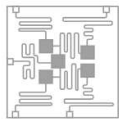


Week 4: Grinding gates in
quantum computers:
Quantum gates and circuit
model of quantum
computation, introduction
to IBM's Qiskit, Grover's
quantum search algorithm,
amplitude amplification

Sign up for user account

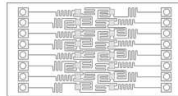
- In order to access IBM's cloud quantum computers
- Each user is given a token (for accessing)

IBM Q Backend Access



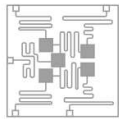
ibmqx4

Full Access



ibmqx5

Access using
QISKit



ibmqx2

MAINTENANCE

old machines

Promotional Code

Apply Code

API Token

`19a29c55a5fb8ece4cdaedaaba6240ab50f7ac_2025f020ea8fd6`

 **Copy API Token**

Regenerate

QISKit

Python software development kit (SDK) for working with OpenQASM and the IBM Q Experience (QX).

Download

Delete Account

If you delete your account, we will remove your email address and delete your personal data and you will not have access to IBM Q Experience.

How to run programs on IBM Q?

① □ Web interface

② □ Use “QISKit”, a python based package
install

- ✓ Quantum circuit part uses OpenQasm, quantum assembly language for Gate and operation specification for quantum circuits
- ✓ Can use IPython notebooks interactive environment and display quantum circuits
- ✓ Includes applications (QISKit Algorithms and Circuit for QUantum Applications) to : (1) Chemistry, (2) AI, and (3) Optimization, etc.

Web interface: drop-down gates

IBM Q 5 Tenerife [ibmqx4] ACTIVE: CALIBRATING

We are calibrating the device

	Q0	Q1	Q2	Q3	Q4
Frequency (GHz)	5.25	5.30	5.35	5.43	5.18
T1 (μ s)	53.50	51.50	39.40	42.70	51.30
T2 (μ s)	48.50	19.50	40.90	9.10	15.30
Gate error (10^{-3})	0.69	1.89	1.12	2.83	1.46
Readout error (10^{-2})	5.20	6.10	1.90	3.50	4.20
MultiQubit gate error (10^{-3})	CX1_0	CX2_0	CX3_2	CX4_2	
	2.88	2.11	11.80	4.69	
	CX2_1	CX3_4			
		3.89	7.11		

Last Calibration: 2018-07-09 22:02:28

IBM Q 5 Yorktown [ibmqx2] MAINTENANCE

Experiment #20180710101323 [Add a description](#) New Save Save as

[Switch to Qasm Editor](#) Backend: ibmqx4 My Units: 156 Experiment Units: 3 Run Simulate

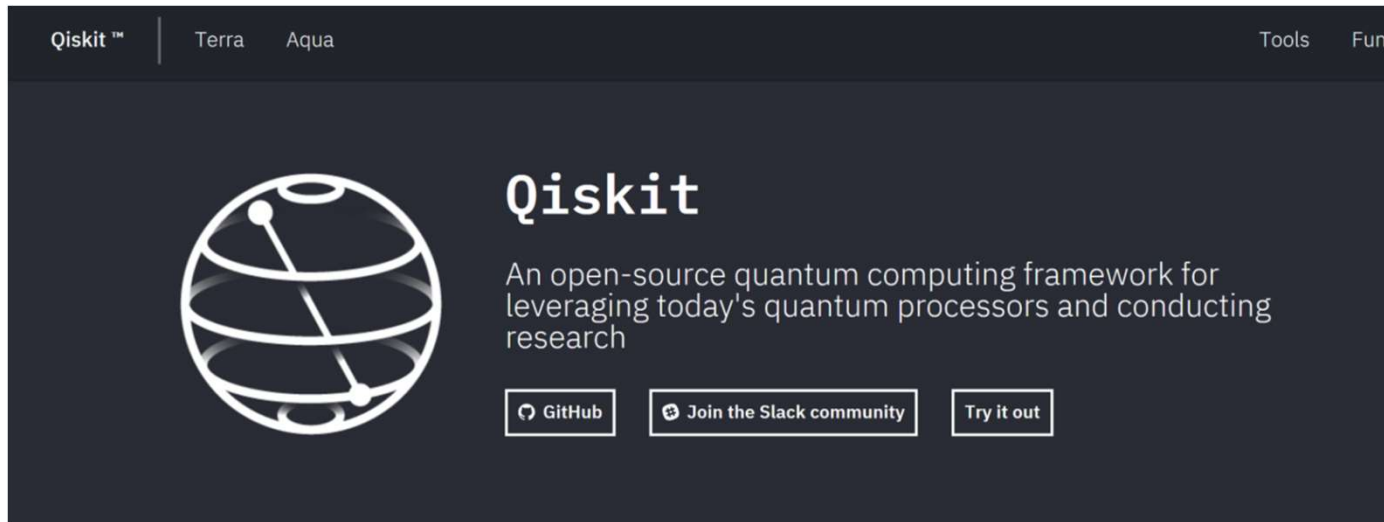
GATES Advanced

id	X	Y	Z
H	S	S [†]	+
T	T [†]		

BARRIER

OPERATIONS

http://qiskit.org



The screenshot shows the top navigation bar of the Qiskit website. On the left, there are links for 'Qiskit™', 'Terra', and 'Aqua'. On the right, there are links for 'Tools' and 'Fun'. Below the navigation bar is a dark blue banner. On the left side of the banner is the Qiskit logo, a white wireframe sphere with two dots and lines representing quantum gates. To the right of the logo, the word 'Qiskit' is written in a large, white, sans-serif font. Below the logo and title, there is a paragraph of text: 'An open-source quantum computing framework for leveraging today's quantum processors and conducting research'. At the bottom of the banner, there are three white buttons with black text: 'GitHub', 'Join the Slack community', and 'Try it out'.

Introducing VSCode extension!

Simplifying Qiskit to make developing quantum circuits and applications faster

[More information](#)

Getting started with Qiskit

In this episode Doug McClure, Qiskitter at IBM, introduces us to Qiskit and its functions. You'll learn all about how to run your first quantum program on real IBM Q hardware.

[Watch the video](#)

Python and Qiskit

Python and Qiskit

Qiskit requires Python 3 and you can install Qiskit according to the instruction here at the [documentation](#). [Python \(in particular Python 3\)](#) can be installed in Mac, Windows and Linux (Mac and Linux may come with it). We need at least Python 3.6. It is recommended to install on your own PC or laptop.

However, if you are comfortable with using online softwares and do not want to install these packages, you can sign up for a "CoCalc" account at here: <https://cocalc.com/>. It has the software you need for this course.

<https://cocalc.com>



[Features](#) [Software](#) [Pricing](#) [Policies](#) [Shared Files](#) [Doc](#) [Sign In](#)

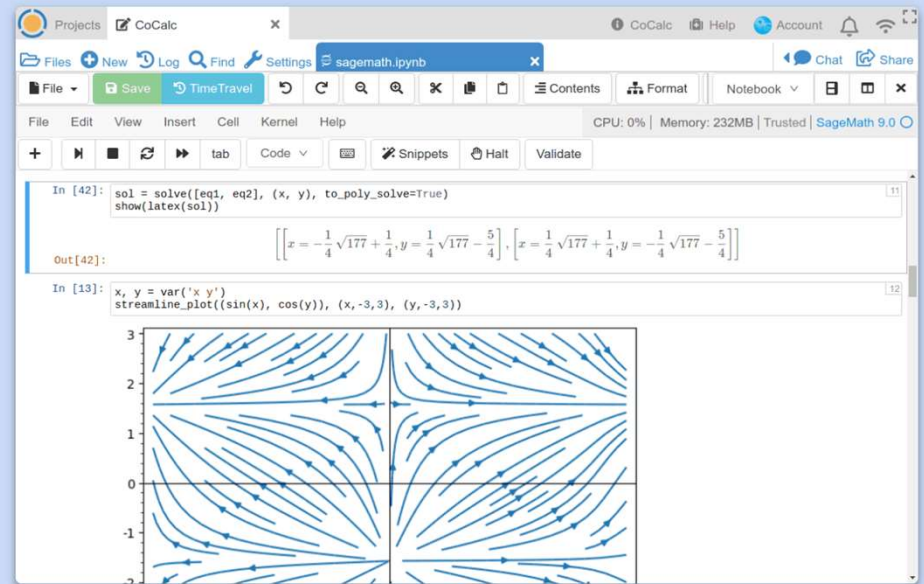
Collaborative Calculation and Data Science [Jupyter](#) [LaTeX](#) [Linux](#) [Octave](#) [Python](#) [R Stats](#) [Teaching](#) [Terminal](#) [X11](#) [Compare](#) [API](#)

Your best choice for teaching remote scientific courses!

Save weeks of class time troubleshooting software and make your TA's more effective.

[Run CoCalc Now](#)

[Sign In](#)



⚠ Thank you for trying CoCalc! Please [sign up](#) to avoid losing your work. ⚠

Files + New Log Find Welcome to CoCalc.ipynb x

File Save TimeTravel Contents Format Notebook

File Edit View Insert Cell Kernel Help

+ Stop Refresh tab Code Snippets Halt Validate

Select a Kernel

This notebook has no kernel. A working kernel is required in order to evaluate the code in the notebook. Please select one for the programming language you want to work with.

Do not ask, instead default to your most recent selection (you can always show this screen again by clicking on the kernel name in the upper right)

Suggested kernels

- Julia 1.5.1
- Python 3 (system-wide)**
- R (system-wide)

[The Julia Programming Language](#)

[Python 3 programming language](#)

[R statistical programming language](#)

→ Can run Qiskit codes on CoCalc (free)

```
In [1]: # Importing everything
from qiskit import *
from qiskit.visualization import plot_histogram

In [2]: # Define a function that takes a QuantumCircuit (qc)
# and two integers (a & b)
def create_bell_pair(qc, a, b):
    qc.h(a) # Apply a h-gate to the first qubit
    qc.cx(a,b) # Apply a CNOT, using the first qubit as the control

In [3]: # Define a function that takes a QuantumCircuit (qc)
# a qubit index (qubit) and a message string (msg)
def encode_message(qc, qubit, msg):
    if msg == "00":
        pass # To send 00 we do nothing
    elif msg == "10":
        qc.x(qubit) # To send 10 we apply an X-gate
    elif msg == "01":
```