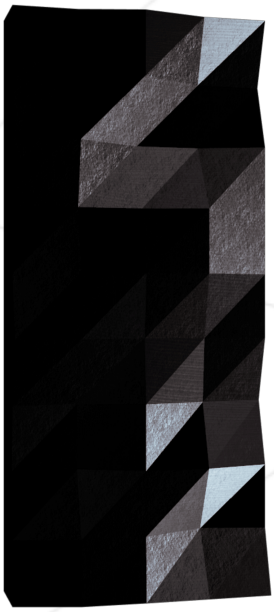


Rush

Numpy

femi femi@42.us.org
gaetan gaetan@42.fr

Summary: An introduction to Numerical Python (NumPy)

Contents

Chapter I

Foreword

What is Math for?

"When someone asks you what math is for, they're not asking you about applications of mathematical science. They're asking you, why did I have to study that bullshit I never used in my life again?

It's true that math doesn't need to serve a purpose, it's true that it's a beautiful structure, a logical one, probably one of the greatest collective efforts ever achieved in human history. It's also true that there, where scientists and technicians are looking for mathematical theories that allow them to advance, they're within the structure of math, which permeates everything.It's true that we have to go somewhat deeper, to see what's behind science.

Science operates on intuition, creativity. Math controls intuition and tames creativity. Almost everyone who hasn't heard this before is surprised when they hear that if you take a 0.1 millimeter thick sheet of paper, the size we normally use, and, if it were big enough, fold it 50 times, its thickness would extend almost the distance from the Earth to the sun. Your intuition tells you it's impossible. Do the math and you'll see it's right. That's what math is for.

It's true that science, all types of science, only makes sense because it makes us better understand this beautiful world we live in. And in doing that, it helps us avoid the pitfalls of this painful world we live in. There are sciences that help us in this way quite directly. Oncological science, for example. And there are others we look at from afar, with envy sometimes, but knowing that we are what supports them. All the basic sciences support them, including math. All that makes science, science is the rigor of math. And that rigor factors in because its results are eternal.

You probably said or were told at some point that diamonds are forever, right? That depends on your definition of forever! A theorem – that really is forever. The Pythagorean theorem is still true even though Pythagoras is dead, I assure you it's true. Even if the world collapsed the Pythagorean theorem would still be true. Wherever any two triangle sides and a good hypotenuse get together the Pythagorean theorem goes all out. It works like crazy.

Mathematicians devote [themselves] to come up with theorems. Eternal truths. But it isn't always easy to know the difference between an eternal truth, [a] theorem, and a mere conjecture. You need proof. For example, let's say I have a big, enormous, infinite field. I want to cover it with equal pieces, without leaving any gaps. I could use squares, right? I could use triangles. Not circles, those leave little gaps. Which is the best shape to use? One that covers the same surface, but has a smaller border. In the year 300, Pappus of Alexandria said the best is to use hexagons, just like bees do. But he didn't prove it. The guy said, "Hexagons, great! Let's go with hexagons!" He didn't prove it, it remained a conjecture. "Hexagons!" And the world, as you know, split into Pappists and anti-Pappists, until 1700 years later when in 1999, Thomas Hales proved that Pappus and the bees were right – the best shape to use was the hexagon. And that became a theorem, the Honeycomb Theorem, that will be true forever and ever, for longer than any diamond you may have.

So, if you want to tell someone that you will love them forever you can give them a diamond. But if you want to tell them that you'll love them forever and ever, give them a theorem! But hang on a minute! You'll have to prove it, so your love doesn't remain a conjecture." [1]

[1] Excerpt of a TED talk by Eduardo Sáenz de Cabezón

Chapter II

Introduction

Python is a interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. [1]

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. [2]

IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history.
IPython provides the following features: [3]

- A browser-based notebook with support for code, text, mathematical expressions, inline plots and other media.
- Interactive shells (terminal and Qt-based).
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.

```
Python, unlike C and many other languages, does not use curly braces
to define scope, but rather whitespaces.  Your bug could be because
of the wrong number of tabs.
```

[1] Python
[2] NumPy
[3] IPython

Chapter III

Subject

This project will be done using an IPython Jupyter Notebook. A Dockerfile is provided to save you installation time. Feel free to read through the Dockerfile. Follow the instructions (at the bottom of the Dockerfile) to launch the Docker container and create a new IPython (.ipynb) file to do this project. At the end of this project you will turn in the IPython file. The author's login should be the first cell.

III.1 Part I

Example:
Create a matrix (r,c) of all ones or zeros.

```
In [131]: np.ones((3,2))

Out[131]: array([[ 1.,  1.],
                 [ 1.,  1.],
                 [ 1.,  1.]])
```

1. Create a matrix of random values of a distribution of your choice.

2. Create a 1-dimensional array of 12 sequential numbers and convert it to a 4x3 array.

3. Write a function that creates an incremental N-dimensional (nd) array of dimension (1,n) with values between 0 and 1. Use `arr.shape` to verify

4. Generate a 10x12 array and extract row 0-4 of columns 8-12.

5. Using the function in Q3, get m vectors and bind them together (to have a m x n) matrix. Plot the matrix with matplotlib's `imshow`

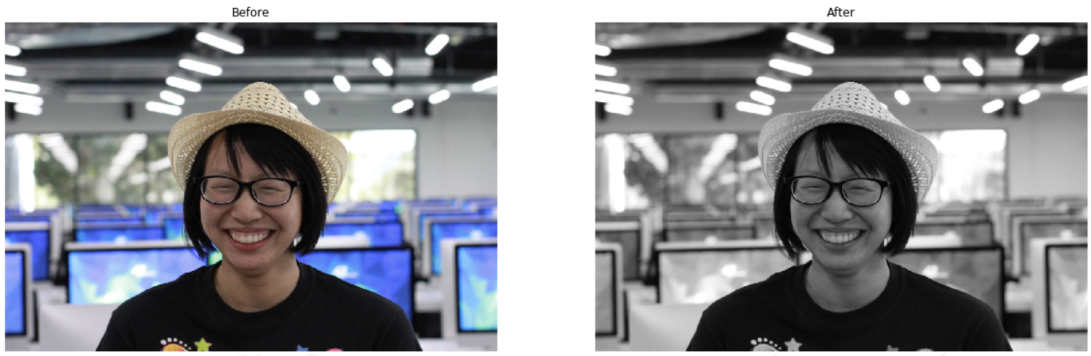
6. Multiply the resulting matrix from Q5 with the matrix of a picture of your choice. Plot the resulting matrix.

III.2 Part II

Filters.
Image editing / processing is done by changing the values of the matrix (pixel by pixel).
In this section you will code FOUR (non-grayscale) commonly used image filters.

Example:

```
Out[34]: (-0.5, 1199.5, 799.5, -0.5)
```



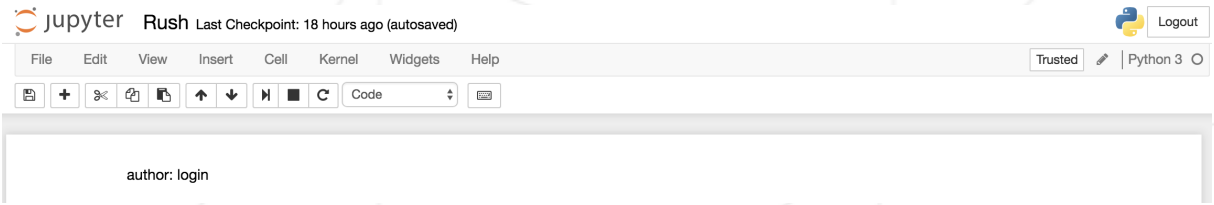
Filters to choose from: (description of each can be found here)

- Amaro
- Mayfair
- Rise
- Hudson
- Valencia
- X-Pro II
- Nashville
- Lo-Fi
- Sierra
- Earlybird
- Sutro
- Toaster

Chapter IV

Instructions

- Your root folder must be named `notebooks`.
- Your IPython file (.ipynb) must be in the `notebooks` folder.
- The first cell of your IPython Notebook must contain your login entered as a Markdown



- You can only use modules installed via the provided Dockerfile.
- You can ask your questions on slack.
- This subject could change, up until 1 hour before the deadline.

Chapter V

Bonus part

More filters :)

Chapter VI

Turn-in and peer-evaluation

Turn your work in using your `GiT` repository, as usual. Only work present on your repository will be graded in defense.