# PYTHON FOR MATHEMATICAL COMPUTING

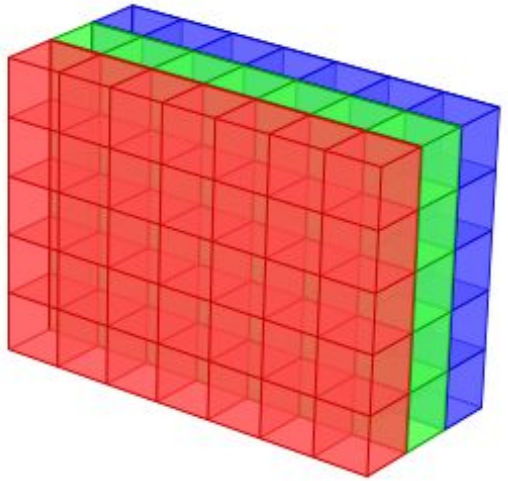
# OUTLINE

- Intro to Jupyter notebook and numpy installation
- Matrix manipulations
- Solving linear equations
- Eigenvalues/Eigenvectors

# A WISH LIST

- we want to work with vectors and matrices

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

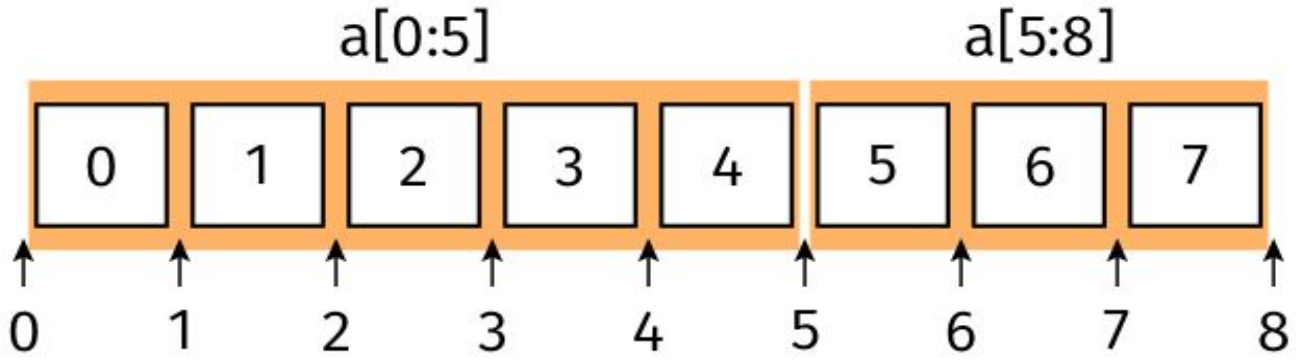


colour image as $N \times M \times 3$-array

- we want our code to run fast
- we want support for linear algebra
- ...

# LIST SLICING

basic syntax: `[start:stop:step]`




- if `step=1`
  - slice contains the elements `start` to `stop-1`
  - slice contains `stop-start` elements
- `start`, `stop`, and also `step` can be negative
- default values:
  - `start` 0, i.e. starting from the first element
  - `stop` N, i.e up to and including the last element
  - `step` 1

# WORK WITH LISTS

Can we use lists of lists to work with matrices?

$$\begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{pmatrix}$$

```
matrix = [[0, 1, 2],
          [3, 4, 5],
          [6, 7, 8]]
```

- How can we extract a row?
- How can we extract a column?

# WORK WITH LISTS

Can we use lists of lists to work with matrices?

$$\begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{pmatrix}$$

```
matrix = [[0, 1, 2],
          [3, 4, 5],
          [6, 7, 8]]
```

- How can we extract a row? ☺
- How can we extract a column? ☹

Lists of lists do not work like matrices

# NEW APPROACH

We need a new object

**ndarray**

multidimensional, homogeneous array of fixed-size items

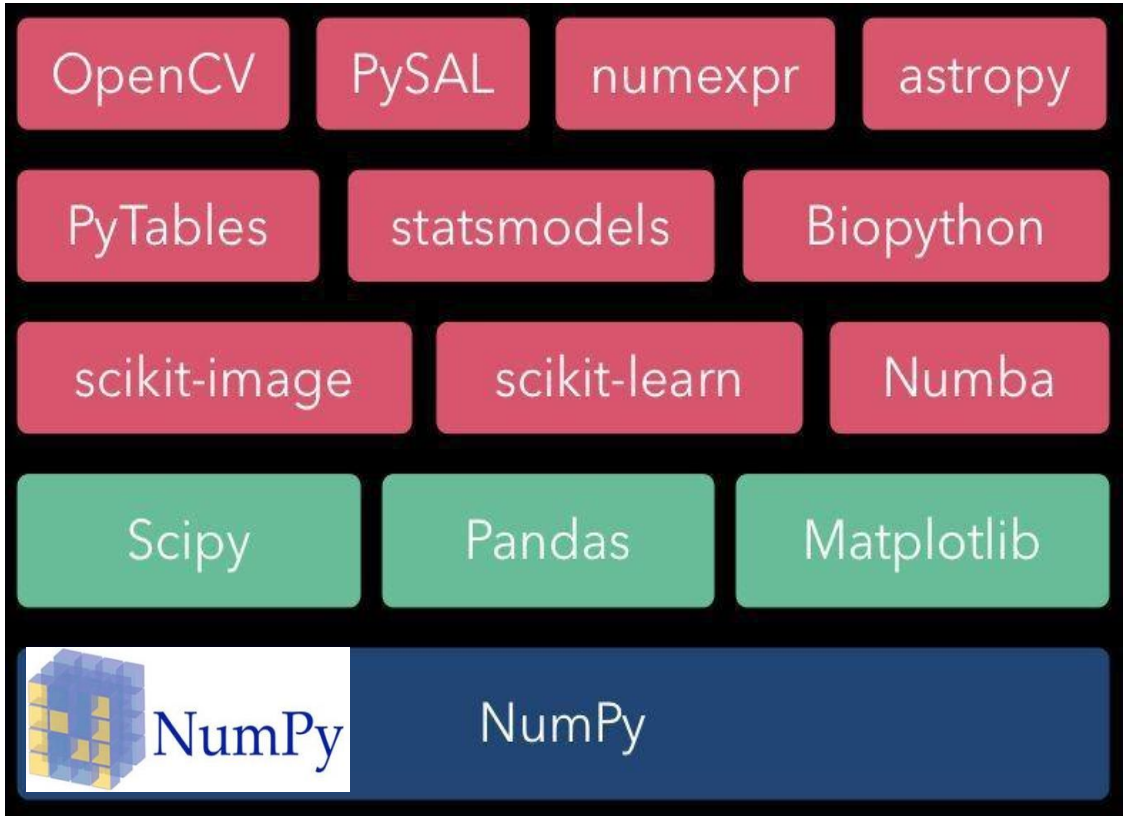
# NUMPY

- NumPy is a Python C extension library for array-oriented computing  Efficient In-memory Contiguous (or Strided) Homogeneous (but types can be algebraic)

**NumPy is suited to many applications**

- Image processing
- Signal processing
- **Linear algebra**
- A plethora of others

The libraries on the right were all built on top of numpy.



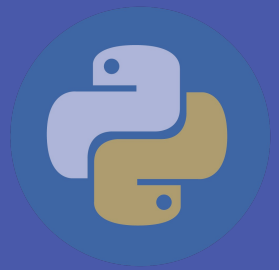
# WHAT DO YOU EXPECT

- How to import NumPy
- How to create multidimensional NumPy ndarrays using various methods
- Apply the Linear Algebra operations on them
- How to access and change elements in ndarrays
- How to use slicing to select or change subsets of an ndarray
- Understand the difference between a view and a copy an of ndarray
- How to use Boolean indexing and set operations to select or change subsets of an ndarray
- How to sort ndarrays
- How to perform element-wise operations on ndarrays
- Understand how NumPy uses broadcasting to perform operations on ndarrays of different sizes.

# WHERE TO CODE

https://jupyter.org/try

Alternative: Google CoLab

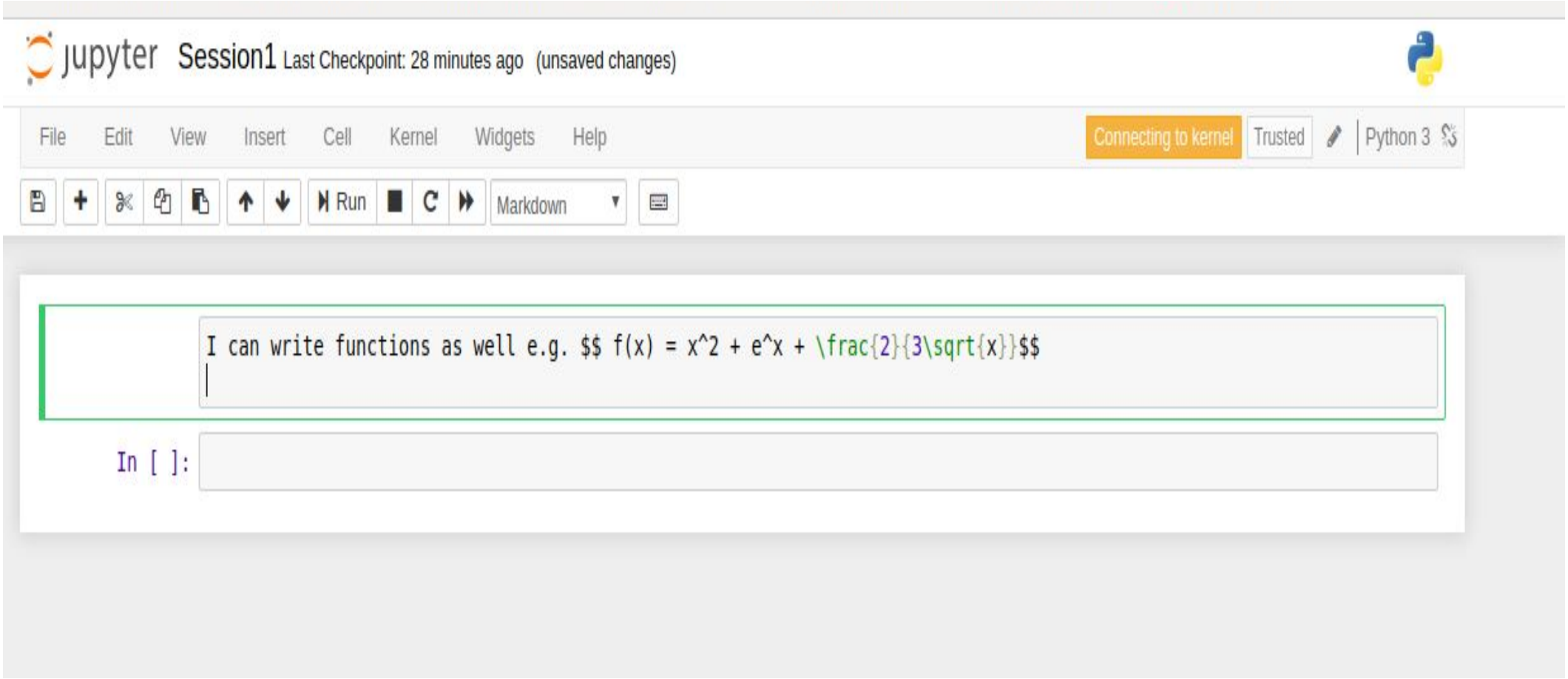


# Jupyter Notebook

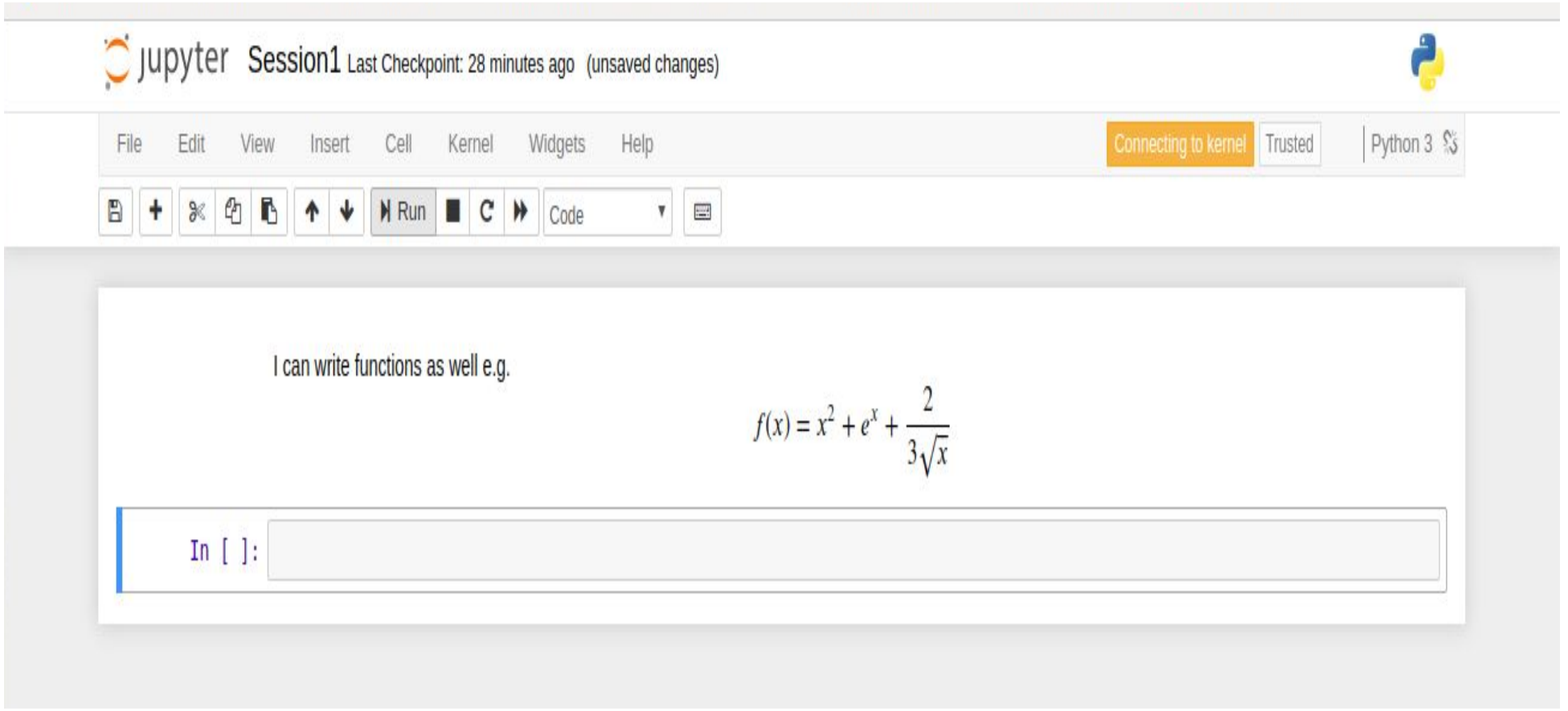
**Why Jupyter?**

1. Centralized platform to document code, run it, visualize it and view the results.
2. Sharing code.
3. Code is written in individual cells - great for prototyping

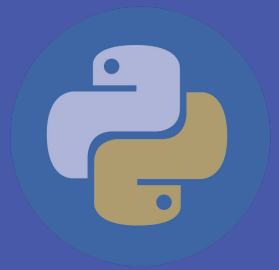
Has an inbuilt latex translator. Try it..



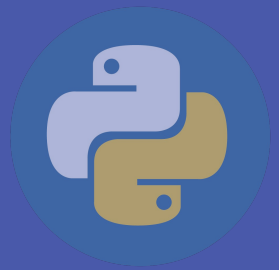
# Jupyter Notebook



# HOW DOES NUMPY HELP IN LINEAR ALGEBRA? JUPYTER



# LET'S WORK OUT SOME MATH PROBLEMS



# CREATING A MATRIX/VECTOR

1. Write code that adds two random (9x9) matrices and then scales the result by 0.3

2. Write code that takes two random matrices, one (3x4) the other (4x1) and multiplies them

3. Write code that results in the dot product of [1,4,3,2]transpose and [3,-2,2,-4]transpose

# MORE MATRIX OPERATIONS

Addition: np.add(A,B) or A+B

Multiplication: np.multiply(A,B, dtype = np.float128) or A*B

Transpose: A.transpose()

Determinant: np.linalg.det(a)

Inverse: np.linalg.inv(x)

Dot product: np.dot(A,B)

Eigenvalues and eigenvectors:

# MORE MATRIX OPERATIONS

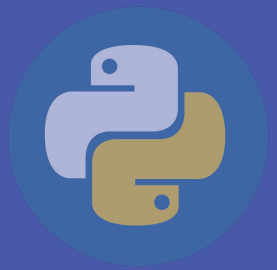
Let's compute the:

1. Determinant of a random 5x5 matrix
2. The inverse of a 6x4 matrix
3. The dot product of two 7x1 matrices

# NEXT SESSION

- MORE NUMPY
- SVD & PCA

Refer to that github repository for all materials



# THANK YOU!!!

ANY QUESTIONS?