Quantum state tomography

Bruno Fédrici

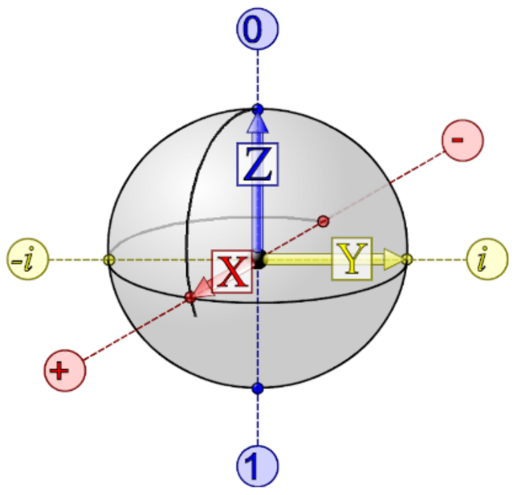
March 29, 2024

In this problem we would like you to write a generic Qiskit program that returns the Bloch vector coordinates x, y and z of an arbitrary single qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ given as an input:

$$\mathbb{C}^2 \longrightarrow \mathbb{R}^3$$

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} \longrightarrow \begin{bmatrix}x\\ y\\ z\end{bmatrix}$$

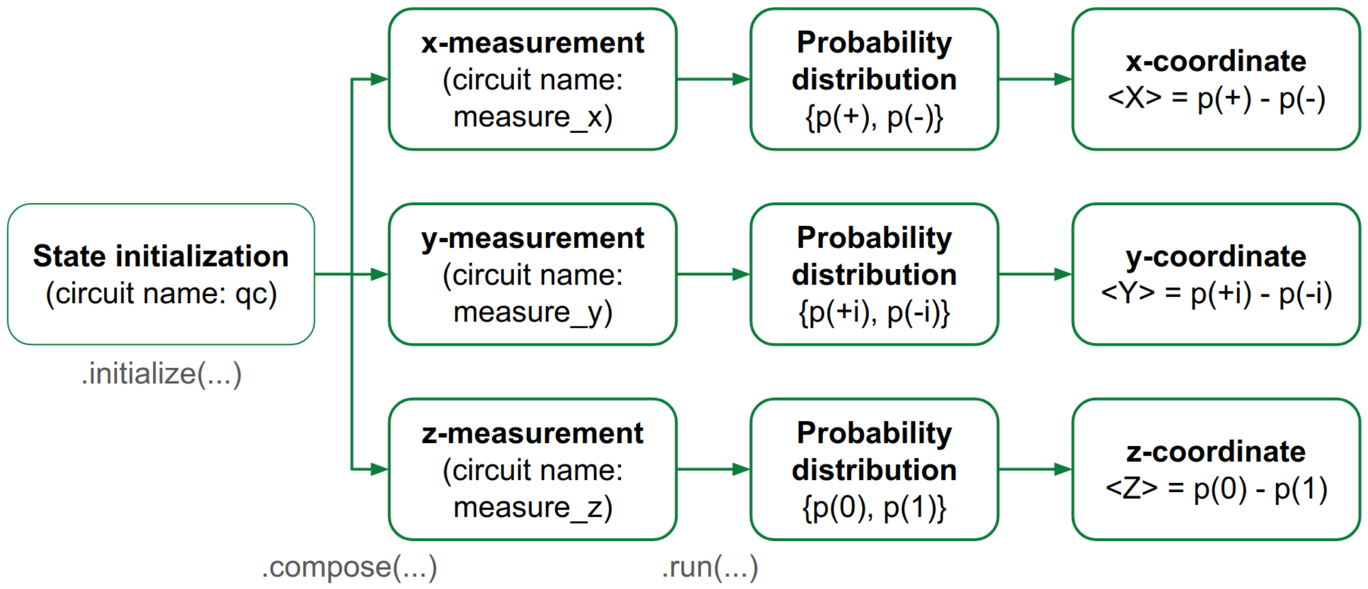
The technique known as quantum state tomography consists in calculating the expectation value of Pauli operators X, Y, and Z to reconstruct the Bloch vector:



$$\longrightarrow \begin{bmatrix}x\\ y\\ z\end{bmatrix} = \begin{bmatrix}\langle X\rangle & = & p(+) - p(-)\\ \langle Y\rangle & = & p(+i) - p(-i)\\ \langle Z\rangle & = & p(0) - p(1)\end{bmatrix}$$

To implement this technique, you will first initialize a single qubit in a given state $|\psi\rangle$, and then simulate, for this qubit, series of projective measurements along x, y and z axes. The different empirical probability distributions - to be constructed from the measurement results - can finally be used to compute respective expectation values.

More precisely, your program will be structured as follow:



When it comes to initializing a single qubit in a given state $|\psi\rangle$, you might find useful the `.initialize` method of the `QuantumCircuit` class, a black box operation for quantum state initialization. For instance, suppose you want to initialize the first two qubits of a quantum circuit `qc` in state $(|00\rangle + |10\rangle + |11\rangle)/\sqrt{3}$, all you have to do is to execute:

```
qc.initialize([sqrt(1/3), sqrt(1/3), 0, sqrt(1/3)], [0,1])
```

Where we have first specified the components of the target state vector, and then the qubit indices.

To simulate measurements along the different axes of the Bloch sphere, remember we have the following relationships in between operators: $X = HZH$ and $Y = SXS^\dagger$.

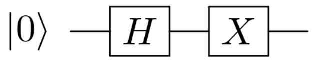
Also note that in the proposed scheme, different circuits are requested for initialization and measurement steps. Those circuits can ultimately be "plugged" together using the `.compose` method of the `QuantumCircuit` class so as to build a single object you can later simulate. As an example use of `.compose`, running the following command lines:

```
qc1 = QuantumCircuit(1)
qc1.h(0)

qc2 = QuantumCircuit(1)
qc2.x(0)

qc3 = qc1.compose(qc2)
qc3.draw()
```

will output the following circuit:



After having stored the coordinates of the Bloch vector, as returned by your program, in a list object `bloch_vector`, it is also possible to draw this vector in a Bloch sphere using:

```
from qiskit.visualization import plot_bloch_vector
plot_bloch_vector(bloch_vector)
```

As a way to check if your implementation is successful, for input state $|\psi\rangle = \sqrt{\frac{2}{3}}\,|0\rangle + \sqrt{\frac{1}{3}}\,|1\rangle$, your program should return empirical results very close to quantum theory predictions, i.e.: $\langle X\rangle = \frac{2\sqrt{2}}{3}$, $\langle Y\rangle = 0$, and $\langle Z\rangle = \frac{1}{3}$.