USC University of Southern California

EE 105: Introduction to Electrical Engineering

Lab 0.2

Python bootcamp

Yasser Khan

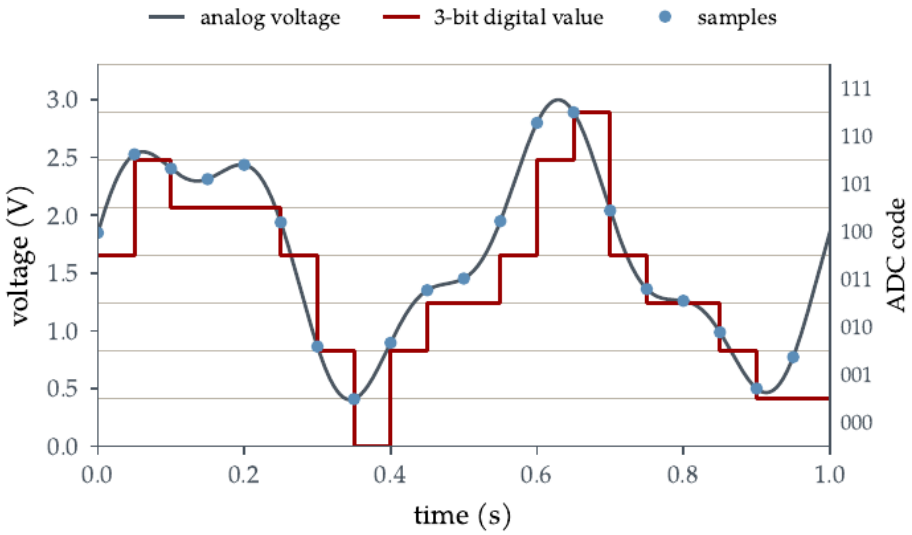
Rehan Kapadia

Today's mission

Drill the Python and NumPy you will use every week: six questions, a linear system, real data.

By the end of this lab you can:

- Pass the **six bootcamp questions** with the autograder
- Solve **A x = b** with np.linalg.solve and check the answer
- Summarize measurements: **mean, standard deviation, ADC codes**
- Find nine voltage levels in 2400 readings with **reshape** and **axis=1**



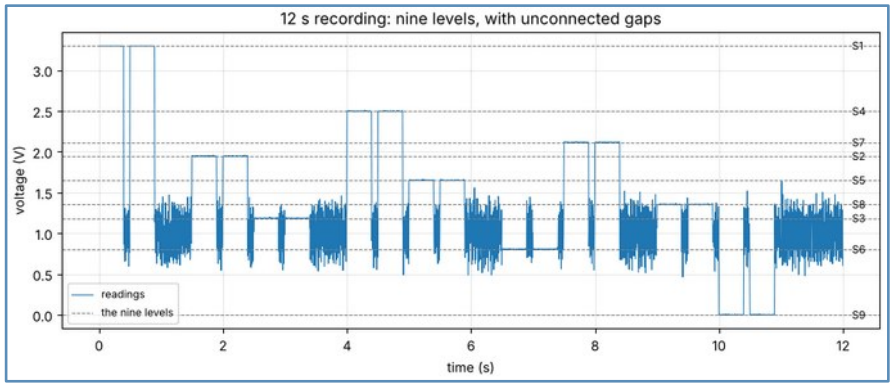
The ADC turns a voltage into whole-number codes; Python works with the numbers.

A Start **5 min** → **B** The guide **25 min** → **C** Six questions **30 min** → **D** Linear equations **15 min** → **E** Measurements **30 min**

Key takeaways

What to remember from this lab, with the numbers that matter.

- **Lists join, arrays compute.** [1, 2] + [1, 2] has four elements; arrays add element by element, and @ is the matrix product
- **Slices stop before the stop.** A[1::2, ::2] keeps the odd rows and the even columns
- **axis=0 down the columns, axis=1 across the rows.** After reshape(-1, 10), axis=1 gives one mean per 50 ms window
- **A x = b is np.linalg.solve(A, b).** Check it: A @ x - b near 1e-15
- **One ADC step is 3.22 mV.** A steady voltage falls on a few whole codes, about one step apart



Expected result of E3: flat stretches on the nine levels, wild swings in the unconnected gaps.

SETUP

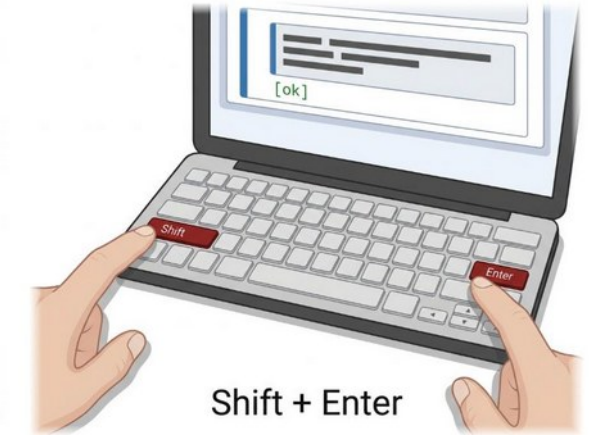
No board today. Run the kit's setup file once, then open the lab notebook in JupyterLab.

Windows

1. Unzip **ee105-labs.zip**
2. Double-click **setup-windows.bat**

Mac

1. Unzip **ee105-labs.zip**
2. First time: right-click **setup-mac.command**, Open
3. Later: a double-click starts JupyterLab



Shift+Enter runs a cell

```
ee105-labs/
  lab0.2-python-bootcamp/
    lab0.2-python-bootcamp.ipynb    <- open this
    bootcamp-guide.ipynb    bootcamp-questions.ipynb
```

Then, on both: open **lab0.2-python-bootcamp.ipynb** and run **A1 • RUN**: every line starts with **[ok]**. No board today.

Part B: The guide

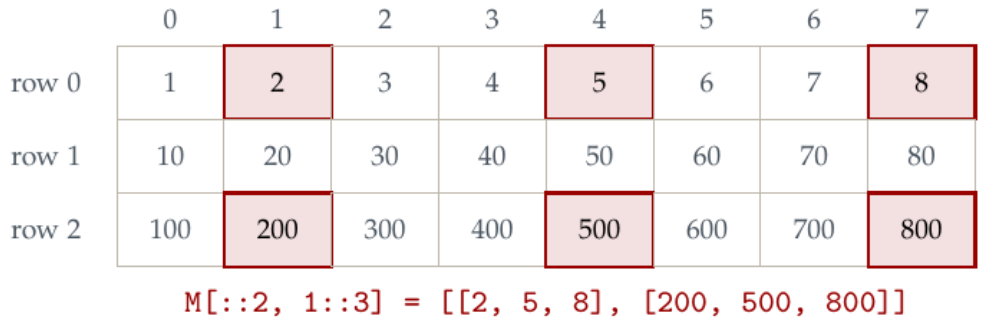
USC University of Southern California

Open the guide notebook and run every example. Predict each output before you run it.

PART B

Goal: work through bootcamp-guide.ipynb and change at least one value in every example.

- Open **bootcamp-guide.ipynb** from the file list; keep the lab notebook open
- Before each cell, **predict** the output; then run it
- Watch three things: **lst + lst** versus **arr + arr**, slicing with steps, and **@** versus element-wise *



M[::2, 1::3] keeps rows 0 and 2 and columns 1, 4 and 7.

You should see: [1, 2] + [1, 2] gives **[1, 2, 1, 2]**, but np.array([1, 2]) + np.array([1, 2]) gives **[2, 4]**. A * B multiplies element by element; A @ B is the matrix product.

Part C: The six questions

Six short functions in bootcamp-questions.ipynb; each cell runs its own test.

PART C

Goal: complete the six functions in bootcamp-questions.ipynb until the score is 6/6.

- **C1 · RUN** Load the autograder: **%run autograder_v2.py**
- **C2 to C7 · EDIT** Questions 1 to 6: write the function, run the cell; it tests itself
- **C8 · RUN test_all(...)** prints your score
- **Save** the questions notebook (Ctrl+S, or Cmd+S on a Mac)

Expected result · C8

```
Question 1 Passed!
Question 2 Passed!
Question 3 Passed!
Question 4 Passed!
Question 5 Passed!
Question 6 Passed!
Score: 6/6
```

You should see: Score: 6/6. Write down which questions failed at first and why (Question 1 asks for it).

Check (Checkpoint 1): the questions notebook prints Score: 6/6.

Part D: Linear equations

D1 · RUN: three cafe orders, three unknown prices, written as A x = b and solved.

```
A_cafe = np.array([[2.0, 1.0, 1.0],
                   [1.0, 2.0, 0.0],
                   [1.0, 1.0, 2.0]])
b_cafe = np.array([13.25, 8.00, 13.75])
x_cafe = np.linalg.solve(A_cafe, b_cafe)

print(x_cafe)       # coffee, bagel, juice
print(A_cafe @ x_cafe - b_cafe)   # about 0
```

One row per order, one column per item

$2c + b + j = 13.25$

$c + 2b = 8.00$

$c + b + 2j = 13.75$

D2 · EDIT: the school store

- 3 pens, 2 notebooks, 1 folder: $8.40
- 1 pen, 4 notebooks: $3.80
- 2 pens, 1 notebook, 2 folders: $10.05

You should see: cafe: coffee **3.50**, bagel **2.25**, juice **4.00** dollars, and A @ x - b of about **1e-15**: rounding, not a mistake.

Part E: 200 readings

USC University of Southern California

a1_readings.csv holds 200 readings of one steady voltage. How noisy is it?

PART E

Goal: compute the mean and standard deviation and count the ADC codes.

- **E1 · RUN** Load **a1_readings.csv**: columns t_s, raw and volts
- **E2 · EDIT** v_mean with **np.mean**, v_std with **np.std**; the cell compares v_std with one ADC step, **3.22 mV**, and counts the codes
- **E3 · RUN** Readings and a histogram of the codes, then **levels.csv**

Expected result · E1

	t_s	raw	volts
0	0.000	479	1.54365
1	0.005	479	1.54365
2	0.010	478	1.54043
3	0.015	478	1.54043
4	0.020	478	1.54043

You should see: mean **1.539 V**, standard deviation **3.08 mV** (0.96 of one step), **5** codes, 476 to 480. The histogram has gaps: nothing lands between two codes.

Part E: Find the nine levels

A 12 s recording switches between nine voltages, with unconnected gaps in between.

PART E

Goal: label every steady 50 ms window with its level, then count them.

- **E4 · EDIT nearest_level(v)** with **np.argmin**; expect **S5 S1 None**
- **E5 · EDIT reshape(-1, 10)**: a row per window; mean and spread with **axis=1**
- **E6 · EDIT** Count the windows of each level (a dictionary comprehension)

How a window gets a name

1. 10 readings = one 50 ms window
2. steady if its spread is below 40 mV (about 12 steps)
3. named after the nearest level, if within 50 mV
4. a level is found with 2 or more windows

You should see: 240 windows, **144** steady, **16** steady windows (0.8 s) for each of the nine levels: found **9 of 9**.

Check (Checkpoint 2): Part D prints the three prices with A @ x - b near zero, and Part E finds all nine levels.

Stuck?

USC University of Southern California

Check these first. If the fix does not help, raise your hand.

A [!!] line in Part A

A package is missing: close JupyterLab and run the setup file again.

Save reads old answers

Save bootcamp-questions.ipynb first (Ctrl+S or Cmd+S), then run Save again.

Q2 picks the wrong cells

Rows first, then columns: odd rows and even columns is A[1::2, ::2].

Q5 is one element short

np.arange(1, N) stops at N - 1. Use np.arange(1, N + 1).

Q6 gives a shape error

Divide by A.sum(axis=1, keepdims=True) so each row is divided by its own sum.

%run autograder_v2.py fails (C1)

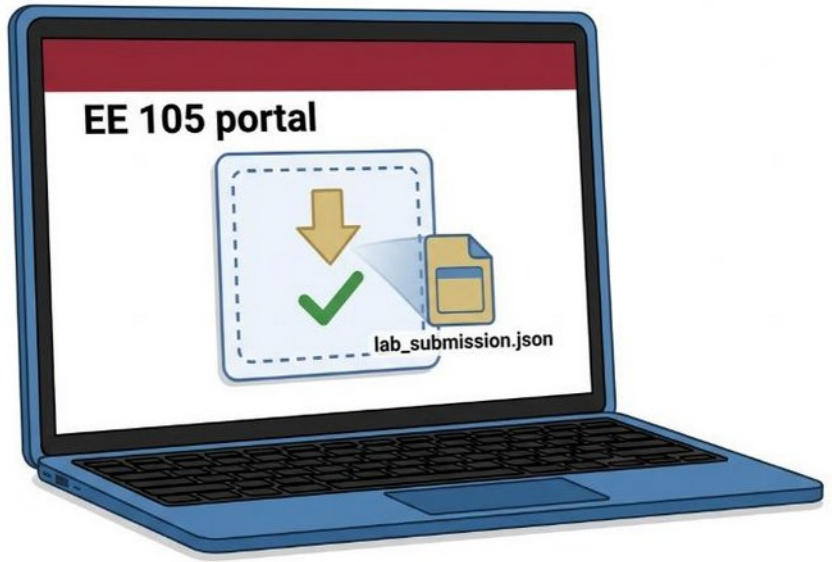
Open the questions notebook from its own folder in JupyterLab.

Submit your results

The last cell saves your results in one file. Upload it on the course portal.

`lab0.2_submission.json`

1. Save **bootcamp-questions.ipynb** first
2. **Save · RUN** runs your six functions on new inputs and writes **lab0.2_submission.json**
3. Log in at **ee105.org/portal** with your student ID and portal PIN; open Lab 0.2
4. Upload the file. It is graded automatically; you can upload again after a fix



ee105.org/portal

Check: the portal checks your answers on new inputs, so a function that passes the tests only by accident shows up there.

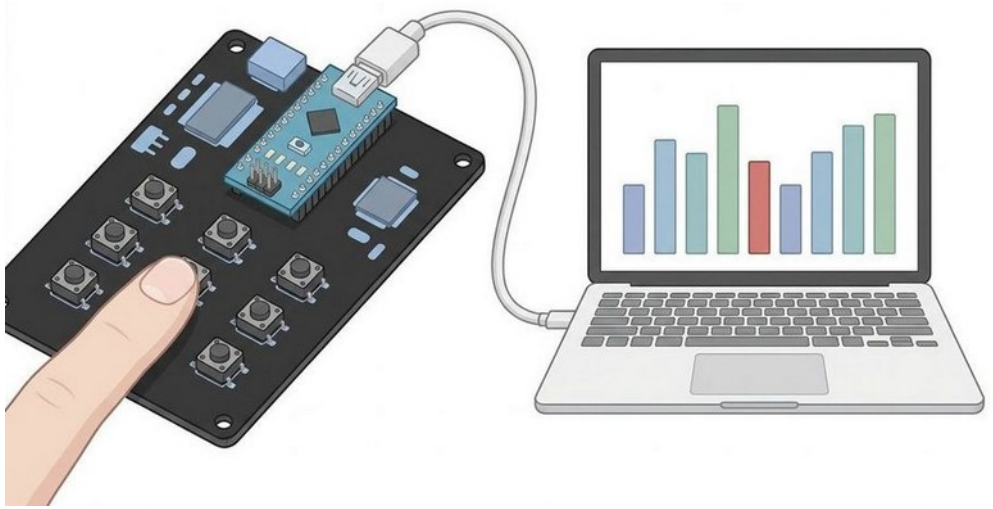
Lab complete

USC University of Southern California

You have the Python toolkit for the semester: arrays, linear algebra, real data.

You built:

- Six working NumPy functions
- A solver for small linear systems, checked with A @ x
- A noise and ADC-step analysis of 200 readings
- A level finder that finds nine voltages in 2400 readings



Next: the nine levels are the nine buttons of this touch panel.

Did you know?

Your laptop has about two million times the RAM of the Apollo 11 guidance computer, which had 2048 words of 16 bits (Chapter 0).

Next

Lab 1.1, asymmetric touch panel: measure the nine buttons and decode them. Bring the board.