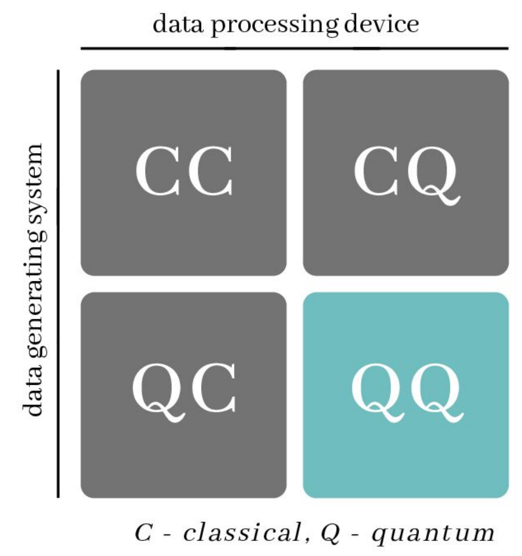
Overview of Quantum Machine Learning

QCB
Fall 2022



- Why it works?
- How it works?
- When it works?

Let NATURE do the work

Why it works? (Part 1/2)

Statistical Mechanics + Machine Learning $\Rightarrow$ QML?

Brief Stat Mech Intro

- Have a system of particles, can either be in spin-up (Energy = E) and spin-down (Energy = 0) states
- Place the particles in an environment with thermal energy U0 and temperature T
- Find the average energy of the system

Derivation (Kittel, *Thermodynamics*)

- Using function g (gives the number of arrangement to achieve a certain energy) and entropy (σ = ln(g))
  → Find the probability ratio of finding a state with energy E versus 0

$$\frac{P(\varepsilon)}{P(0)} = \frac{g(U_0 - \varepsilon)}{g(U_0)} = \frac{\exp[\sigma(U_0 - \varepsilon)]}{\exp[\sigma(U_0)]}.$$

- Taylor expand σ because E is small w.r.t. U

$$\sigma(U_0 - \varepsilon) \simeq \sigma(U_0) - \varepsilon(\partial\sigma/\partial U_0) = \sigma(U_0) - \varepsilon/\tau$$

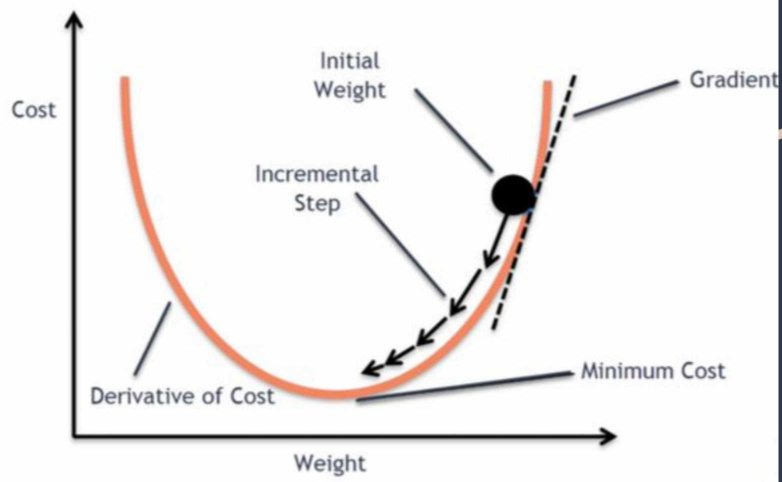
- Simplify the probability ratio

$$P(\varepsilon)/P(0) = \exp(-\varepsilon/\tau).$$

- Find the expectation value of the energy

$$\langle\varepsilon\rangle = \sum_i \varepsilon_i P(\varepsilon_i) = 0 \cdot P(0) + \varepsilon P(\varepsilon) = \boxed{\frac{\varepsilon \exp(-\varepsilon/\tau)}{1 + \exp(-\varepsilon/\tau)}}$$

Brief ML Intro (1-layer NN Classifier/Logistic Regression)



- Given input ***x***, want to output a result ***y*** that matches the truth values
- Approach
  - Initialize (pseudo)random vector $\boldsymbol{\beta}$
  - Compute an output ***y-pred (or Oi)*** by $\boldsymbol{\beta} \cdot \boldsymbol{x}$
  - If more than one layers, then repeat the computation above with non-linear function in between
  - Compute a loss using the Cross Entropy Function

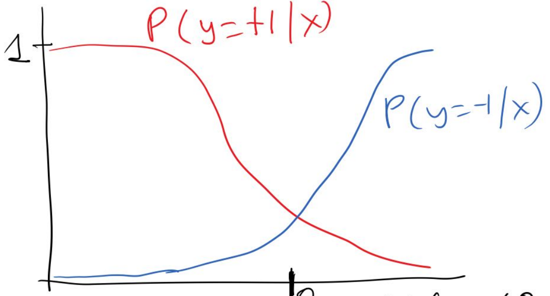
$$L = -\sum_{\text{input data}} (y_i \ln O_i + (1-y_i) \ln(1-O_i))$$

  - Update $\boldsymbol{\beta}$ by using Gradient Descent

$$\beta_i \leftarrow \beta_{i-1} - \alpha \cdot \nabla L(\beta_{i-1})$$

  - Stop when *L* is minimized (or $\boldsymbol{\beta}$ no longer changing)

Derivation (Malik, *CS 189*)



So should we guess $y=+1$ for $x<\theta$
-1 for $x>\theta$

- Assume many input samples → **x** is Gaussian
- Output two probabilities P(y=+1|x) and P(y=-1|x), decide y based on them
- Want to distinguish P(y=+1|x) and P(y=-1|x) = 1 - P(y=+1|x) as much as possible → maximize P(y=+1|x)/(1 - P(y=+1|x)) ⇔ max ln{P(y=+1|x)/(1 - P(y=+1|x))}

$$\ln \frac{p(C_2|x)}{p(C_1|x)}$$

$$= -\frac{1}{2}(x-\mu_2)^T \Sigma^{-1}(x-\mu_2) + \frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1) + \ln\frac{\pi_2}{\pi_1}$$

$$= \mu_2^T \Sigma^{-1} x \underbrace{-\frac{1}{2}\mu_2^T \Sigma^{-1}\mu_2}_{} - \mu_1^T \Sigma^{-1} x \underbrace{+\frac{1}{2}\mu_1^T \Sigma^{-1}\mu_1 + \ln\frac{\pi_2}{\pi_1}}_{}$$

$$= \beta^T x + \alpha$$

$$\frac{p}{1-p} = e^{\beta^T x + \alpha}$$

$$p = e^{\beta^T x + \alpha} - p\, e^{\beta^T x + \alpha}$$

$$p = \frac{e^{\beta^T x + \alpha}}{1 + e^{\beta^T x + \alpha}} \quad \text{or}$$

$$P(Y=1 \mid x) = \frac{1}{1 + \exp(-\beta^T x)}$$

$$P(Y=1 \mid x) = \mu(x)$$

Derivation (Malik, CS 189)

- Redenote P for simplicity

$$P(Y=1 \mid x) = \frac{1}{1+\exp(-\beta^T x)} \qquad P(Y=1 \mid x) = \mu(x)$$

- Calculate the case for general P(y|x) and total probability for n samples

$$P(y \mid x) = \mu(x)^{y} (1-\mu(x))^{1-y}$$

$$P(y_1, \dots y_n \mid x_1, \dots x_n, \theta) = \prod_{i=1}^{n} \mu_i^{y_i} (1-\mu_i)^{(1-y_i)}$$

- Maximize the log of probability for simpler calculation

$$\ell(\theta \mid D) = \sum_i y_i \ln \mu_i + (1-y_i) \ln (1-\mu_i)$$

- Flip the sign and turn max into min

$$L = -\sum_{\text{input data}} (y_i \ln O_i + (1-y_i) \ln(1-O_i))$$

Comparison

$$\langle \varepsilon \rangle = \sum_i \varepsilon_i P(\varepsilon_i) = 0 \cdot P(0) + \varepsilon P(\varepsilon) = \boxed{\frac{\varepsilon \exp(-\varepsilon/\tau)}{1 + \exp(-\varepsilon/\tau)}}$$

$$U \equiv \langle \varepsilon \rangle$$

$$1 - U/\varepsilon = ?$$

$$L = -\sum_{\text{input data}} \left(y_i \ln O_i + (1-y_i) \ln(1-O_i) \right)$$

$$\ell(\theta | D) = \sum_i y_i \ln \mu_i + (1-y_i) \ln(1-\mu_i)$$

$$P(Y=1 | x) = \boxed{\frac{1}{1+\exp(-\beta^T x)}}$$

Energy in Quantum?

$$\hat{\mathrm{H}}\,|\Psi\rangle = E|\Psi\rangle$$

Minimal Energy?

⇒ Ground State!

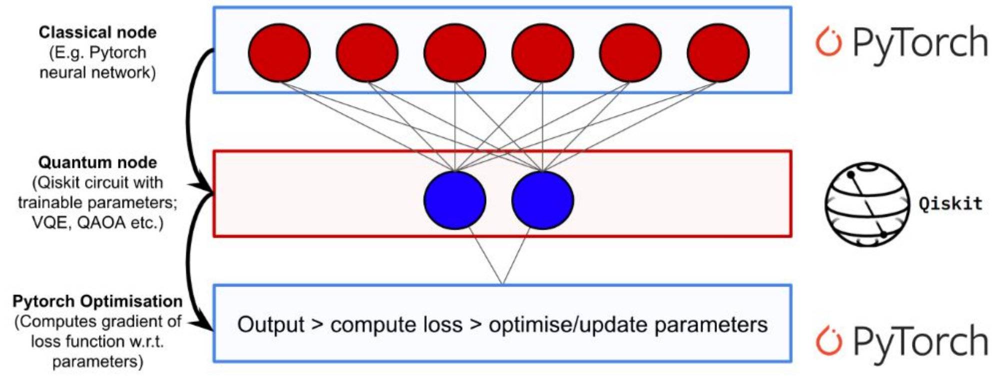
⇒ How to get there?

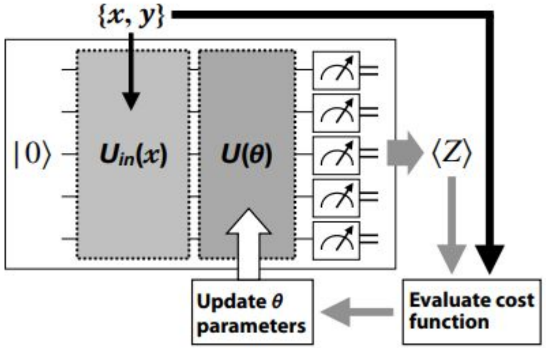
Variational Algorithm & Quantum Annealing

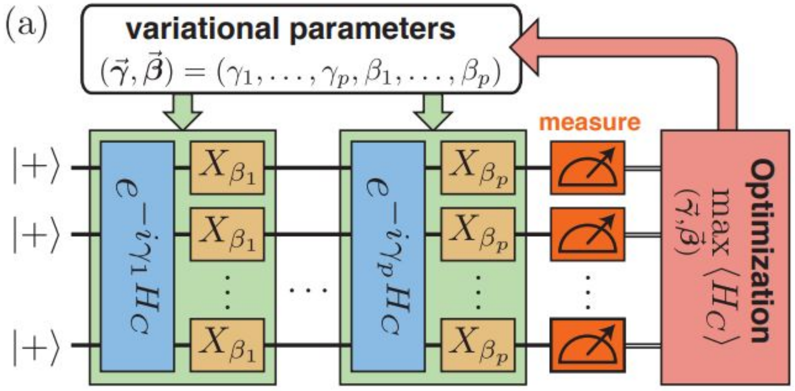
How it works? (Part 1/2)

Variational Algorithm

- Trainable parameters are the rotation degrees in the gates
- Hyperparameters include the circuit depth, rotation axis and entanglement scheme
- Often use a classical device to tune and optimize the parameters
- Can be viewed as a layer in NN

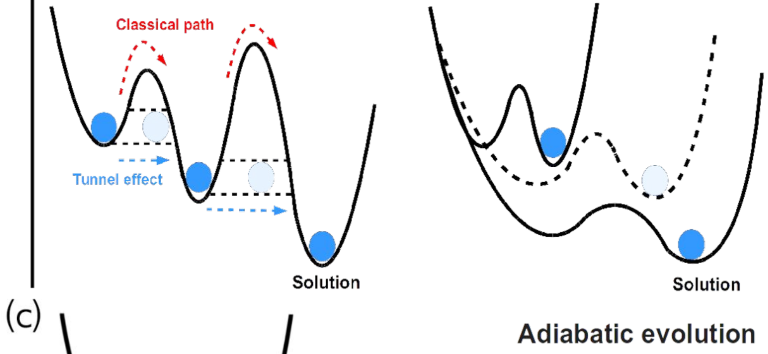


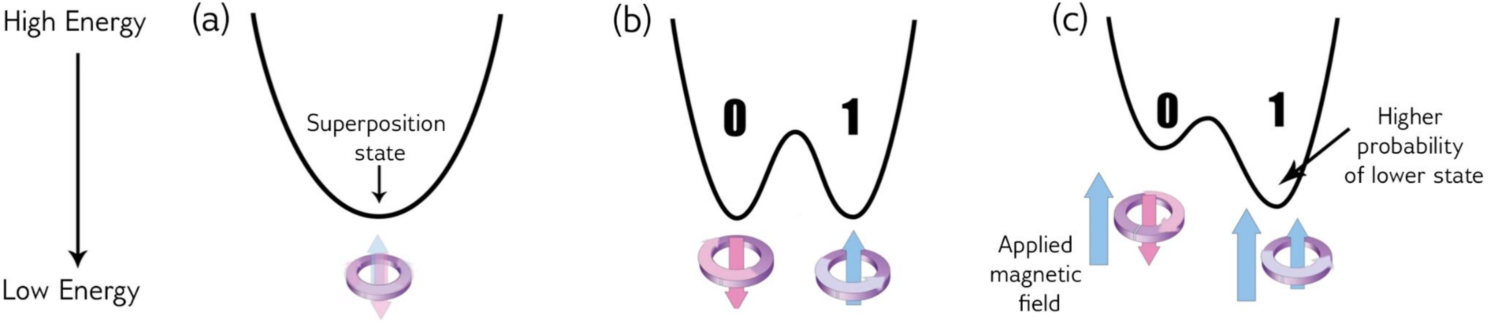




Quantum Annealing

- Device-dependent and task-specific
- Sometimes compared with Photonics Circuits
- Does not use gates
- Use magnetic spins as qubits and use fields to let them evolve in time and maintain in the ground state (adiabatic process)





Other methods

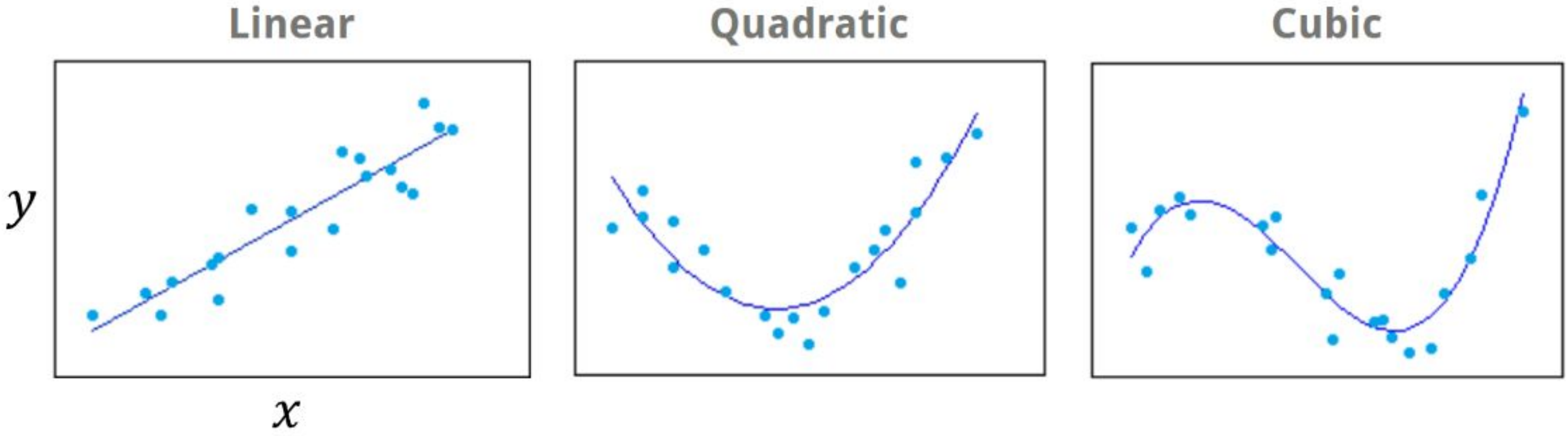
- Single-shot solution to linear system → HHL Algorithm
- Quantum Boltzmann Machine → Two layer NN but uses adiabatic/annealing processes to minimize the energy/loss

Everything can be LINEAR

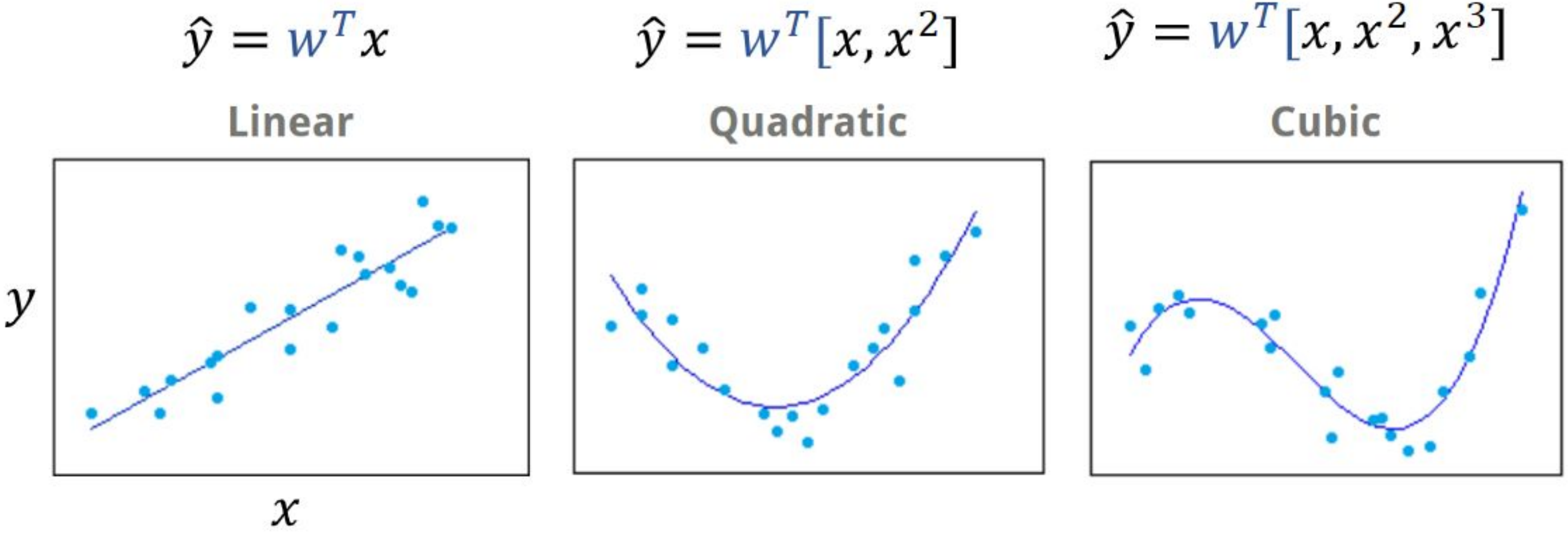
Why it works? (Part 2/2)

Neural Networks rely on Nonlinearities?

Can you use a linear model?



Can you use a linear model?



If $x = [x_1, x_2]$, then for quadratic features, we get $[x_1, x_2, x_1^2, x_2^2, x_1 x_2]$, etc.

SVM & PCA

How it works? (Part 2/2)

Quantum Support Vector Machine (SVM)

- Simplest supervised machine learning algorithms:
  - Linear SVMs
  - Perceptrons
- Quantum SVM $\rightarrow$ canonical example for QML techniques
  1. Data input (qRAM or other subroutine)
  2. Process data with Quantum Phase Estimation and Matrix Inversion
- Operations to construct hyperplane take $\log N$

Quantum Principal Component Analysis

- Principal Component Analysis used to compress our data's representation
- Simplest form $\rightarrow$ diagonalizing the covariance matrix:

$$C = \sum_k e_k c_k c_k^\dagger$$

- Performing qPCA on classical data:
  - Use qRAM (quantum Random Access Memory) - classical data vector gets mapped to quantum state ($v_j \mapsto |v_j\rangle$)

Before we continue... What is qRAM?

- We know Random-Access Memory (RAM) uses "n bits to randomly address $N = 2^n$ distinct memory cells" [GLM08]
- quantum Random-Access Memory (qRAM) theoretically uses "n qubits to address any quantum superposition of N memory cells" [GLM08]
    - Large qubit-overhead $\implies$ Not feasible in near-term
    - Costly memory call

Quantum Principal Component Analysis (Cont.)

- Suppose vectors live in d-dimensional space so that $d = 2^n = N$
- Principal components:

$$v = \sum_k v_k c_k$$

- Classical time complexity $\rightarrow \mathcal{O}(d^2)$
- Quantum time complexity $\rightarrow \mathcal{O}[(\log N)^2]$
    - Quantum state has $\log d$ qubits

How much faster anyways?

Method	Speedup	Amplitude amplification	HHL	Adiabatic	qRAM
Bayesian inference[106,107]	$O(\sqrt{N})$	Yes	Yes	No	No
Online perceptron[108]	$O(\sqrt{N})$	Yes	No	No	Optional
Least-squares fitting[9]	$O(\log N)$*	Yes	Yes	No	Yes
Classical Boltzmann machine[20]	$O(\sqrt{N})$	Yes/No	Optional/No	No/Yes	Optional
Quantum Boltzmann machine[22,61]	$O(\log N)$*	Optional/No	No	No/Yes	No
Quantum PCA[11]	$O(\log N)$*	No	Yes	No	Optional
Quantum support vector machine[13]	$O(\log N)$*	No	Yes	No	Yes
Quantum reinforcement learning[30]	$O(\sqrt{N})$	Yes	No	No	No

*There exist important caveats that can limit the applicability of the method[51].

When it works?

Well, obviously, when you have a usable quantum computer…

Applications

- Medical Diagnoses
- Logistical Optimizations
- Image Processing
- Speech Recognition
- Audio/Video Generation
- Recommender Systems
- Computational Sciences
- Controlling Hardwares
  - [Google Quantum Advantage] Learning from Experiments
- Etc…

Potential Areas to Explore

- Unsupervised Learning → Clustering algorithms
- Interpolation Regime → Generalizing well with random labelings?
- Adversarial Attacks → Robust against noise/wrong labels?
- Converging time → How many samples are needed to train?
- Performance on various devices → Can device impact performance?
- Representational Learning → More concise hidden representations
- Denoising models → Can model find the most important parts of data?
- Regression Models → Not a label, but a continuous number
- Link vs global classification → One task better than another?
- Deep learning → NLP/CV?
- Encoding Schemes → Best way to represent data?
- Etc…

TODO

- **Fill out the Survey Form**
- **Reminder: written project proposal with literature reviews and methods is required for re-joining the program next semester (likely 2 decal/course units)**
- Thought Experiment: How many qubits do you need to represent a 32-bit floating point number (assume between 0 and 1)?