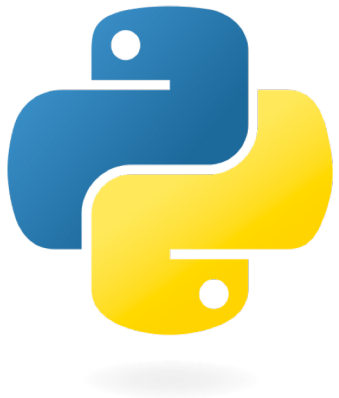


python™

for scientific computations

Barış Deniz Sağlam

bdsaglam@gmail.com

@bdsaglam

What is Python?

An interpreted, high-level programming language
for general-purpose programming.

open source	multi-paradigm	large standard library
automatic memory management	dynamic & strong type	huge community

History

- Guido Van Rossum started in 1989
- Python 1.0 released in 1994
- Python 2.0 released in 2000
- Python Software Foundation (PSF) formed in 2001
- Python 3.0 released in 2008

Philosophy

Python has a design philosophy that emphasizes code readability.

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.

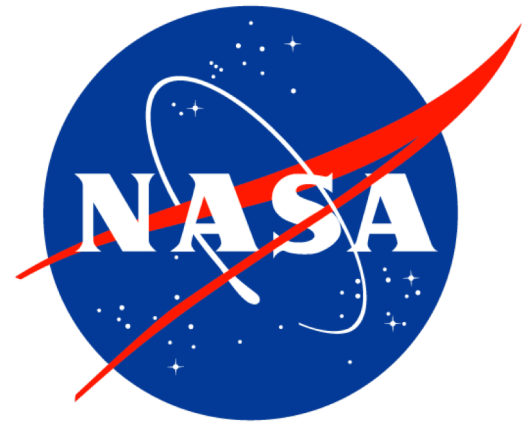
TIOBE Index

Jul 2018	Jul 2017	Change	Programming Language	Ratings	Change
1	1		Java	16.139%	+2.37%
2	2		C	14.662%	+7.34%
3	3		C++	7.615%	+2.04%
4	4		Python	6.361%	+2.82%
5	7	↑	Visual Basic .NET	4.247%	+1.20%
6	5	↓	C#	3.795%	+0.28%
7	6	↓	PHP	2.832%	-0.26%
8	8		JavaScript	2.831%	+0.22%
9	-	⇈	SQL	2.334%	+2.33%
10	18	⇈	Objective-C	1.453%	-0.44%

Applications

- Scientific & Numeric
- Desktop GUIs
- Web and Internet Development
- Education
- Database Access
- Network Programming
- Software & Game Development

WIKIPEDIA
The Free Encyclopedia

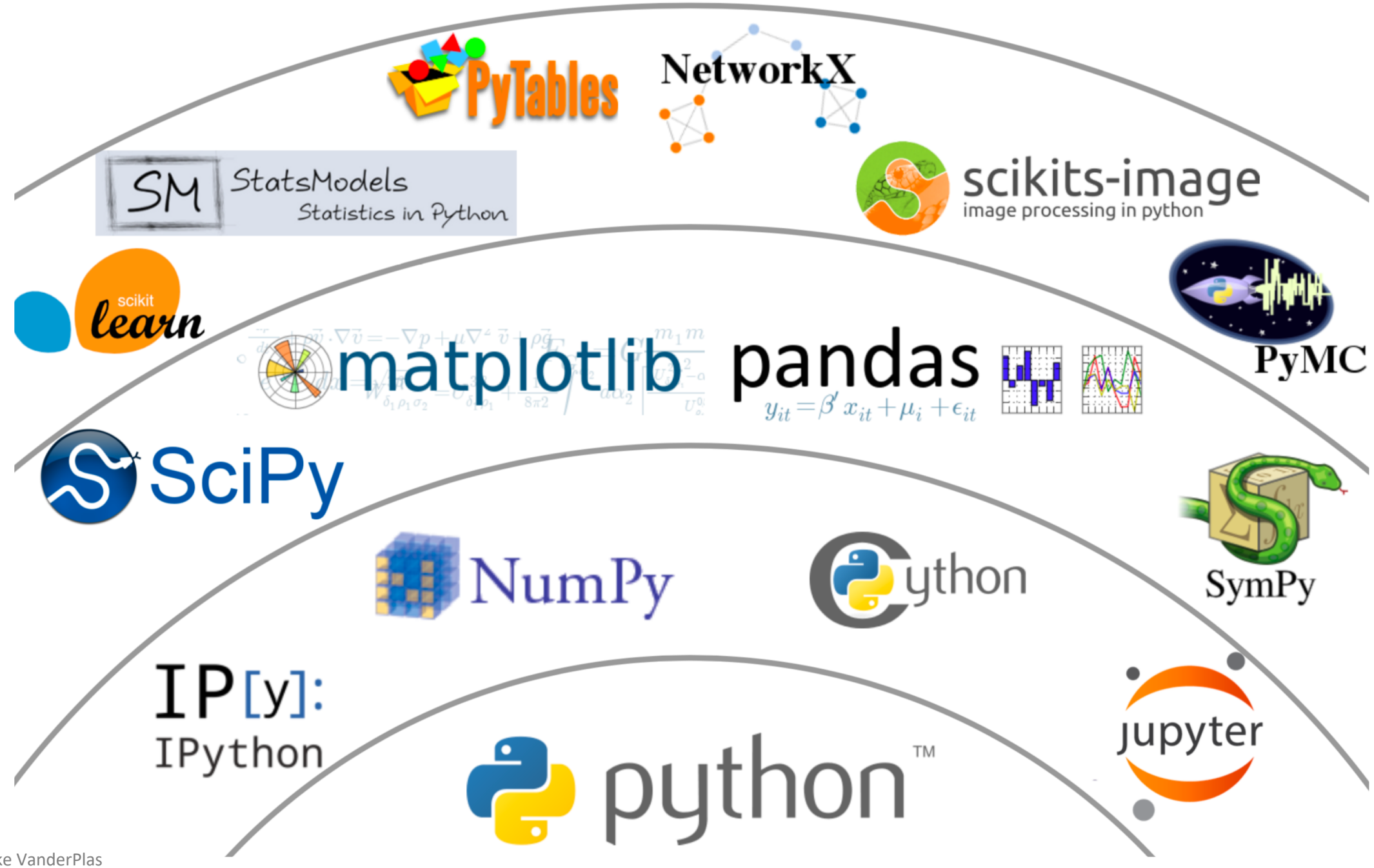


Python glues together this hodge-podge of scientific tools.

High-level syntax wraps low-level C/Fortran libraries, which is (mostly) where the computation happens.

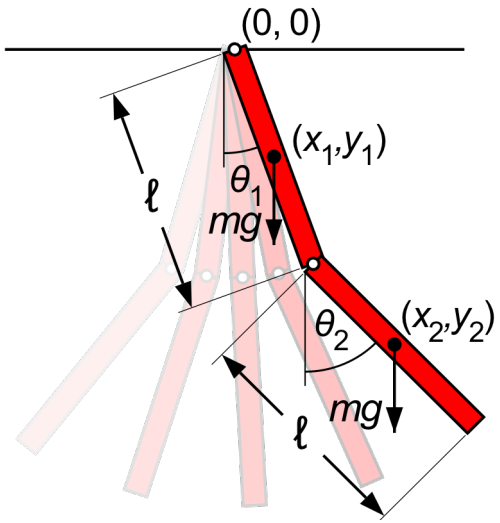
Python is Glue.

Slide from Jake VanderPlas



Slide from Jake VanderPlas

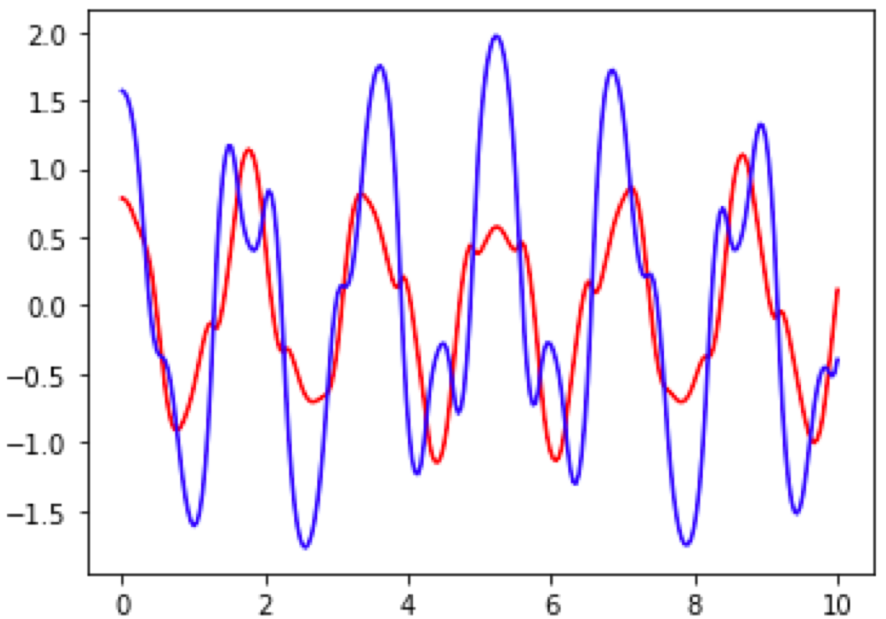
In [20]:
```
g = 9.82
L = 0.5
m = 0.1

def dx(x, t):
    """
    The right-hand side of the pendulum ODE
    """
    x1, x2, x3, x4 = x[0], x[1], x[2], x[3]

    dx1 = 6.0/(m*L**2) * (2 * x3 - 3 * cos(x1-x2) * x4)/(16 - 9 * cos(x1-x2)**2)
    dx2 = 6.0/(m*L**2) * (8 * x4 - 3 * cos(x1-x2) * x3)/(16 - 9 * cos(x1-x2)**2)
    dx3 = -0.5 * m * L**2 * ( dx1 * dx2 * sin(x1-x2) + 3 * (g/L) * sin(x1))
    dx4 = -0.5 * m * L**2 * (-dx1 * dx2 * sin(x1-x2) + (g/L) * sin(x2))

    return [dx1, dx2, dx3, dx4]
```

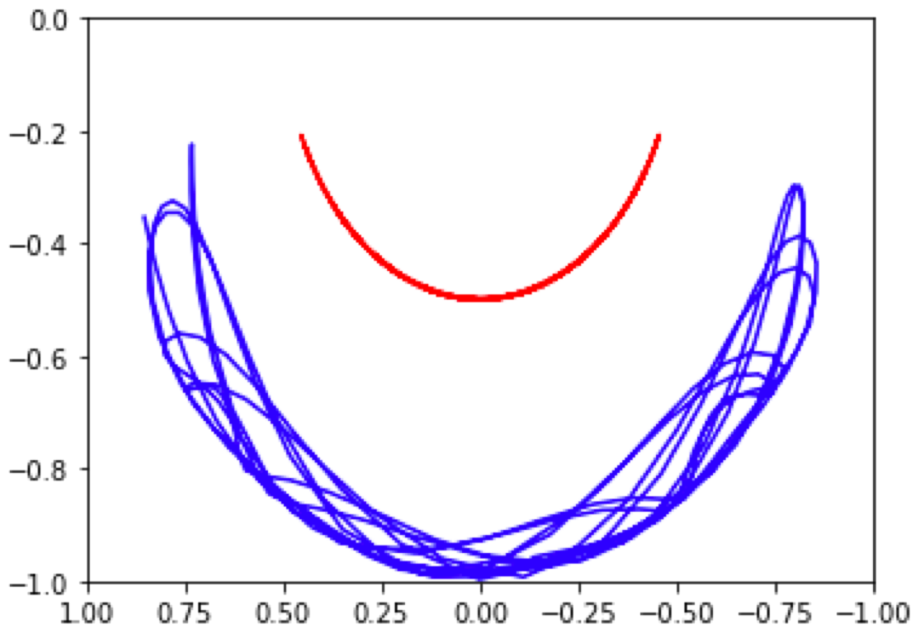


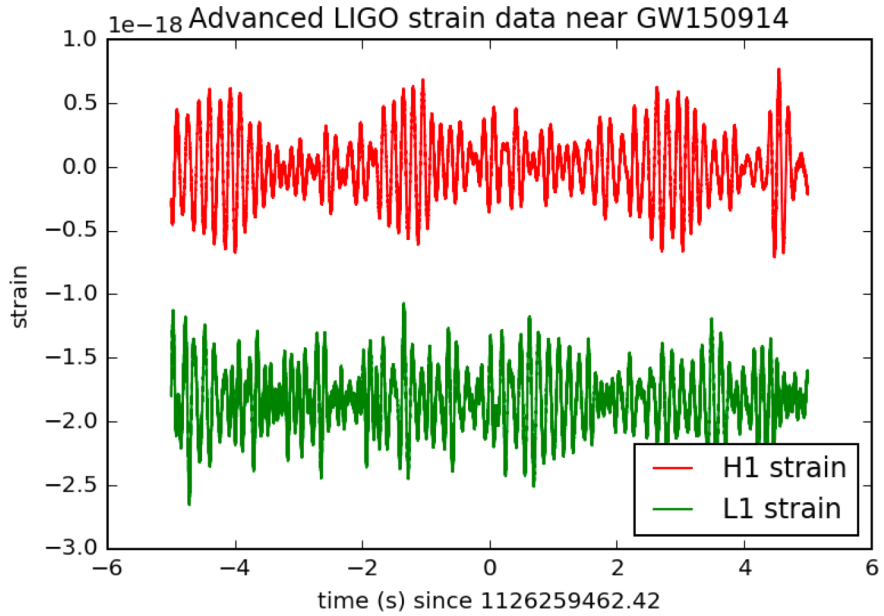
In [21]:
```
# choose an initial state
x0 = [pi/4, pi/2, 0, 0]
```

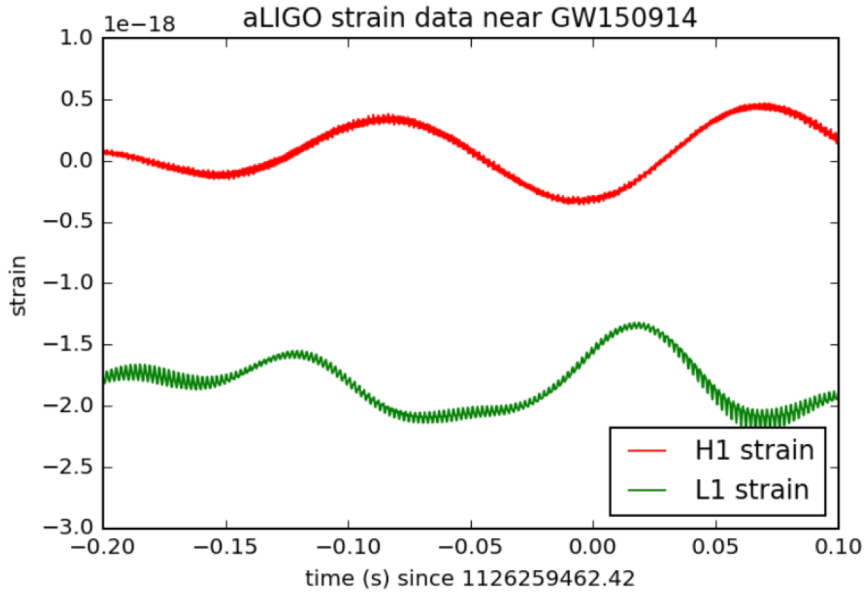
In [22]:
```
# time coodinate to solve
t = linspace(0, 10, 250)
```

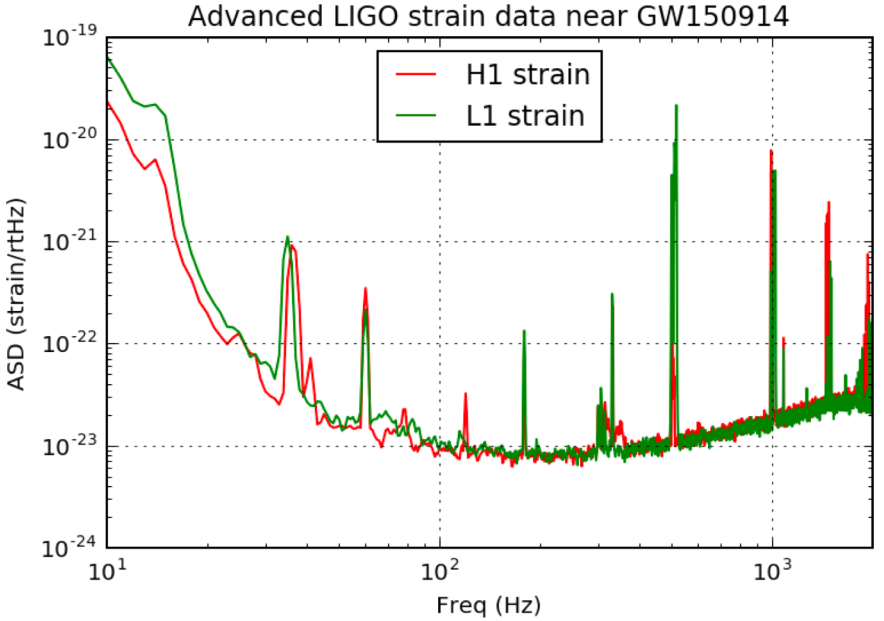
In [23]:
```
# solve the ODE problem
x = odeint(dx, x0, t)
```

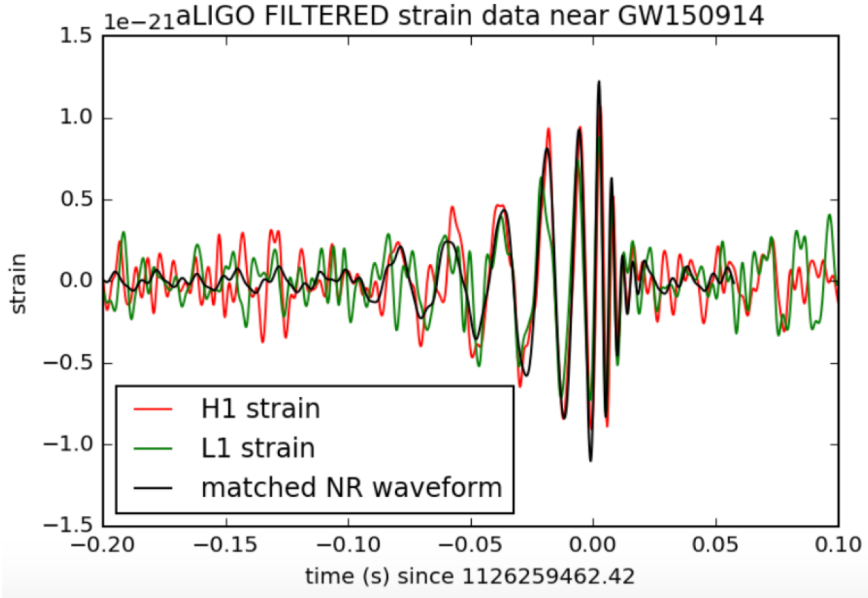


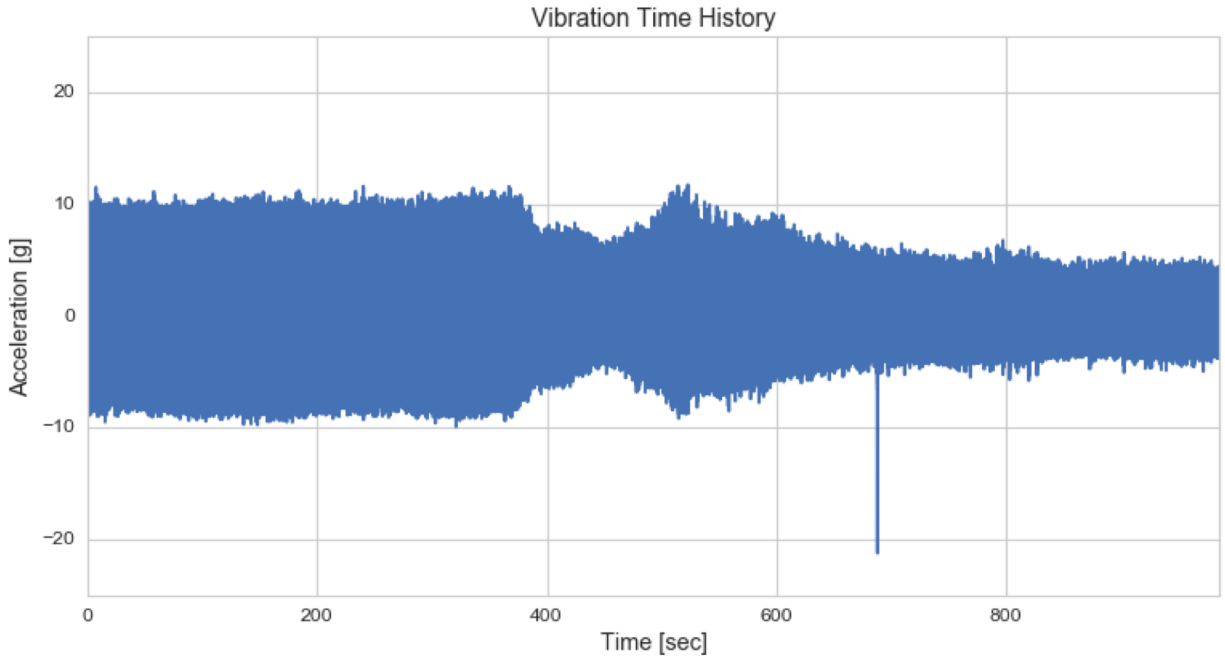


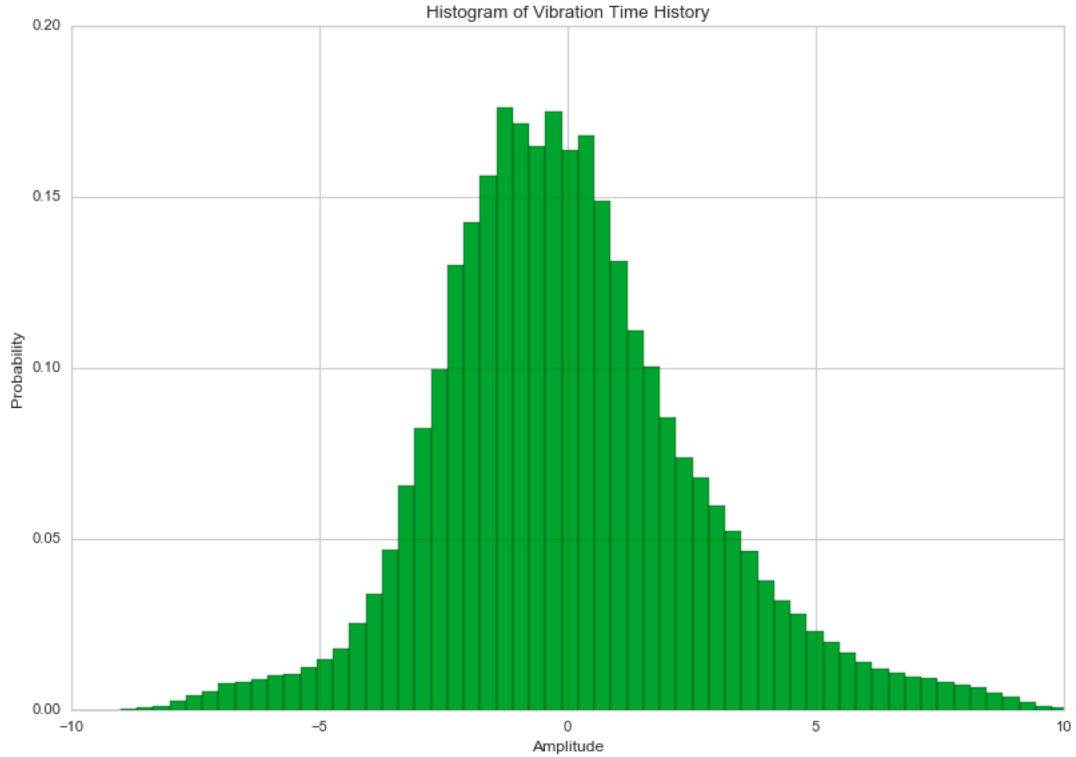

Advanced LIGO strain data near GW150914
1e−18
strain
time (s) since 1126259462.42
H1 strain
L1 strain

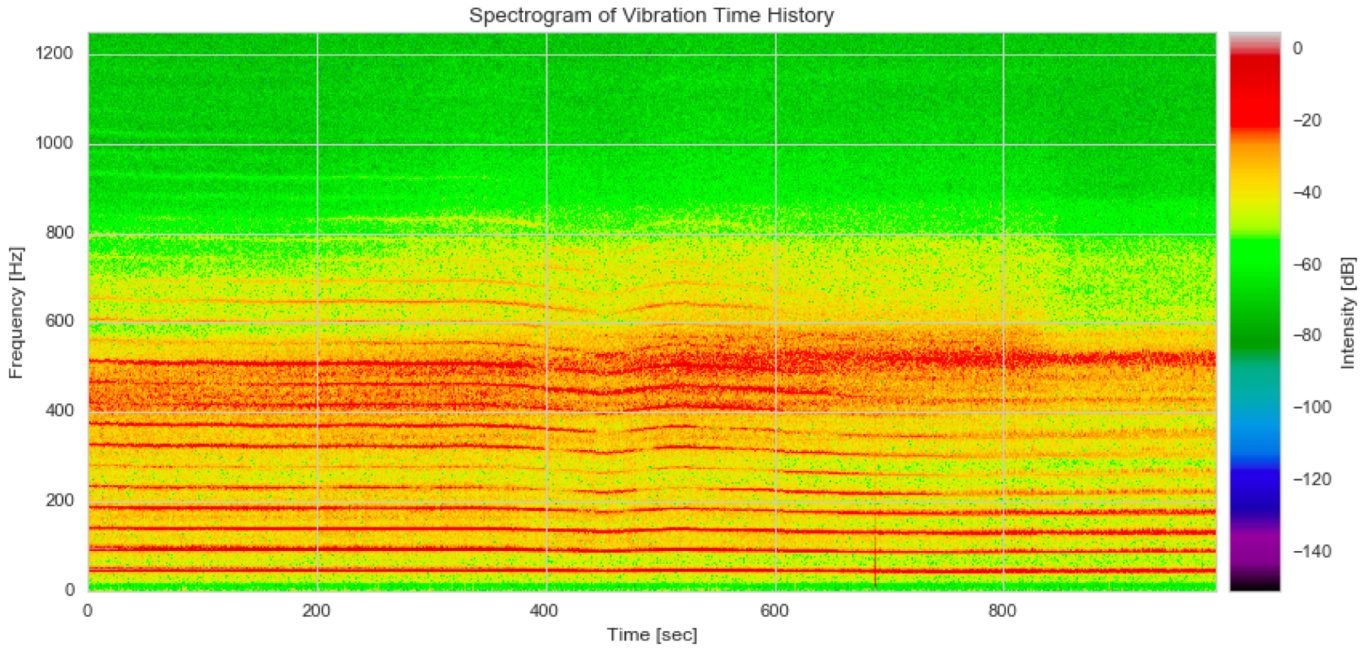

aLIGO strain data near GW150914
1e−18
strain
time (s) since 1126259462.42
H1 strain
L1 strain

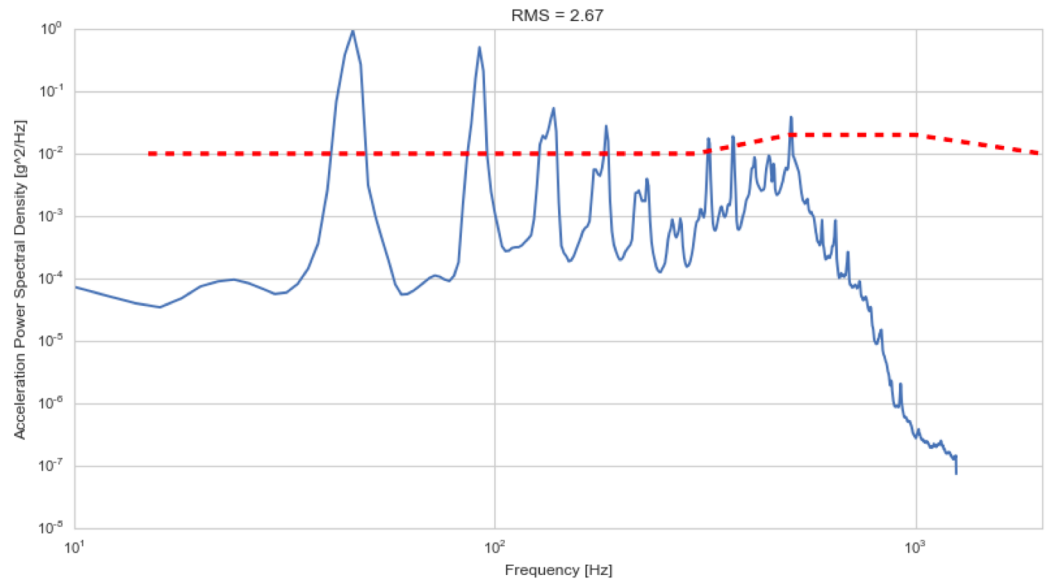

Advanced LIGO strain data near GW150914
ASD (strain/rtHz)
Freq (Hz)
H1 strain
L1 strain

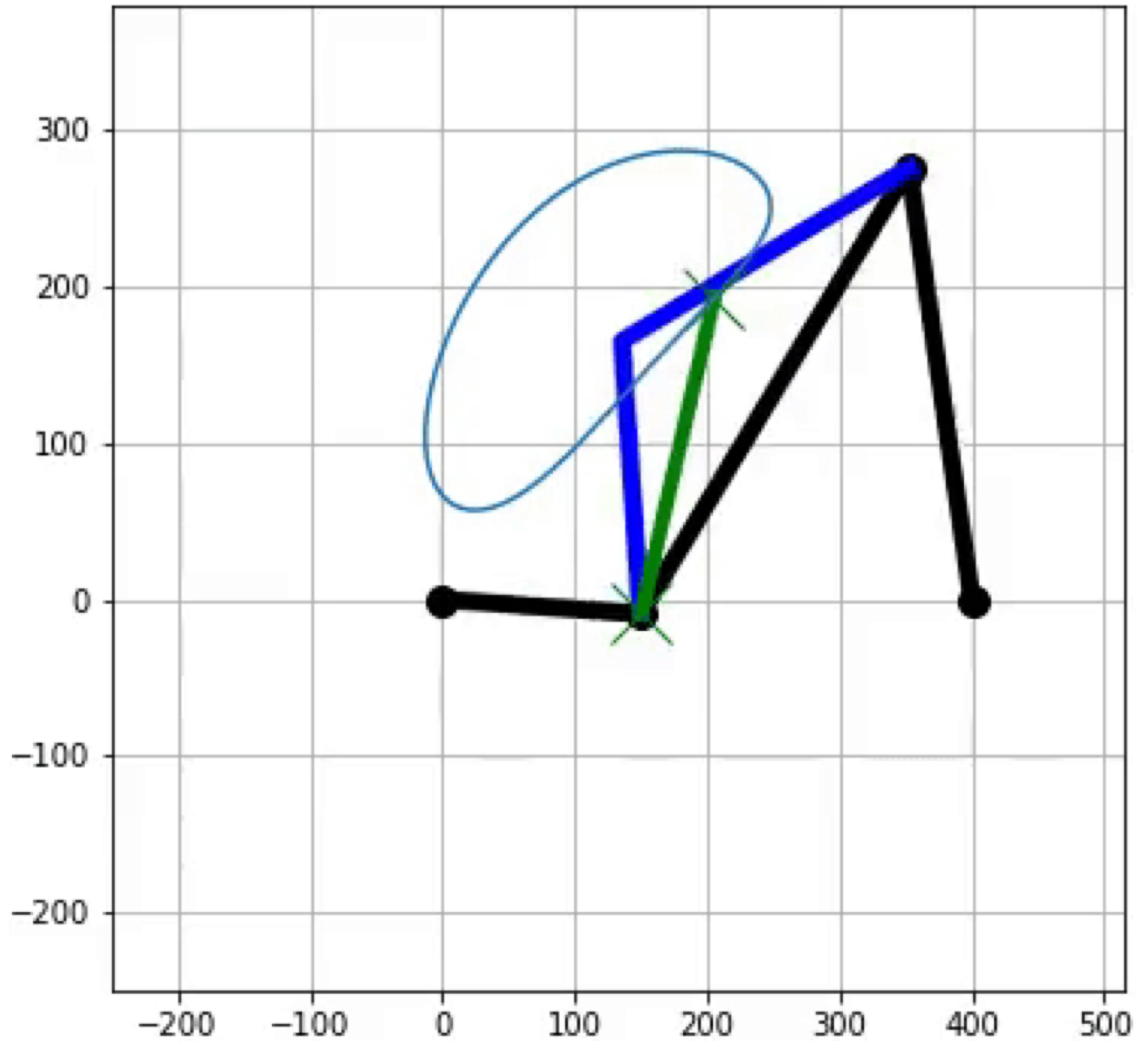

1e−21
aLIGO FILTERED strain data near GW150914
strain
time (s) since 1126259462.42
H1 strain
L1 strain
matched NR waveform


Vibration Time History
Acceleration [g]
Time [sec]
20
10
0
−10
−20
0
200
400
600
800

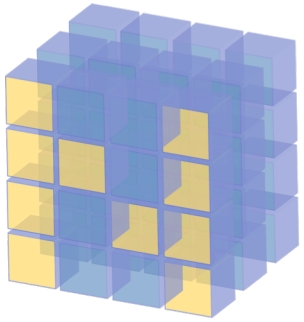

Histogram of Vibration Time History
Probability
Amplitude
0.20
0.15
0.10
0.05
0.00
−10
−5
0
5
10


Spectrogram of Vibration Time History
Frequency [Hz]
Time [sec]
Intensity [dB]
1200
1000
800
600
400
200
0
0
200
400
600
800
0
−20
−40
−60
−80
−100
−120
−140


RMS = 2.67
Acceleration Power Spectral Density [g^2/Hz]
Frequency [Hz]
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{1}
10^{2}
10^{3}



watch on web

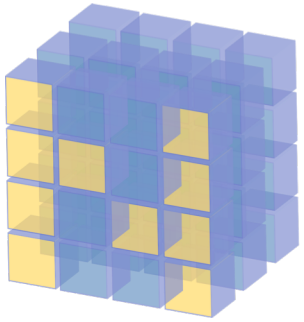


NumPy

The fundamental package for scientific computing with Python

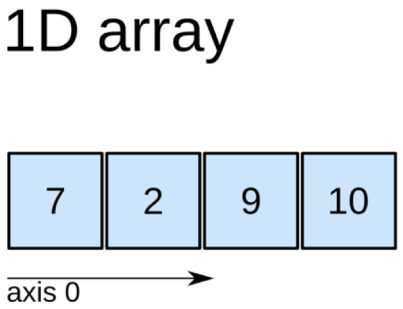
- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

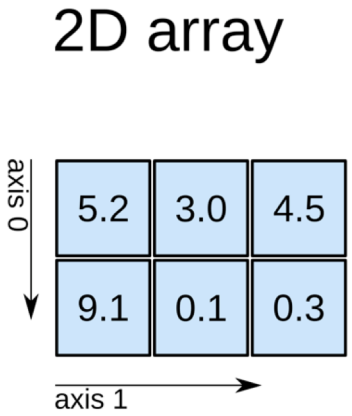
Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

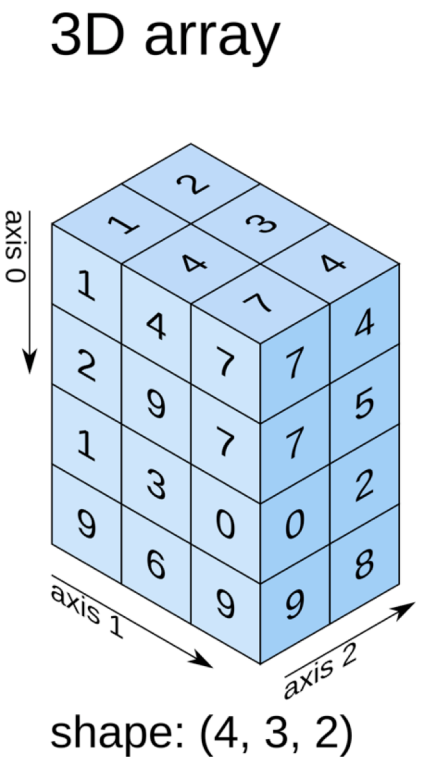


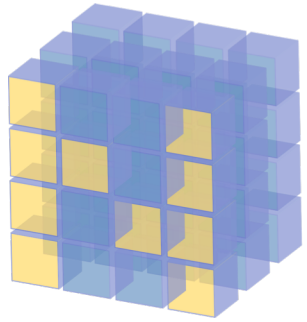
NumPy

```
>>> import numpy as np
>>> a = np.arange(15).reshape(3, 5)
>>> a
array([[ 0, 1, 2, 3, 4],
       [ 5, 6, 7, 8, 9],
       [10, 11, 12, 13, 14]])
```









NumPy

- 18,387 commits made by 821 contributors; 290,520 lines of code
- Mostly written in C and Python
- First commit in December, 2001; most recent commit about 1 day ago
- Estimated cost $ 4,241,474

SciPy is a collection of mathematical algorithms and convenience functions built on the Numpy extension of Python.

Provides the user with high-level commands and classes for manipulating and visualizing data.

With Python, a data-processing and system-prototyping environment rivaling systems such as MATLAB, IDL, Octave, R-Lab, and SciLab.

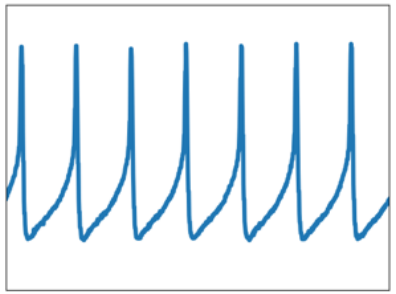
Scientific applications using SciPy benefit from the development of additional modules in numerous niches of the software landscape by developers across the world. Everything from parallel programming to web and data-base subroutines and classes have been made available to the Python programmer.

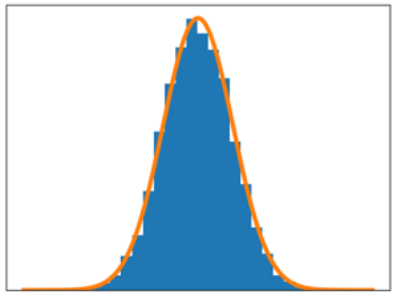
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions

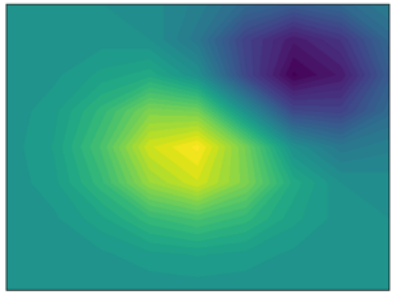
- 19,550 commits made by 804 contributors representing 394,890 lines of code
- Mostly written in Python, Fortran and C
- First commit in February, 2001; most recent commit 1 day ago
- Estimated cost $ 5,850,604

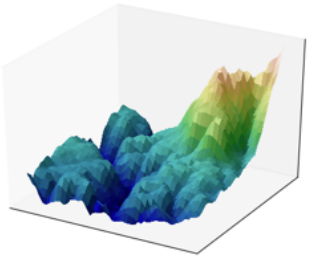
matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

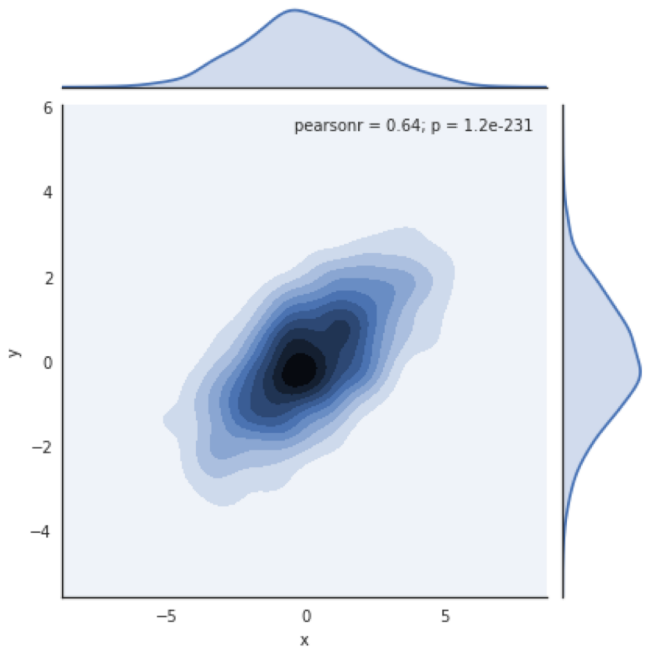


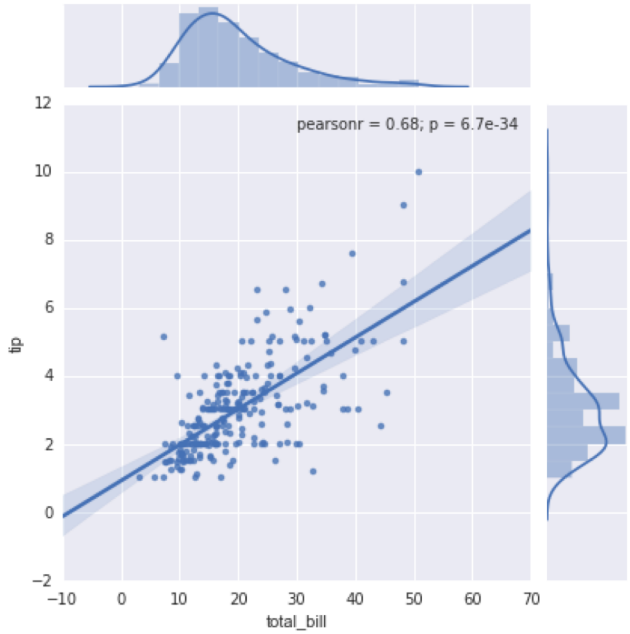


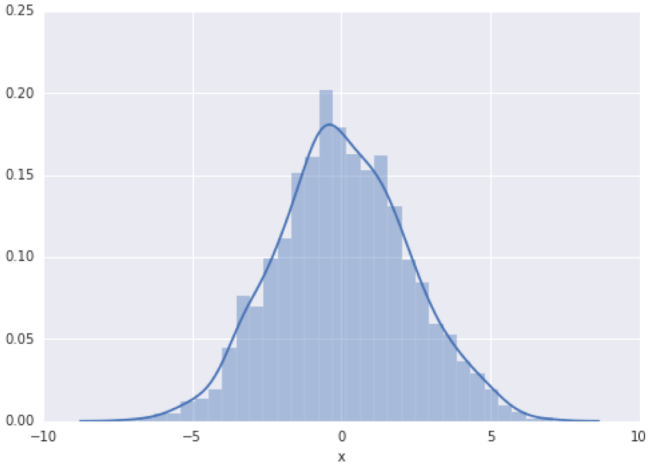


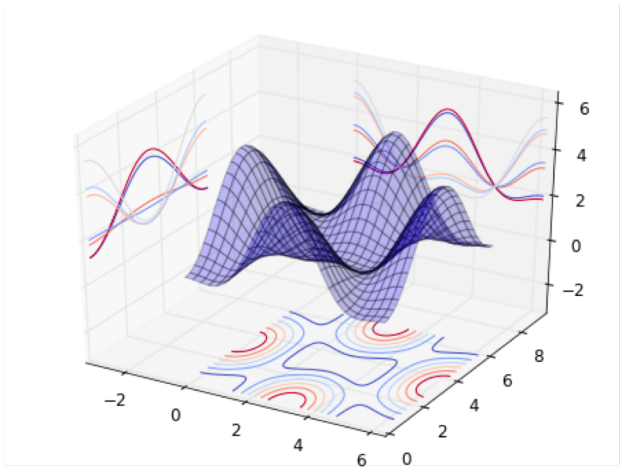


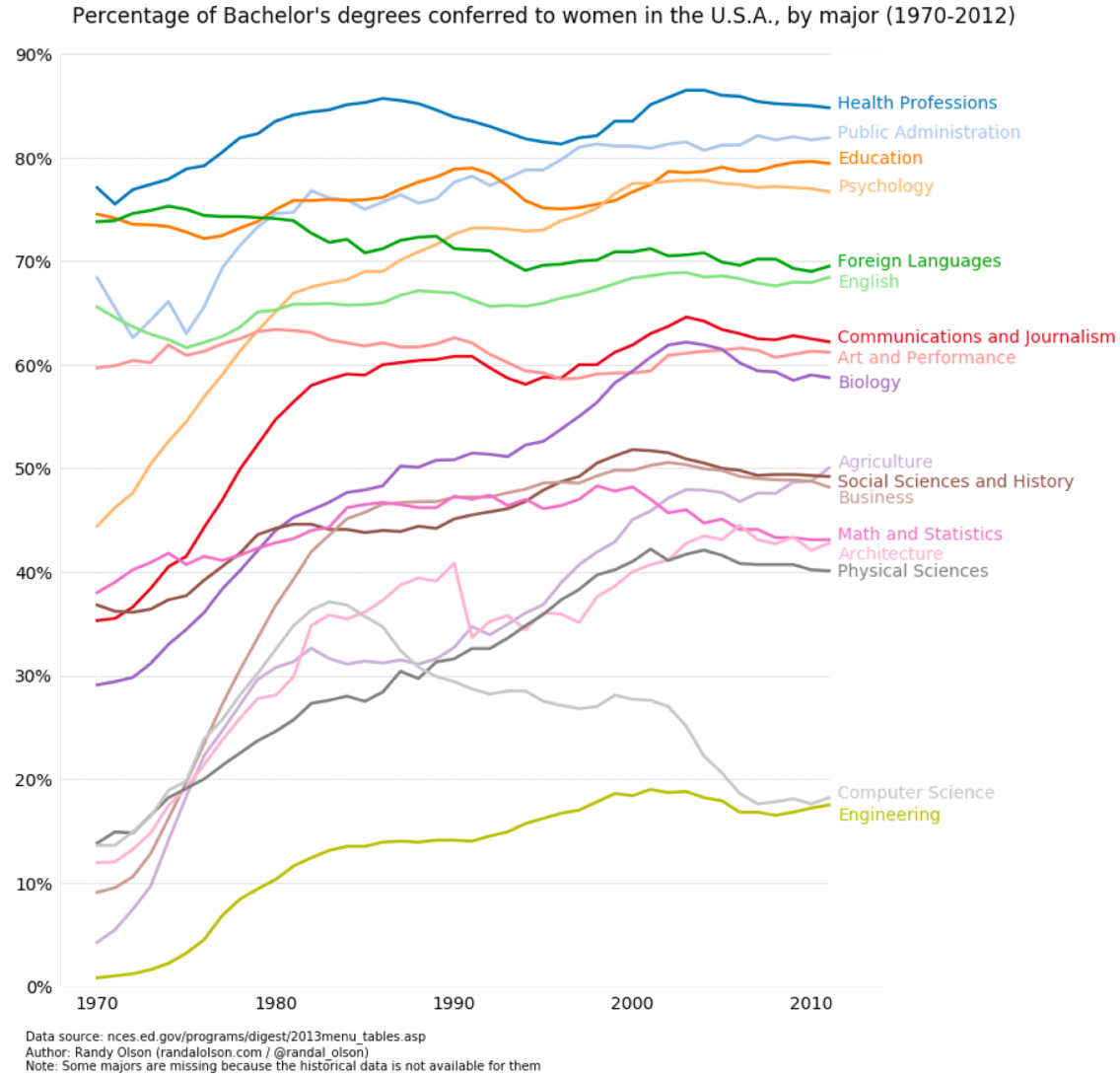
- 26,459 commits made by 911 contributors representing 237,691 lines of code
- Mostly written in Python and C++
- First commit in May, 2003; most recent commit 1 day ago
- Estimated cost $ 3,427,176


pearsonr = 0.64; p = 1.2e-231
x
y


pearsonr = 0.68; p = 6.7e-34
total_bill
tip


x

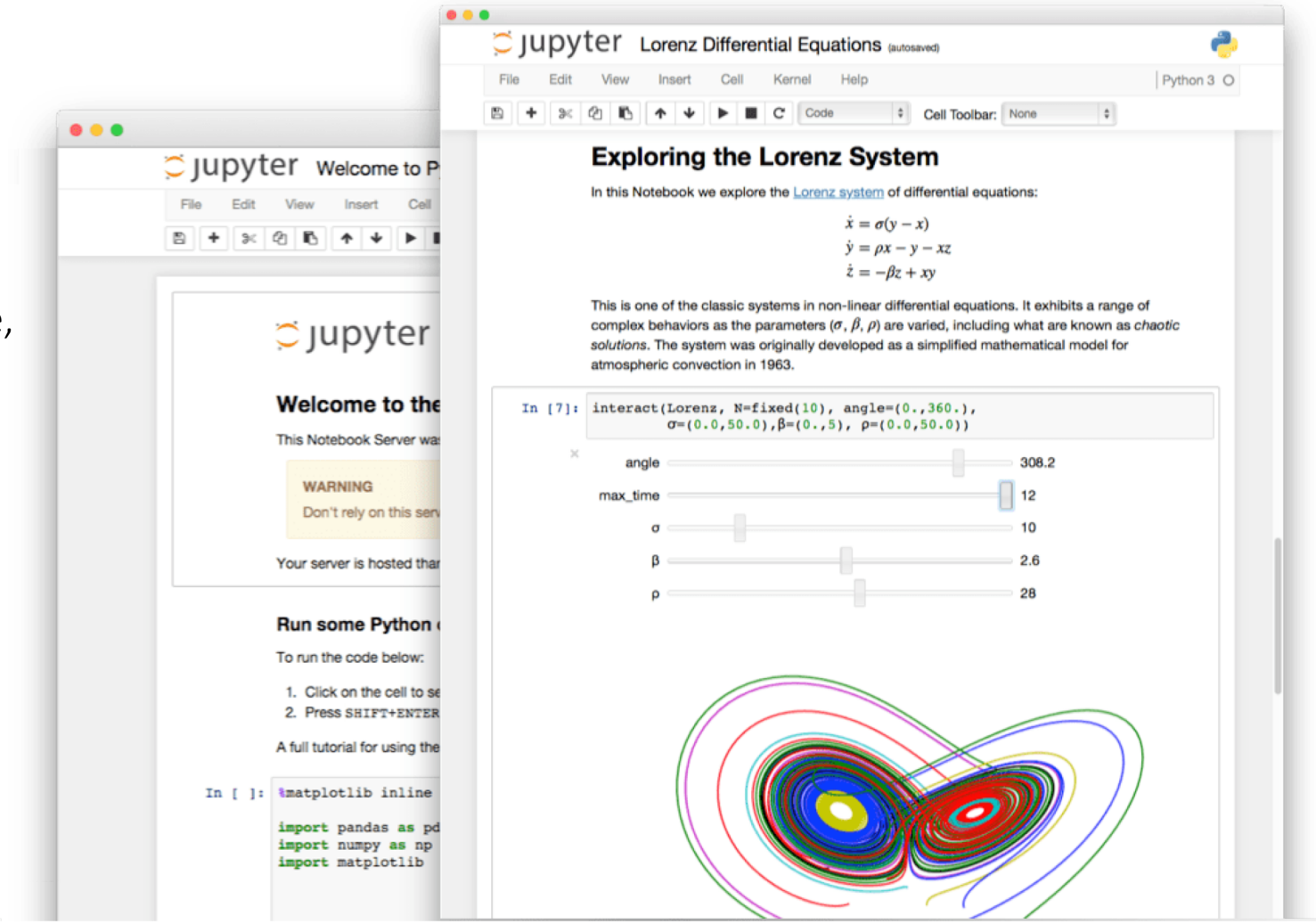



Percentage of Bachelor's degrees conferred to women in the U.S.A., by major (1970-2012)
Health Professions
Public Administration
Education
Psychology
Foreign Languages
English
Communications and Journalism
Art and Performance
Biology
Agriculture
Social Sciences and History
Business
Math and Statistics
Architecture
Physical Sciences
Computer Science
Engineering
Data source: nces.ed.gov/programs/digest/2013menu_tables.asp
Author: Randy Olson (randalolson.com / @randal_olson)
Note: Some majors are missing because the historical data is not available for them

jupyter

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.



- 11,108 commits made by 416 contributors representing 34,256 lines of code
- Written in JavaScript, Python and HTML
- First commit in July, 2005; most recent commit 1 day ago
- Estimated cost $ 445,225

pandas

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

pandas is an open source library providing high-performance, easy-to-use data structures and data analysis tools

- A fast and efficient **DataFrame** object for data manipulation with integrated indexing
- Tools for **reading and writing data** between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format
- Intelligent **data alignment** and integrated handling of **missing data**: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form
- Intelligent label-based **slicing**, **fancy indexing**, and **subsetting** of large data sets
- Aggregating or transforming data with a powerful **group by** engine allowing split-apply-combine operations on data sets
- High performance **merging and joining** of data sets
- **Hierarchical axis indexing** provides an intuitive way of working with high-dimensional data in a lower-dimensional data structure;
- **Time series**-functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging. Even create domain-specific time offsets and join time series without losing data;
- Highly **optimized for performance**, with critical code paths written in Cython or C.
- Python with *pandas* is in use in a wide variety of **academic and commercial** domains, including Finance, Neuroscience, Economics, Statistics, Advertising, Web Analytics, and more.

pandas

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

In [2]:
```
df_1 = pd.read_csv("../data/ozone.csv")
```

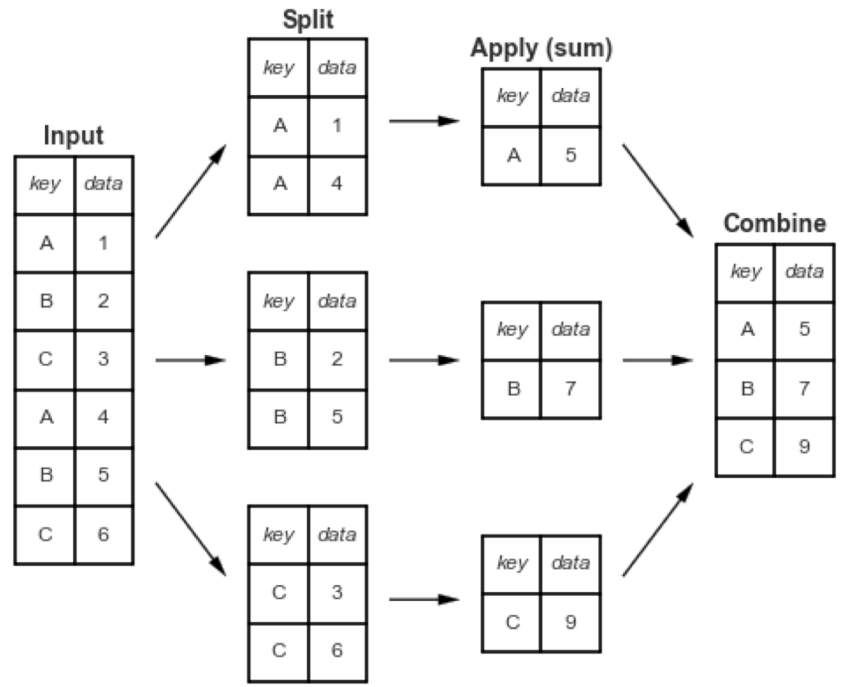
Get a summary of the DataFrame:

In [3]:
```
df_1.describe()
```

Out[3]:

	Ozone	Solar.R	Wind	Temp	Month	Day
count	116.000000	146.000000	153.000000	153.000000	153.000000	153.000000
mean	42.129310	185.931507	9.957516	77.882353	6.993464	15.803922
std	32.987885	90.058422	3.523001	9.465270	1.416522	8.864520
min	1.000000	7.000000	1.700000	56.000000	5.000000	1.000000
25%	18.000000	115.750000	7.400000	72.000000	6.000000	8.000000
50%	31.500000	205.000000	9.700000	79.000000	7.000000	16.000000
75%	63.250000	258.750000	11.500000	85.000000	8.000000	23.000000
max	168.000000	334.000000	20.700000	97.000000	9.000000	31.000000

List the first five rows of the DataFrame:



pandas

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

- 16,901 commits made by 1,296 contributors representing 270,747 lines of code
- Mostly written in Python
- First commit in July, 2009; most recent commit 4 months ago
- Estimated cost $ 3,898,020

SymPy

SymPy is a Python library for symbolic mathematics. It aims to become a full-featured computer algebra system (CAS) while keeping the code as simple as possible in order to be comprehensible and easily extensible. SymPy is written entirely in Python.

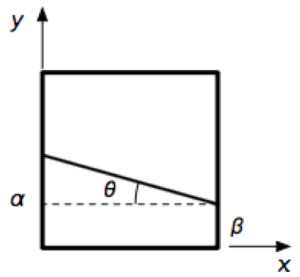
jupyter symblock (autosaved)

File Edit View Insert Cell Kernel Help Python 2

Code Cell Toolbar: None

Three sides wetted

Reorient our point of view to make the analysis easier. Start by assuming that three sides of the block are at least partially immersed. Assume also that $\rho \leq 1/2$ and $\theta \leq 45°$.



In [1]:
```
from sympy import *
init_printing()
```

In [2]:
```
alpha, beta, rho, x, y, theta = symbols('alpha beta rho x y theta')
```

Geometric equations:

In [3]:
```
e1 = rho - (alpha + beta)/2
e2 = tan(theta) - (alpha - beta)
s12 = solve((e1, e2), (alpha, beta))
```

In [4]:
```
s12
```

Out[4]: $\left\{\alpha : \rho + \frac{1}{2}\tan(\theta), \quad \beta : \rho - \frac{1}{2}\tan(\theta)\right\}$

Position of the center of buoyancy, (x, y):

Cython is a programming language that makes writing C extensions for the Python language as easy as Python itself.

Its main feature on top of these is support for optional static type declarations as part of the language. The source code gets translated into optimized C/C++ code and compiled as Python extension modules. This allows for both very fast program execution and tight integration with external C libraries, while keeping up the high programmer productivity for which the Python language is well known.

Installation

```
pip install cython
```

Using inside Jupyter notebook

Load th cythonmagic extension.

In [1]:
```
%load_ext cython
```

Then, simply use the magic function to start writing cython code.

In [2]:
```
%%cython

cdef int a = 0
for i in range(10):
    a += i
print(a)
```

```
45
```

Add `--annotate` or `-a` for showing a code analysis of the compiled code

In [3]:
```
%%cython --annotate

cdef int a = 0
for i in range(10):
    a += i
print(a)
```

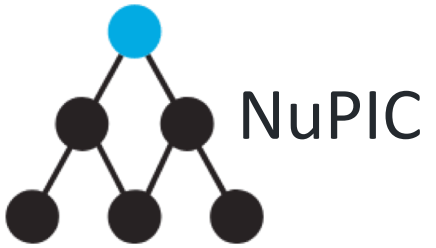
```
45
```

Out[3]:

Generated by Cython 0.25.2

Yellow lines hint at Python interaction.
Click on a line that starts with a "+" to see the C code that Cython generated for it.

```
 1:
+2: cdef int a = 0
+3: for i in range(10):
+4:     a += i
+5: print(a)
```


NuPIC


TensorFlow


theano


scikits
learn


OpenCV


K
Keras


Scrapy


DASK


biopython


scikit-image
image processing in python


Bokeh

Teşekkürler

Development Environments and Distributions

- Plain Python
- Anaconda
- WinPython

- PyCharm
- Sublime
- Visual Studio
- Spyder
- Pydev Eclipse
- IDLE

Resources for Language

Python Official Documentation

Beginning Python From Novice to Professional, Magnus Lie Hetland

Learning Python, Mark Lutz

Problem Solving with Algorithms and Data Structures, Brad Miller & David Ranum

Python Essential Reference, David M. Beazley

Python Cookbook, David M. Beazley

Pro Python, J. Burton Browning & Marty Alchin

Python için Türkçe Kılavuz, Fırat Özgül

Resources for Scientific Computation

Scipy Documentation

Scipy Lecture Notes

Matplotlib

Python For Data Analysis, Wes McKinney (creator of pandas)

Resources – Websites

Runestone Interactive

Automate the Boring Stuff with Python

Full Stack Python

Python Cheatsheet

Resources – People

Raymond Hettinger

Brandon Rhodes

David Beazley

Greg Ward

Alex Martelli

Kenneth Reitz

Kevin London

Ned Batchelder

Yaşar Arabacı

Resources – Events

PyCon

PyData

PyBay

EuroPython

Scipy Conference

DjangoCon

Django Girls