Calculation of the Gradient of a Quantum Cost Function using "Threading". Application of these "threaded gradients" to a Quantum Neural Net inspired by Quantum Bayesian Networks

Robert R. Tucci
tucci@ar-tiste.com

April 21, 2019

1 Introduction

Hybrid Quantum Classical (HQC) computation as being pursued by Rigetti Inc. involves minimizing a "quantum cost function", i.e., the mean value of a Hermitian operator, wherein that mean value is calculated empirically from the data yielded by a physical quantum computer.

There are many methods available for minimizing a cost function on a classical computer. Which minimizing method performs best for a particular case depends on the nature of the cost function and of the computing resources at the user's disposal. Some methods, like, for instance, the Powell method (available on `scipy.optimize.minimize`), do not require calculating the gradient of the cost function. Others, the "gradient based methods", do. One method that is particularly well suited to AI problems when distributed computing (GPU, TPU) resources are available, is gradient descent used in conjunction with a method of calculating gradients called back-propagation (BP). This begs the question, can we find a good method like BP for calculating the gradients of a quantum cost function, and then use gradient descent to minimize the cost function? The grand goal is to give an exact analytical formula for the gradients of a quantum cost function, and to express that formula as a sum of parts that can each be readily evaluated empirically on a qc.

Several teams from the Univ. of Tokyo Refs.[1][2], Xanadu Inc. Ref.[3], and QCWare Inc. Ref.[4] have already implemented their own algorithms for accomplishing this grand goal. Refs. [1] and [3] use a formula that allows them to split the calculation of the gradient of a single uncontrolled rotation along either the X, Y or Z axes into two quantum circuits. Refs.[2] and [4] use a tomographic approach.

The method proposed in this paper to tackle this grand goal is not tomographic like the methods of Refs.[2] and [4]. Our method is more akin to the approach of Refs. [1] and [3], except that we use an ancilla qubit (they don't), and we handle the case of gates with multiple controls (they only consider derivatives of gates with no controls). Furthermore, we apply our method to a novel quantum circuit, a quantum neural net inspired by Quantum Bayesian Networks. Last but not least, we point out the huge benefits of "qc threading" for our method. The benefits are particularly high for NISQ (Noisy Intermediate Scale Quantum) devices such as those available on the Rigetti Cloud.

This paper describing our method is being released concurrently with a full implementation, open source, in Python, of our method. The implementation is part of the Qubiter repo at GitHub.

2 Derivative of Uncontrolled U(2) gate

In this section, we will show how to express the derivative of a 2-dim U(2) matrix with respect to one of its 4 parameters, as a linear combination of three U(2) matrices. Luckily, the parameters of most standard quantum gates always appear inside a 2-dim U(2) matrix with zero, one, or more controls attached. So, in this section, we will only concern ourselves with calculating the derivatives of a U(2) matrix.

Mitarai, et al. in Ref. [1] and the authors of Xanadu's PennyLane software Ref.[3] have come up with a similar scheme, but their method differs from ours in some important respects. To tackle a general U(2) transformation, they first decompose it into an Euler product of 3 rotations along the standard X,Y or Z axes, and then they take the derivatives of each of those 3 rotations. In this paper, we give a method that can handle an arbitrary U(2) transformation, without having to do an Euler decomposition first.

Let us first consider a rotation about a standard axis, X, Y or Z, instead of a general U(2) matrix. If we let σ_k for $k = 1, 2, 3$ denote the Pauli matrices, then a rotation about the Z axis is

$$U(\theta_3) = e^{i\sigma_3\theta_3} = C + i\sigma_3 S \ , \tag{1}$$

where θ_3 is some real number and we abbreviate $S = \sin\theta_3, C = \cos\theta_3$. Then

$$\frac{dU}{dt} = \dot{\theta}_3(-S + i\sigma_3 C) \ . \tag{2}$$

Hence

$$\frac{dU}{d\theta_3} = -S + i\sigma_3 C = e^{i(\frac{\pi}{2}+\theta_3)\sigma_3} = U(\frac{\pi}{2} + \theta_3) \ . \tag{3}$$

Thus, for a rotation along a standard axis, one can evaluate the derivative of a gate simply by replacing that gate by that gate with its angle advanced by $\frac{\pi}{2}$. No need to take finite differences.

Now let us consider the most general U(2). We will parameterize it as

$$U = e^{i(\theta_0+\theta_1\sigma_X+\theta_2\sigma_Y+\theta_3\sigma_Z)} \ , \tag{4}$$

where θ_k for $k = 0, 1, 2, 3$ are real numbers. Derivatives with respect to θ_0 are trivial so we will set $\theta_0 = 0$ henceforth. Using the Einstein summation convention,

$$U = e^{i\sigma_k\theta_k} = C + i\sigma_k\frac{\theta_k}{\theta}S \ , \tag{5}$$

where we are abbreviating

$$\theta = \sqrt{\theta_k\theta_k}, S = \sin\theta, C = \cos\theta \ . \tag{6}$$

Then, it's easy to show that

$$\frac{dU}{dt} = -S\frac{\theta_k}{\theta}\dot{\theta}_k + i\sigma_k\dot{\theta}_r\left[\frac{\theta_k\theta_r}{\theta^2}C + \frac{S}{\theta}(-\frac{\theta_k\theta_r}{\theta^2} + \delta_{k,r})\right] \ . \tag{7}$$

In the rest of this section, we will try to recast the right hand side of Eq.(7) into a form that is more convenient for empirical calculation from qc data. Before embarking on this task, let us introduce some notation.

Henceforth, we will represent a real-valued unit vector by a letter with a caret above it: $\hat{a} = \frac{\vec{a}}{|\vec{a}|}$. Also, for any real-valued 3-dim vector $\vec{a}$, let

$$\sigma_{\vec{a}} = \vec{a} \cdot \vec{\sigma} \ . \tag{8}$$

Eq.(8) is a natural generalization of the Pauli matrix notation. If $\hat{e}_A$ is the unit vector in direction A for $A = X, Y, Z$, then $\hat{e}_A \cdot \vec{\sigma} = \sigma_A$ for $A = X, Y, Z$. Expressed in the notation of Eq.(8), two familiar Pauli matrix identities are

$$\sigma_{\vec{a}}\sigma_{\vec{b}} = \vec{a} \cdot \vec{b} + i\sigma_{\vec{a}\times\vec{b}} \ , \tag{9}$$

where $\vec{a}$ and $\vec{b}$ are any two 3-dim vectors, and

$$e^{i\theta\sigma_{\hat{n}}} = \cos\theta + i\sigma_{\hat{n}}\sin\theta \ , \tag{10}$$

where θ is a real number and $\hat{n}$ is a 3-dim unit vector. Eq.(10) can be proven by Taylor expanding, and using $\sigma_{\hat{n}}^2 = 1$.

We will also use the following notation for the projectors P_0 and P_1 along the direction of $|0\rangle$ and $|1\rangle$, respectively, in a 1-qubit space.

$$\begin{array}{l} n = P_1 = |1\rangle\langle 1|, \\ \bar{n} = 1 - n = P_0 = |0\rangle\langle 0| \end{array} \ . \tag{11}$$

n is often called the number operator. Whenever we say $\Omega(\alpha)$ for a 1-qubit operator Ω, we mean Ω applied to qubit α.

Eq.(7) for the general form of $\dot{U}$ when parameterized as Eq.(4) (with $\theta_0 = 0$) is fairly opaque. To clarify Eq.(7), we start by specializing it to $t = \theta_1$. The cases $t = \theta_2, \theta_3$ can be obtained from the case $t = \theta_1$ simply by replacing 1 subscripts in our final result by 2 or 3. So, setting $t = \theta_1$ in Eq.(7), we immediately get:

$$\frac{\partial U}{\partial \theta_1} = \left\{ \begin{array}{l} -\frac{\theta_1 S}{\theta^2}\left[i\sigma_{\hat{\theta}}\right] \\ +\frac{\theta_1}{\theta}\left[-S + i\sigma_{\hat{\theta}}C\right] \\ +\frac{S}{\theta}\left[i\sigma_1\right] \end{array} \right. \ . \tag{12}$$

If we define

$$p_1 = \frac{\theta_1}{\theta}, \ \ p_S = \frac{S}{\theta} \ , \tag{13}$$

then

$$\boxed{\frac{\partial U}{\partial \theta_1} = \left\{ \begin{array}{l} -p_1 p_S \left[e^{i\frac{\pi}{2}\sigma_{\hat{\theta}}} \right] \\ +p_1 \left[e^{i(\frac{\pi}{2}+\theta)\sigma_{\hat{\theta}}} \right] \\ +p_S \left[e^{i\frac{\pi}{2}\sigma_1} \right] \end{array} \right. .} \tag{14}$$

Henceforth, we will refer to these 3 pieces as "dparts", which stands for "derivative parts". Note that we can replace 1 by 2 or 3 in Eq.(14) to get partials of U with respect to θ_2 and θ_3.

3 Derivative of Quantum Cost Function

In this section, we show how to express the gradient of a quantum cost function as a sum of mean values that can be readily evaluated empirically on a real qc.

Suppose a, A are integers such that $a \leq A$, and Ω_j are operators acting on N_b qubits. Define

$$\Omega_{[a,A]} = \Omega_a \Omega_{a+1} \dots \Omega_A \tag{15}$$

and

$$\Omega_{[A,a]} = \Omega_A \dots \Omega_{a+1} \Omega_a \ . \tag{16}$$

Note that

$$(\Omega_{[a,A]})^\dagger = \Omega^\dagger_{[A,a]} \ . \tag{17}$$

The cost function that we minimize in Hybrid Quantum Classical computing can be expressed analytically as

$$C = \langle \psi_0 | (\Omega_{[T-1,0]})^\dagger \mathcal{H} \Omega_{[T-1,0]} | \psi_0 \rangle \ , \tag{18}$$

where the Ω_j are unitary operators and $\mathcal{H}$ is a Hermitian operator acting on N_b qubits. Now let us focus on taking the derivative of Ω_τ, the gate for a particular time $\tau \in \{0, 1, \dots T-1\}$. Define

$$|\psi_\tau\rangle = \Omega_{[\tau-1,0]} |\psi_0\rangle \tag{19}$$

and

$$\mathcal{H}_\tau = (\Omega_{[T-1,\tau]})^\dagger \mathcal{H} \Omega_{[T-1,\tau]} \ . \tag{20}$$

Henceforth, we will use the angled brackets to denote an average with respect to state $|\psi_\tau\rangle$:

$$\langle\psi_\tau| \cdot |\psi_\tau\rangle = \langle\cdot\rangle \; . \tag{21}$$

Using the above notation, the cost function and its derivative with respect to a parameter $\theta_{\tau 1}$ that lives inside the operator $\Omega_{\tau 1}$, can be expressed as

$$C = \langle\mathcal{H}_\tau\rangle \; , \tag{22}$$

and

$$\frac{\partial C}{\partial\theta_{\tau 1}} = \langle\mathcal{H}_\tau\Omega_\tau^\dagger\frac{\partial\Omega_\tau}{\partial\theta_{\tau 1}}\rangle + h.c. \tag{23}$$

h.c. denotes the hermitian conjugate of the preceding twin expression.

Let $U_\tau(0)$ be an element of $SU(2)$ acting on qubit 0. $U_\tau(0)$ can be parameterized as

$$U_\tau(0) = e^{i[\theta_{\tau 1}\sigma_X(0)+\theta_{\tau 2}\sigma_Y(0)+\theta_{\tau 3}\sigma_Z(0)]} \; , \tag{24}$$

where the $\theta_{\tau d}$ for $d = 1, 2, 3$ are real numbers, and $\sigma_X(0), \sigma_Y(0), \sigma_Z(0)$ are the Pauli matrices acting on qubit 0. For definiteness and as a good illustration, we will henceforth assume that the unitary operator Ω_τ has the special form of an SU(2) gate with two controls:

$$\Omega_\tau = U_\tau(0)^{n(1)n(2)} \; , \tag{25}$$

where, as usual, $n = |0\rangle\langle 0|$ is the number operator, and $n(1), n(2)$ are number operators acting on qubits 1 and 2, respectively. As was shown in Section 2, one can express the partial derivative of $U_\tau(0)$ with respect to its parameter $\theta_{\tau 1}$, as a linear combination

$$\frac{\partial U_\tau(0)}{\partial\theta_{\tau 1}} = \sum_k \lambda_{\tau 1,k} V_{\tau 1,k}(0) \; , \tag{26}$$

where the coefficients $\lambda_{\tau 1,k}$ are real numbers and the $V_{\tau 1,k}(0)$ are elements of $SU(2)$. Here $k = 1, 2, 3$. From Eqs.(25) and (26), it follows that

$$\Omega_\tau^\dagger\frac{\partial\Omega_\tau}{\partial\theta_{\tau 1}} = \sum_k \lambda_{\tau 1,k}\left[n(1)n(2)U_\tau^\dagger(0)V_{\tau 1,k}(0)\right] \; , \tag{27}$$

which, when substituted into Eq.(23), yields:

$$\frac{\partial C}{\partial \theta_{\tau 1}} = \sum_k \lambda_{\tau 1,k} \left\langle \mathcal{H}_\tau n(1)n(2)U_\tau^\dagger(0)V_{\tau 1,k}(0) + h.c.\right\rangle \; . \tag{28}$$

Note that $U_\tau^\dagger(0)V_{\tau 1,k}(0)$ is a product of SU(2) matrices, so it is itself an SU(2) matrix acting on qubit 0. Hence, it can be expressed as

$$U_\tau^\dagger(0)V_{\tau 1,k}(0) = \exp[i\alpha_{\tau 1,k}\sigma_{\hat{\alpha}_{\tau 1,k}}(0)] = W_{\tau 1,k}(0) \; , \tag{29}$$

where $\alpha_{\tau 1,k}$ is a real number and $\hat{\alpha}_{\tau 1,k}$ is a real valued unit vector.

Recall that if $\sigma_Z(\beta)$ is the Z Pauli matrix, and $n(\beta) = |0\rangle\langle 0|_\beta$ is the number operator, acting on qubit β, then

$$1 - 2n(\beta) = (-1)^{n(\beta)} = \sigma_Z(\beta) \; . \tag{30}$$

This result relies on the fact that the projection operator n satisfies $n^2 = n$ and, therefore, it can only have two eigenvalues, 0 and 1. Let

$$\eta(1,2) = n(1)n(2) \; . \tag{31}$$

Eta's square also equals itself so $\eta \in \{0,1\}$. Therefore

$$1 - 2\eta = (-1)^\eta = (-1)^{n(1)n(2)} = \sigma_Z(1)^{n(2)} = \sigma_Z(2)^{n(1)} \; . \tag{32}$$

Let "dpart" (derivative part) stand for

$$\text{dpart} = \langle \mathcal{H}_\tau \eta(1,2)W(0)\rangle + h.c. \; . \tag{33}$$

Since

$$\eta = \frac{1}{2}(1 - (-1)^\eta) \; , \tag{34}$$

it follows that

$$\text{dpart} = \left\{ \begin{array}{c} \frac{1}{2}\langle \mathcal{H}_\tau W\rangle + h.c. \\ -\frac{1}{2}\langle \mathcal{H}_\tau (-1)^\eta W\rangle + h.c. \end{array} \right. \; . \tag{35}$$

Next we add to the quantum circuit an extra ancilla qubit called ξ. We will refer to the top (resp., bottom) term on the right hand side of Eq.(35) as depart$_+$ (resp., dpart$_-$). The sign of the dpart will be referred to as its "polarity". We can get dpart$_\pm$ by starting the ancilla qubit ξ at $|0\rangle$, applying a Hadamard matrix to it, and measuring the mean value of σ_X for ξ. In other words, one can verify that

$$\text{dpart}_+ = \left[\begin{array}{ccccc} \langle\psi_\tau| & | & \mathcal{H}_\tau & | & |\psi_\tau\rangle \\ & (W^\dagger)^{n(\xi)} & & W^{n(\xi)} & \\ \langle 0|_\xi Had(\xi) & | & \sigma_X(\xi) & | & Had(\xi)|0\rangle_\xi \end{array}\right] , \tag{36}$$

and

$$\text{dpart}_- = -\left[\begin{array}{ccccc} \langle\psi_\tau| & | & \mathcal{H}_\tau & | & |\psi_\tau\rangle \\ & (W^\dagger)^{n(\xi)}\sigma_Z(\xi)^\eta & & \sigma_Z(\xi)^\eta W^{n(\xi)} & \\ \langle 0|_\xi Had(\xi) & | & \sigma_X(\xi) & | & Had(\xi)|0\rangle_\xi \end{array}\right] . \tag{37}$$

4 Application to a type of Quantum Neural Net inspired by Quantum Bayesian Networks

So far, we have considered the derivative of any single gate of a quantum cost function. We considered a multi-controlled U(2) gate, and a derivative with respect to any of its 4 parameters. But we left the quantum cost function arbitrary. In this section, we apply the results of previous sections to a very special type of quantum cost function.

More specifically, in this section we will consider a quantum cost function $\langle\psi_T|\mathcal{H}|\psi_T\rangle$ whose state $|\psi_T\rangle$ is a quantum circuit that could be described as a Quantum Neural Net inspired by Quantum Bayesian Networks. We will also assume that the Hermitian operator $\mathcal{H}$ is given to us already expressed as a "QubitOperator". `QubitOperator` is a class in the open-source software OpenFermion. It stores $\mathcal{H}$ as a a linear combination with real coefficients c_r of tensor products (aka Pauli strings) of a Pauli operator (or the identity) acting on each qubit:

$$\mathcal{H} = \sum_r c_r \prod_{\beta=0}^{N_b-1} \sigma_{d_{\beta,r}}(\beta) \; . \tag{38}$$

Here $d_{\beta,r} \in \{0,1,2,3\}$ so as to include the identity and the 3 Pauli matrices.

In Qubiter's implementation of the ideas of this paper, we call the quantum circuit representing $|\psi_T\rangle$ a "Stairs Circuit". For example, this is what the Qubiter's Picture file of a Stairs Circuit looks like for 3 qubits

```
U   |   |
O---U   |
@---U   |
O---O---U
O---@---U
@---O---U
@---@---U
```

Here, U is a general U(2) matrix with 4 parameters, all of which can be made into placeholder variables or simply kept as floats. If each U is represented by a node and the controls of each U represent its parents, then this quantum circuit can be represented by a fully connected Quantum Bayesian Network (QB net). (See my 10 year old blog called “Quantum Bayesian Networks” for more info than you would ever want to know about QB nets).

Qubiter can also be asked to construct a QB net that is **not** fully connected, by limiting the number of controls for a given U to fewer than a control on all the qubits to the left of U. For example, suppose that in the 3 qubits case, we restrict the parents of the U in the last gate to just one, instead of the 2 parents that it has in the fully connected case. Then we get

```
U   |   |
O---U   |
@---U   |
O---+---U
@---+---U
```

when qubit 0 has qubit 2 but not 1 as a parent, or

```
U   |   |
O---U   |
@---U   |
|   O---U
|   @---U
```

when qubit 0 has qubit 1 but not 2 as a parent.

Qubiter appends at the end of the above quantum circuits a “coda” (tail). Note that

$$e^{-i\frac{\pi}{4}\sigma_Y}\sigma_Z e^{i\frac{\pi}{4}\sigma_Y} = \sigma_X \ , \tag{39}$$

$$e^{i\frac{\pi}{4}\sigma_X}\sigma_Z e^{-i\frac{\pi}{4}\sigma_X} = \sigma_Y \ . \tag{40}$$

A coda is a rotation $e^{i\frac{\pi}{4}\sigma_Y}$ for every Pauli σ_X, and a rotation $e^{-i\frac{\pi}{4}\sigma_X}$ for every Pauli σ_Y, whenever σ_X or σ_Y occur on the right hand side of Eq.(38). If such a coda is appended to the quantum circuit, then we only have to measure mean values of σ_Z or 1 on all qubits. Such mean values are easy to evaluate because they are diagonal matrices of ones and minus ones.

The quantum circuit corresponding to a fully connected QB net has a number of U(2) gates which grows exponentially with the number of qubits, and each U(2) gate has as controls all the qubits to its left.[1] Clearly, the complexity of such quantum circuits blows up exponentially. But suppose we limit the number of parents of every U(2) to a fixed number, say, for example, to 2 (except for the first gate, the "prior", which has no parents, and the second gate, which can have only one parent). Such parent-limited quantum circuits have a complexity which increases polynomially in the number of qubits. Which 2 parents are chosen for each U(2) can be decided heuristically in numerous ways. For example, one can move one parent, chosen at random, from qubit A to qubit B, evaluate the cost function for each case, and choose the parent with the lower cost.

5 Threaded Gradients

As an example, suppose that we are taking the derivative of the third gate, of the fully connected 3 qubit graph with respect to $t1 = \theta_1$:

```
        U   |   |
        0---U   |
d/dt1   @---U   |
        0---0---U
        0---@---U
        @---0---U
        @---@---U
```

When we do so, we get a quantum circuit with positive polarity:

[1]Furthermore, current quantum computers require exponential complexity in number of controls to calculate multi-controlled gates. I characterize this as a paradox and a serious shortcoming.

```
|   U   |   |
|   O---U   |
@---@---U   |
|   O---O---U
|   O---@---U
|   @---O---U
|   @---@---U
```

and one with negative polarity:

```
|   U   |   |
|   O---U   |
@---@---U   |
Z---@   |   |
|   O---O---U
|   O---@---U
|   @---O---U
|   @---@---U
```

We would also have to append codas to these circuits. Note that an ancilla qubit was added, so the quantum circuit being differentiated has 3 qubits, but its derivative parts have 4 qubits.

The idea of "threaded gradients" is to partition all the qubits of a qc into groups of 4 qubits (4 is just for this example), and to evolve each of these 4 qubit islands independently, and concurrently. Each 4 qubit island is assigned the task of evaluating the mean value of one of the dparts.

Obviously, multi-threading the task of evaluating a gradient of a quantum cost function, is ideal for a NISQ device, because it only requires quantum entanglement inside small islands of 4 (4 is just for this example) qubits each, but no correlation among islands.

References

[1] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. "Quantum circuit learning" arXiv:1803.00745.

[2] Ken M. Nakanishi, Keisuke Fujii, Synge Todo, "Sequential minimal optimization for quantum-classical hybrid algorithms", arXiv:1903.12166 [quant-ph]

[3] PennyLane documentation has large list of references to arXiv papers by themselves and others.

[4] Robert M. Parrish, Joseph T. Iosue, Asier Ozaeta, Peter L. McMahon, “A Jacobi Diagonalization and Anderson Acceleration Algorithm For Variational Quantum Algorithm Parameter Optimization”, arXiv:1904.03206 [quant-ph]