Single Systems – Lesson 01

Deval Deliwala

December 17, 2023

Contents

1 Classical Information **2**
- 1.1 Classical States & Probability Vectors . . . 2
- 1.2 Measuring Probabilistic States . . . 2
- 1.3 Standard Basis Vectors . . . 3
- 1.4 Operations . . . 3
- 1.5 Probabilistic Operations & Stochastic Matrices . . . 5
- 1.6 Compositions of Probabilistic Operations . . . 5

2 Quantum Information **5**
- 2.1 Quantum State Vectors . . . 6
- 2.2 Measuring Quantum States . . . 8
- 2.3 Unitary Operations . . . 8
- 2.4 Important examples of Unitary Operations on Qubits . . . 9
- 2.5 Compositions of Qubit Unitary Operations . . . 10

3 Qiskit Examples **11**

Notation

– X refers to the system being considered
– Σ refers to the set of classical states of X

Here are a few examples:

- if X is a bit, $\Sigma = 0, 1$ – the *binary alphabet*
- if X is a six-sided die, $\Sigma = 1, 2, 3, 4, 5, 6$
- if X is an electric fan switch, $\Sigma =$ high, medium, low, off

1 Classical Information

1.1 Classical States & Probability Vectors

In Quantum Computing (QC) our knowledge of X is uncertain. We thus represent our knowledge of the classical state of X by assigning *probabilities* to each classical state resulting in a *probabilistic state.*

For example, suppose X is a bit. In this case, based on our past experience or what we know about X, there is a $3/4$ chance its classical state is 0 and a $1/4$ chance it's 1. Therefore,

$$\Pr(X = 0) = \frac{3}{4} \qquad \Pr(X = 1) = \frac{1}{4}$$

We can represent this more succinctly with a column vector:

$$\begin{pmatrix} \frac{3}{4} \\ \frac{1}{4} \end{pmatrix}$$

The probability of the bit being 0 is placed at the top; the probability of the bit being 1 is placed at the bottom. We can represent any probabilistic state via a column vector satisfying two properties:

Probabilistic Vector Requirements

1. All entries of the vector are *nonnegative numbers*
2. The sum of the entries is equal to 1

1.2 Measuring Probabilistic States

Intuitively, we can never "see" a system in a probabilistic state; a measurement always yields exactly one of the allowed states.

Measuring changes our knowledge of the system, and therefore changes the probabilistic state we associate with the system. If we recognize that X is in the classical state $a \in \Sigma$, then the new probability vector representing our knowledge of X becomes a vector having 1 in the entry corresponding to a and 0 for all other entries.

$$\begin{pmatrix} 0.3 \\ 0.1 \\ 3.14 \\ 2.72 \\ \vdots \end{pmatrix} \quad \rightarrow \quad \boxed{\text{Measurement}} \quad \rightarrow \quad \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}$$

1.3 Standard Basis Vectors

We can define any probabilistic state vector as a *linear combination* of standard basis vectors. For example, assuming the system we have in mind is a bit, the standard basis vectors are given by

Computational Basis States

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

For example, we have

$$\begin{pmatrix} \frac{3}{4} \\ \frac{1}{4} \end{pmatrix} = \frac{3}{4}|0\rangle + \frac{1}{4}|1\rangle$$

1.4 Operations

Deterministic Operations

Deterministic Operations transform each classical state $a \in \Sigma$ into $f(a)$ for some function f of the form $f : \Sigma \rightarrow \Sigma$.

For example, if $\Sigma = 0, 1$, there are four functions of the form f_1, f_2, f_3, f_4 which can be represented as follows:

$$\begin{aligned} f_1(0) &= 0 \quad f_1(1) = 0 \\ f_2(0) &= 0 \quad f_2(1) = 1 \\ f_3(0) &= 1 \quad f_3(1) = 0 \\ f_4(0) &= 1 \quad f_4(1) = 1 \end{aligned}$$

The first and last of these functions are *constant*, where the output remains constant regardless of input. The middle two are *balanced* where the two possible output values occur the same # of times. The function f_2 is the **identity function** where $f(a) = a$ for $a \in \Sigma$. The function f_3 is the NOT function where each input is flipped for an output.

Every deterministic operation on probabilistic states can be represented as a matrix, where

$$M|a\rangle = |f(a)\rangle$$

for every $a \in \Sigma$. Such a matrix always exists and is unique. For the above constant and balanced functions:

$$M_1 = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}, \quad M_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad M_3 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad M_4 = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}$$

Deterministic matrix operations always have exactly one 1 in each column, and 0 for all other entries.

Let us denote $\langle a|$ the *row* vector having a 1 in the entry corresponding to a and 0 for all other entries, for each $a \in \Sigma$.

For example if $\Sigma = 0, 1$,

$$\langle 0| = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad \text{and} \quad \langle 1| = \begin{pmatrix} 0 & 1 \end{pmatrix}$$

If we perform matrix multiplication on a column vector defined by $|b\rangle$ and a row vector defined by $\langle a|$, we obtain a square matrix having a 1 in the entry corresponding to the (b, a) location in the matrix and 0 everywhere else. For example,

$$|0\rangle\langle 1| = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Using this notation, we can express M corresponding to the function f as

$$M = \sum_{a \in \Sigma} |f(a)\rangle\langle a|$$

If we switch the order of multiplication – $\langle a||b\rangle$, we obtain a 1 x 1 scalar. For the sake of tidiness we write the product as $\langle a|b\rangle$. We will later define $\langle a|b\rangle$ as the *inner product* of a and b.

$$\langle a|b\rangle = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases}$$

1.5 Probabilistic Operations & Stochastic Matrices

In addition to deterministic operations, we have *probabilistic operations.*
For example, consider an operation on a bit where, if the classical state of the bit is 0, it is left alone, and if the classical state of the bit is 1, it is flipped to 0 with probability $\frac{1}{2}$. This operation is represented by the matrix

$$\begin{pmatrix} 1 & \frac{1}{2} \\ 0 & \frac{1}{2} \end{pmatrix}$$

We can check this matrix does the correct thing by multiplying with the standard basis vectors:

$$\begin{pmatrix} 1 & \frac{1}{2} \\ 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & \frac{1}{2} \\ 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$$

All probabilistic operations are defined by *stochastic* matrices, which are matrices satisfying the following properties:

Stochastic Matrices

1. All entries are nonnegative real numbers
2. The entries in every column sum to 1

Equivalently, stochastic matrices are matrices whose columns all form probability vectors

1.6 Compositions of Probabilistic Operations

Suppose X is a system having classical state Σ, and $M_1, \cdots, M_n$ are stochastic matrices representing probabilistic operations on X.

If we apply M_1 to the probabilistic state represented by a probability vector u, the resulting probabilistic state is $M_1 u$. If we then apply a second operation M_2, we obtain the probability vector

$$M_2(M_1 u) = (M_2 M_1)u.$$

The order in which operations are applied in a composition can change the resulting operation.

2 Quantum Information

Now we move onto quantum information where we use a different type of vector to represent *quantum* states. In this section, we'll be considered with systems that are finite in length with nonempty sets of classical states. We will continue using *Dirac* notation.

2.1 Quantum State Vectors

A *quantum state* is represented the same way as probabilistic states. Vectors representing quantum states are characterized by these following properties:

Properties of Quantum States

1. The entries of a quantum state vector are *complex #s*
2. The sum of the *absolute values squared* of the entries of a quantum state vector is 1.

Any speedup from a quantum computer, or improvement in communication protocol, ultimately derives from these simple mathematical changes.

The *Euclidean norm* of a column vector

$$v = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix}$$

is defined as follows:

$$||v|| = \sqrt{\sum_{k=1}^{n} |\alpha_k|^2}$$

Thus, quantum state vectors are *unit vectors* with respect to the Euclidean norm.

Examples of Qubit States

The term *qubit* refers to a quantum system whose classical state set is 0, 1 – the same as a bit, but a quantum state.

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

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

and

$$\begin{pmatrix} \frac{1+2i}{3} \\ -\frac{2}{3} \end{pmatrix} = \frac{1+2i}{3}|0\rangle - \frac{2}{3}|1\rangle$$

Computing the sum of the absolute values squared of the entries respectively yields

$$|1|^2 + |0|^2 = 1 \quad \text{and} \quad |0|^2 + |1|^2 = 1,$$

$$\left|\frac{1}{\sqrt{2}}\right|^2 + \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2} + \frac{1}{2} = 1$$

and

$$\left|\frac{1+2i}{3}\right|^2 + \left|-\frac{2}{3}\right|^2 = \frac{5}{9} + \frac{4}{9} = 1$$

These are therefore all valid quantum state vectors and are all linear combinations, *superpositions* of the 0 and 1 states.

The second example of a qubit state vector is commonly encountered and is called the *plus* state:

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

We also use the notation

$$|-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

for the *minus* state.

It is common to use the notation $|\psi\rangle$ to denote an arbitrary vector that may not necessarily be a standard basis vector.

If we have a vector $|\psi\rangle$ whose indices correspond to some classical state set Σ, and if $a \in \Sigma$ is an element of this classical state set, then the (matrix) product $\langle a|\psi\rangle$ is equal to the entry of the vector $|\psi\rangle$ whose index corresponds to a.

For example, if $\Sigma = 0, 1$ and

$$|\psi\rangle = \frac{1+2i}{3}|0\rangle - \frac{2}{3}|1\rangle = \begin{pmatrix} \frac{1+2i}{3} \\ -\frac{2}{3} \end{pmatrix}$$

then

$$\langle 0|\psi\rangle = \frac{1+2i}{3} \quad \text{and} \quad \langle 1|\psi\rangle = -\frac{2}{3}$$

Row vector notation, $\langle\psi|$ refers to the row vector obtained by taking the *conjugate transpose* of the column vector $|\psi\rangle$.

The same $|\psi\rangle$ defined above would define $\langle\psi|$

$$\langle\psi| = \frac{1-2i}{3}\langle 0| - \frac{2}{3}\langle 1| = \begin{pmatrix} \frac{1-2i}{3} & -\frac{2}{3} \end{pmatrix}$$

The reason we take the complex conjugate in addition to the transpose will become clear when we discuss the *inner product*.

2.2 Measuring Quantum States

The rule is simple: If a quantum state is measured, each classical state of the system results with probability equal to the *absolute value squared* of the entry in the quantum state vector corresponding to that classical state. This is known as the *Born rule* in quantum mechanics.

For example, measuring the plus state

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

results in two outcomes, 0 and 1, with probabilities

$$\Pr(\text{outcome is } 0) = |\langle 0|+\rangle|^2 = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$$

$$\Pr(\text{outcome is } 1) = |\langle 1|+\rangle|^2 = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$$

2.3 Unitary Operations

The reason quantum information is fundamentally different from classical information is because the set of allowable *operations* that can be performed on a quantum state is different than it is for classical information. All operations on quantum states are defined with *Unitary Matrices*. A square matrix U having complex # entries is *unitary* if it satisfies the equations

Unitary Matrices

$$UU^\dagger = I$$
$$U^\dagger U = I$$

Here, I is the identity matrix, and $U^\dagger$ is the *conjugate transpose* of U. If M is not a square matrix it is possible for $M^\dagger M = I$ but not $MM^\dagger = I$. This equivalence is only true for square matrices.

The condition that U is unitary is equivalent to the condition that multiplication by U does not change the euclidean norm of any vector.

In this sense, Unitary matrices are the quantum analogous of the classical Stochastic matrices.

2.4 Important examples of Unitary Operations on Qubits

Pauli Operations

The four Pauli matrices are as follows:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

A common notation is that $X = \sigma_x$, $Y = \sigma_y$, and $Z = \sigma_z$. The X operation is also called the *bit flip* or a NOT operation because it induces this action on bits:

$$X|0\rangle = |1\rangle \quad \text{and} \quad X|1\rangle = |0\rangle$$

The Z operation is also called a *phase flip* because it has this action:

$$Z|0\rangle = |0\rangle \quad \text{and} \quad Z|1\rangle = -|1\rangle$$

Hadamard Operation

The Hadamard Operation is described by this matrix:

$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

Applying the Hadamard Gates on well-known quantum states yields

$$\begin{aligned} H|0\rangle &= |+\rangle \\ H|1\rangle &= |-\rangle \\ H|+\rangle &= |0\rangle \\ H|-\rangle &= |1\rangle \end{aligned}$$

It is worth pausing to consider the fact that $H|+\rangle = |0\rangle$ and $H|-\rangle = |1\rangle$. Both $|+\rangle$ and $|-\rangle$ produce 0 and 1 with the same probabilities as each other – so one would think having both states is useless and we wouldn't know which of the states a qubit is in – $|+\rangle$ or $|-\rangle$. However, if we apply a Hadamard Operation and then measure we obtain 0 with certainty if the original state was $|+\rangle$

and 1 if the original state was $|-\rangle$.

Thus the quantum states $|+\rangle$ and $|-\rangle$ can be discriminated *perfectly*. This reveals that sign changes, or more generally changes to the *phases* of the complex # entries of a quantum state vector, an significantly change the state.

Here is another example, this time of the action of a T operation on a plus state:

$$T|+\rangle = T\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{\sqrt{2}}T|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1+i}{2}|1\rangle$$

Notice we did not bother to convert to the equivalent matrix/vector forms. We instead used the linearity of matrix multiplication together with the formulas

$$T|0\rangle = |0\rangle \quad \text{and} \quad T|1\rangle = \frac{1+i}{\sqrt{2}}|1\rangle$$

Phase Operations

A phase operation is one described by the matrix

$$P_\theta = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$$

for any choice of real number θ, that is, any complex # with a norm of 1. The operations

$$S = P_{\pi/2} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \quad \text{and} \quad T = P_{\pi/4} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1+i}{\sqrt{2}} \end{pmatrix}$$

are particularly important examples. Other examples include $I = P_0$ and $Z = P_\pi$

2.5 Compositions of Qubit Unitary Operations

If we first apply a Hadamard Operation, followed by the S operation, followed by another Hadamard Operation, the resulting operation R is as follows:

$$R = HSH = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{pmatrix}$$

This unitary operation R is an interesting example. By applying this operation twice, we obtain a NOT operation.

$$R^2 = \begin{pmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{pmatrix}^2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

3 Qiskit Examples

In python, matrix and vector computations can be performed using the `array` class from the `NumPy` library. Here is an example of a code cell that defines two vectors, `ket0` and `ket1` corresponding to their qubit state vectors $|0\rangle$ and $|1\rangle$, and displays their average.

```
from numpy import array

ket_0 = array([1, 0])
ket_1 = array([0, 1])

ket_0 / 2 + ket_1 / 2
```

Output:

```
array([0.5, 0.5])
```

We can also use `array` to create matrices that represent operations.

```
M_1 = array([[1, 1], [0, 0]])
M_2 = array([[1, 1], [1, 0]])

M_1 / 2 + M_2 / 2
```

Output:

```
array([[1. , 1. ],
       [0.5, 0. ]])
```

Matrix multiplication can be performed using `matmul` function from `NumPy` :

```
from numpy import matmul

display(matmul(M_1, ket_1))
display(matmul(M_1, M_2))
display(matmul(M_2, M_1))
```

Output:

```
array([1, 0])
array([[2, 1],
       [0, 0]])
array([[1, 1],
       [1, 1]])
```

single_systems

December 18, 2023

[2]:
```
# Vectors & Matrices in Python

from numpy import array

ket0 = array([1, 0])
ket1 = array([0, 1])

ket0 / 2 + ket1 / 2 # Taking average of ket0 and ket1
```

[2]:
```
array([0.5, 0.5])
```

[3]:
```
# Matrices & Operations

M1 = array([[1, 1], [0, 0]])
M2 = array([[1, 1], [1, 0]])

M1 / 2 + M2 / 2
```

[3]:
```
array([[1. , 1. ],
       [0.5, 0. ]])
```

[4]:
```
# Matrix Multiplication using matmul

from numpy import matmul

display(matmul(M1, ket1))
display(matmul(M1, M2))
display(matmul(M2, M1))
```

```
array([1, 0])
```

```
array([[2, 1],
       [0, 0]])
```

```
array([[1, 1],
       [1, 1]])
```

0.0.1 States, Measurements and Operations

Defining and Displaying State Vectors Qiskits `Statevector` class provides functionality for defining and manipulating quantum state vectors.

[5]:
```
from qiskit.quantum_info import Statevector
from numpy import sqrt

u = Statevector([1/sqrt(2), 1/sqrt(2)])
v = Statevector([(1 + 2.0j)/3, -2/3])
w = Statevector([1/3, 2/3])

print("State vectors u, v, and w have been defined")
```

```
State vectors u, v, and w have been defined
```

The `Statevector` class provides a `draw` method for displaying state vectors including `latex` and `text` options for different visualizations

[6]:
```
display(u.draw("latex"))
display(v.draw("text"))
```

$$\frac{\sqrt{2}}{2}|0\rangle + \frac{\sqrt{2}}{2}|1\rangle$$

```
[ 0.33333333+0.66666667j,-0.66666667+0.j        ]
```

The `Statevector` class also includes the `is_valid` method, which checks to see if a given vector is a valid quantum state vector (it has a euclidean norm = 1)

[7]:
```
display(u.is_valid())
display(w.is_valid())
```

```
True
```

```
False
```

Simulating measurements using the `measure` method from the `Statevector` class

[8]:
```
v = Statevector([(1+2.0j)/3, -2/3])
v.draw("latex")
```

[8]:

$$(\frac{1}{3} + \frac{2i}{3})|0\rangle - \frac{2}{3}|1\rangle$$

Running the `measure` method stimulates a standard basis measurement. It returns the result of that measurement, plus the new quantum state of our system after that measurement

[9]:
```
v.measure()
```

[9]:
```
('1',
 Statevector([ 0.+0.j, -1.+0.j],
```

dims=(2,)))

Measurements are probabilistic, so the same method can return different results. If the measurement yields 0 the quantum state vector after the measurements takes place to be

$$\frac{1+2i}{\sqrt{5}}|0\rangle$$

(rather than $|0\rangle$).

If the measurement yields 1, the quantum state becomes

$$-|1\rangle$$

(rather than $|1\rangle$).

In both cases the alternatives are equivalent – they are said to *differ* by a *global phase* because one is equal to the other multiplied by a complex number on the unit circle.

As an aside, `Statevector` will throw an error if the `measure` method is applied to an invalid quantum state vector.

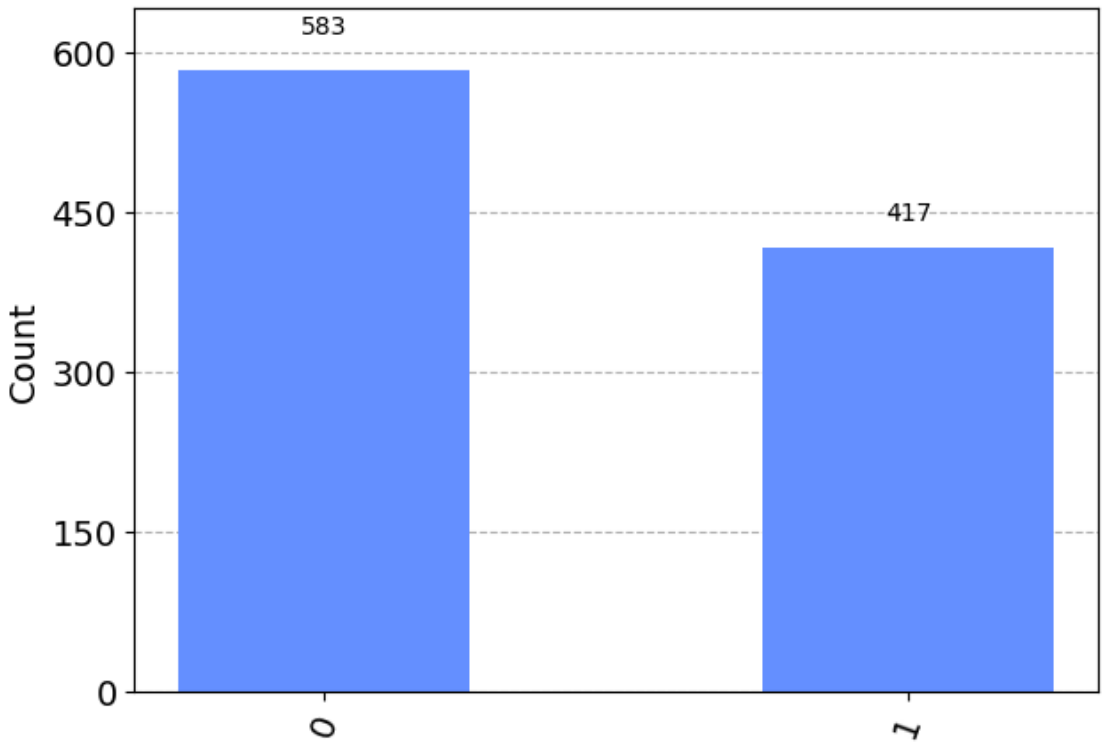
`Statevector` also comes with a `sample_counts` method that allows for the simulation of any # of measurements on the system. We can use `plot histogram` to visualize the results.

[10]:
```
from qiskit.visualization import plot_histogram

statistics = v.sample_counts(1000)
display(statistics)
plot_histogram(statistics)
```

```
{'0': 583, '1': 417}
```

[10]:



0.0.2 Peforming Operations with `Operator` and `Statevector`

Unitary operations can be defined and performed on state vectors in Qiskit using the `Operator` class

[11]:
```
from qiskit.quantum_info import Operator

X = Operator([[0, 1], [1, 0]])
Y = Operator([[0, -1.0j], [1.0j, 0]])
Z = Operator([[1, 0], [0, -1]])
H = Operator([[1 / sqrt(2), 1 / sqrt(2)], [1 / sqrt(2), -1 / sqrt(2)]])
S = Operator([[1, 0], [0, 1.0j]])
T = Operator([[1, 0], [0, (1 + 1.0j) / sqrt(2)]])

v = Statevector([1, 0])

v = v.evolve(H)
v = v.evolve(T)
v = v.evolve(H)
v = v.evolve(T)
v = v.evolve(Z)

v.draw("latex")
```

[11]:

$$(0.8535533906 + 0.3535533906i)|0\rangle + (-0.3535533906 + 0.1464466094i)|1\rangle$$

Looking ahead towards Quantum Circuits

We can define Quantum Circuits (which right now will simply be a sequence of unitary operations performed on a single qubit)

[12]:
```
from qiskit import QuantumCircuit

circuit = QuantumCircuit(1)

circuit.h(0)
circuit.t(0)
circuit.h(0)
circuit.t(0)
circuit.z(0)

circuit.draw()
```

[12]:
```
q:  H  T  H  T  Z
```

The operations are applied sequentially.

Let us first initialize a starting quantum state vector and evolve that state according to the above sequence of operations.

[13]:
```
ket0 = Statevector([1, 0])
v = ket0.evolve(circuit)
v.draw("latex")
```

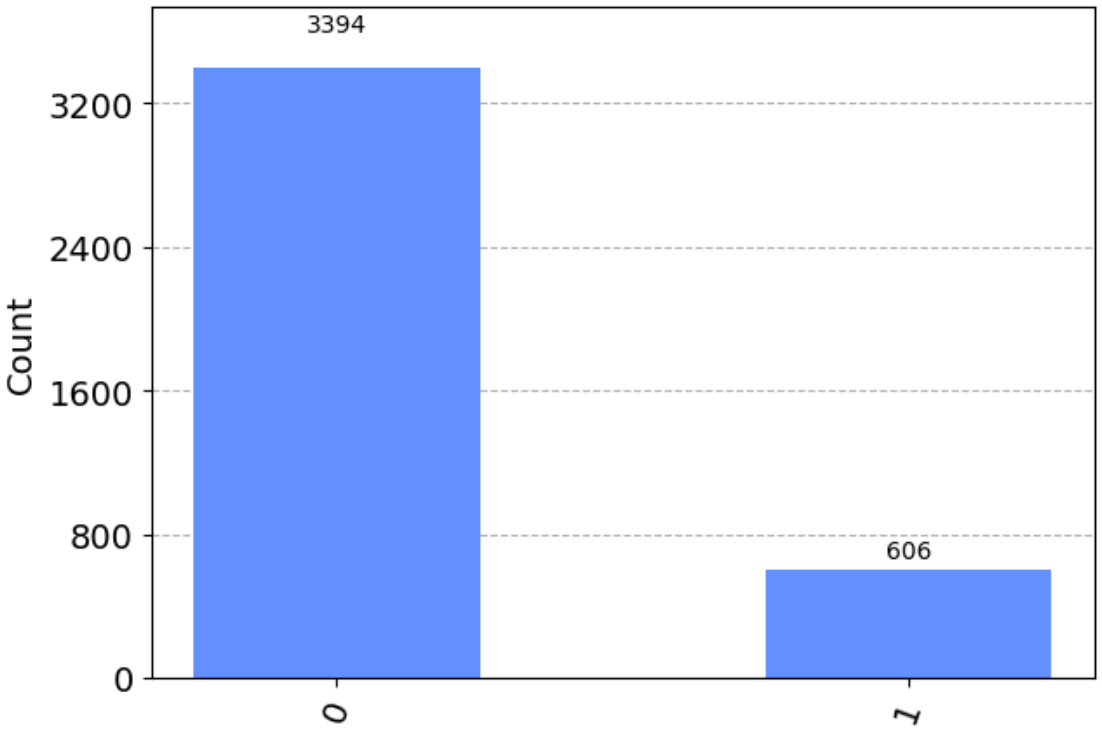
[13]:

$$(0.8535533906 + 0.3535533906i)|0\rangle + (-0.3535533906 + 0.1464466094i)|1\rangle$$

Finally let us simulate the result of running this experiment (i.e. preparing the state $|0\rangle$, applying the sequence of operations represented by the circuit and measuring) 4000 times.

[14]:
```
statistics = v.sample_counts(4000)
plot_histogram(statistics)
```

[14]:


3394
606
3200
2400
1600
800
0
Count
0
1