bloch

September 16, 2022

0.1 Bloch representation and some comparison with binary computers

Bloch representation The idea behind the representation of a Qubit is that a Qubit has a state

- $|0\rangle$ if its amplitude lies at the top of the sphere
- $|1\rangle$ if its amplitude lies at the bottom of the sphere

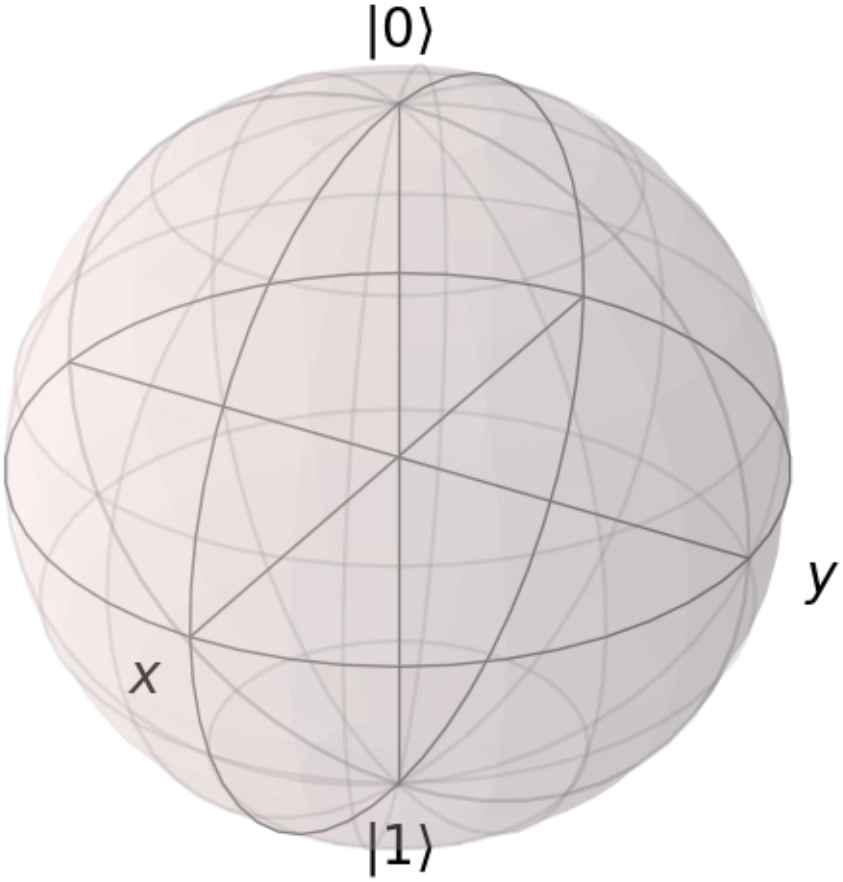
Quantum mechanics is mathematically formulated in Hilbert space or projective Hilbert space.

The pure states of a quantum system correspond to the one-dimensional subspaces of the corresponding Hilbert space (and the "points" of the projective Hilbert space).

In $\mathbb{R}^3$ we have the Bloch sphere which can be mapped to the Riemann sphere. (source : https://en.wikipedia.org/wiki/Bloch_sphere)

[1]:
```
import numpy as np
import pylab
import qutip

# qubit modelisation and representation
bloch_sphere = qutip.Bloch()
bloch_sphere.show()
```



Amplitudes and vectors A Qubit $|\psi\rangle$ is often represented by the following formula:

$|\psi\rangle = a|0\rangle + b|1\rangle$

or

$|\psi\rangle = a|\uparrow\rangle + b|\downarrow\rangle$

Here (a, b) are Qubit-coordinates on the Bloch sphere according to the states $|0\rangle$ and $|1\rangle$

Bloch sphere is a unit 2-sphere of $\mathbb{R}^3$ then we have $\langle\psi|\psi\rangle = \|\psi\|^2 = 1$

Take a vector $\vec{a}$ of the Bloch sphere. For example $\vec{a} = \left(\frac{1}{\sqrt(3)}, \frac{1}{\sqrt(3)}, \frac{1}{\sqrt(3)}\right)$

Here is a way to represent it on Bloch sphere (source : https://qutip.org/docs/latest/guide/guide-bloch.html?highlight=basis)

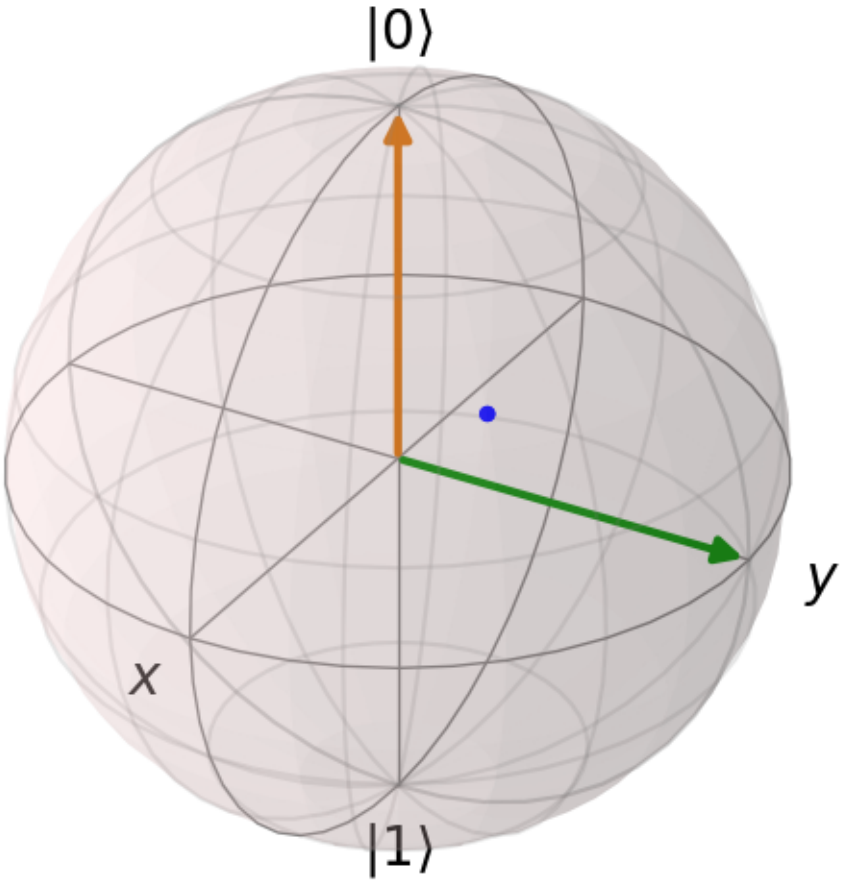
[2]:
```
# adding the vector a on the Bloch sphere
vector_a = [1./np.sqrt(3), 1./np.sqrt(3), 1./np.sqrt(3)]
```

```
bloch_sphere.add_points(vector_a)

# adding a vector y
vec = [0, 1, 0]
bloch_sphere.add_vectors(vec)

# adding the up-state
up = qutip.basis(2, 0)
bloch_sphere.add_states(up)

bloch_sphere.render()
bloch_sphere.show()
```



What is important here : a Qubit is a superposition of states.

=> The Qubit holds both states even in our example it seems closer to the $|0\rangle$

=> It is NOT a binary state like 0 or 1

The main consequence of this states superposition is the following fact :

$=>$ x Qubits can represent ALL values between 0 & 2^x at a time

While

$=>$ x bits can represent ONE value between 0 & 2^x at a time

Scaling "Quantum computing is a new generation of technology that involves a type of computer 158 million times faster than the most sophisticated supercomputer we have in the world today. It is a device so powerful that it could do in four minutes what it would take a traditional supercomputer 10,000 years to accomplish" (Livescience)

Here is a table giving us a comparison between classical and quantum bits at different scale. (source : https://vincentlauzon.com/2018/03/21/quantum-computing-how-does-it-scale/)

# of qubits	# bits / # loops	RAM
1	2	2 bits
2	4	4 bits
3	8	1 byte
4	16	2 bytes
5	32	4 bytes
6	64	8 bytes
7	128	16 bytes
8	256	32 bytes
9	512	64 bytes
10	1024	128 bytes
11	2048	256 bytes
12	4096	512 bytes
13	8192	1 kB

First simulation There are several python tools to develop quantum algorithms. An interesting one is SimulaQron, a Quantum Internet simulator (source : https://softwarequtech.github.io/SimulaQron/html/index.html)

And we will find a first classic test with entangled particles :

A first test from their documentation)))

Alice and Bob generates one EPR pair, that is, two maximally entangled Qubits A and B of the form

$|\Psi\rangle_{AB} = \frac{1}{\sqrt{2}} (|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B)$

Alice and Bob are informed of the identifiers of the qubits and are informed that entanglement was generated. Both Alice and Bob measure their respective qubits to obtain a classical random number $x \in \{0, 1\}$

"While the task we want to realize here is completely trivial, the addition of step 3 does however already highlight a range of choices on how to realize step 3 and the need to find good abstractions to allow easy application development. One way to realize step 3 would be to hardwire Alices and Bobs measurements: if the hardware can identify the correct qubits from the entanglement generation, then we could instruct it to measure it immediately without asking for a notification from the entanglement generation process. It is clear that in a network that is a bit larger than our tiny three node setup, identifying the right setup requires a link between the underlying qubits and classical control information: this is the objective of the classical/quantum combiner"

Follow the link : https://softwarequtech.github.io/SimulaQron/html/GettingStarted.html#testing-a-simple-example