods (Eg: Kernel Methods as a lot of ML operations such as inner products appear naturally in quantum mechanics) and how principles of Quantum Computation can help build different classical Machine Learning Models (Eg: Tensor Networks based on tensor products). There are also interesting proposals for extending quantum computing with machine learning (eg: Variational Quantum Eigensolvers). Extensive research of QML algorithms has opened the door for more general Variational Algorithms which take on a hybrid approach where a part of the computation is done on a quantum computer and the post processing is performed on a classical computer. This helps keep quantum circuits short, thus attempting to tackle the problem of decoherence due to large quantum circuits on NISQ devices as mentioned before. Finally, research and experiments are also being done to see how machine learning and quantum computing algorithms can be used as subroutines in larger applications. In this report, we study a set of methods know as the parameter shift rules [Mitarai et al. (2018)], that specifically allow gradient based optimization on real quantum hardware. The study of quantum specific optimization methods is necessary as traditional methods such as finite differences, automatic differentiation with back propagation etc. fall short in a real quantum setting as we will see in the coming chapters. Finally, we provide our own analysis for potential ways to approximate these quantum gradients to allow for more cost effective optimization of variational circuits.

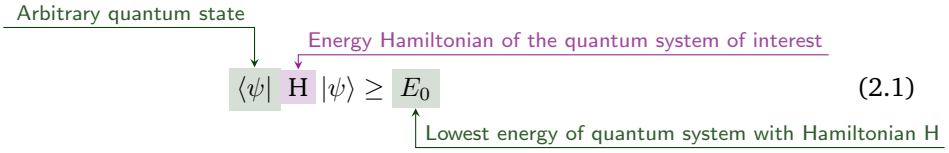
Chapter 2

Background

This chapter will first introduce some Quantum Machine Learning Concepts, then draw some parallels between QML and ML, and finally take a look at existing optimization approaches for QML problems with a specific focus on the Parameter Shift Rules. The reader is encouraged to refer Nielsen and Chuang (2002) for more details on Quantum Computing Fundamentals

2.1 Quantum Machine Learning Fundamentals

In Quantum Mechanics, physicists and chemists often find themselves solving problems that involve computing the ground state or ground state energy of a quantum system. If they try to test all possible state configurations to see which one corresponds to the lowest energy, the issue is that the domain of all possible quantum states a system could be in is infinite. The Ritz method for computing solutions to boundary value problems is one way to tackle such problems. The main observation to make is that for any arbitrarily chosen quantum state ψ, the energy corresponding to the state follows the inequality



(2.1)

Eq 2.1 tells us that the energy corresponding to any chosen state of the system is an upper bound to the ground state energy of the quantum system. We need to find a way to smartly pick trial states to compute the energy of the system. This motivates the need to parametrise these quantum states, and convert the problem to a problem of optimization of a cost function, where, the cost function corresponds to the expectation value/energy $E(\theta)$ of the parametrised state $\psi(\theta)$. The sections below talk about how to construct these parametrised quantum states, and methods to optimize on their expectation values.

2.1.1 Parametrised Circuits

Quantum Circuits could have gates with fixed values (Eg: Hadamard, Pauli-X, Pauli-Y, Pauli-Z, CNOT), or gates that are Parametrised (Eg: $R_X(\theta)$, $R_Y(\theta)$, $R_Z(\theta)$). Some very important applications of parametrised circuits include, but are not limited to, Variational Quantum Eigensolver for finding the lowest energy eigenstate of a Hamiltonian[1] (very useful for quantum chemistry simulations),

[1] A Hamiltonian, denoted by a matrix H, describes the energy and evolution of a quantum system. The ground state would correspond to the state with the lowest eigenvalue of the Hamiltonian

Quantum Approximate Optimization Algorithm (QAOA)[Farhi et al. (2014)] for finding approximate solutions to combinatorial optimizations (generally NP-hard to find exact solution in the classical case) and Quadratic Unconstrained Binary Optimization (QUBO) problems. The output of the cir-

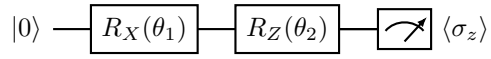


Figure 2.1: General anatomy of a parametrised quantum circuit. The above circuit applies to a single qubit 0 state a Pauli-X rotation followed by a Pauli-Z rotation gate. Finally, we measure in the Pauli-Z operator's basis

cuit can be any function of the measurements. Generally, we compute the expectation value of the measurements. Therefore, the circuit needs to be run for a number of shots to compute anything meaningful. The most general set of problems that involve the use of parametrised quantum circuits are framed as "*find the ground state or ground state energy of a system, given that it's total energy is described by the Hamiltonian H*".

2.1.2 Dissecting the Cost Function

Let use consider a toy problem of preparing a ground state of the Pauli-Z (σ_z) Observable. We know that the ground state should correspond to eigenvalue -1 (as all Pauli observables $\{\sigma_i\}_{i\in\{x,y,z\}}$ have eignevalues ± 1).
Therefore, the problem can be formulated as a minimization problem where we are trying to minimize the expected value of the σ_z measurement of some parametrised quantum circuit.

$$\theta^* = \arg\min_{\theta} \overbrace{\langle 0| U^\dagger(\theta)\sigma_z \underbrace{U(\theta)|0\rangle}_{\text{State prepared with } \theta \text{ params}}}^{\text{Expectation value of the observable}} \tag{2.2}$$

Using the example of the circuit in fig 2.1, we have to solve

$$\theta_1^*, \theta_2^* = \arg\min_{\theta_1,\theta_2} \langle 0| R_x(\theta_1)^\dagger R_z(\theta_2)^\dagger \sigma_z R_z(\theta_2) R_x(\theta_1) |0\rangle \tag{2.3}$$

Now, this boils down to an optimization problem where we are trying find parameters that minimize Eq 2.2. Various learning/optimization methods will be reviewed in the next section based on the conditions in which these computations performed (simulator vs real hardware, gradient-free vs gradient-based optimization etc.).

2.2 Optimization Methods for Parametrised Quantum Circuits

As mentioned before, the optimization method of choice would depend on the type of device we are performing our computations on.

2.2.1 Optimization on Classical Simulators

The first question to ask is, why should we even run our computations on a classical simulator of a quantum computer? Current day quantum computers are not powerful enough yet to handle big experiments without succumbing to noise. On the other hand, GPUs have become a lot faster at matrix operations (which is the main kind of operations that are involved in Quantum Mechanics). Using the power of GPUs, we would be able to test the quantum algorithms much faster and cheaper. As Quantum Computers get better, the speed-up offered by real quantum hardware is expected to far outperform the classical simulations run on GPUs. However, until we reach that point, bigger quantum models can easily be built and tested without having to rely on good quantum computers.

When using classical simulators, methods such as automatic differentiation with back propogation, symbolic differentiation and adjoint differentiation [Jones and Gacon (2020)] are a 3 out of many methods that we can use (in addition to the algorithms we can run on real quantum hardware which will be introduced later). The reason backpropogation is highlighted here is because, on real hardware, we do not have access to the intermediate computations of the quantum computer. We only have access to the final measurement. Hence, it would be very hard to use backpropogation on real quantum hardware. However, in the case of a simulator, we can keep track of all the computations and make use of them in our gradient computations.

Adjoint Differentiation [Jones and Gacon (2020)], which works efficiently only on classical simulators, exploits the fact that a quantum circuit involves gates which are constrained to be Unitary matrices. Since a qunum circuit is a series of gates applied to a quantum state (or system), we can undo any of these gates by applying their corresponding adjoint. This method involves directly taking derivates of the unitaries gates, and changing the circuit by replacing each unitary with it's corresponding derivative. This wouldn't work on real hardware because it is not certain that the derivate of a parametrised gate would also be unitary.

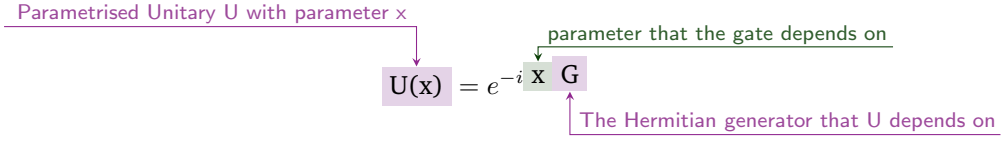
2.2.2 Optimization on Real Quantum Hardware

Gradient free methods such as COBYLA, Neadler Meade, Kernel Methods (exploiting the fact that high dimensional inner products can be easily computed on quantum hardware and is a basic property in quantum mechanics) and gradient based methods such as SPSA, Finite Differences, Parameter Shift etc. have been previously proposed as optimization algorithms for quantum computers. Gradient based methods are preferred over gradient free methods due to theoretical guarantees for convergence of the algorithms. The parameter shift rules are analogous to the finite differences method, however, since we are working with NISQ devices, the finite difference lacks in this respect because the noise of quantum computations has a higher influence on the expectation values (output of the circuit) for small changes in the parameters. This prevents us from calculating any useful descent directions (let alone gradients) for gradient descent. The parameter shift rules, while still involving shifting of parameters to compute gradients, tackle this problem by providing a method for computing exact gradients of the parameters. The rest of the report is dedicated to a review of the existing parameter shift rules and why they work.

2.3 Parameter Shift Rules

2.3.1 Two Term Parameter Shift Rule

The parameters shift rule was first introduced in Mitarai et al. (2018). The key benefit of the parameter shift rules is that the same circuit we are trying to optimize can also be used for the gradient computation. By evaluating the quantum circuit at n different points for each circuit (based on certain rules), we can compute a gradient that is exact in theory. The first versions of the parameter shift rules were only applicable to parametried unitaries with Pauli Generators i.e. the Pauli Rotation gates ($R_X(\theta)$, $R_Y(\theta)$, $R_Z(\theta)$)



(2.4)

Figure 2.2: The type of unitaries that Mitarai et al. (2018)'s parameter shift rules worked on

The rules by Mitarai et al. (2018) were extended further by Schuld et al. (2019) for variational circuits with unitaries formed of hermitian generators with at most 2 unique eigenvalues

The proof below has been adapted from Schuld et al. (2019) with some modifications and explanations for completeness

Output of a variational circuit — Measurement observable — Full unitary describing the variational circuit

$$f(\theta) = \langle 0| U(\theta)^\dagger \hat{Q}\ U(\theta) |0\rangle \tag{2.5}$$

We can decompose U into $VG(\mu)W$

Now, we will compute the derivative of the cost (expectation $f(\theta)$) with respect to a single parameter μ. As mentioned in Eq 2.5, we can decompose $U(\theta)$ (into $VG(\mu)W$) by isolating the parametrised gate ($G(\mu)$) we want to differentiate and absorb V and W into the $|0\rangle$ state and observable $\hat{Q}$ respectively

$|\psi\rangle = W|0\rangle$ — $B = V^\dagger \hat{Q} V$

$$f(\theta) = \langle\psi|\, \mathcal{G}^\dagger(\mu)\, B\, \mathcal{G}(\mu) |\psi\rangle$$

This term is the hermitian conjugate (h.c.) of the 1st term

$$\partial_\mu f = \langle\psi|\, \partial\mathcal{G}^\dagger B \mathcal{G} |\psi\rangle + \overbrace{\langle\psi|\, \mathcal{G}^\dagger B \partial\mathcal{G} |\psi\rangle} \tag{2.6}$$

For any 2 operators C, D:

$$\begin{aligned}\langle\psi|\, C^\dagger \hat{Q} D |\psi\rangle + h.c. =& \frac{1}{2}(\langle\psi|\, (C+D)^\dagger \hat{Q}(C+D) |\psi\rangle \\ & - \langle\psi|\, (C-D)^\dagger \hat{Q}(C-D) |\psi\rangle)\end{aligned} \tag{Identity 1}$$

Comparing Identity 1 with Eq 2.6, we can see that if $G \pm \partial G_\mu$ is unitary, then we can modify the circuit slightly to compute the derivate w.r.t a parameter μ.
Any matrix ($\mathcal{G}(\mu)$) where $\mathcal{G}(\mu) = e^{-i\mu G}$ is Unitary if G is Hermitian. Its derivative is therefore given

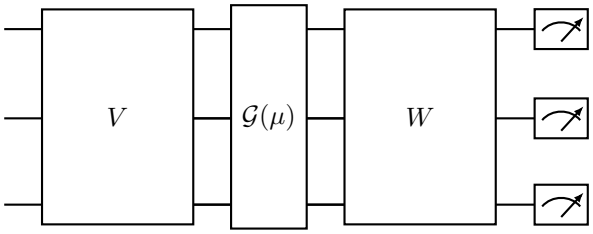


Figure 2.3: The ansatz with the parametrised gate $\mathcal{G}(\mu)$ isolated

by Eg: 2.7

$$\partial\mathcal{G}_\mu = -iGe^{-i\mu G} \tag{2.7}$$

Using Eq 2.7 in Eq 2.6

$|\psi'\rangle = \mathcal{G}|\psi\rangle$

$$\partial f_\mu = \langle\psi'|\, B(-iG) |\psi'\rangle + h.c. \tag{2.8}$$

Given G is a hermitian generator with 2 unique eigenvalues ($\lambda_{g1}, \lambda_{g2};\ \lambda_{g1} < \lambda_{g2}$), we can assume that the eigenvalues are symmetric with respect to 0 ($\lambda_2, \lambda_1 = \pm r$). This is equivalent to applying a global shift to make the eigenvalues of the Generator symmetric. The shift can be done by multiplying by $\mathcal{G}$ by $e^{-i\mu\frac{-(\lambda_{g2}+\lambda_{g1})}{2}\mathbb{I}}$, thus giving us a generator ($G \longrightarrow G - \frac{(\lambda_{g2}+\lambda_{g1})}{2}\mathbb{I}$) with symmetric eigenvalues $\pm r$ where $r = \frac{(\lambda_{g2}-\lambda_{g1})}{2}$. Doing this doesn't change the cost function in any way as it just adds a global phase to the final state (which has no observable effects **Theorem** 3). Hence, it is valid to assume that the eigenvalues are symmetric.
Now, using Identity 1 on Eq 2.9 with $C = \mathbb{I}\ \&\ D = -ir^{-1}G$

$$\partial f_\mu = \frac{r}{2}(\langle\psi'|\,(\mathbb{I} - ir^{-1}G)^\dagger B(\underbrace{\mathbb{I} - ir^{-1}G}_{=C+D.\text{ We will prove that this is Unitary}})\,|\psi'\rangle - \langle\psi'|\,(\underbrace{\mathbb{I} + ir^{-1}G}_{=\text{C - D. We need to prove this is also a unitary}})^\dagger B(\mathbb{I} + ir^{-1}G)\,|\psi'\rangle) \tag{2.9}$$

Using **Theorem** 1, we know that if $\mathcal{G}(\mu) = e^{-i\mu G}$ with G having exactly 2 distinct eigenvalues $\pm r$, then $\mathcal{G}(\pm\frac{\pi}{4r}) = \frac{1}{\sqrt{2}}(\mathbb{I} \mp ir^{-1}G)$
The above statement implies that we can compute $\partial_\mu f$ by applying the gate $\mathcal{G}(\pm\frac{\pi}{4r})$ right after the parametrised gate (with parameter μ) we wish to differentiate.
Also, using the fact that if $[A, B] = 0$, then $e^A e^B = e^{A+B}$, we can see that the circuit with the gates $\mathcal{G}(\mu)$ and $\mathcal{G}(\pm\frac{\pi}{4r})$ next to each other has the property that

$$\begin{aligned}\mathcal{G}(\mu)\mathcal{G}(\pm\tfrac{\pi}{4r}) &= \underbrace{e^{-i\mu G}e^{-i(\pm\frac{\pi}{4r})G}}_{[G,G]=0} \\ &= e^{-i(\mu\pm\frac{\pi}{4r})G} \\ &= \underbrace{\mathcal{G}(\mu \pm \tfrac{\pi}{4r})}_{\mathcal{G}\text{ with }\mu\text{ shifted by }\pm\frac{\pi}{4r}}\end{aligned} \tag{2.10}$$

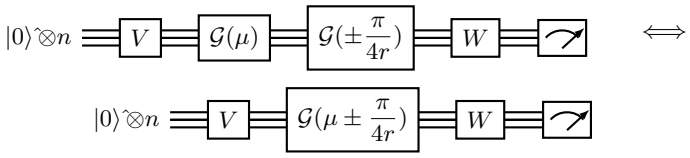


Figure 2.4: Parameter Shift rule equivalence with shift $s = \frac{\pi}{4r}$

Therefore, we can finally write the gradient of the cost function $\partial f(\mu)$ (wrt μ) in terms of evaluations of the cost function at 2 shifted values

Cost fn with paramater μ shifted

Shift parameter s

$$\begin{aligned}\partial_\mu f =& r(\, f(\mu + s) \, - f(\mu - \, s \,)) \\ =& r(f(\mu + \frac{\pi}{4r}) - f(\mu - \frac{\pi}{4r}))\end{aligned} \tag{2.11}$$

2.3.2 Four Term Parameter Shift Rule

Other variants of the parameter shift rule exist based on the specificnumber of eigenvalues in the unitary $\mathcal{G}'s$ hermitian generator G. The 4 term parameter shift rule can be used when the gate generator has eigenvalues $\{-1, 0, 1\}$. Without going in to the proof, this shift rule has the form:

depends on x_1 and μ

Arbitrarily chosen shift param

$$\partial_\mu f = y_1 \, (f(\mu + \, x_1 \,) - f(\mu - x_1)) + y_2 \, (f(\mu + \, x_2 \,) - f(\mu - x_2)) \tag{2.12}$$

depends on x_2 and μ

Arbitrarily chosen shift param

2.3.3 Parameter Shift Rule with Gate Decomposition

If the gate that we are working with can be decomposed into a product of unitaries with 2 or fewer eigenvalues, then we can use the two-term parameter shift rule for each of the gates in the decomposition, and then use the chain rule to get the final derivative w.r.t. the parameter [Crooks (2019)]. Crooks (2019) uses the observation that any 2 qubit unitary can be decomposed into a product of single qubit gates and the Canonical gate (U_{CAN}):

$$U_{\mathrm{CAN}} = \exp\left(-i\frac{\pi}{2}\left(t_x X \otimes X + t_y Y \otimes Y + t_z Z \otimes Z\right)\right) \tag{2.13}$$

The gradients of the original parameter can be computed by applying the parameter shift rule on the decomposed gates (as they would all have 2 eigenvalues), and then combining the decomposed gradients using the chain rule of differentiation. The main downside is that this method scales very poorly 2*(number of gate in decomposition)

2.3.4 Stochastic Parameter Shift Rule

The Stochastic Parameter Shift Rules [Banchi and Crooks (2021)] work on unitaries with no restriction on the generators that they are composed of. The parametrised unitary we are trying to differentiate is replaced by 3 gates, and we obtain the gradient of interest by computing the modified ansatz's expectation value dependant on a uniformly generated variable s. See Banchi and Crooks (2021) for more details.

2.3.5 General Parameter Shift Rules

The general parameter shift rules [Wierichs et al. (2022), Izmaylov et al. (2021), Kyriienko and Elfving (2021)] give us a set of rules that extend the type of gates the original parameter shift rules target. The main idea revolves around the fact that the variational cost function (expectation value), when viewed in terms of a single parameter boils down to a finite term Fourier series. The idea then is to use evaluations of the quantum circuit at various points to find the Fourier Coefficients and reconstruct the Fourier Series. Now, the gradients can be computed through closed form. Another benefit of using the general parameter shift rules is that computing higher derivatives becomes very cheap (as we have the full form of the variational cost function in terms of the parameter of interest).

The Fourier Series form will be derived below:

We assume a unitary $\mathcal{G}(\mu) = e^{i\mu G}$ to be part of a bigger ansatz. Similar to the previous proof, we will decompose the ansatz into three unitaries $V\mathcal{G}(\mu)W$. Using **Theorem** 2 , if the eigenvalues of G are $[\omega_j]_{j\in[1..d]}$ (where $x < y \implies \omega_x \leq \omega_y$), then the eigenvalues of $\mathcal{G}(\mu) = e^{i\mu G}$ are $\{e^{i\mu\omega_j}\}_{j\in[1..d]}$. The cost function wrt the parameter μ is

Decomposition in $\mathcal{G}(\mu)$'s eigenbasis. $\sum_{i,j=1}^{d} b_{ij} |g_i\rangle \langle g_j|$ $|\psi\rangle = \sum_{i=1}^{d} \psi_i |g_i\rangle$; Decomposition using $\mathcal{G}(\mu)$'s Eigenbasis

$$
\begin{aligned}
f(\mu) &:= \langle\psi| \mathcal{G}^\dagger(\mu) \hat{B} \mathcal{G}(\mu) |\psi\rangle \\
&\quad \text{(Has eigenbasis } \{|g_i\rangle\}_{i\in[1..d]}\text{)} \\
&:= \sum_{j,k=1}^{d} \overline{\psi_j e^{i\omega_j\mu}} b_{jk} \psi_k e^{i\omega_k\mu} \\
&:= \sum_{j<k}^{d} [\overline{\psi_j} b_{jk} \psi_k e^{i(\omega_k-\omega_j)\mu} + \psi_j \overline{b_{jk}\psi_k e^{i(\omega_k-\omega_j)\mu}}] + \sum_{j=1}^{d} b_{jj} \|\psi_j\|^2
\end{aligned}
\tag{2.14}
$$

Naively using the parameter shift rule with gate decomposition, we may end up with $2*$*(number of gates in decomposition)* evaluations to get the final gradient $\partial_\mu f$. However, we can do better in certain cases. In Eq 2.15, the exponential powers are determined by the differences between the eigenvalues $\{w_k - w_j\}_{j<k}$ of the generator G. First, we gather the set of unique eigenvalue differences of G: $\{\Omega_\ell\}_{\ell\in[R]} := \{\omega_k - \omega_j \mid j,k \in [d], \omega_k > \omega_j\}$. Now, we collect the coefficients $c_{jk} := \{\overline{\psi_j} b_{jk} \psi_k\}$ of the exponentials $\{e^{i(\omega_k-\omega_j)\mu}\}$ with common differences into coefficient c_l.

For the set of r distinct eigenvalues of G, there can be at most $R < \frac{r(r-1)}{2}$ unique differences. These differences may not be equidistant integer multiples, however, as noted in Wierichs et al. (2022), they are indeed equidistant for sums of commuting Pauli Strings/Words [2] ($G = \sum_{i=1}^{p} \pm P_i$ where $P_i = \{\sigma_x, \sigma_y, \sigma_z, \mathbb{I}\}^{\otimes n}$).

zero frequency term $= \sum_{j,k=1}^{d} \delta_{\omega_j\omega_k} b_{jk} \overline{\psi_j} \psi_k$

$$
\begin{aligned}
f(\mu) &= a_0 + \sum_{\ell=1}^{R} c_\ell e^{i\Omega_\ell\mu} + \sum_{\ell=1}^{R} \overline{c_\ell} e^{-i\Omega_\ell\mu} \\
&\quad \left(c_\ell = \Sigma_{\{(\omega_k-\omega_j)=\Omega_\ell\}} \overline{\psi_j} b_{jk} \psi_k\right) \\
&= a_0 + \sum_{\ell=1}^{R} a_\ell \cos(\Omega_\ell\mu) + b_\ell \sin(\Omega_\ell\mu) \\
&\quad \left(a_\ell = (c_\ell + \overline{c_\ell})\right) \qquad \left(b_\ell = i(c_\ell - \overline{c_\ell})\right)
\end{aligned}
\tag{2.15}
$$

The above equation is a finite term fourier series. Through evaluations of the quantum circuit at 2R + 1 points of the parametrised gate $\mathcal{G}(\mu)$, we get a set of linear equations that can be solved to finally get the coefficients $a_0, \{a_\ell\}_{\ell\in[1..R]}, \{b_\ell\}_{\ell\in[1..R]}$. Assuming that the frequencies are equidistant i.e. $\Omega_\ell = \ell\Omega$, we get a system of linear equations:

$$
f(x_\mu) = a_0 + \sum_{\ell=1}^{R} a_\ell \cos(\ell\Omega x_\mu) + b_\ell \sin(\ell\Omega x_\mu) \,; \mu \in [0..2R] \tag{2.16}
$$

If the chosen evaluation points $\{x_\mu\}_{\mu\in[0..2R]}$ are equidistant, the solution to Equations 2.16 is the *Discrete Fourier Transform*

[2] Pauli words the most widely use set of generators, so it makes sense to come up with rules that can easily differentiate these gates

A set of experiments were done to show the trigonometric polynomial (fourier series) form of the cost function with respect to a single parameter μ in the ansatz. Four ansatz's were prepared with 1, 2, 4 and 8 qubits. An R_Z gate was applied to each qubit in the 4 ansatz's and the dependance of the cost function on the parameter μ as well as the ansatz are shown below in figures 2.6 and 2.5 respectively.

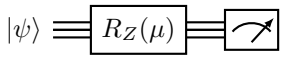


Figure 2.5: The family of circuits used to show the fourier series form in figure 2.6. Four anstaz's were created with 1, 2, 4 and 8 qubits.

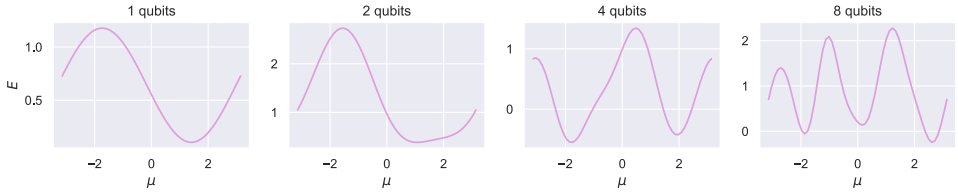


Figure 2.6: The dependence of the parameter μ was graphed over the period $[-\pi, \pi]$. We can see that each of these functions are composed of trignometric functions. In other words, these graphs represent the Fourier Series form of the cost function $f(\mu)$ with respect to the parameter μ for ansatzs with varying number of qubits (1, 2, 4, 8). The circuits are composed of single R_Z gates applied to each qubit all dependent on the same parameter μ [Bergholm et al. (2018)]

Chapter 3

Experiments and Analysis

Through numerical analysis, in this chapter, we propose certain gradient approximation methods to counter the computational costs of the general parameter shift rules

In the previous chapter, we have seen how to compute gradients of parametrised unitaries using the two-term parameter shift rules. However, these rules were restricted to gates with generators that have atmost 2 unique eigenvalues. We also saw ways around this 2 eigenvalue restriction based on the four-term parameter shift rule and the parameter shift rule using gate decomposition. Finally, a more general set of rules (General Parameter Shift Rules) were reviewed, that work for generators with arbitrary eigenvalues. However, the general parameter shift rules fall short in a couple of areas. They scale very poorly, at the rate of 2R (where R is the number of unique eigenvalue differences of the generator), and we have to finally reconstruct the Fourier series using these 2R + 1 evaluations to get the full representation of the cost function in terms of the parameter of interest. The natural question now is, can we do better?

3.1 Approximating the analytic gradients

One reason why the two-term parameter shift rules don't work for gate generators G with more than 2 eigenvalues is because we cannot reduce the gradients of these gates to a nice analytic form that satisfies equation 2.9. We can, however, analyse how closely the gradients computed using the two-term parameter shift rules (for > 2 eigenvalue generators) approximate the analytic gradients. This would be especially useful in situations where we are constrained by the cost or number (shots) of evaluations of quantum circuits for the optimization problem. The experiments below will compare two optimization approaches (analytic gradients vs approximate gradients). One using an analytic gradient approach which could've been done using the general parameter shift rules, but we ended up using backgropagation as these experiments were run on classical simulators [Bergholm et al. (2018)]. This method is compared with our proposed approximate gradient method (the details of which can be seen in the experiments below).

3.1.1 Experiment 1

In *experiment 1* , an ansatz dependant on 6 parameters was prepared. The circuit is as shown in Fig 3.2. We wish to find the minimum eigenvalue of a randomly generated Observable. For the parameters $\{\mu_i\}_{i\in[2..5]}$, since their gate's generators have exactly 2 eigenvalues, the 2 term parameter shift rules should work just fine. In the case of the gates with more eigenvalues $\{\mathcal{G}(\mu_1) = e^{-i\mu_1 G}, \mathcal{G}(\mu_6) = e^{-i\mu_6 G}\}$, our hypothesis is that if set $r = \frac{(\lambda_{max}(G) - \lambda_{min}(G))}{2}$ [1], we should get a good approximation to the analytic gradients.
As we can see in Fig 3.2, the gradients wrt $\{\partial f_{\mu_i}\}_{i\in[2..5]}$ are exactly the ones we get from the 2 term

[1] The idea is similar to the approach used in the two-term parameter shift rule proof to assume symmetric eigenvalues

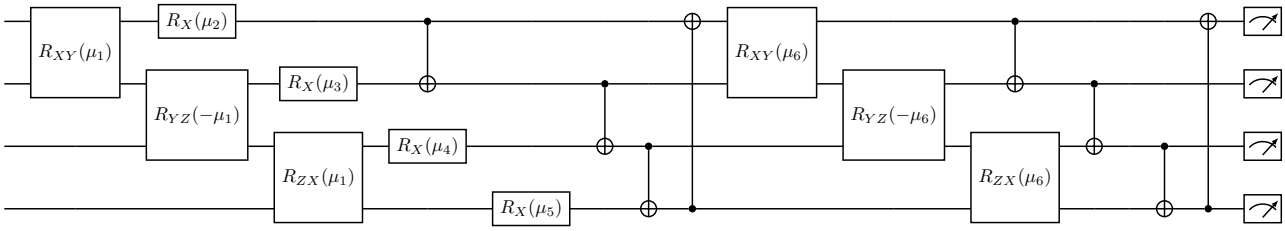


Figure 3.1: Prepared ansatz for toy experiment with 6 parameters and measuring a random hermitian observable

Unitary Generators		
Parameter	Generator (G)	Eigenvalues of G
μ_1, μ_6	$0.5(-X \otimes Y \otimes \mathbb{I} \otimes \mathbb{I} + \mathbb{I} \otimes Y \otimes Z \otimes \mathbb{I} - \mathbb{I} \otimes \mathbb{I} \otimes Z \otimes X)$	$-\frac{3}{2}, -\frac{1}{2}, \frac{1}{2}, \frac{3}{2}$
μ_2, μ_3, μ_4, μ_5	$-0.5X$	$-\frac{1}{2}, \frac{1}{2}$

Table 3.1: Mapping between parameters μ_i and generators G in $\mathcal{G}(\mu_i) = e^{-i\mu_i G}$, and eignevalues of generator G

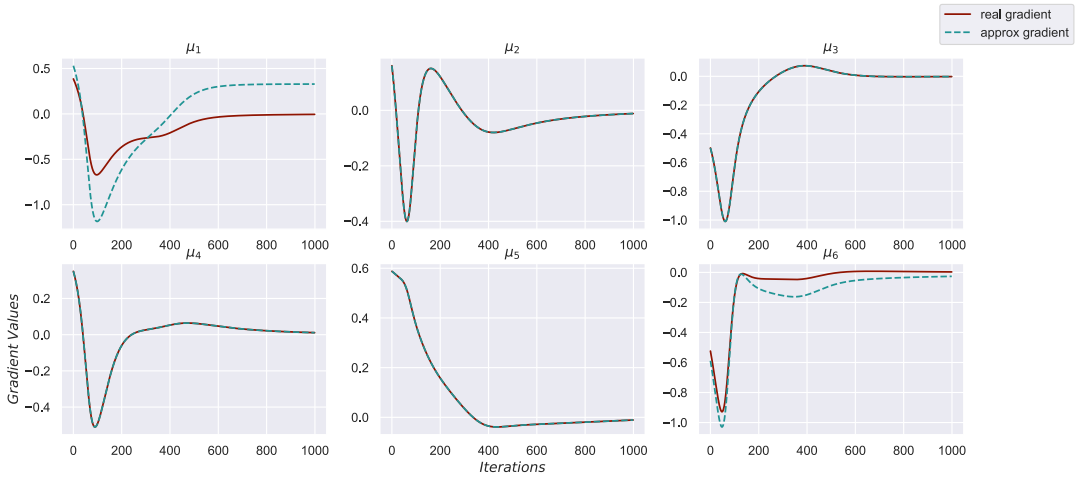


Figure 3.2: Comparison between analytic gradients computed using backpropagation and approximate gradients using our method for ansatz in Fig

parameter shift rule. However, the gradients we are more interested in are for the parameters μ_1 and μ_6 as they correspond to generators with > 2 eigenvalues. The values of the approximate gradients ∂f_{μ_1} and ∂f_{μ_2} follow a similar trend to the analytic (real) ones.

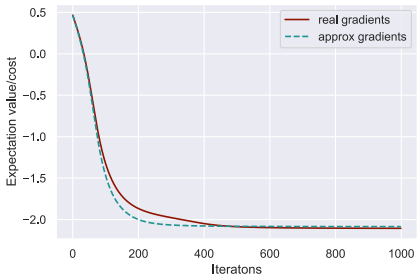


Figure 3.3: Expectation value comparison between the model that follows the real gradients vs the model that follows the approximate gradient

We can see that the approximate gradient method converges faster than the real gradient method from Fig 3.3, and both methods finally converge to the same value.

3.1.2 Experiment 2

For the *2 nd experiment*, an ansatz was prepared with 2 parameters and gates with generators similar to the ansatz in *experiment 1* . Here, we study how fast the approximate gradient algorithm converges as compared to the real gradient algorithm, and whether the approximate gradients follow a sensible descent direction throughout the optimization process.

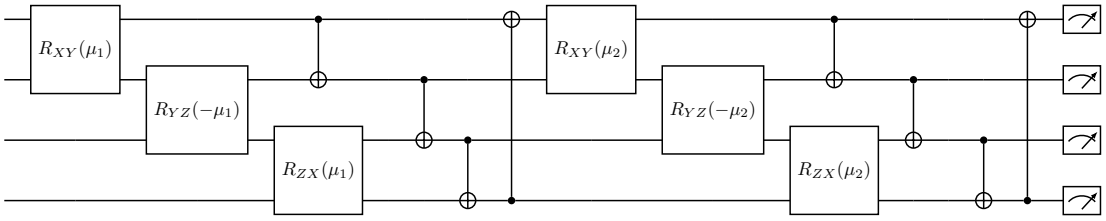


Figure 3.4: Prepared ansatz for toy experiment with 2 parameters and measuring a random hermitian observable

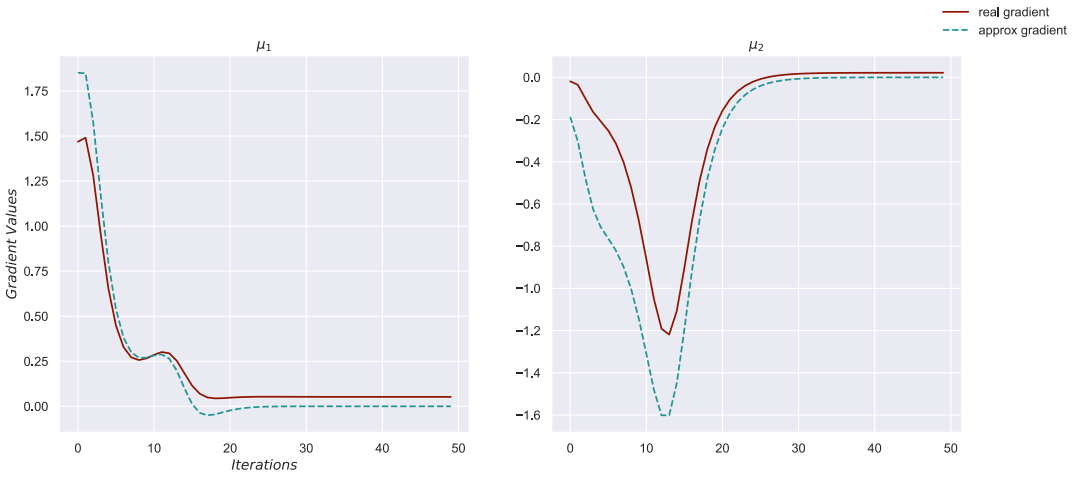


Figure 3.5: Gradient plots for $\partial_{\mu_1} f, \partial_{\mu_2} f$

As we can see from the gradient comparison in Fig 3.5, the gradients from our proposed approximate gradient can be seen to overestimate the real/true gradients, but do indeed follow a similar trend to the true gradients.

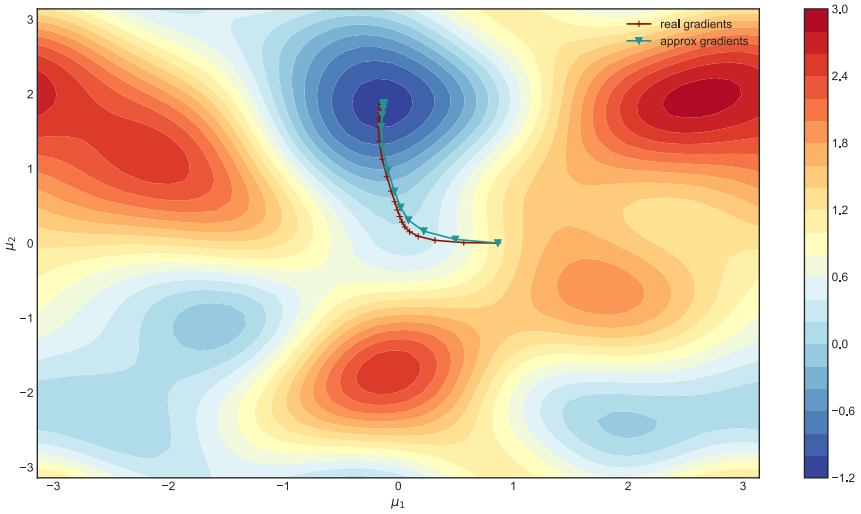


Figure 3.6: Contour plot of cost landscape with 2 parameters μ_1, μ_2. The surface has been plotted between the values $-\pi$ and π. The Approximate gradient approach can be seen to reach the minimum faseter than the analytic gradient approach

In the contour plots (Fig 3.6) for the gradients, we notice that the approximate gradients find the minimum much faster than the true gradients, and also follow a good descent directions throughout the optimization process

Finally, we ran the experiments for 20 pairs of different randomly initialized parameters (μ_1, μ_2), and analysed how good the real gradient and approximate gradient approaches perform on average. Fig 3.7 shows that the approximate gradient method tends to find lower solutions on average and have a lower variance with respect to the random initialization.

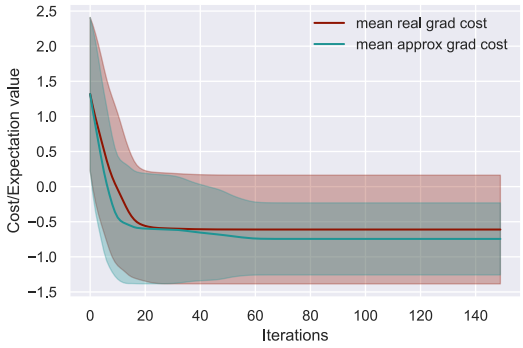


Figure 3.7: Mean and variance comparison of the expectation values for the real and approximate gradient methods in Experiment 2. The experiments were run for 20 repetitions with randomly initialised μ_1, μ_2

3.1.3 Experiment 3

In *experiment 3* , we set up QAOA [Farhi et al. (2014)] algorithm for finding the Minimum Vertex Cover[2] of a graph. A fully connected 3 node graph was used (fig 3.8). Using the QAOA circuit,

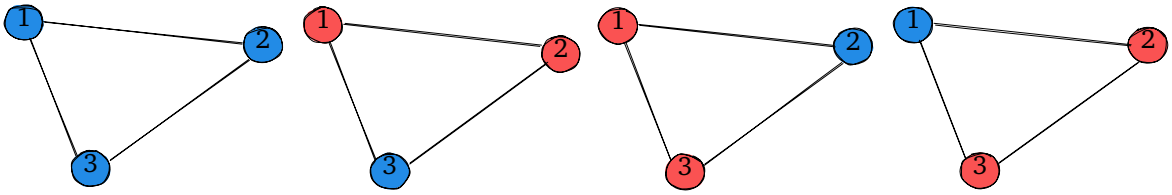


Figure 3.8: The graph (far left) we are using the QAOA algorithm to find the min vertex cover for. The 3 graphs with the red nodes denote the possible min vertex covers of this fully connected 3 node graph.

the solutions are encoded as a binary representation of the states (Eg: a solution of 4 would be the state $|100\rangle$, which would correspond to the first node in the graph), with higher probabilities assigned to more likely solutions. Our approximate gradient algorithm was benchmarked against the real/analytic gradient method. We can see (Fig 3.9) that both methods assign higher probabilities

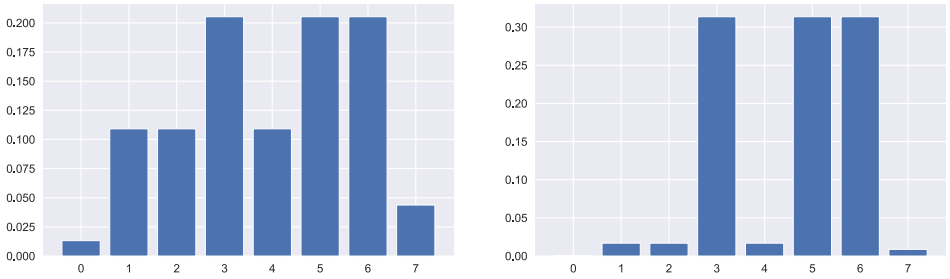


Figure 3.9: Comparison of solutions for min vertex cover using the approx gradients (left) and real gradients (right). The solutions are 3, 5 and 6 which correspond to the states $|011\rangle, |101\rangle$ and $|110\rangle$ respectively (which are indeed the solutions to our problem as they correspond to the set of solutions $\{3 \longrightarrow |011\rangle \longrightarrow \{2,3\}, 5 \longrightarrow |101\rangle \longrightarrow \{1,3\}, 6 \longrightarrow |110\rangle \longrightarrow \{1,2\}\}$)

to the min vertex covers. However, in this case, the analytic gradient method does a better job at amplifying the solutions and suppressing the non solutions compared to our method.
After analysing the gradients (Fig 3.10) in our method, we found that some of the gradients of the parameters in our QAOA ansatz vanish very quickly. A good future step could be to see why this happens, and if there are better ways to make our method more robust to problems that involve using QAOA ansatzs.

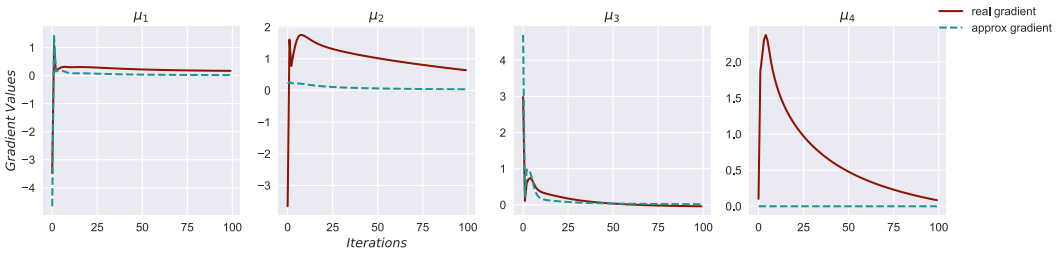


Figure 3.10: Gradient comparison between our method and true/analytic gradients method for finding the min vertex cover using QAOA

[2]A vertex cover is subset of nodes in a graph, where every edge in the graph is adjacent to some vertex in the subset of nodes. A min vertex cover is smallest possible subset of nodes that satisfies the vertex cover condition

Chapter 4

Conclusion and Next Steps

This chapter concludes this report on the parameter shift rules, and offers some guidance on possible ways to take the analysis in this report forward.

This report started off by introducing the field of Quantum Machine Learning. We had seen the important usecases and applications of parametrised circuits, and why studying quantum specific optimization methods matters. Next, we introduced the parameter shift rules, and its variants based on the type of gates that the ansatz is comprised of. We also saw how expensive these rules are, and motivated a need to find some more computationally cheap methods to do optimization on variational circuits. Finally, we conjectured an approach to compute approximate gradients (approximate descent directions rather) for parametrised gates with > 2 eigenvalues and tested it numerically to on two 2 experiments. Our method not only gave good approximations (in *experiment 1 and 2*) for the analytic gradients, but also found the solutions faster than the approach with the real/analytic gradients. Also, in the case of *experiment 3*, our method found the min vertex covers of a fully connected 3 node graph. However, in this case, the real/analytic gradient method was able to do a better job of suppressing the bad solutions by assigning lower probabilities to them. We also noticed a vanishing gradient problem while using the approximate gradient method for QAOA.

Although our numerical experiments did show that the approximate gradient approach did perform better than the analytic gradient one in certain cases, we cannot make any further conclusions without further more rigorous numerical experimentation. Hence, we propose a set of open questions that need to be resolved to make the results from this report more concrete:

1. Run experiments using the approximate gradient approach on bigger problems in VQE and QAOA to see if we get similar or even better solutions than the analytic gradient approach. We also need to understand how the approximate gradient method can be adapted to these problems to give better solutions.

2. On a more theoretical note, can we find good bounds on the gradients that our method computes so that it can be guranteed that we will always move in a direction of descent

3. Finally, is there a better value of r that we can use (instead of $\frac{\lambda_{max} - \lambda_{min}}{2}$)? Since the gradients obtained using our approach were larger in magnitude than the true gradients, we believe that $r = \frac{\lambda_{max} - \lambda_{min}}{2}$ scales the gradients a bit too much. A different value of r, possibly the root mean squared of the eigenvalues of the generator G, might give us better gradient estimates.

Chapter 5

Appendix

Theorem 1 : *If $\mathcal{G}(\mu) = e^{-i\mu G}$ is a unitary matrix with Hermitian G having 2 unique symmetric eigenvalues ($\pm r$), then $\mathcal{G}(\pm\frac{\pi}{4r}) = \frac{1}{\sqrt{2}}(\mathbb{I} \mp ir^{-1}G)$*

Proof:
Adapted from Schuld et al. (2019)
If G is hermitian, and has 2 unique eigenvalues, then it has a spectral decomposition i.e

$$
\begin{aligned}
G &= \sum_{i=1}^{d} \pm r \left|g_i\right\rangle \left\langle g_i\right| \qquad \begin{cases} 0 & i \neq j \text{ the eigenvectors are perpendicular to each other} \\ 1 & i = j \text{ the eigenvectors are unit vectors and parallel} \end{cases} \\
\implies G^2 &= \sum_{i,j=1}^{d} r^2 \left|g_i\right\rangle \left\langle g_i\right|\left.g_j\right\rangle \left\langle g_j\right| \\
G^2 &= r^2 \sum_{i=1}^{d} \left|g_i\right\rangle \left\langle g_i\right| \\
G^2 &= r^2 \mathbb{I}
\end{aligned}
\tag{5.1}
$$

We rewrite the exponential form of $\mathcal{G}(\mu)$ using its Taylor expansion:

$$
\begin{aligned}
\mathcal{G}(\mu) =& \sum_{k=0}^{\infty} \frac{(-i\mu)^k G^k}{k!} \qquad = (G^2)^k = (r^2)^k \mathbb{I} = r^{2k}\mathbb{I} \\
=& \sum_{k=0}^{\infty} \frac{(-i\mu)^{2k} G^{2k}}{(2k)!} + \sum_{k=0}^{\infty} \frac{(-i\mu)^{2k+1} G^{2k+1}}{(2k+1)!} \\
=& \mathbb{I} \sum_{k=0}^{\infty} \frac{(-1)^k (r\mu)^{2k}}{(2k)!} - ir^{-1} G \sum_{k=0}^{\infty} \frac{(-1)^k (r\mu)^{2k+1}}{(2k+1)!} \\
=& \mathbb{I}\cos(r\mu) - ir^{-1} G \sin(r\mu) \\
\implies \mathcal{G}\left(\pm\frac{\pi}{4r}\right) =& \frac{1}{\sqrt{2}}\left(\mathbb{I} \mp ir^{-1}G\right)
\end{aligned}
\tag{5.2}
$$

Theorem 2 : *If $\mathcal{G}(\mu) = e^{-i\mu G}$, and G has eigenvalues $\{\omega_i\}_{i\in[1..d]}$, then the eigenvalues of $\mathcal{G}(\mu)$ are $\{e^{-i\mu\omega_i}\}_{i\in[1..d]}$*

Proof:
Let the eigenvectors of G be $\{|G_i\rangle\}_{i\in[1..d]}$ with corresponding eigenvalues $\{\omega_i\}_{i=0..k}$. The Taylor expansion of $\mathcal{G}(\mu)$ is

$$\mathcal{G}(\mu) = \sum_{k=0}^{\infty} \frac{(-i\mu)^k G^k}{k!}$$

for any eigenvector $|G_i\rangle$ of the generator G

$$\begin{aligned}\mathcal{G}(\mu)\,|G_i\rangle =&(\sum_{k=0}^{\infty} \frac{(-i\mu)^k G^k}{k!})\,|G_i\rangle \\ =&\sum_{k=0}^{\infty} \frac{(-i\mu)^k G^k\,|G_i\rangle}{k!} \\ =&(\sum_{k=0}^{\infty} \frac{(-i\mu\omega_i)^k}{k!})\,|G_i\rangle \\ =&e^{-i\mu\omega_i}\,|G_i\rangle\end{aligned} \tag{5.3}$$

Equation 5.3 $\implies$ $\{|G_i\rangle\}_{i\in[1..d]}$ are eigenvectors of $\mathcal{G}(\mu)$ with corresponding eigenvalues $\{\omega_i\}_{i\in[1..d]}$

Theorem 3 : *Shifting the eigenvalues of a generator G (of a unitary $\mathcal{G}(\mu) = e^{-i\mu G}$) has no observable effects in the expectation value of the ansatz*
Proof:
Let us attempt to shift the eigenvalues of G by a value s. Thus, we need to add a matrix $s\mathbb{I}$ to G to obtain a new generator

$$G' = G + s\mathbb{I} \tag{5.4}$$

To perform this shift to a generator of a unitary matrix $e^{-i\mu G}$, we multiply the matrix by $e^{-i\mu(s\mathbb{I})}$. Using Eq 5.2 with $r = 1$, $G = \mathbb{I}$ and $\mu \longrightarrow s\mu$, we get:

$$\begin{aligned}e^{-i\mu(s\mathbb{I})} =&\mathbb{I}cos(s\mu) - i\mathbb{I}sin(s\mu) \\ =&e^{-is\mu}\mathbb{I}\end{aligned} \tag{5.5}$$

The gate in Eq: 5.5 is an identity matrix that add a global phase $e^{-is\mu}$ to the quantum state it is applied to. In the original ansatz, the expectation value would be $E(\mu) = \langle\psi|\,\hat{B}\,|\psi\rangle$. When computing an expectation value with this eigenvalue shift gate added to the ansatz, we get:

$$\begin{aligned}\langle\psi|\,e^{is\mu}\mathbb{I}\hat{B}e^{-is\mu}\mathbb{I}\,|\psi\rangle =&e^{is\mu}e^{-is\mu}\,\langle\psi|\,\hat{B}\,|\psi\rangle \\ =&\langle\psi|\,\hat{B}\,|\psi\rangle = E(\mu)\end{aligned} \tag{5.6}$$

Thus, shifting the eigenvalues of the generator G has no observable effects to the expectation value of the ansatz. Hence, we assume the eigenvalues to be symmetric wrt 0 in the two-term parameter shift rule proof.

Bibliography

Banchi, L. and Crooks, G. E. (2021). Measuring analytic gradients of general quantum evolution with the stochastic parameter shift rule. *Quantum*, 5:386. pages i, 9

Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M. S., Ahmed, S., Arrazola, J. M., Blank, C., Delgado, A., Jahangiri, S., et al. (2018). Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*. pages 11, 12

Crooks, G. E. (2019). Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition. *arXiv preprint arXiv:1905.13311*. pages i, 9

Farhi, E., Goldstone, J., and Gutmann, S. (2014). A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*. pages 5, 16

Izmaylov, A. F., Lang, R. A., and Yen, T.-C. (2021). Analytic gradients in variational quantum algorithms: Algebraic extensions of the parameter-shift rule to general unitary transformations. *Physical Review A*, 104(6):062443. pages 9

Jones, T. and Gacon, J. (2020). Efficient calculation of gradients in classical simulations of variational quantum algorithms. *arXiv preprint arXiv:2009.02823*. pages 6

Kyriienko, O. and Elfving, V. E. (2021). Generalized quantum circuit differentiation rules. *Physical Review A*, 104(5):052417. pages 9

Mitarai, K., Negoro, M., Kitagawa, M., and Fujii, K. (2018). Quantum circuit learning. *Physical Review A*, 98(3):032309. pages i, 3, 6

Nielsen, M. A. and Chuang, I. (2002). Quantum computation and quantum information. pages 2, 4

Schuld, M., Bergholm, V., Gogolin, C., Izaac, J., and Killoran, N. (2019). Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331. pages 6, 7, 18

Wierichs, D., Izaac, J., Wang, C., and Lin, C. Y.-Y. (2022). General parameter-shift rules for quantum gradients. pages i, 9, 10