

Scientific Programming with Python

Pedro Magaña (pmagana@ugr.es)

Outline

- Why learn to code?
- Introduction to Python
- Python for science, where to begin?
- Python language
- Scientific libraries

Why learn to code?

Start



```
for i in people.data.users:
    response = client.api.statuses.user_timeline.get(screen_name=i.screen_name)
    print 'Got', len(response.data), 'tweets from', i.screen_name
    if len(response.data) != 0:
        ltdate = response.data[0]['created_at']
        ltdate2 = datetime.strptime(ltdate, '%a %b %d %H:%M:%S +0000 %Y')
        today = datetime.now()
        howlong = (today-ltdate2).days
        if howlong < daywindow:
            print i.screen_name, 'has tweeted in the past', daywindow,
            totaltweets += len(response.data)
            for j in response.data:
                if j.entities.urls:
                    for k in j.entities.urls:
                        newurl = k['expanded_url']
                        urlset.add((newurl, j.user.screen_name))
        else:
            print i.screen_name, 'has not tweeted in the past', daywindow
```


Apple CEO Tim Cook: Learn to code, it's more important than English as a second language





Catherine Clifford | 12:58 PM ET Thu, 12 Oct 2017
























IEEE Spectrum | Trending | **Jobs** | Open | Custom

[Edit Ranking](#) | [Add a Comparison](#) |  

(click to hide)

 Web |  Mobile |  Enterprise |  Embedded

Rank	Language	Job Ranking	Score
1.	Java	  	100.0
2.	C	  	99.1
3.	Python	 	95.8
4.	C++	  	95.7
5.	C#	  	91.9
6.	JavaScript	 	90.7
7.	PHP		86.6
8.	SQL		85.0
9.	Ruby	 	83.6
10.	Shell		79.1

Python Comparision to other language

To Display "Hello World"

"Hello World!" Program in Python

```
print("Hello World!")
```

"Hello World!" Program in C

```
#include <stdio.h>
int main( )
{
printf("Hello World !");
return 0 ;
}
```

"Hello World!" Program in C++

```
#include <iostream>
using namespace std ;
int main( )
{
cout << "Hello World !";
return 0 ;
}
```

"Hello World!" Program in Java

```
public class HelloWorld {
public static void main(Strings[ ] args) {
System.out.println("Hello World !");
}
}
```


TOOLBOX

PICK UP PYTHON

A powerful programming language with huge community support.

ILLUSTRATION BY THE PROJECT TWINS



BY JEFFREY M. PERKEL

Last month, Adina Howe took up a post at Iowa State University in Ames. Officially, she is an assistant professor of agricultural and biosystems engineering. But she works not in the greenhouse, but in front of a keyboard. Howe is a programmer, and a key part of her job is as a 'data professor' — developing curricula to teach the next generation of graduates about the mechanics and importance of scientific programming.

Howe does not have a degree in computer science, nor does she have years of formal training. She had a PhD in environmental engineering and expertise in running enzyme assays when she joined the laboratory of Titus Brown at Michigan State University in East Lansing.

Brown specializes in bioinformatics and uses computation to extract meaning from genomic data sets, and Howe had to get up to speed on the computational side. Brown's recommendation: learn Python.

Among the host of computer-programming languages that scientists might choose to pick up, Python, first released in 1991 by Dutch programmer Guido van Rossum, is an increasingly popular (and free) recommendation. It combines simple syntax, abundant online resources and a rich ecosystem of scientifically focused toolkits with a heavy emphasis on community.

HELLO, WORLD

With the explosive growth of 'big data' in disciplines such as bioinformatics, neuroscience and astronomy, programming know-how

is becoming ever more crucial. Researchers who can write code in Python can deftly manage their data sets, and work much more efficiently on a whole host of research-related tasks — from crunching numbers to cleaning up, analysing and visualizing data. Whereas some programming languages, such as MATLAB and R, focus on mathematical and statistical operations, Python is a general-purpose language, along the lines of C and C++ (the languages in which much commercial software and operating systems are written). As such, it is perhaps more complicated, Brown says, but also more capable: it is amenable to everything from automating small sets of instructions, to building websites, to fully fledged applications. Jessica Hamrick, a psychology PhD student at the University of California, Berkeley, has been ►

Programming: Pick up Python

A powerful programming language with huge community support.

4 February 2015

Nature

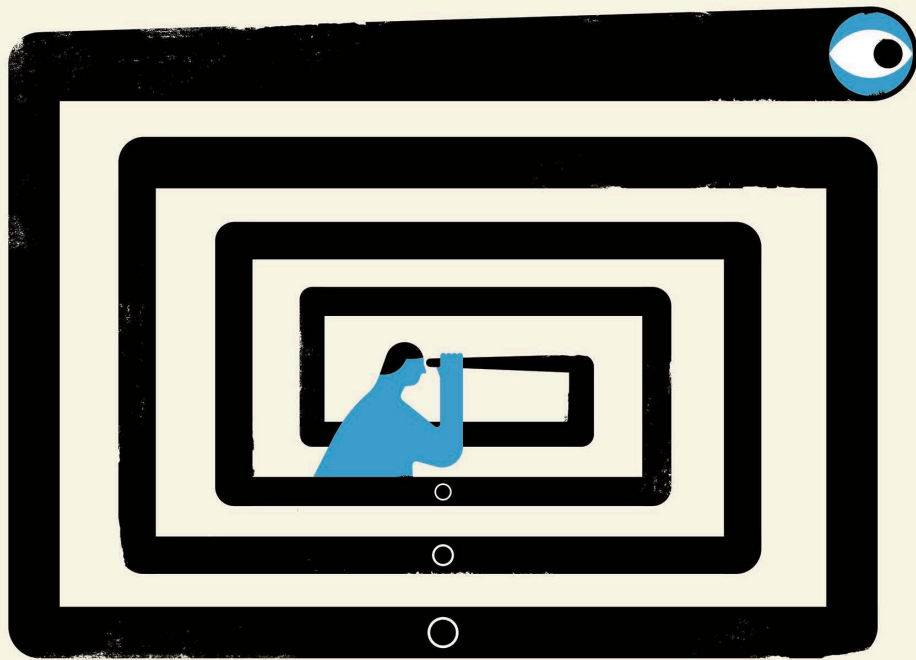
<http://doi.org/10.1038/518125a>

TOOLBOX

BY JUPYTER, IT ALL MAKES SENSE

Thanks to an improved interface and tools for running in the cloud, take-up of this computational notebook is rocketing.

ILLUSTRATION BY THE PROJECT TWINS



BY JEFFREY M. PERKEL

Perched atop the Cerro Pachón ridge in the Chilean Andes is a building site that will eventually become the Large Synoptic Survey Telescope (LSST). When it comes online in 2022, the telescope will generate terabytes of data each night as it surveys the southern skies automatically. And to crunch those data, astronomers will use a familiar and increasingly popular tool: the Jupyter notebook.

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single

document. Computational notebooks have been around for decades, but Jupyter in particular has exploded in popularity over the past couple of years. This rapid uptake has been aided by an enthusiastic community of user-developers and a redesigned architecture that allows the notebook to speak dozens of programming languages — a fact reflected in its name, which was inspired, according to co-founder Fernando Pérez, by the programming languages Julia (Ju), Python (Py) and R.

One analysis of the code-sharing site GitHub counted more than 2.5 million public Jupyter notebooks in September 2018, up from 200,000 or so in 2015. In part, says Pérez, that growth is due to improvements in the web

software that drives applications such as Gmail and Google Docs; the maturation of scientific Python and data science; and, especially, the ease with which notebooks facilitate access to remote data that might otherwise be impractical to download — such as from the LSST. “In many cases, it’s much easier to move the computer to the data than the data to the computer,” says Pérez of Jupyter’s cloud-based capabilities. “What this architecture helps to do is to say, you tell me where your data is, and I’ll give you a computer right there.”

For data scientists, Jupyter has emerged as a de facto standard, says Lorena Barba, a mechanical and aeronautical engineer at George Washington University ▶

By Jupyter, it all makes sense

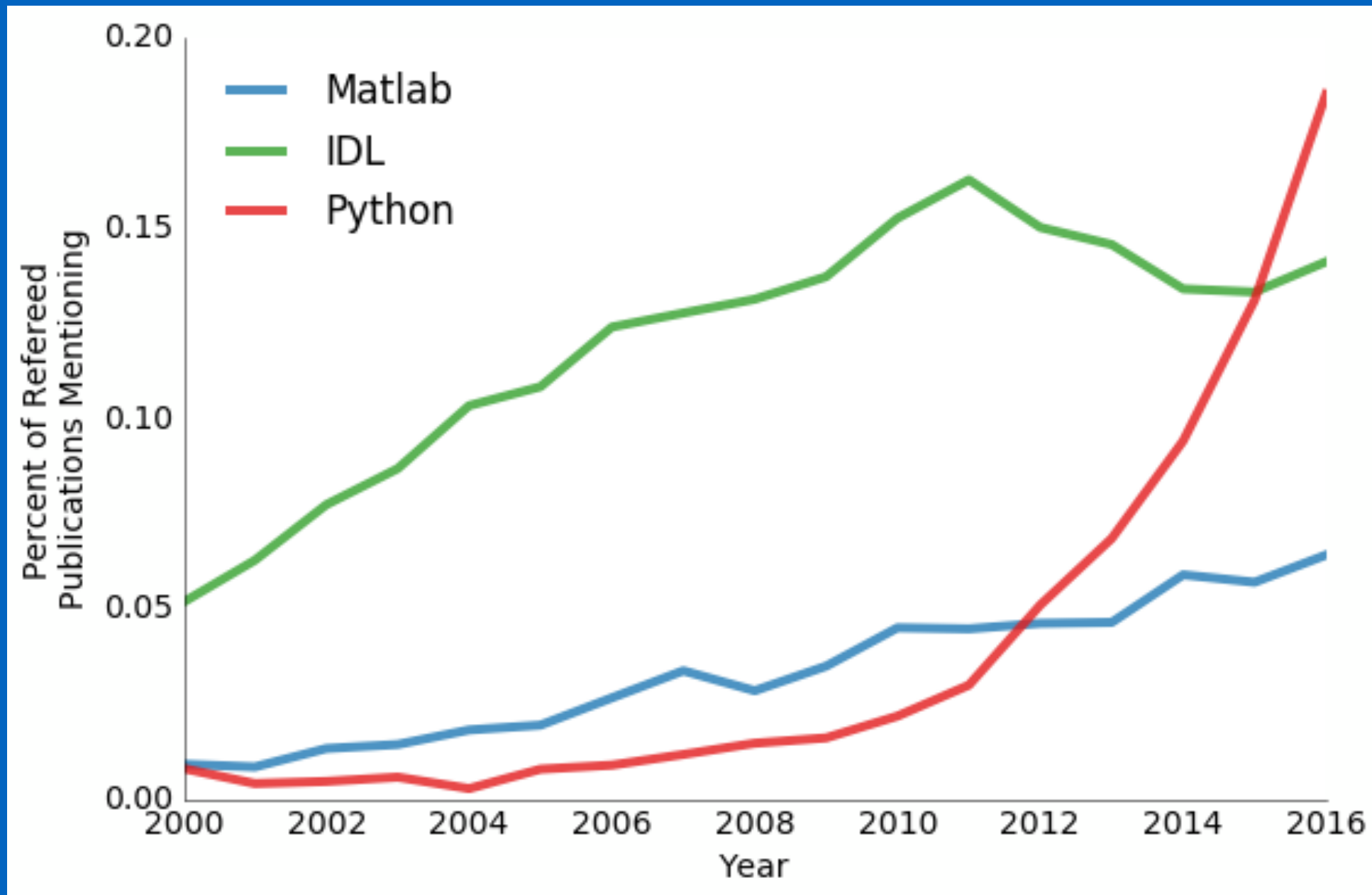
Why Jupyter is data scientists’ computational notebook of choice

1 November 2018

Nature

<http://doi.org/10.1038/d41586-018-07196-1>

Counting programming language mentions in astronomy papers (2015)



Programming languages

Python has brought computer programming to a vast new audience

And its inventor has just stepped down

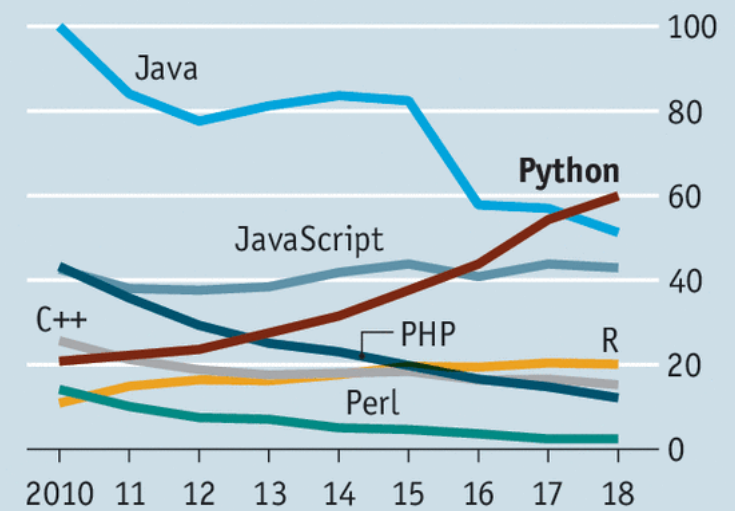


Print edition | Science and technology >
Jul 19th 2018



Biggus uptickus

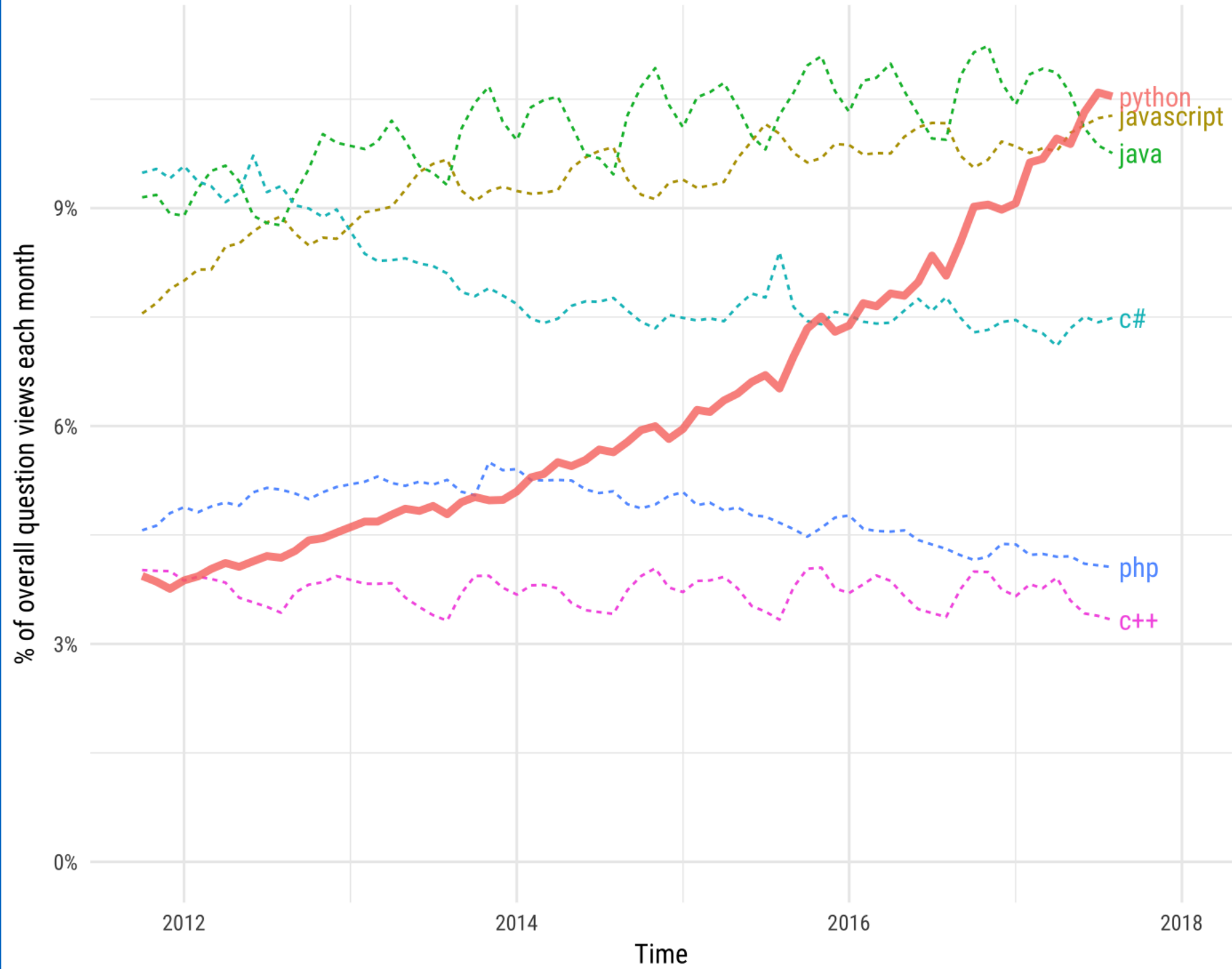
US, Google searches for coding languages
100=highest annual traffic for any language



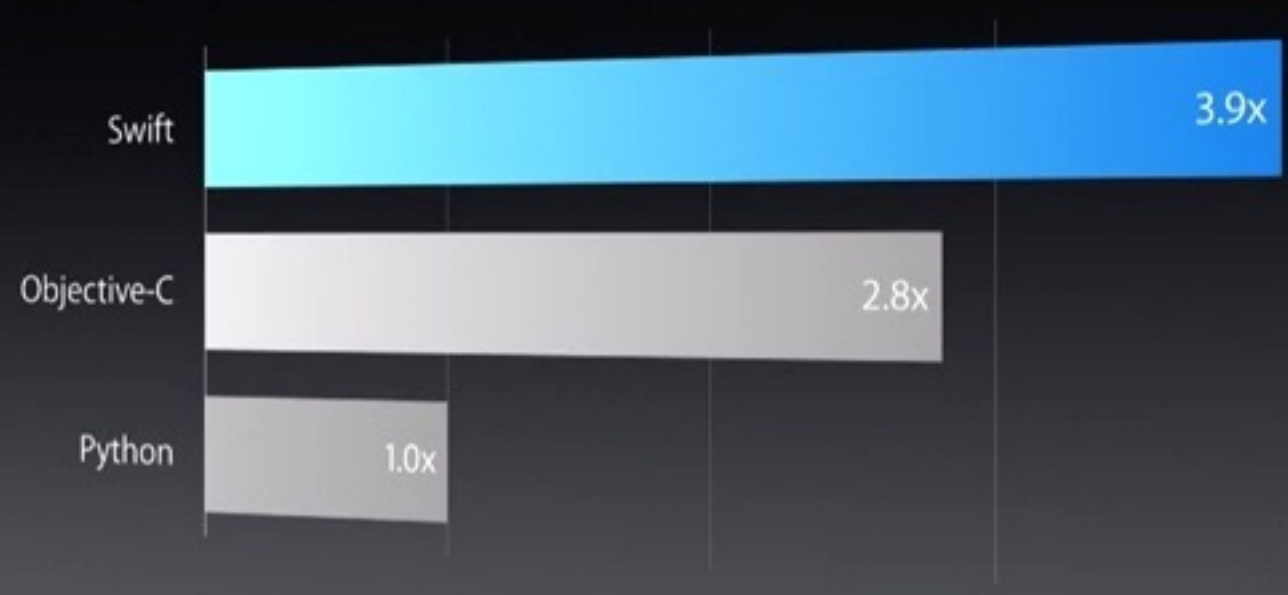
Source: Google Trends

Growth of major programming languages

Based on Stack Overflow question views in World Bank high-income countries



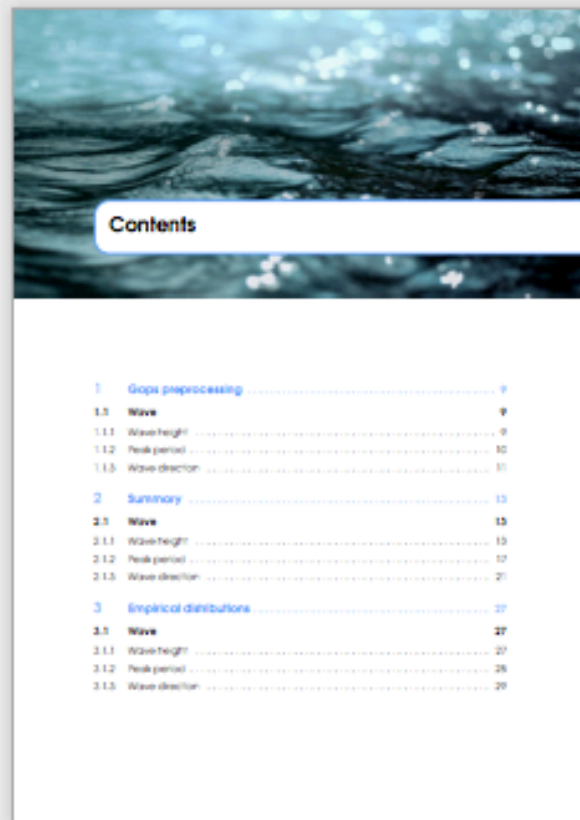
Complex object sort



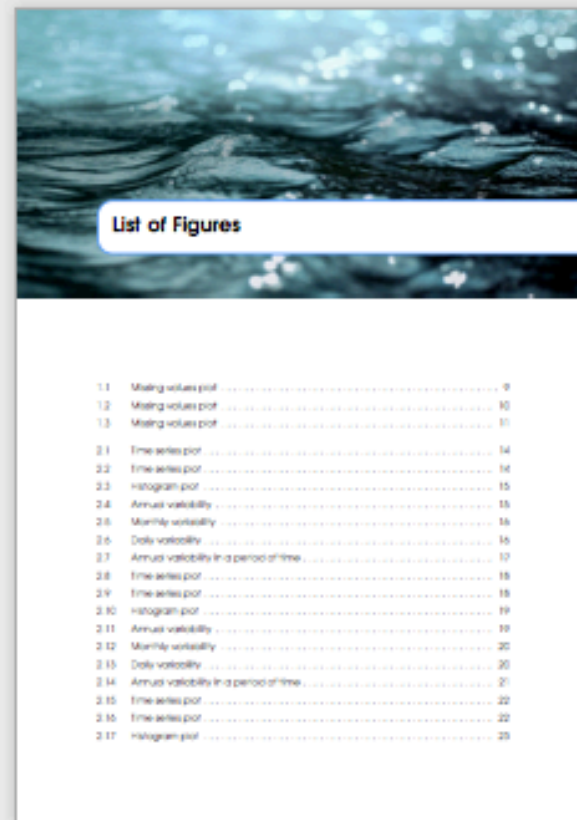
Samples



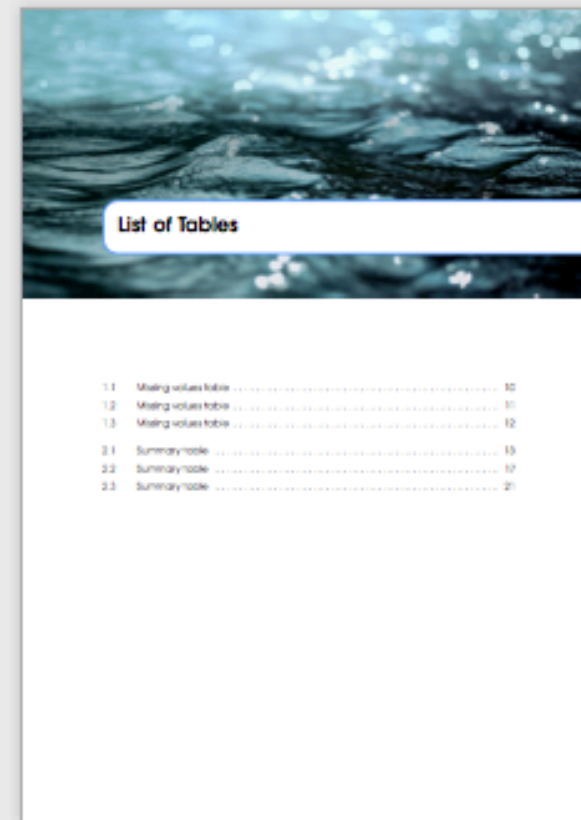
1



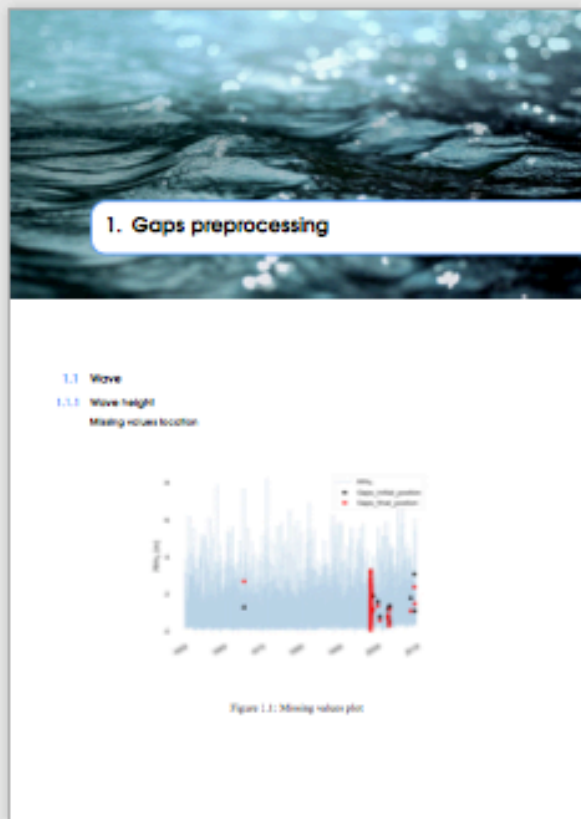
2



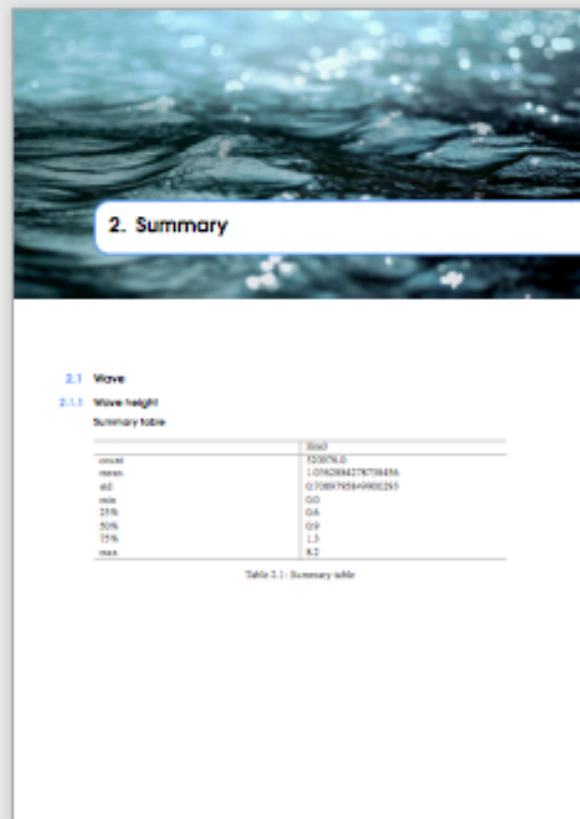
3



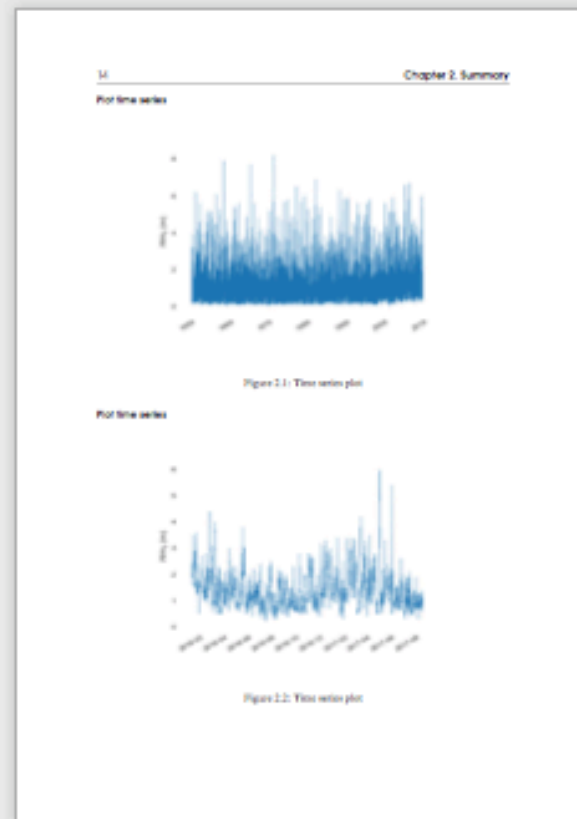
4



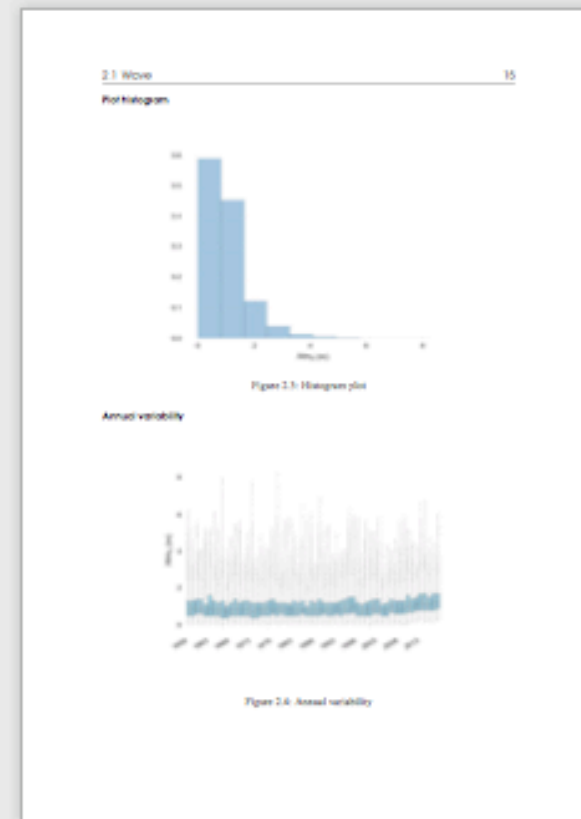
5



6



7



8

TOTAL WATER LEVEL

Study site

Guadalete

Result location

Point

1

X-UTM (m)

747614.42

Y-UTM (m)

4052587.12

Time zone

29N

Next



Back

Start

Select agents

Astronomical tide Storm surge Wave River discharge

Forecasted water level

Past water level

Select simulation parameters

Initial year

2020

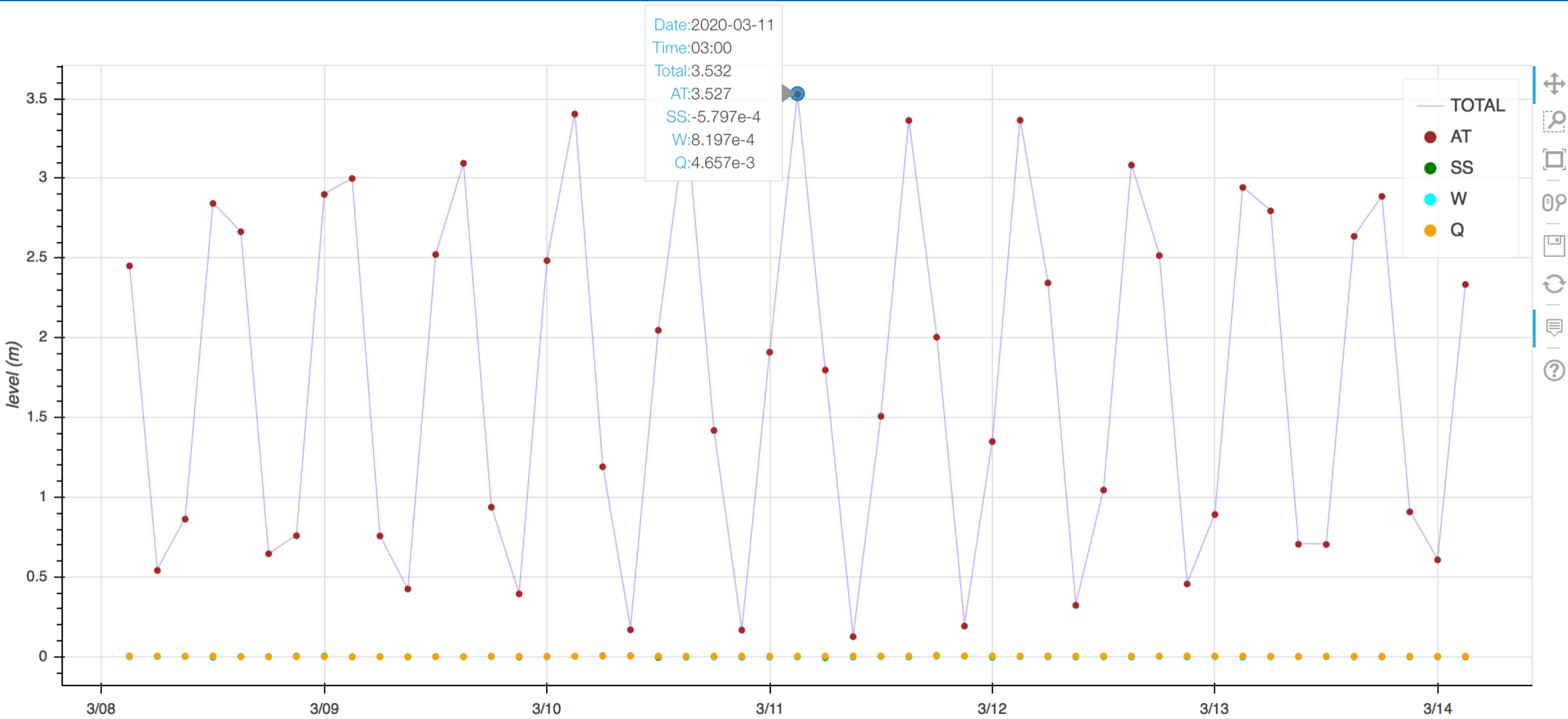
Number of years

2

Select conditions

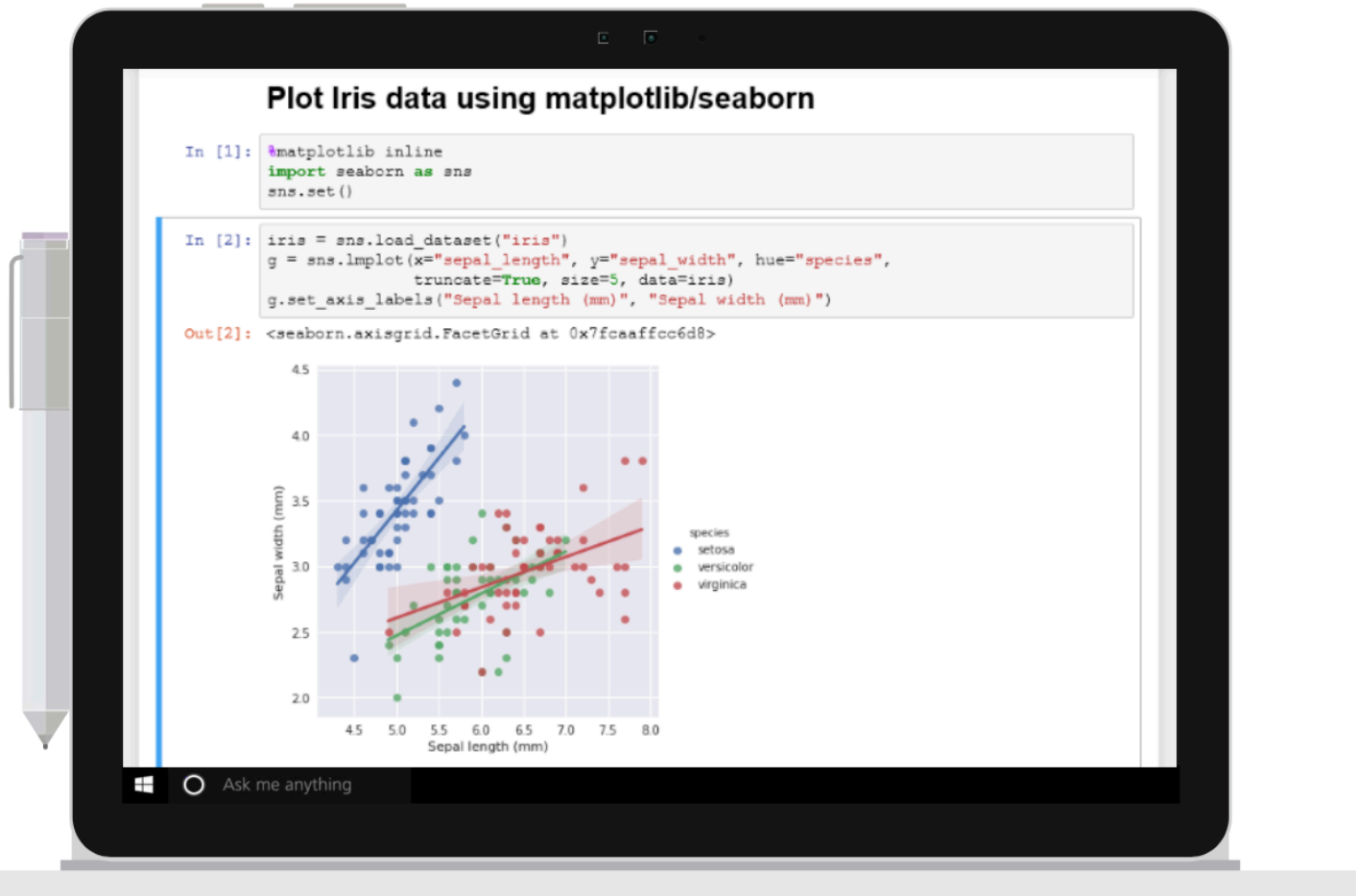
Operation

Extreme





Featured: Dr. Garth Wells' Eng101 @ Cambridge University



Interactive coding in your browser

Free, in the cloud, powered by [Jupyter](#)

Get Started




Te damos la bienvenida a Colaboratory

Introducción

Funciones destacadas

- Ejecución de TensorFlow
- GitHub
- Visualización
- Compatibilidad con tiempos de ejecución locales

+ SECCIÓN



Te damos la bienvenida a Colaboratory

Colaboratory es un entorno gratuito de Jupyter Notebook que no requiere configuración y que se ejecuta completamente en la nube. Puedes consultar más información en la sección de [preguntas frecuentes](#).

Introducción

- [Descripción general de Colaboratory](#)
- [Cargar y guardar datos: archivos locales, Drive, Hojas de cálculo y Google Cloud Storage](#)
- [Importar bibliotecas e instalar dependencias](#)
- [Usar Google Cloud BigQuery](#)
- [Formularios, Gráficos, Markdown y Widgets](#)
- [TensorFlow con GPU](#)
- [Curso intensivo de aprendizaje automático: Introducción a Pandas y Primeros pasos con TensorFlow](#)

▼ Funciones destacadas

▼ Ejecución de TensorFlow

Colaboratory permite ejecutar código de TensorFlow en el navegador con un solo clic. En el siguiente ejemplo se añaden dos matrices.

$$\begin{bmatrix} 1. & 1. & 1. \\ 1. & 1. & 1. \end{bmatrix} + \begin{bmatrix} 1. & 2. & 3. \\ 4. & 5. & 6. \end{bmatrix} = \begin{bmatrix} 2. & 3. & 4. \\ 5. & 6. & 7. \end{bmatrix}$$

```
[ ] import tensorflow as tf
```


7 Popular Software Programs Written in Python

Python is a popular coding language for several reasons – it's relatively easy to learn and read, has a massive library to help you solve many of your coding problems, and a very active and welcoming community of users.

Even if you have no idea what kind of language Python is, chances are you're quite familiar with many programs that are written in Python. Here's a list of some of the more popular ones:

YouTube

With over 4 million views per day and 60 hours of video uploaded every minute, YouTube has become one of the most visited sites on the planet. Python is used for different purposes all over the site and because of its speed, it allows for the development of maintainable features in record time. Every time you watch a video, you're executing Python code.

Google

Python is recognized as an official language at Google and has been with them since the beginning. Its flexibility, rapid development, scalability and excellent performance are the reasons why Python is so actively used – in things such as system administration tools and lots of Google App Engines apps. Google has a strong relationship with the language and sponsors various Python conferences.

Instagram

Founded in 2010, Instagram has become one of the most popular photo / video sharing social media apps with over 300 million users. The app utilises many languages but it's application servers are built using iterations of Python with Django as the web framework.

Reddit

An entertainment, social networking, and news site – all rolled into one. It's one of the biggest communities on the web and its registered users, people like you, provide the content. Originally written in Common Lisp, it was rewritten in Python in 2005 to gain greater development flexibility and access to Python's plethora of code libraries.

Spotify

Spotify is a popular music streaming service and a big fan of Python – they use it in their back-end services and in data analysis. The Python module, Luigi, is used to power the Radio and Discover features, as well as the recommendations for people to follow. Speed is an important factor at Spotify and Python accomplishes this. Spotify is also active in the Python community and sponsors conferences.

Dropbox

Dropbox lives in the cloud – offering services in cloud storage, data management, file sharing, and client software. Originally, both the Dropbox server (running on the cloud) and desktop client software were primarily written in Python. Drew Houston, co-founder of Dropbox, considers Python one of his favorite languages due to its simplicity, flexibility, and elegance.

Quora

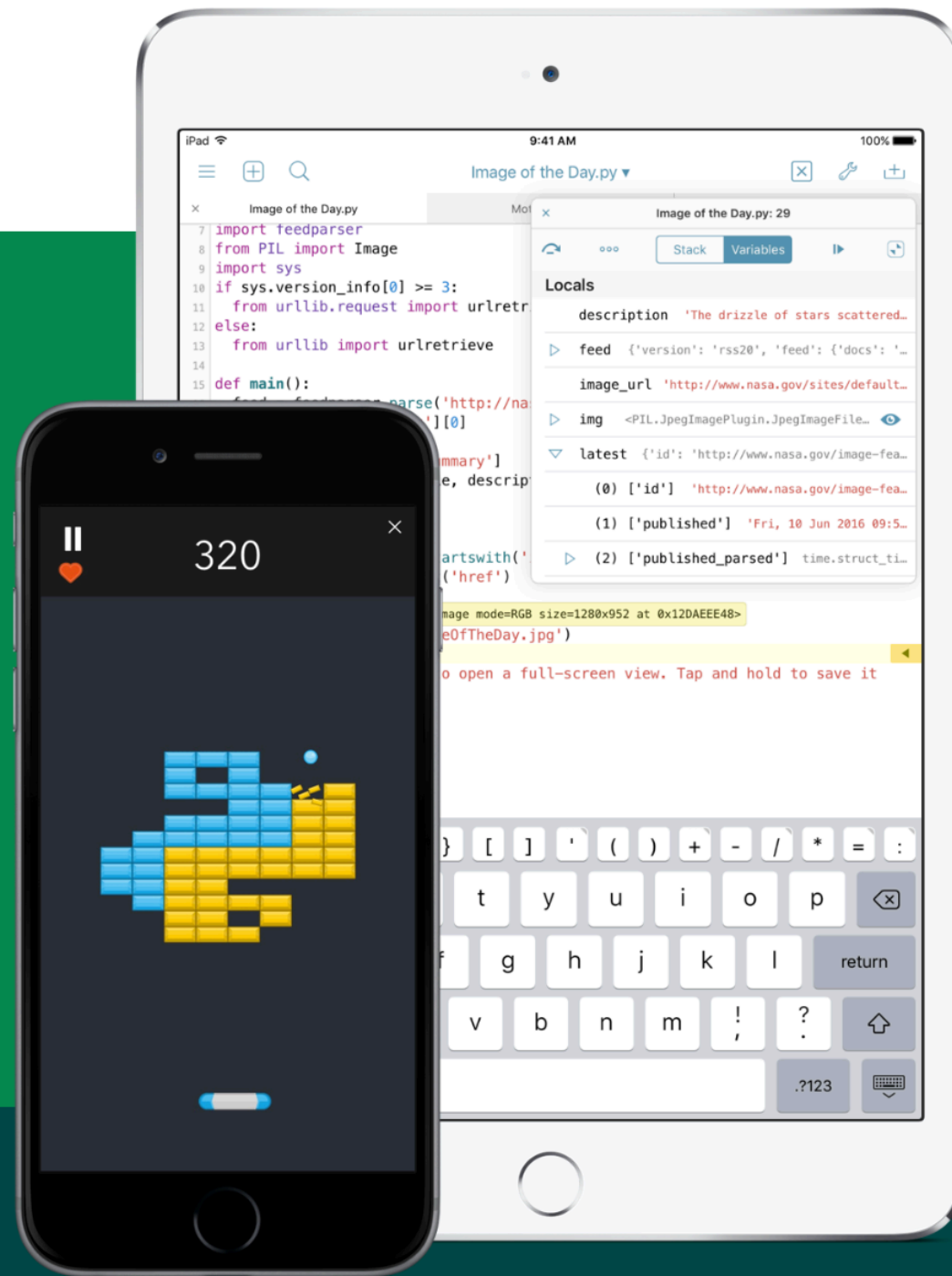
Got a question? You can ask it here – on just about any topic you can think of. The creators of Quora, who used to work for Facebook, chose Python because it's expressive and quick to write. LiveNode, one of the internal systems that manages the display of content on the webpage, is partly written in Python.

Pythonista 3

A Full Python IDE for iOS

Pythonista is a complete development environment for writing Python™ scripts on your iPad or iPhone. Lots of examples are included — from games and animations to plotting, image manipulation, custom user interfaces, and automation scripts.

In addition to the powerful standard library, Pythonista provides extensive support for interacting with native iOS features, like contacts, reminders, photos, location data, and more.



Universal App for iPhone + iPad

Introduction to Python

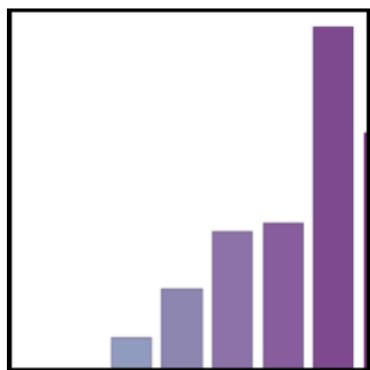
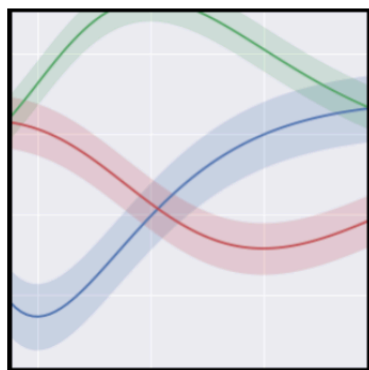
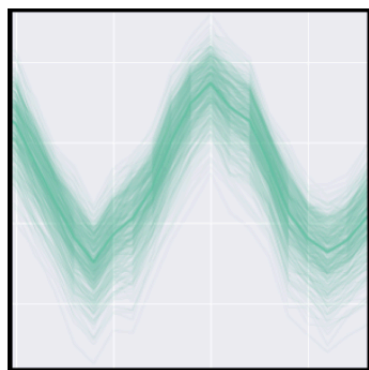
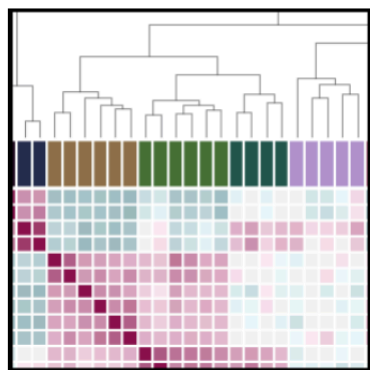
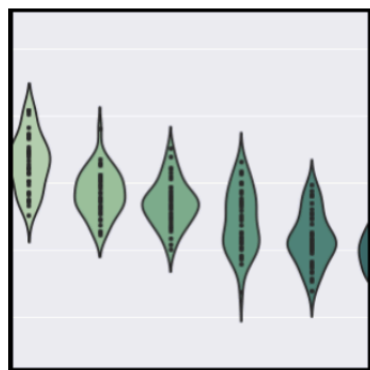
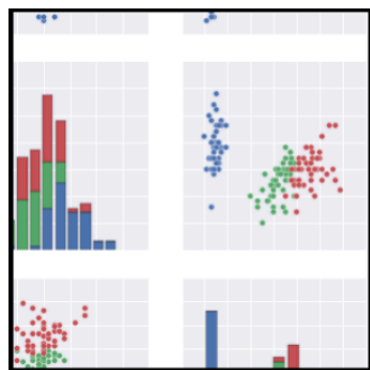
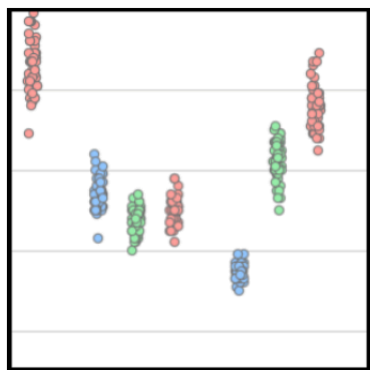
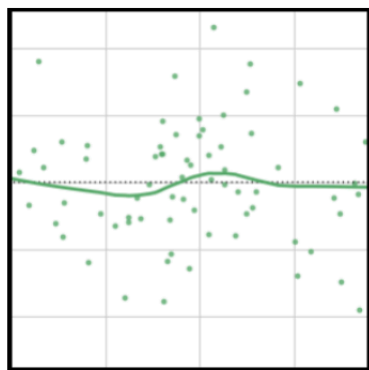
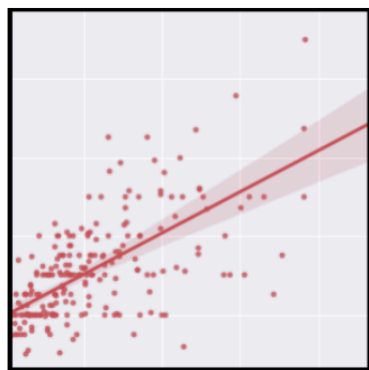
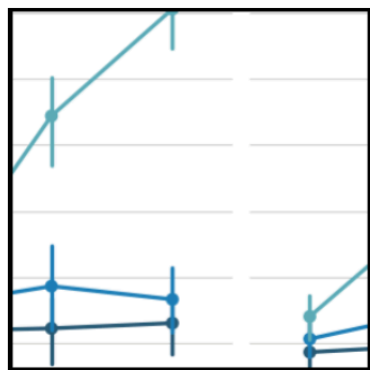
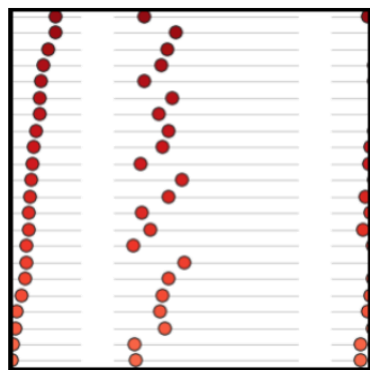
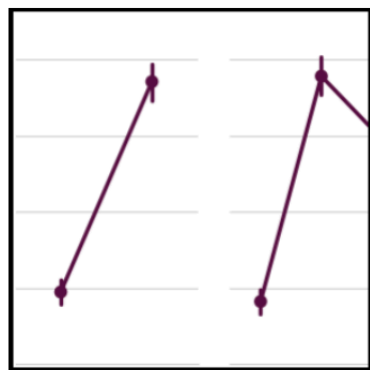
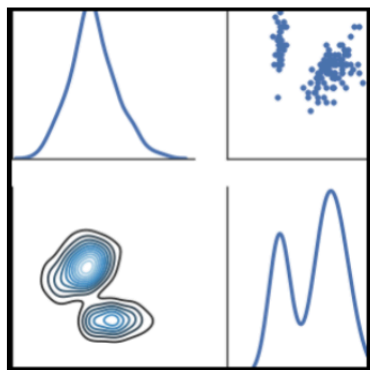
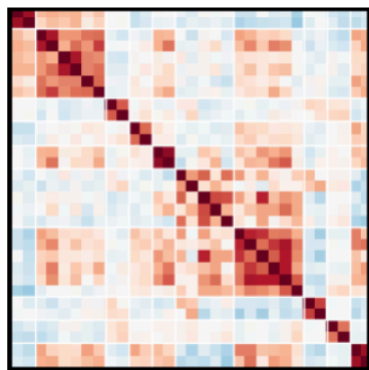
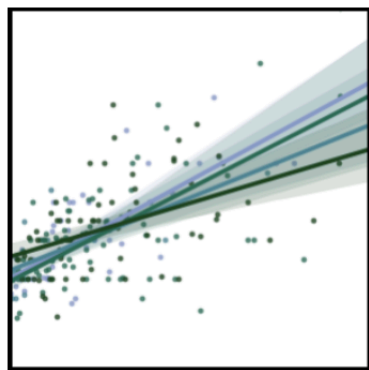
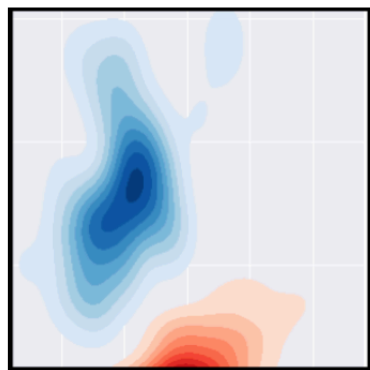
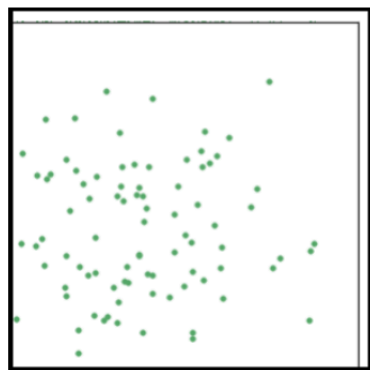
What is Python?

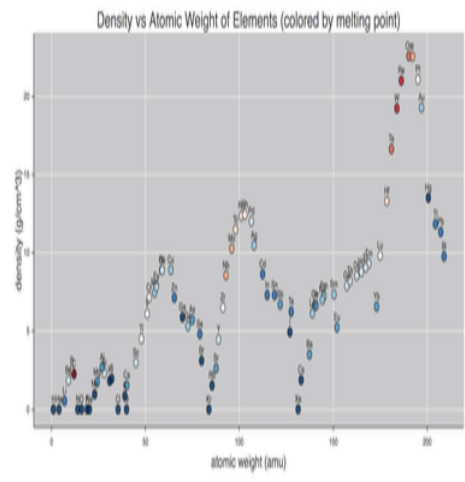
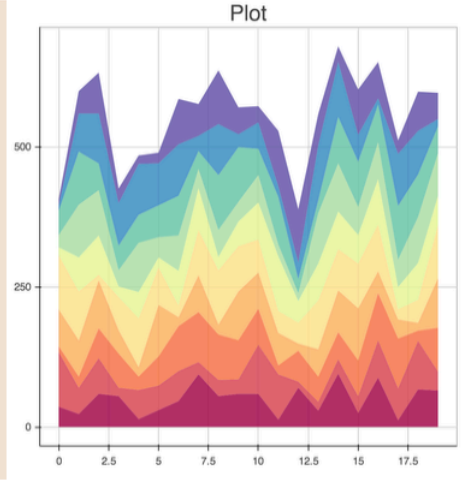
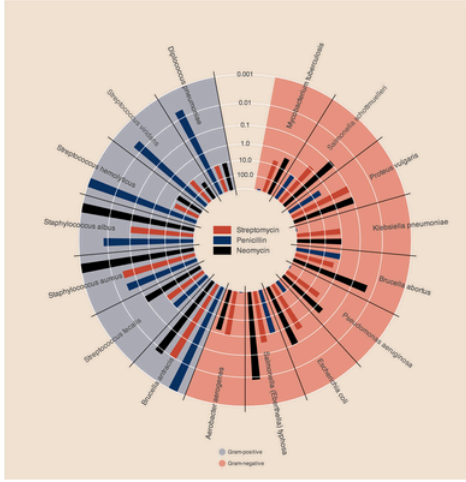
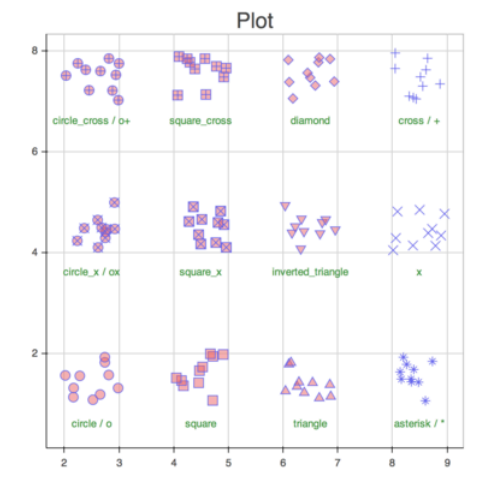
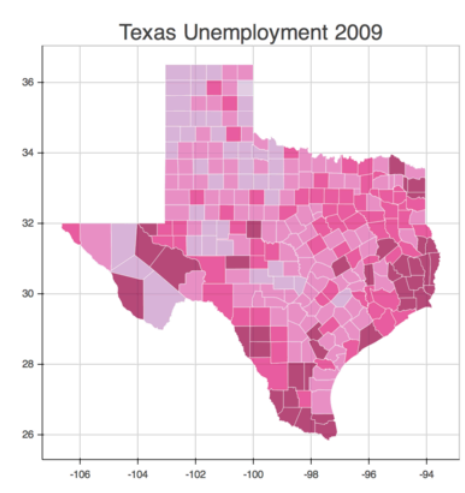
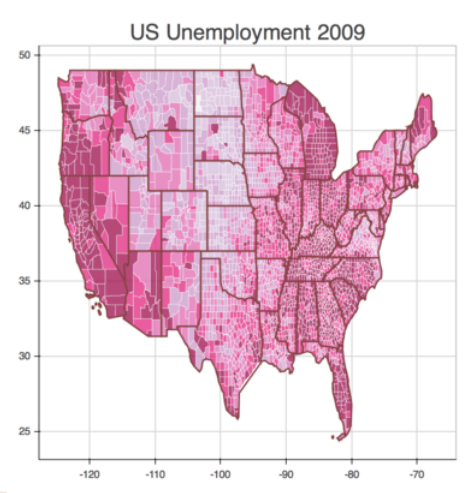
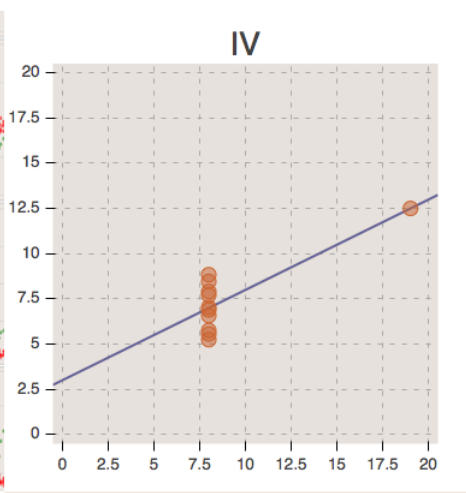
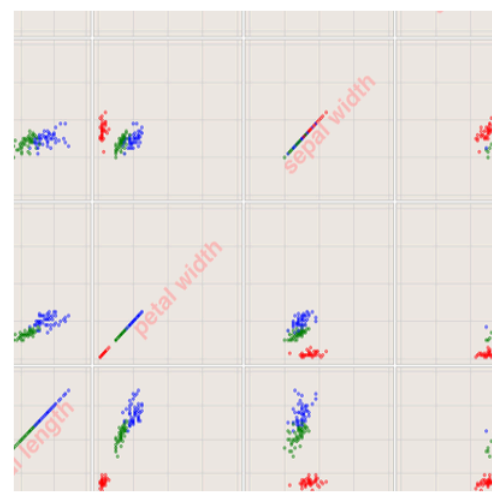
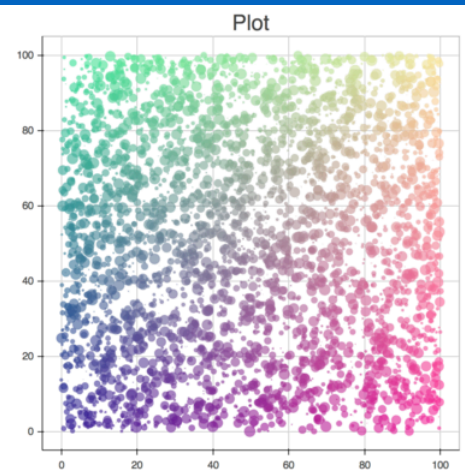
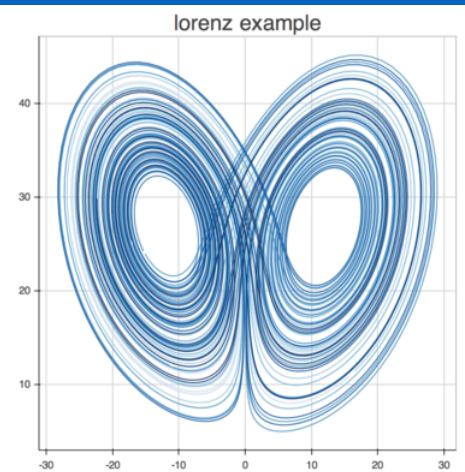
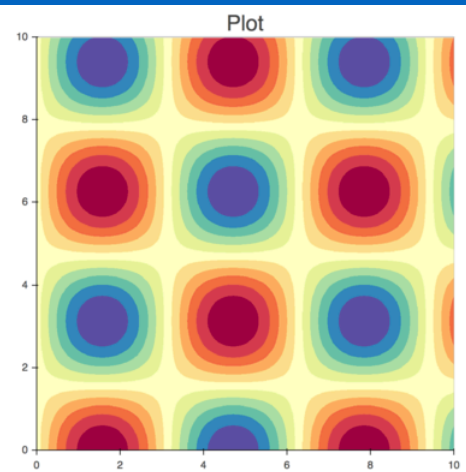
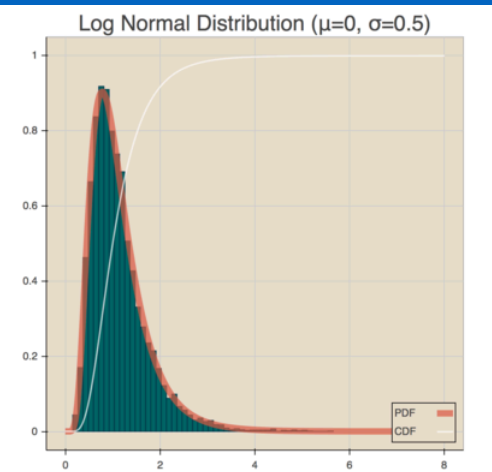
Python is a modern, general-purpose, object-oriented, high-level programming language.

General characteristics of Python:

- **clean and simple language:** Easy-to-read and intuitive code, easy-to-learn minimalistic syntax, maintainability scales well with size of projects.
- **expressive language:** Fewer lines of code, fewer bugs, easier to maintain.
- **encourage many good programming practices:** Modular, documentation tightly integrated with the code, etc.

- Extensive ecosystem of **scientific libraries**
 - **NumPy**: numerical Python \approx MATLAB matrices and arrays
 - **SciPy**: scientific Python \approx MATLAB toolboxes
 - **pandas**: extends NumPy
 - **Matplotlib**: graphics library
 - **Sympy**: symbolic mathematics library





- **Scientific (and non-scientific) development environments available**
 - **spyder**: MATLAB-like environment
 - **Jupyter/IPython notebooks**: environment for interactive and exploratory Python
 - **Visual Studio Code**: new Python lightweight environment
 - **PyCharm**: Python environment for developers
- **Great performance due to** close integration with time-tested and highly **optimized codes written in C/C++ and Fortran**
- Readily available and **suitable for use** on high-performance **computing clusters**
- **No license costs**, no unnecessary use of research budget

Python for science, where to begin?

Why are there two versions of Python?

- At one time, there were a lot of modules not compatibles with Python 3
- Python 2 is still **actively supported**. For example, many Linux distributions and Macs are still using internally 2.x as default



It's 2018. Why to choose Python 3?

- **Differences** between Python 2 and 3 are **relatively minor** for *beginner programmers*
- Python 3 brings **many improvements over Python 2**
- Python 2 end-of-life will be on **January 1st, 2020**

Scientific-oriented Python Distributions

Provide a **Python interpreter** with commonly used **scientific libraries** in science like NumPy, SciPy, Pandas, matplotlib, etc. already installed. In the past, it was usually painful to build some of these packages.

Also, include **development environments** with advanced editing, debugging and introspection features.

- **Anaconda**
 - Cross-platform
 - Supports Python 2 and 3
 - **Most widely adopted**
- **Canopy**
 - Cross-platform
 - Supports Python 2 and 3
 - Includes a built-in IDE
- **WinPython**
 - Windows-only platform
 - Only supports Python 3
- **Python(x,y)**
 - Windows-only platform
 - Only supports Python 2
 - Not actively developed



Included in Anaconda

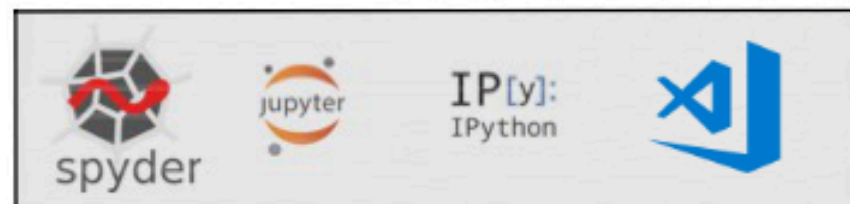
Python interpreter



Scientific libraries



Development environments



Anaconda Navigator

The screenshot displays the Anaconda Navigator desktop application. The window title is "Anaconda Navigator". The top bar features the Anaconda logo and the text "ANACONDA NAVIGATOR" on the left, and a "Sign in to Anaconda Cloud" button on the right. A left-hand sidebar contains navigation options: "Home" (selected), "Environments", "Learning", and "Community". Below these are buttons for "Documentation" and "Developer Blog", and social media icons for Twitter, YouTube, and GitHub. The main content area is titled "Applications on" with a dropdown menu set to "base (root)" and a "Channels" button. A "Refresh" button is in the top right of this area. The applications are arranged in a 2x3 grid:

- Jupyter Notebook** (5.7.0): Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis. [Launch](#)
- Qt Console** (4.3.1): PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more. [Launch](#)
- Spyder** (3.3.1): Scientific PYTHON Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features. [Launch](#)
- Glueviz** (0.13.3): Multidimensional data visualization across files. Explore relationships within and among related datasets. [Launch](#)
- JupyterLab** (0.35.0): An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture. [Launch](#)
- Orange 3** (3.16.0): Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox. [Launch](#)

Anaconda Navigator: installing new packages

The screenshot displays the Anaconda Navigator application window. The title bar reads "Anaconda Navigator". The main header features the "ANACONDA NAVIGATOR" logo and a "Sign in to Anaconda Cloud" button. A left sidebar contains navigation options: Home, Environments (highlighted), Learning, and Community. Below these are buttons for "Documentation" and "Developer Blog", and social media icons for Twitter, YouTube, and GitHub.

The central "Environments" panel shows a search bar for "Search Environments" and a list of environments: "base (root)" and "prophet". Below the list are "Create", "Clone", "Import", and "Remove" buttons.

The right panel shows the "Installed" tab for the "prophet" environment. It includes a "Channels" dropdown, an "Update index..." button, and a "Search Packages" search bar. Below this is a table of installed packages:

Name	Description	Version
<input checked="" type="checkbox"/> _nb_ext_conf		0.4.0
<input checked="" type="checkbox"/> aenum	Advanced enumerations (compatible with python's stdlib enum), namedtuples,	2.1.2
<input checked="" type="checkbox"/> affine	Matrices describing affine transformation of the plane.	2.2.1
<input checked="" type="checkbox"/> agate	A data analysis library that is optimized for humans instead of machines.	1.6.1
<input checked="" type="checkbox"/> agate-dbf	Agate-dbf adds read support for dbf files to agate.	0.2.0
<input checked="" type="checkbox"/> agate-excel	Agate-excel adds read support for excel files (xls and xlsx) to agate.	0.2.2
<input checked="" type="checkbox"/> agate-sql	Agate-sql adds sql read/write support to agate.	0.5.3
<input checked="" type="checkbox"/> alabaster	Configurable, python 2+3 compatible sphinx theme.	0.7.12
<input checked="" type="checkbox"/> altair		1.2.1
<input checked="" type="checkbox"/> anaconda	Simplifies package management and deployment of anaconda	↗ custom
<input checked="" type="checkbox"/> anaconda-clean	Delete anaconda configuration files	1.1.0
<input checked="" type="checkbox"/> anaconda-client	Anaconda.org command line client library	1.7.2

At the bottom of the right panel, it states "377 packages available".

Spyder

The image shows the Spyder Python IDE interface. The main window is titled "Spyder (Python 2.7)" and shows a code editor with a Python script. The script defines functions for printing variables, getting values, and saving data. The right-hand side of the interface features a help window for the `pandas.DataFrame` class, showing its definition and type. Below the help window are tabs for "Variable explorer", "File explorer", and "Help". At the bottom right, there is an IPython console window showing the IPython version and some introductory text. The status bar at the bottom of the IDE displays various settings: Permissions: RW, End-of-lines: LF, Encoding: UTF-8, Line: 28, Column: 23, and Memory: 56 %.

```
16 def print_variables(dataset):
17     for d in dataset.variables:
18         desc = dataset.variables[d].name + ': ' + dataset.variables[d].long_name
19         desc += ' [' + dataset.variables[d].units + ']' if 'units' in dataset.variables[d].nc
20
21         print desc
22
23
24 def get_value(dataset, variable, cp):
25     # extract index for the station
26     m = cp[0]
27     n = cp[1]
28     stations = pd.DataFrame(dataset.variables['MNSTAT'][0], columns=['M', 'N'])
29     station_index = stations[(stations.M == m) & (stations.N == n)].index[0]
30
31     # extract julian date
32     start_time = datetime.strptime(dataset.variables['time'].units, 'seconds since %Y-%m-%d %
33     seconds_offset = np.array(dataset.variables['time'][:, :], dtype='double')
34
35     time_vector = np.array([date2num(start_time + timedelta(seconds=s)) + 366 for s in seconds
36                             dtype='double'])
37
38     # extract values
39     values = np.array([], dtype='double')
40     if variable == 'ZWL':
41         values = np.array(dataset.variables[variable][:, station_index], dtype='double')
42     elif variable == 'ZCURU':
43         values = np.array(dataset.variables[variable][:, 0, station_index], dtype='double')
44
45     data = {'data_nc': OrderedDict([
46         ('X', np.array(dataset.variables['XSTAT'][:, station_index], dtype='double')),
47         ('Y', np.array(dataset.variables['YSTAT'][:, station_index], dtype='double')),
48         ('XUnits', 'm'),
49         ('YUnits', 'm'),
50         ('Val', values),
51         ('Time', time_vector),
52         ('Name', dataset.variables[variable].long_name),
53         ('Units', dataset.variables[variable].units)])
54     ]}
55
56     return data
57
58
59 def save_mat(data, filename):
60     savemat(filename, data, oned_as='column')
61
```

DataFrame

Definition : DataFrame(data=d)

Type : Present in pandas module

class DataFrame():

Two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Arithmetic operations align on both row and column labels. Can be thought of as a dict-like container for Series objects. The primary pandas data structure.

data : numpy ndarray (structured or homogeneous), dict, or DataFrame

IPython console

Python 2.7.15 [Anaconda custom (64-bit)] (default, May 1 2018, 18:37:05)
Type "copyright", "credits" or "license" for more information.

IPython 5.8.0 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]:

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 28 Column: 23 Memory: 56 %

Jupyter notebooks

The screenshot displays a Jupyter notebook interface with a toolbar at the top containing icons for file operations, navigation, and execution. The notebook content is organized into three distinct sections, each with a 'Slide Type' dropdown menu in the top right corner.

Section 1: The 'Slide Type' is set to '-'. The code cell contains:

```
In [57]:  
from sympy import diff, sin, exp  
diff(sin(x)*exp(x), x)
```

The output is: **Out[57]:** $e^x \sin(x) + e^x \cos(x)$

Section 2: The 'Slide Type' is set to 'Fragment'. The text cell contains: Compute $\int (e^x \sin(x) + e^x \cos(x)) dx$

Section 3: The 'Slide Type' is set to '-'. The code cell contains:

```
In [58]:  
from sympy import integrate, cos  
integrate(exp(x) * sin(x) + exp(x) * cos(x), x)
```

The output is: **Out[58]:** $e^x \sin(x)$

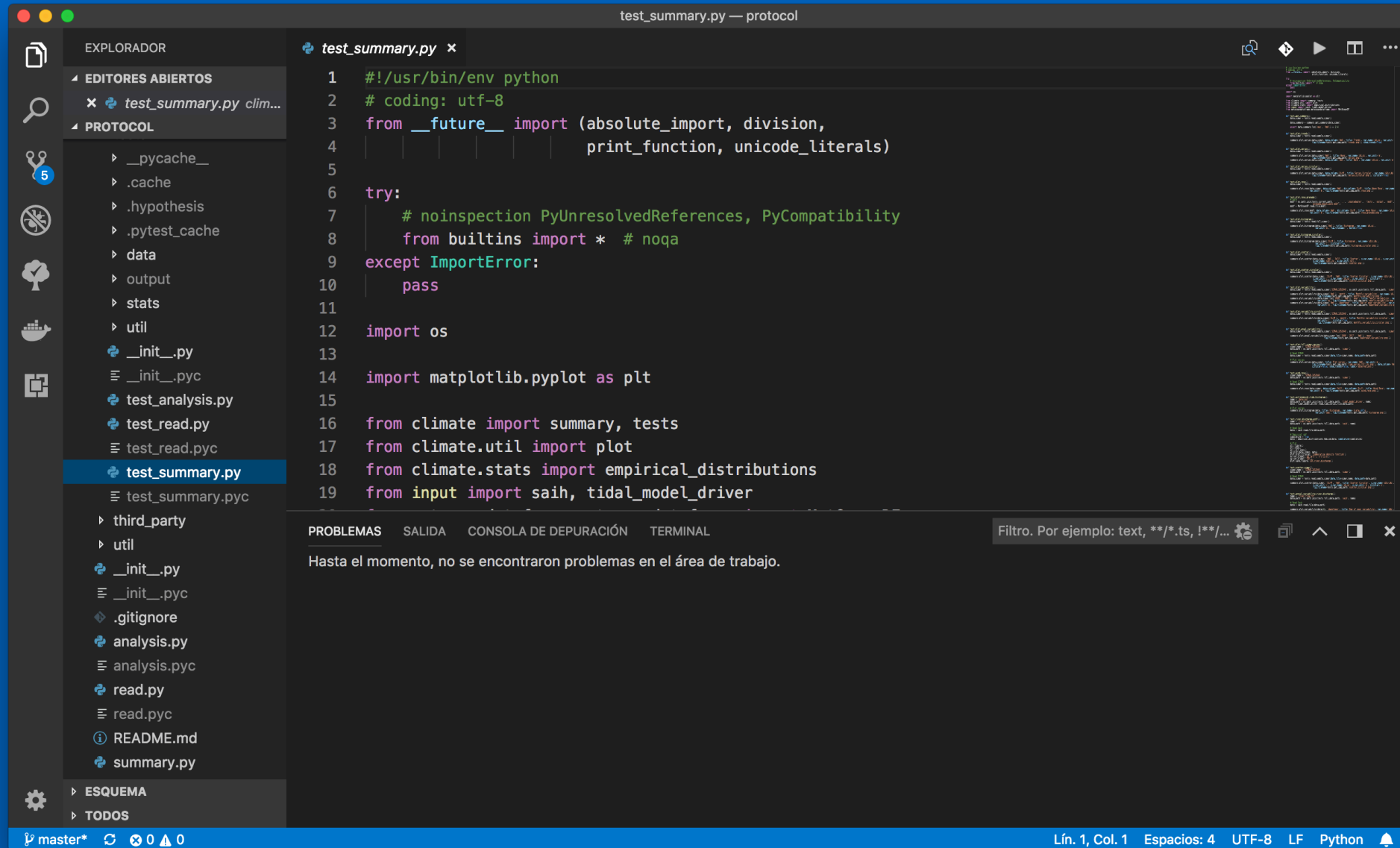
Section 4: The 'Slide Type' is set to 'Sub-Slide'. The text cell contains: Compute $\int_{-\infty}^{\infty} \sin(x^2) dx$

Section 5: The 'Slide Type' is set to '-'. The code cell contains:

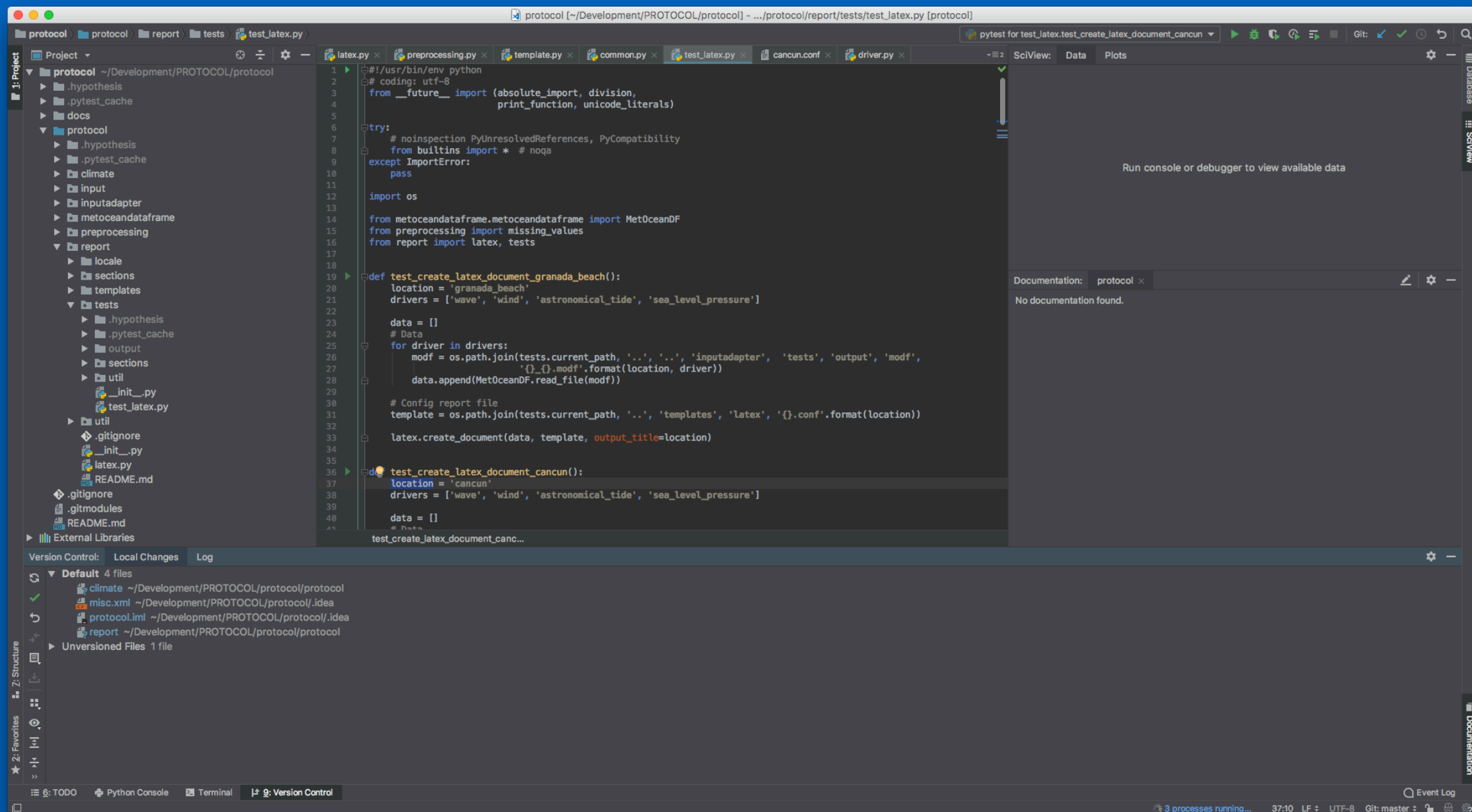
```
In [59]:  
from sympy import oo  
integrate(sin(x**2), (x, -oo, oo))
```

The output is: **Out[59]:** $\frac{\sqrt{2}\sqrt{\pi}}{2}$

Visual Studio Code



PyCharm (need to be installed separately from Anaconda)



Editor	Learning curve	Users	Benefits
Spyder	pretty short	Matlab and R background	mature, many features
Jupyter	smooth	teachers	interactive
Visual Studio Code	moderate	scientifics / developers	code quality
PyCharm	steep	developers	professional code

Where to look for help?

- **Official documentation:** <http://www.scipy.org/docs.html>
- Usually included in development environments as **contextual help**:
 - *Spyder*: Ctrl+I (Windows) or Cmd+I (Mac)
 - *Visual Studio Code*: Ctrl+Space (Windows/Mac)
 - *PyCharm*: F1 (Windows/Mac)
- **Be careful about code you get on the internet!**

Bibliography

- [Elegant SciPy: The Art of Scientific Python](#) por Juan Nunez-Iglesias, Stéfan van der Walt y otros (2017). ISBN: 9781491922873.
- [Python for Data Analysis \(2nd Edition\)](#) por Wes McKinney (2017). ISBN: 1491957662.
- [Pandas Cookbook: Recipes for Scientific Computing, Time Series Analysis and Data Visualization using Python](#) por Theodore Petrou (2017). ISBN: 9781784393878.

MOOC (Online Courses)

- [Python for Data Science \(University of California\)](#)
- [Introduction to Python for Data Science \(Microsoft\)](#)
- [Intro to Python for Data Science \(Datacamp\)](#)
- [MOOC aggregator](#)