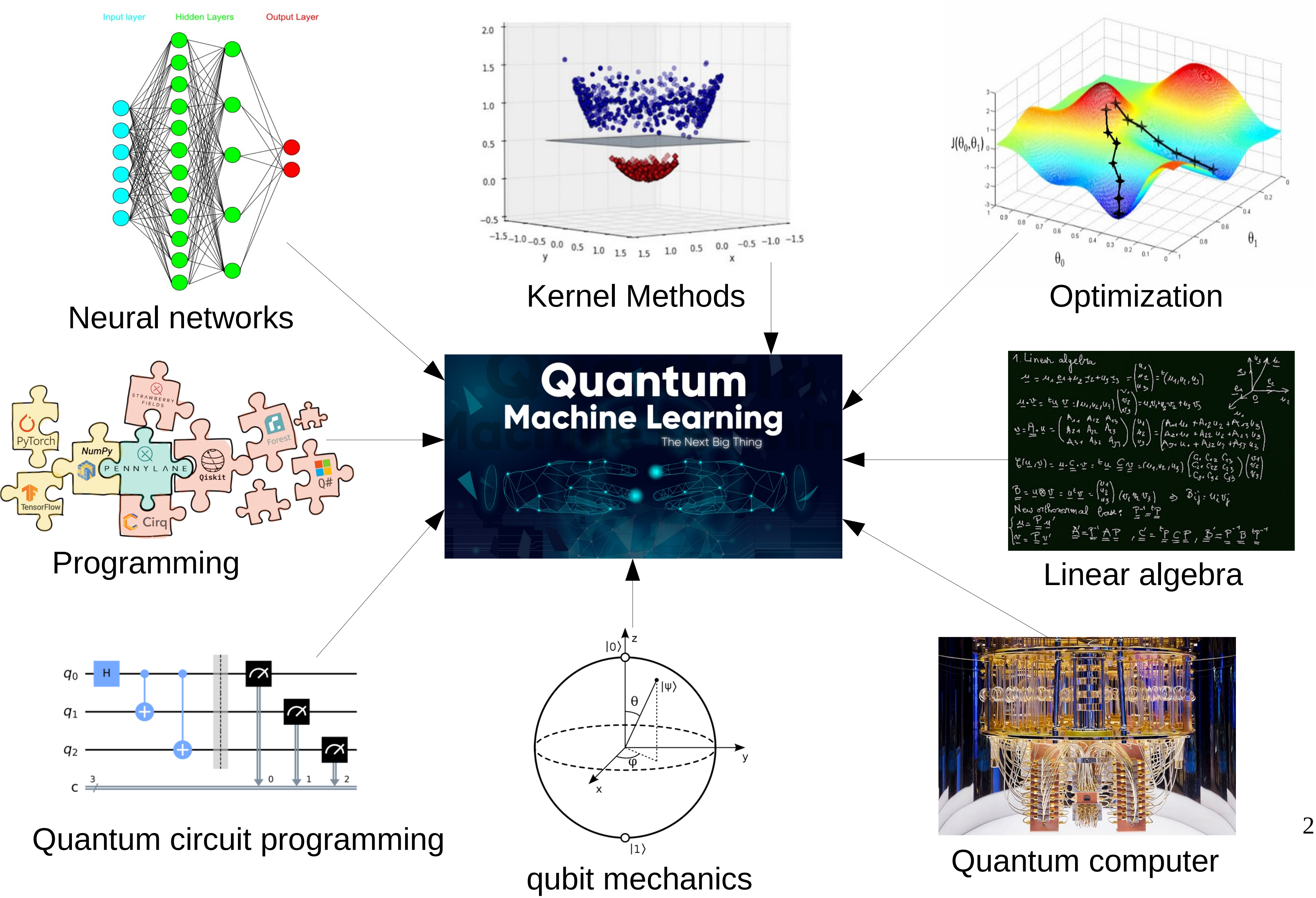
Laboratoire Leprince-Ringuet

Frédéric Magniette

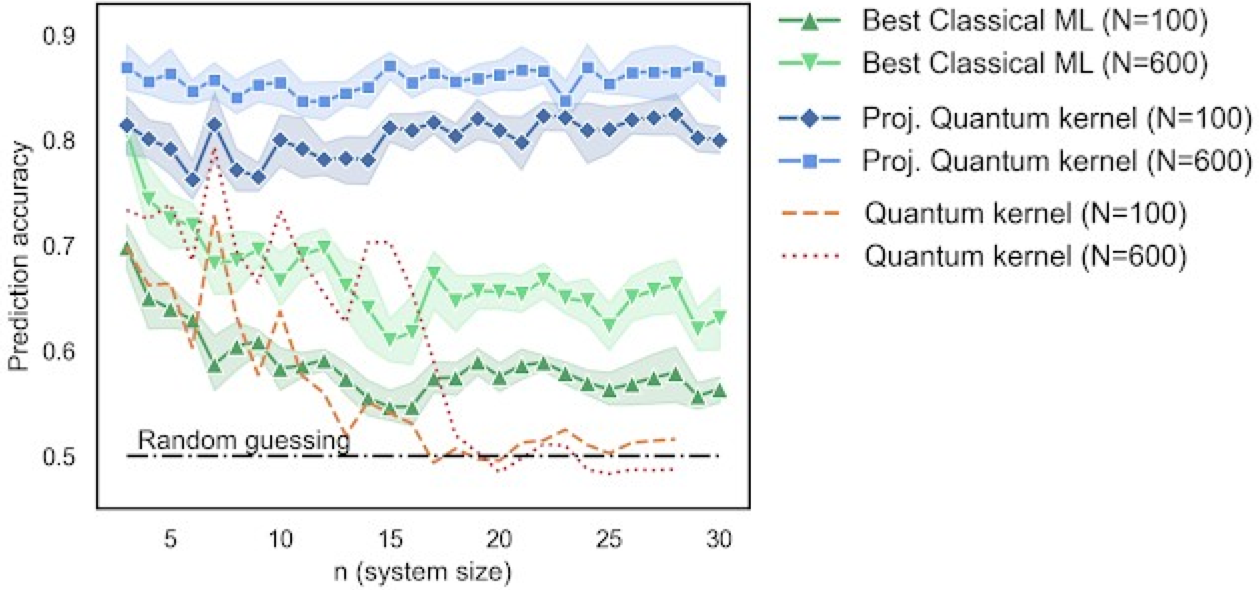
QC2I Project

QML is composite



Objectives of QML

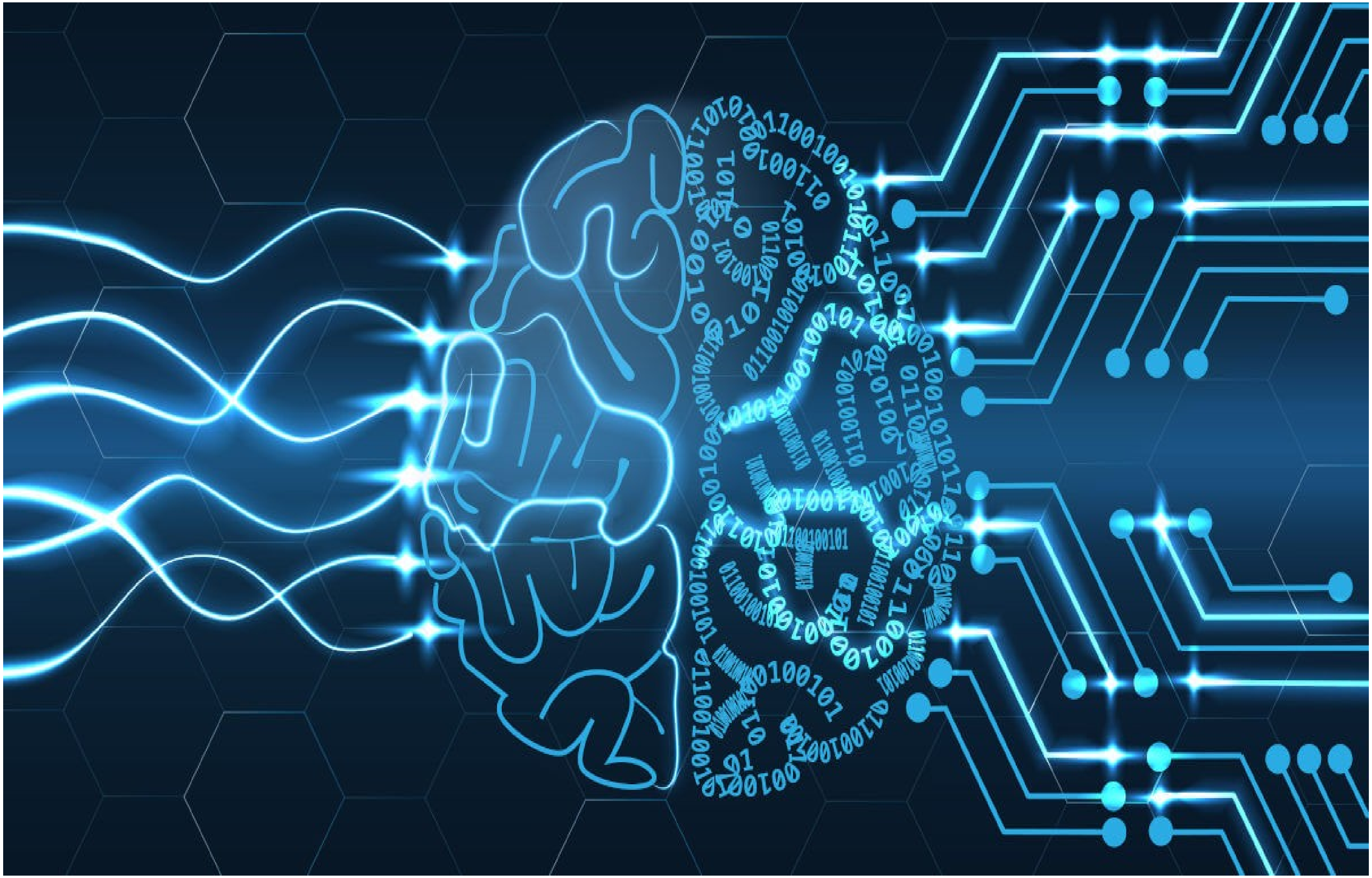
- Applying Machine Learning techniques on quantum computers
  - Try to reduce computational complexity of ML operations
  - Increase the powerfulness of the ML
  - Take advantage of the incredible properties of the quantum space (Hilbert space)
- In HEP Field
  - Event classification
  - Multi-dimensional space exploration



From https://ai.googleblog.com/2021/06/quantum-machine-learning-and-power-of.html

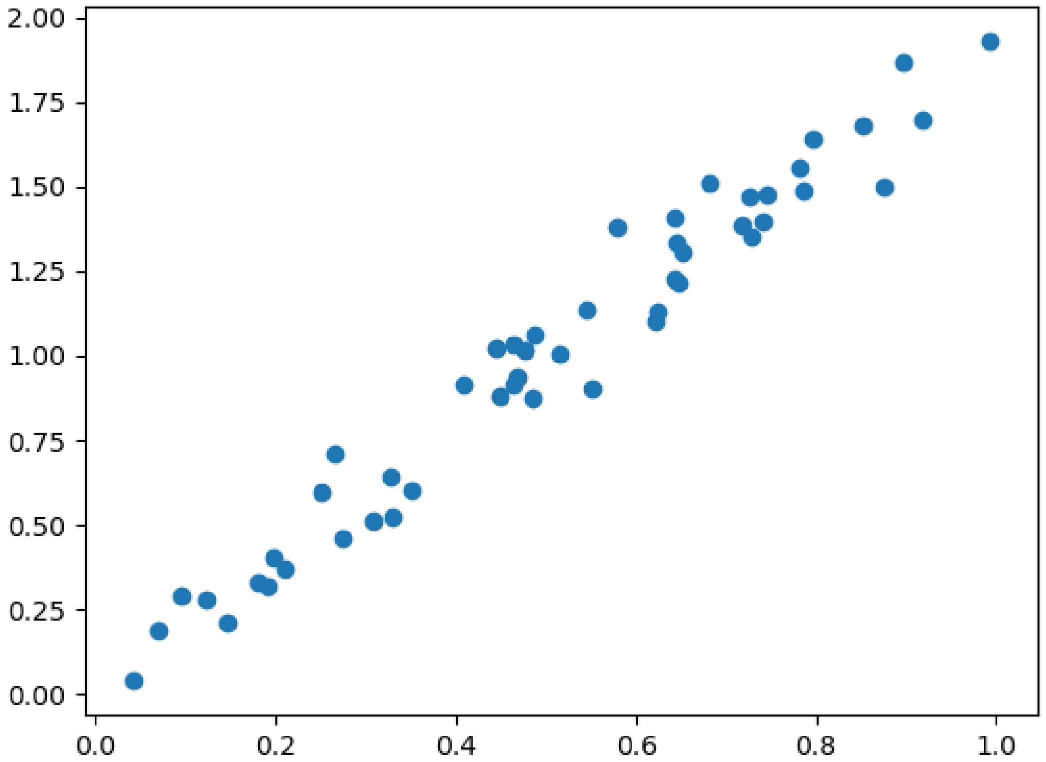
- Machine learning
- Qubit mechanics
- Quantum computers
- Quantum machine learning
- QML in action with Pennylane
- QML @ LLR

Machine Learning



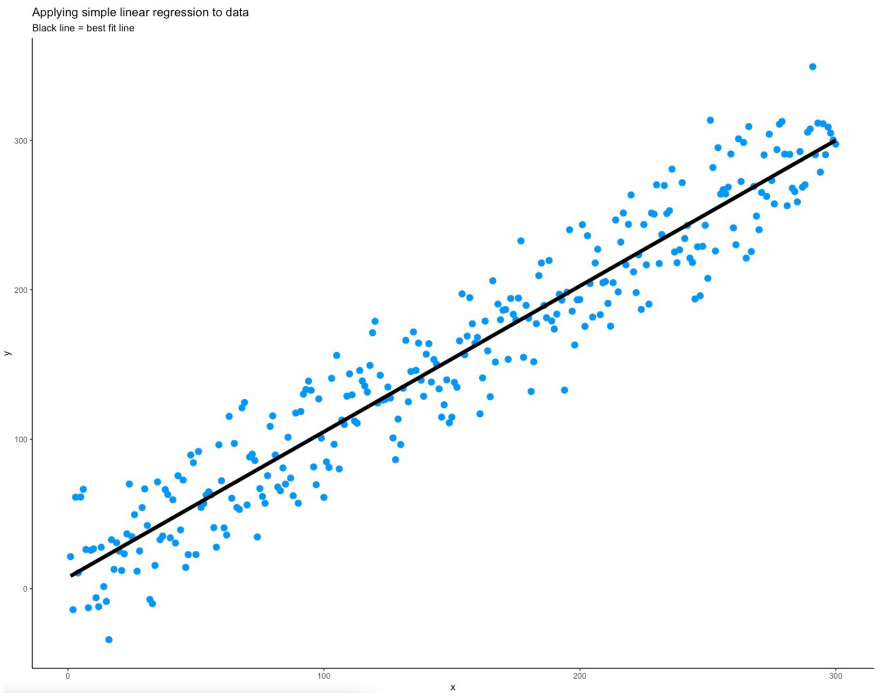
Problematics

x[0]=0.326785 y[0]=0.640017
x[1]=0.484787 y[1]=0.877112
x[2]=0.728836 y[2]=1.350927
x[3]=0.190499 y[3]=0.321072
x[4]=0.717005 y[4]=1.384066
x[5]=0.648116 y[5]=1.216906
x[6]=0.488057 y[6]=1.062203
x[7]=0.917032 y[7]=1.697487
x[8]=0.274938 y[8]=0.460703
x[9]=0.197535 y[9]=0.404545
x[10]=0.122173 y[10]=0.277121
x[11]=0.852632 y[11]=1.682158
x[12]=0.991762 y[12]=1.930109
...



- Supervised learning : n pairs of (x,y) samples are given
- What is the most probable y value for non-given x=0.356 ?

Modelization



- Simple linear regression
- $\hat{y}=f(a,x)=a.x$
- x is the input
- a is the parameter (to determine)
- f is the model → a priori choice
- $\hat{y}$ is the estimated result
- y is the true value (know as ground truth)
- $L(y,\hat{y})$ is the loss function. For example $| y - \hat{y} |$
- How to evaluate a ?

Estimator

- For simple models, we can compute estimators of the parameters

$$\hat{a}(x_i, y_i) = \frac{\sum_{i=0}^{n} y_i/x_i}{n}$$

- Example : estimator for a

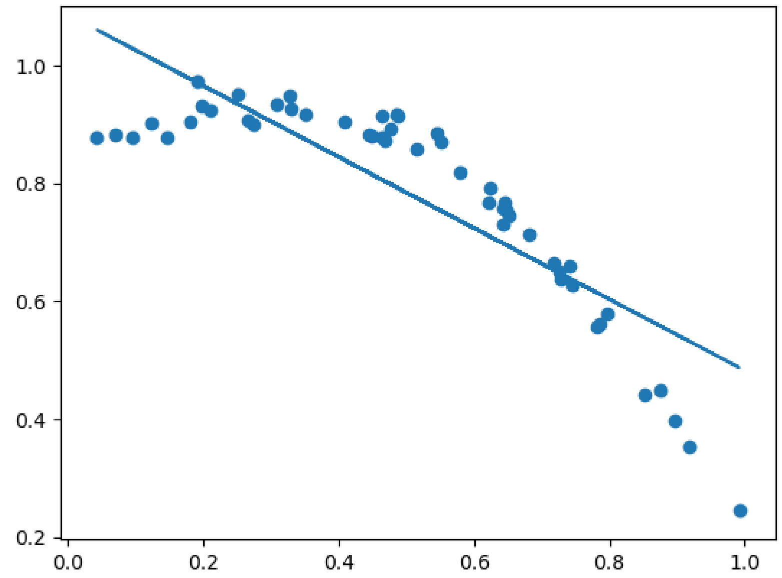
```
>>>np.mean(y/x)
1.987
>>>0.356*1.987
0.707
```

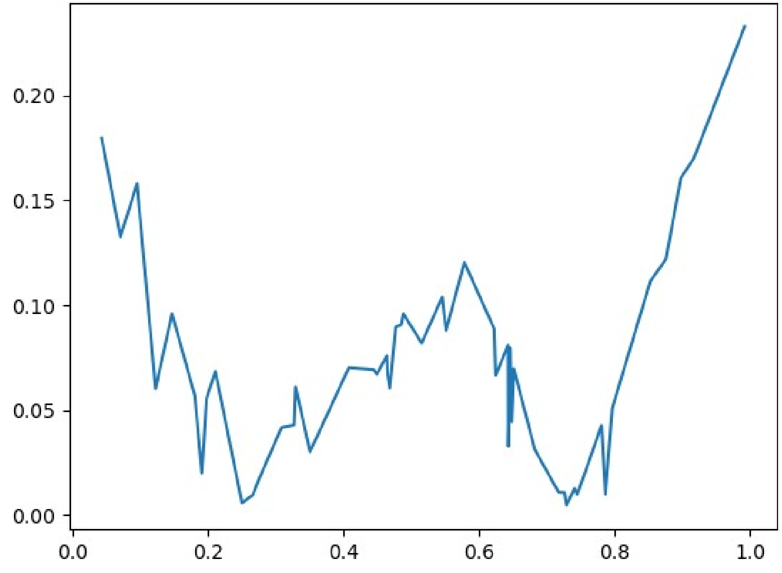
- We expect the estimator to be asymptotically unbiased

$$\lim_{n \to +\infty} \hat{a}(x_1, \ldots, x_n, y_1, \ldots, y_n) = a$$

Bias and model choice

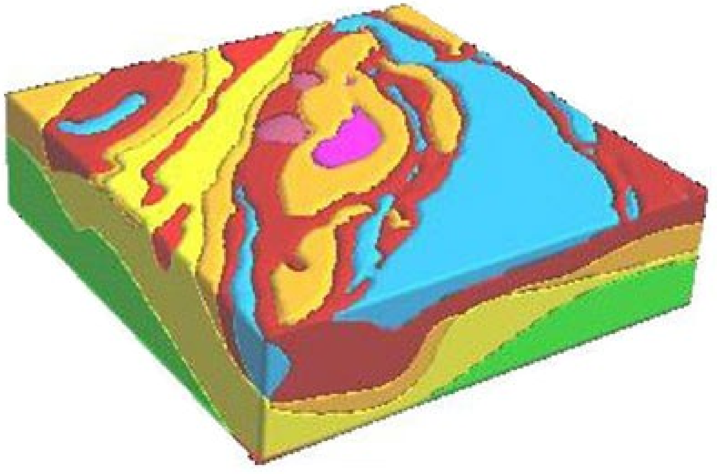
- If the model is not adapted, the error (bias) becomes important
- Model : y=ax+b
- Model is not expressive enough
- Example error=|y-ŷ|





General form of models

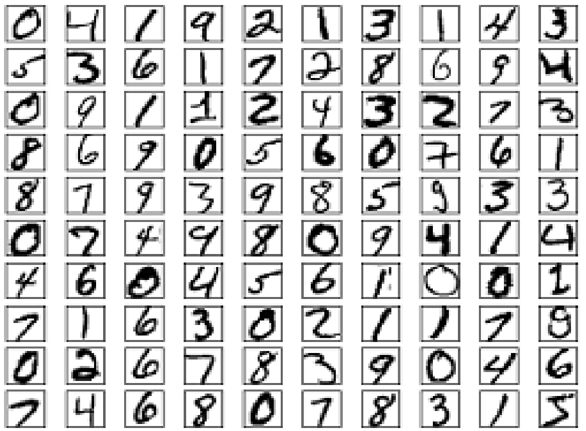
- x is extended to vector X
- a is extended to vector Θ
- y is extended to vector Y
- $\hat{Y}=f(X,\Theta)$
- Loss function $L(\hat{Y},Y)$
- No estimators for finding Θ



How to choose a good model ?

- What mathematical form ?
- How many parameters ?
- Is there universal models ?
- Is it possible to have good estimators for all my parameters ?

Plenty of models



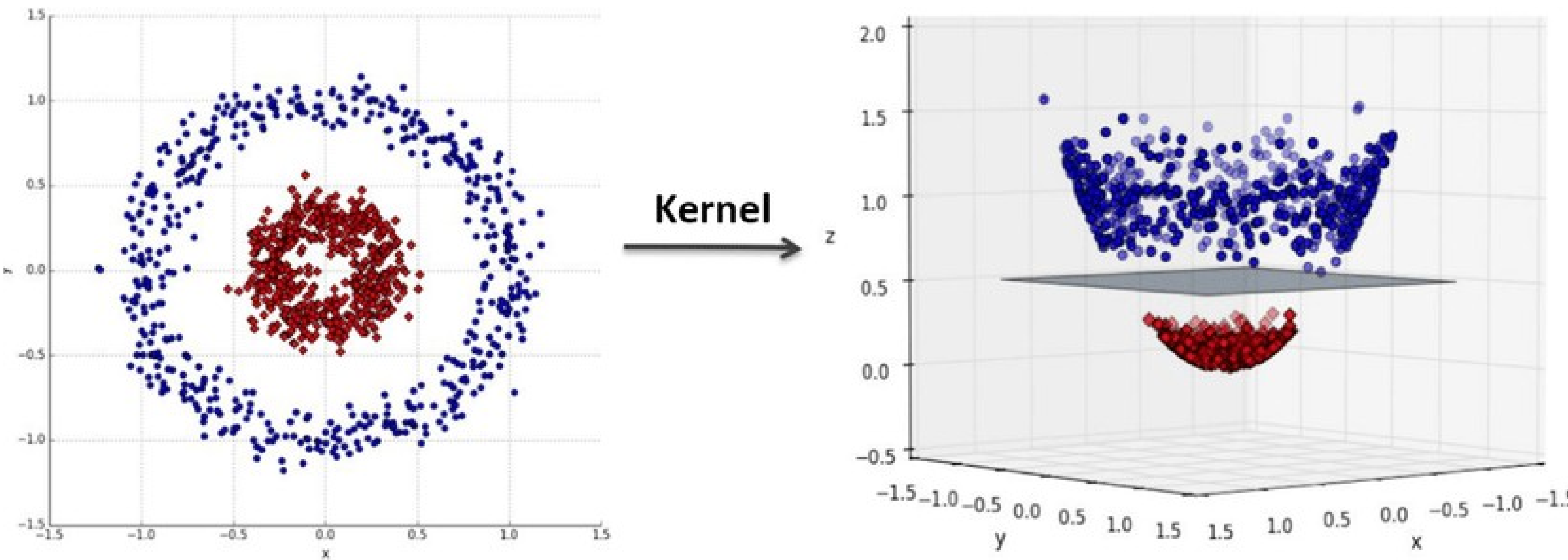
MNIST is a handwritten digit dataset used as a benchmark since 90's. The problem (to associate the picture with a digit) is difficult .

Method	Error Rate	Authors	Year
Linear Classifier	12	Lecun et al.	1998
2-layer NN, 300 hidden units	4.7	LeCun et al.	1998
KNN	0.63	Belongie et al.	2002
Virtual SVM, deg-9 poly, 2-pixel jittered	0.56	DeCoste and Scholkopf	2002
6-layer NN 784-2500-2000-1500-1000-500-10 [elastic distortions]	0.35	Ciresan et al.	2010
Convolutional NN, 1-20-P-40-P-150-10 [elastic distortions]	0.23	Ciresan et al.	2012
Human brain	0.2	Ciresan et al.	
Random multi-model Deep Learning	0.18	Kowsari et al.	2018
Branching/Merging CNN Homogeneous Vector Capsule	0.13	Byerly et al.	2020

From http://yann.lecun.com/exdb/mnist/

Kernel Methods

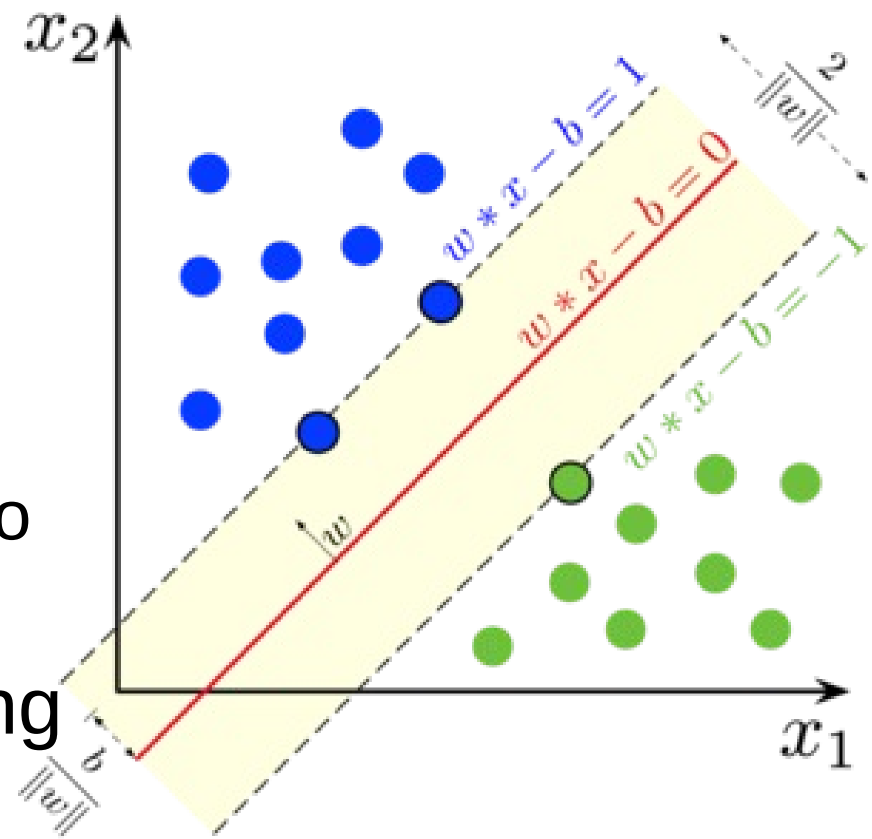
- Adapted to non linearly separable data
- Adding features to the original data, i.e. creating a new representation of the data in a space with more dimensions
- Applying a linear classifier (i.e. separating hyperplane)
- Unclear universality
- Example $\varphi(a,b)=(a,b,a^2+b^2)$:



from http://borisburkov.net/2021-08-03-1/

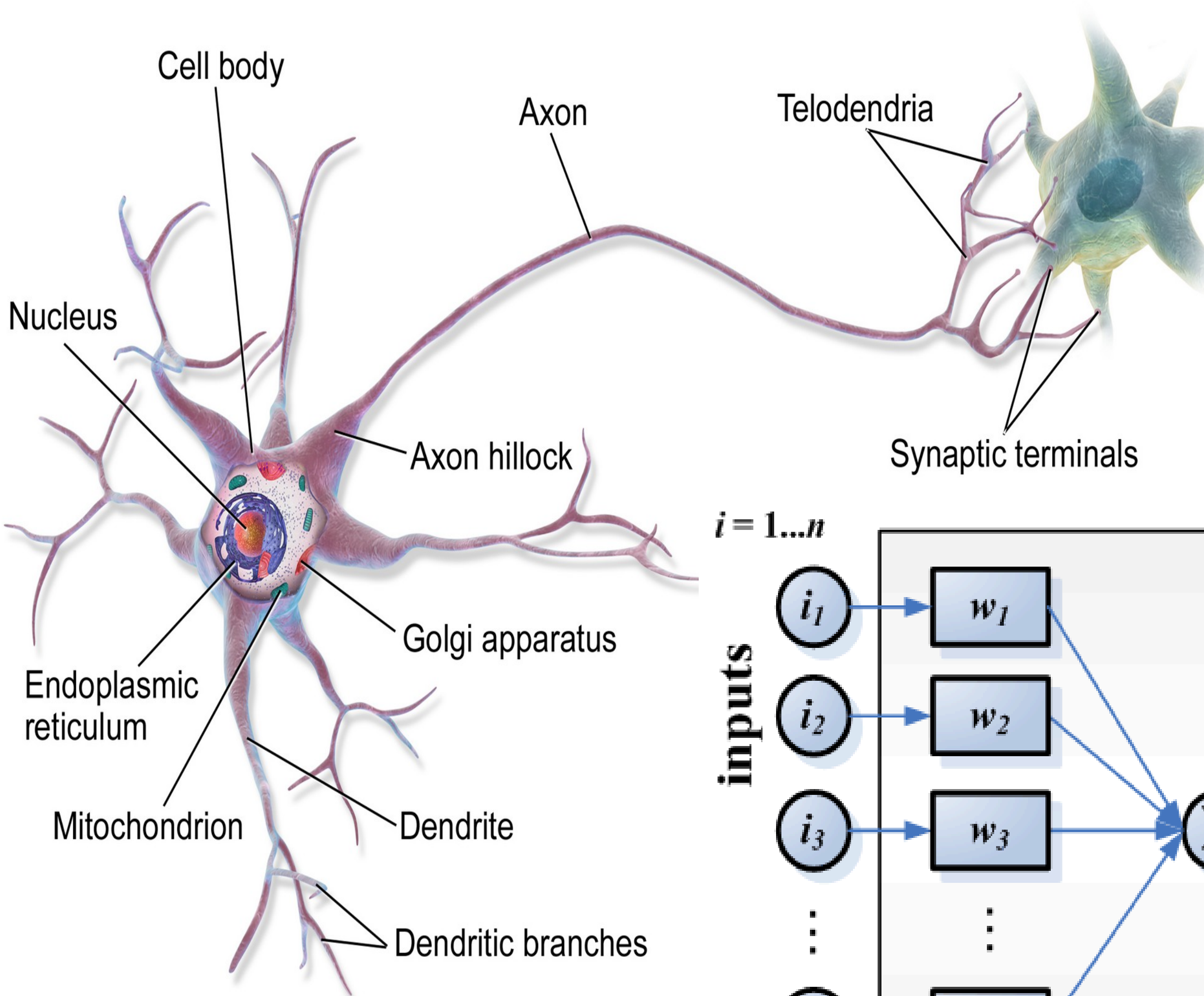
Support Vector Machine (SVM)

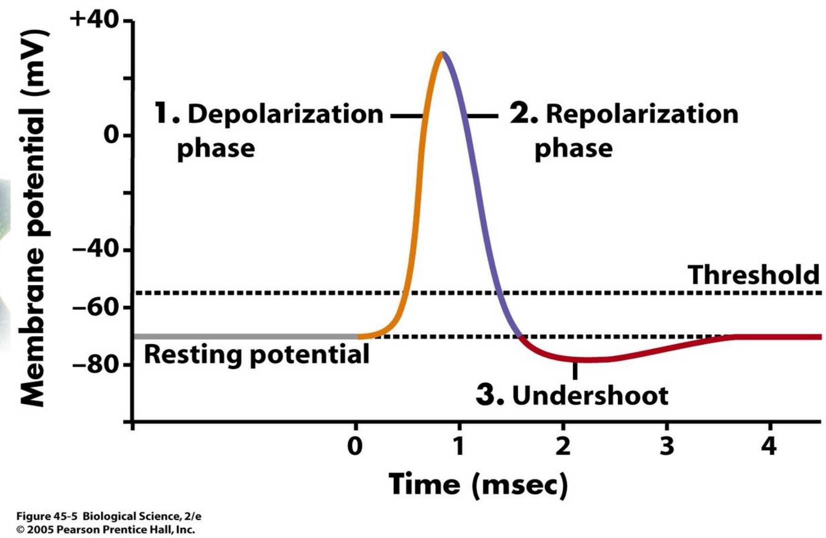
- Enrich data with kernel features
  - Polynomial kernel $K(x,y)=(x^T.y+1)^d$
  - Gaussian Kernel
    $K(x,y)=\exp(-\|x-y\|^2/2\sigma^2)$
- Find the support vectors
  - the points of each class closer to the other class
- Find the hyperplane maximizing the distance to the support vectors (hard-margin)

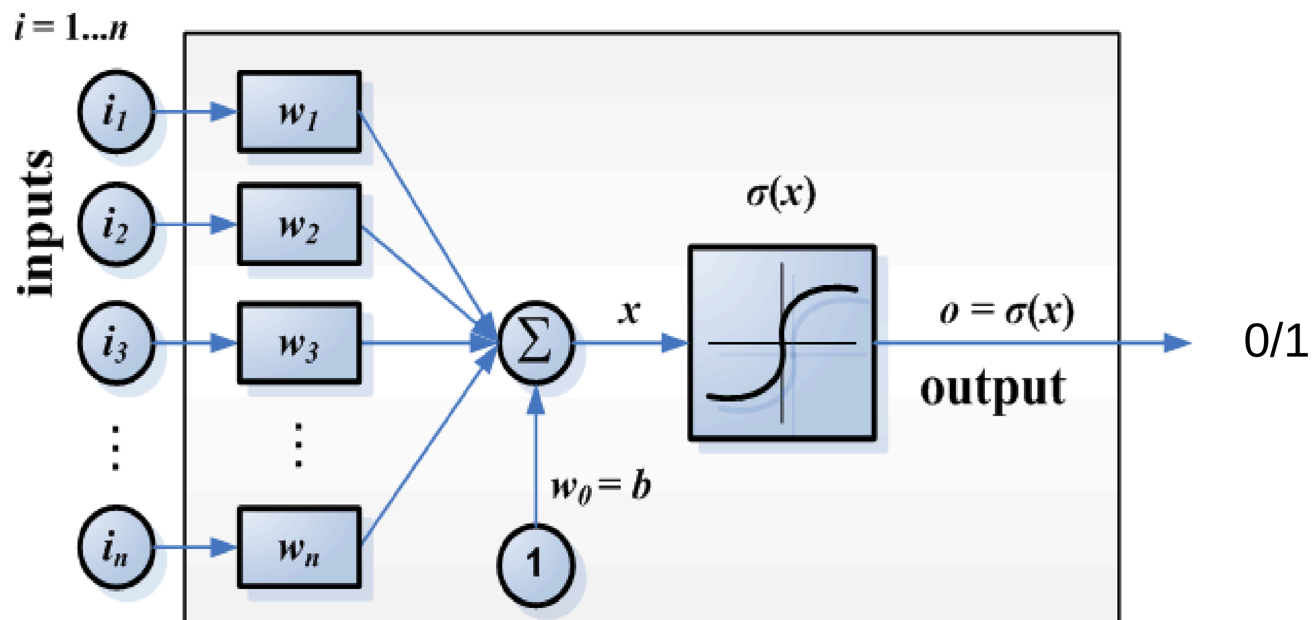


From https://en.wikipedia.org/wiki/Support-vector_machine

From real to formal neuron



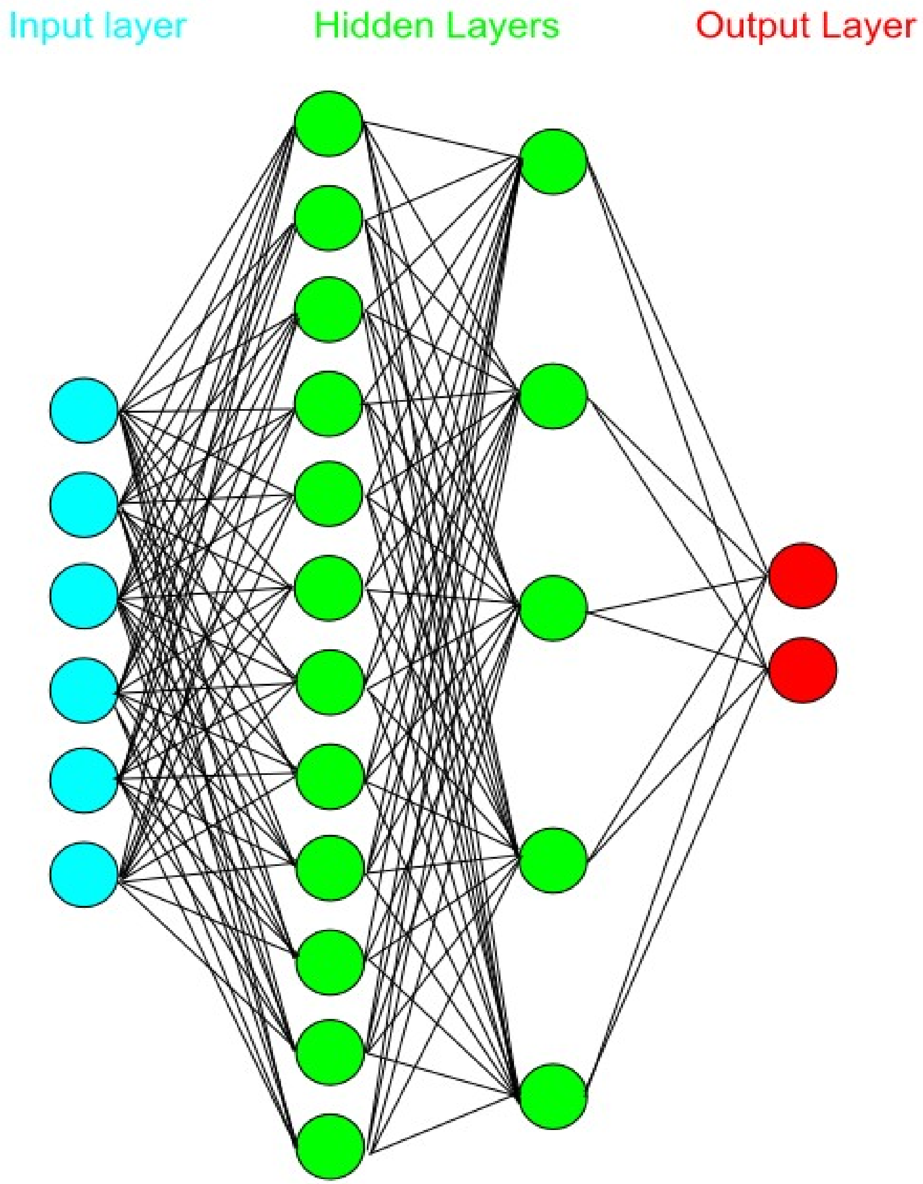




From https://en.wikipedia.org/wiki/Neuron

Warren McCulloch & Walter Pitts 1943

Multi-layer Perceptron



- Neural neurons
- Organized by layers
- Different size of layers
- Full connectivity between layers
- Input layer
- Output layer
- Hidden layers

Multi-layer perceptron formalism

- mono-layer (matricial form) $\hat{Y} = \sigma(W^T X + B)$
- multi-layer

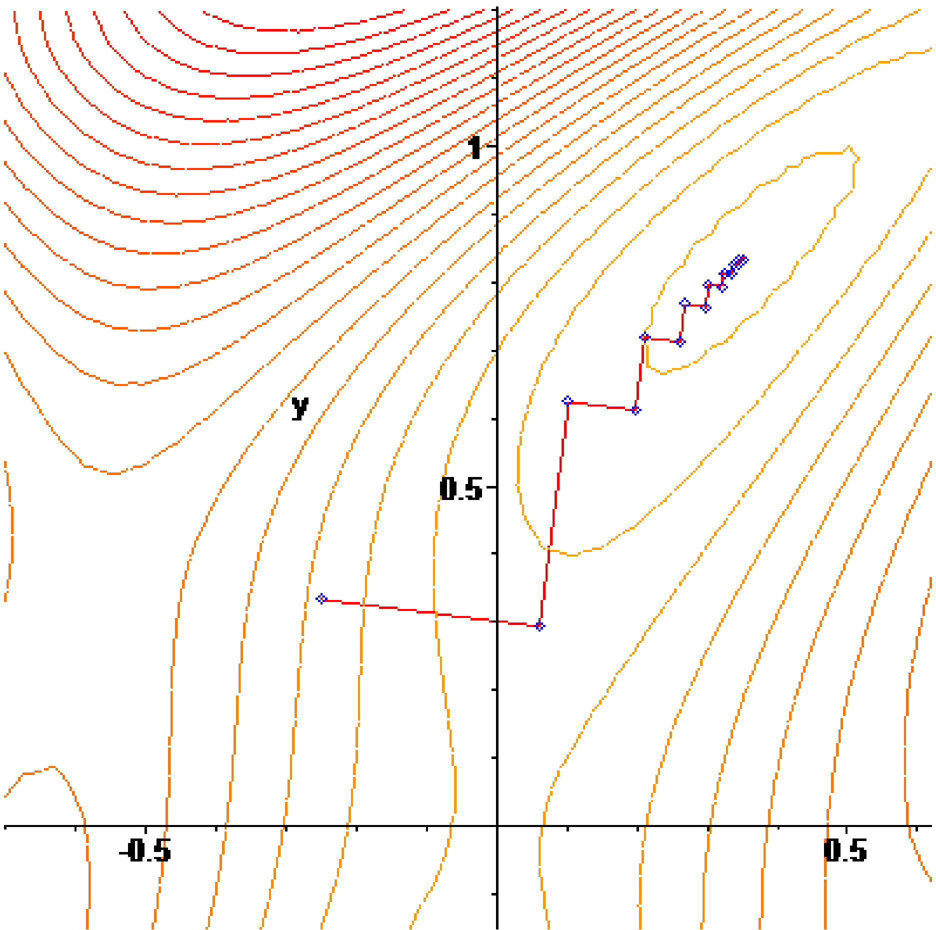
$$\hat{Y} = \sigma(W_1^T \sigma(W_2^T \dots \sigma(W_n^T X + B_n) + \cdots + B_2) + B_1)$$

- Depth of deep learning is the number of σ applications
- W_i and B_i are the parameters (Weights and Biases)
- X are the inputs
- No general form → universal approximator
- No estimator for parameters
- How to evaluate the optimal parameters ?

Gradient Descent

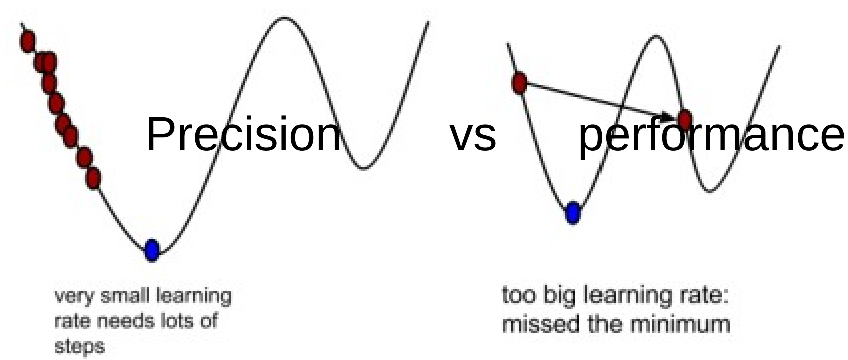
- « Following the slope method »
- Calculating the gradient vector with respect to $\Theta = < \theta_1, \ldots, \theta_n >$

$$\nabla L(X,\Theta) = < \frac{\partial L(X,\Theta)}{\partial \theta_1}, \frac{\partial L(X,\Theta)}{\partial \theta_2}, \ldots, \frac{\partial L(X,\Theta)}{\partial \theta_n} >$$

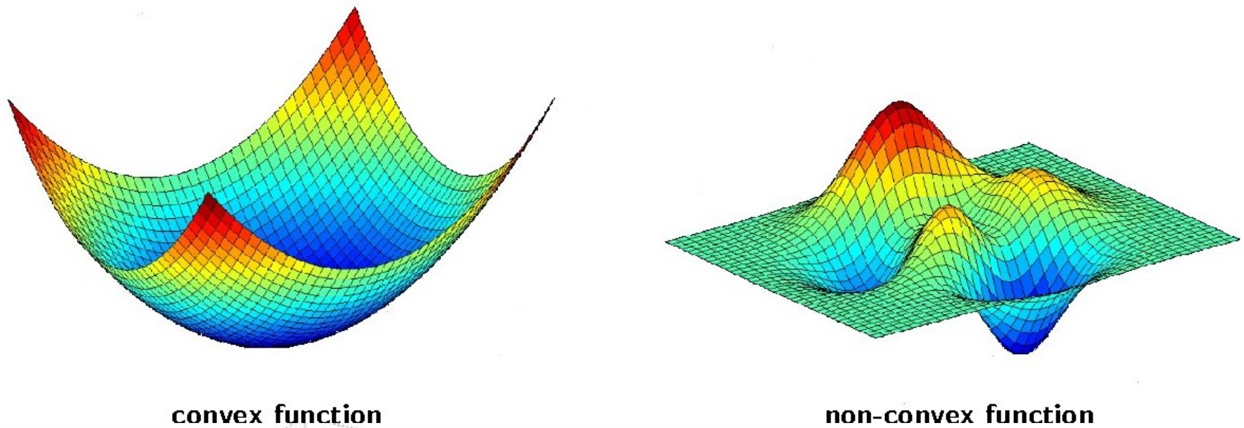


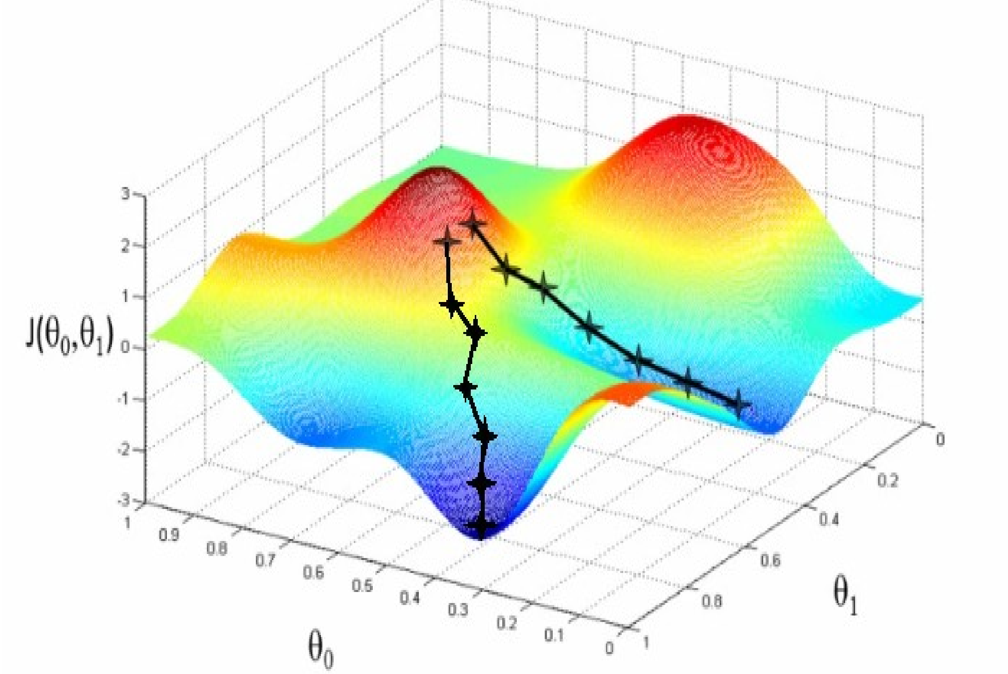
$$\Theta = \Theta - \alpha \nabla L(X,\Theta)$$

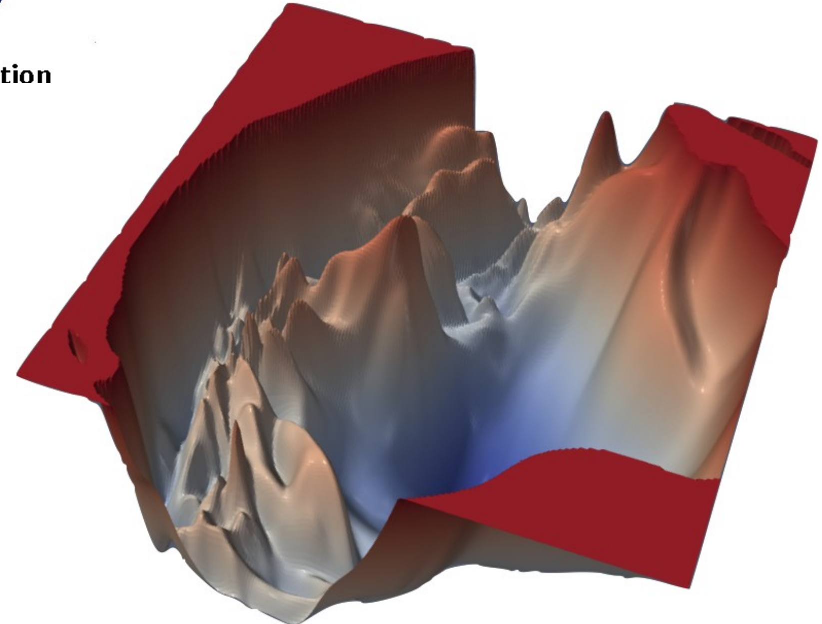
α : step size



Gradient Descent & Convexity





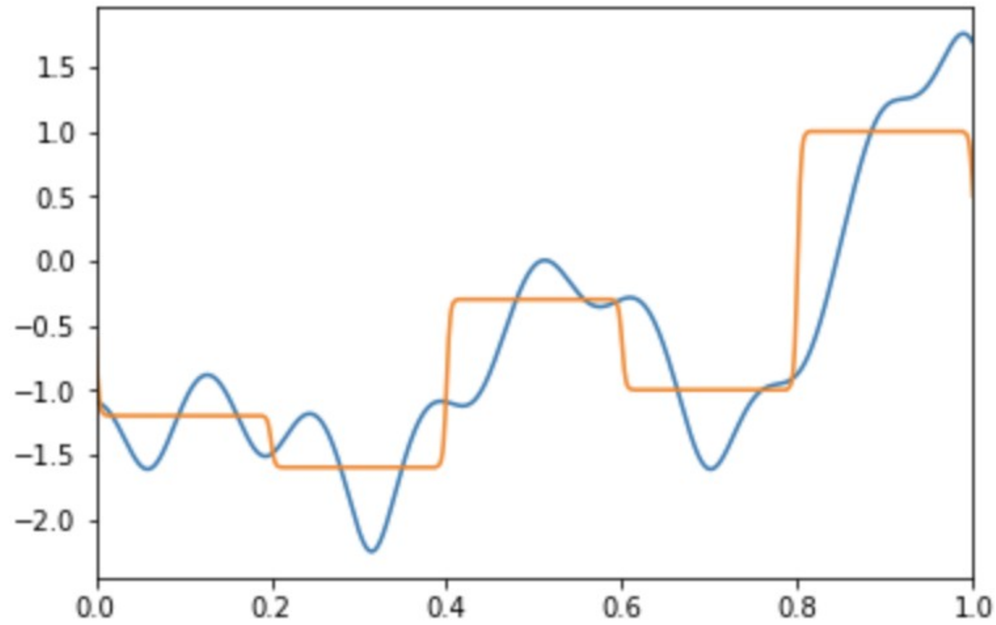


Li & al, « Visualizing the loss landscape of neural nets, 2018, 1712.09913

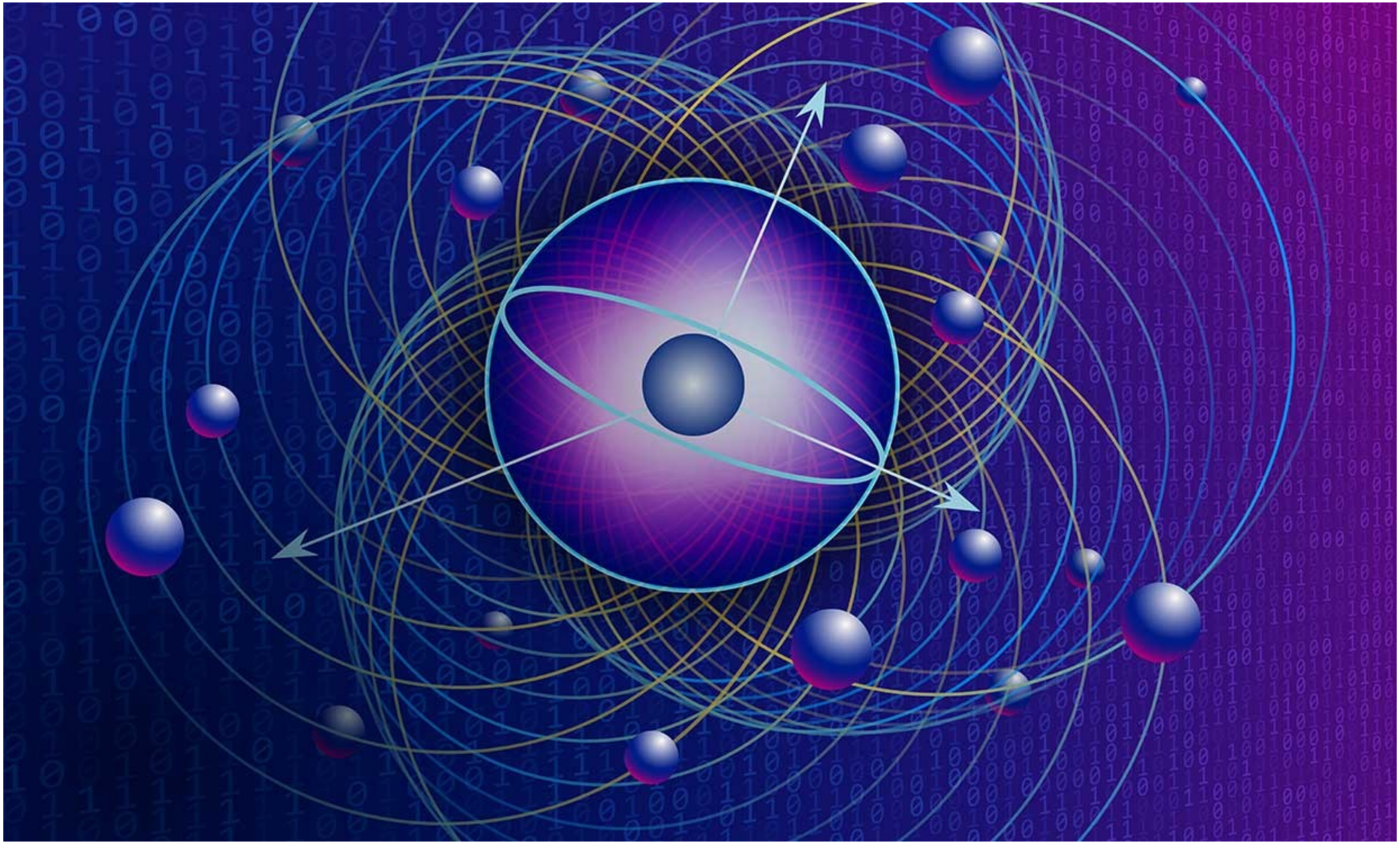
- Result depends on the starting point
  → require convexity (unique minima)
- Practical solution : multiple random starts

What kind of function can I train ?

- Any continuous multivariate function on R (Hornik et ak 1989) → universal approximator
- Extended to R^n (Sun and Cheney 1992)
- Extension to classification problems (Cybenko 1989) → universal classifiers
- Caution : the theorem gives no clue about learnability

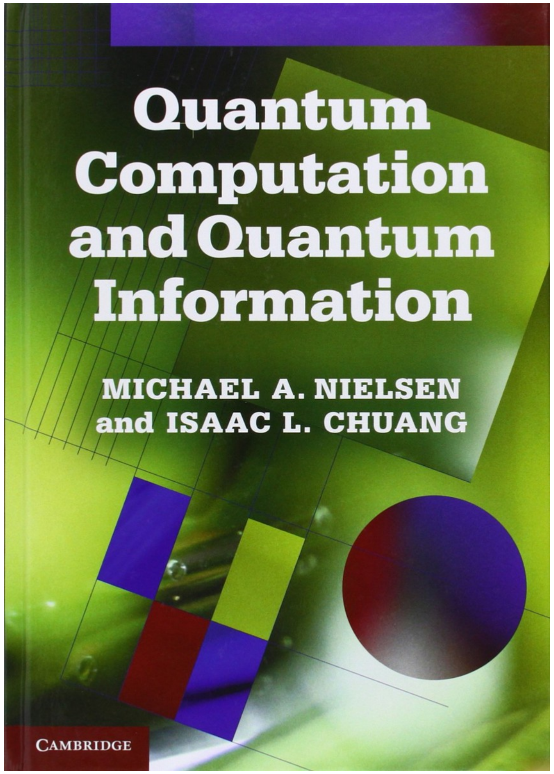


Qubit Mechanics

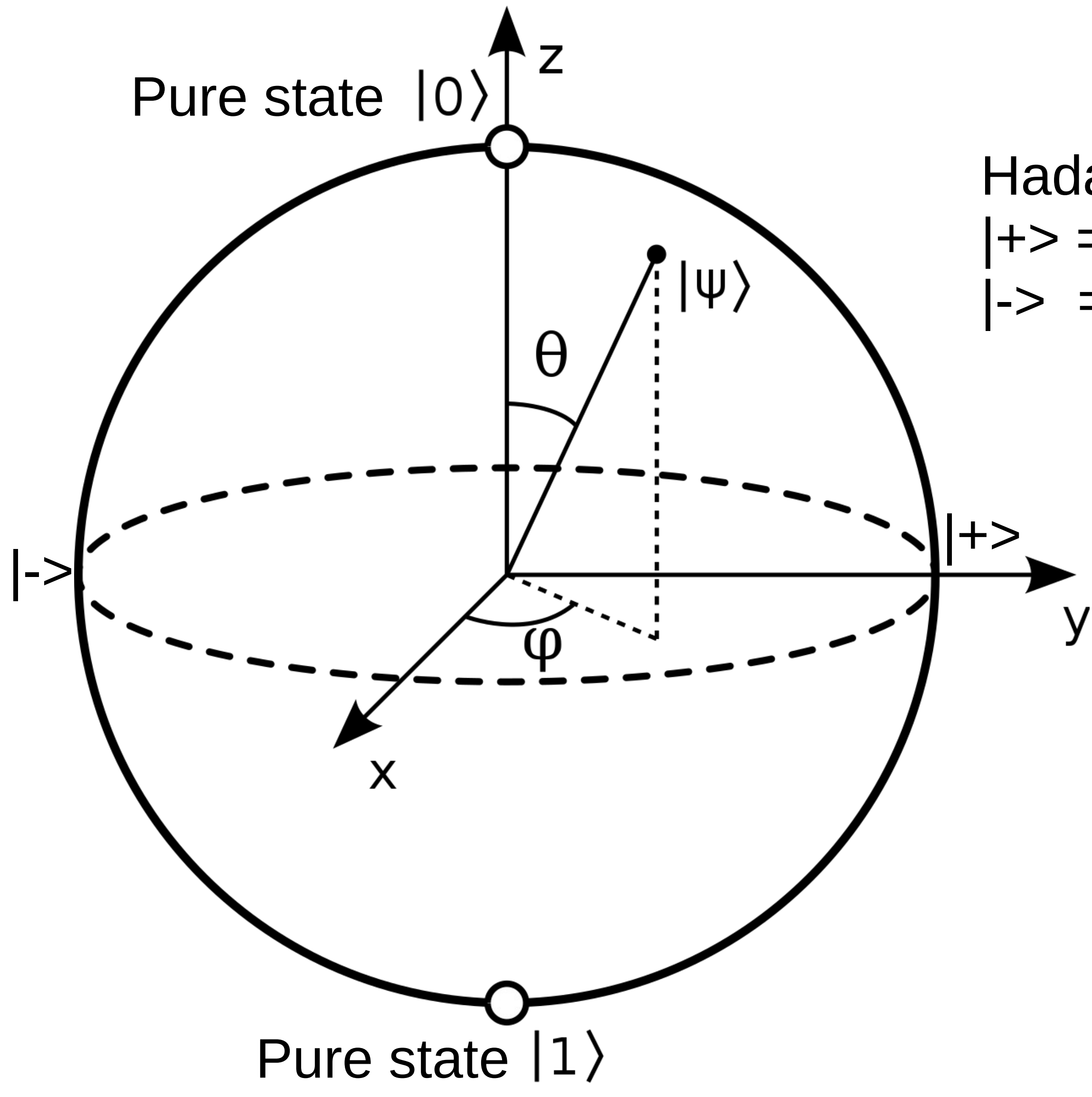


Quantum computing

- Using quantum object to perform computations
- Base object : quantum bit or qubit
- Qubit
  - Two pure states q=|0> or q=|1>
  - Superposition principle q=a|0> + b|1> with a and b complex numbers is also a valid qubit
  - Normalization $|a|^2+|b|^2=1$

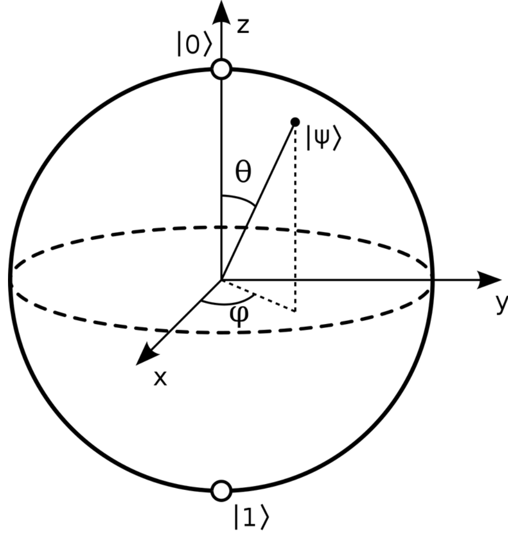


Bloch Sphere representation



Hadamard states
|+> = 1/√2 (|0> + |1>)
|-> = 1/√2 (|0> - |1>)

Qubit Measurement

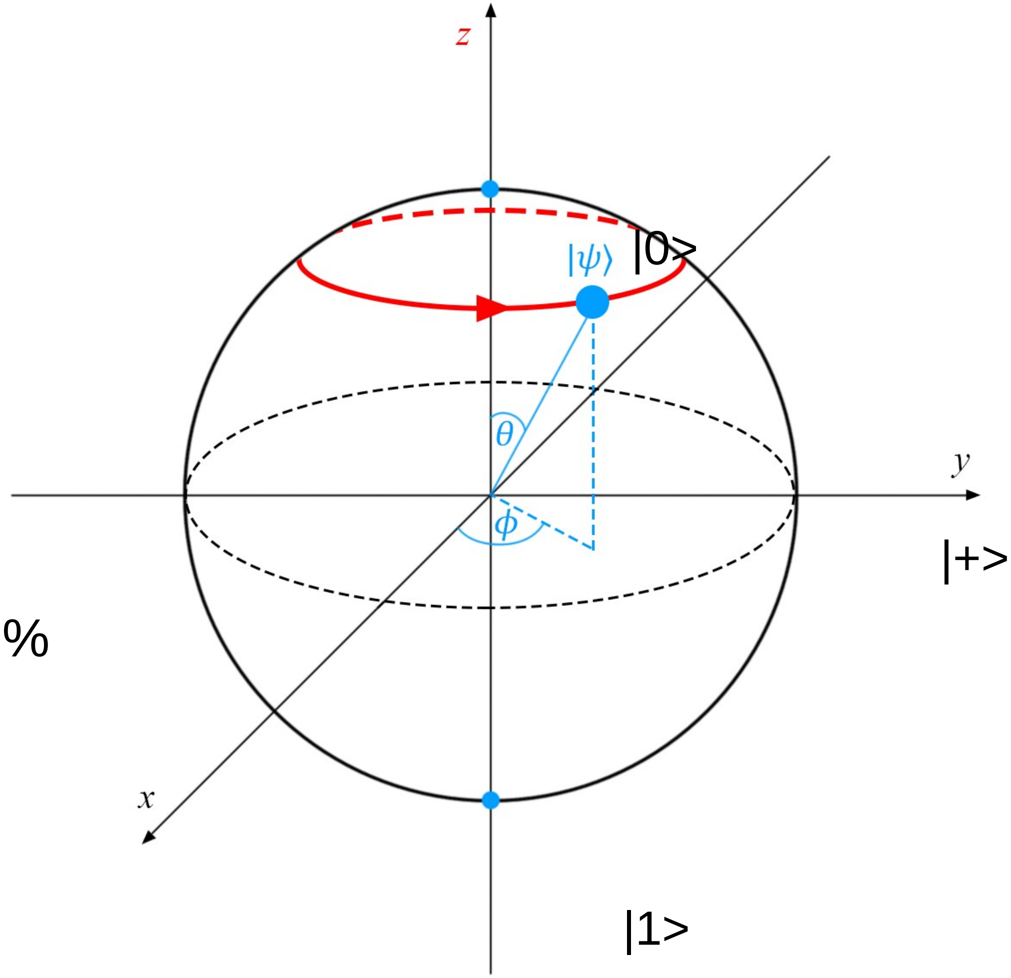


- The internal state (a,b) of a qubit a|0>+b|1> cannot be measured
- When measured we obtain randomly 0 or 1 only
- Born rule: we obtain 0 with $|a|^2$ probability

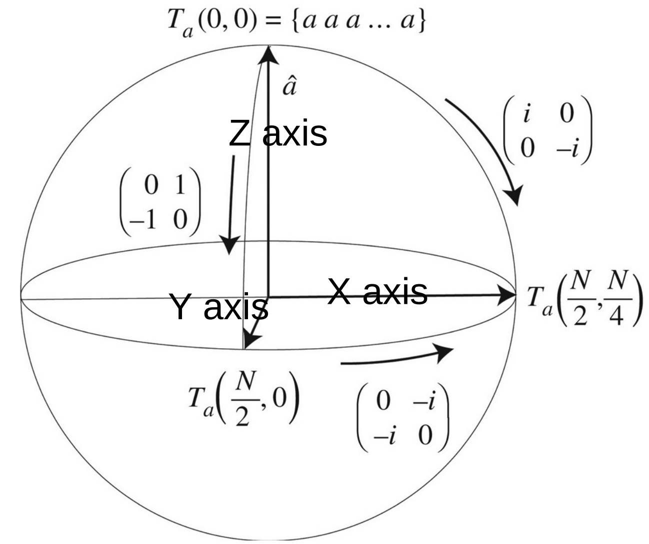
  and 1 with $|b|^2$ probability
- The measurement is a projection on the z axis
- Destructive operation: the qubit value is fixed to the measured value (wave function collapse)
- Effective measurement procedure
  - perform 1000 setups and measures
  - calculate the empirical probability (approx. $|a|^2$)

Measurement examples

- Pure state |0>
  - 0 with 100 %
  - 1 with 0 %
- Pure state |1>
  - 0 with 0 %
  - 1 with 100 %
- Hadamard state |+>
  - 0 with 50 % $(1/\sqrt{2})^2 = 1/2 = 50\%$
  - 1 with 50 %
- Any other state
  - 0 with $|a|^2$
  - 1 with $|b|^2$
  - without respect to phase (all points on the red circle give the same measures)



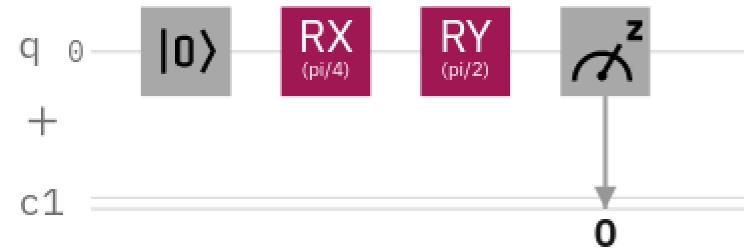
One qubit operator



- A qubit can evolve but need to preserve its normalization (to stay on the sphere)
- Operators are unitary 2x2 complex matrices

$$\begin{bmatrix} m_1^* & m_3^* \\ m_2^* & m_4^* \end{bmatrix} \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- Can be decomposed in a rotation basis with one parameter Φ
- Four different representations




```
qreg_q = QuantumRegister(1, 'q')
creg_c = ClassicalRegister(1, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.reset(qreg_q[0])
circuit.rx(pi/4, qreg_q[0])
circuit.ry(pi/2, qreg_q[0])
circuit.measure(qreg_q[0], creg_c[0])
```

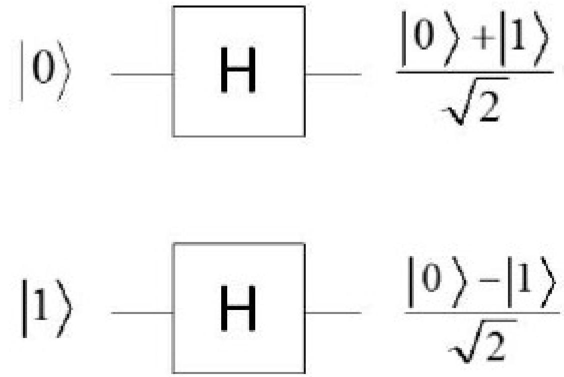
$$Rx(\phi) = \begin{bmatrix} cos\phi & -isin\phi \\ -isin\phi & cos\phi \end{bmatrix}$$

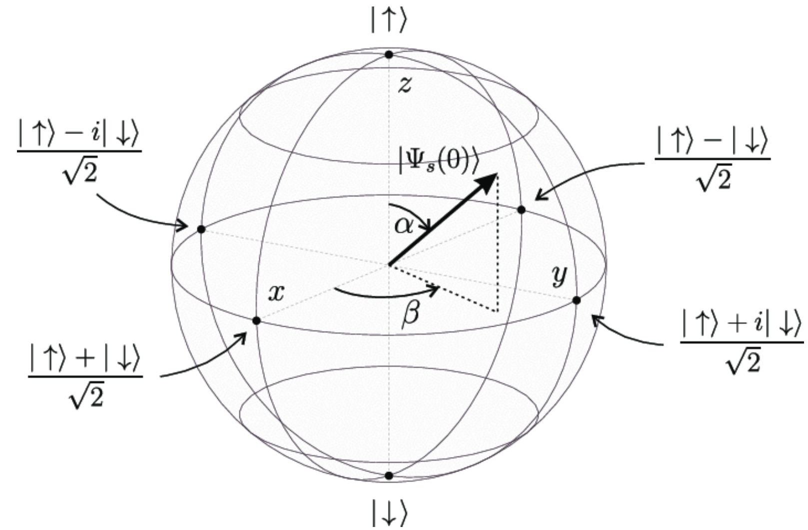
$$Ry(\phi) = \begin{bmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{bmatrix}$$

$$Rz(\phi) = \begin{bmatrix} e^{-i\phi} & 0 \\ 0 & e^{i\phi} \end{bmatrix} = \begin{bmatrix} cos\phi - isin\phi & 0 \\ 0 & cos\phi + isin\phi \end{bmatrix}$$

The Hadamard operator

- Create an equiprobable mix of |0> and |1> from a pure state
- H(|0>)=1/√2 (|0>+|1>)=|+>
- H(|1>)=1/√2 (|0>-|1>)=|->
- Which are both measured |0> or |1> with probability 1/2
- Matrix form $H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$






```
circuit.h(qreg_q[0])
```

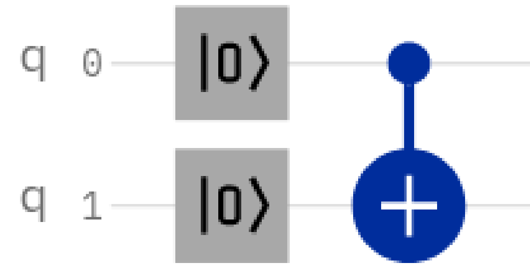
Composed states

- Composed of multiple states
- Tensor product or Kronecker product for vectors
- No Bloch representation
- In state notation
  - a|00>+b|01>+c|10> + d|11>

$$\begin{bmatrix} a \\ b \end{bmatrix} \otimes \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} ac & ad \\ bc & bd \end{bmatrix}$$

- Mesurement
  - obtain 0,0 with probability $|a|^2$ ...
- Some states (entangled) cannot be obtained by Kronecker product
- Operators can be also obtained also by Kronecker product but not all of them (entanglement)

The CNOT operator



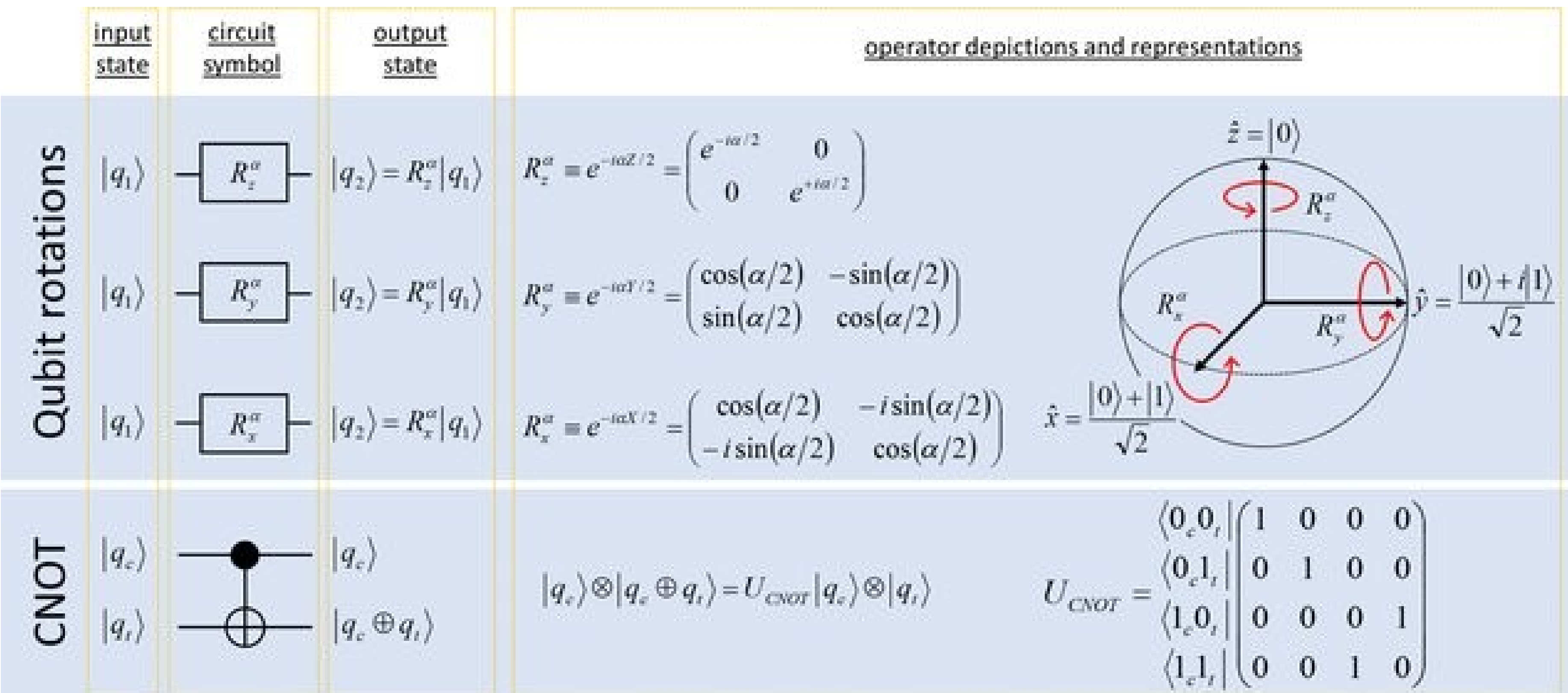
- Stands for « controlled not »
- Apply on two qubits, the control and the target
- If control is |1>, flip the target, else do nothing
- Effect on two qubit states
  - |00> → |00>
  - |01> → |01>
  - |10> → |11>
  - |11> → |10>

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

```
circuit.cx(qreg_q[0],qreg_q[1])
```

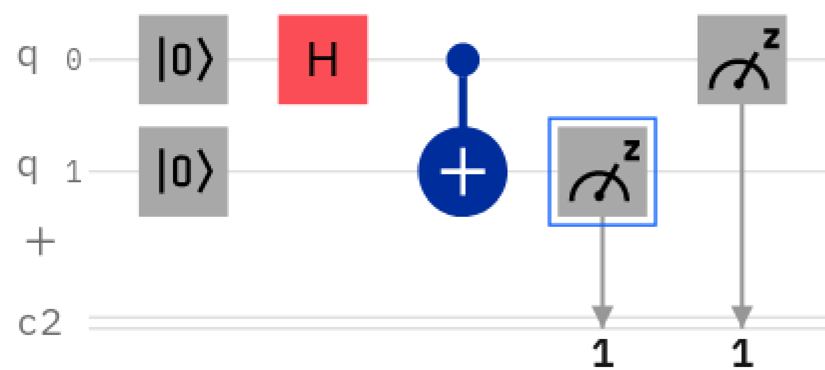
Minimal set of operators

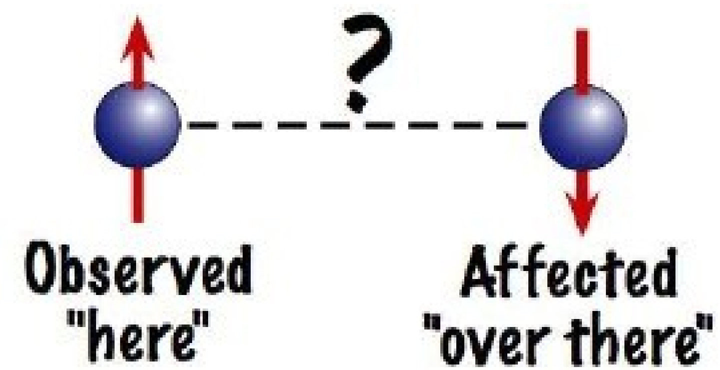
It has been proven that all quantum state of any system can be obtained by a combination of rotations and binary CNOT



Obtaining entanglement with CNOT and Hadamard

- Apply Hadamard operator on control qubit, target qubit stays |0> and then apply CNOT
- |00> $\xrightarrow{H}$ 1/√2 (|00>+|10>) $\xrightarrow{CNOT}$ 1/√2 (|00>+|11>)
- When measured, obtain always 0,0 or 1,1 with probability 1/2
- Still work when the qubits are very far away (quantum non locality)
- EPR paradox: information goes faster than light

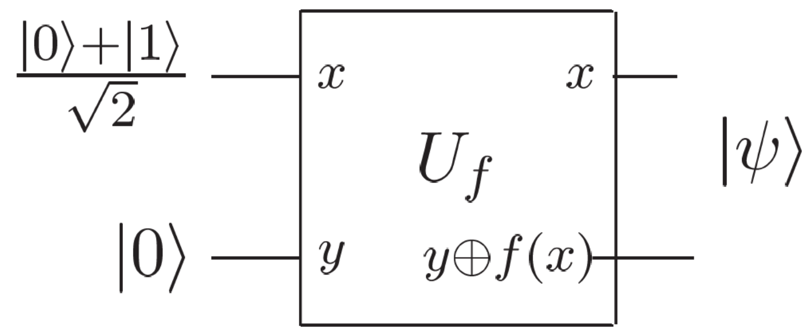




Tested over 1200km in satellite based experiment 2020 Yin & al

Quantum Parallelism

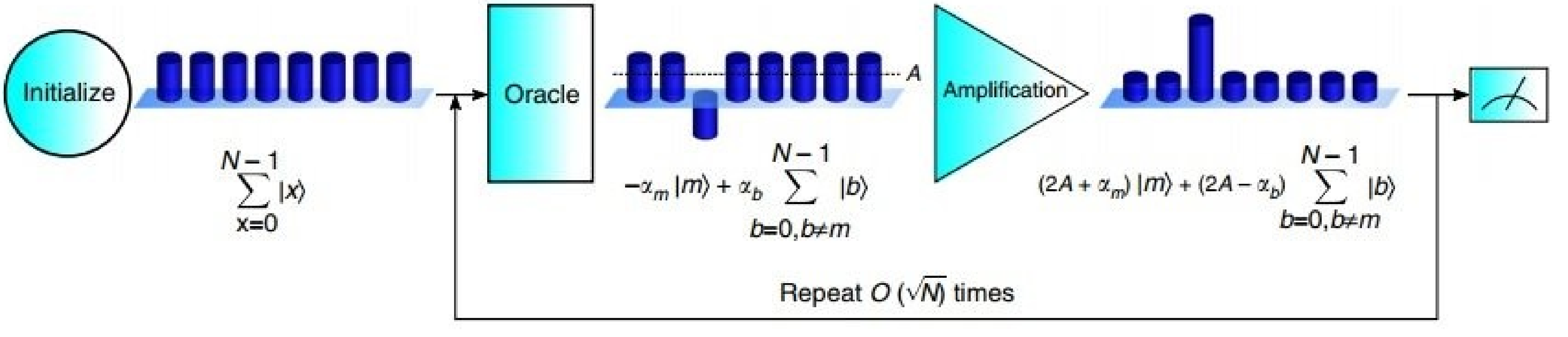
- Compute multiple values at the same time
- x is in |+> state
- Uf computes y+f(x)



- Result is 1/√2 (|0, f(0)>+|1, f(1)>)
- Read x and then f(x) until all value of x has been seen
  - If M(x)=0 M(Uf(x))=f(0)
  - if M(x)=1 M(Uf(x))=f(1)
- Extendable to any size of x

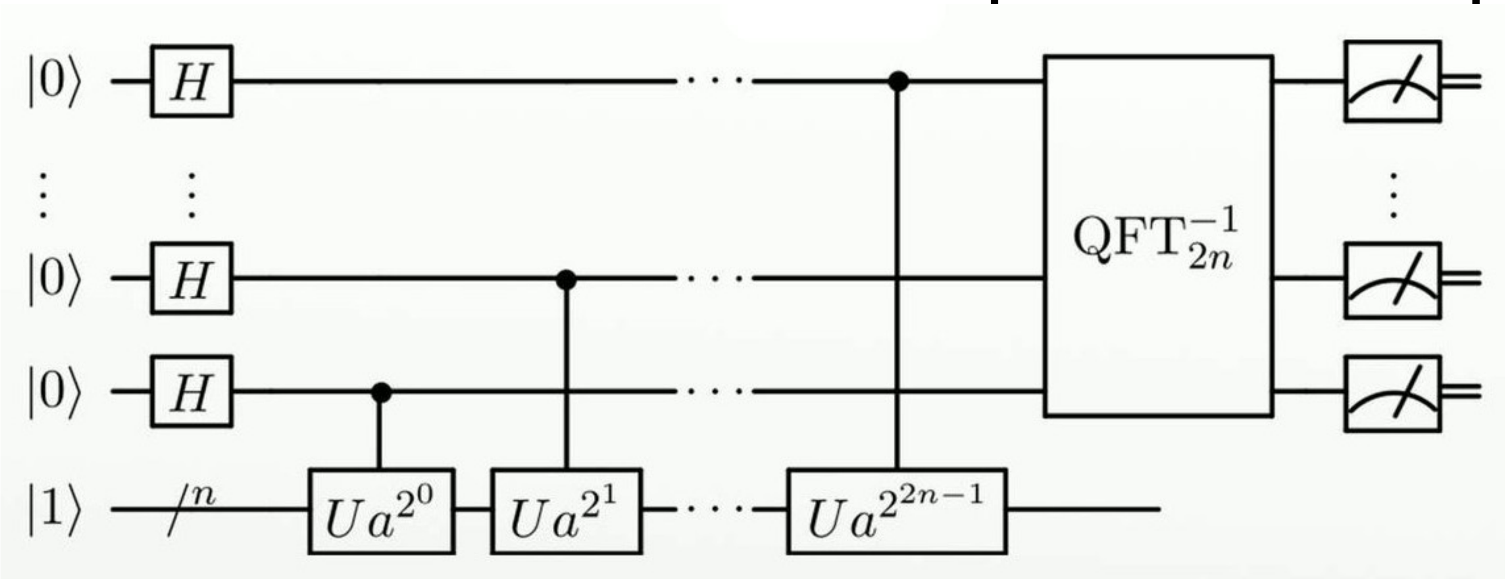
Grover Algorithm (1996)

- Search a name in a phone book by knowing a number (extendable to any kind of search with oracle)
- Solves the task of function inversion
- Classic computing is O(n)
- Based on quantum parallelism plus specific initialization called amplitude amplification
- Quantum complexity O(√n) (almost optimal)



Shor Algorithm (1994)

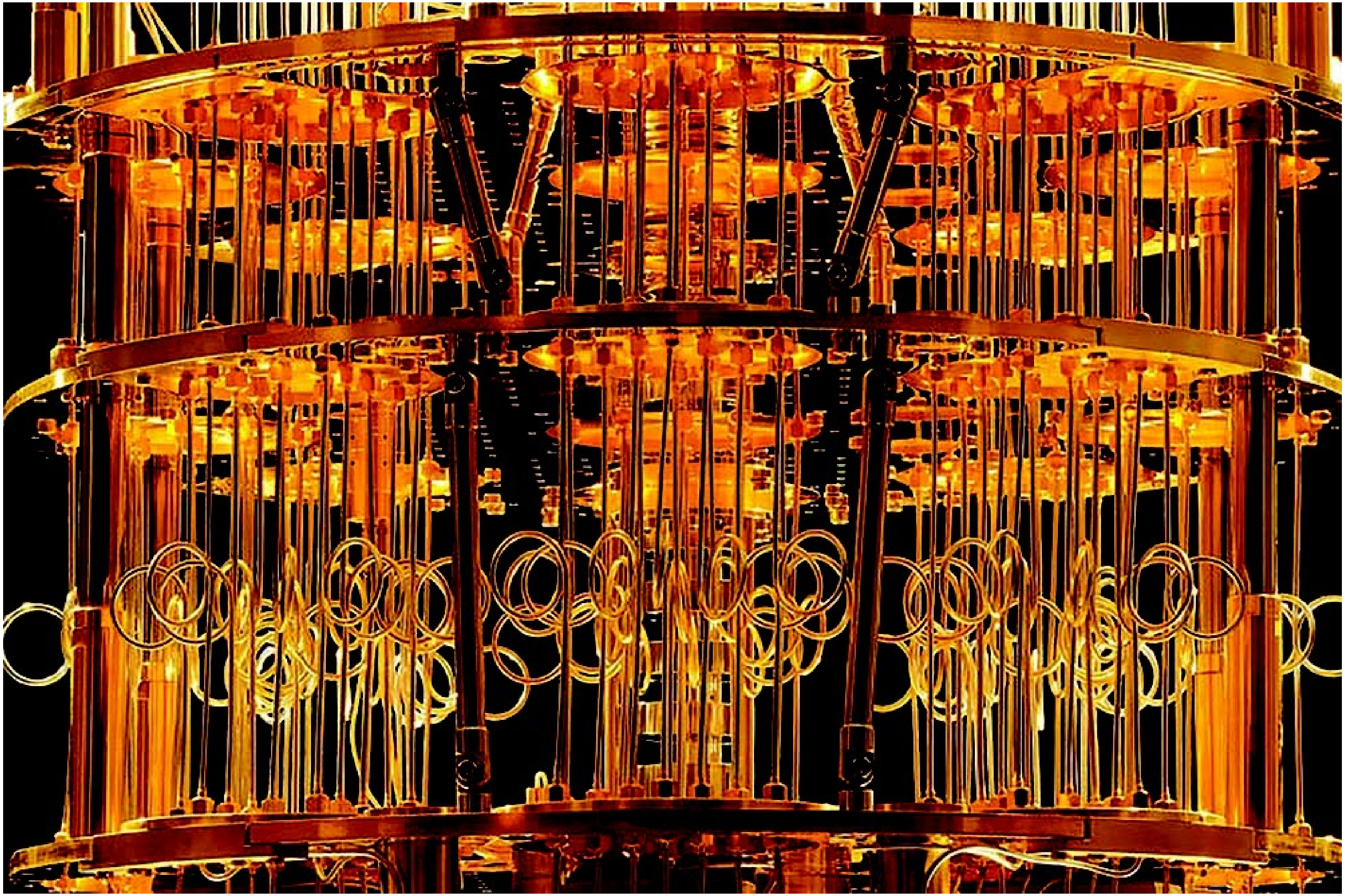
- Prime factor decomposition of arbitrary size N
- Complexity $O((\log N)^3)$ in time $O(\log N)$ in space
- End of traditional cryptography (RSA) based on hardness of factorization
- Induced a lot of interest for quantum computing



Programming languages

- Qiskit (IBM) python framework
- Cirq (Google) python framework
- Openqasm for quantum assembler
- Q# Microsoft
- Silq ETH Zurich
- Pennylane (differentiable for QML)

Quantum computers

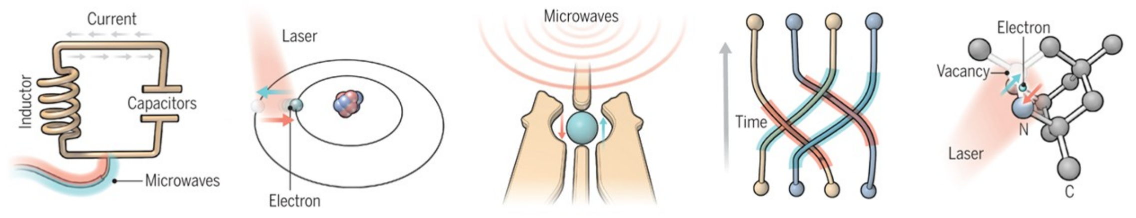


Plenty of implementations

- A lot of physical processes are qubits
  - spins
  - energy levels
  - photons
  - Josephson currents

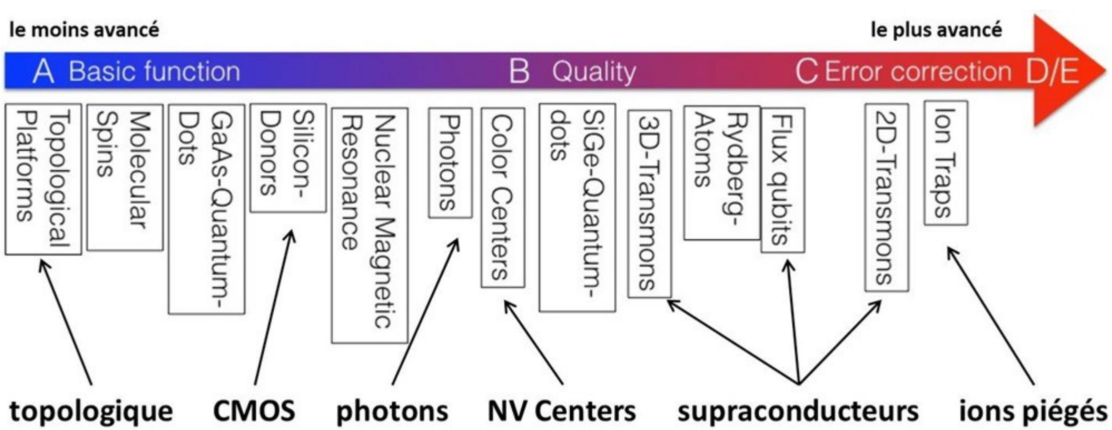
A bit of the action

In the race to build a quantum computer, companies are pursuing many types of quantum bits, or qubits, each with its own strengths and weaknesses.



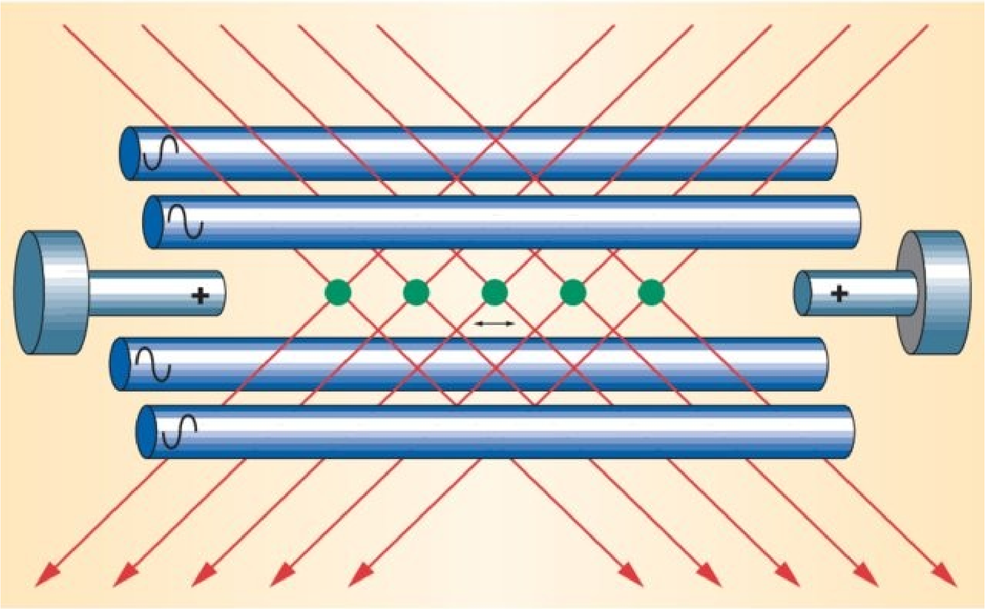
	Superconducting loops	Trapped ions	Silicon quantum dots	Topological qubits	Diamond vacancies
	A resistance-free current oscillates back and forth around a circuit loop. An injected microwave signal excites the current into super-position states.	Electrically charged atoms, or ions, have quantum energies that depend on the location of electrons. Tuned lasers cool and trap the ions, and put them in superposition states.	These "artificial atoms" are made by adding an electron to a small piece of pure silicon. Microwaves control the electron's quantum state.	Quasiparticles can be seen in the behavior of electrons channeled through semi-conductor structures. Their braided paths can encode quantum information.	A nitrogen atom and a vacancy add an electron to a diamond lattice. Its quantum spin state, along with those of nearby carbon nuclei, can be controlled with light.
Longevity (seconds)	0.00005	>1000	0.03	N/A	10
Logic success rate	99.4%	99.9%	~99%	N/A	99.2%
Number entangled	9	14	2	N/A	6
Company support	Google, IBM, Quantum Circuits	ionQ	Intel	Microsoft, Bell Labs	Quantum Diamond Technologies
Pros	Fast working. Build on existing semiconductor industry.	Very stable. Highest achieved gate fidelities.	Stable. Build on existing semiconductor industry.	Greatly reduce errors.	Can operate at room temperature.
Cons	Collapse easily and must be kept cold.	Slow operation. Many lasers are needed.	Only a few entangled. Must be kept cold.	Existence not yet confirmed.	Difficult to entangle.

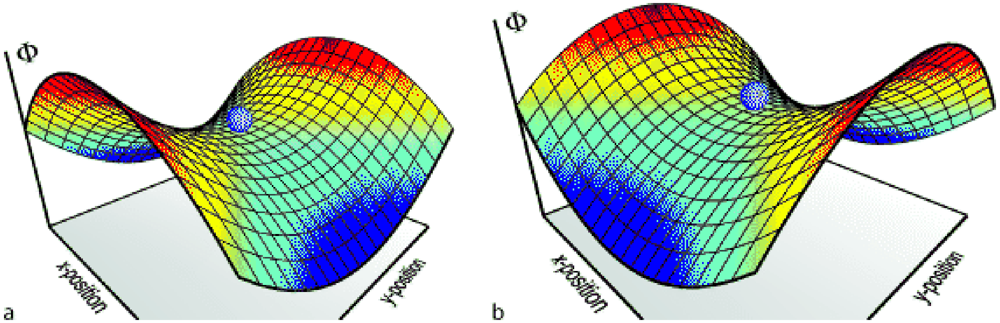
Note: Longevity is the record coherence time for a single qubit superposition state, logic success rate is the highest reported gate fidelity for logic operations on two qubits, and number entangled is the maximum number of qubits entangled and capable of performing two-qubit operations.



Trapped ions

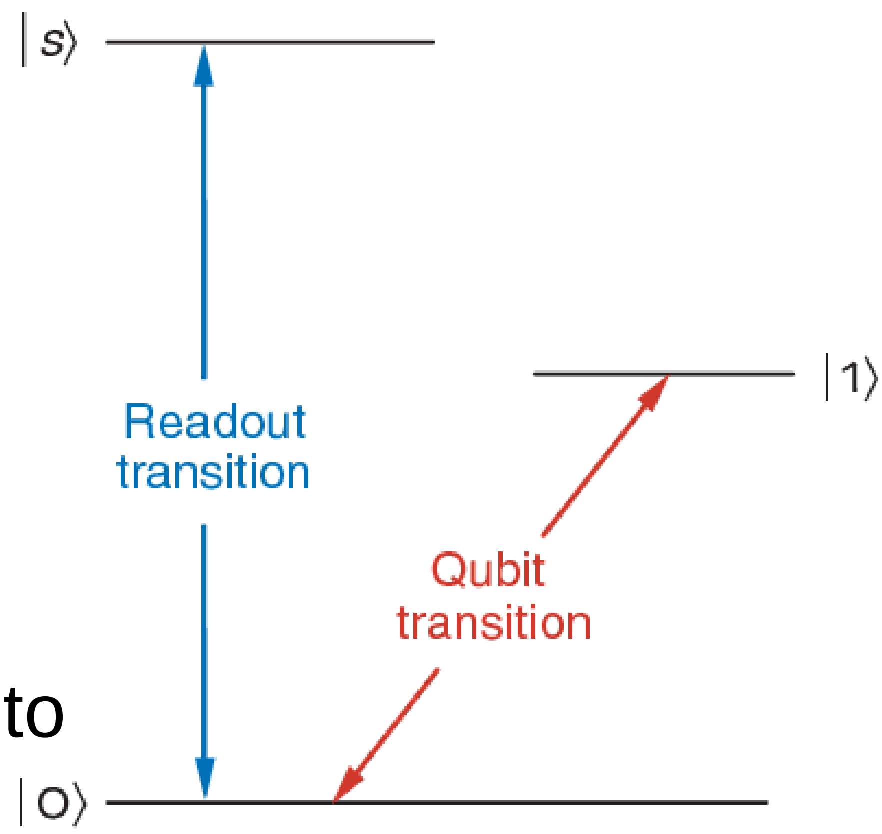
- Linear Paul trap
- Atoms are ionized by removing 1 valence electron
- Positive endcap
- Oscillating electro-magnetic field on bars
- The oscillation garantees the stability



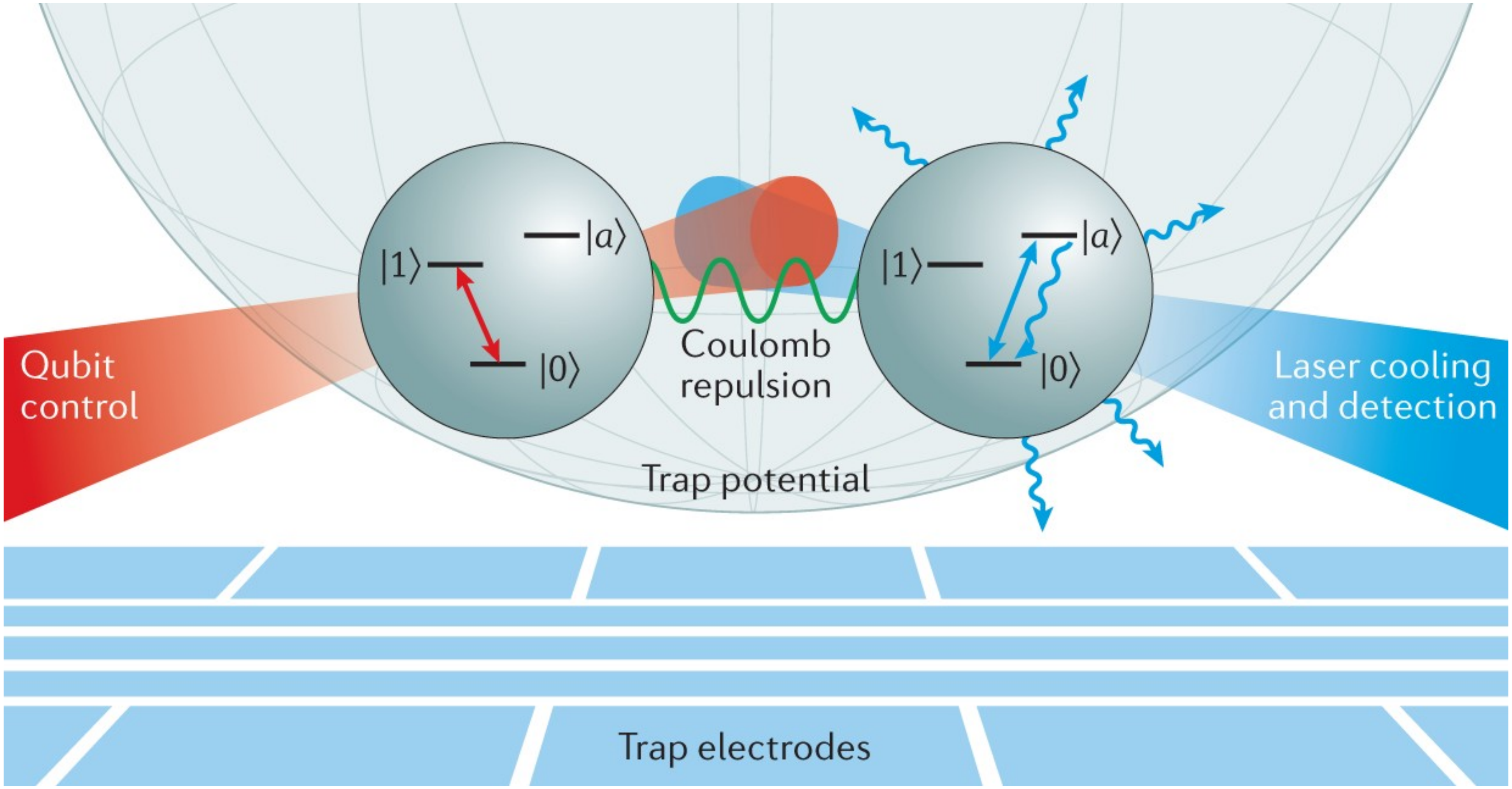


Ions

- Typically alkaline earth atoms (Be^+, Mg^+, Ca^+, Sr^+) or ytterbium Yb^+
- Produced by an « oven »
- Ionized by a laser
- Energy levels
  - Ground state |0>
  - Short lived excited state |s> strongly coupled by a transition to the ground state
  - Long lived excited state |1>



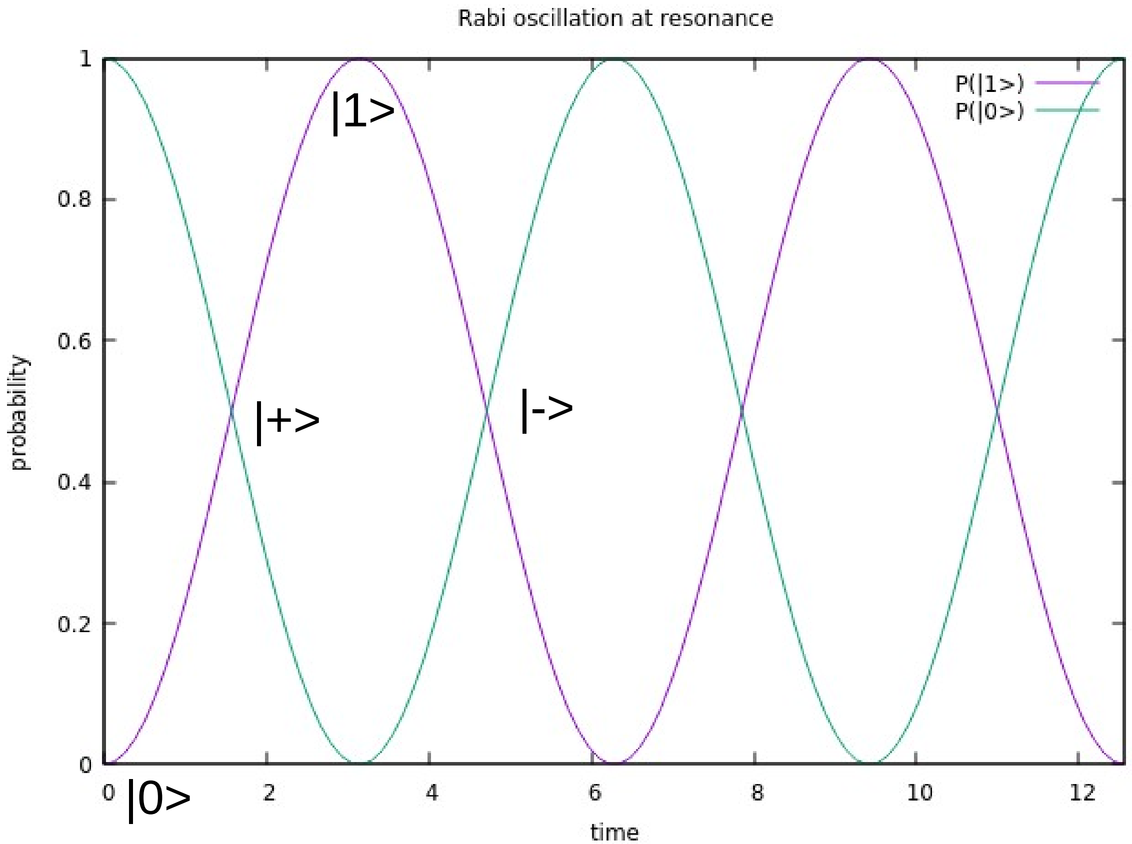
Laser qubit operations

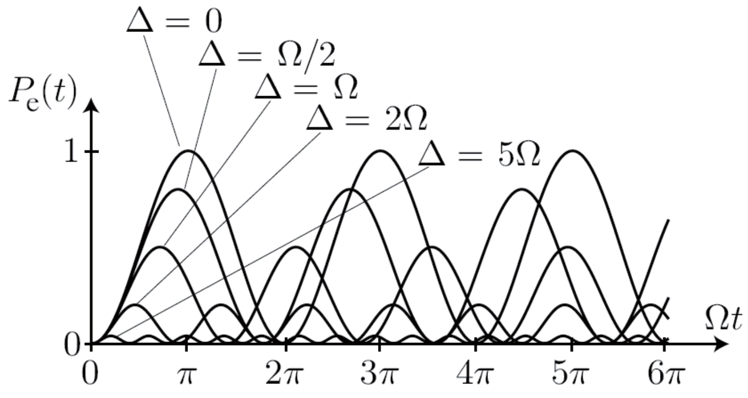


- Lasers are used to change the ion level
- Measurement by fluorescence
- Constraints: advanced vacuum, focused lasers

Implementing rotation

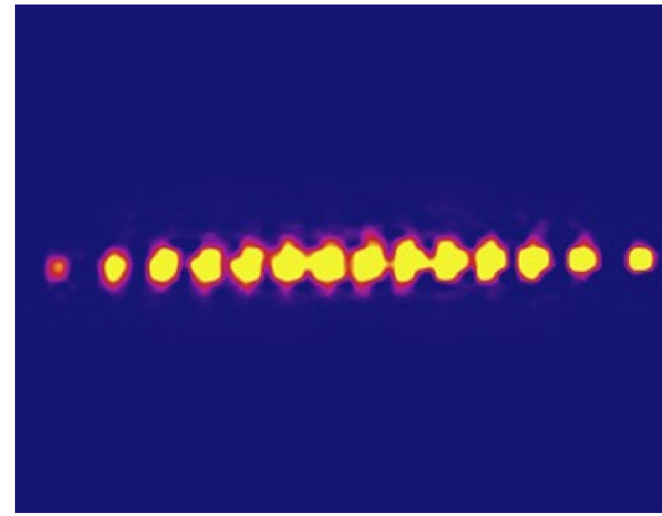
- Rabi precession implements rotation
- Change the mix between pure states
- Induced by photon interaction (laser)

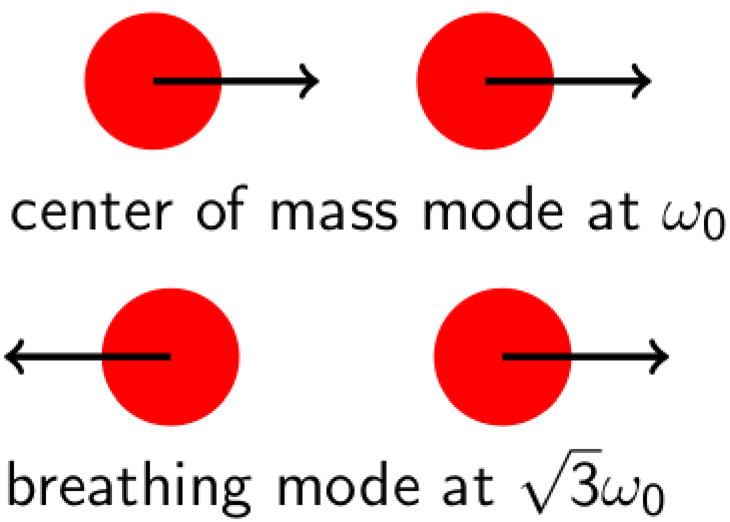


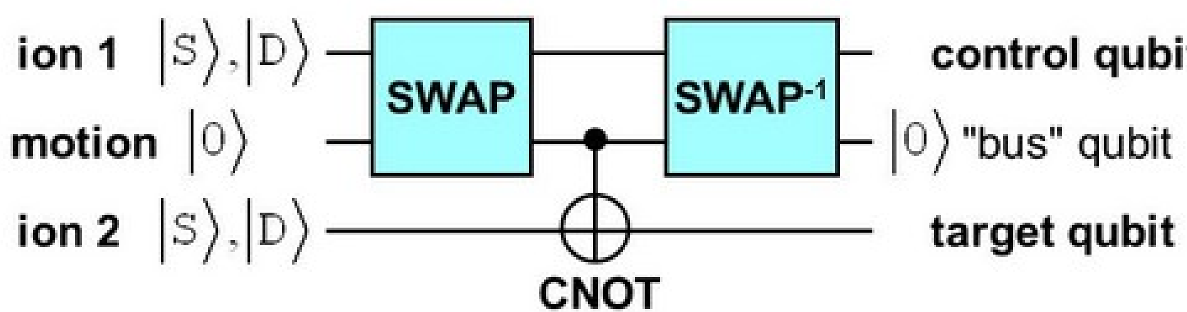


Entanglement

- No direct interaction between ions
- Entanglement is based on state transfer to common motion of the ions
- phonon with two states
- Swap state between individual ion and phonon
- Cirac-Zoller CNOT implementation



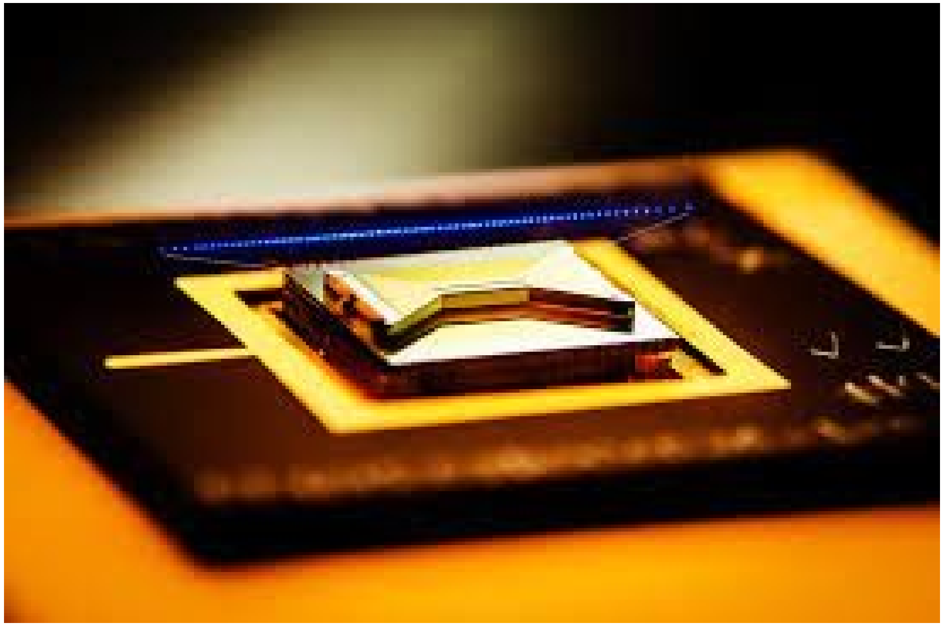


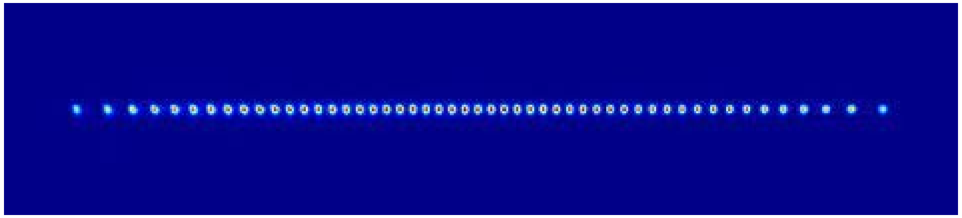


Rainer Blatt & David Wineland, Entangled states of trapped atomic ions, Nature 2008

IonQ Aria

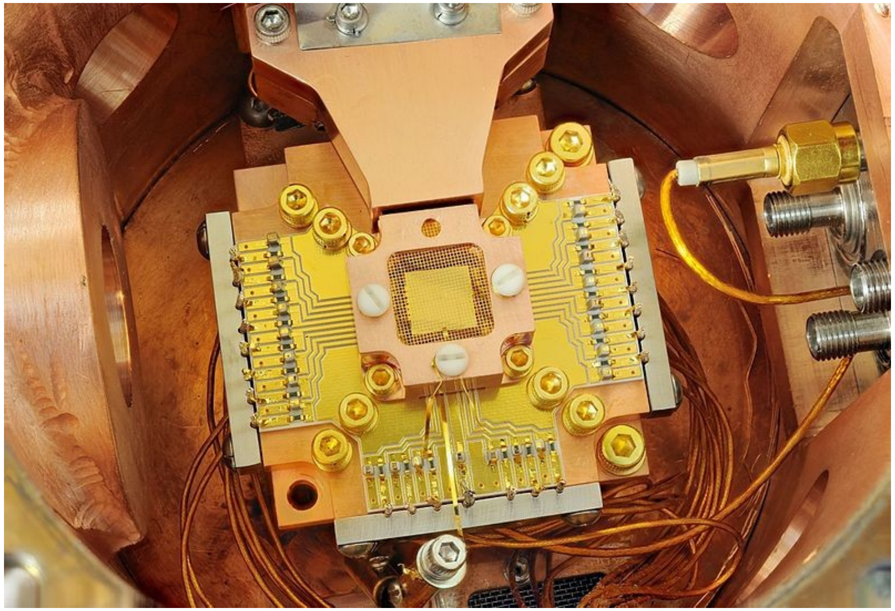
- 11 qubits fully tested (up to 160 in future)
- Linear arrangement
- Available on Amazon cloud (AWS)
- IonQ (Maryland)

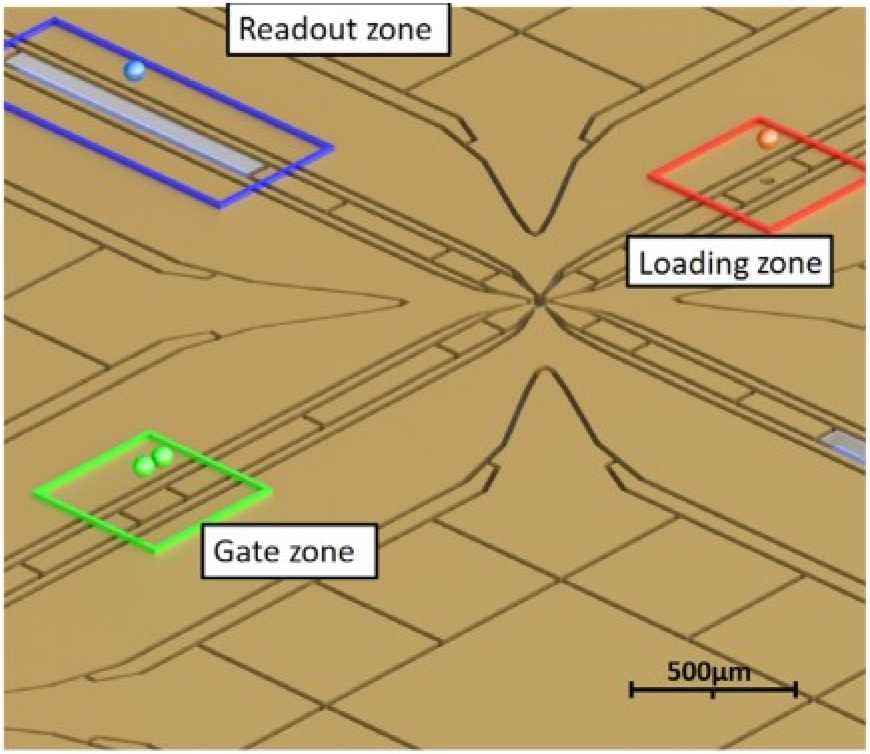




Future implementation

- Ions are inside a chip in a dedicated spot
- Ions can migrate to be entangled with others ions
- Magnetic field simplify the rotations (microwaves instead of lasers)

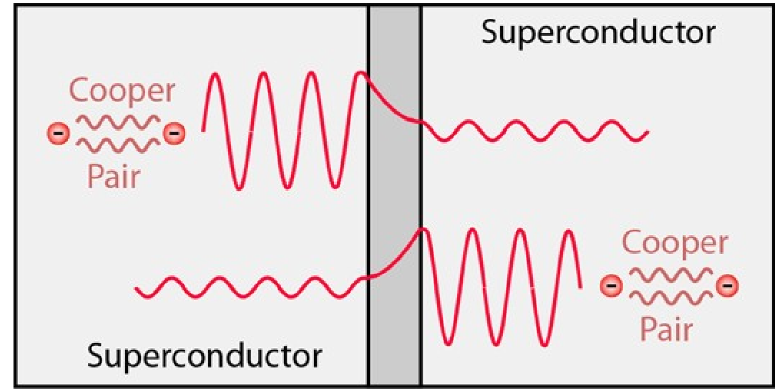


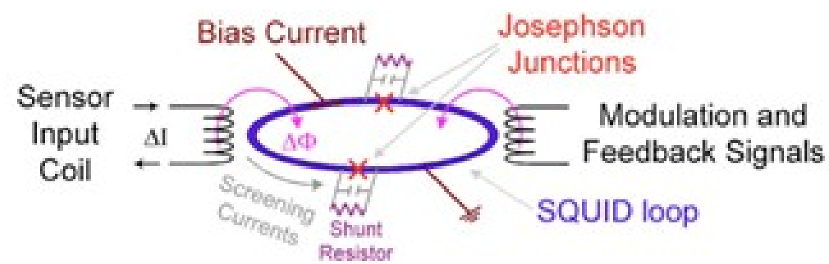


Lekitsch & al, Blueprint for a microwave trapped ion quantum computer, 2017

Superconducting loops

- Josephson current
- Appear spontaneously between two supraconducter separated by a thin isolating barrier (tunnel effect)
- Phase encoding qubit (2 states only because very cold)
- Rotations controlled by conducted microwaves
- Measurement done by a magnetometer inside the circuit
- Constraints : superconducting temperature, very low decoherence time
- Actors: IBM, Google, Intel, D-Wave





From https://www.oezratty.net/wordpress/2018/comprendre-informatique-quantique-qubits/

Multi-qubits operations

- Only Conditional phase (CP) is available on superconducting systems

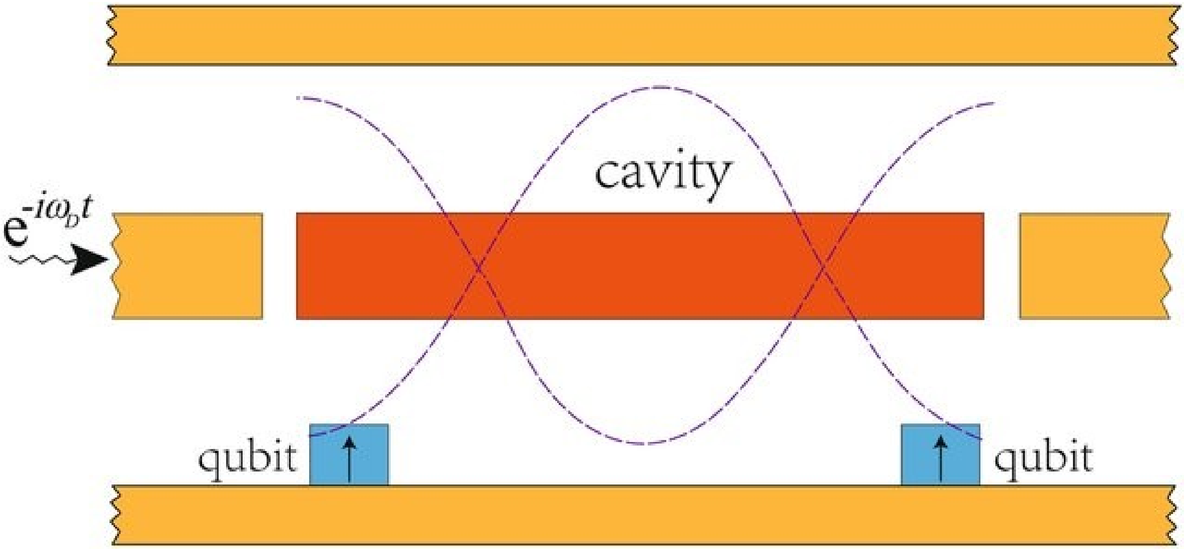
$$CP(\gamma) = \begin{bmatrix} e^{i\gamma/2} & 0 & 0 & 0 \\ 0 & e^{-i\gamma/2} & 0 & 0 \\ 0 & 0 & e^{-i\gamma/2} & 0 \\ 0 & 0 & 0 & e^{i\gamma/2} \end{bmatrix}$$

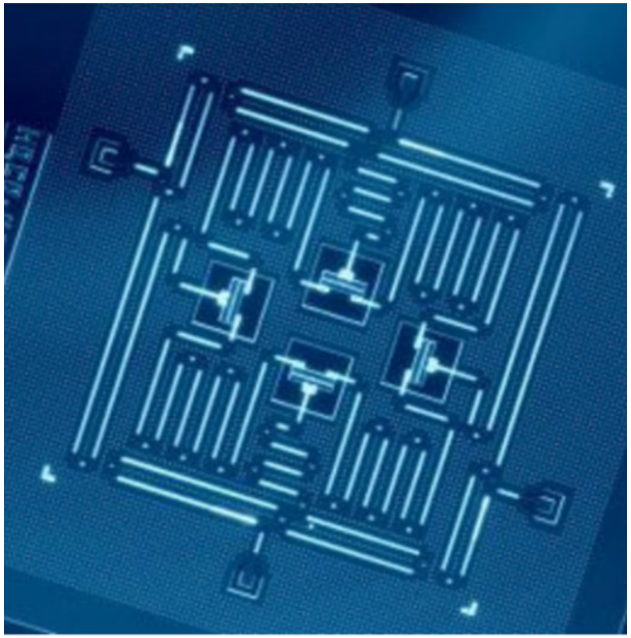
- CNOT is implementable from CP and rotations

$$CNOT_{12} = e^{i5\pi/4} R_{x2}(\pi/2) R_{z2}(\pi/2) R_{x2}(\pi/2) R_{z2}(\pi) R_{z1}(\pi/2) \times CP(\pi/2) R_{x2}(\pi/2) R_{z2}(3\pi/2) R_{x2}(\pi/2)$$

CP implementation

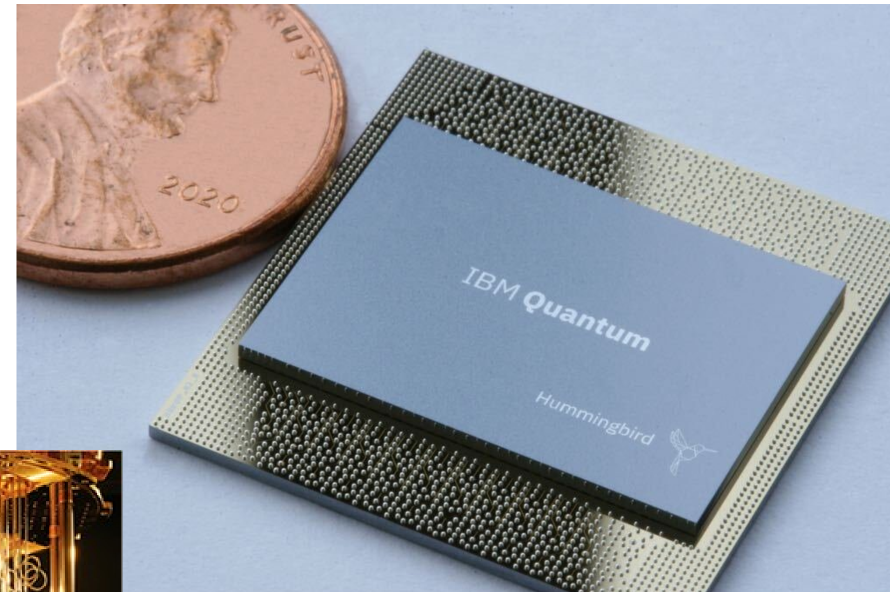
- Use a « Transmon », transmission line shunted plasma oscillation qubit
- Superconducting charge qubit (reduced noise sensibility)
- Controlled by microwaves



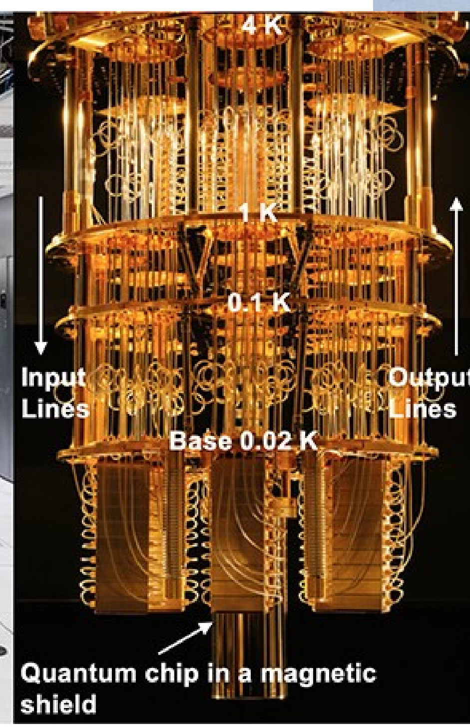


IBM Rochester

- 53 superconducting qubits
- Limited connectivity
- Available through dedicated cloud (IBM-Q)

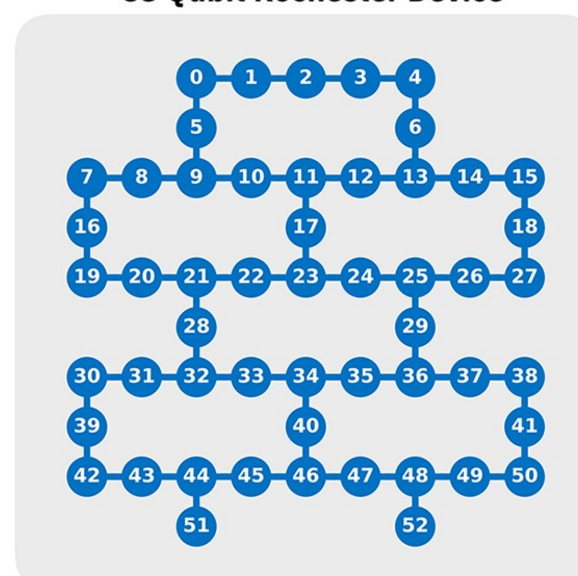


Dilution fridge setup: outside view



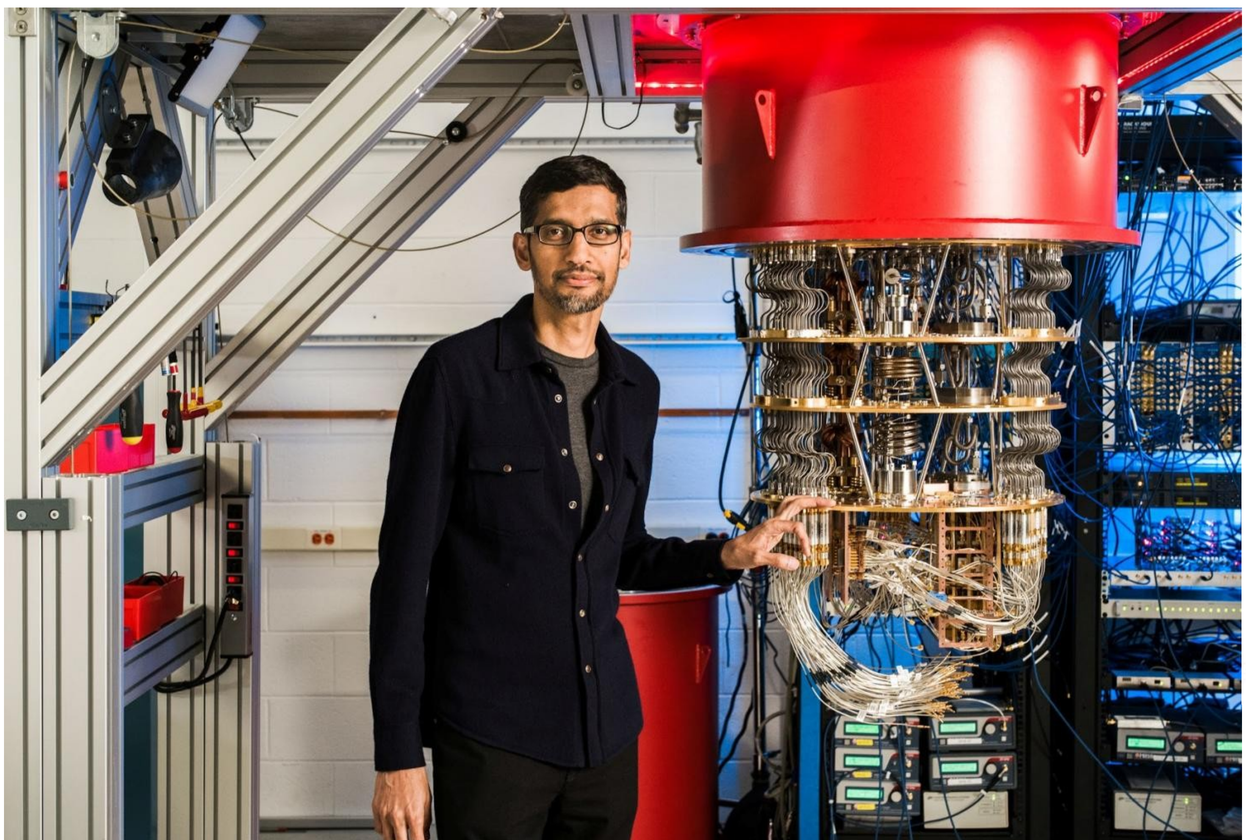
Dilution fridge setup: inside view

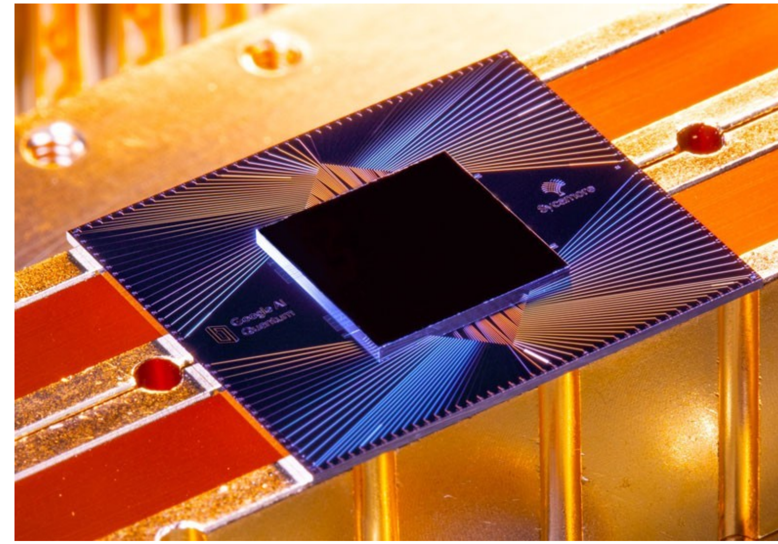
53 Qubit Rochester Device

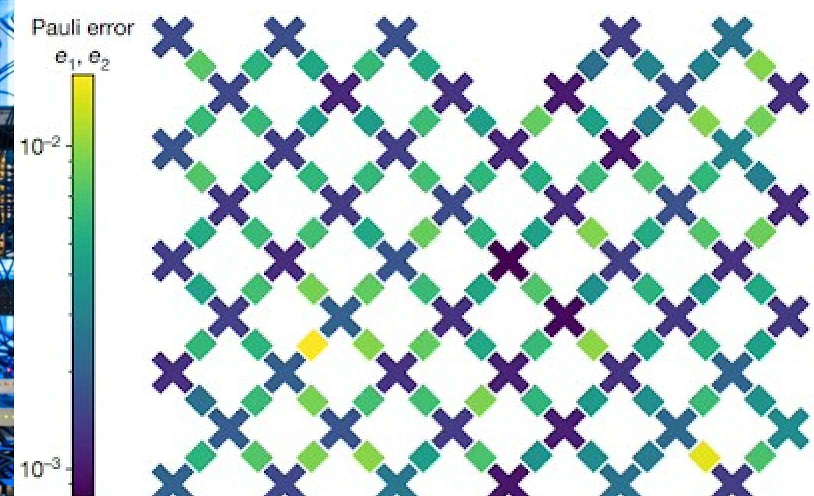


Google Sycamore

- 54 superconducting qubits
- 4 neighbours connectivity
- Available through Amazon cloud (AWS)







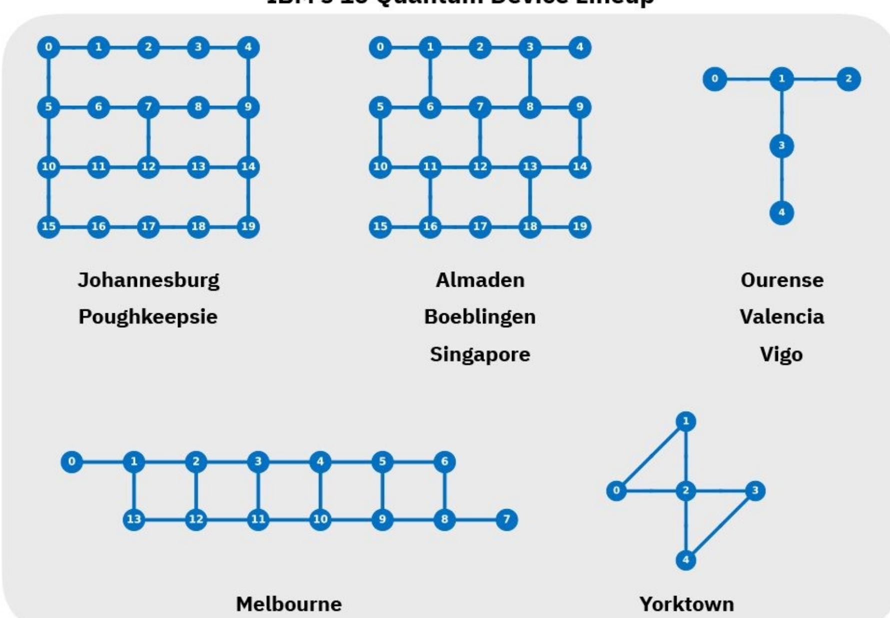
Transpilling

- Operation transforming code to operative sequence
- No compilation stricto sensu
- Affectation of qubits
- Operation are encoded and timed as effector actions (laser pulse, microwave pulse…)
- Low level gates are rotations and dedicated entanglement operators

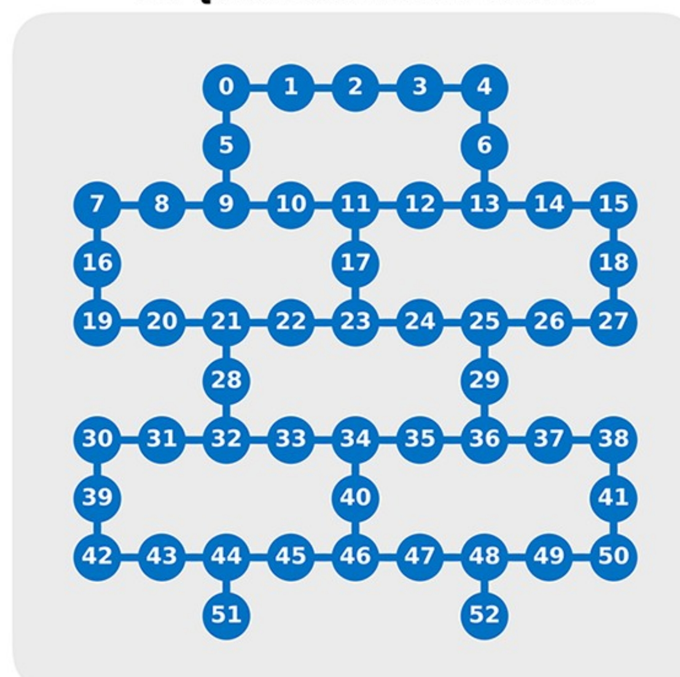
NISQ Era (1)

- Noisy Intermediate Scale Quantum
- Low number of qubits (from 1 to 100), low connectivity

IBM's 10 Quantum Device Lineup

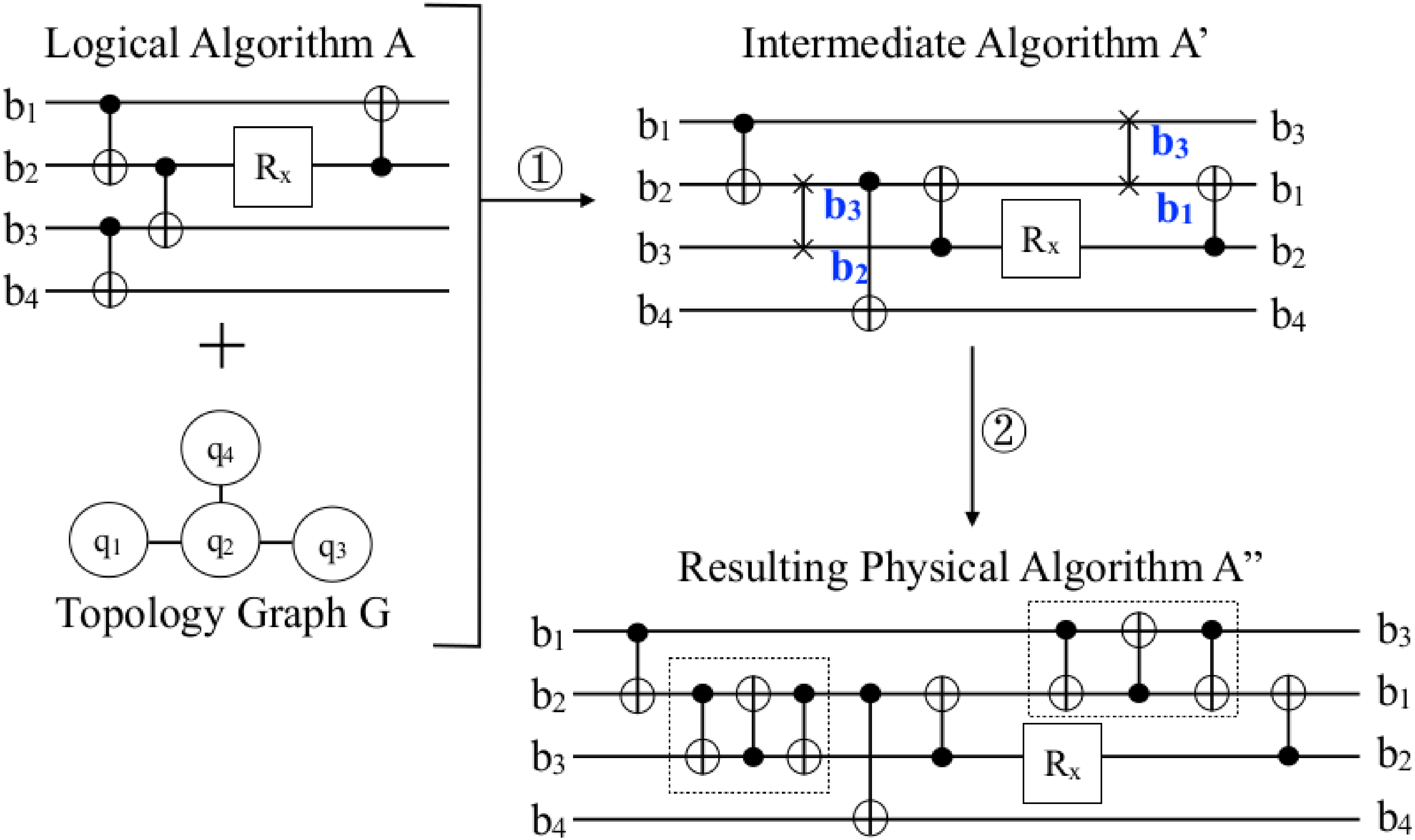


53 Qubit Rochester Device

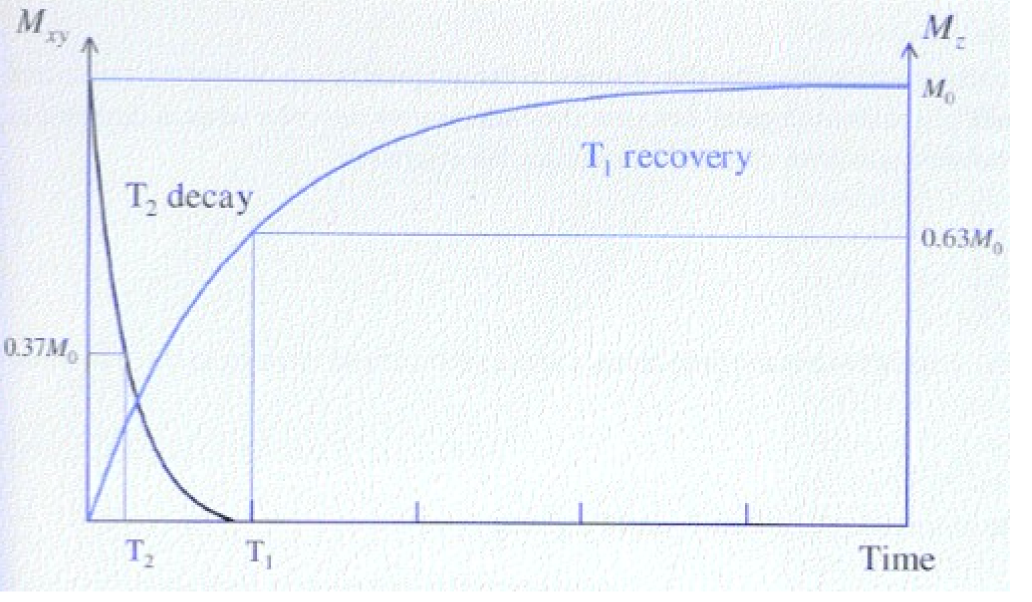


From https://www.ibm.com/blogs/research/2019/09/quantum-computation-center/

Algorithm rewriting for topology



NISQ Era (2)



- Noisy computers
  - spin-spin relaxation (decoherence)
  - T2=5.10^{-5}s at worst (superconducting), best understood as number of operations (~1000)
  - spin-lattice relaxation (thermodynamic equilibrium) T1

NISQ Limitations

- The topology induces an increase of gate number (swapping)
- The emulation of the standard gate by hardware dedicated gates
- The noise and decoherence limit the depth of the circuits
- The number of qubits is very low

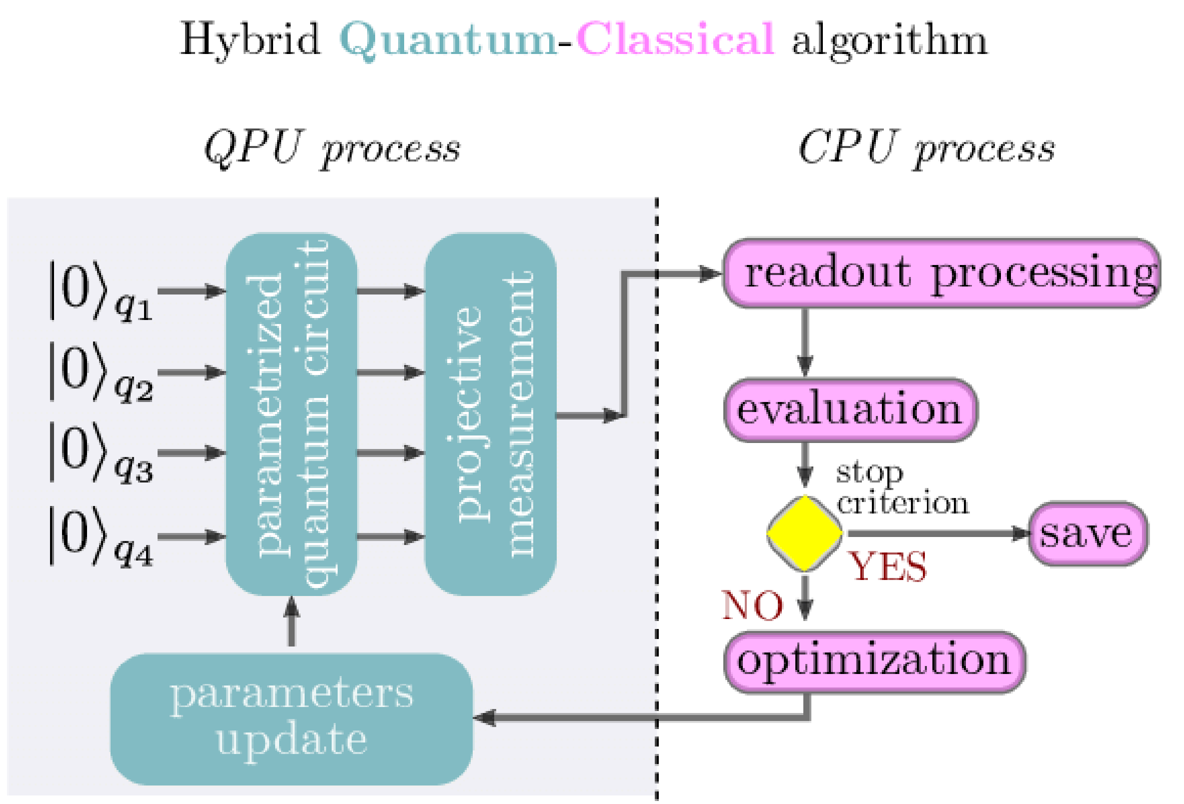
→ Only very simple circuits can be implemented now

Quantum machine learning

Problematics

- Implementing neural-network-like model and gradient descent on quantum computer
- Reconciling two computing models
  - Estimating
    - $\hat{y} = M(x , \theta)$
    - $\hat{y} = U(x , \theta)$ with U unitary
  - Training
    - loop : $\theta = \theta - \alpha \, . \, \partial L(y , M(x , \theta)) / \partial \theta$
    - $\theta = O(x , \theta , y)$

Variational Hybrid QC Algo.



- Only a small part is handled by the quantum computer (adapted to NISQ)
- The quantum part encodes the problem in qubit formalism (Ansatz)

McClean & Al, The theory of variational hybrid quantum-classical algorithms, 2015

QNN First generation

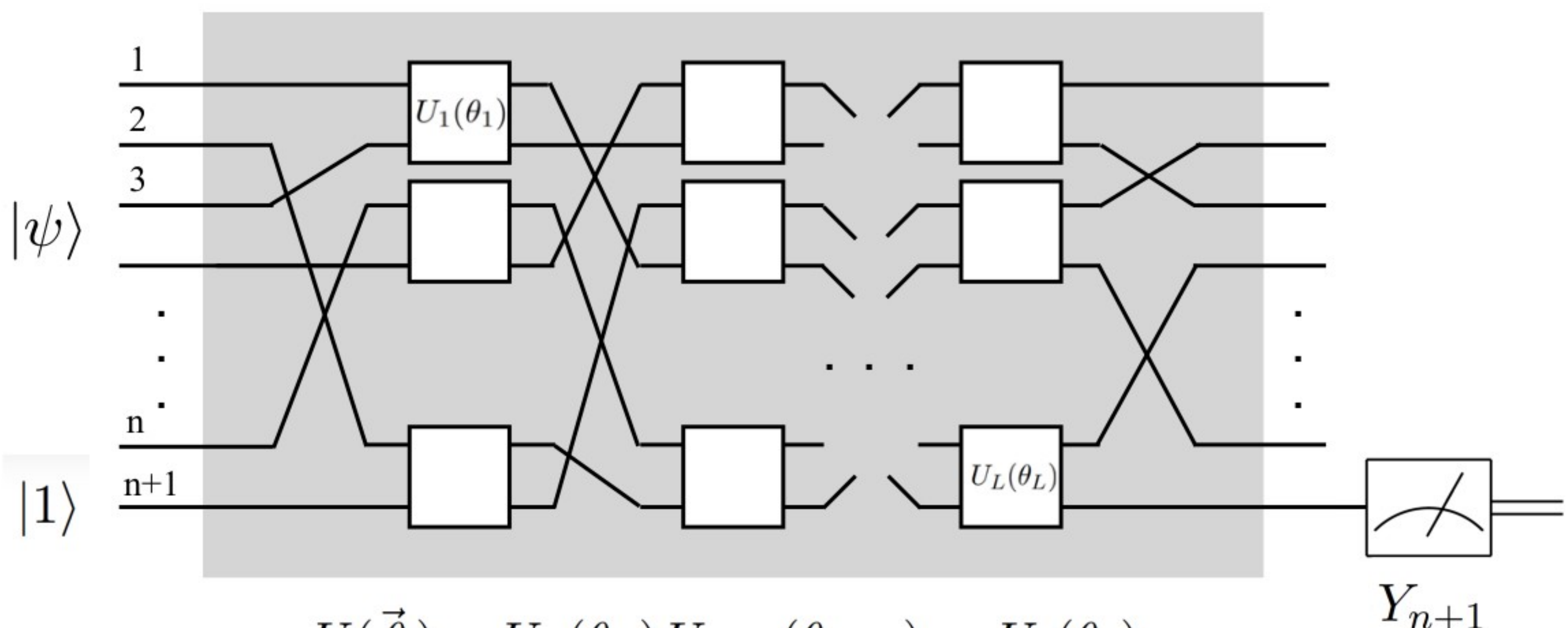
- Farhi & Neven (Google), *Classification with Quantum Neural Networks on Near Term Processors*, 2018
- Schuld, Bocharov, Svore & Wiebe, *Circuit-centric quantum classifiers*, 2018
- Declination of the Variational hybrid computation to Machine Learning
- First successful implementations

Classification with QNN (1)

- Farhi & Neven (Google), 2018
- One of the first « QNN » implementation
- Adapted to both classical or quantum inputs z
- Designed for binary classification : binary label l(z) (no label noise)
- Based on variational hybrid computation and gradient descent

Classification with QNN (2)

- Based on qubit data encoding |ψ> is the input plus one ancillary qubit
- A sequence of binary unitary parametrized operators U_i
- Measurement of the ancillary bit (the answer) converted from probability $|a|^2$ to {-1,1}



$$U(\vec{\theta}\,) = U_L(\theta_L)\,U_{L-1}(\theta_{L-1})\ldots U_1(\theta_1)$$

Classification with QNN (3)

- The operator part is evaluated by the quantum circuit (mean of M measurement)

$$< z, 1|U^\dagger(\theta)Y_{n+1}U(\theta)|z, 1 >$$

- The loss function is evaluated on the classical part

$$L(\theta, z) = 1 - l(z) < z, 1|U^\dagger(\theta)Y_{n+1}U(\theta)|z, 1 >$$

- Learning by gradient descent, calculated by nulmerical differentiation

$$\frac{df}{dx}(x) = \frac{(f(x+\epsilon) - f(x-\epsilon))}{2\epsilon} + O(\epsilon^2)$$

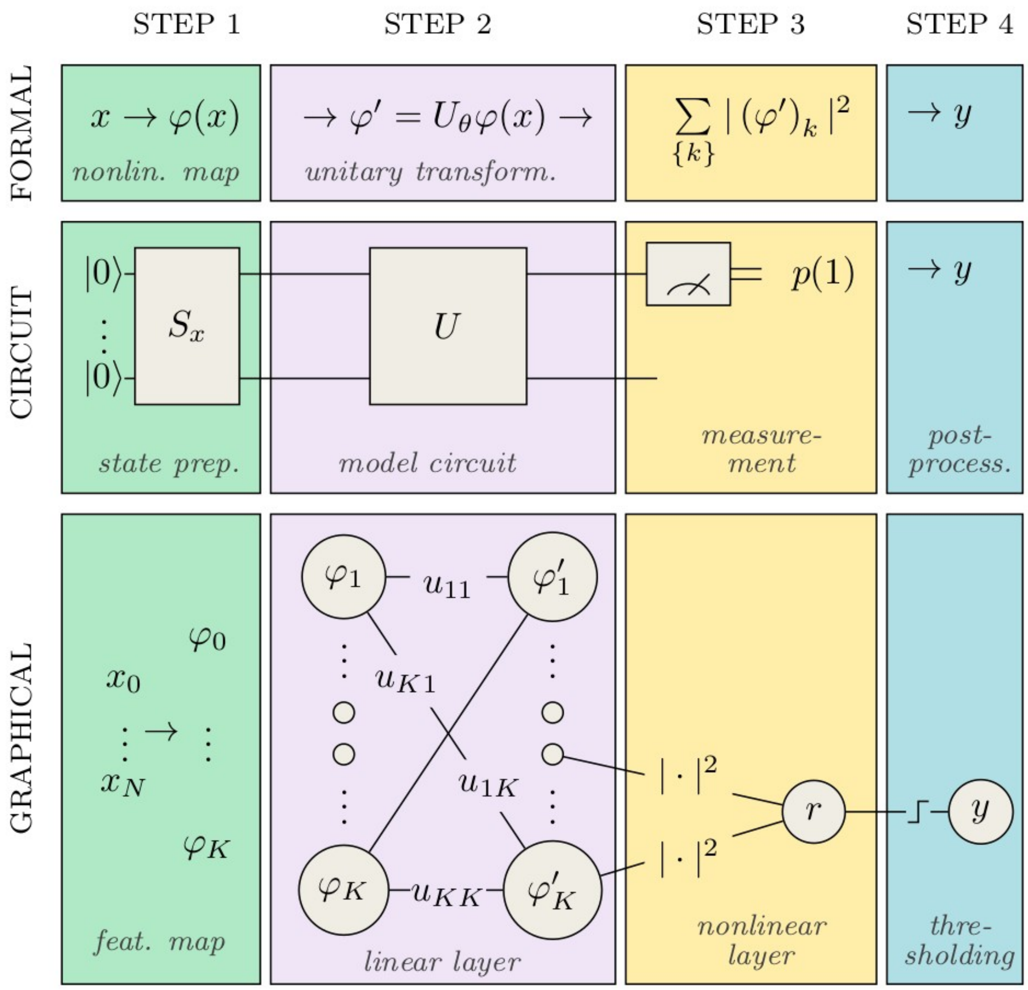
→ obtained by 2L*M quantum circuit evaluation

Classification with QNN (4)

- Tested on binary parity and majority
- $U_i(\theta)$ are designed specifically for these problems
- Tested on downsampled MNIST digits
- All tests are conclusive, the network learns
- Nice proof of concept

Standard scheme

- The scheme used in Farhi and Schuld has been extensively used everywhere



From Schuld & al 2018

Basis encoding

- The data are digitized and then encoded in a sequence of qubit
- |1> or |0> are obtained by initialization at |0> state and rotation Ry(π)
- Example
  - x1=5=0b101
  - x2=6=0b110
  - encoded by |101110>
- very qubit consuming and time consuming

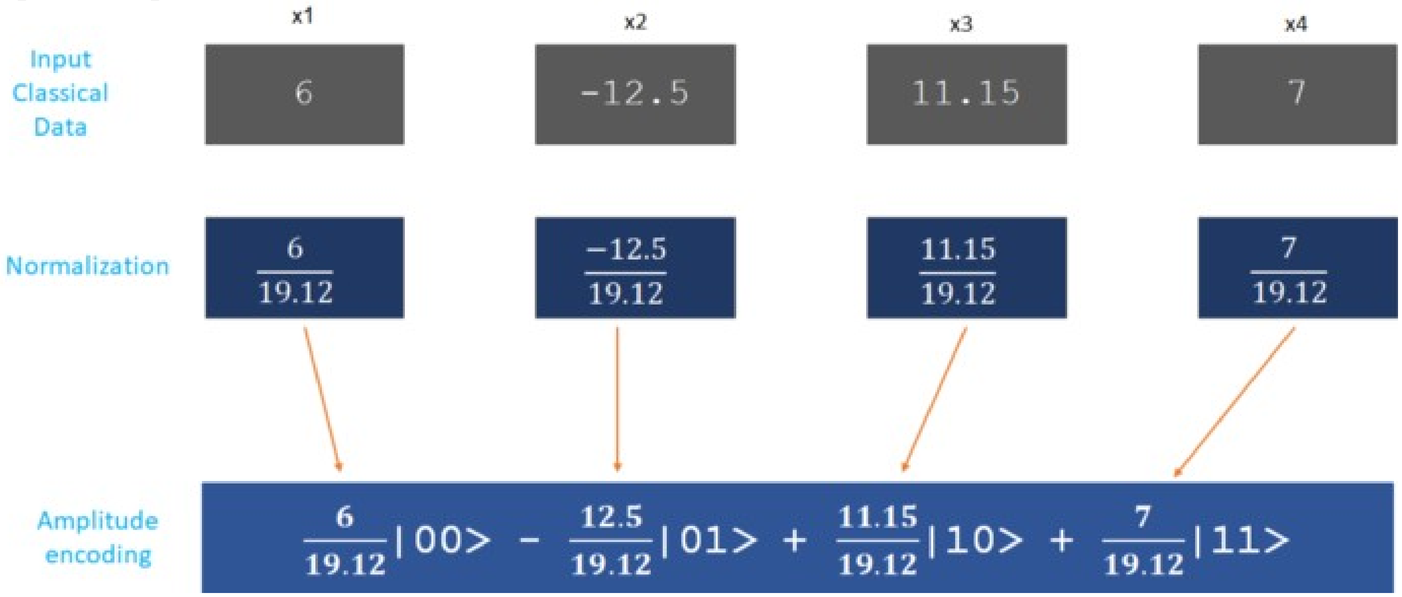
Quantum associative memory

- the data are encoded a a superposition to reduce the number of qubits

Input variable	Input Classical Data	Binary Number	Basis encoded Quantum Data	QUAM encoded value
X1	10	1010	$\|1010\rangle$	$\frac{1}{\sqrt{3}}\|1010\rangle + \frac{1}{\sqrt{3}}\|1111\rangle + \frac{1}{\sqrt{3}}\|1000\rangle$
X2	15	1111	$\|1111\rangle$	
X3	8	1000	$\|1000\rangle$	

Amplitude encoding

- encode the data as the coefficients of a superposition of states



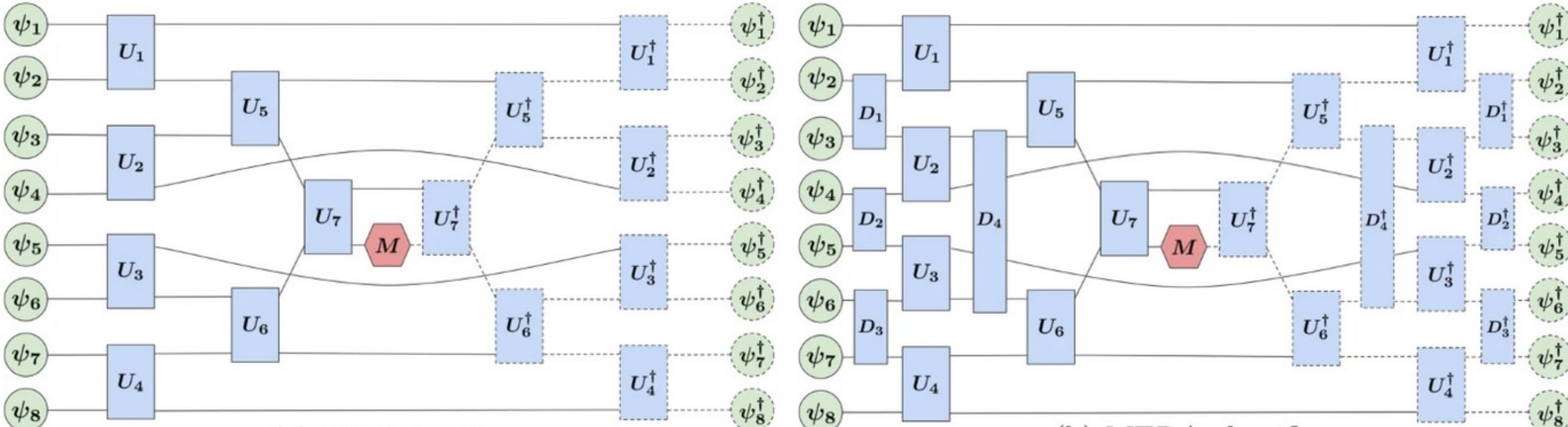
- Use very few qubits :log2(n)
- Very time consuming exp(n) not compatible with NISQ

Angle encoding

- Each data is encoded as an angle on a single qubit by applying $Ry(x_i)$ on |0>
- x_i has to be normalized (over π)
- The best trade-off between time and qubits (n) : used almost everywhere
- Could be densified using the phase (dense angle encoding) → n/2 qubits

Processing

- As measure is often done on only one qubit, some kind of entanglement has to be implemented
- TTN and MERA are good candidates for regular architecture
- TTN is very economical in parameters
- MERA is a bit more efficient (more parameters)



(a) TTN classifier (b) MERA classifier

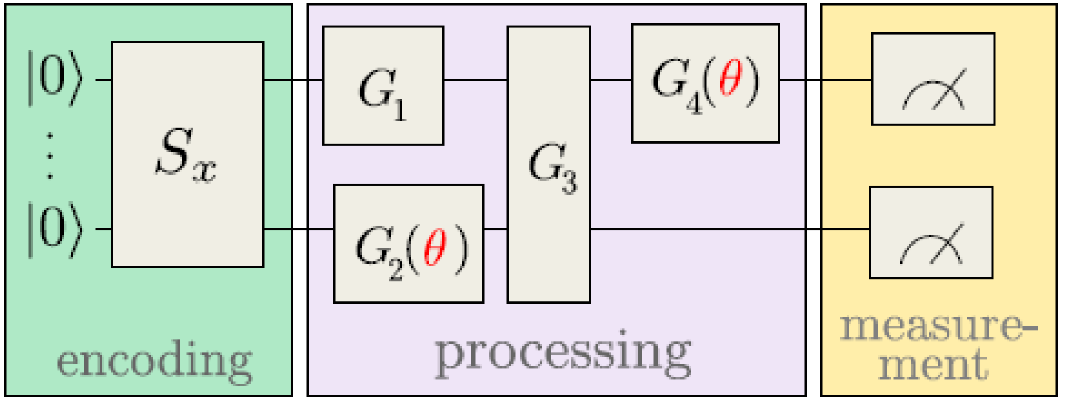
Grant & al, Hierarchical quantum classifiers, 2018, 1804.03680

Processing limit

- Whatever is the nature of the parametrized circuit, it can expressed as a single global unitary circuit
- A unitary circuit is linear in its inputs
- Thus this kind of encoding / processing scheme is linear in its outputs
- Data are plunged into a bigger space (Hilbert space) and discriminated by a linear classifier
- This is kernel method, not QNN

Every QNN is kernel method ?

- Article from Maria Schuld « Quantum machine learning models are kernel methods » 2021
- Encoding is the kernel
- In my opinion, only the scheme induces kernel methods, not the quantum nature



QNN for HEP

- Plenty of articles using this design for HEP analysis
  - Quantum Machine Learning in High Energy Physics, Guan & al, 2020, 2005.08582 (survey)
  - Performance of particle tracking using a quantum graph neural network, Tüysüz & al, 2021, 2012.01379
  - A quantum algorithm for the classification of supersymmetric top quark events, Bargassa & al, 2021, 2106.00051
  - Dual-parametrized quantum circuit GAN model in HEP, Chang & al, 2021, 2103.15470
  - ….

QNN first generation drawbacks

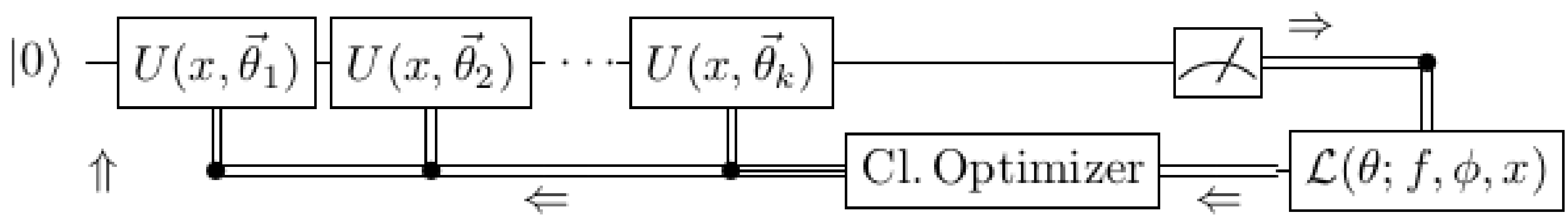
- Limited to kernel methods
- No integrated non-linearity in the quantum part
- Size of entries limited by number of qubits
- Numerical problems on differentiation

QNN second generation

- Based on 2 new techniques
  - Re-uploading
  - Shift-rule differentiation

The re-uploading technique

- Published in 2019 by Perez-Salinas & al « Data re-uploading for a universal quantum classifier »
- input is taken as parameter of every operators instead of input of a global operator
- Non linerarity appears
- Save a lot of qubits



$$L(i) = U(\theta_i^{(k)} + w_i^{(k)} \circ x^{(k)}) \dots U(\theta_i^{(1)} + w_i^{(1)} \circ x^{(1)})$$

Re-uploader as universal approximator

- Published in 2021 by Perez-Salinas & al « One qubit as a universal approximant »
- A single qubit can approximate any bounded function by using the input x multiple time as operator parameter
- Heavy tests give satisfying results on non linearity like tanh and ReLU
- Give a hope for implementing real QNN on quantum circuits

Improve the differentiation

- The numerical differentiation on noisy device is almost intractable (too small shift)

$$\frac{df}{dx}(x) = \frac{(f(x+\epsilon) - f(x-\epsilon))}{2\epsilon} + O(\epsilon^2)$$

- A property of some quantum operator has been discovered called « parameter shift rule »

$$\frac{\partial G}{\partial \theta} = G(\theta + s) - G(\theta - s)$$

- s is not small (it is a fixed value)
- The derivative is exact
- The other operators are decomposable in sequence of shift-rule operators

- Mitarai & al, Quantum circuit learning, 2018
- Schuld & al, Evaluating analytic gradients on quantum hardware, 2018

Shift rule example

- Let’s consider f(x)=sin(x) and its derivative cos(x)
- We know as a property of sin and cos

$$sin(a+b) - sin(a-b) = 2cos(a)sin(b)$$

- Thus

$$\forall s \frac{df}{dx} = cos(x) = \frac{sin(x+s) - sin(x-s)}{2sin(s)}$$

- We can choose any s, for example π/2

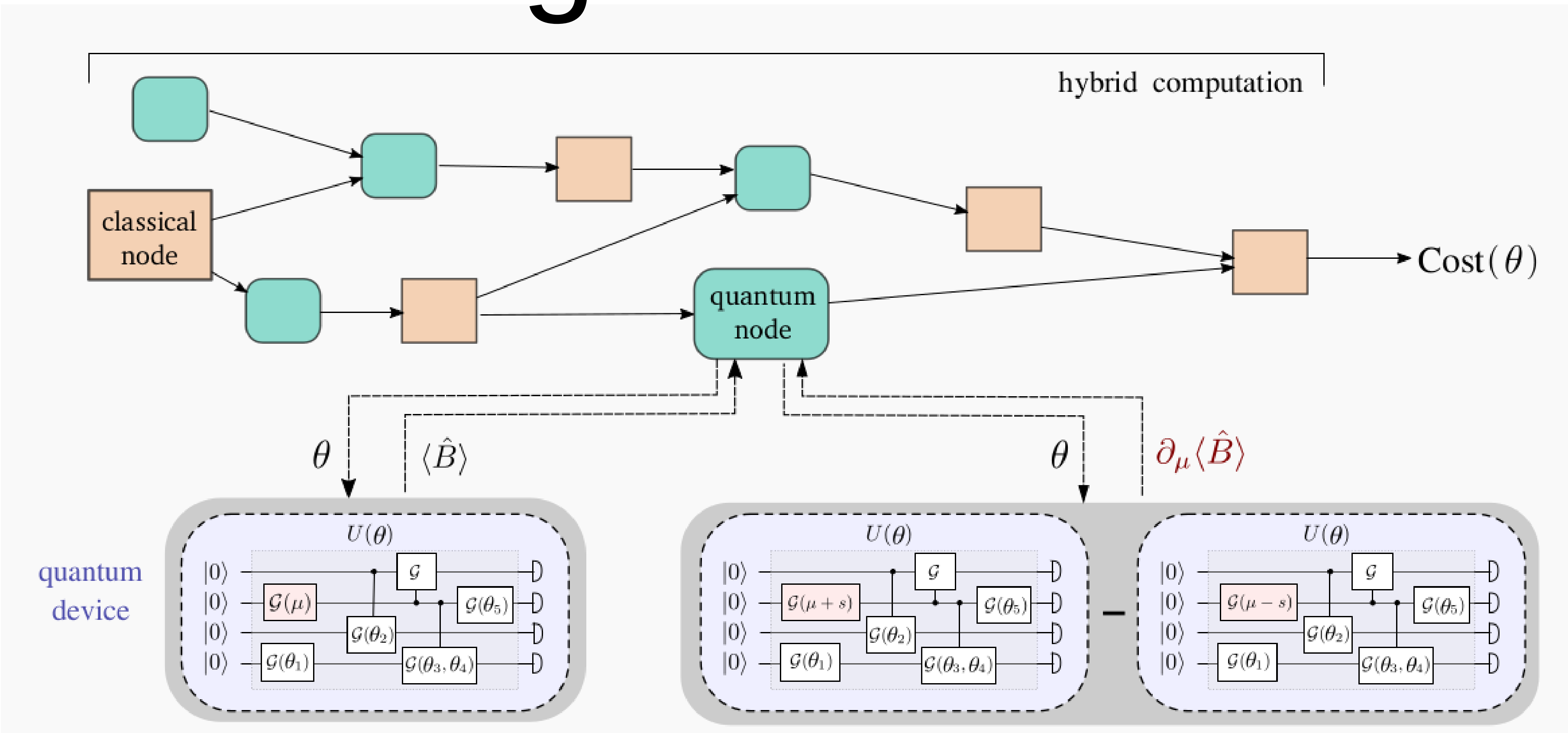
$$\frac{df}{dx} = \frac{sin(x+\pi/2) - sin(x-\pi/2)}{2}$$

- Cos can be evaluated **exactly** by two evaluations of sin

Global use of parameter shift rule

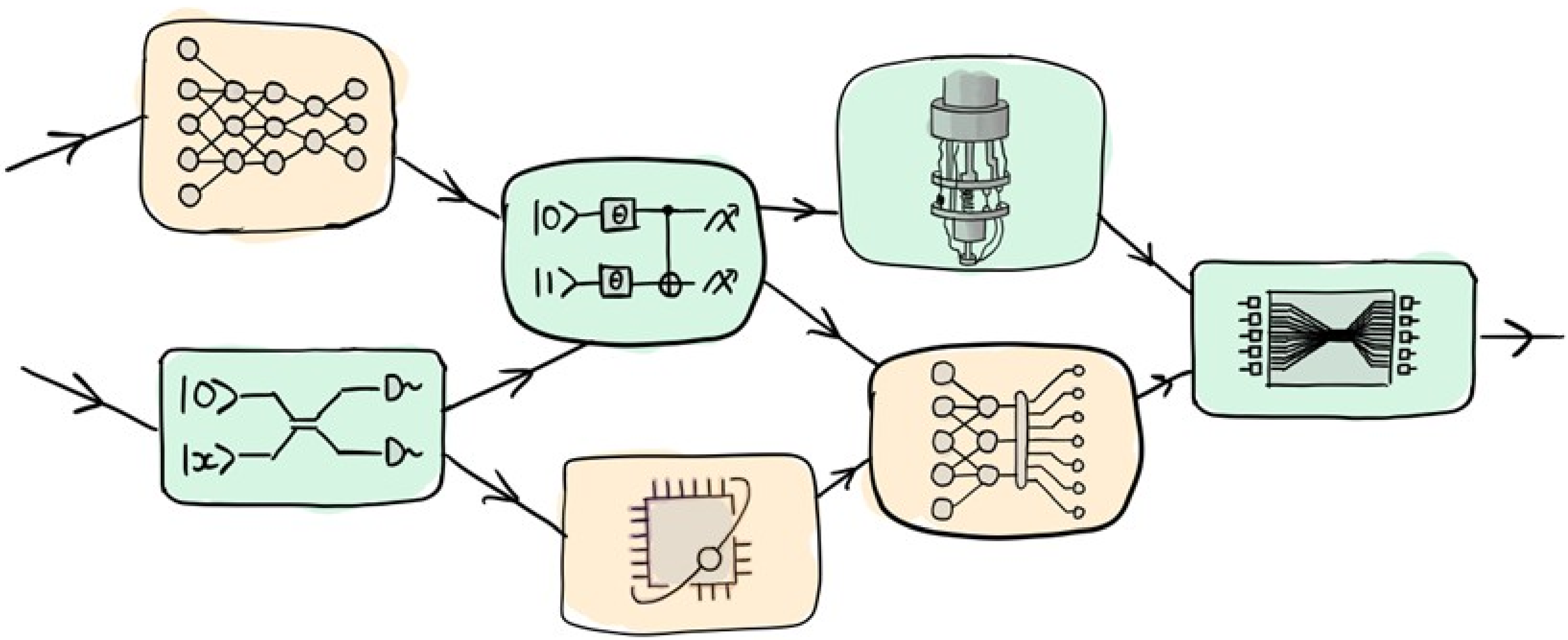
- Can be extended to any unitary operator
- G.E. Crooks, Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, 2019
- Implemented in Pennylane

Hybrid computation gradients



The shift-rule differentiation can be integrated in the derivation tree of classical machine learning (for example Pytorch) by chain rule

QML in action with



PENNYLANE

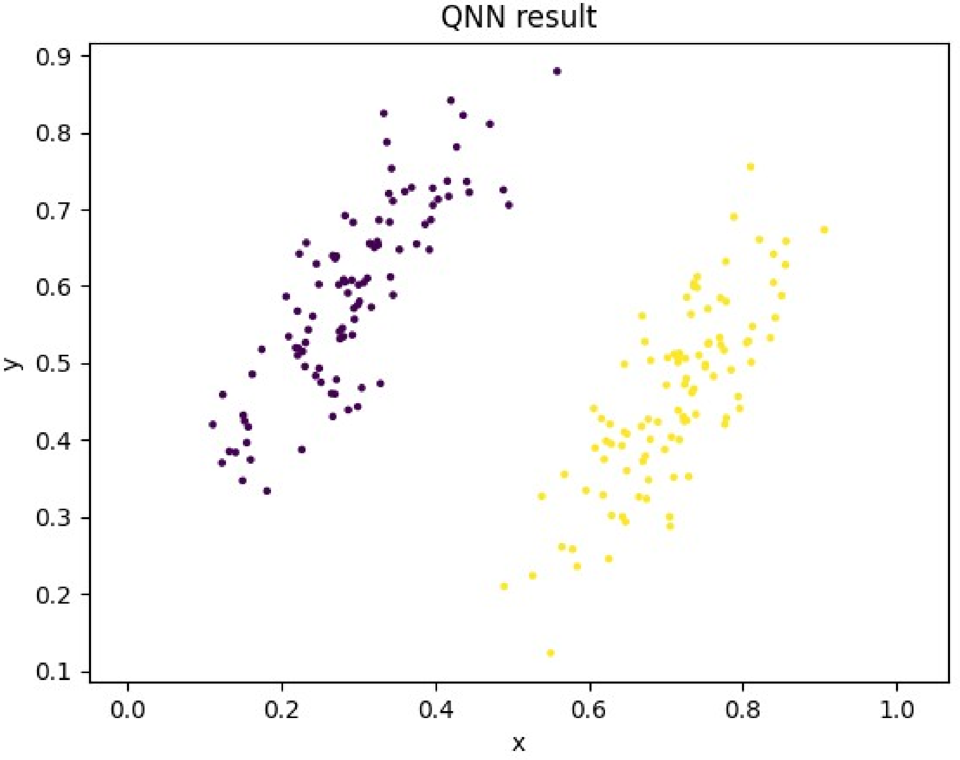
XANADU

Pennylane

- Pennylane is a python library implementing hybrid differentiable quantum computation
- Compatible with PyTorch
- Developed by XANADU
- A company from Toronto developping photonic hardware www.xanadu.ai
- Available on pip `#pip install pennylane`
- Open-source and well-documented

Test on synthetic data

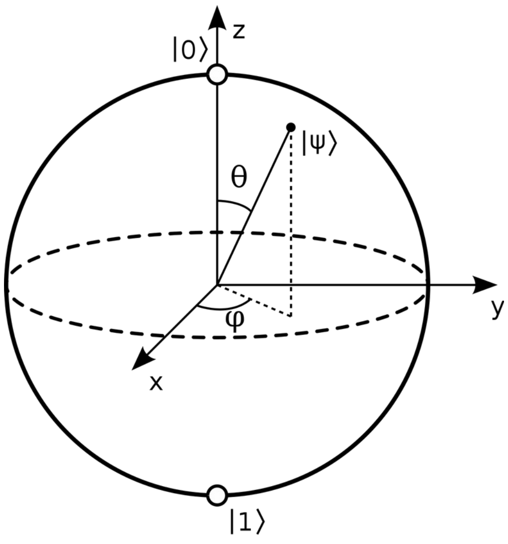
- multinomial distributions (100 points each)
- Classification from coordinates [0,1]x[0,1] to label {0,1}
- Data are linearly separable
- Classifiable by a linear model with 2 parameters



Re-uploading circuit

- Using re-uploading to solve the problem with 4 parameters
- Uploading two times x1 and x2 with RY operator
- Uploading the 4 parameters with RX operator

```
def circuit(params,x1,x2):
    qml.RX(params[0],wires=0)
    qml.RY(x1,wires=0)
    qml.RX(params[1],wires=0)
    qml.RY(x2,wires=0)
    qml.RX(params[2],wires=0)
    qml.RY(x1,wires=0)
    qml.RX(params[3],wires=0)
    qml.RY(x2,wires=0)
    return qml.expval(qml.PauliZ(0))
```



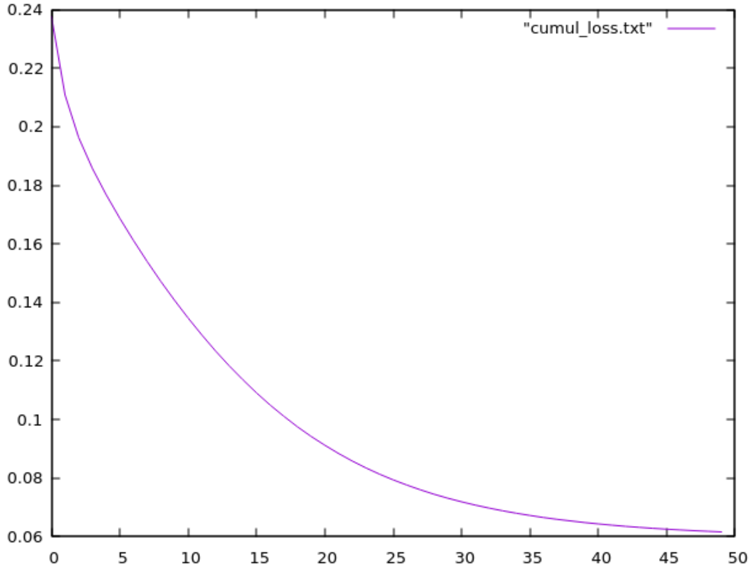
Model form

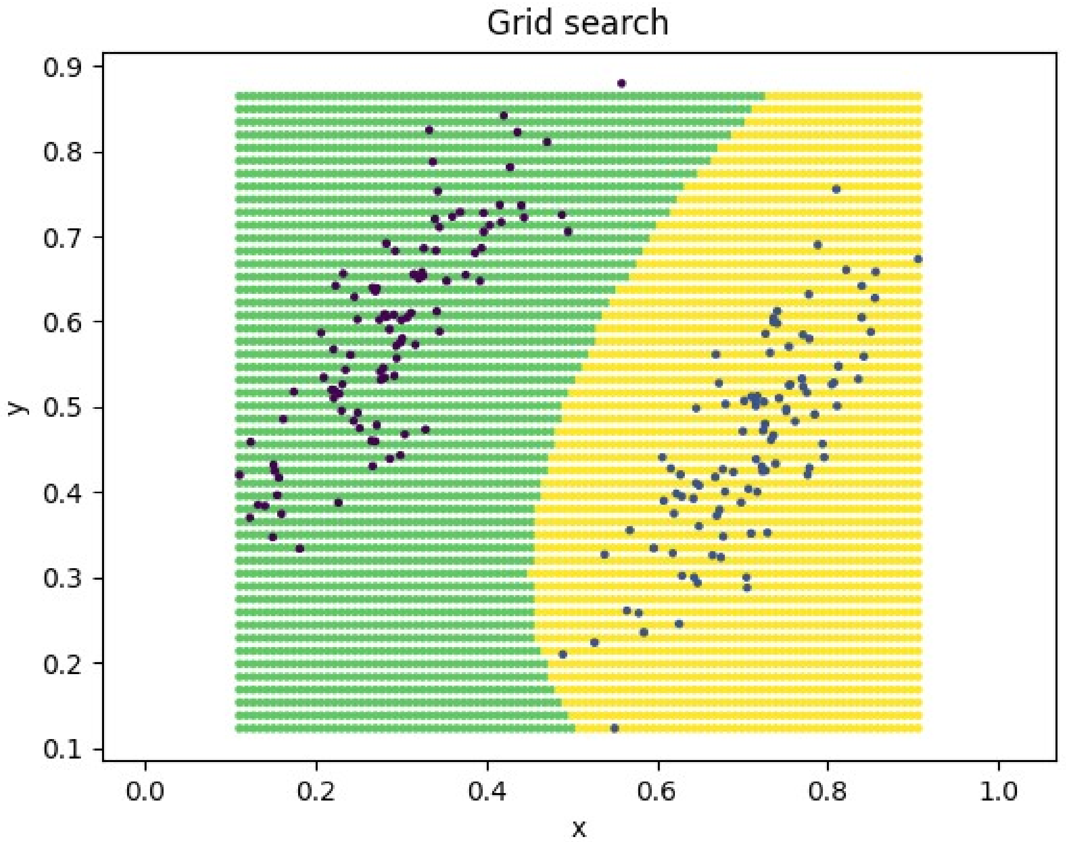
$$M(x_1, x_2, p_1, p_2) =< 0|X_{p1}^{\dagger} Y_{x1}^{\dagger} X_{p2}^{\dagger} Y_{x2}^{\dagger} Z Y_{x2} X_{p2} Y_{x1} X_{p1}|0 >$$

$$\begin{gathered} M(x_1, x_2, p_1, p_2) = \\ \cos(x_1)\cos(p_1) \\ -\sin(x_1)\sin(x_2)\cos(p_1) \\ -\sin(p_1)\sin(p_2)\cos(x_2) \\ +\cos(x_1)\cos(x_2)\cos(p_1)\cos(p_2) \end{gathered}$$

- The number of terms grows exponentially with the number of operators
- Only 2 operators here because the 4 operator expression does not fit in the slide !
- With re-uploading $\cos^n$ and $\sin^n$ appears providing non-linearity

Result of learning

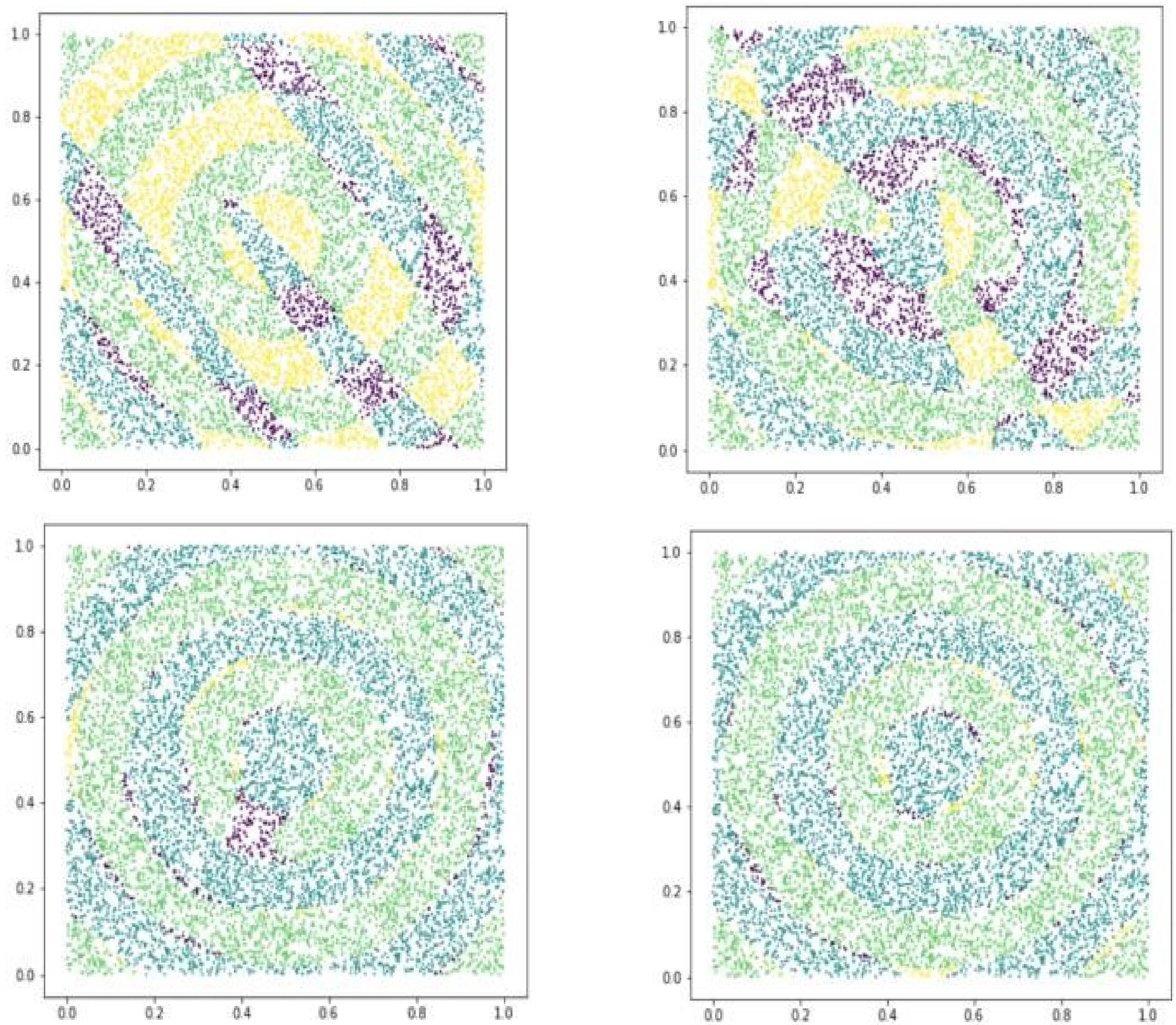




- Very low MSE reached
- The circuit has learned a non linear curve
- Early stopping at 50 epochs

More complicated example

- Concentric circles
- Solvable with 40 parameters and re-uploading



Credit Andrea Sartirana

Running on IBM-Q

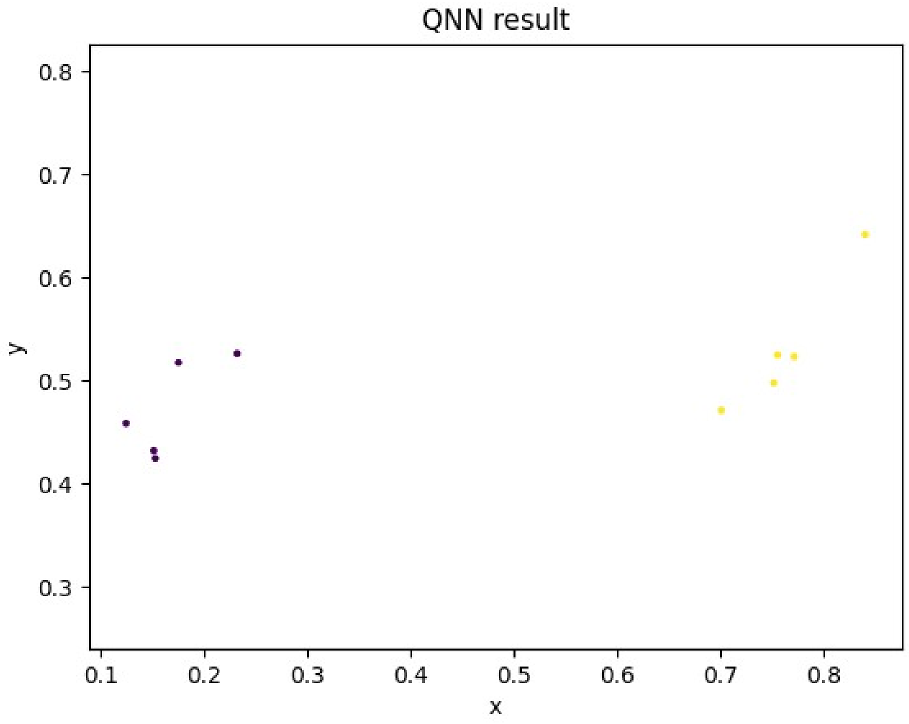
- Some quantum computers are free of use on IBM-Q (1 to 5 qubits)
- An account is required

  https://quantum-computing.ibm.com/

- Obtain the API token → .qiskitrc file
- Using Armonk : mono-qubit free QC

Results on IBM-Q

- No classification error on the 10 tests
- Very slow : 3 minutes for 10 tests (no derivatives) on a high disponibility phase
- Error Management
  - Gate precision error : irreducible with mono-qubit
  - Systematic error : should be handled by training the system directly on armonk but very VERY long training time → untractable now



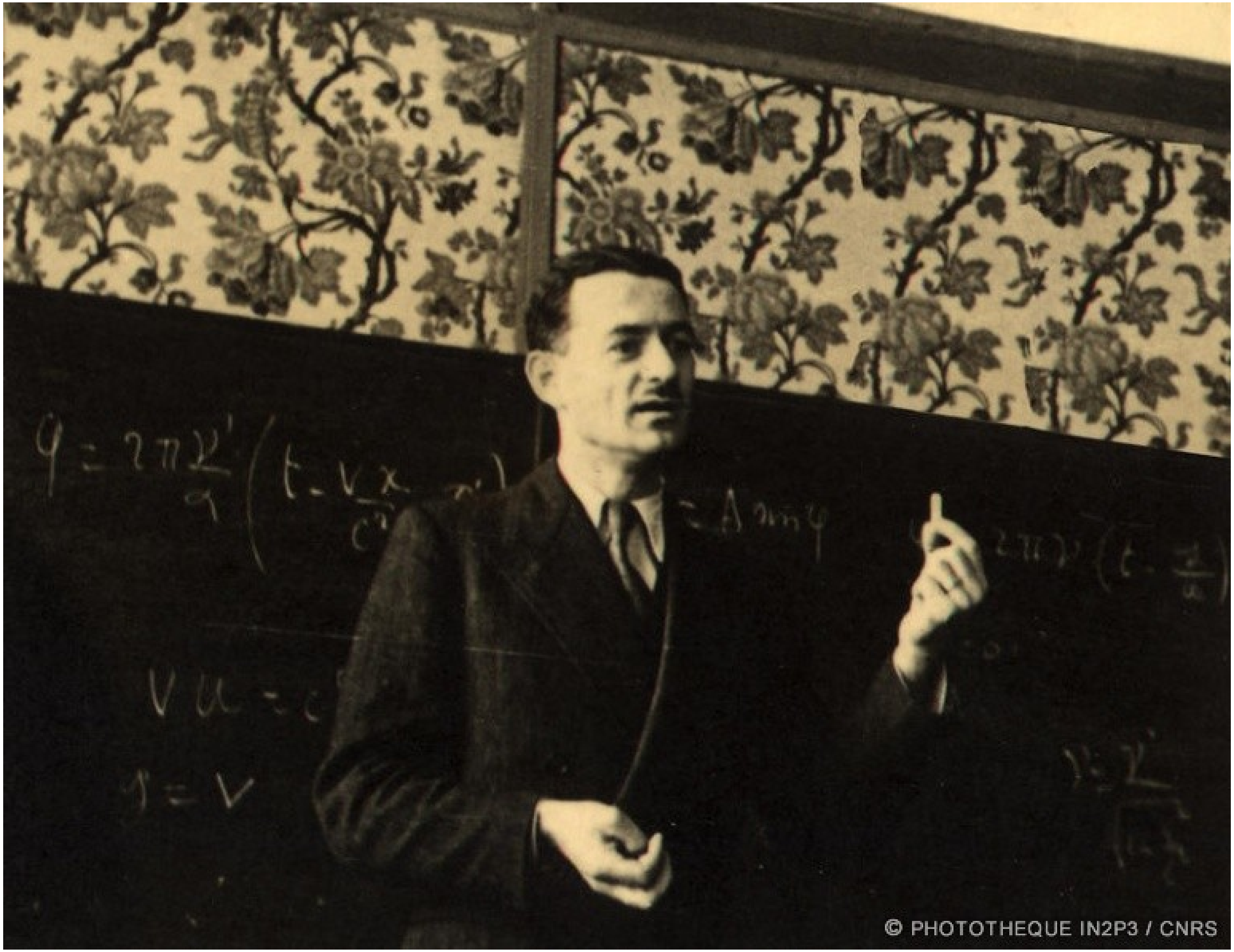
No classification error

On L2 distance

Result on simul= 0.176
Result on armonk= 0.113
Error : 35 %

Result on simul= 0.736
Result on armonk= 0.586
Error : 20 %

QML @ Laboratoire Leprince-Ringuet



QC2I IN2P3 Master Project

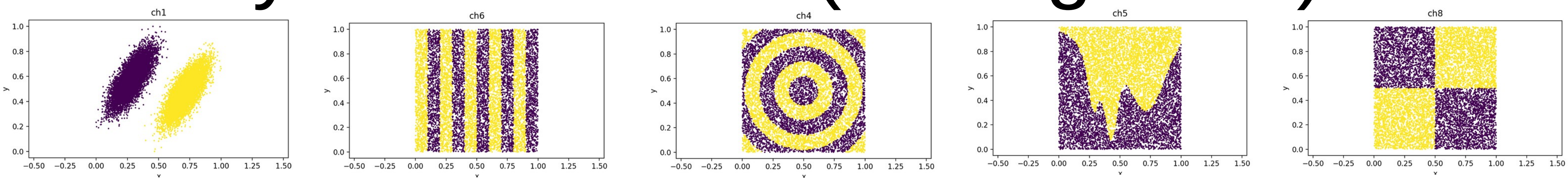
- Computing project supported by IN2P3
- Goal: explore the possible applications of quantum computing for HEP
- Scientific Resp. Denis Lacroix (IJCLab)
- Technical Resp. Bogdan Vulpescu (LPC)
- 3 themes
  - Simulation of complex quantum system (Denis Lacroix)
  - Prepare the Quantum Computing Revolution (Bogdan Vulpescu)
  - Quantum Machine Learning (Frédéric Magniette)
- Access to Cloud quantum computers (AWS & IBMQ)
- Website https://qc.pages.in2p3.fr/web/

QML @ Laboratoire Leprince-Ringuet

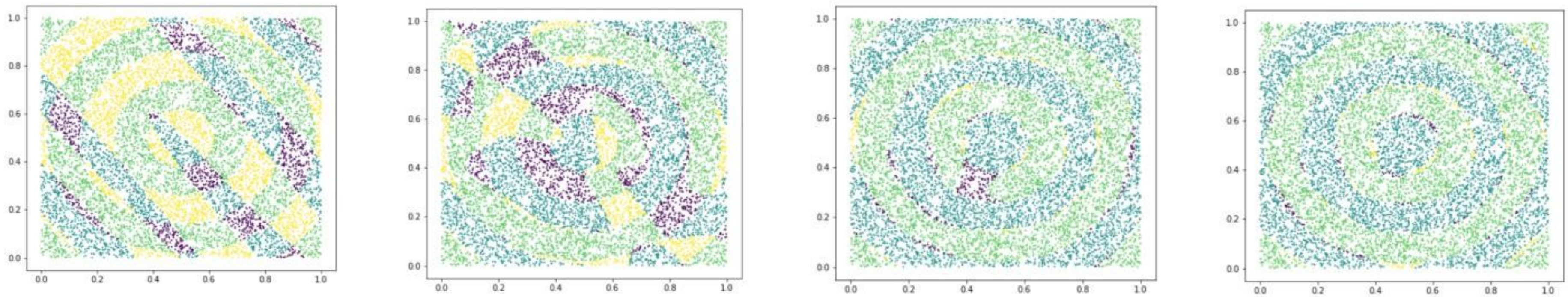
- LLR is an active member of QC2I
- 6 members of QC2I @ LLR
- F. Magniette head of the QML thematics (previously A. Sartirana)
- Interests in QML
  - QNN classifiers
  - Re-uploading techniques
  - Classical / quantum ML model convergence

QML @ Laboratoire Leprince-Ringuet

- Definition of benchmarks 2 coords → binary classification (F. Magniette)



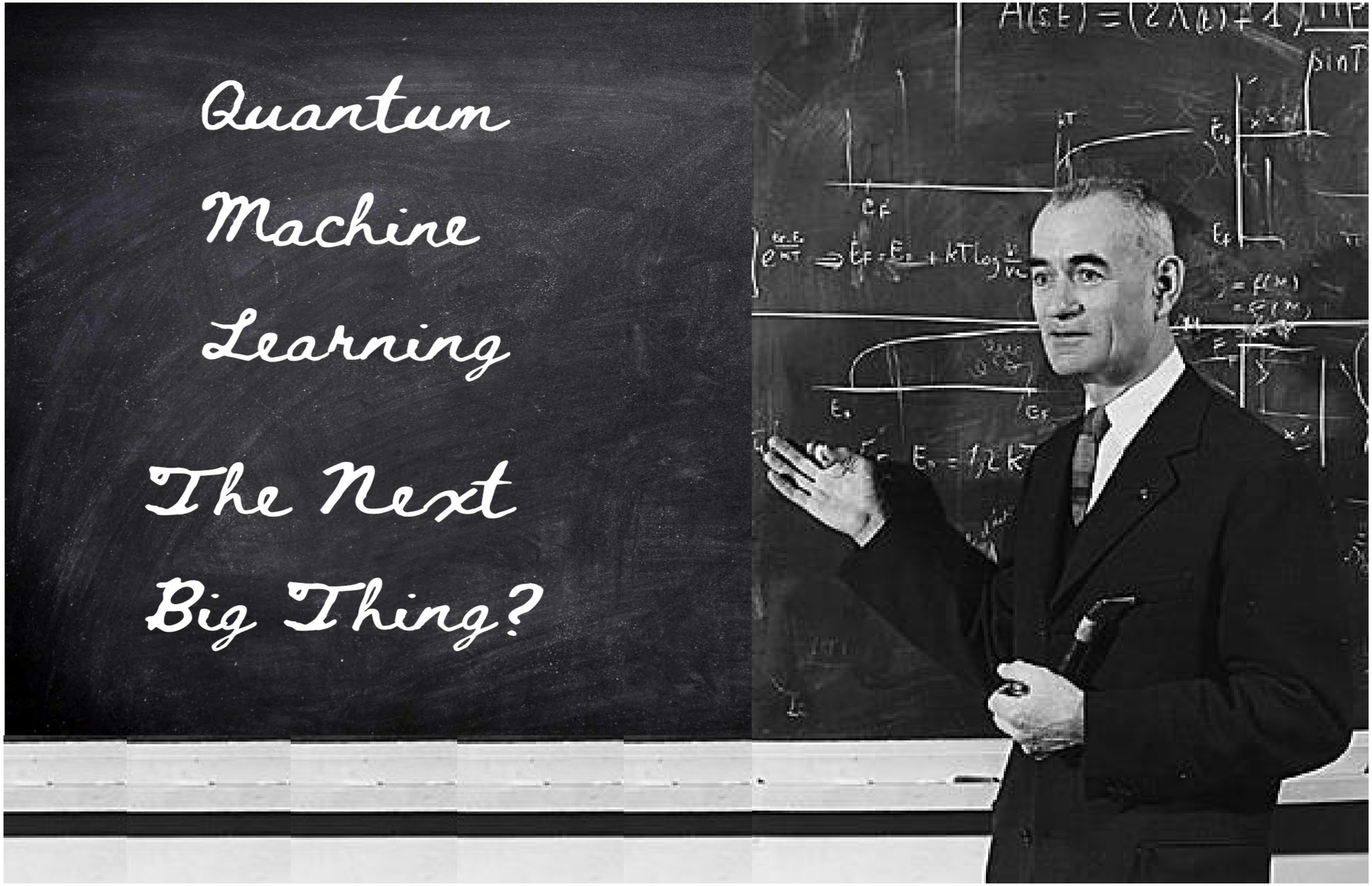
- Simulation of re-uploading learning circuits on benchmarks & particle physics data (A. Sartirana, F. Magniette)



QML @ Laboratoire Leprince-Ringuet

- P2IO project TutoQML in collaboration with Denis Lacroix (IJCLab)
  - 2 year post-doc Yann Beaujeault-Taudiere (since 1st December 2021)
  - Methodological study of QML models expressivity on synthetic and real data
  - Theoretical work on QC/DNN models identification

Any question ?



1959, Louis Leprince-Ringuet talking about QML at College de France...