Procesamiento Digital de Señales

Introducción a Python

Facultad de Ingenieria, Universidad de Antioquia

PhD(c) Juan Camilo Vásquez Correa

19 de junio de 2019

Contenido

Evaluación

- 9 laboratorios.
- Se entrega el informe la antes de la siguiente clase.
- Penalidad por retraso en el tiempo de entrega.
- 60 %

Cada punto de la práctica se evaluará 40 % práctico (que el código funcione adecuadamente y sea claro y ordenado) y 60 % teórico (observaciones y conclusiones).

UNIVERSIDAD DE ANTIOQUIA
1803

Laboratorio Final

- Proyecto Individual.
- Resume todas las practicas.
- Cada uno con señales diferentes.
- Dos entregas parciales durante el semestre, y entrega final.
  - Entrega 1: julio 24 (semana 5): motivación y descripción del problema.
  - Entrega 2: agosto 28 (semana 10): avance 1.
  - Entrega final: octubre 16 (semana 16).
- 40 %

UNIVERSIDAD DE ANTIOQUIA
1803

Próxima sección

Outline

Why Python

Guido Van Rossum

UNIVERSIDAD DE ANTIOQUIA
1803

Why Python

Recuperar las palabras de un documento en ABC

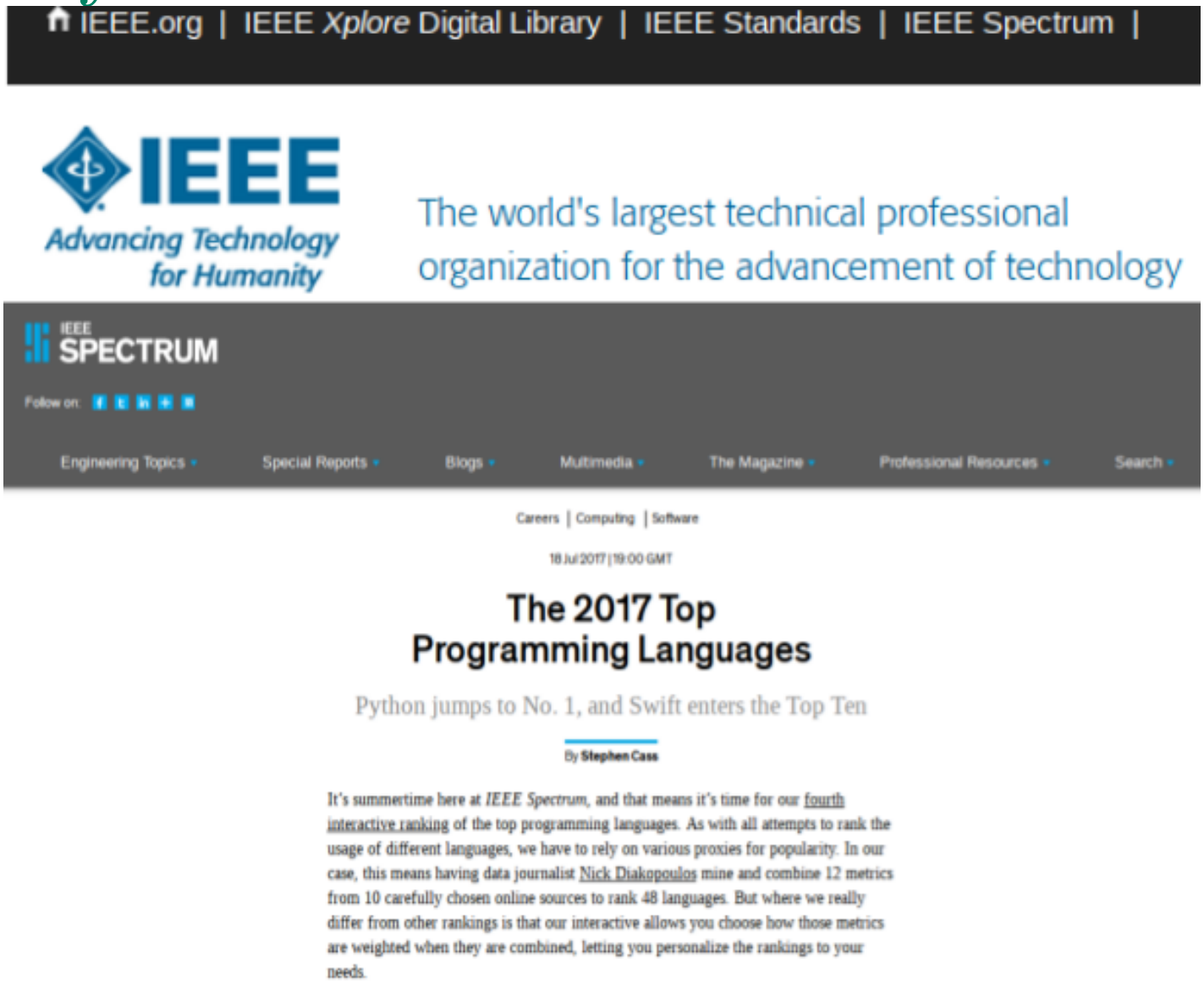
```
HOW TO RETURN words document:
    PUT {} IN collection
    FOR line IN document:
        FOR word IN split line:
            IF word not.in collection:
                INSERT word IN collection
    RETURN collection
```

Recuperar las palabras de un documento en Python

```
def words(document):
    collection = set()
    for line in document:
        for word in line.split():
            if word not in collection:
                collection.add(word)
    return collection
```

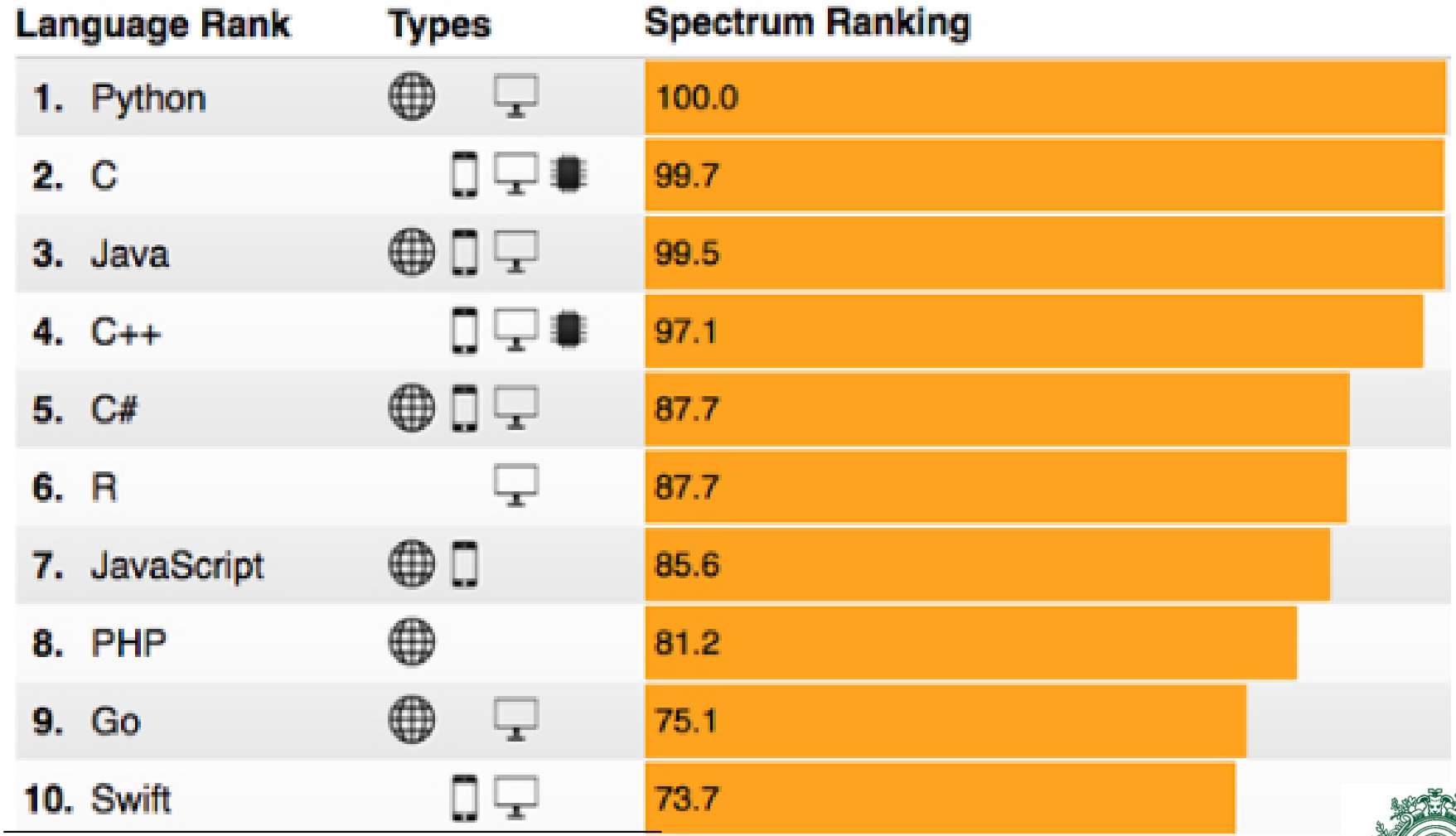
UNIVERSIDAD DE ANTIOQUIA
1803

Why Python



UNIVERSIDAD DE ANTIOQUIA
1803

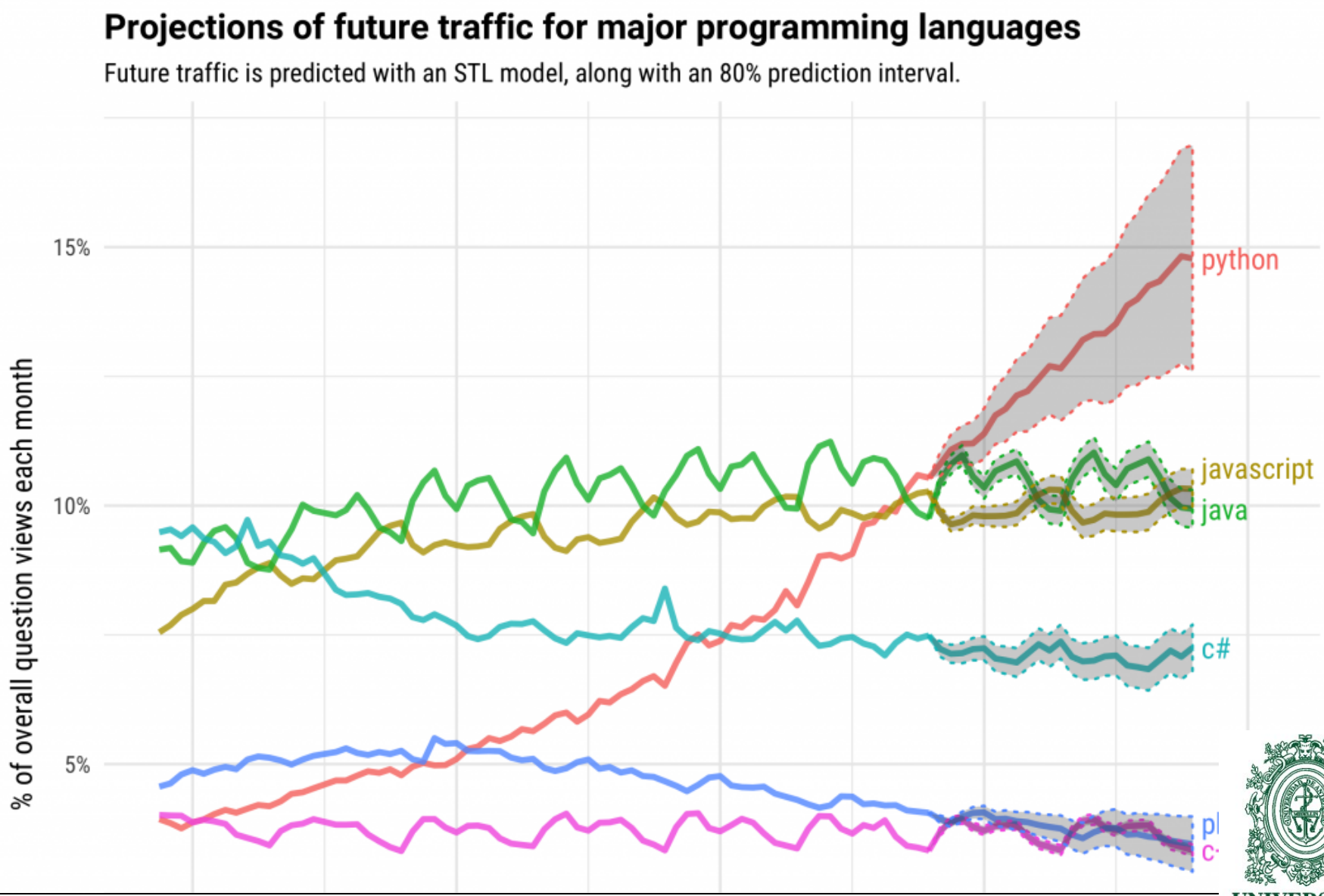
Why Python



[1]Source: IEEE Spectrum 2017: http://spectrum.ieee.org/computing/software/the-2017-top-programming-

UNIVERSIDAD DE ANTIOQUIA
1803

Why python


Projections of future traffic for major programming languages
Future traffic is predicted with an STL model, along with an 80% prediction interval.
% of overall question views each month
15%
10%
5%
python
javascript
java
c#

Why python

ANACONDA®

UNIVERSIDAD DE ANTIOQUIA
1803

Python3

A (very incomplete) list of properties Python

- Multi-paradigm programming language (e.g. object-oriented)
- Garbage collection for memory management
- The intention is to have highly readable code
- Indentation instead of (curly) braces
- Dynamic typing
- Data types / structures

Lists	`a = []`	Tuple	`b = ()`
Dictionaries	`d = {'one':1, 'two':2}`	Strings	`str = "hi"`
Integer	`i = 3`	Float	`j = 1.5`

  Type verification: `type(i)`

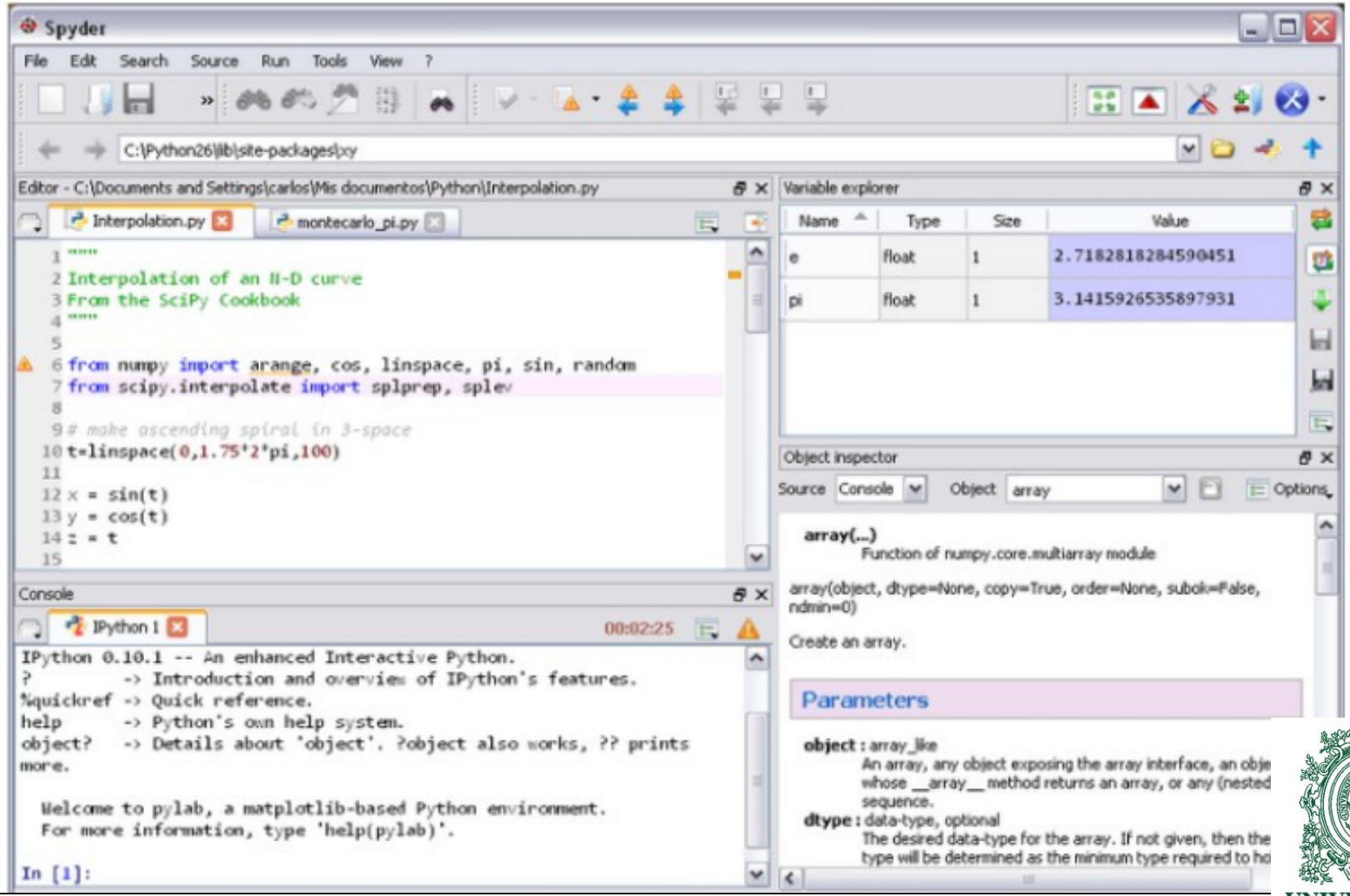
- Due to Python's popularity there are numerous libraries available for everything one can think of.

Why python

UNIVERSIDAD DE ANTIOQUIA
1803

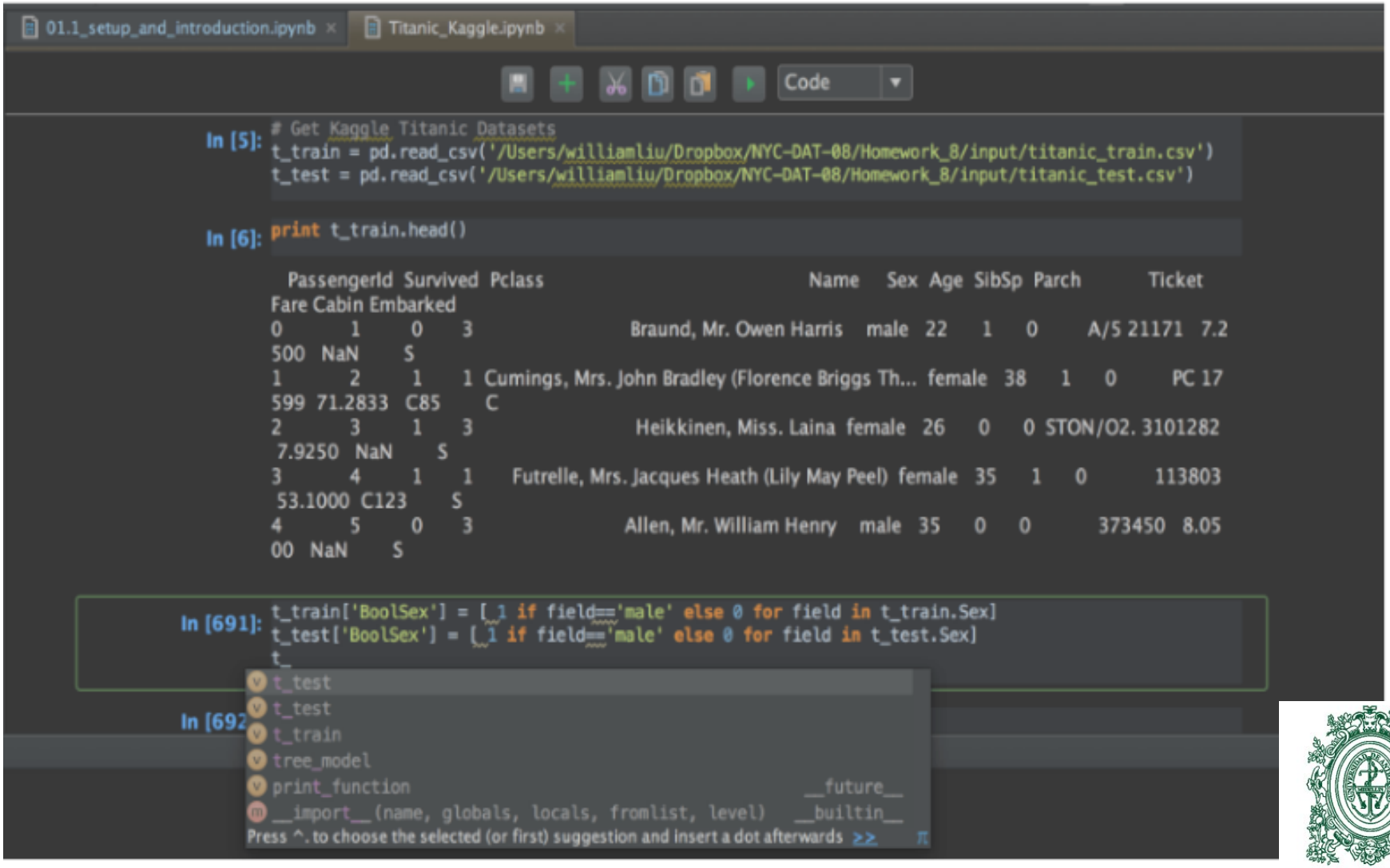
Why python

Spyder



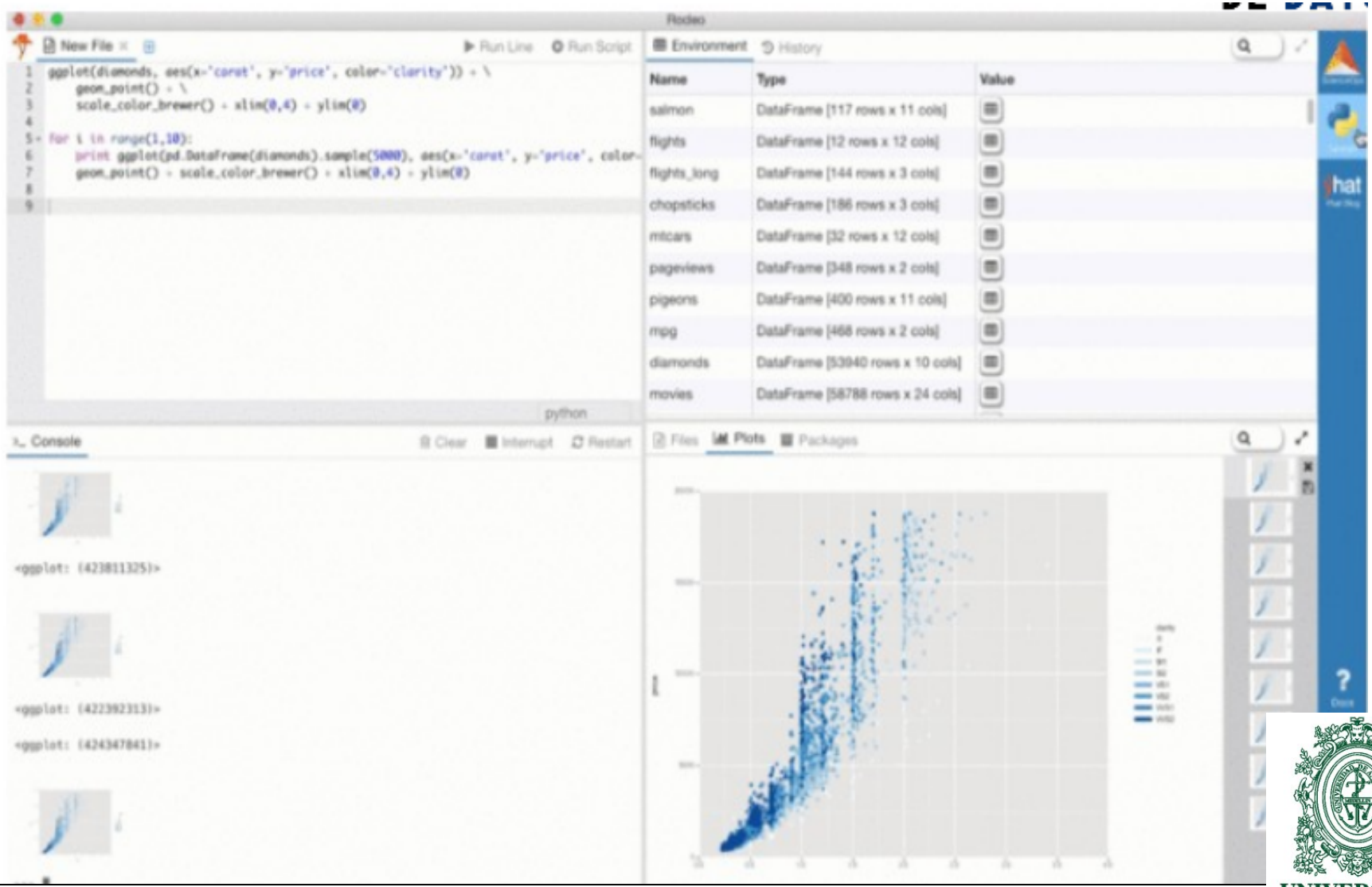
Why python

Pycharm



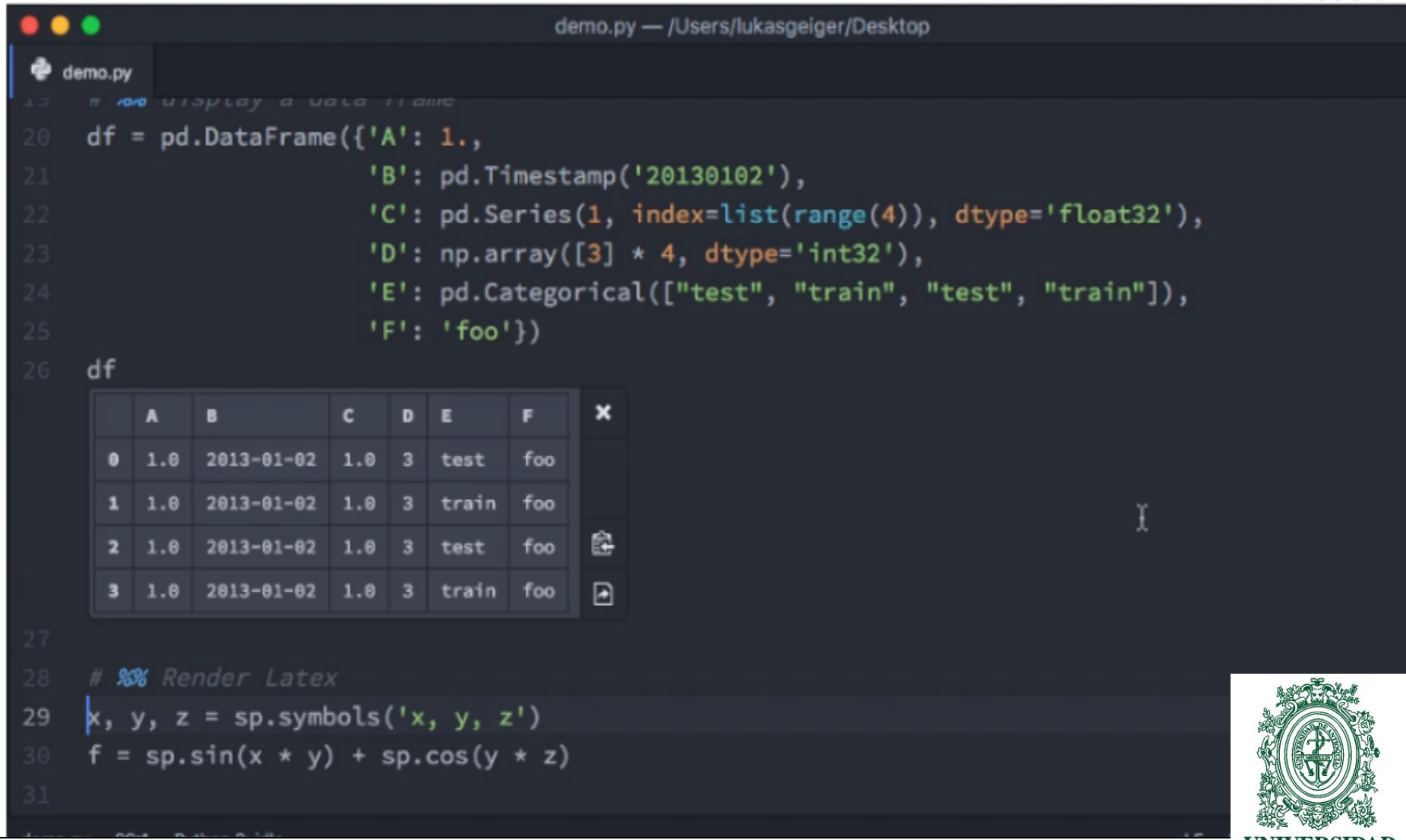
Why python

Rodeo



Why python

Atom



UNIVERSIDAD DE ANTIOQUIA
1803

Why python

IP[y]: Notebook spectrogram Last Checkpoint: a few seconds ago (autosaved)

IPython (Python 3)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

Simple spectral analysis

An illustration of the Discrete Fourier Transform using windowing, to reveal the frequency content of a sound signal.

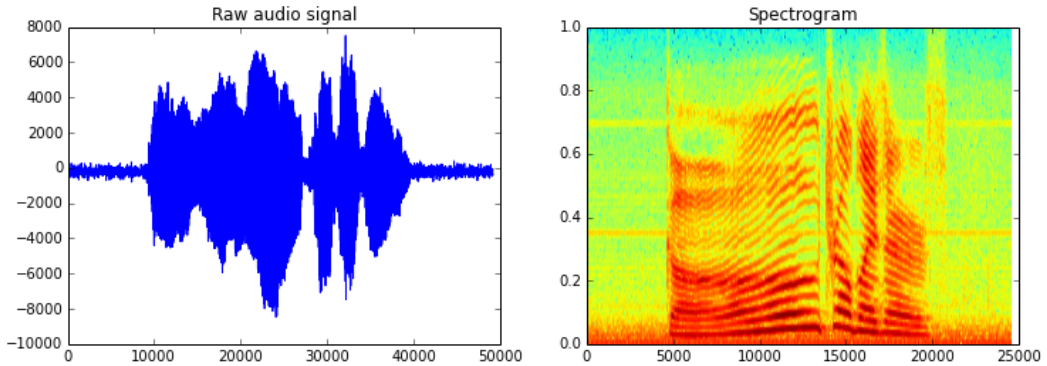
$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

In [1]:
```
from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin `specgram` routine:

In [2]:
```
%matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```



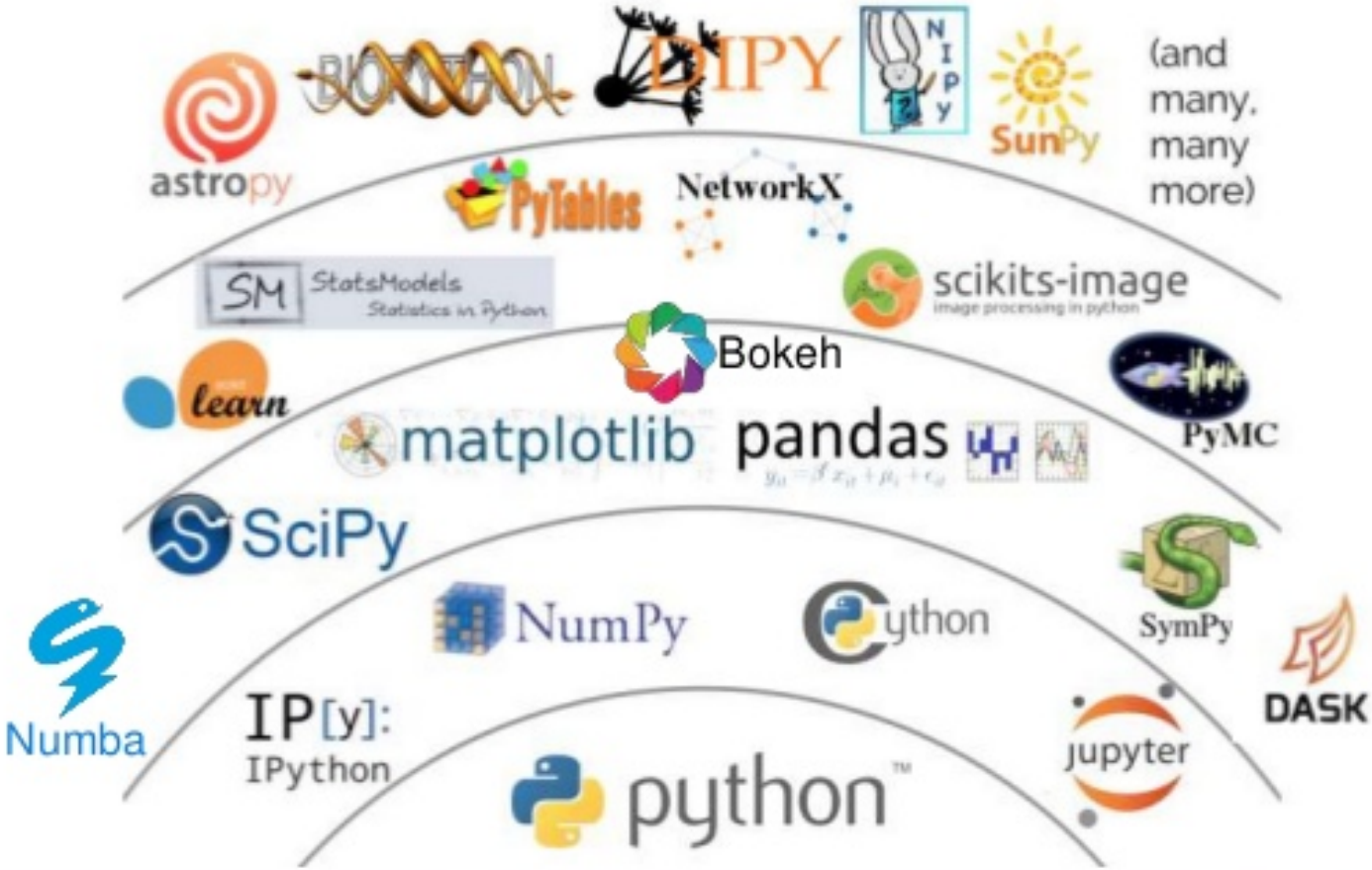
Jupyter

UNIVERSIDAD DE ANTIOQUIA 1803

Próxima sección

UNIVERSIDAD DE ANTIOQUIA
1803

Tools



UNIVERSIDAD DE ANTIOQUIA
1803

Tools

- Jupyter
- Numpy
- Pandas
- Matplotlib
- Scipy

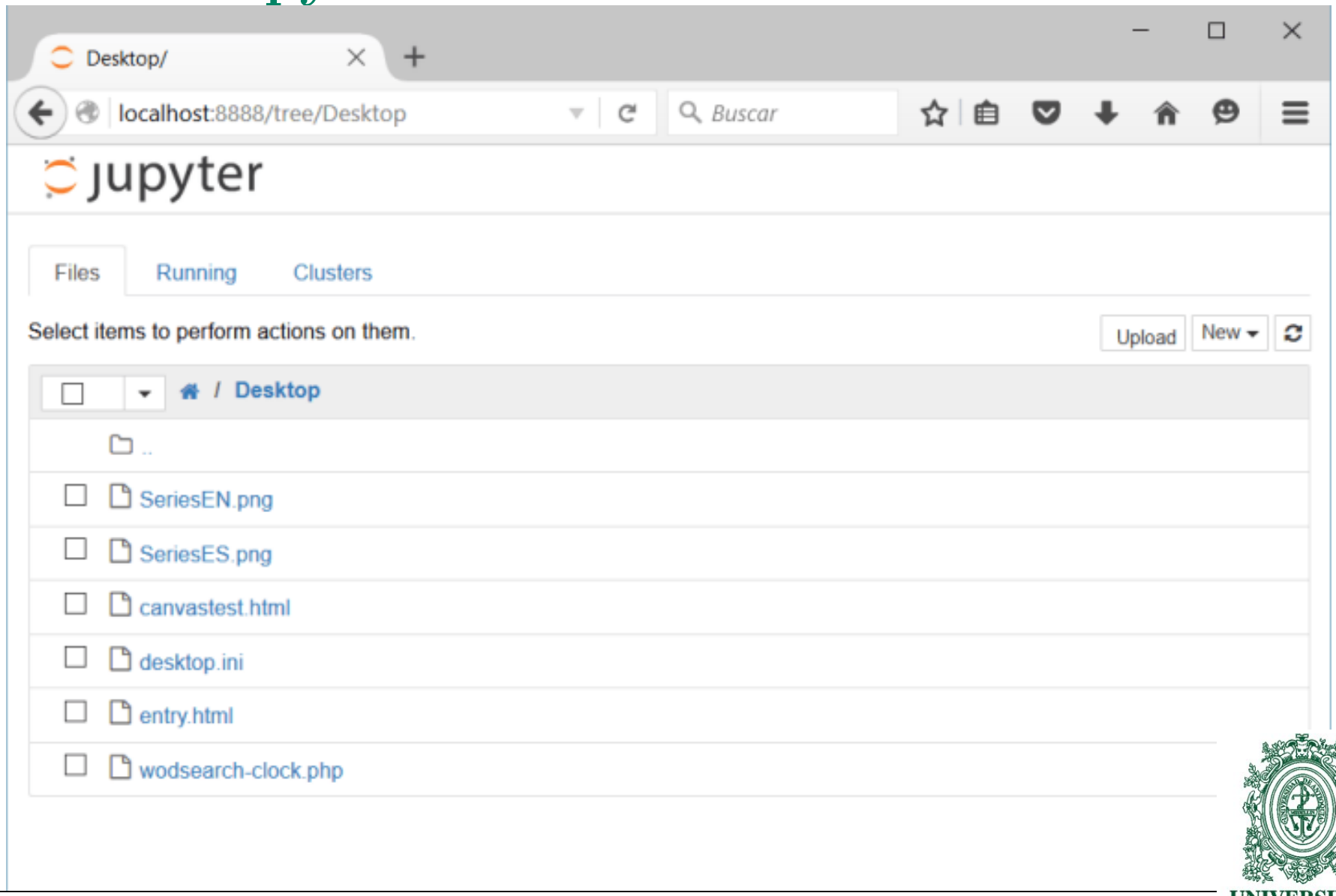
Tools: Jupyter

Interactive environment to create documents, code, interactive widgets, graphs, texts and equations.

[2]

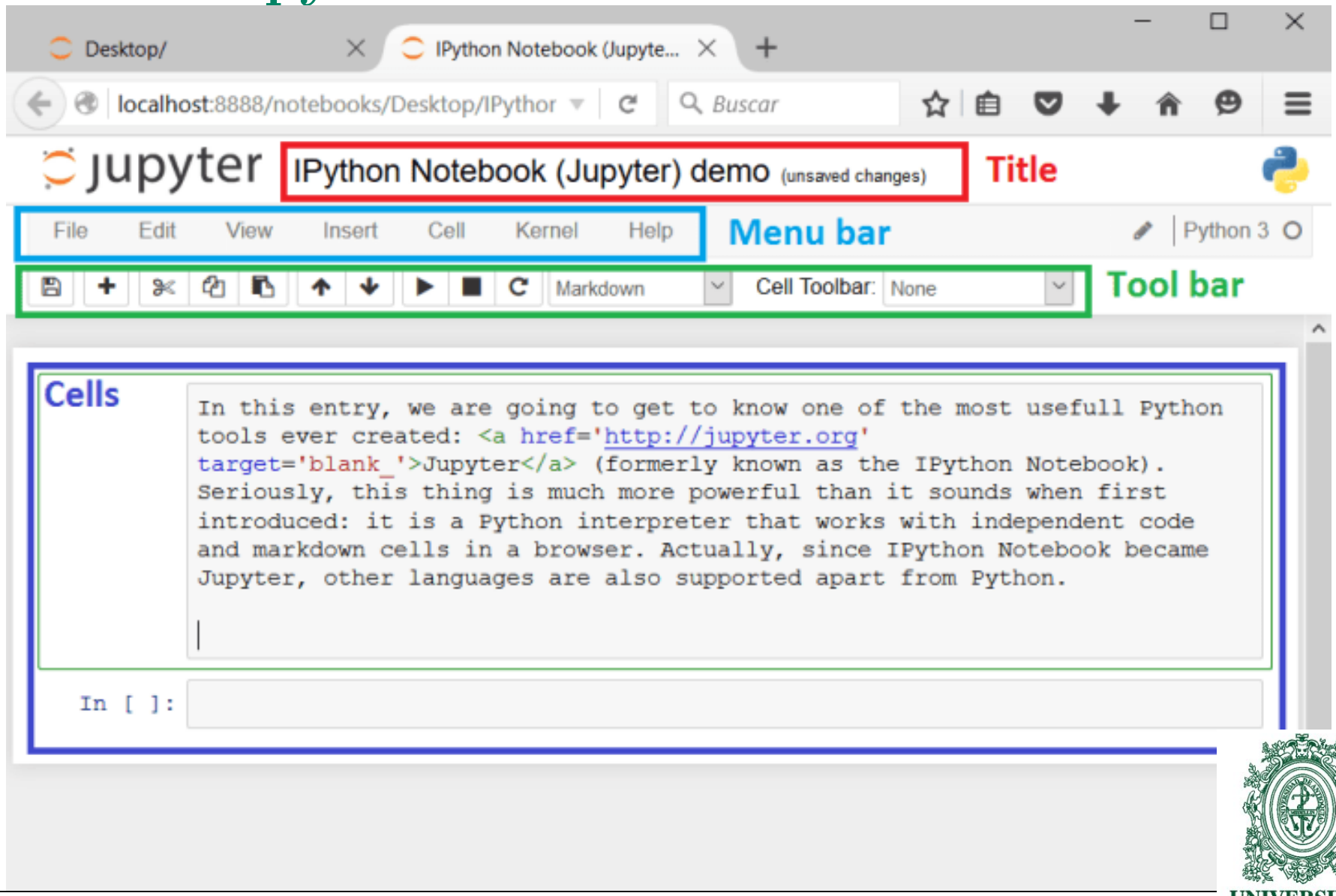
[2] http://jupyter.org/

Tools: Jupyter



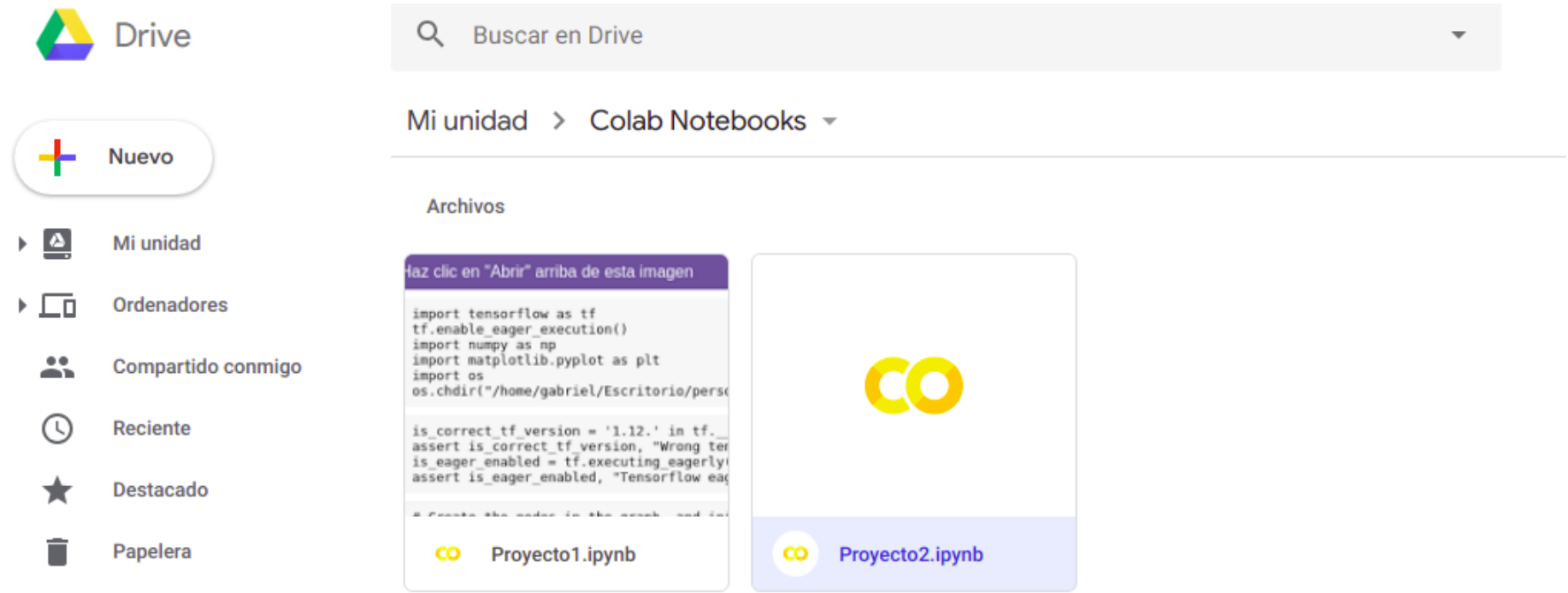
UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Jupyter



UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Google Colab

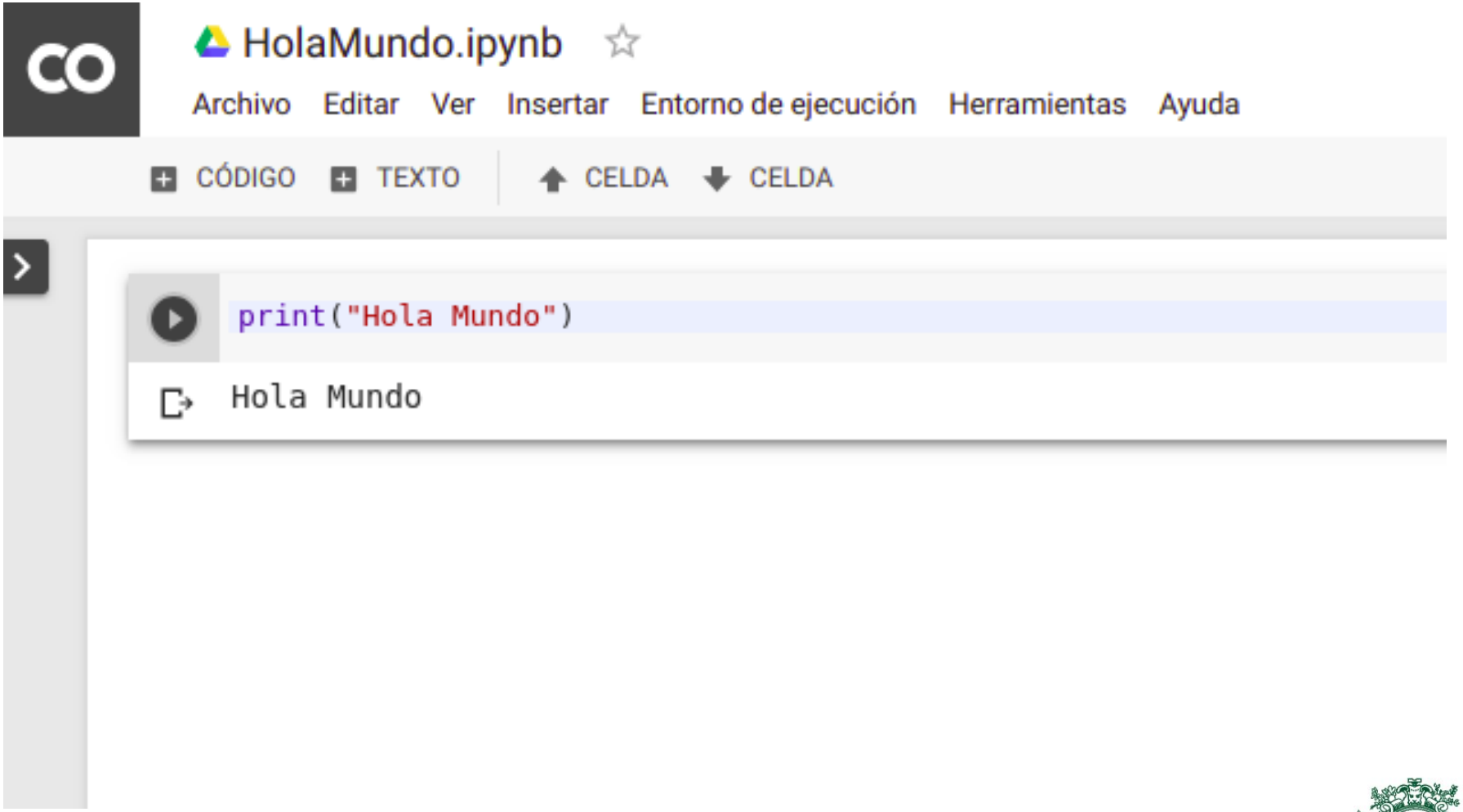


5

[5] https://colab.research.google.com/notebooks/welcome.ipynb

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Google Colab



6

[6] https://colab.research.google.com/notebooks/welcome.ipynb

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: List processing

- Lists are just like the arrays, declared in other languages.
- Lists need not be homogeneous always which makes it a most powerful tool in Python.
- A single list may contain DataTypes like Integers, Strings, as well as Objects.
- Lists are also very useful for implementing stacks and queues.

Tools: List processing

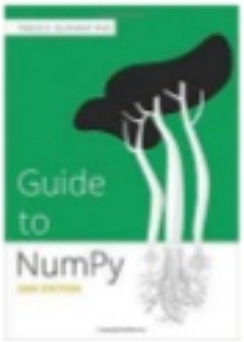
```
1  # Creating a List
2  List = []
3  print("Intial blank List: ")
4  print(List)
5
6  # Creating a List with
7  # the use of a String
8  List = ['GeeksForGeeks']
9  print("\nList with the use of String: ")
10 print(List)
11
12 # Creating a List with
13 # the use of multiple values
14 List = ["Geeks", "For", "Geeks"]
15 print("\nList containing multiple values: ")
16 print(List[0])
17 print(List[2])
```

Tools: List processing

```
1  # Creating a List with
2  # the use of Numbers
3  # (Having duplicate values)
4  List = [1, 2, 4, 4, 3, 3, 3, 6, 5]
5  print("\nList with the use of Numbers: ")
6  print(List)
7
8  # Creating a List with
9  # mixed type of values
10 # (Having numbers and strings)
11 List = [1, 2, 'Geeks', 4, 'For', 6, 'Geeks']
12 print("\nList with the use of Mixed Values: ")
13 print(List)
```

Tools: Numpy

NumPy: an Array Extension of Python



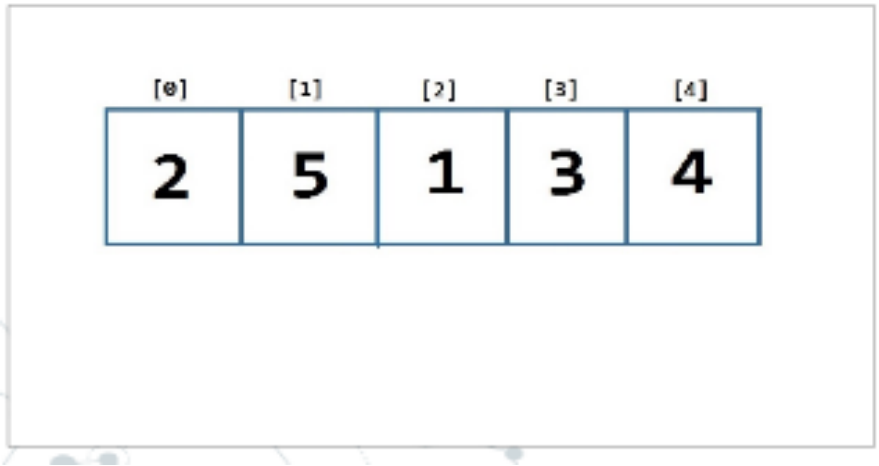
- Data: the array object
  - slicing and shaping
  - data-type map to Bytes
- Fast Math (ufuncs):
  - vectorization
  - broadcasting
  - aggregations

ANACONDA

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Numpy

- Array, matrix, vectorial operations.
- Import numpy as np



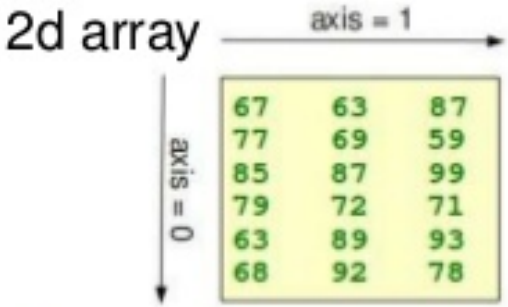
NumPy

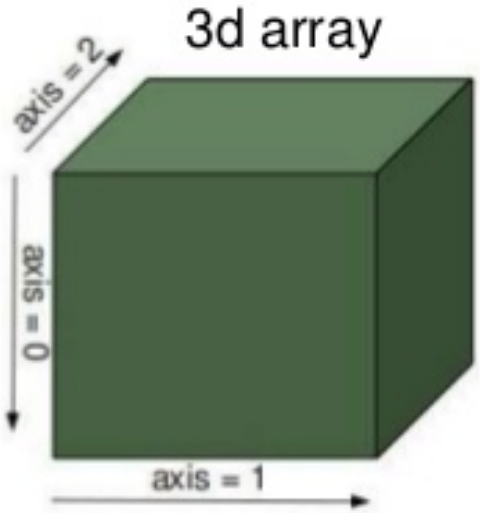
7

[7] http://www.numpy.org/

Tools: Numpy

NumPy Examples






```
1  import numpy as np
2
3  x = np.array([[67, 63, 87],
4                [77, 69, 59],
5                [85, 87, 89],
6                [79, 72, 71],
7                [63, 89, 93],
8                [68, 92, 78]])
9  print x.sum(axis=0), x.sum(axis=1)
```

```
[439 472 477]
[217 205 261 222 245 238]
```

```
12  y = 3*np.random.randn(10,20,30)+10
13  print y.mean(), y.std()
```

```
9.98330639789 2.96677717122
```

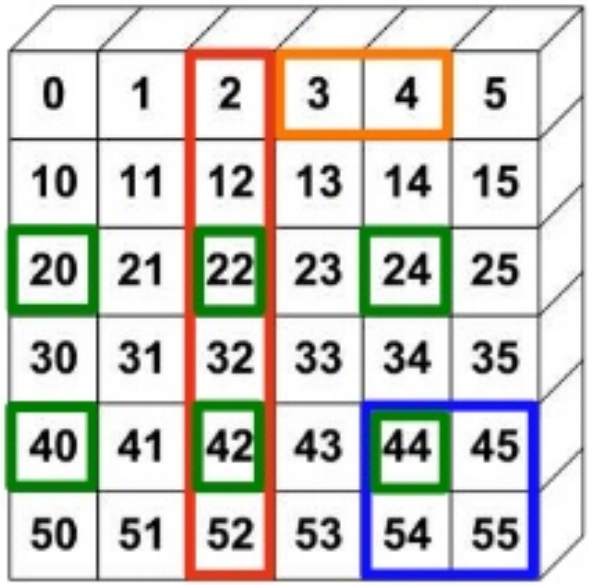
ANACONDA

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Numpy

NumPy Slicing (Selection)

```
>>> a[0,3:5]
array([3, 4])

>>> a[4:,4:]
array([[44, 45],
       [54, 55]])

>>> a[:,2]
array([2,12,22,32,42,52])

>>> a[2::2,::2]
array([[20, 22, 24],
       [40, 42, 44]])
```



ANACONDA

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Numpy

```
1  # manejo de arreglos
2
3  import numpy as np #libreria estandar para manejo de datos y senales en Python
4
5  a = np.zeros((2,2))  # Crea un arreglo tipo matriz de ceros
6  print(a)              # Prints "[[ 0.  0.]
7                        #          [ 0.  0.]]"
8  b = np.ones((1,2))   # Crea un arreglo tipo vector fila de unos
9  print(b)             # Prints "[[ 1.  1.]]"
10
11 c = np.full((2,2), 7) # Crea un arreglo tipo matriz de un valor constante
12 print(c)               # Prints "[[ 7.  7.]
13                       #          [ 7.  7.]]"
14 c = 7*np.ones((2,2)) # Otra forma
15 print (c)
16
17 d = np.eye(2)         # Crea un arreglo con la matriz identidad
```

Tools: Numpy

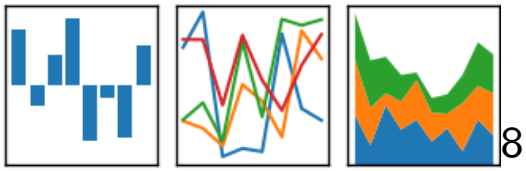
```
1  e = np.random.random((2,2)) # Crea un arreglo tipo matriz de numeros aleatorios entre 0 y 1
2  print (e)          # print "[[ 0.91940167  0.08143941]
3                     #        [ 0.68744134  0.87236687]]"
4  print (e[0,1])                  # Seleccionar un elemento
5  print (e[0,:])                 # Seleccionar la primera fila
6  print (e[:,1])                  # Seleccionar la segunda columna
7
8  xrang=np.arange(20) # Crea un arreglo de numeros consecutivos entre 0 y 19
9  print (xrang)            # print "[0 1 2 3 4 5... 19]"
10 # Para seleccionar partes del arreglo use los siguientes comandos
11 print (xrang[0:5])      # print "[0 1 2 3 4]"
12 print (xrang[10])       # print [10]
13
14 A=np.random.random(100)
15 media=np.mean(A) # calcula la media de un arreglo A
16 maxA=np.max(A) # calcula el maximo valor de un arreglo
17 minA=np.min(A) #calcula el minimo valor de un arreglo
```

Tools: Pandas

Python has long been great for data munging and preparation, but less so for data analysis and modeling. pandas helps fill this gap, enabling you to carry out your entire data analysis workflow in Python

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



[8]

[8] https://pandas.pydata.org/

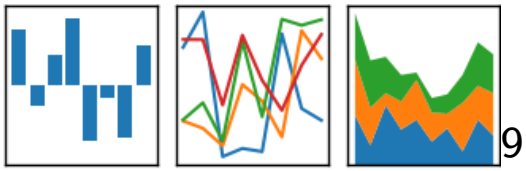
Tools: Pandas

Data structure and tools for data analysis.

- DataFrames
- GroupBy, merge, join
- Data reading and writing
- Import pandas as pd

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



[9]

[9] https://pandas.pydata.org/

UNIVERSIDAD DE ANTIOQUIA
1803

Tools: Pandas

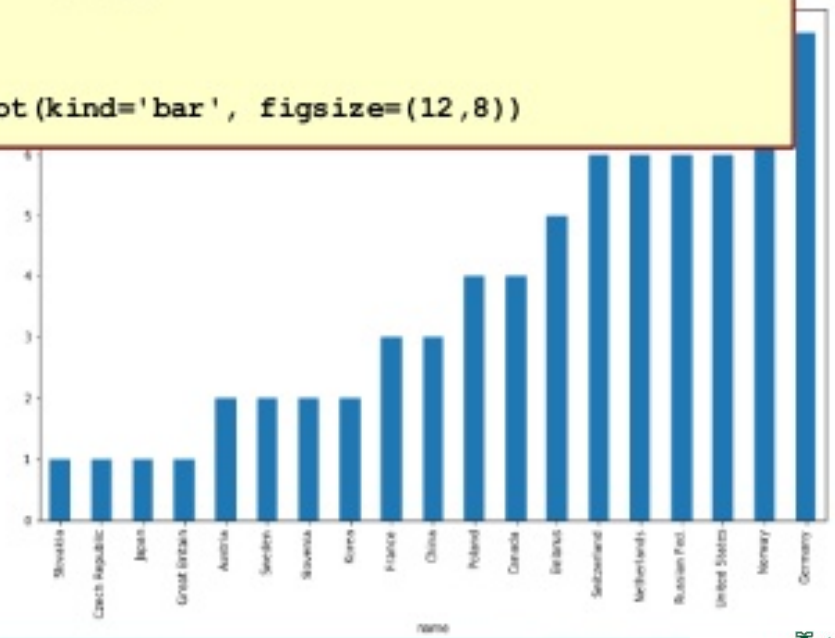
pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$

Easy Data Wrangling

```
medals = pd.read_csv('data/medals.csv', index_col='name')
medals.head()
gold = medals['medal'] == 'gold'
won = medals['count'] > 0
medals.loc[gold & won, 'count'].sort_values().plot(kind='bar', figsize=(12,8))
```

	count	medal	country
name			
Australia	1	bronze	AUS
Australia	2	silver	AUS
Australia	0	gold	AUS
Austria	1	bronze	AUT
Austria	6	silver	AUT



ANACONDA

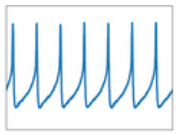
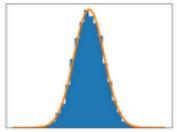
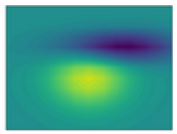
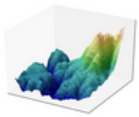
UNIVERSIDAD DE ANTIOQUIA 1803

Tools: Matplotlib

Graphics in Python

- Frequency diagram
- Histograms
- Scatter-plots
- 3D graphics
- Surface

Import matplotlib

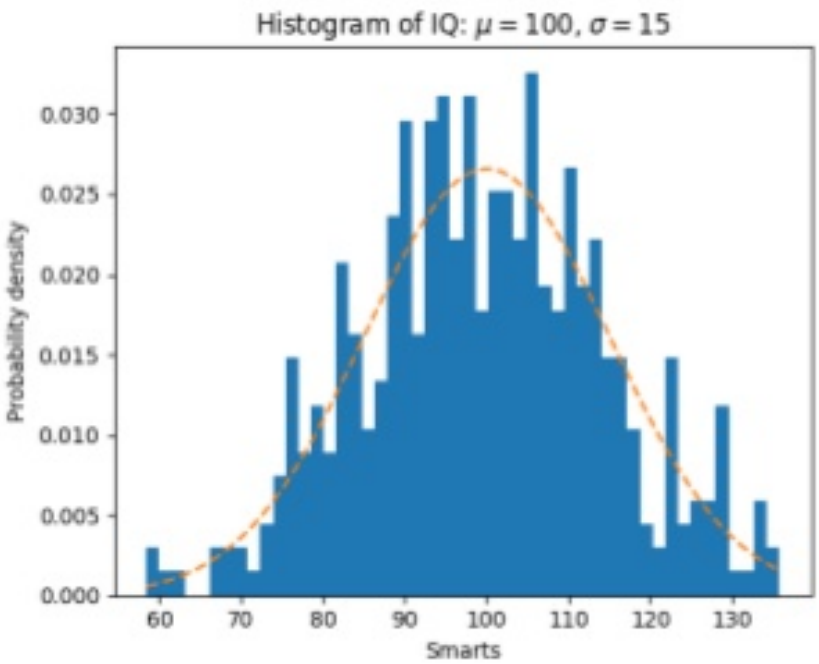
10

[10] https://matplotlib.org/

Tools: Matplotlib

matplotlib a powerful plotting engine

```
import numpy as np
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt

np.random.seed(0)

# example data
mu = 100  # mean of distribution
sigma = 15  # standard deviation of distribution
x = mu + sigma * np.random.randn(437)

num_bins = 50

fig, ax = plt.subplots()

# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, normed=1)

# add a 'best fit' line
y = mlab.normpdf(bins, mu, sigma)
ax.plot(bins, y, '--')
ax.set_xlabel('Smarts')
ax.set_ylabel('Probability density')
ax.set_title(r'Histogram of IQ: $\mu=100$,
$\sigma=15$')

# Tweak spacing to prevent clipping of ylabel
fig.tight_layout()
plt.show()
```



ANACONDA

UNIVERSIDAD DE ANTIOQUIA 1803

Tools: Matplotlib

matplotlib a powerful plotting engine

```
import matplotlib.pyplot as plt
import numpy as np

with plt.xkcd():
    fig = plt.figure()
    ax = fig.add_axes((0.1, 0.2, 0.8, 0.7))
    ax.spines['right'].set_color('none')
    ax.spines['top'].set_color('none')
    plt.xticks([])
    plt.yticks([])
    ax.set_ylim([-30, 10])
    data = np.ones(100)
    data[70:] -= np.arange(30)

    plt.annotate('THE DAY I REALIZED\nI COULD
COOK BACON\nWHENEVER I WANTED',
        xy=(70, 1),
        arrowprops=dict(arrowstyle='->'),
        xytext=(15, -10))

    plt.plot(data)

    plt.xlabel('time')
    plt.ylabel('my overall health')
    fig.text(0.5, 0.05, '"Stove Ownership" from
xkcd by Randall Monroe',
        ha='center')
```

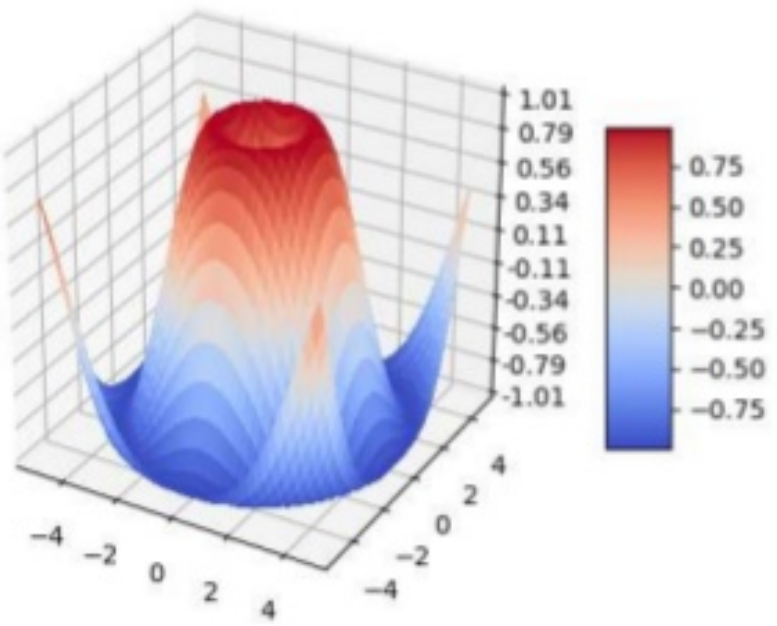
"STOVE OWNERSHIP" FROM XKCD BY RANDALL MONROE

UNIVERSIDAD DE ANTIOQUIA 1803

Tools: Matplotlib

matplotlib a powerful plotting engine

```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator,
FormatStrFormatter
import numpy as np

fig = plt.figure()
ax = fig.gca(projection='3d')
# Make data.
X = np.arange(-5, 5, 0.25)
Y = np.arange(-5, 5, 0.25)
X, Y = np.meshgrid(X, Y)
R = np.sqrt(X**2 + Y**2)
Z = np.sin(R)
# Plot the surface.
surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,
linewidth=0, antialiased=False)

# Customize the z axis.
ax.set_zlim(-1.01, 1.01)
ax.zaxis.set_major_locator(LinearLocator(10))
ax.zaxis.set_major_formatter(FormatStrFormatter('%
.02f'))

# Add a color bar which maps values to colors.
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.show()
```

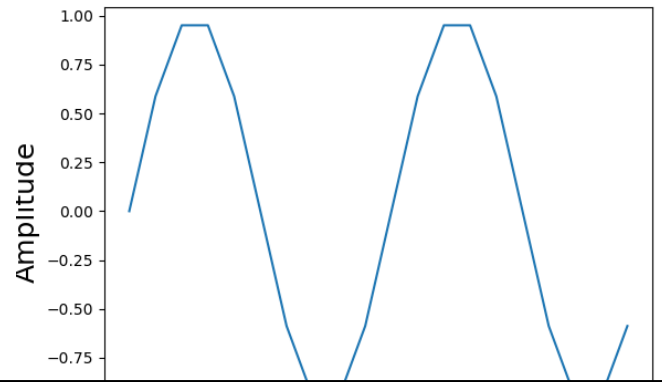


ANACONDA

UNIVERSIDAD DE ANTIOQUIA 1803

Tools: Matplotlib

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  %matplotlib inline
4  f=1.0 # Signal frequency
5  fs=10.0 # Sampling frequency
6  t=np.arange(0, 2.0, 1.0/fs) # Vector de tiempo
7  x = np.sin(2*np.pi*f*t)
8  plt.plot(t,x)
9  plt.xlabel('Time',fontsize=18)
10 plt.ylabel('Amplitude',fontsize=18)
11 plt.show()
```



UNIVERSIDAD DE ANTIOQUIA
1803

Dash

Build beautiful web-based interfaces in Python

- Dash is a Python framework for building analytical web applications. No JavaScript required.
- Equivalent R-Shiny for python

12

[12] https://plot.ly/products/dash/

UNIVERSIDAD DE ANTIOQUIA
1803

File managment

Load Text files containing vector or matrices data.

```
1 import numpy as np
2 data=np.loadtxt("file.txt")
```

Write a Text file with vector or matrices data.

```
1 Data=np.random.rand((100,20))
2 np.savetxt("file.txt", Data)
```

UNIVERSIDAD DE ANTIOQUIA
1803

File managment

Loading and plotting a Text file that contains a signal.

```
1    data=np.loadtxt("PrecioDolar.txt")
2    signal=data[:,3]
3    t=np.arange(len(signal))
4    plt.plot(t,signal)
5    plt.xlabel("samples")
6    plt.ylabel("TRM COP -US")
7    plt.show()
```

```
1    Ts=1.0/365
2    t=np.arange(1992, 2016+244*Ts+5*Ts, Ts)
3    plt.plot(t,signal)
4    plt.xlabel("Fecha")
5    plt.ylabel("TRM COP-US")
6    plt.show()
```

File managment

Load data from excel files (.csv)

```
1    import pandas as pd
2    import matplotlib.pyplot as plt
3    data=pd.read_csv("DataSiata.csv")
4    signal=data["pm10"]
5    t=data["index"]
6    plt.plot(t,signal)
7    plt.xlabel("samples")
8    plt.ylabel("Concentracion particulas PM10")
9    plt.show()
```

File managment

Load audio signals (.wav)

```
1      from scipy.io.wavfile import read
2      import matplotlib.pyplot as plt
3      fs,signal=read("098_readtext_PCGITA.wav")
4      t=np.arange(0,len(signal)/fs, 1./fs)
5      plt.plot(t,signal)
6      plt.xlabel("Time [s]")
7      plt.ylabel("Amplitude")
8      plt.show()
```

File managment

Write signal into .wav filee

```
1  from scipy.io.wavfile import write
2  import numpy as np
3
4  fs=44100
5  t=np.arange(0,10,1.0/fs)
6  signal=np.sin(2*np.pi*fs*(t**3))
7
8  signal2=np.asarray(signal*2**15, dtype=np.int16)
9  write("signal.wav", fs, signal2)
```

Functions

If a sequence of commands is used more often, a function is helpful:

```
1    def say_hello():
2        print("hello")
```

```
1    def say_hello_to(name):
2        print("hello " + name)
```

Call the function like this:

```
1    say_hello()
2    say_hello_to("Donald")
```

Functions

```
1     def energy(signal):
2         s2=signal**2
3         energy=sum(s2)
4         return(energy)
```

Call the function like this:

```
1     signal=[2,4,6,0,0,1,2,4,0]
2     E=energy(signal)
3     print("Energy="+str(E))
```

Important: Make sure you get the indentation right, especially when you do cut and paste (so your code works).

Próxima sección

UNIVERSIDAD DE ANTIOQUIA
1803

Let's code

Terminal: **jupyter notebook**

Preguntas

Thanks!

jcamilo.vasquez@udea.edu.co

UNIVERSIDAD DE ANTIOQUIA
1803