

# Evaluating Gradients of Quantum Neural Nets

by Dr. Robert R. Tucci



Khyber Pass, mountain pass between Pakistan and Afghanistan

## **choke point**

*noun*

a point of congestion or blockage.  
"the tunnel is a choke point at rush hour"

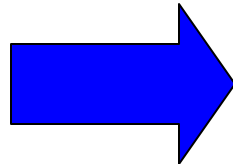
Evaluating  
Gradients  
is a choke point of  
classical and  
quantum AI

# Classical Neural Networks

Minimize “cost”  
function using  
“gradient descent”

Gradients calculated using  
“back-propagation”

Python software libs:  
TensorFlow (Google),  
PyTorch (Facebook),  
Pyro(Uber)



# Quantum Neural Networks

same

Gradients calculated using  
Our New Algo

Rigetti PyQuil,  
IBM QisKit,  
Google Cirq,  
Our Qubiter

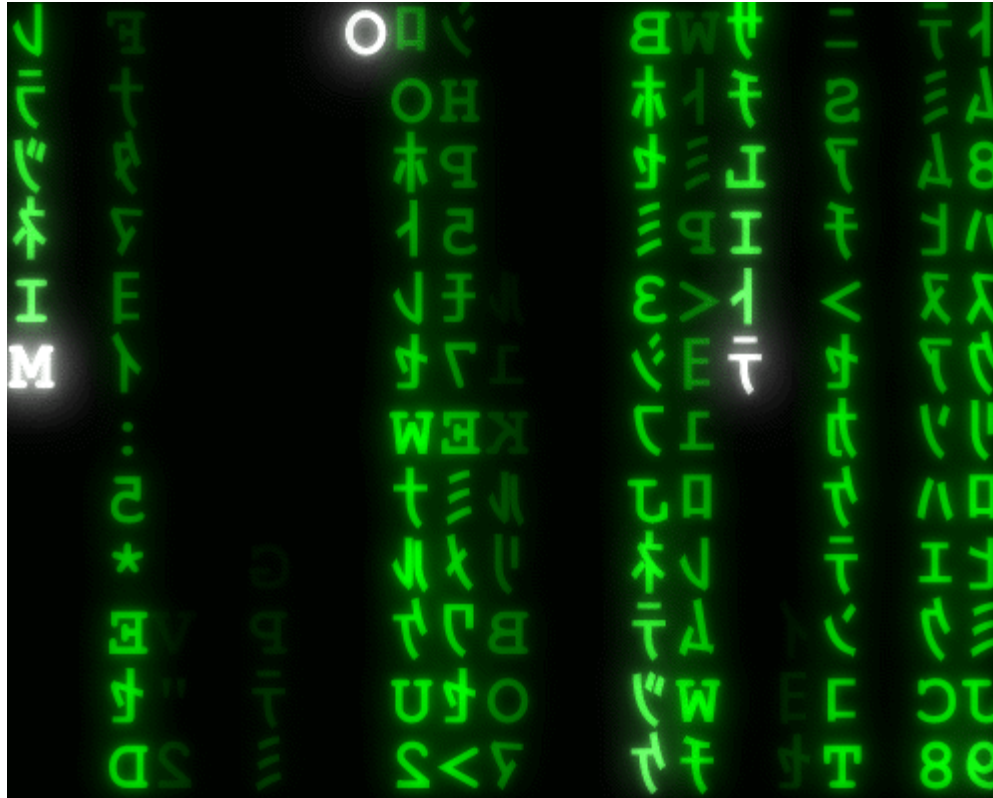
# Our Algo uses Multi-threading

## **So what do I (Tucci) mean by threading?**

(I and most people use the words threading and multi-threading synonymously) I believe I am one of the first persons to use the word threading in connection with quantum computing. What I mean by it is the strategy of partitioning the qubits in a (gate model) quantum computer into small, disjoint sets (“islands”) that are uncorrelated from each other and run concurrently. The qubits within one of these islands are strongly correlated but qubits from different islands are probabilistically independent. This is an ideal scenario for NISQ (Noisy Intermediate Scale Quantum) devices and HQC (Hybrid Quantum Classical) computing being pursued by Rigetti Inc. and others. It is also a good fit for calculating the gradient of quantum cost functions: Each island, after many shots and final measurements, yields a mean value, and a linear combination of the mean values from all the islands equals the gradient. In an artistic, poetical sense, qc threading reminds me of what is commonly called “digital rain”, especially if one draws quantum circuits with time pointing downwards, like Qubiter does.

# Digital Rain

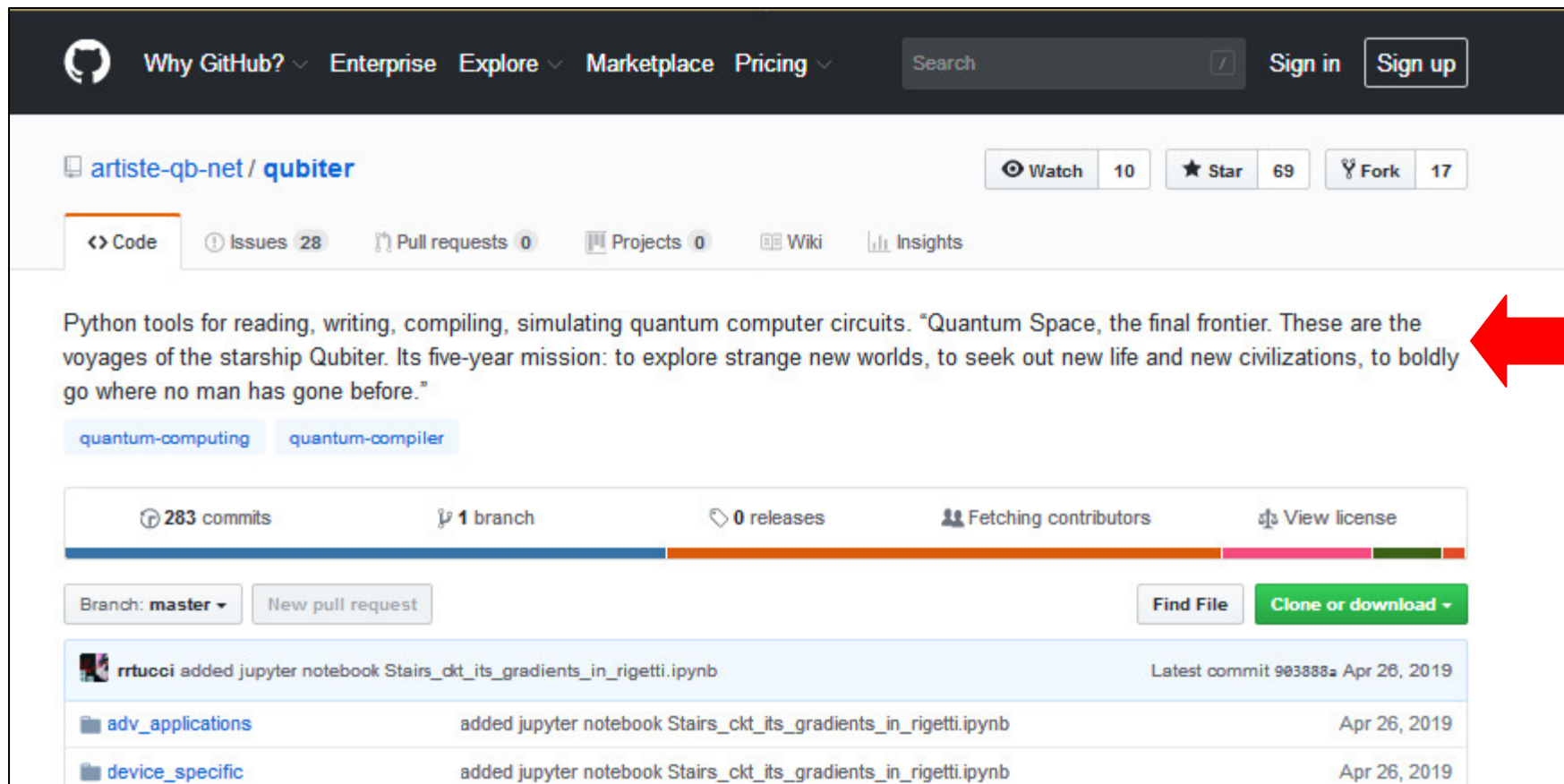
Made famous by The Matrix movie series



Animated gif from  
[https://en.wikipedia.org/wiki/Matrix\\_digital\\_rain](https://en.wikipedia.org/wiki/Matrix_digital_rain)

# Our Algo is fully implemented As part of Qubiter Software Library (Free, Open Source, BSD license)

<https://github.com/artiste-qb-net/qubiter>



The screenshot shows the GitHub repository page for 'artiste-qb-net/qubiter'. The repository description is: "Python tools for reading, writing, compiling, simulating quantum computer circuits. "Quantum Space, the final frontier. These are the voyages of the starship Qubiter. Its five-year mission: to explore strange new worlds, to seek out new life and new civilizations, to boldly go where no man has gone before." A red arrow points to this description. The repository has 10 watches, 69 stars, and 17 forks. It contains 283 commits, 1 branch, and 0 releases. The current branch is 'master'. The commit history shows a commit by 'rrtucci' on Apr 26, 2019, adding a jupyter notebook file.

Python tools for reading, writing, compiling, simulating quantum computer circuits. "Quantum Space, the final frontier. These are the voyages of the starship Qubiter. Its five-year mission: to explore strange new worlds, to seek out new life and new civilizations, to boldly go where no man has gone before."

quantum-computing quantum-compiler

283 commits 1 branch 0 releases Fetching contributors View license

Branch: master New pull request Find File Clone or download

Commit	Author	Message	Date
Latest commit 903888	rrtucci	added jupyter notebook Stairs_ckt_its_gradients_in_rigetti.ipynb	Apr 26, 2019
	adv_applications	added jupyter notebook Stairs_ckt_its_gradients_in_rigetti.ipynb	Apr 26, 2019
	device_specific	added jupyter notebook Stairs_ckt_its_gradients_in_rigetti.ipynb	Apr 26, 2019

# White Paper Describing Algo

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

**Available at**  
**GitHub repository**  
**for Qubiter**  
[https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/adv\\_applications/threaded\\_grad.pdf](https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/adv_applications/threaded_grad.pdf)

# Two Jupyter Notebooks put our algo through its paces

[https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/jupyter\\_notebooks/Stairs\\_circuit\\_and\\_its\\_gradients\\_in\\_native.ipynb](https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/jupyter_notebooks/Stairs_circuit_and_its_gradients_in_native.ipynb)

This notebook evaluates Gradients using Qubiter's native simulator

---

[https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/jupyter\\_notebooks/Stairs\\_ckt\\_its\\_gradients\\_in\\_rigetti.ipynb](https://github.com/artiste-qb-net/qubiter/blob/master/qubiter/jupyter_notebooks/Stairs_ckt_its_gradients_in_rigetti.ipynb)

This notebook evaluates Gradients using Rigetti's actual physical device (QPU) Or their simulator (Quantum Virtual Machine, QVM) Both are accessible via Rigetti Cloud Service