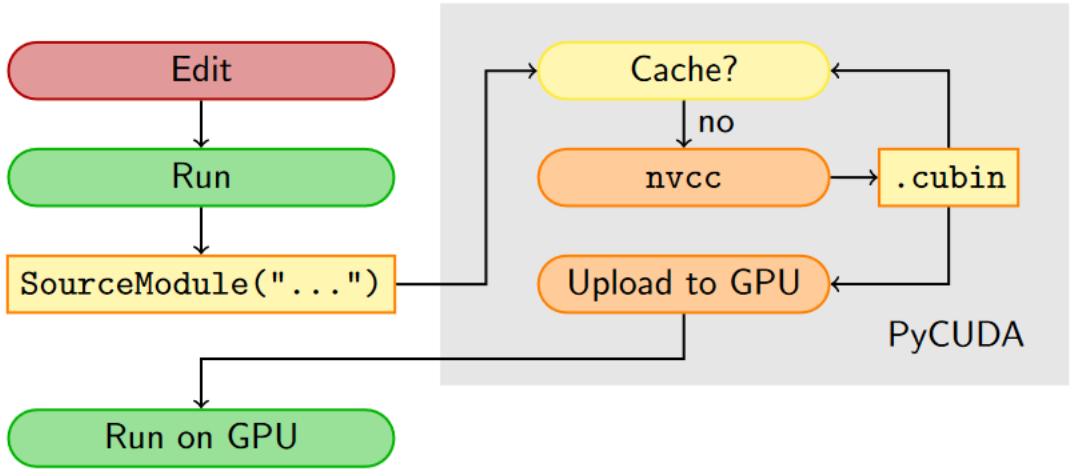
Introduction to GPU computing (3)

Computing Methods for Experimental Physics and Data Analysis
GPU & Python Hands-on : Lecture 4

gianluca.lamanna@unipi.it

PyCuda Module & Numba

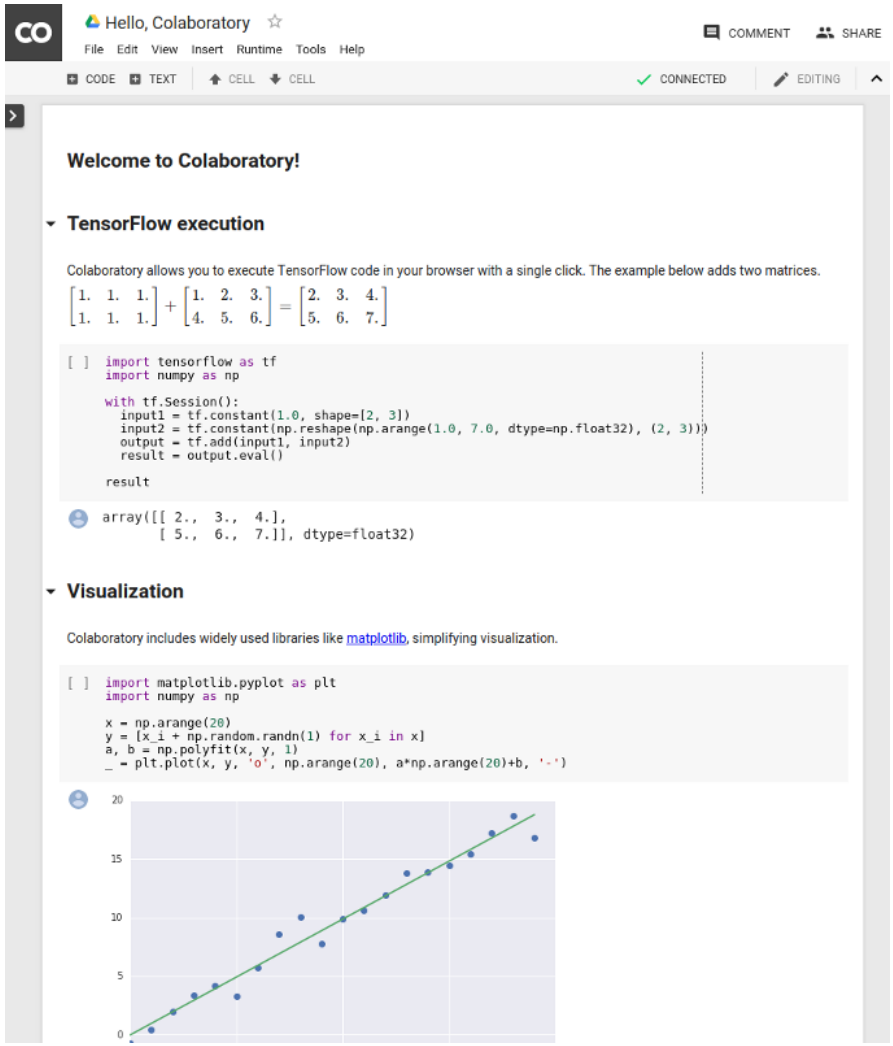
- PyCUDA lets you access Nvidia's CUDA parallel computation API from Python
  - ➞ All the CUDA features can be accessed through pyCUDA
- Supports Just-in-time compilation of the CUDA kernels in C
- Small overhead with respect to the C implementation to the GPU part
- Several additional features
  - ➞ Example: cuda exceptions translated to python exception
- One of the virtues of PyCUDA is that is allows us to use the class **GPUArray**
- We will use Numba to compile ufuncs on GPU



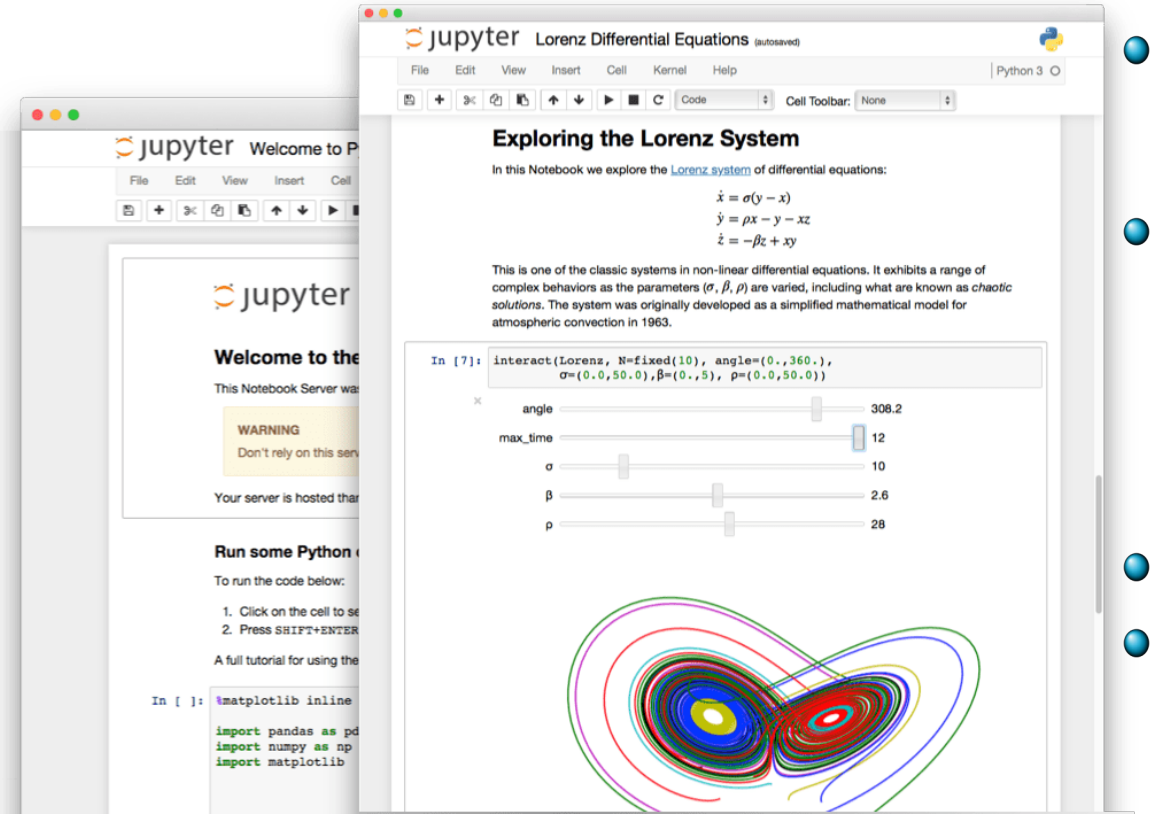
- https://pypi.org/project/pycuda/
- https://documen.tician.de/pycuda/
- http://numba.pydata.org/numba-doc/latest/cuda/index.html

Colab

- Also know as *Colaboratory,* is free *Jupyter* notebook running on google cloud
  - ➡ The notebooks are stored in google drive
  - ➡ http://colab.research.google.com
- The notebooks are environment to write text and run code based on Python3
  - ➡ It's possible to run on cloud computers housing GPUs
- Thanks to the IPython library it's possible to run shell commands (including compilers) on the cloud filesystem
- Possibility to add modules in the development environment



Jupyter



https://www.geeksforgeeks.org/how-to-use-jupyter-notebook-an-ultimate-guide/

- Colab implement a cloud version of the Jupyter notebook
  - ➡https://jupyter.org/
- A Jupyter Notebook document is a JSON document
  - ➡ ordered list of input/output cells
  - ➡ can contain code, text, latex, mathematics, plots and media
  - ➡ ".ipynb" extension.
- It's free and open-source
- It implements a language shell (aka interactive toplevel) environment built on IPython library
  - ➡IPython is command shell for interactive python
  - ➡Jupyter is a web-based, graphics implementation of IPython
- Other programming languages (49) are supported including R, Matlab, Julia, etc.