MATHEMATICAL FOUNDATIONS OF QUANTUM MACHINE LEARNING

Roberto Moretti – XXXVIII cycle – 2CFU seminar

Summer School 10-14 Jul. 2023

UNIVERSITÀ DI TRENTO
Dipartimento di Matematica
Q@TN
TIFPA
Trento Institute for Fundamental Physics and Applications
PROVINCIA AUTONOMA DI TRENTO

25/10/2023

OUTLINE

- Qubits and modern quantum processors towards fault-tolerance.
- **Quantum gates and qubit control.**
- Fault-tolerant quantum advantage.
- Adiabatic quantum computing and QAOAs.
- Classical Machine Learning.
- **Variational algorithms**
  - Example: NISQ – Quantum Machine Learning.
  - Boosting gradient descent with the **Parameter-Shift rule**.
- Machine Learning with kernel methods.
- **Quantum Support Vector Machine.**
- Conclusions and outlook.

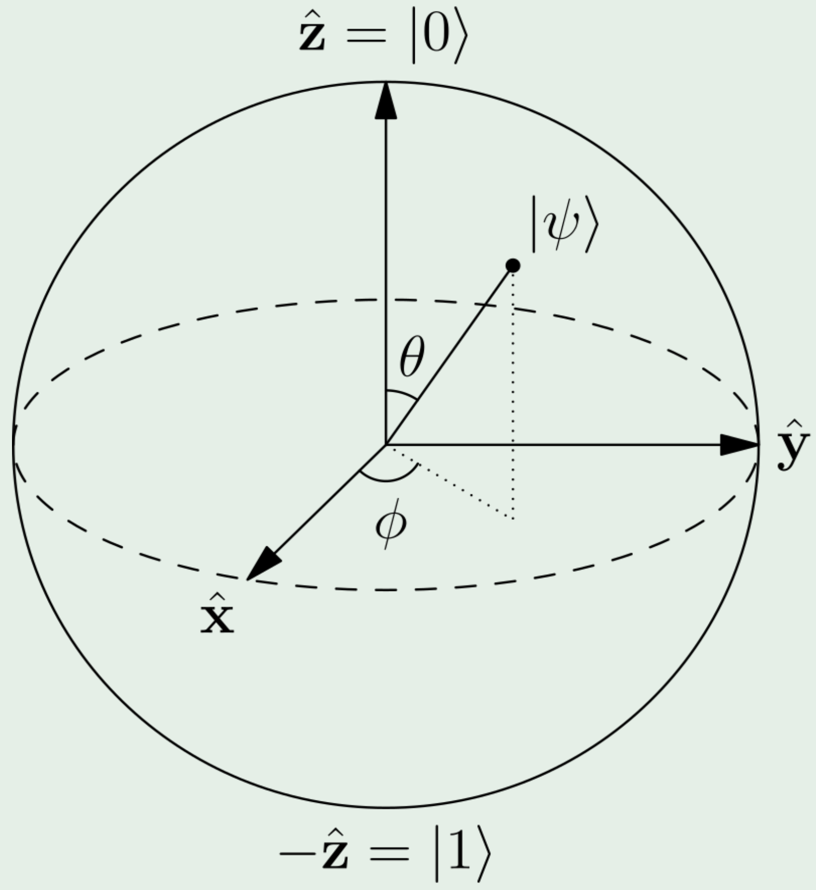
QUBITS

- Two-level quantum systems: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$.
- Quantum Computing fundamental unit of information.
- Quantum coherence allows to leverage the principles of Quantum Mechanics (superposition, entanglement) to achieve speedups in problem solving.

Example: a n-qubit system has 2^n eigenstates → 2^n complex amplitudes. Hard to simulate classically for $n \lesssim 50$ qubits.

$$|\psi\rangle = \left(\frac{1}{\sqrt{2}}\right)^n (|0\rangle + |1\rangle)^{\otimes n} = \left(\frac{1}{\sqrt{2}}\right)^n \sum_{i=1}^{n} |i\rangle$$

Bloch sphere parametrization



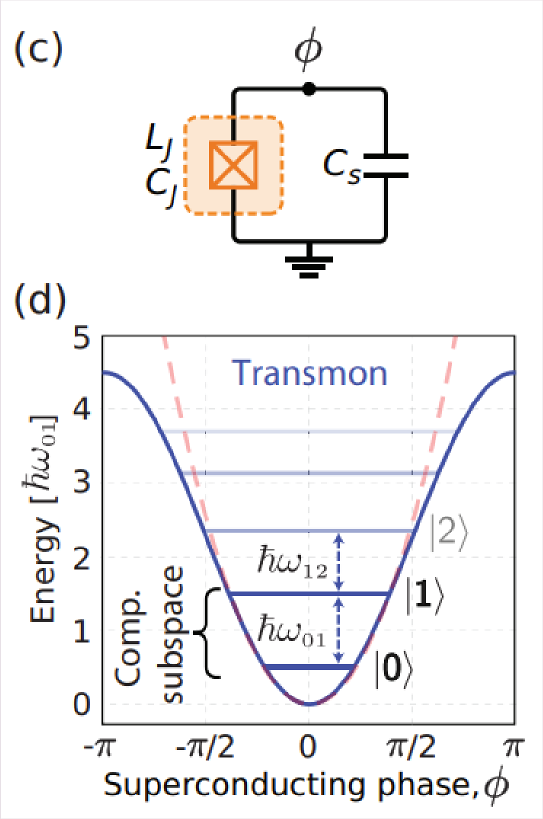
$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

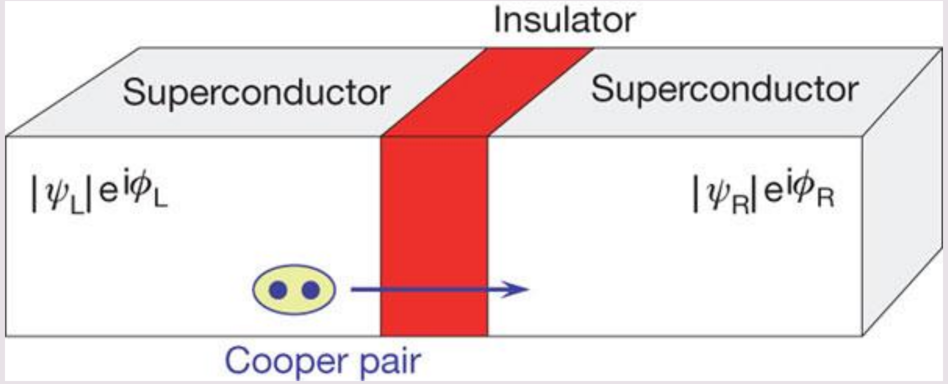
QUANTUM HARDWARE

- Among many platforms, the superconducting (SC) qubit holds as one of the most prominent one for achieving fault-tolerant quantum processing units.

- SC qubits are **anharmonic oscillators** with multiple energy levels, in which we are able to isolate the first two.

- Several issues:
  - Relaxation and dephasing.
  - Cryogenic temperatures required (~ 20 mK).
  - Hard to achieve scalable architectures.
  - Non-standard chip fabrication techniques required.





Development Roadmap

Executed by IBM ✓
On target

IBM Quantum

2019 ✓	2020 ✓	2021 ✓	2022 ✓	2023	2024	2025	2026+
Run quantum circuits on the IBM cloud	Demonstrate and prototype quantum algorithms and applications	Run quantum programs 100x faster with Qiskit Runtime	Bring dynamic circuits to Qiskit Runtime to unlock more computations	Enhancing applications with elastic computing and parallelization of Qiskit Runtime	Improve accuracy of Qiskit Runtime with scalable error mitigation	Scale quantum applications with circuit knitting toolbox controlling Qiskit Runtime	Increase accuracy and speed of quantum workflows with integration of error correction into Qiskit Runtime

Model Developers
- Prototype quantum software applications (on target) → Quantum software applications
  - Machine learning | Natural science | Optimization

Algorithm Developers
- Quantum algorithm and application modules ✓
  - Machine learning | Natural science | Optimization
- Quantum Serverless (on target)
  - Intelligent orchestration
  - Circuit Knitting Toolbox
  - Circuit libraries

Kernel Developers
- Circuits ✓
- Qiskit Runtime ✓
  - Dynamic circuits ✓
  - Threaded primitives (on target)
  - Error suppression and mitigation
  - Error correction

System Modularity
- Falcon ✓ 27 qubits
- Hummingbird ✓ 65 qubits
- Eagle ✓ 127 qubits
- Osprey ✓ 433 qubits
- Condor (on target) 1,121 qubits
- Flamingo 1,386+ qubits
- Kookaburra 4,158+ qubits
- Scaling to 10K-100K qubits with classical and quantum communication
- Heron (on target) 133 qubits x p
- Crossbill 408 qubits

Noise Intermediate Scale Quantum (NISQ) era:
Small/medium sized QPUs, significant error rate.

QUANTUM GATES

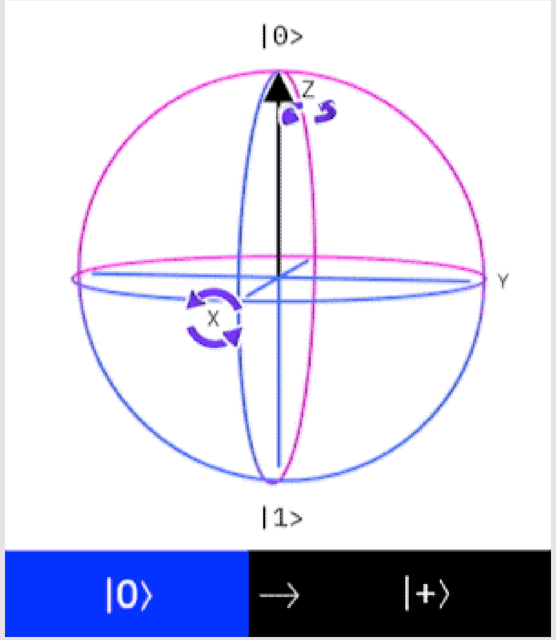
- Multi-qubit states are governed by **quantum gates**.
- The most common are single-qubit (can be visualized on the Bloch sphere) and two-qubit gates.
- A sequence of quantum gates acting on an initial state implements a **quantum algorithm**, often indicated as **quantum circuit**.

Single qubit gates are SU(2) rotations:

$$U(\theta,\phi,\lambda) = \begin{pmatrix} e^{-i\frac{\phi+\lambda}{2}} cos(\theta/2) & -e^{-i\frac{\phi-\lambda}{2}} sin(\theta/2) \\ e^{i\frac{\phi-\lambda}{2}} sin(\theta/2) & e^{i\frac{\phi+\lambda}{2}} cos(\theta/2) \end{pmatrix}$$

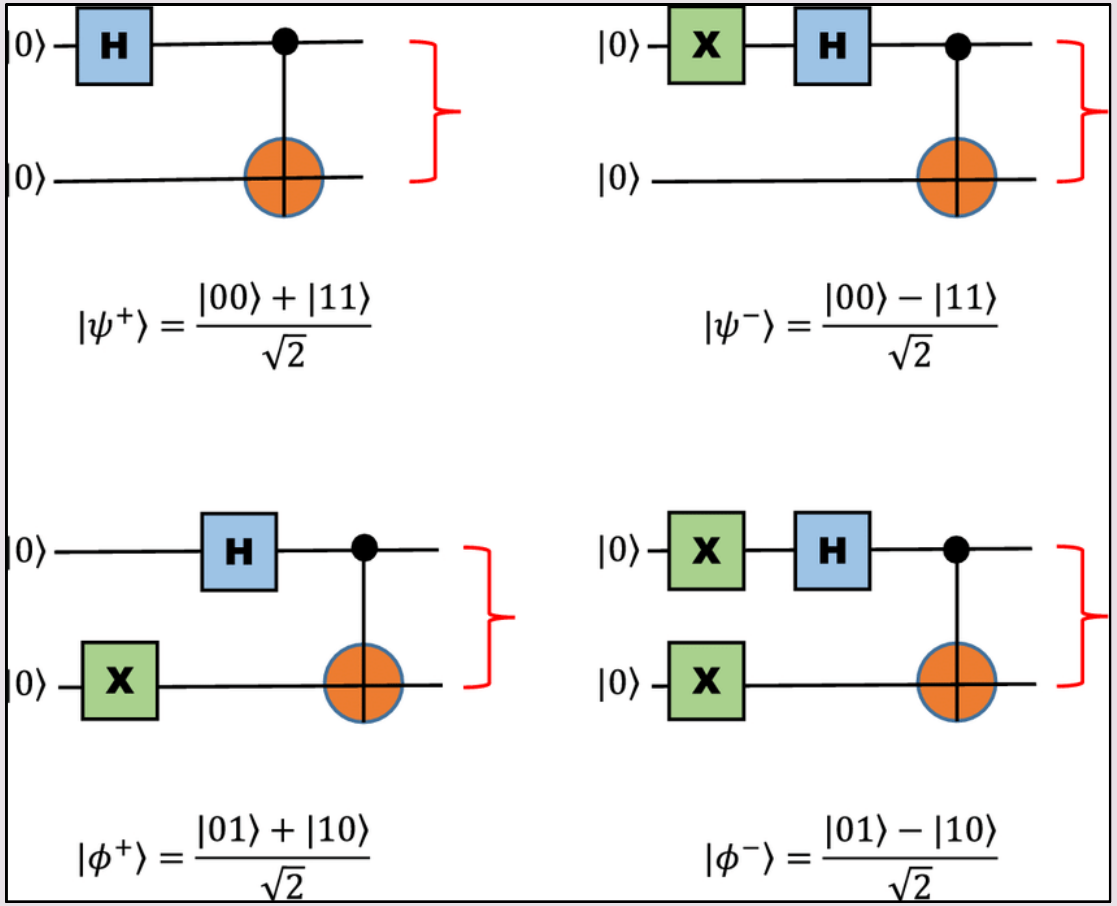
$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Name	Symbol	Matrix rep.	Action
Hadamard	H	$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	$H\|0\rangle = \frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$ $H\|1\rangle = \frac{1}{\sqrt{2}}(\|0\rangle - \|1\rangle)$
Not	X	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	$X\|0\rangle = \|1\rangle$ $X\|1\rangle = \|0\rangle$
Pauli rotations	RZ RX RY	$R_i(\theta) = e^{-\frac{i\theta}{2}\vec{\sigma}_i}$	Rotation along X, Y, Z axis on the Bloch Sphere
C-Not	⊕	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$CX\|\psi_0\rangle\|\psi_1\rangle = \|\psi_0\rangle\| \psi_0 \oplus \psi_1\rangle$



QUANTUM CIRCUIT EXAMPLES

Entangling two qubits with a quantum gate set



$$|\psi_0\rangle = |00\rangle$$

$$|\psi_1\rangle = H_0|00\rangle = (H|0\rangle)\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle$$

$$|\psi_2\rangle = CNOT\,|\psi_1\rangle = \frac{1}{\sqrt{2}}(\,|00\rangle + |11\rangle)$$

«Bell pairs»
Few gates can produce two-qubit, maximally entangled states.

Entanglement is what makes quantum advantage possible.
Otherwise, any quantum circuit would be easy to reproduce classically.

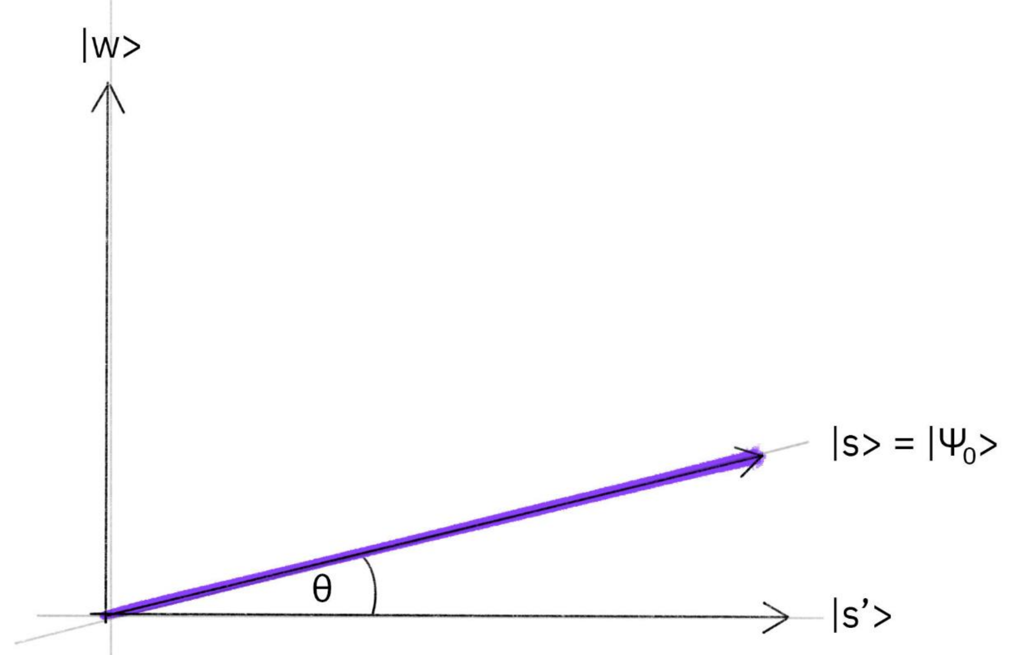
GROVER SEARCH I

A remarkable example of Quantum Advantage that will be achievable through fault-tolerant quantum computers.

- **Goal:** searching for an item in an unstructured database.

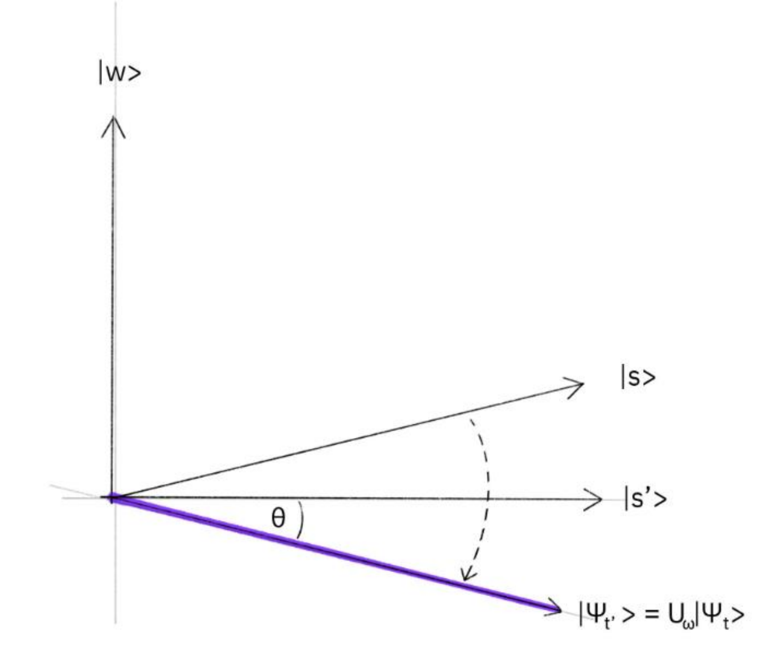
  Given a set X of N elements and a boolean function $f: X \rightarrow \{0,1\}$, finding $x^* \in X$ such that $f(x^*) = 1$.
  Let's assume that only one x^* exists and represent it with state $|\omega\rangle$, where all the other states are represented by $|s'\rangle$.

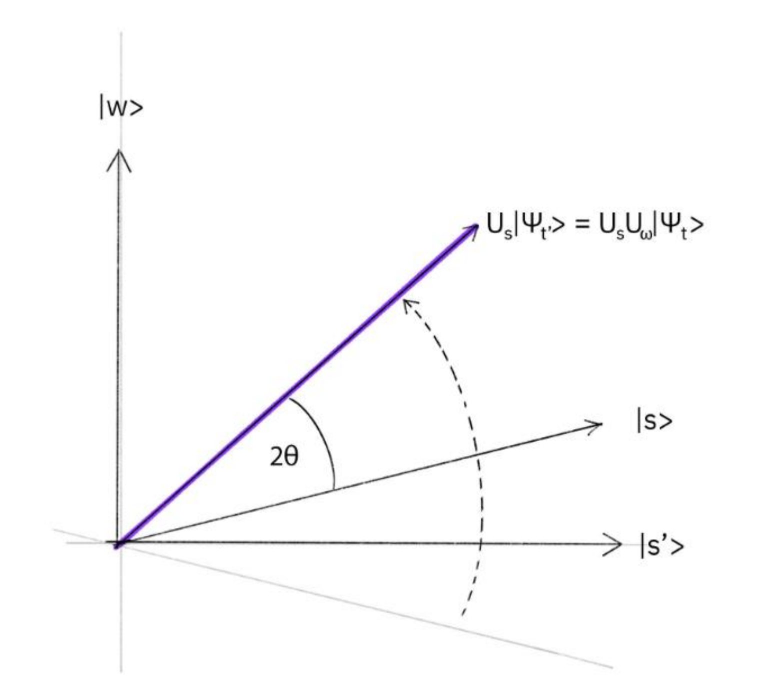
1. Apply a Hadamard gate to all the n qubits: $|s\rangle = H^n|0\rangle^n$. $|s\rangle$ will be almost perpendicular to $|\omega\rangle$ if the database is large enough.



GROVER SEARCH II

2. Apply an «oracle» operation U_ω which flips the amplitude sign associated with $|\omega\rangle$



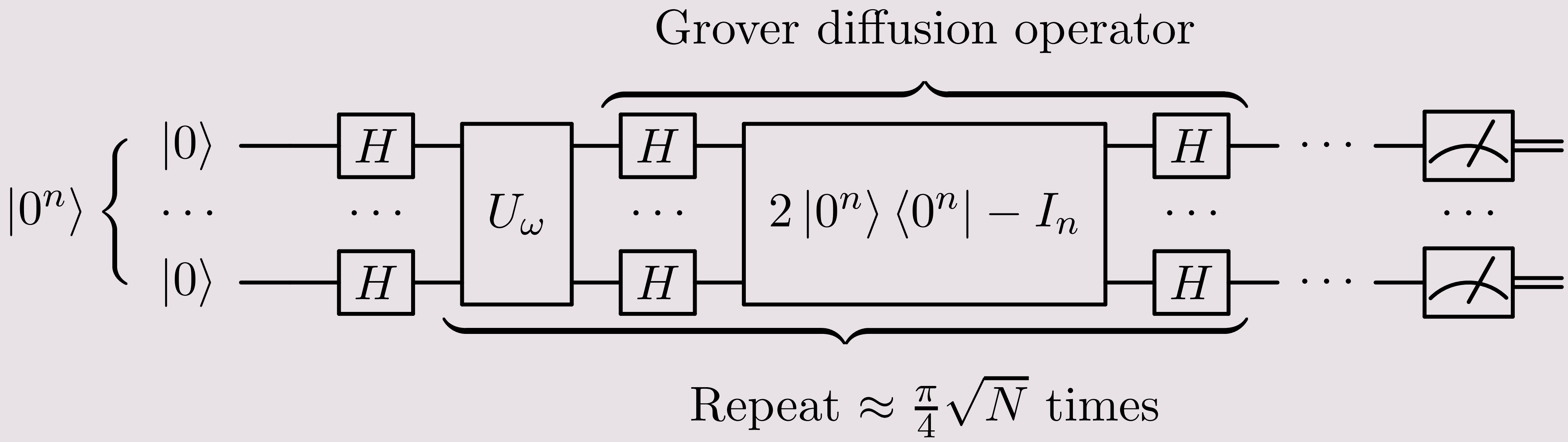


3. Apply a «diffusion» operation $U_s = 2|s\rangle\langle s| - I$ which applies a reflection about the $|s\rangle$ state.

The probability of measuring the state $|\omega\rangle$ as a measurement outcome is now increased.

- Iterating step 2 and 3 → $|\omega\rangle$ becomes more and more likely.

GROVER SEARCH III

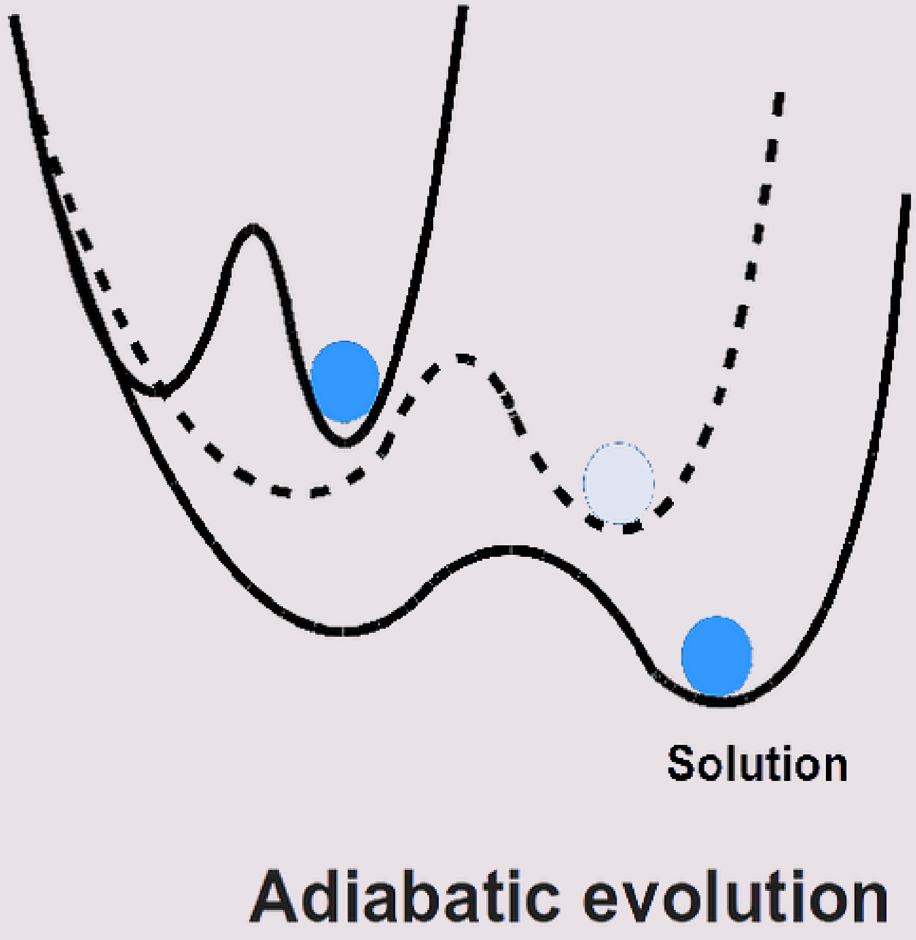


The number of gates scales as $\boldsymbol{O(\sqrt{N})}$. U_ω «checks» all the items in the dataset simultaneously, thanks to superposition.
Classically, to find the solution in an unsorted database we need to check elements one by one until we find the correct one, i.e. the number of checks scales as $\boldsymbol{O(N)}$.

→ Quantum Advantage for database, SAT problems, solving sudokus, etc...
But it requires many qubit, and is not robust to errors.

ADIABATIC QUANTUM COMPUTING

- Let's consider hamiltonians H_S and H_C, such that for H_S it is simple to prepare the ground state, while for H_C is complicated.
- Then we can consider the following time evolution:
  - $H\left(\frac{t}{T}\right) = \left(1 - \frac{t}{T}\right) H_S + \frac{t}{T} H_C$, where T is the total evolution time.
- If T is big enough, the qubit state initially prepares in the ground state for H_S will evolve to stay in the ground state of $H\left(\frac{t}{T}\right)$ (adiabatic limit).
- At $t = T$, we successfully prepared our qubits in the ground state of H_C.

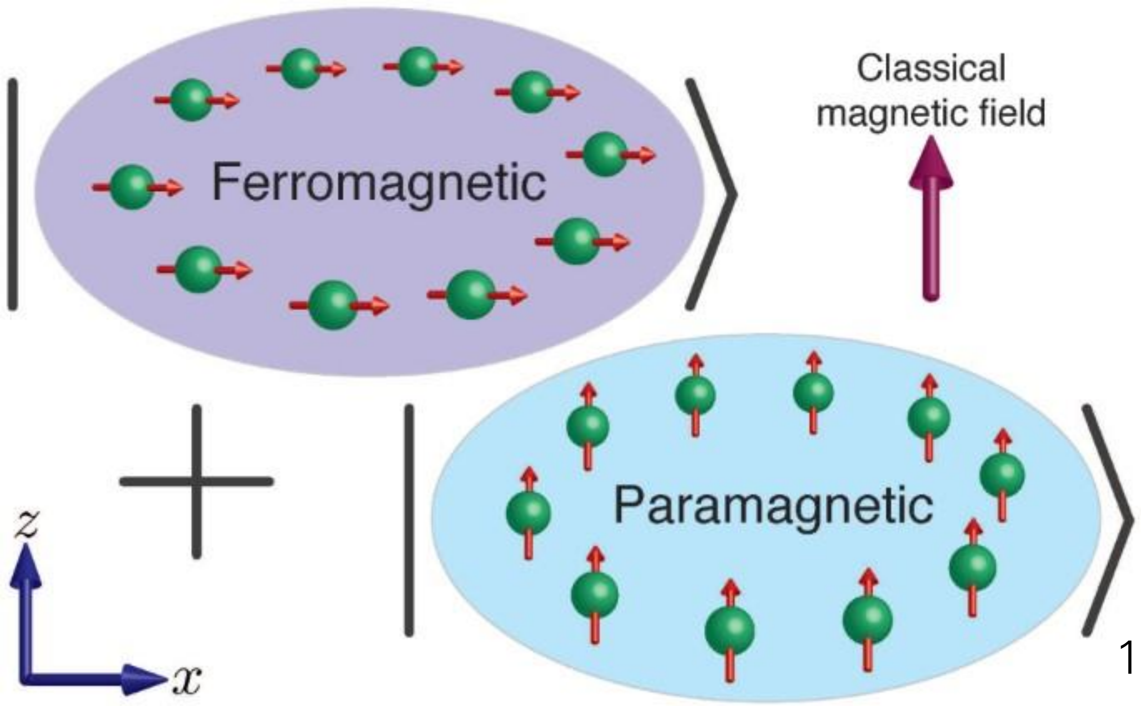


ADIABATIC QC IN PRACTICE

- If we can encode the solution to a problem in H_C, adiabatic quantum computing will be a powerful tool. Luckily, it is possible to define H_C to solve a some combinatorial problems and Ising models.
- Example: a chain of fermions and the interaction between their spins:
  - $H = \sum_{i,j} J_{ij} \sigma_i \sigma_j$ where J_{ij} is the interaction between particle i and j.
  - Many combinatorial problems can be encoded using Ising models (1302.5843).

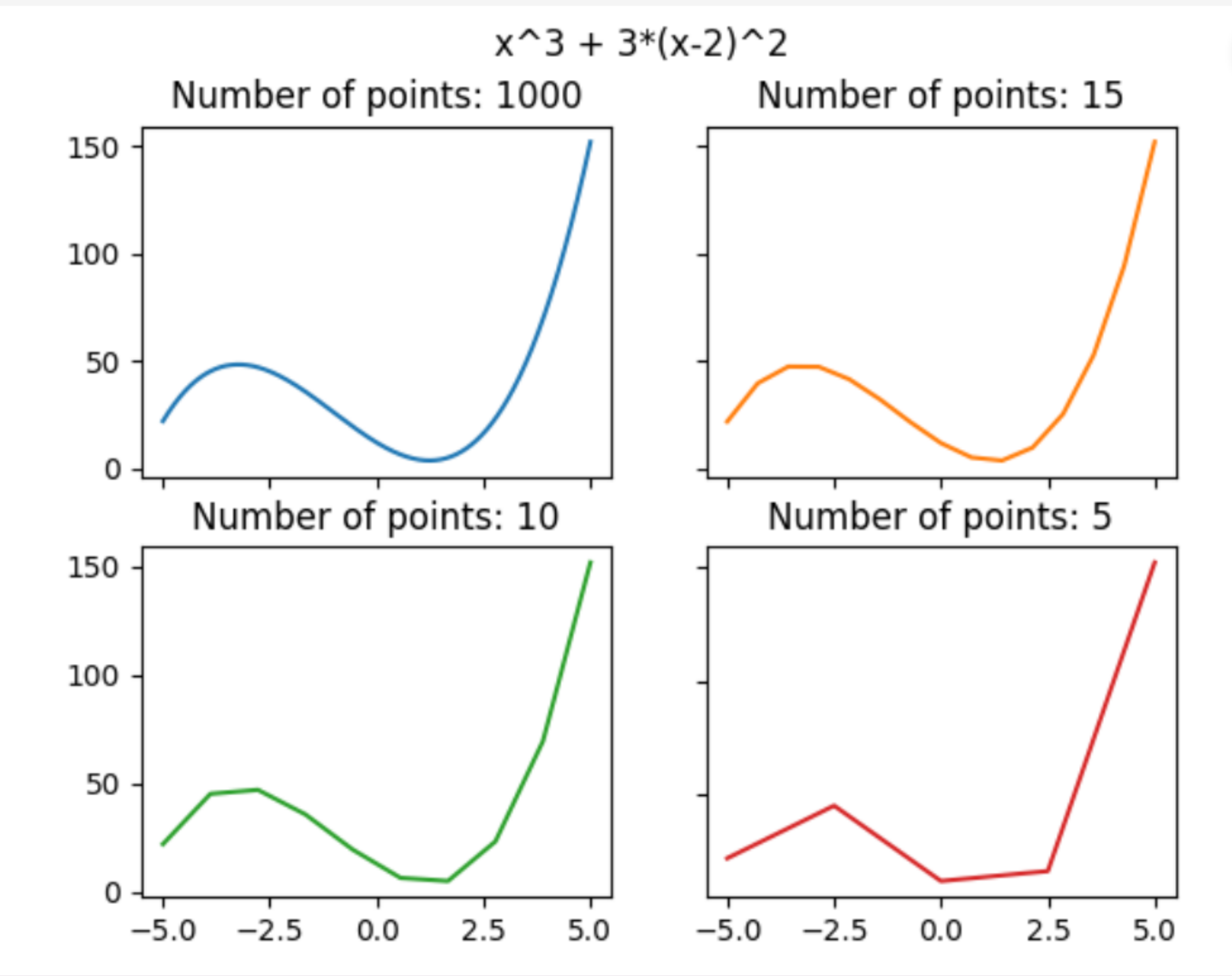
Problems:

- The quantum system must be extremely well isolated from the environment.
- T can be very long.
- Very hard to run this on gate-based quantum computers.



TROTTERIZATION

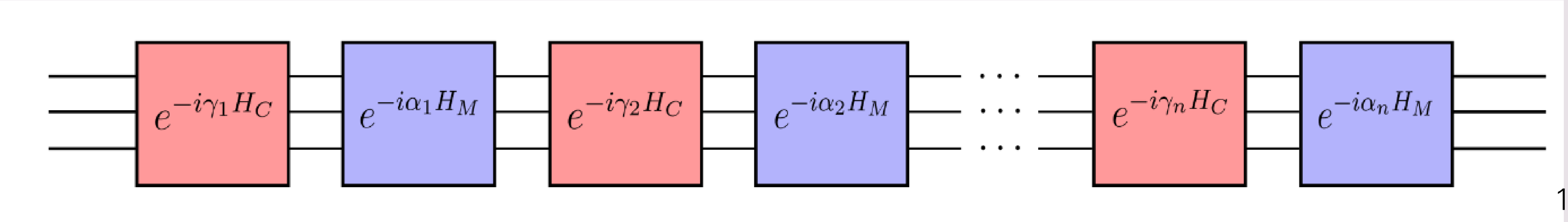
- A powerful tool for approximating hard-to-compute Hamiltonian ground states.
- If $H = H_A + H_B$, then:
  - $e^{H_A + H_B} = \lim_{n\to\infty} \left(e^{\frac{H_A}{n}} + e^{\frac{H_B}{n}}\right)^n$
- This can be interpreted as applying H_A and H_B for time intervals $\frac{1}{n}$ alternatively. A classical analogy is representing a curve with piecewise approximations.



QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM

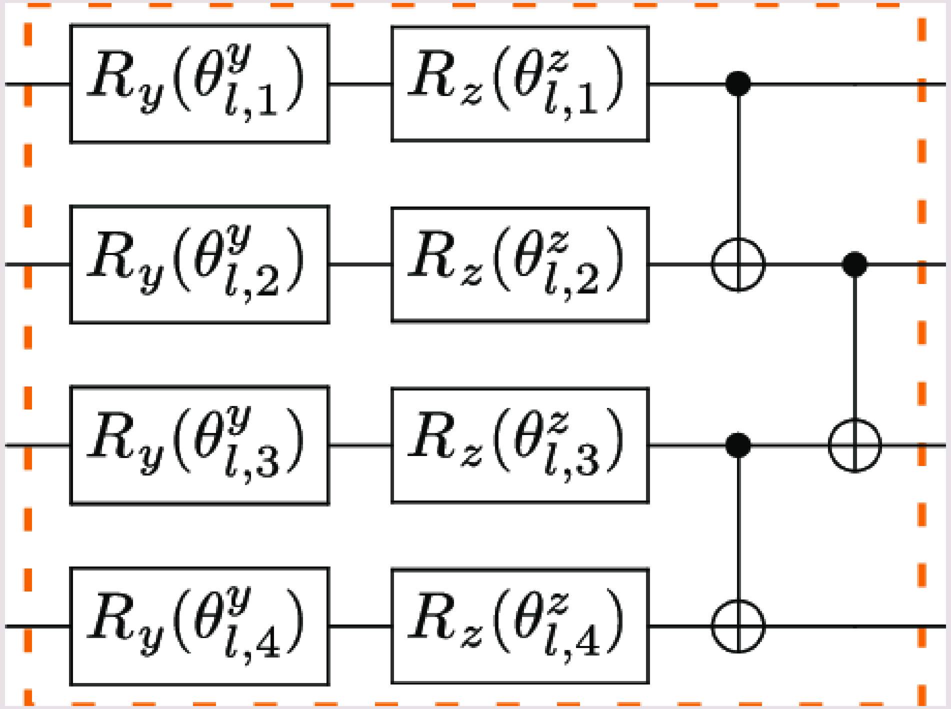
- We can put together the adiabatic computing idea and trotterization for designing a gate-based algorithm that solves combinatorial problems: QAOA.
- Idea: creating a state $|\gamma, \alpha\rangle = U(H_B, \alpha_p)U(H_C, \gamma_p) \ldots U(H_B, \alpha_1)U(H_C, \gamma_1)|s\rangle$
  - Where the U are the unitary evolution operators corresponding to the Hamiltonian H for time α or γ.
  - This can mimic the adiabatic evolution of a state from being the ground state of H_B to being the ground state of H_C, but αs and γs must be appropriately tuned for that.
  - H_C, the «cost» hamiltonian that we want to found the ground state of.
  - H_B must be a simple Hamiltonian, e.g. $\sum_i \sigma_i^x$, that does not commute with H_C. H_B helps us to not get stucked in eigenstates of H_C.

How can we tune the time-steps $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$?



NISQ-ERA PARAMETRIC QUANTUM CIRCUITS

- A prominent approach to quantum algorithms, typically robust to errors and suitable for NISQ devices.
- **General idea:** parametrizing quantum gates, allowing us to:
  - Minimize complicated cost functions
    - Quantum Neural Networks, Quantum Eigensolvers, ...
  - Encode classical data in large Hilbert spaces:
    - Amplitude/angle embedding, Quantum Kernels, ...

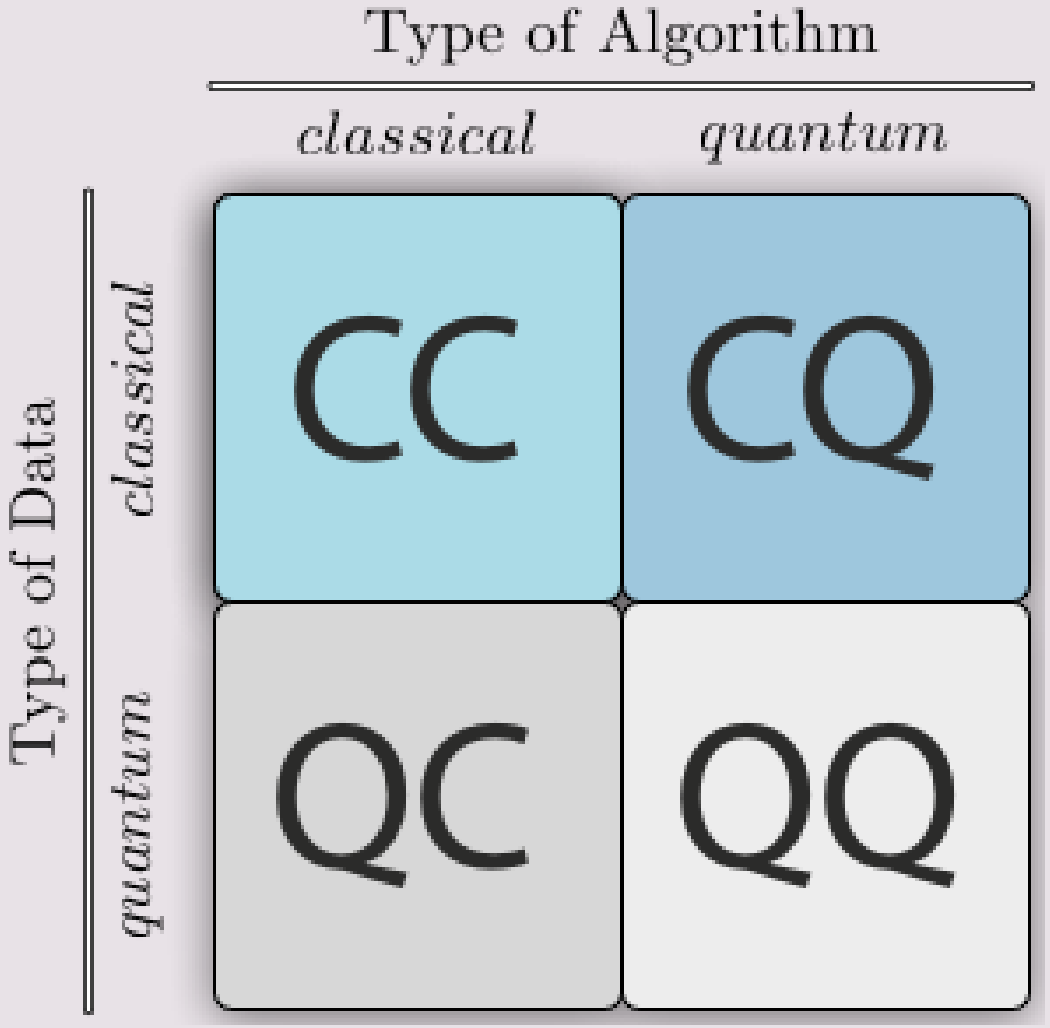


→ NISQ-era devices allow to implement **Quantum Machine Learning**

QUANTUM MACHINE LEARNING

A vast research field in rapid expansion.
Tipically divided into four cathegories:

- Classical, or quantum-inspired classical models for analysing classical data.
- Quantum models for analysing classical data
  - New approaches for data analysis in physics
- Classical, or quantum-inspired classical models for analysing quantum data.
- Quantum models for analysing quantum data
  - May prove to be useful in quantum sensing.

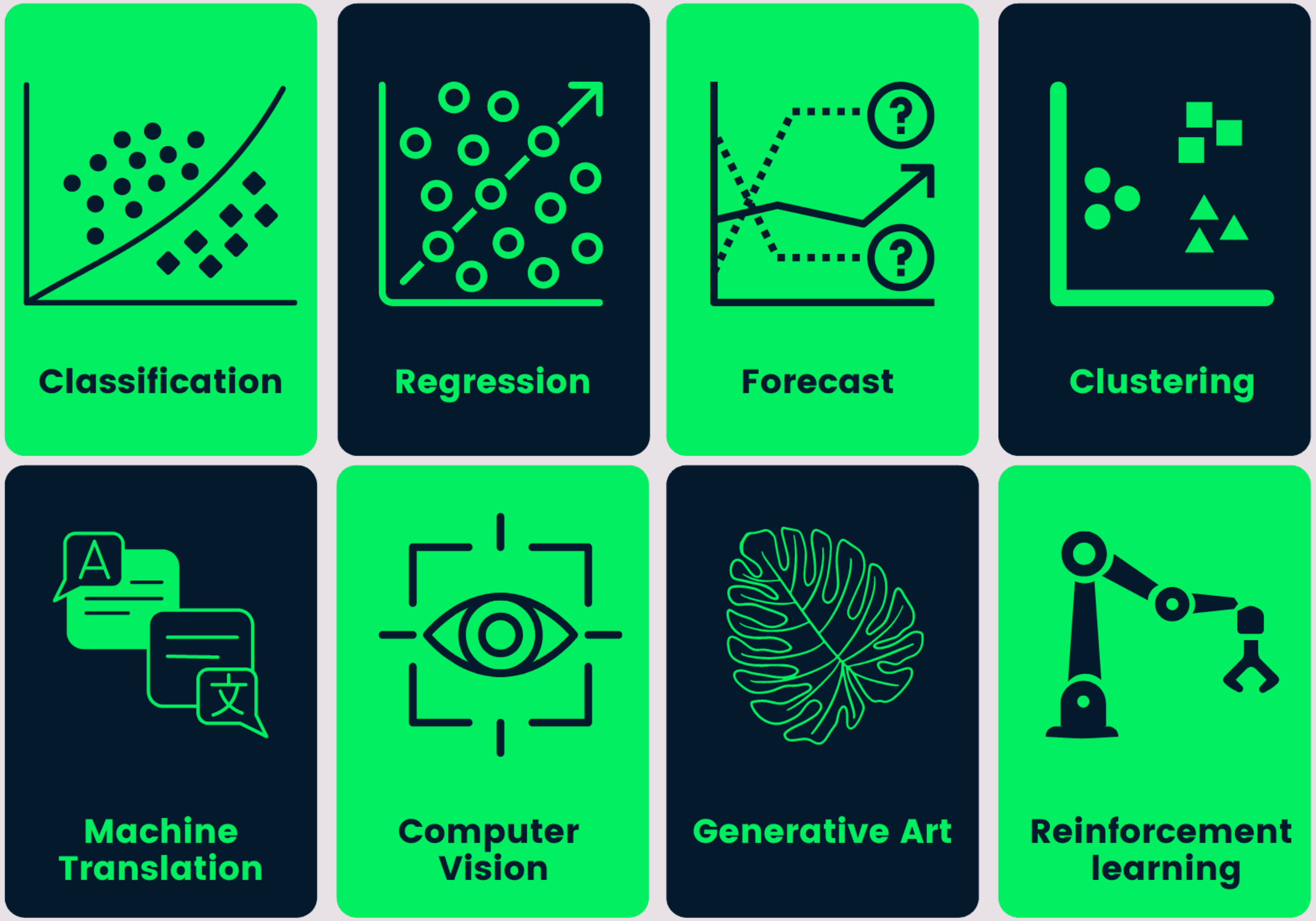


DIGRESSION: MACHINE LEARNING

Resolving problems without explicit programming through **data-oriented** approaches.

A very typical approach is to give a model **many degrees of freedom,** and looking for the best way to fix their values, i.e. Trying to minimize a cost function, which is related to how well a model solves the problem.

In a neural network, such degrees of freedom are the weights that connect the neurons trough different layers. Minimization occurs though **stochastic gradient descent**.



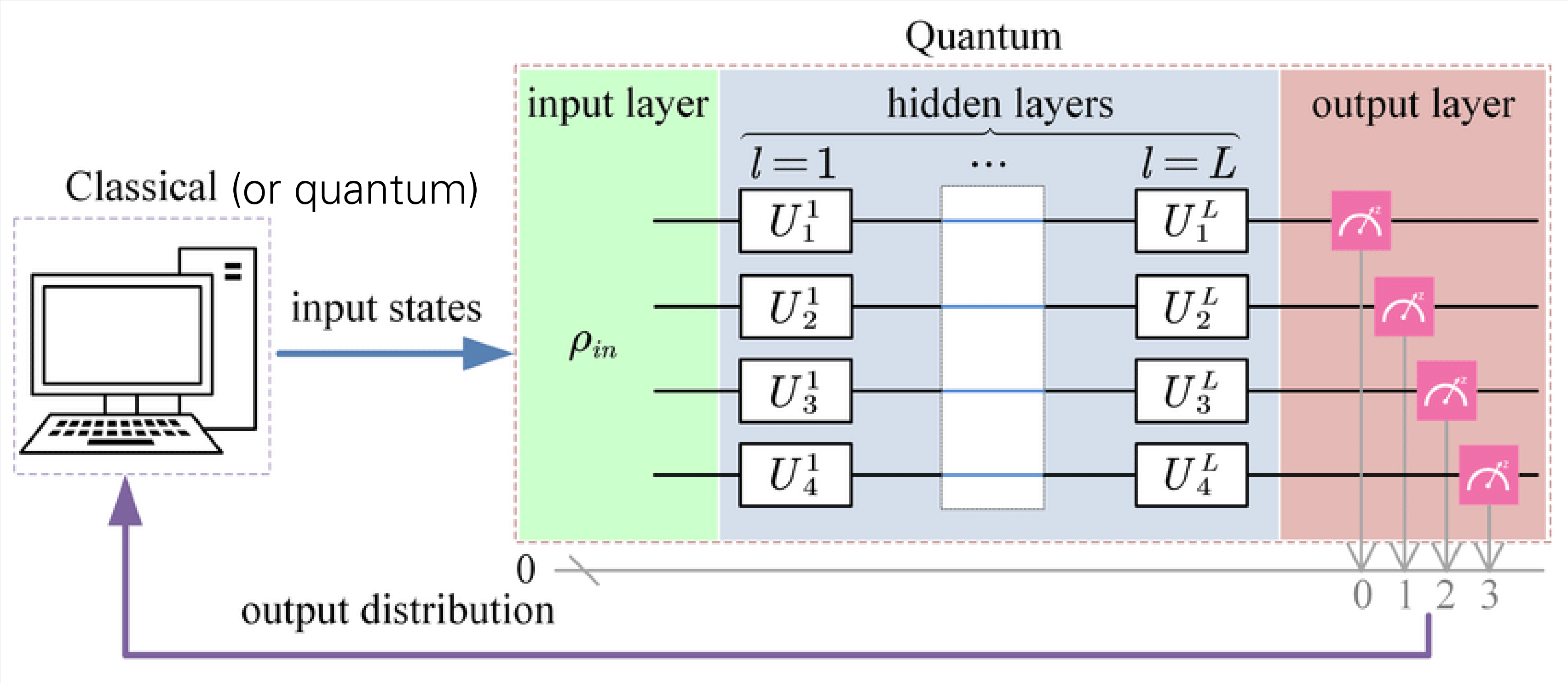
QUANTUM NEURAL NETWORKS

Recipee:

1. Encoding data in a quantum Hilbert space.
2. Weighted gates (quantum layers).
3. Output retrieved by running the circuit many times and averaging
4. Updating the weights via **classical** or **quantum** method.

Sometimes, classical NN layers may be useful (hybrid QNNs) for dimensionality reductions.

Quantum layers topology might be inspired by the physics of the process, or enforce particular rules for the system.



QNN LEARNING METHODS

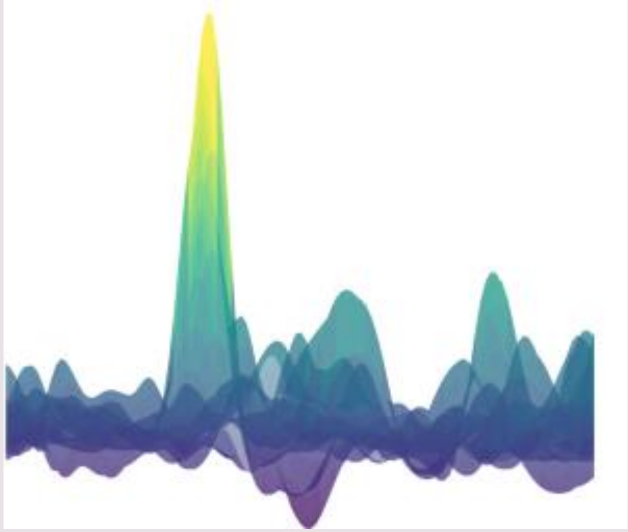
Finding the global minimum of a QNN loss function is nontrivial:

- **Complicated landscapes** with many local minima.
- Occurrence of **barren plateaus** when increasing the number of qubits.
- **Noise** of quantum device and statistical fluctuations from repeating the measurements.

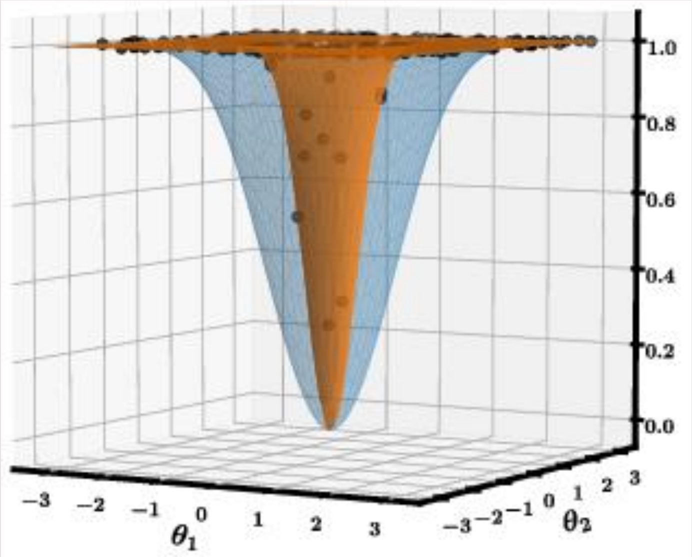
Possible approaches:

- Classical evaluation of stochastic gradient descent through finite difference
- Quantum evaluation of stochastic gradient descent through finite difference
- Quantum evaluation of stochastic gradient descent through exact gradient evaluation through the Parameter-shift rule.

Hard to minimize cost funciton



Barren plateau example



PARAMETER-SHIFT RULE

If we can express a function $g(x)$ as $g(x) = f(x;\theta_i)$ where x, θ_i are parameters of a quantum circuit, i.e:

$$f(x;\theta_i) = \langle 0|U_0^\dagger(x)U_i^\dagger(\theta_i)\hat{B}U_i(\theta_i)U_0(x)|0\rangle = \langle x|U_i^\dagger(\theta_i)\hat{B}U_i(\theta_i)|x\rangle$$

Then we can write the **exact** derivative of the function as:

$$\nabla_\theta f(x;\theta) = \frac{1}{2}\left[f(x;\theta+\tfrac{\pi}{2}) - f(x;\theta-\tfrac{\pi}{2})\right]$$

- Way more efficient than quantum finite difference method when minimizing cost functions in QNN.
- Other applications are possible, e.g. solving differential equations, evaluating integrals. Typically, θ parameters must be trained in turn to realize the $g(x) = f(x;\theta_i)$ equality.

SUPPORT VECTOR MACHINE

- Well-known Machine Learning model suited for binary and multilabel classification.

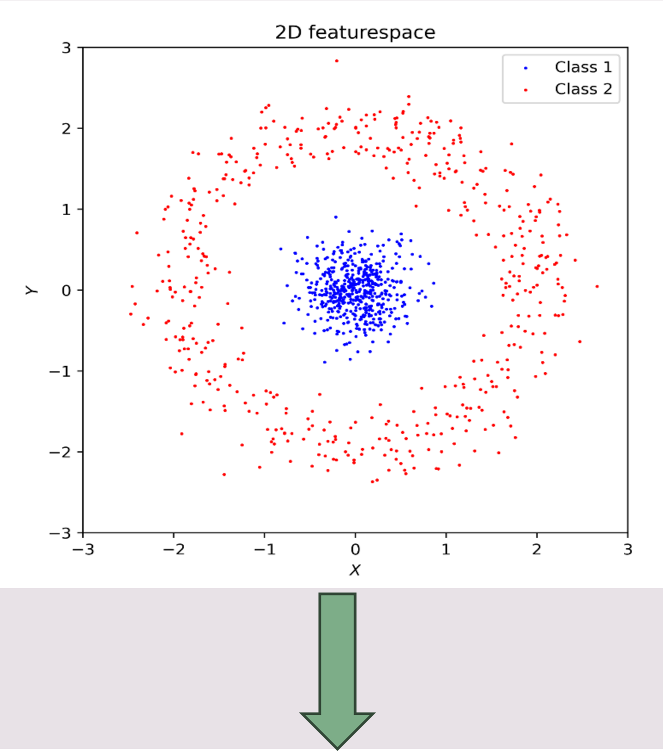
Task: binary classifications of feature vectors $\vec{\boldsymbol{x}} \in \mathbb{R}^{\boldsymbol{n}}$
i.e. predicting the class outcome $y \in \{-1;\ +1\}$.

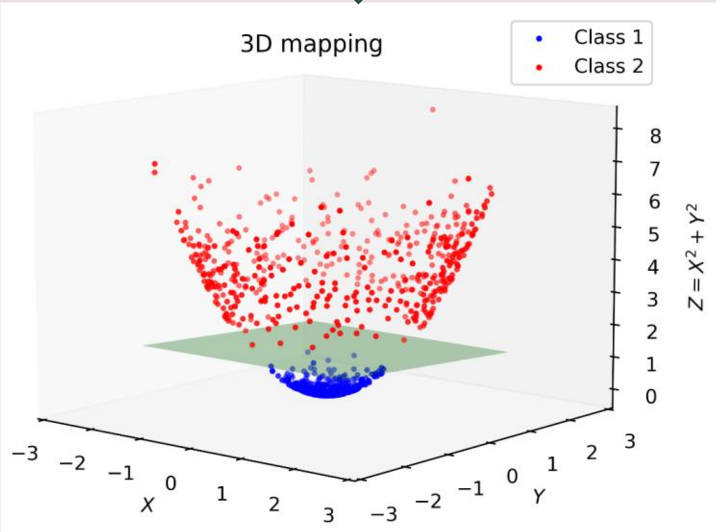
Idea: given a **feature map** $\phi(\vec{x}),\ \phi(\vec{x}_i) \in M\colon \dim(M) = m > n$,finding the best linear decision boundary $\vec{w}^{\boldsymbol{T}}\boldsymbol{\phi}(\vec{\boldsymbol{x}}) - \boldsymbol{b} = \boldsymbol{0}$
by maximizing:

$$f(c_1, c_2, \dots, c_n) = \sum_i c_i - \frac{1}{2}\sum_{ij} y_i c_i y_j c_j \langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle$$

with $\vec{w} = \sum_i \boldsymbol{c_i y_i \phi}(\vec{\boldsymbol{x}}_i)$.

When projecting on the original feature space, the decision boundary will be generally nonlinear.





SVM KERNEL FUNCTIONS

$$f(c_1, c_2, \dots, c_n) = \sum_i c_i - \frac{1}{2} \sum_{ij} y_i c_i y_j c_j \underbrace{\langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle}_{\boldsymbol{k(\vec{x}_i, \vec{x}_j)}}$$

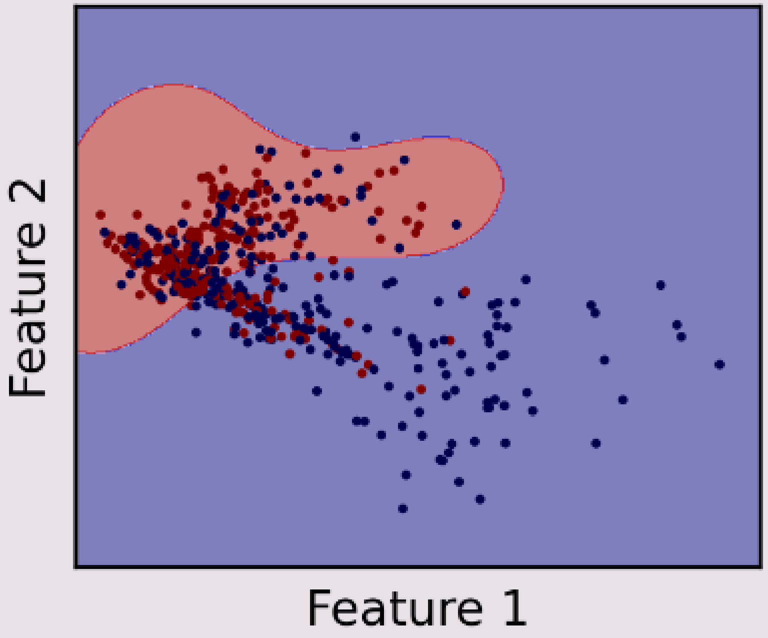
↓

Kernel function

- High-dimensional featuremaps are implicitly defined by a kernel function, which keeps the original feature dimension (hence more efficient to compute).
- The kernel function can be interpreted as a **distance** between samples from the same dataset.

Common kernel choices:

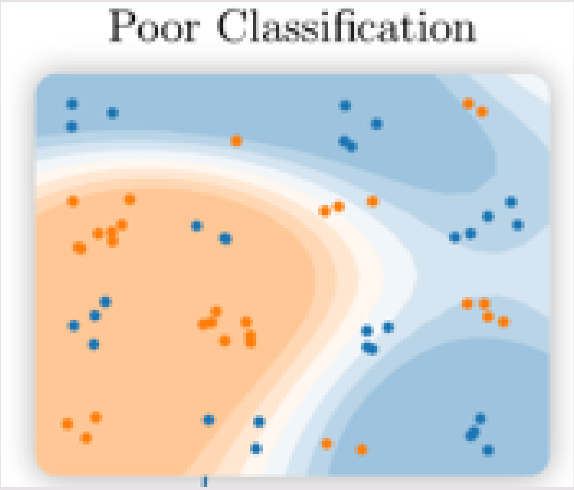
Linear	$K(\vec{x}_i\,,\,\vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$
Polynomial	$K(\vec{x}_i, \vec{x}_j) = (\gamma\, \vec{x}_i \cdot \vec{x}_j + r)^d$
Gaussian	$K(\vec{x}_i, \vec{x}_j) = \exp\left(-\gamma \|\|\vec{x}_i - \vec{x}_j\|\|^2 + C\right)$

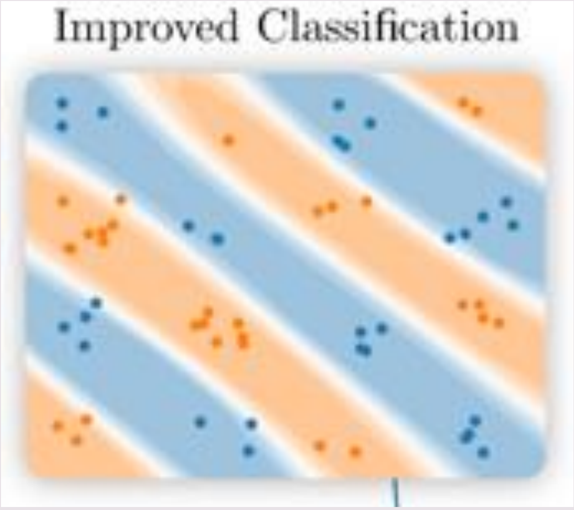


QUANTUM SVM

- The kernel function is a **non-trainable** function of the inputs. Other weights that appears in the loss function can be trained classically.
- Simple loss function landscape (it’s a quadratic form, easy to find global minimum).
- The kernel function determines the overall classification performance, and model’s expressivity.

Quantum Support Vector Machine relies on a Quantum Computer to evaluate the kernel function, i.e. a Quantum Kernel.





Conjecture: some Quantum Kernels are too hard to compute classically, and such kernels may lead better classification performance.

QUANTUM KERNELS

Promoting the classical feature mapping to a quantum state:

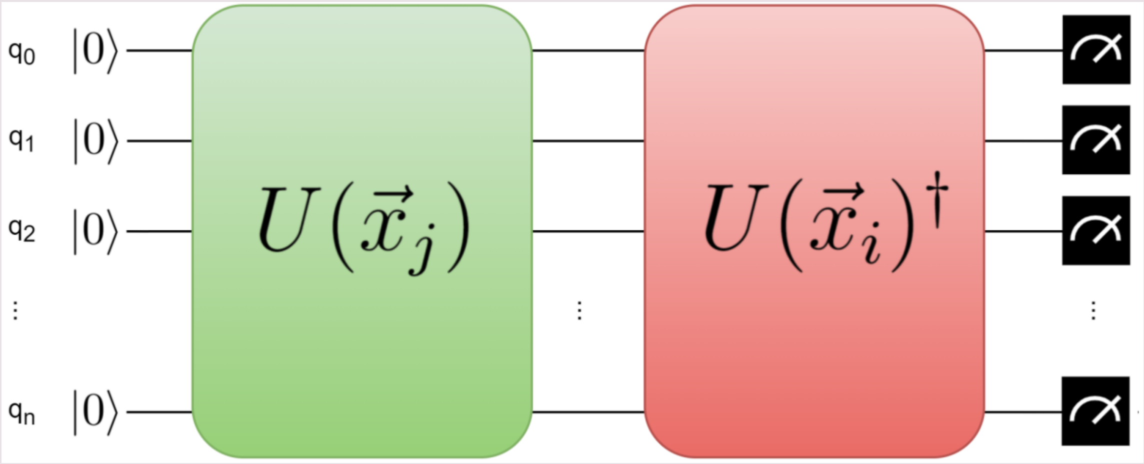
$$\phi(\vec{x}) \rightarrow |\phi(\vec{x})\rangle\langle\phi(\vec{x})| =$$

$$= U(\vec{x})|0\rangle\langle 0|U(\vec{x})^\dagger$$

$$K(\vec{x}_i,\vec{x}_j) = |\langle 0|U(\vec{x}_i)^\dagger U(\vec{x}_j)|0\rangle|^2$$

- Feature maps are still implicitly defined.
- Kernel function is still a measure of similarity between different samples.

Quantum circuits of with this structure are suitable kernels.



Pros:

- Hilbert space grows rapidly with qubit's number
  - Expressive classifiers.
- Quantum kernels are generally hard to compute classically
  - No classical counterpart.
- Good results even with small sized circuits
  - Is a NISQ-era algorithm.

Room for quantum advantage.

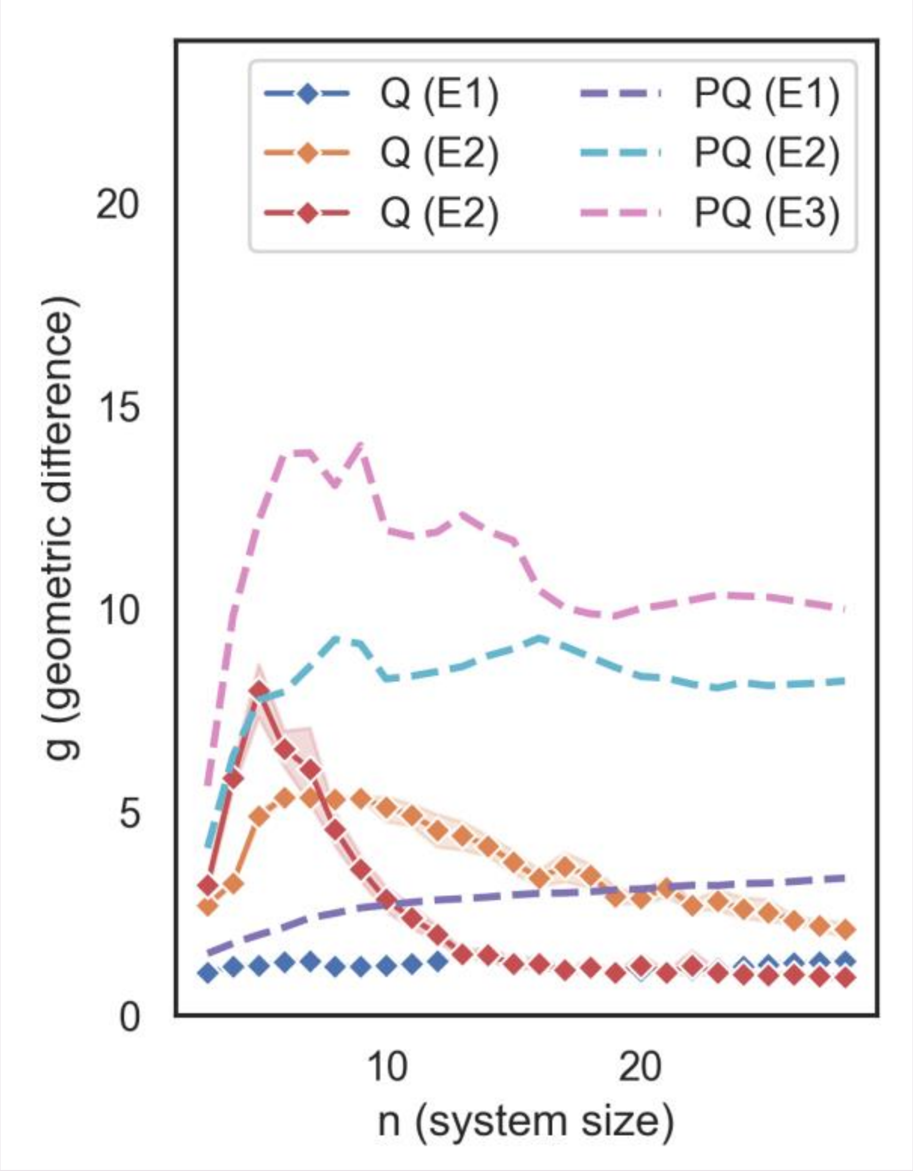
Cons:

- Lack of featuremap explainability
  - Unintuitive relation between circuit and outcome.
- Usually set arbitrarily
  - Problem of chosing a good Quantum Kernel.

QUANTUM KERNEL ISSUES

One risk of quantum kernels, which is observed when the number of qubit increases, is the tendency to map the initial features very «far from each other», because the Hilbert space have a big dimension. This leads to overfitting, i.e. a limited prediction power of the model to data unseen during the training phase, and additionally, due to NISQ hardware noise, kernel values $K(x_i, x_j) \ll 1$ would be very hard to estimate.

One technique that mitigates this effect is the Projected Quantum Kernel, in which part of the quantum information is trown away through partial tracing.



CONCLUSIONS

- Quantum computing is a field in rapid expansion, which is gaining more and more interests.
- Quantum advantage has been demonstrated (on paper) for many-qubits, fault-tolerant computing.
- Nowadays, no quantum advantage has been proven experimentally, principally due to hardware limitations.
- However, modern NISQ-era computing is leading to several intriguing applications, one of the most prominent field being Quantum Machine Learning.
- Parametrized quantum circuits are the key for QML success:
  - Variational: trainable circuits, suitable QNN layers, QAOAs, ...
  - Feature-embedding: mapping data in high-dimensional Hilbert spaces, boosting discrimination power of classical algorithms.