

# Doubly stochastic gradient descent

EEE 606: Adaptive Signal Processing

Rajesh, M.S. CS, On-Campus

# Overview

Problem Statement

QSGD Introduction

Variational Quantum Eigensolver

Quantum Neural Networks

Results

Conclusion

# Problem Statement

- ✓ Understand and implement Quantum doubly stochastic gradient descent (QSGD) for two problems:
  - i. Variational Quantum Eigensolver
  - ii. Quantum Neural Networks
- ✓ Compare the convergence curve of QSGD and classical (vanilla) gradient descent

## Main Reference Paper:

1. Ryan Sweke, Frederik Wilde, Johannes Jakob Meyer, Maria Schuld, Paul K. Fährmann, Barthélémy Meynard-Piganeau, Jens Eisert. “Stochastic gradient descent for hybrid quantum-classical optimization.” arXiv:[1910.01155](https://arxiv.org/abs/1910.01155), 2019.

# Stochastic Gradient Descent algorithm

Consider the  $n$  training samples  $\mathcal{D} = (\langle x^{[1]}, y^{[1]} \rangle, \langle x^{[2]}, y^{[2]} \rangle, \dots, \langle x^{[n]}, y^{[n]} \rangle) \in (\mathbb{R}^m \times \{0,1\})^n$

- Initialize weights and bias,  $w := 0^{m-1}, b := 0$
- For iteration  $t \in [1, \dots, T]$ :
  - We will draw one random sample with replacement:  $\langle x^{[i]}, y^{[i]} \rangle \in \mathcal{D}$
  - Calculate the prediction  $\hat{y}^{[i]} := h(x^{[i]})$
  - Calculate the loss function  $\mathcal{L}^{[i]} := L(\hat{y}^{[i]}, y^{[i]})$
  - Compute the gradients:  $\Delta w := -\nabla_{\mathcal{L}^{[i]}} w, \Delta b := \frac{\partial \mathcal{L}^{[i]}}{\partial b}$
  - Update the new weights and bias,  $w := w + \Delta w, b := b + \Delta b$

**Variation:** If we sample  $d$  mini samples, instead of only one, we call that mini batch stochastic gradient descent

## Advantages

- The loss function  $\mathcal{L}$  can be computed much more efficiently with only one sample or small random samples than using all data samples  $\mathcal{D}$ .
- The stochasticity can help avoid local minima and saddle points
- The convergence properties are much more superior than regular gradient descent

# Doubly stochastic gradient descent: VQA

- In variational quantum algorithms (VQA), we encode the data samples using parameterized quantum circuits, denoted by the operator  $U(\theta)$  and set of observables  $\{A_i\}$
- The parameterized quantum circuit (or) ansatz  $U(\theta)$  is optimized with respect to its parameters  $\theta$  to minimize the expectation values. If  $\{A_i\}$  are a set of observables, the expectation values are given by,

$$\langle A_i \rangle = \langle 0 | U(\theta)^\dagger A_i U(\theta) | 0 \rangle$$

- The loss function is defined by,

$$\mathcal{L}(\theta, \langle A_1, \dots, A_M \rangle)$$

- The loss function is optimized using gradient descent in a classical optimization loop

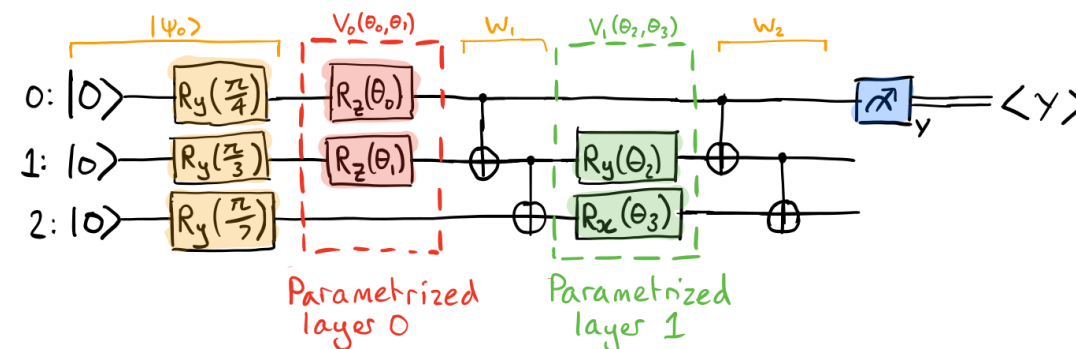


Fig 1. Sample parameterized quantum circuit.  
Reference from [Pennylane](https://pennylane.ai)

# Doubly stochastic gradient descent: Parameter-shift rule

- In order to compute the partial derivatives in the gradient descent algorithm, we use the parameter-shift rule in quantum machine learning
- We use variational quantum algorithms to compute the partial derivatives using the parameter-shift rule. It is given by,

$$\nabla_{\theta} \mathcal{L} = \frac{\mathcal{L}(\theta_1) - \mathcal{L}(\theta_2)}{\theta_1 - \theta_2}$$

- In order to evaluate each of the terms to compute the gradient, the expectation values are measured from all the samples with the variational quantum circuits
- We can also express the parameter-shift rule as a linear combination of our quantum measurement. It is proved in the paper that we can also sample the terms to compute the gradients
- This second sampling of  $k$  terms (even 1 term) of expectation values is why we call this algorithm “double stochastic”.

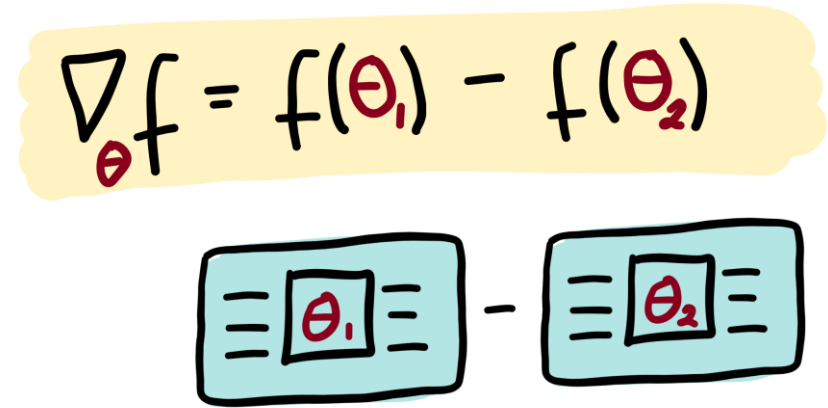

$$\nabla_{\theta} f = f(\theta_1) - f(\theta_2)$$

Fig 2. Parameter-shift rule, [pennylane](https://pennylane.ai)

# Variational Quantum Eigensolver

- It is a variational quantum algorithms (VQA) suited for near term quantum computers to determine the ground state of the given molecule
- It is one of the most important problem in quantum chemistry
- The inputs to a variational quantum Eigensolver are the Hamiltonian\* of the molecule and a parameterized quantum circuit.
- The loss function is defined in terms of the expectation value of the Hamiltonian in the trial state

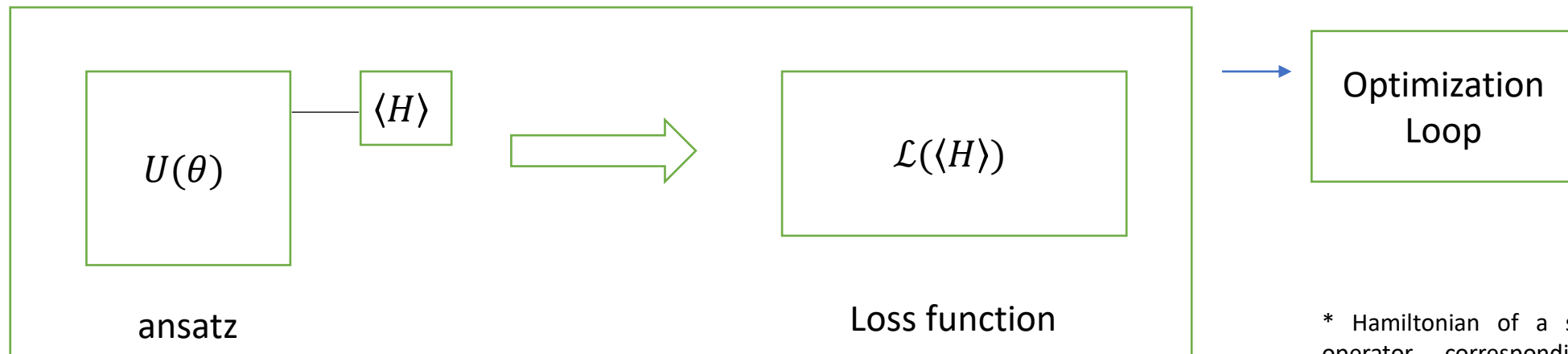


Fig 3. VQE block diagram

\* Hamiltonian of a system is an operator corresponding to the energy of the system

# VQE: Single-shot SGD

- We determined the ground state of the below Hamiltonian:  
 $H = [[8, 4, 0, -6], [4, 0, 4, 0], [0, 4, 8, 0], [-6, 0, 0, 0]]$
- We used [pennylane](#) to simulate the optimization of this Hamiltonian. The below ansatz was used.

Parameters: Learning rate:  $\eta = 0.01$ , steps = 200

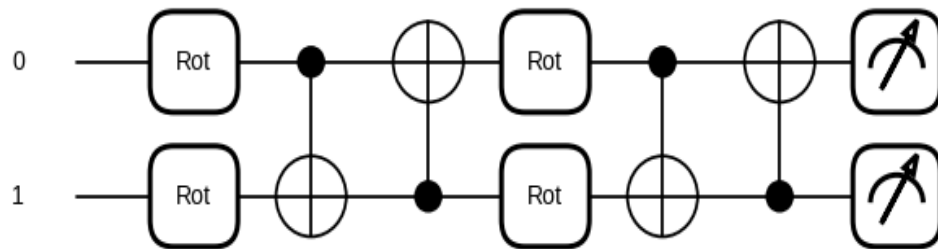


Fig 4. VQE SGD ansatz

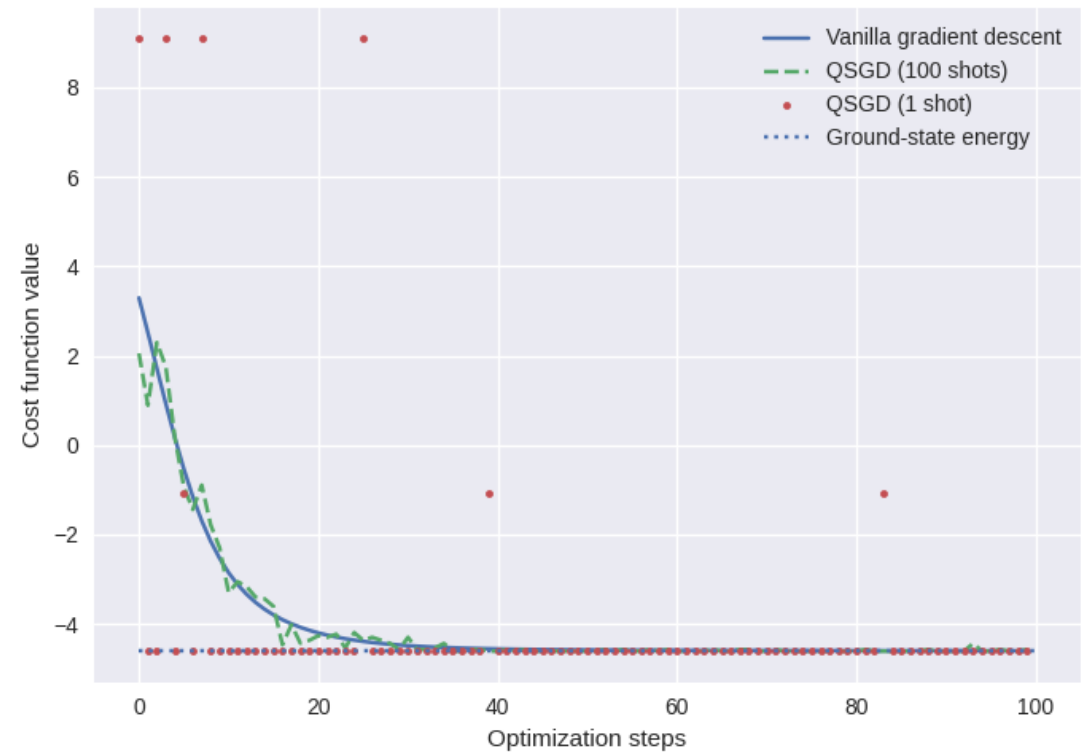


Fig 5. VQE single-shot SGD convergence curve

Observation: We can see that the loss function is almost the same as vanilla GD



# VQE: Doubly Stochastic Gradient Descent

- In this result, we randomly sample  $k$  terms of the Hamiltonian  $H$ , after expressing them as a sum of pauli matrices

$$H = \sum_{i,j=0,1,2,3} a_{i,j}(\sigma_i \otimes \sigma_j)$$

where  $\sigma = \{I, X, Y, Z\}$  represent pauli gates

- In order to keep track of the optimization convergence, we include a “moving average” of the cost evaluations

Observation: We can see that the loss function is almost the same as vanilla GD, with the moving average, but does not work well without it

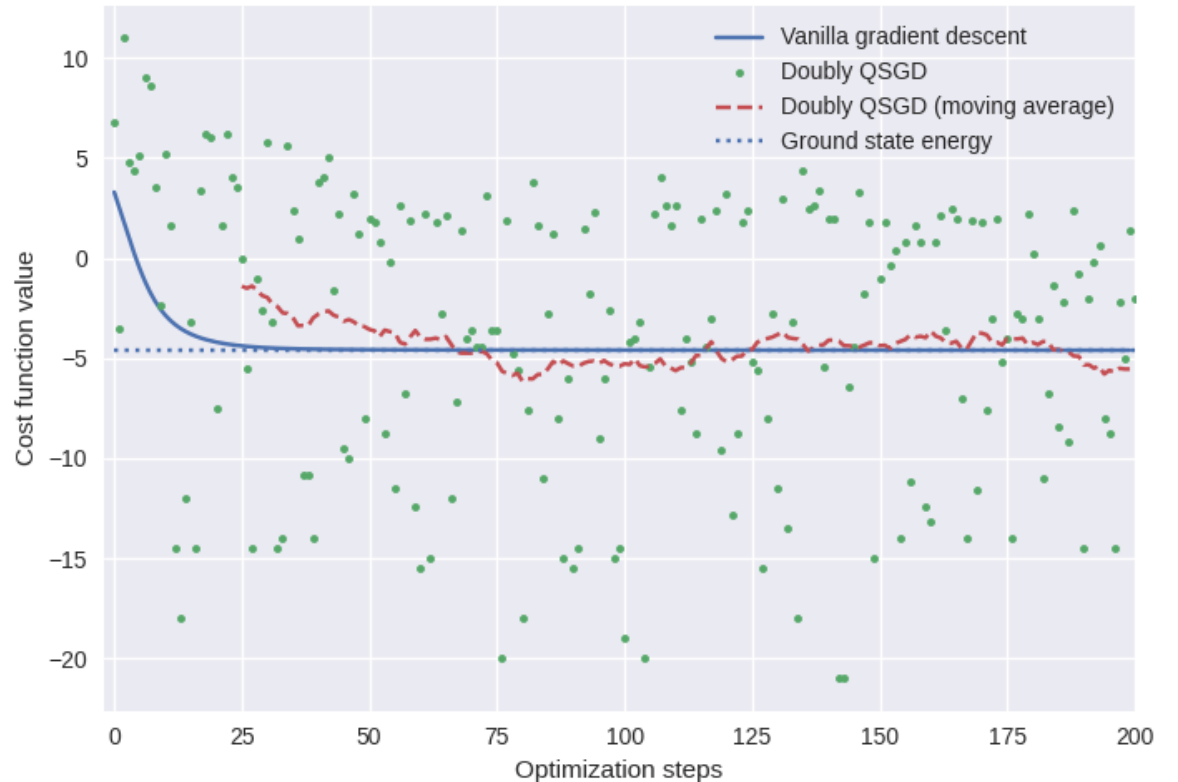


Fig 6. VQE doubly stochastic GD convergence curve

# Quantum Neural Networks (QNN)

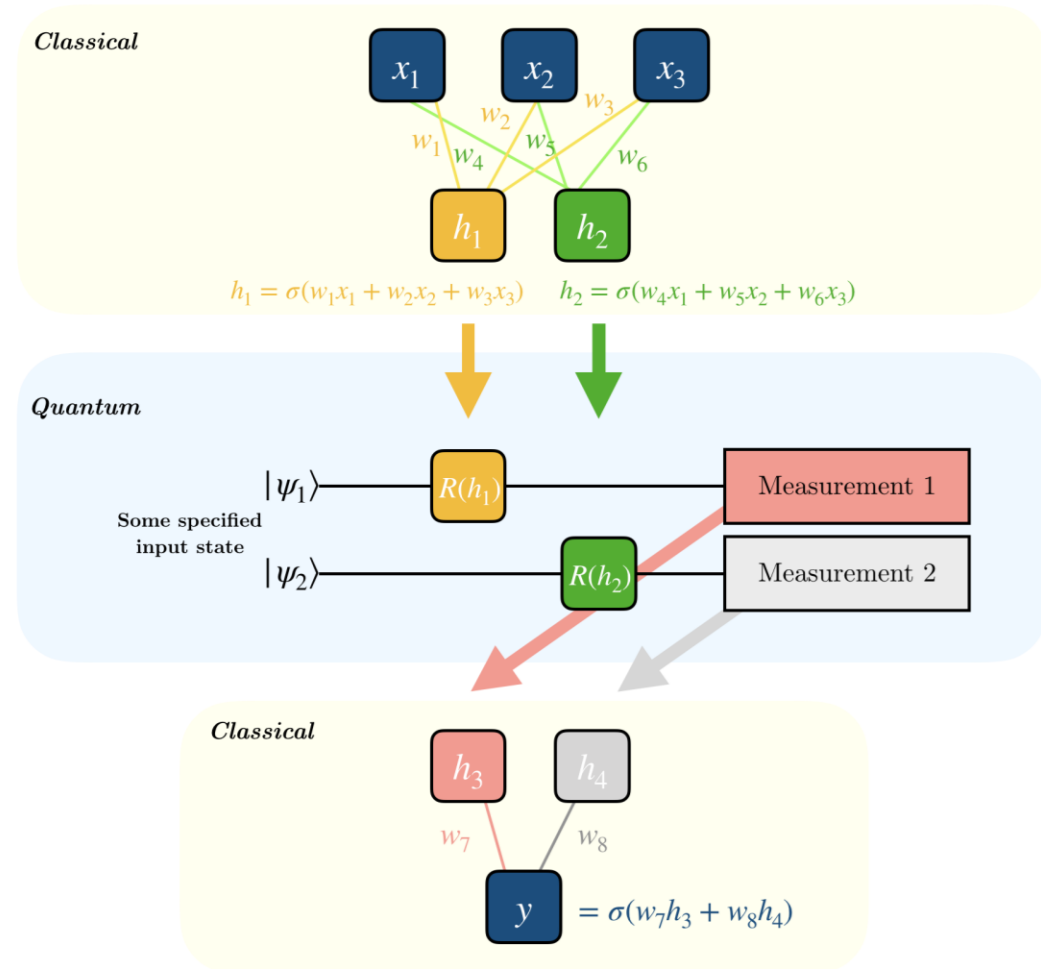


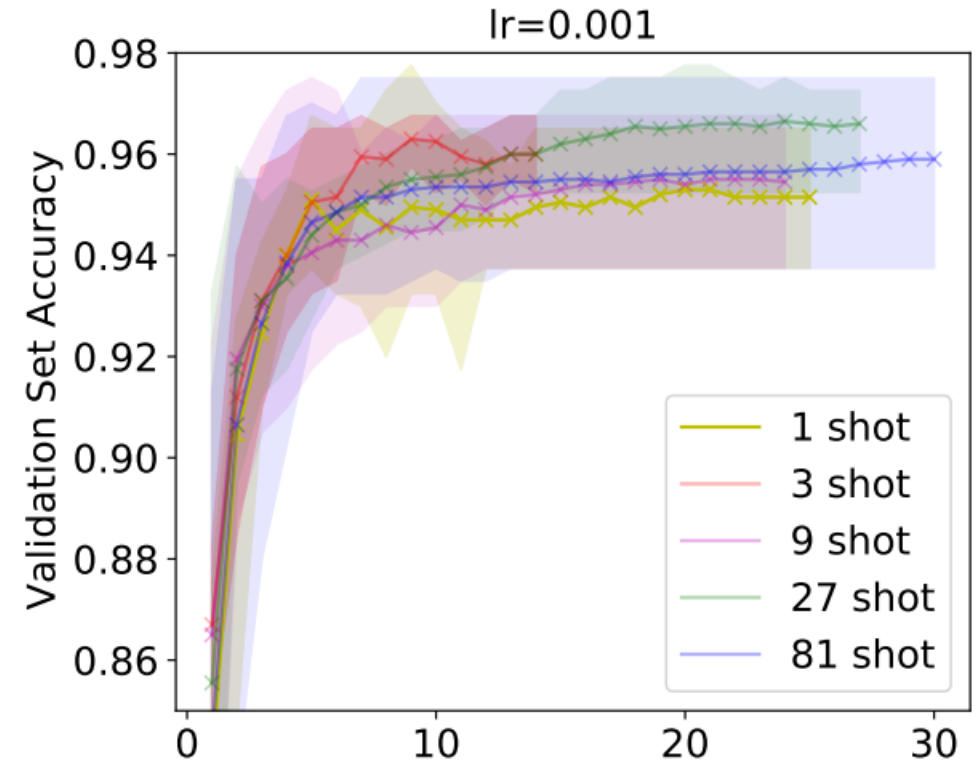
Fig 7. Architecture of a QNN, [IBM](#)

- We use Hybrid quantum-classical Neural Network as a VQA in near-term quantum devices
- It consists of classical quantum ansatz sandwiched in-between classical Neural Network layers.
- The output from other layers are fed as the parameters to the quantum ansatz
- The measurement values/expectation is fed to the input of the next classical layer
- Usually, the optimization is done in the classical layer.
- In this project, we have computed the optimization part using doubly stochastic gradient

# QNN: doubly stochastic gradient descent

- For this classification task, we used the MNIST dataset and took 2000 samples of the digit 3's and 6's as training set and 200 samples for validation set.
- We encode the data to the quantum circuit using a technique called [amplitude encoding](#)
- We encode the data using 8 qubits
- After the sampling the data, we also sample the terms used in the parameter-shift rule for implementing the doubly-stochastic gradient descent for the classifier

Observation: We can see that the accuracy improves as we increase the number of shots



# Conclusion

- In this project, we studied the doubly stochastic gradient descent using variational quantum circuits
- We simulated two problems such as the VQE and QNN for the MNIST dataset
- With the VQE problem:
  - In single shot SGD, the convergence of the ground state is very similar with the vanilla gradient descent
  - In doubly stochastic GD, the convergence is very similar to vanilla GD, with a moving average
- With QNN, we saw the accuracy is improved as the number of shots is increased
- The above simulations showed that we are able to implement the stochastic gradient descent using quantum circuits
- While we get almost similar convergence properties with vanilla gradient descent, the advantages of quantum computation come in terms of speed, when we have noiseless quantum computers in the future