

IP[y]:
IPython

Berkeley
Division of
Data Sciences



How the union of inferential thinking and computation
are transforming research and education at Berkeley.

Fernando Pérez

fernando.perez@berkeley.edu



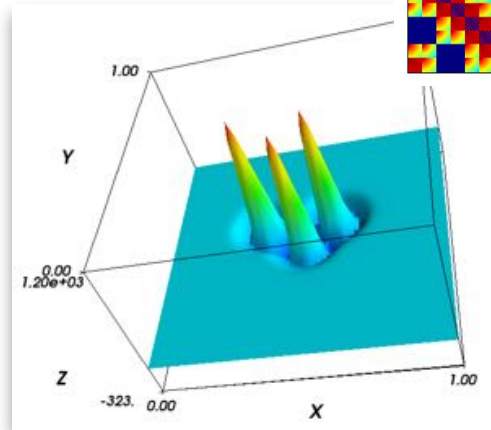
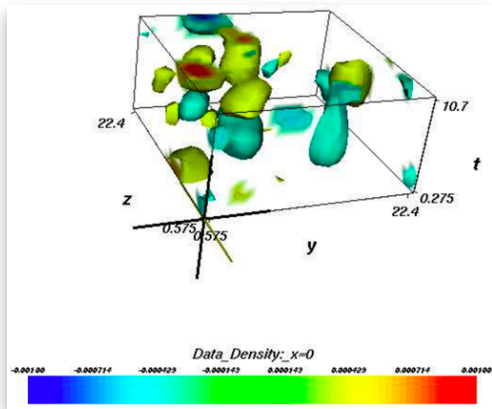
University of California, Berkeley
DEPARTMENT OF STATISTICS



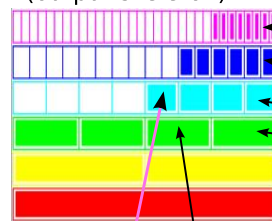
A really odd career



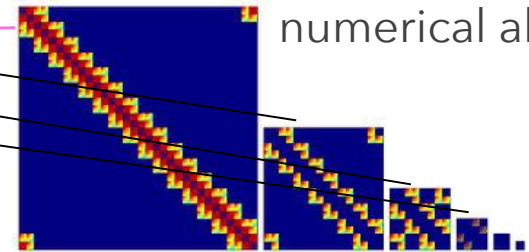
Physics PhD: Lattice QCD
Simulations



Redundant tree of input
(output skeleton)

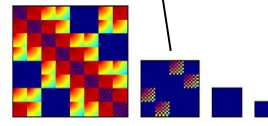


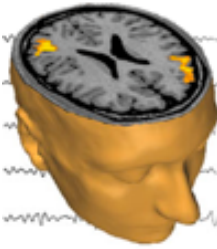
Applied Math Postdoc:
numerical algorithms



Terminal

Non-terminal





BRAIN IMAGING CENTER

UNIVERSITY OF CALIFORNIA, BERKELEY

The Henry H. Wheeler, Jr. Brain Imaging Center (BIC) is one of four Technology Centers established under the auspices of the Helen Wills Neuroscience Institute. It is a campus-wide resource that supports advanced brain imaging technologies dedicated solely to basic brain research.



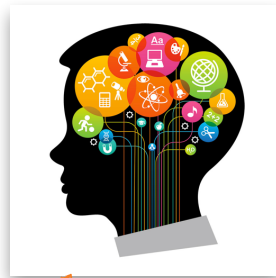
BIDS
BERKELEY INSTITUTE
FOR DATA SCIENCE



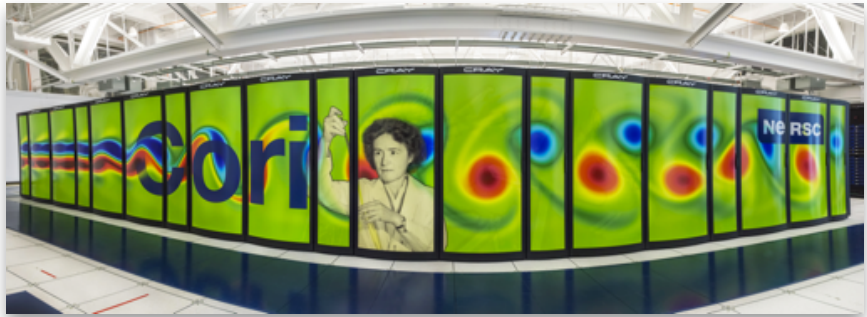
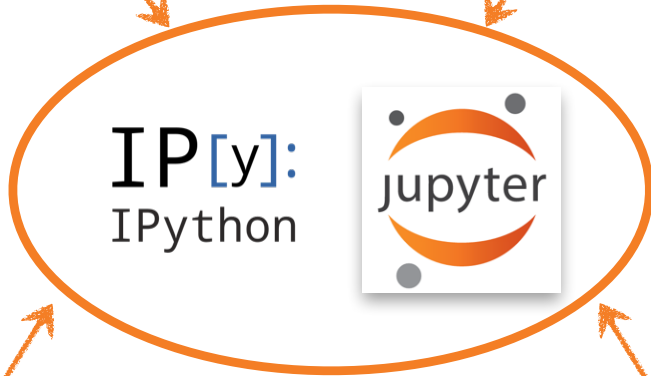
University of California, Berkeley
DEPARTMENT OF STATISTICS

“The purpose of computing is insight,
not numbers”

-Hamming'62



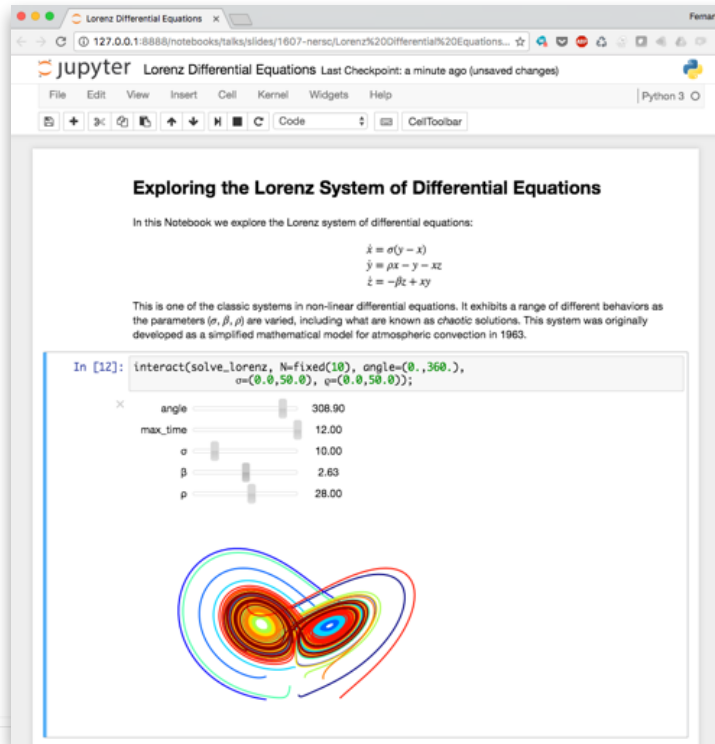
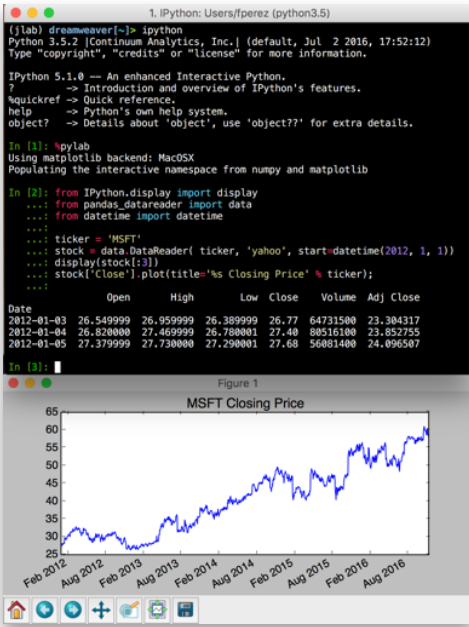
Interactive Computing



What is jupyter

- ❖ Software
- ❖ Standards and Protocols
- ❖ Community

Software



nbviewer

A simple way to share Jupyter Notebooks

Enter the location of a Jupyter Notebook to have it rendered here:

Programming Languages

- Python: IP[y]: IPython Interactive Computing
- Ruby: IRuby: Notebook
- Julia: An Julia Preview

binder (beta)

Turn a Git repo into a collection of interactive notebooks

Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.

Build and launch a repository

GitHub repository name or URL

Git branch, tag, or commit

Path to a notebook file (optional)

Copy the URL below and share your Binder with others:

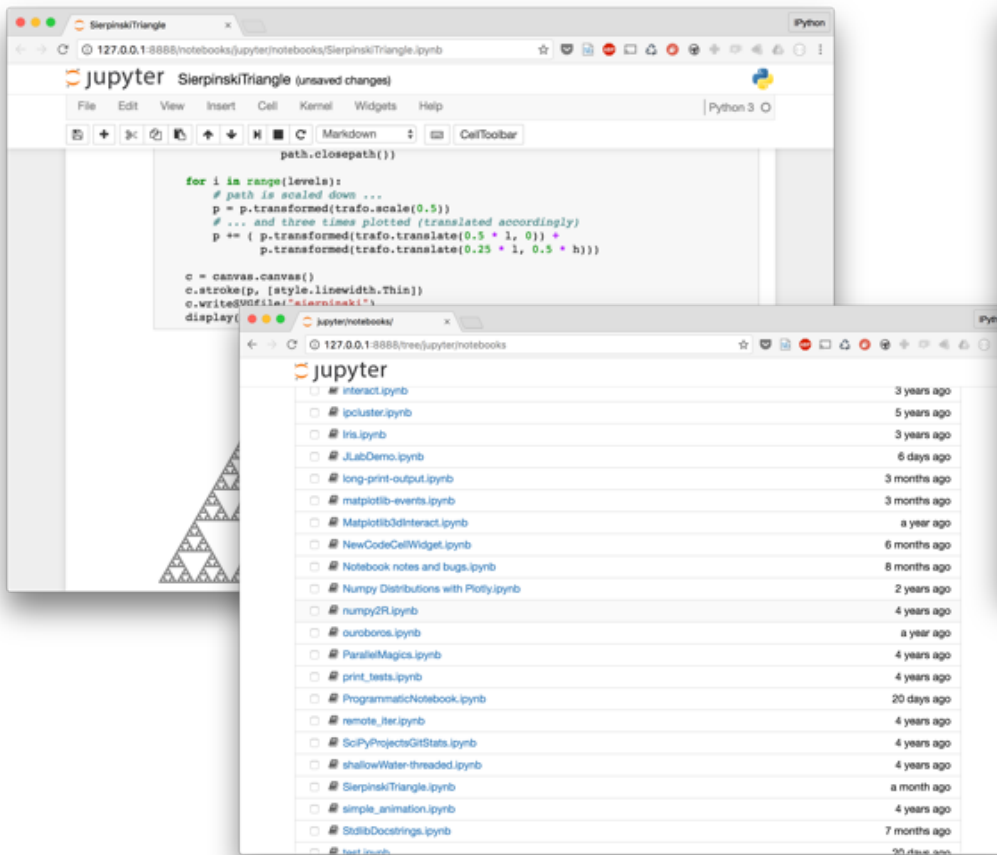
Copy the text below, then paste into your README to show a binder badge:



Jupyter for Organizations

JupyterHub is a multiuser version of the notebook designed for centralized deployments in companies, university classrooms and research labs.

Classic 'Notebook'...

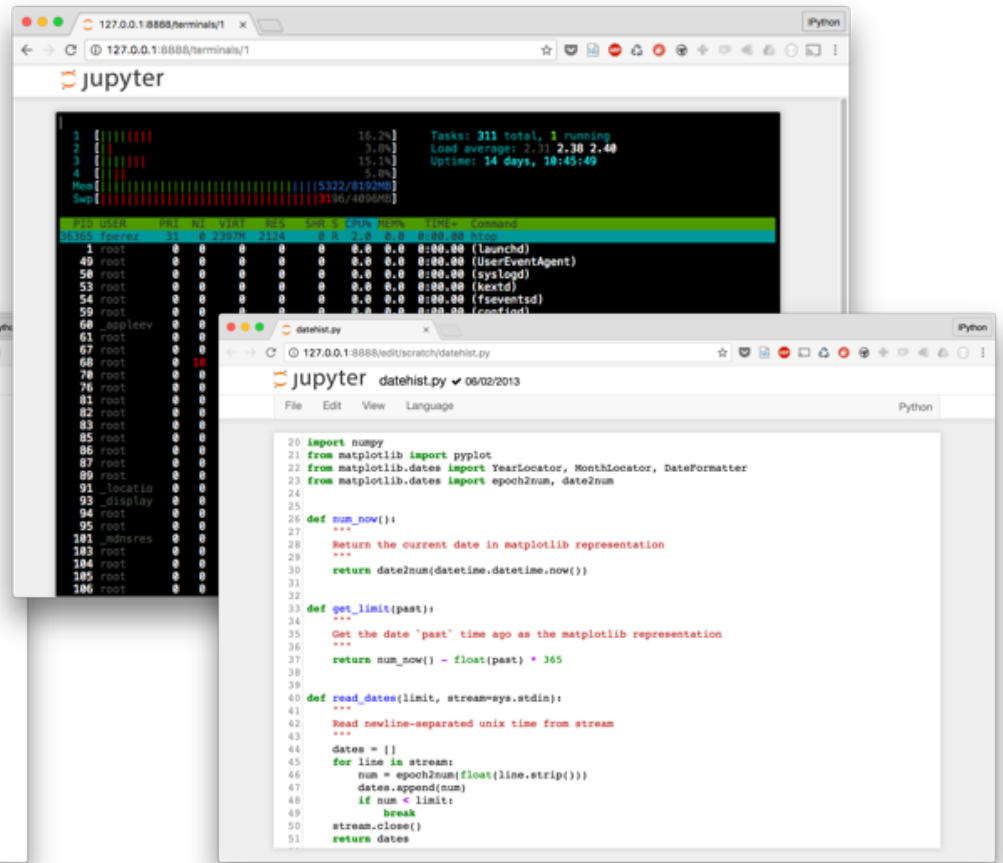


A screenshot of a Jupyter Notebook titled "SierpinskiTriangle". The code defines a function to generate a Sierpinski triangle fractal using a path object and a canvas. The code includes comments explaining the scaling and translation of the path. The output shows a fractal image of a Sierpinski triangle.

```
path.closepath()

for i in range(levels):
    # path is scaled down ...
    p = p.transformed(trrafo.scale(0.5))
    # ... and three times plotted (translated accordingly)
    p += [ p.transformed(trrafo.translate(0.5 * 1, 0)) +
          p.transformed(trrafo.translate(0.25 * 1, 0.5 * h)) ]

c = canvas.canvas()
c.stroke(p, [style.linewidth.Thin])
c.writePNGfile("sierpinski")
display(c)
```



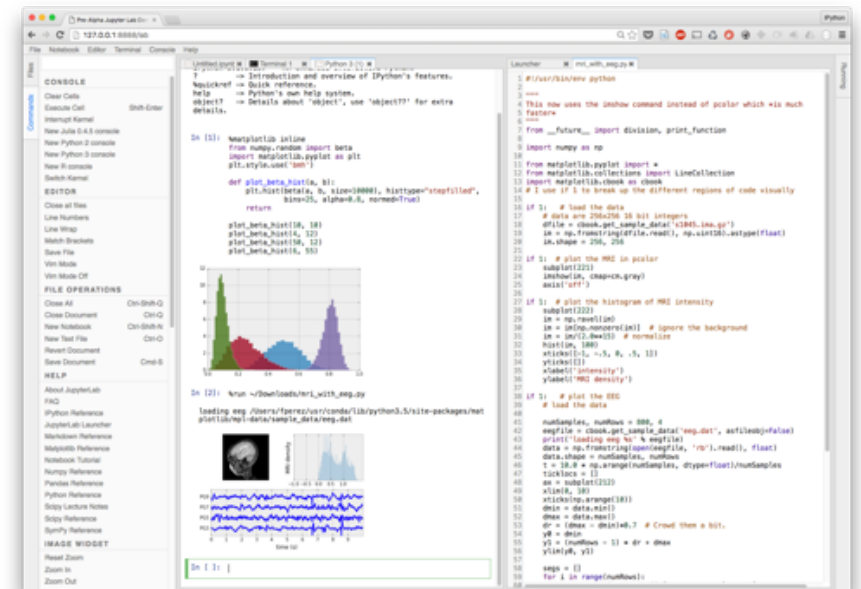
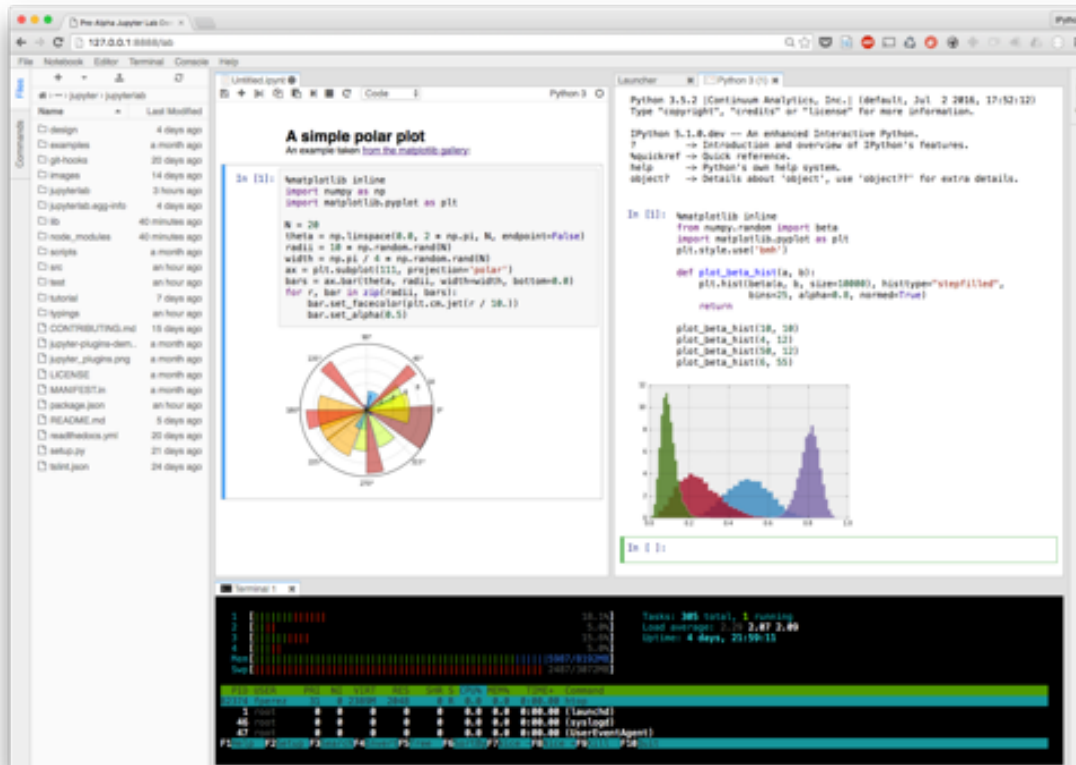
A screenshot of a Jupyter Notebook showing system status and a date histogram. The top part of the notebook displays system information such as tasks, load average, and uptime. The bottom part shows a date histogram with a legend and a plot of data points.

```
1 [|||||] 16.2s] Tasks: 311 total, 1 running
2 [|||||] 7.0s] Load average: 7.3: 2.38 2.48
3 [|||||] 15.1s]
4 [|||||] 5.5s]
Mem [|||||] 5322/8192M]
Swp [|||||] 3196/4899M]

1 root 0 0 0 0 0 0 0 0 0:00.00 (Launched)
49 root 0 0 0 0 0 0 0 0 0:00.00 (UserEventAgent)
59 root 0 0 0 0 0 0 0 0 0:00.00 [syslogd]
53 root 0 0 0 0 0 0 0 0 0:00.00 (kexsd)
54 root 0 0 0 0 0 0 0 0 0:00.00 (fseventsd)
59 root 0 0 0 0 0 0 0 0 0:00.00 (confd)
68 _app/leev 0 0 0 0 0 0 0 0 0:00.00
61 root 0 0 0 0 0 0 0 0 0:00.00
67 root 0 0 0 0 0 0 0 0 0:00.00
66 root 0 10 0 0 0 0 0 0 0:00.00
78 root 0 0 0 0 0 0 0 0 0:00.00
76 root 0 0 0 0 0 0 0 0 0:00.00
82 root 0 0 0 0 0 0 0 0 0:00.00
83 root 0 0 0 0 0 0 0 0 0:00.00
85 root 0 0 0 0 0 0 0 0 0:00.00
86 root 0 0 0 0 0 0 0 0 0:00.00
87 root 0 0 0 0 0 0 0 0 0:00.00
89 root 0 0 0 0 0 0 0 0 0:00.00
91 _locatio 0 0 0 0 0 0 0 0 0:00.00
93 _display 0 0 0 0 0 0 0 0 0:00.00
94 root 0 0 0 0 0 0 0 0 0:00.00
95 root 0 0 0 0 0 0 0 0 0:00.00
101 _adres 0 0 0 0 0 0 0 0 0:00.00
103 root 0 0 0 0 0 0 0 0 0:00.00
104 root 0 0 0 0 0 0 0 0 0:00.00
105 root 0 0 0 0 0 0 0 0 0:00.00
106 root 0 0 0 0 0 0 0 0 0:00.00
```

```
20 import numpy
21 from matplotlib import pyplot
22 from matplotlib.dates import YearLocator, MonthLocator, DateFormatter
23 from matplotlib.dates import epoch2num, date2num
24
25
26 def num_now():
27     """
28     Return the current date in matplotlib representation
29     """
30     return date2num(datetime.datetime.now())
31
32
33 def get_limit(past):
34     """
35     Get the date 'past' time ago as the matplotlib representation
36     """
37     return num_now() - float(past) * 365
38
39
40 def read_dates(limit, stream=sys.stdin):
41     """
42     Read newline-separated unix time from stream
43     """
44     dates = []
45     for line in stream:
46         num = epoch2num(float(line.strip()))
47         dates.append(num)
48         if num < limit:
49             break
50     stream.close()
51     return dates
```


JupyterLab: a grand unified theory of Jupyter



Huge Team Effort!



C. Colbert, S. Corlay, A. Darian, B. Granger, J. Grout, P. Ivanov, I. Rose, S. Silvester, C. Willing, J. Zosa-Forde ...

Notebooks++

The screenshot displays a Jupyter Notebook interface with the following components:

- File Explorer:** Shows the file `Lorenz.ipynb`.
- Code Editor:**
 - Contains the title **The Lorenz Differential Equations**.
 - Text: "Before we start, we import some preliminary libraries. We will also import (below) the accompanying `lorenz.py` file, which contains the actual solver and plotting routine."
 - Code cell 1:

```
In [1]: from IPython import interactive, fixed
```
 - Text: "We explore the Lorenz system of differential equations:"
 - Equations:
$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xy \\ \dot{z} &= -\beta z + xy \end{aligned}$$
 - Text: "Let's change (σ, β, ρ) with `ipywidgets` and examine the trajectories."
 - Code cell 2:

```
In [2]: from IPython import interactive, fixed
w = interactive(solve_lorenz, sigma=(0.9, 50.0), rho=(0.0, 50.0))
```
- Output View:**
 - Shows three interactive sliders for parameters: `sigma` (10.10), `beta` (2.63), and `rho` (23.30).
 - Displays a 3D plot of the Lorenz attractor.
- PDF Viewer:** Shows a preview of the notebook content as a PDF document.

Beyond notebooks

The screenshot shows the JupyterLab interface with two tabs: 'Launcher' and 'markdown_python.m'. The 'Launcher' tab shows the source markdown file with the following content:

```
1 # Markdown
2
3 This is a regular markdown file. In JupyterLab you can open and
4 edit
5 the file in the file editor, or in the markdown viewer. As you
6 edit the
7 file the rendered markdown will automatically update.
8
9 $$ k+\alpha+ \sum_{i=0} y_i $$
10
11 # Including Python code
```

The 'markdown_python.m' tab shows the rendered markdown file. It displays the text from the source file, the rendered mathematical equation $x + x + \sum_{i=0}^n y_i$, and the Python code block. Below the code block, the output of the code is shown as a table:

```
[3]:
```

	x	y	color	size
0	0.617536	0.508976	0.945603	8.052391
1	0.051126	0.322041	0.237476	49.005570
2	0.247221	0.476783	0.876143	61.492632
3	0.713989	0.480705	0.462035	7.631572
4	0.312106	0.888199	0.640439	53.201591

The rendered markdown file also includes the following text and code:

Markdown

This is a regular markdown file. In JupyterLab you can open and edit the file in the file editor, or in the markdown viewer. As you edit the file the rendered markdown will automatically update.

$$x + x + \sum_{i=0}^n y_i$$

Including Python code

Here is a block of Python code in the markdown file:

```
a = 100
```

Let's attach a Python 3 Kernel and Console to this markdown file. Then we can select lines of code in the markdown file and run them in the console by pressing `Shift+Enter`. Let's do something more complicated:

First import `matplotlib`, `numpy` and `pandas`, and create a data frame:

```
%matplotlib inline
from matplotlib import pyplot as plt
from matplotlib import style
import numpy as np
import pandas as pd
data = {
    'x': np.random.rand(100),
    'y': np.random.rand(100),
    'color': np.random.rand(100),
    'size': 100.0*np.random.rand(100)
}
df = pd.DataFrame(data)
df.head()
```

Ln 7, Col 4 Spaces: 4 markdown_python.md

Data

The screenshot displays a Jupyter Notebook environment with three main components:

- Launcher:** Shows a large image of the Helix Galaxy.
- iris.csv:** A CSV file containing data for the Iris dataset. The data is as follows:

	sepal_len	sepal_wid	petal_len	petal_wid	species
1	5.1	3.5	1.4	0.2	se
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	2.4	1.4	0.3	setosa
- Museums_in_DC:** A JSON array of museum data points. The first two entries are:

```
1. [{"type": "FeatureCollection", "features": [{"type": "Feature", "properties": {"OBJECTID": 1, "ADDRESS": "716 MONROE STREET NE", "NAME": "AMERICAN POETRY MUSEUM", "ADDRESS_ID": 389744, "LEGALNAME": "HERITAGE US", "ALTNAM": "AMERICAN POETRY MUSEUM", "WEBURL": "http://americanpoetrymuseum.org/"}, "geometry": {"type": "Point", "coordinates": [-76.99588783568, 38.932842879235]}}, {"type": "Feature", "properties": {"OBJECTID": 2, "ADDRESS": "719 6TH STREET NW", "NAME": "GERMAN-AMERICAN HERITAGE MUSEUM", "ADDRESS_ID": 238849, "LEGALNAME": "CORCORAN GALLERY OF ART", "ALTNAM": " ", "WEBURL": "http://gahmusa.org/"}, "geometry": {"type": "Point", "coordinates": [-77.01958878328639, 38.89911961896782]}}, {"type": "Feature", "properties": {"OBJECTID": 3, "ADDRESS": "1387 NEW HAMPSHIRE AVENUE NW", "NAME": "HEURICH HOUSE FOUNDATION", "ADDRESS_ID": 741958, "LEGALNAME": "U.S. DEPARTMENT OF THE"}]}]
```

A 3D visualization of a fly brain, rendered in a purple color, showing the complex structure of the central nervous system.

JupyterLab is extensible: FlyBrainLab

An Interactive Computing Platform for the Fly Brain

A diagram of a neural network with numerous nodes connected by lines, representing the connectivity of the fly brain.

BIONET Group, Columbia University

<http://www.bionet.ee.columbia.edu>

A 3D visualization of a fly brain with neural activity highlighted in green and yellow, showing the distribution of neurons and their connections.

Aurel A. Lazar (PI)

Tingkai Liu

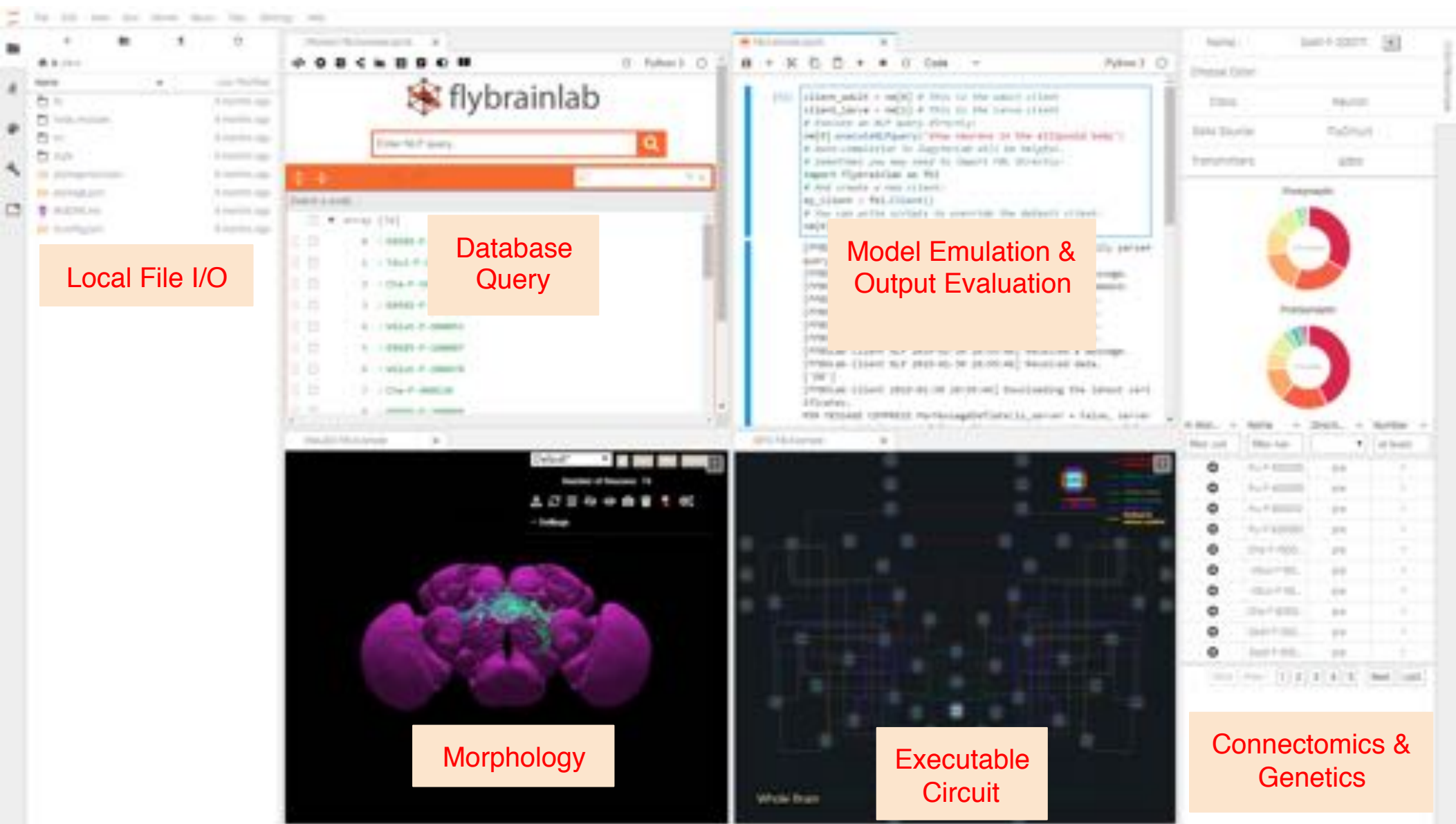
Mehmet K. Turkcan

Chung-Heng Yeh

Yiyin Zhou

A 3D visualization of a fly brain with neural activity highlighted in orange and red, showing the distribution of neurons and their connections.

<http://fruitflybrain.org>



Local File I/O

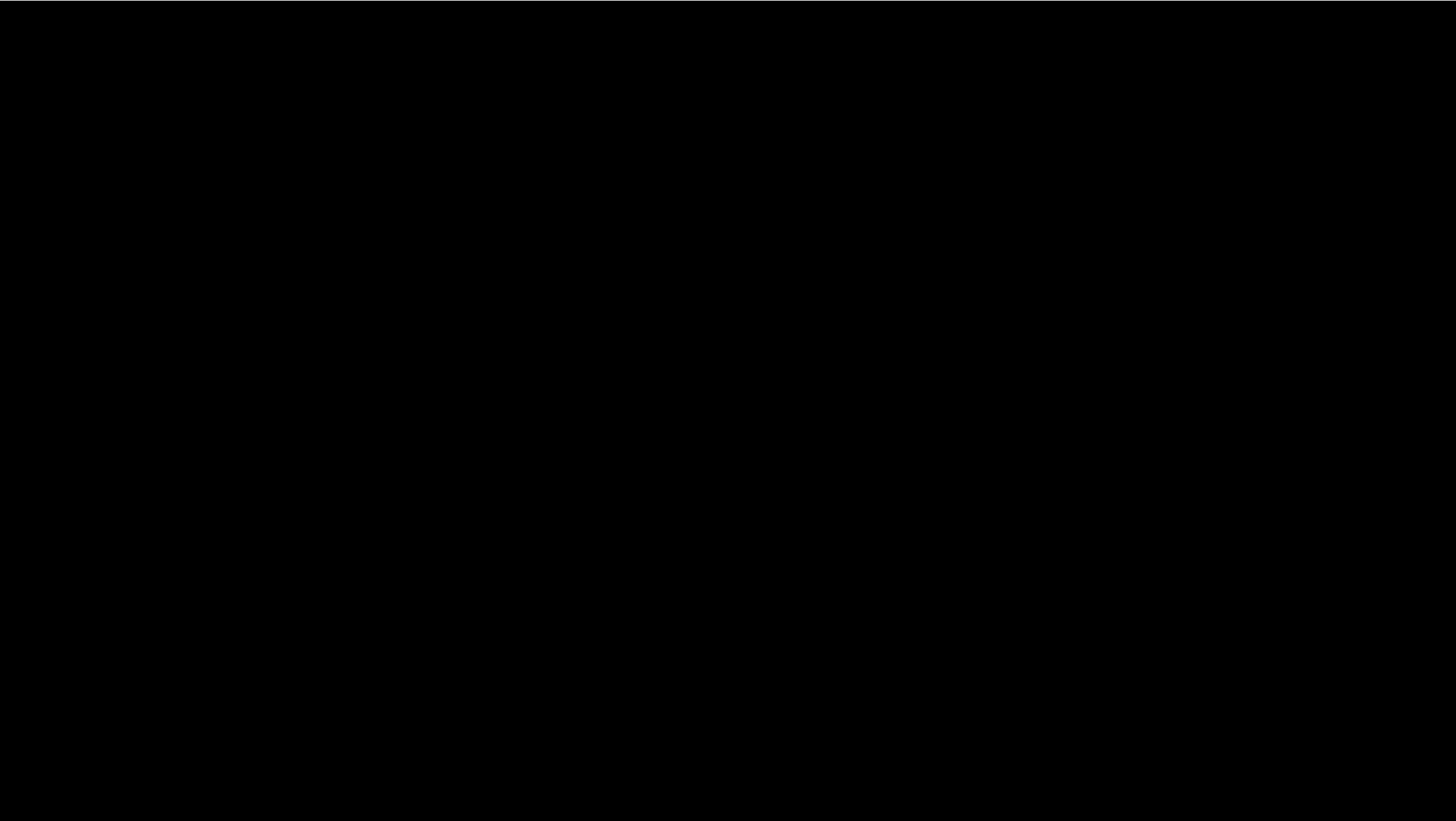
Database Query

Model Emulation & Output Evaluation

Morphology

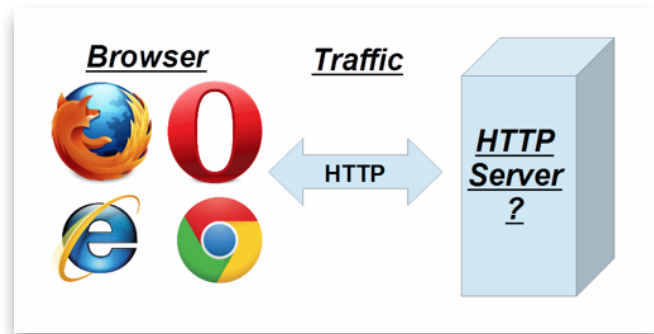
Executable Circuit

Connectomics & Genetics



Standards and Protocols

Core ideas of the web: HTTP & HTML



HTTP: protocol to connect clients and servers
HyperText Transport Protocol

Image credit: eviltester.com

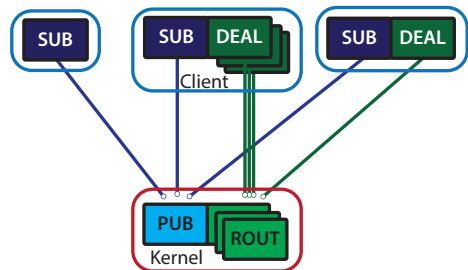
```
<div id="page-wrapper">  
  <div id="main" class="clearfix">  
    <div id="content" class="column" role="main"><div id="content-inner" class="clearfix">  
      <a id="main-content"></a>  
      <h1 class="title" id="page-title">Statistics at UC Berkeley</h1>
```



HTML: format to represent content
HyperText Markup Language

Core ideas of Jupyter

Interactive Computing Protocol



ØMQ + JSON

Document Format

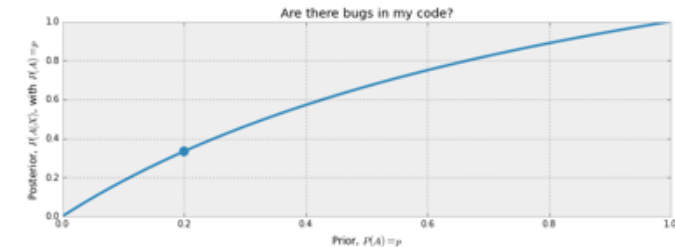
We have already computed $P(X|A)$ above. On the other hand, $P(X| \sim A)$ is subjective: our code can pass tests but still have a bug in it, though the probability there is a bug present is reduced. Note this is dependent on the number of tests performed, the degree of complication in the tests, etc. Let's be conservative and assign $P(X| \sim A) = 0.5$. Then

$$P(A|X) = \frac{1 \cdot p}{1 \cdot p + 0.5(1 - p)}$$
$$= \frac{2p}{1 + p}$$

This is the posterior probability. What does it look like as a function of our prior, $p \in [0, 1]$?

```
figsize(12.5, 4)
p = np.linspace(0, 1, 50)
plt.plot(p, 2 * p / (1 + p), color="#348ABD", lw=3)
# plt.fill_between(p, 2*p/(1+p), alpha=.5, facecolor=["#A60620"])
plt.scatter(0.2, 2 * (0.2) / 1.2, s=140, c="#348ABD")
plt.xlim(0, 1)
plt.ylim(0, 1)
plt.xlabel("Prior, P(A) = p")
plt.ylabel("Posterior, P(A|X) with P(A) = p")
plt.title("Are there bugs in my code?")
```

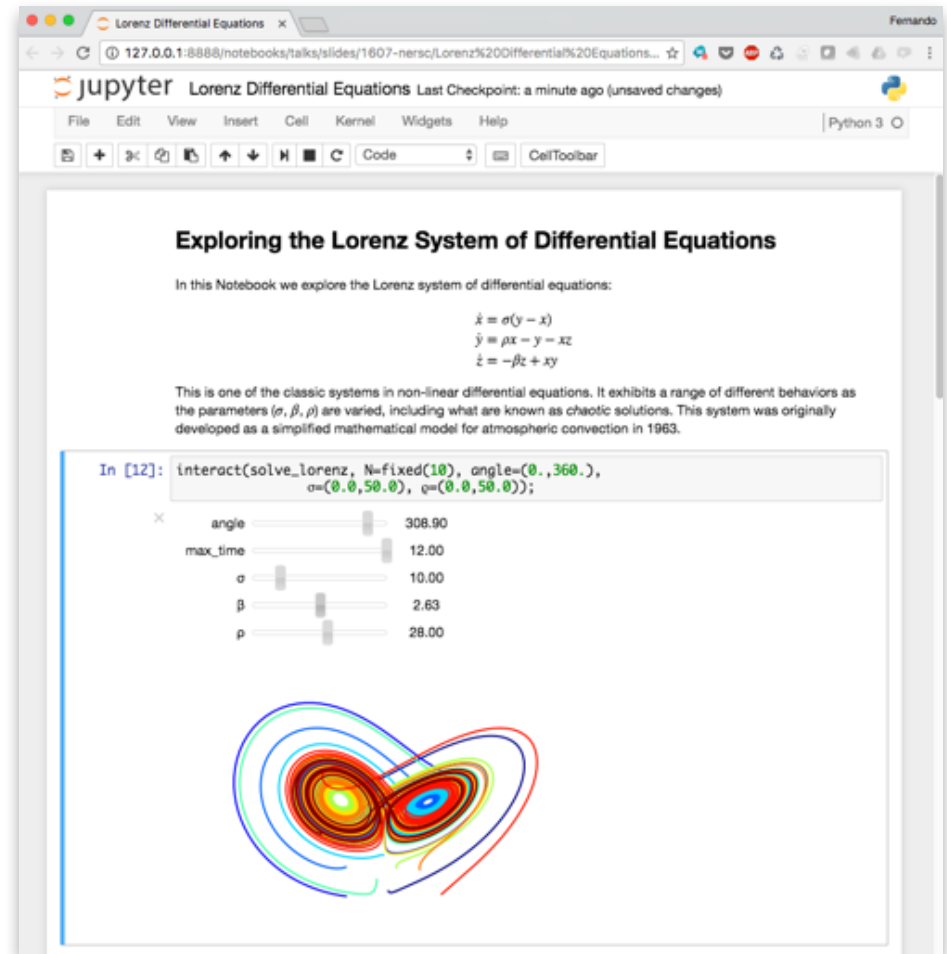
<matplotlib.text.Text at 0x1051de650>



<https://github.com/CamDavidsonPilon/Probabilistic-Programming-and-Bayesian-Methods-for-Hackers>

The Notebook as document & format


- ❖ JSON specification
- ❖ Machine readable
- ❖ Metadata-rich



New tools built atop the format

Jupyter Book

nbgrader: homework assignments



Multiple Categories

https://www.inferentialthinking.com/chapters/11/2/Multiple_Categories

Introduction

Search

1. Data Science
2. Causality and Experiments
3. Programming in Python
4. Data Types
5. Sequences
6. Tables
7. Visualization
8. Functions and Tables
9. Randomness
10. Sampling and Empirical Distributions
11. Testing Hypotheses
- 11.1 Assessing Models
- 11.2 Multiple Categories
- 11.3 Decisions and Uncertainty
12. Comparing Two Samples
13. Estimation
14. Why the Mean Matters
15. Prediction
16. Inference for Regression
17. Classification
18. Updating Predictions

Powered by Jupyter Book

```
jury = Table().with_columns(  
    'Ethnicity', make_array('Asian', 'Black', 'Latino', 'White',  
    'Eligible', make_array(0.15, 0.18, 0.12, 0.54, 0.01),  
    'Panels', make_array(0.26, 0.08, 0.08, 0.54, 0.04)  
)
```

Ethnicity	Eligible	Panels
Asian	0.15	0.26
Black	0.18	0.08
Latino	0.12	0.08
White	0.54	0.54
Other	0.01	0.04

jury.barh("Ethnicity")

Comparison with Panels Selected at Random

What if we select a random sample of 1,453 people from the population of eligible jurors? Will the distribution of their ethnicities look like the distribution of the panels above?

Jupyter Problem 1 Last Checkpoint: a few seconds ago (autosaved)

File Edit View Insert Cell Kernel Help Python 3

Part A (2 points)

Write code to compute the mean of a list of numbers.

```
In [ ]: def mean(x):  
    """Compute the mean of a list of numbers given in 'x'"""  
    ## BEGIN SOLUTION  
    return sum(x) / len(x)  
    ## END SOLUTION
```

```
In [ ]: """Check that the 'mean' function is correct."""  
assert mean([1]) == 1.0  
assert mean([1, 2]) == 1.5  
assert mean([5.5, 0, 2, 3.4]) == 2.725  
assert mean(range(100)) == 49.5  
assert mean(range(100, 0, -1)) == 50.5
```

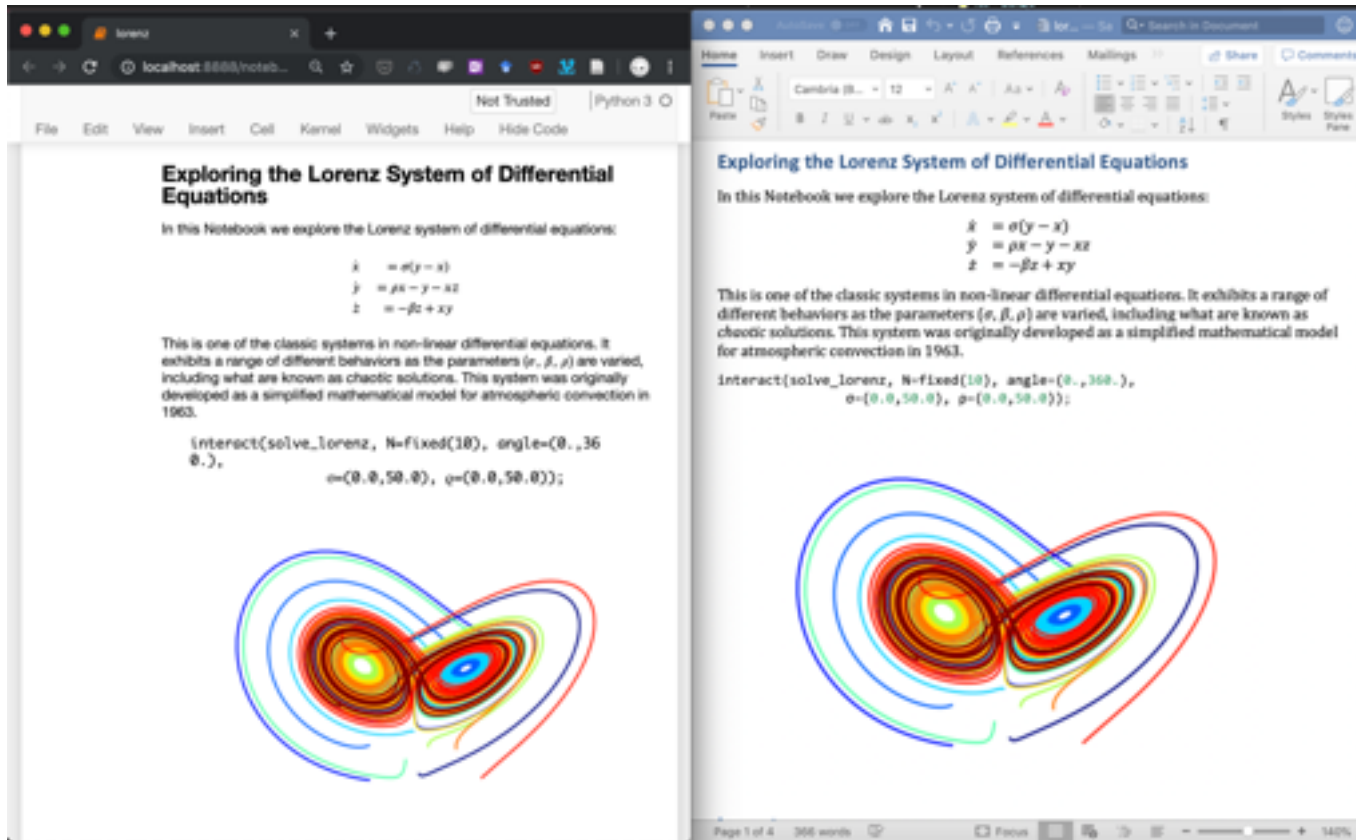
Part B (3 points)

Describe the difference between an arithmetic mean, a harmonic mean, and a geometric mean.

Arithmetic mean:

$$\frac{1}{N} \sum x_i$$

pandoc: ipynb to word (and more)



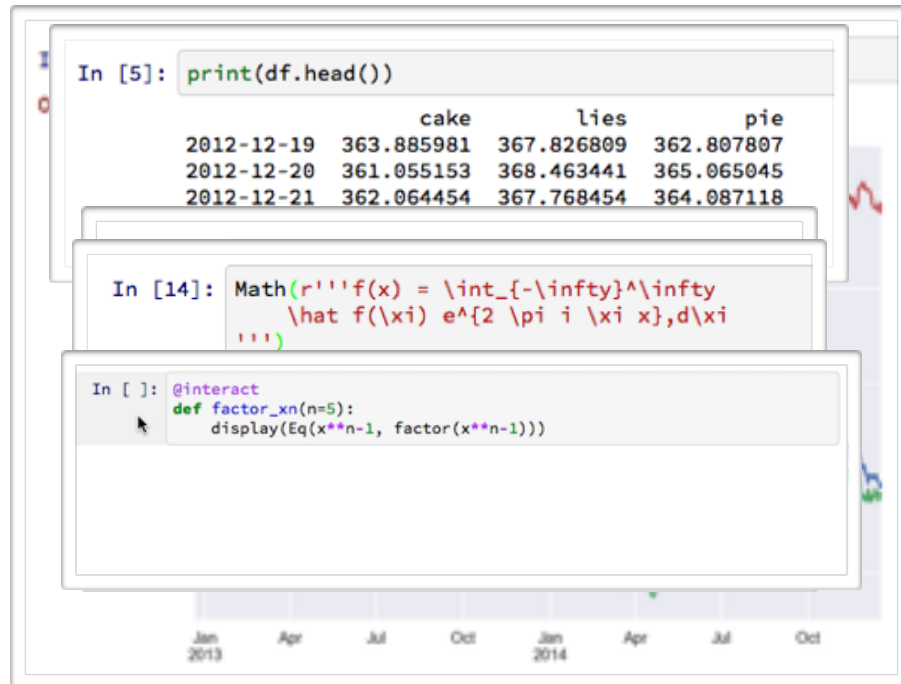
pandoc lorenz.ipynb -o lorenz.docx

Jupyter Protocol

web-age capture of the process of interactive computing

any mime-type output

- ❖ text
- ❖ svg, png, jpeg
- ❖ latex, pdf
- ❖ html, javascript
- ❖ interactive widgets



A language agnostic protocol



~100 different kernels: <https://github.com/jupyter/jupyter/wiki/Jupyter-kernels>

Community

IPython: an afternoon hack, 2001

```
ipython-0.0.1.py x
32 Globals for SI units (including g=9.8) : _load_units = %(_load_units)s
33 Starting number for prompt counter     : _prompt_ini = %(_prompt_ini)s
34 Number of history items to store in cache : _cache_size = %(_cache_size)s
35
36
37 # Configure here
38 _load_Numeric = 1
39 _load_Gnuplot = 1
40 _load_gracePlot = 1
41 _load_units = 1
42 _cache_size = 1000
43 _prompt_ini = 1
44
45 # ** Don't modify below unless you know what you're doing. **
46
47 # Crude first version, with minimal object structure. This could be done much
48 # better, by defining a Cache class (probably using weak references or
49 # generators). But it seems to work ok. Haven't checked for memory circularity
50 # problems, though.
51
52 #*****
53 # Copyright (C) 2001 Fernando P@rez. <fperez@pizero.colorado.edu>
54 #
55 # Distributed under the terms of the GNU General Public License.
56 #
57 # The full text of the GPL is available at:
58 #
59 # http://www.gnu.org/copyleft/gpl.html
60 #*****
61 __author__ = 'Fernando P@rez. <fperez@pizero.colorado.edu>'
62 __version__ = '0.1'
63
64 #*****
65 # Class definitions
66
67 class HistPromot1:
68     """Simple interactive prompt like Mathematica's."""
69     def __str__(self):
70         return '\nIn[*]_prompt_count[*]:'
71
72 class HistPromot2:
73     """Simple interactive continuation prompt."""
74     def __str__(self):
75         return '...'+ '*(len('In[*]_prompt_count[*]')== ')-3)
76
77 #*****
78 # Function definitions
79
80 def _history_print(arg):
81     """Printing with history cache management.
82     This is invoked everytime the interpreter needs to print, and is activated
83     by setting the variable sys.displayhook to it."""
84
85     global _p, _pp, _ppp, _cache, _prompt_count
```

```
1. IPython: Users/fperez (python3.5)
(jlab) dreamweaver[~]> ipython
Python 3.5.2 [Continuum Analytics, Inc.] (default, Jul 2 2016, 17:52:12)
Type "copyright", "credits" or "license" for more information.

IPython 5.1.0 -- 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.

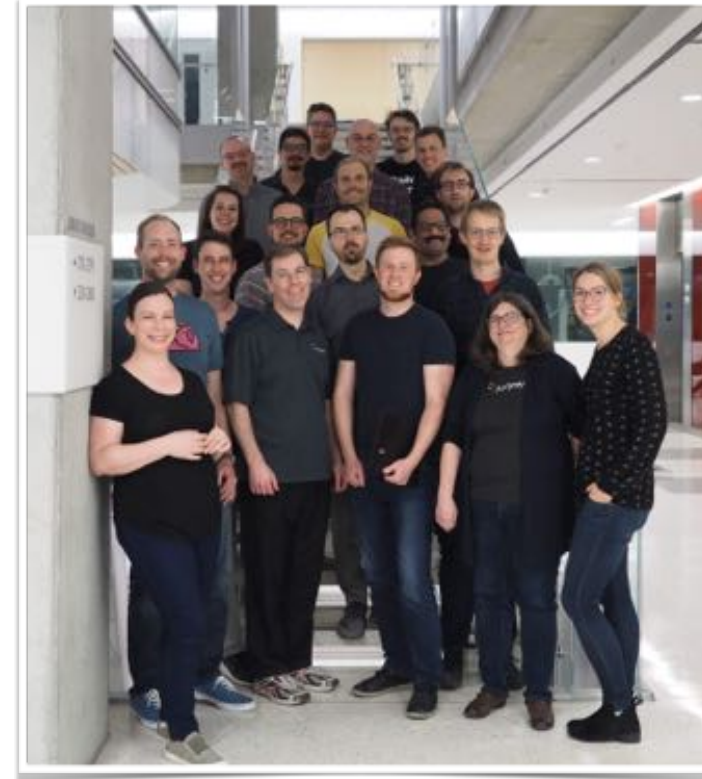
In [1]: %pylab
Using matplotlib backend: MacOSX
Populating the interactive namespace from numpy and matplotlib

In [2]: from IPython.display import display
...: from pandas_datareader import data
...: from datetime import datetime
...:
...: ticker = 'MSFT'
...: stock = data.DataReader(ticker, 'yahoo', start=datetime(2012, 1, 1))
...: display(stock[:3])
...: stock['Close'].plot(title='%s Closing Price' % ticker);
...:
Date                Open          High          Low  Close    Volume  Adj Close
2012-01-03    26.549999    26.959999    26.389999    26.77    64731500    23.304317
2012-01-04    26.820000    27.469999    26.780001    27.40    80516100    23.852755
2012-01-05    27.379999    27.730000    27.290001    27.68    56081400    24.096507

In [3]:

Figure 1
MSFT Closing Price
65
60
55
50
45
40
35
30
25
Feb 2012 Aug 2012 Feb 2013 Aug 2013 Feb 2014 Aug 2014 Feb 2015 Aug 2015 Feb 2016 Aug 2016
```

A true team effort



Plus ~ 1500 more Open source contributors!

Formalized governance



Steering Council

The role of the Jupyter Steering Council is to ensure, through working with and serving the broader Jupyter community, the long-term well-being of the project, both technically and as a community. The Jupyter Steering Council currently consists of the following members (in alphabetical order).



Damian Avila
Anaconda, Inc.
[@damianavila](#) on GitHub



Matthias Bussonnier
UC Merced
[@Carreau](#) on GitHub

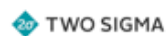


Sylvain Corlay
QuantStack
[@sylvaincorlay](#) on GitHub



Institutional Partners

Institutional Partners are organizations that support the project by employing Jupyter Steering Council members. Current Institutional Partners include:



Sponsors

Project Jupyter receives direct funding from the following sources:

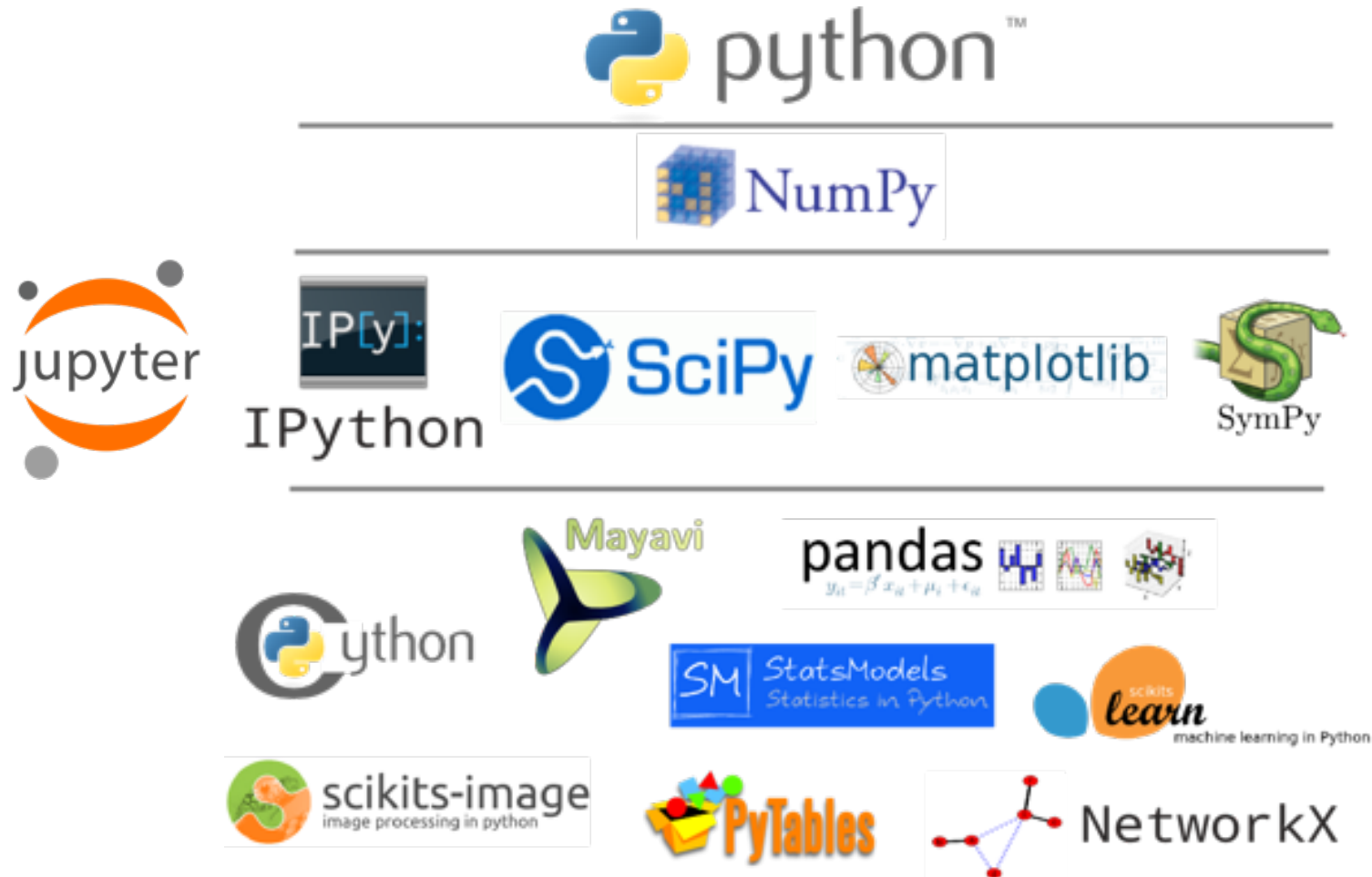


SCHMIDT FUTURES

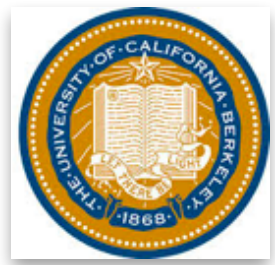
Formal fiscal sponsorship



Jupyter: gateway to an ecosystem



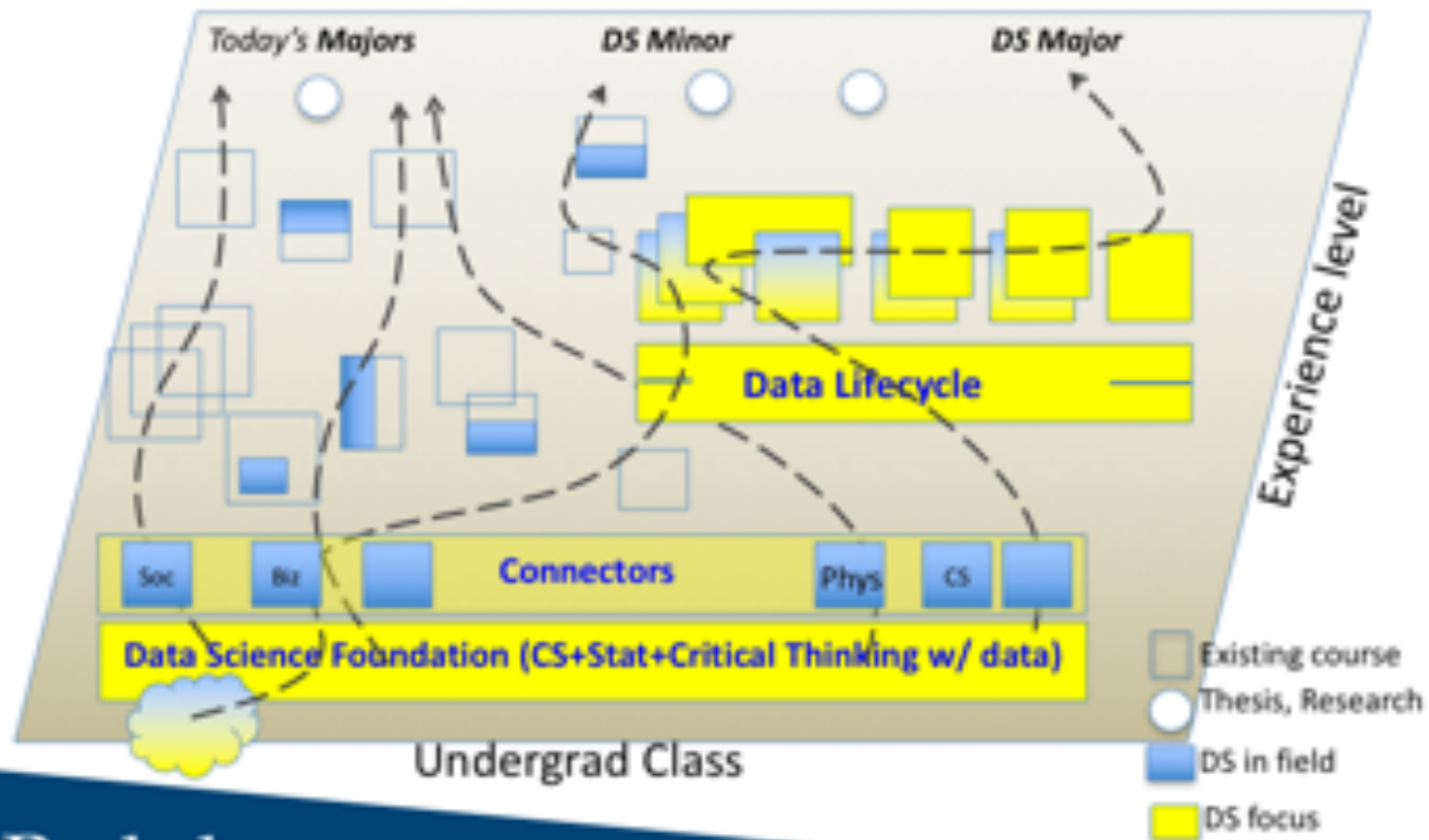
Educational impact:
a view from Berkeley



Chancellor's directive, June 2014

... rethinking at a fundamental level what **every educated person** must know about **quantitative reasoning**: how to effectively **understand, process and interpret information**, to inform decisions in their professional and personal lives and as citizens of the world in the 21st century.

Undergrad experience transformed



Data 8: Foundations of Data Science

- ❖ **Purpose:** “You will learn the **core concepts** of inference and computing, while working hands-on with **real data**.”
- ❖ **A Data Science Course for Everyone:** “designed for **entry-level students from any major**, specifically for students who have not previously taken statistics or computer science courses.”
- ❖ **Topics:** “all the key ideas of **introductory statistics** in a new, modern, hands-on way. It weaves in contextual issues like **data privacy and bias**. At the same time, it gives you a powerful understanding of **key ideas in computing**.”

<http://inferentialthinking.com>



The screenshot shows a web browser window with the URL <https://www.inferentialthinking.com/chapters/intro>. The page title is "Computational and Inferential Thinking". The main heading is "The Foundations of Data Science" by Anil Adhikari and John DeNero. The page includes a sidebar with a "DATA 8" logo and a navigation menu. The main content area provides information about the textbook, including its authors, contributors, and a link to view it online on GitHub Pages. A license notice for Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) is also present.

Secure | <https://www.inferentialthinking.com/chapters/intro>

Computational and Inferential Thinking

The Foundations of Data Science

By [Anil Adhikari](#) and [John DeNero](#)

Contributions by [David Wagner](#) and Henry Milner

This is the textbook for the [Foundations of Data Science class at UC Berkeley](#).

[View this textbook online on GitHub Pages.](#)

Old versions of this textbook: [Fall 2017](#)

The contents of this book are licensed for free consumption under the following license: [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International \(CC BY-NC-ND 4.0\)](#)

ON THIS PAGE

THE FOUNDATIONS OF DATA SCIENCE

DATA 8

HOME

0. INTRODUCTION

1. DATA SCIENCE

- 1.1 Introduction
 - 1.1.1 Computational Tools
 - 1.1.2 Statistical Techniques
- 1.2 Why Data Science?
- 1.3 Plotting the Classics
 - 1.3.1 Literary Characters
 - 1.3.2 Another Kind of Character

Data 100: [ds100.org](https://www.textbook.ds100.org)

The screenshot shows a web browser window with the URL <https://www.textbook.ds100.org>. On the left is a sidebar with a table of contents. The main content area features a 'TOGGLE SIDEBAR' button, the title 'Principles and Techniques of Data Science', authors 'Sam Lau, Joey Gonzalez, and Deb Nolan', and introductory text about the course and license. A 'Next' button is visible at the bottom of the main content area.

Data 100 Homepage

Introduction

About This Book

1. The Data Science Lifecycle
2. Data Design
3. Tabular Data and pandas
4. Exploratory Data Analysis
5. Data Cleaning
6. Data Visualization
7. Web Technologies
8. Working With Text
9. Databases and SQL
10. Modeling and Estimation
11. Gradient Descent
12. Probability and Generalization
13. Linear Regression
14. Feature Engineering
15. Bias-Variance Tradeoff
16. Regularization
17. Classification

← TOGGLE SIDEBAR

Principles and Techniques of Data Science

By [Sam Lau](#), [Joey Gonzalez](#), and [Deb Nolan](#).

This is the textbook for [Data 100](#), the [Principles and Techniques of Data Science](#) course at UC Berkeley.

Data 100 is the upper-division, semester-long data science course that follows [Data 8](#), the [Foundations of Data Science](#). The reader's assumed background is detailed in the [About This Book](#) page.

The contents of this book are licensed for free consumption under the following license: [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International \(CC BY-NC-ND 4.0\)](#)

To set up the textbook for local development, see the [the setup guide](#).

Next >

Live textbooks

Visualizing Categorical Distributions

Interact

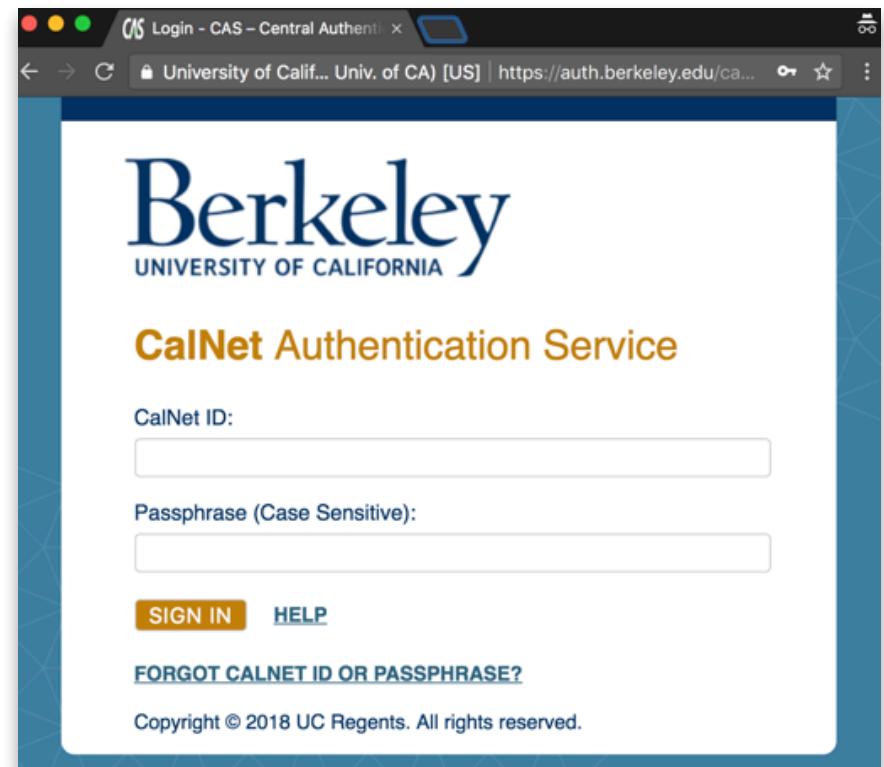
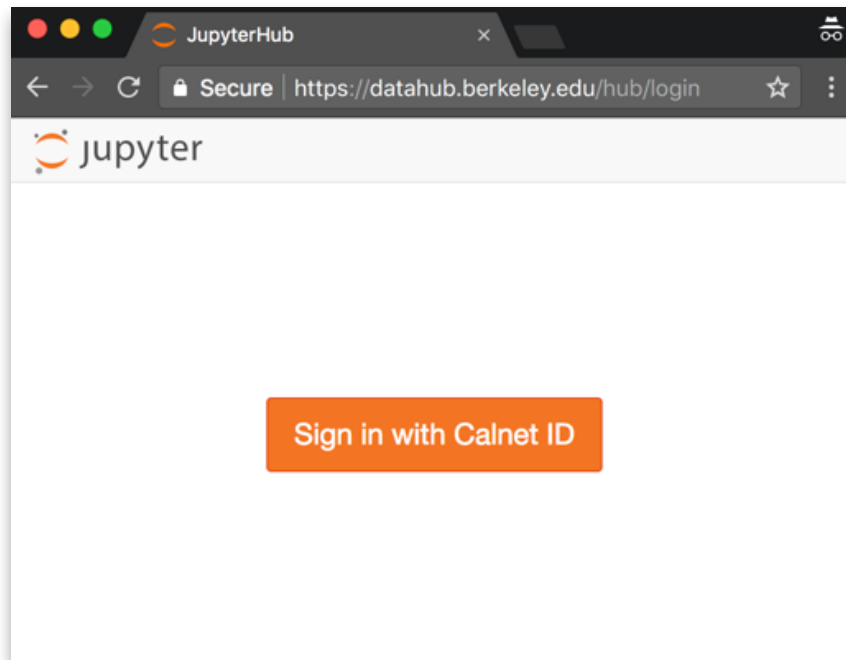
Data come in many forms that are not numerical. Data can be pieces of music, or places on a map. They can also be categories into which you can place individuals. Here are some examples of *categorical* variables.

- The individuals are cartons of ice-cream, and the variable is the flavor in the carton.
- The individuals are professional basketball players, and the variable is the player's team.
- The individuals are years, and the variable is the genre of the highest grossing movie of the year.
- The individuals are survey respondents, and the variable is the response they choose from among "Not at all satisfied," "Somewhat satisfied," and "Very satisfied."

The table `icecream` contains data on 30 cartons of ice-cream.

```
icecream = Table().with_columns(  
  'Flavor', make_array('Chocolate', 'Strawberry', 'Vanilla'),  
  'Number of Cartons', make_array(16, 5, 9)  
)  
icecream
```

Berkeley: datahub.berkeley.edu



Fall 2018



Data 8:
~1,300
students

Data 100:
~800 students



Fastest growing courses in Berkeley history

Data 8 in Fall 2018

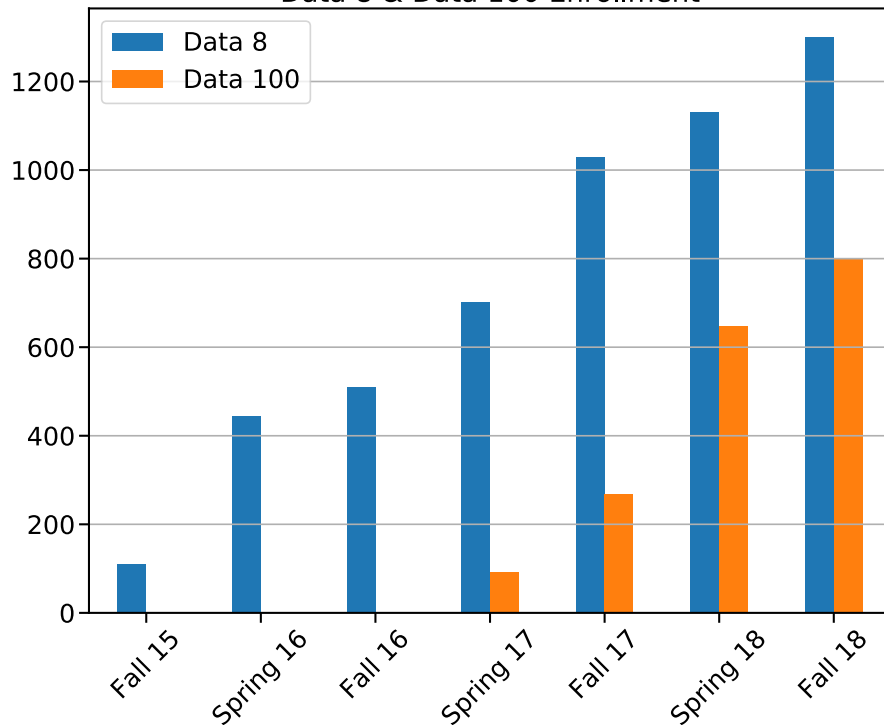
- ❖ ~ 1,300 enrolled students
- ❖ ~ 200 waitlisted

Annual combined numbers

- ❖ Data 8: ~ 3,000 students
- ❖ UC Berkeley: ~ 7,500

At steady state, will easily reach ~50% of campus!

Data 8 & Data 100 Enrollment



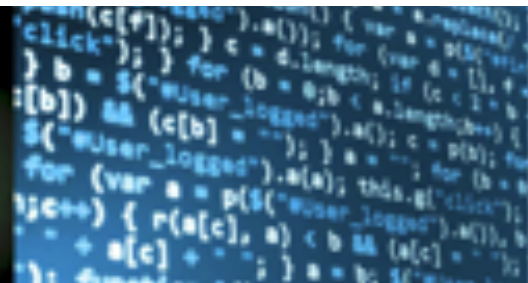
<http://data8.org> - <http://ds100.org>



LEGALST 88
Taking Measure of the Justice System



STAT 88
Probability & Mathematical Statistics



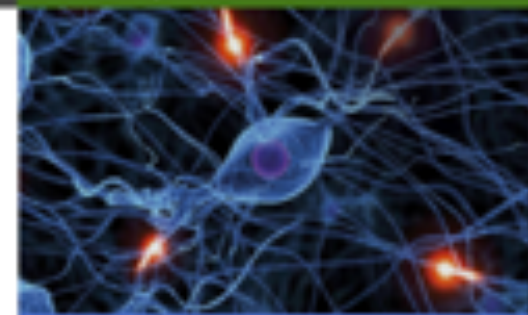
COMPSCI 88
Computational Structures in DS



COGSCI 88
Data Science & the Mind

DATA SCIENCE CONNECTOR COURSES

Companion Courses for Data 8



PSYCH 88
DS for Cognitive Neuroscience



UGBA 96-4 & 5
Data & Decisions



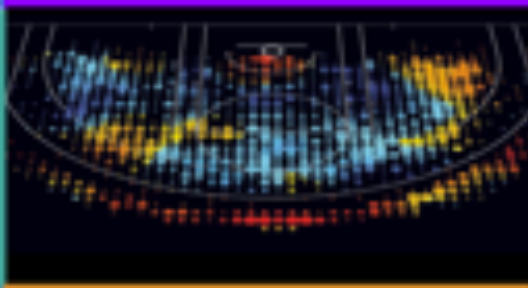
STAT 89A
Linear Algebra for Data Science



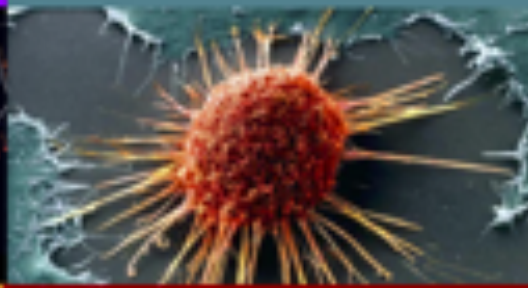
GEOG 88
Data Science Applications in Geography



L&S 88-1
Children in the Developing World



L&S 88-2
Sports Analytics



MCB 88
Immunotherapy of Cancer

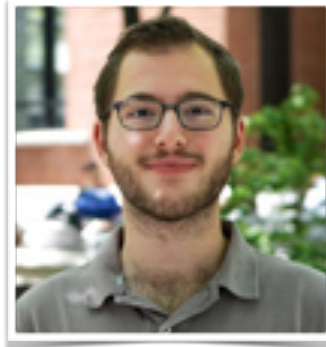
Modules

Fall 2017 Data Science Modules

	INFO 290 Privacy & Security Lab		PSYCH 192 Special Topics		EDUC 223B Special Problems in STEM Education		ART W23AC Data Arts
	IAS 150 International Area Studies		ETHSTD 21AC Immigration & Detention		SOCIOL 130AC Social Inequalities		XRHETOR R1A The Craft of Writing
	EPS C20 Earthquakes in Your Backyard		GWS 131 Gender and Science		ENGLISH 31AC Literature of American Cultures		COGSCI 1B Intro to Cognitive Science
	ISF 189 Intro to Interdisciplinary Studies		NESTUD 190A Text & Web Analysis for Social Sciences		LINGUIS 110 Intro to Phonetics & Phonology		GWS 101 Gender and Women's Studies

Repro. research course: STAT 159/259

- ❖ **Schedule:** 2 Lectures (80 min), 1 lab (2 h)
- ❖ **Prerequisites:** foundations in computation, probability and statistical modeling
- ❖ **Enrollment:** 50 undergrads, 10 grads
- ❖ **Graduate Student Instructor:** Eli Ben-Michael, Stats PhD student
- ❖ **Grading:** weekly readings, quizzes, homework and 3 projects.



<http://bit.ly/stat159-f17>

Goals

❖ **What?**

- ❖ Core ideas: data access, computation, statistical analysis and publication.

❖ **Why?**

- ❖ An essential concern of modern computational research.
- ❖ **Social** and **scientific** implications of lack of reproducibility.
- ❖ Frame the problem in terms **practical**, **ethical** and **epistemological**.

❖ **How?**

- ❖ **Skills** and **habits** necessary to make a practice of reproducibility.
- ❖ An everyday practice, **not a “publication time” concept**.

Core skills

- ❖ **Version control:** Git and GitHub
- ❖ **Programming:** Python
- ❖ **Process automation:** Make
- ❖ **Data analysis:** Numpy, Pandas, Matplotlib, NLTK, Scikit-Learn, ...
- ❖ **Documentation:** Sphinx
- ❖ **Software testing:** PyTest
- ❖ **Continuous Integration:** Travis
- ❖ **Reproducible containers:** Binder

Git and Python workflow everywhere

Stat 159/259 - Reproducible and Collaborative Data Science

Navigation

An interactive Git Tutorial: the tool you didn't know you needed

A quick overview of the Jupyter Notebook and IPython

Reading discussion - Developing open source scientific practice

Reading discussion - Scientific Python, IPython, Jupyter

Stat 159/259 - Reproducible and Collaborative Data Science

All materials for this course are [available on GitHub](#).

The [class syllabus](#) will be updated over the course of the first couple of weeks of class.

Readings

See [here](#) for a list of assigned class readings.

Lectures

- [An interactive Git Tutorial: the tool you didn't know you needed](#)
- [A quick overview of the Jupyter Notebook and IPython](#)
- [Reading discussion - Developing open source scientific practice](#)
- [Reading discussion - Scientific Python, IPython, Jupyter](#)
- [Class practice: strings, lists & numbers](#)
- [Conda and pip - managing environments](#)
- [From September 25 reading](#)
- [Make: automating tasks](#)
- [LIGO: the 2017 Nobel prize in physics, and wrapping up Makefiles](#)



berkeley-stat159-f17 / stat159-f17

112 commits 2 branches 0 releases 3 contributors

File	Description	Latest commit	Time ago
.static/ref	Add .static directory	c885004	7 months ago
labs	Add files I accidentally forgot to put into git		4 months ago
lectures	add chaos notes		4 months ago
syllabus	Add syllabus.		7 months ago
.gitignore	ignore cache in git operations		4 months ago
LICENSE	Add license file		7 months ago
Makefile	Add makefile to refresh slideshows		4 months ago
README.md	Make public website		7 months ago
conf.py	Further warnings fixes/cleanups		4 months ago
conftest.py	Properly handle general skip patterns, not just specific paths.		4 months ago
environment.yml	Add environment.yml file with explicit versions of key packages		4 months ago
instructor	add chaos notes		4 months ago
readings.rst	Add files I accidentally forgot to put into git		4 months ago
resources.md	Update numpy exercise		5 months ago

STAT 159/259 - Reproducible and Collaborative Data Science

Materials for the Fall 2017 edition of UC Berkeley's STAT 159/259 - Reproducible and Collaborative Data Science course.

Live website is [available here](#).

Computational hygiene: a daily habit



A screenshot of the GitHub Classroom interface. The header shows "GitHub Classroom" and "Stat 159/259 Fall 2017 Edition - Reproducible Data Science". Below the header, there is a "Manage classroom" button. The main content area is titled "Assignments" and lists several assignments with their titles, descriptions, and "Copy invitation link" buttons. The assignments are: "Homework 1" (Group assignment for 2-person Teams for Homework 1), "Project #1 - Replicate results of Laken & Strodal 2016" (Group assignment for 2-person Teams for Homework 1), "Quiz 2" (Individual assignment), "Project #2 - Analyzing the State of the Union" (Group assignment for 3-person teams for Project 2), "Homework 3: dataset selection for Project #3" (Group assignment for 3-person teams for Project 2), "Graduate project - Election 2000" (Individual assignment), and "Project #3 - Original data analysis" (Group assignment for 3-person teams for Project 2).

Final Project: an original analysis

- ❖ **Data:** included in repo or linked if too large.
- ❖ **Clean, tested code.**
- ❖ **Analysis notebooks and supporting code**
 - ❖ Break down your analysis into as many notebooks as is reasonable for convenient reading and execution.
- ❖ **Main narrative notebook:** summarizes and discusses results.
- ❖ **Reproducibility support:** `Makefile` and `environment.yml`
- ❖ **Good repository practices:** `README.md`, `LICENSE`, `.gitignore`.
 - ❖ Use Victoria Stodden's ENABLING REPRODUCIBLE RESEARCH: LICENSING SCIENTIFIC INNOVATION.

A "Standard
Playbook"

Brief Analysis on the Marginal Effects of Studying

[build passing](#) [launch binder](#)

As students, we have often wondered what effect an extra hour of studying will have on our grades. When trying to determine whether staying up an extra hour to study for that final exam is truly worth it, we usually are limited by imperfect information and our own superstitions. In this project, we attempt to estimate the "true" marginal effect of studying on students' grades. We try to model the effects of studying first using OLS and then various instruments and 2 stage least squares. This repository is also meant to serve as an example of what a reproducible econometric analysis would look like.

Required Installations

The only installation needed to run this repo is Anaconda. Click [here](#) to learn about how to install Anaconda. Once installed, you should be good to go!

Using Binder

We've enabled Binder for this project which allows you to view jupyter notebooks in an executable environment. Feel free to click the link at the top of this README to launch the binder.

Getting Started

Download the repo onto your local machine and open your command prompt. Simply type in the following commands to run the analysis:

```
make clean
make env
source activate study
make run
```

After all your notebooks have run you should see new files in the results, fig, and data directories. Read about our approach and results in main.ipynb. All the figures from our analysis are saved in the fig directory and our regressions are saved in the results directory as dataframes. You can load in these dataframes and work with them as regression instances (i.e. you can call `.summary()`, `.params()` etc. click [here](#) for OLS documentation and [here](#) for 2SLS documentation)

Licensing

In an effort to enable reproducible, collaborative research our project is subject to the MIT License which allows you to modify and distribute the above code for both private and commercial usage. See LICENSE to learn more.

nadvtdedlis Merge pull request #27 from berkeley-stat159-f17/nadv_actual_final Latest	
data	added reproducibility aspects, split model fitting into 2 ntbs; NOTE...
fig	Fix typos in data_exploration.ipynb
results	Fix typos in model_fitting_2.ipynb
.gitignore	Add caches to .gitignore
.mailmap	adding mailmap to account for config issues
.travis.yml	Add pandas install to Travis
LICENSE	Added LICENSE
Makefile	added reproducibility aspects, split model fitting into 2 ntbs; NOTE...
README.md	added reproducibility sentence to README
data_exploration.ipynb	Minor add to data_exploration.ipynb
environment.yml	added reproducibility aspects, split model fitting into 2 ntbs; NOTE...
instructions.md	Add note about grades in team work
main.ipynb	correction to instruments justification
model_fitting_1.ipynb	Fix typos in model_fitting_1.ipynb
model_fitting_2.ipynb	Fix typos in model_fitting_2.ipynb
p3functions.py	Add two_way function
tests.py	Add two_way function

Analysis notebooks

While these histograms give us some information about the distributions of these individual variables, they don't help with understanding how these variables interact with our dependent variable G3. So lets look at some violin plots to visualize some of these interactions.

For the violin plots we split G3 into 5 bins to more clearly visualize the interactions. We also show the distributions relative to which school the students come from to determine whether there is a difference in the two schools.

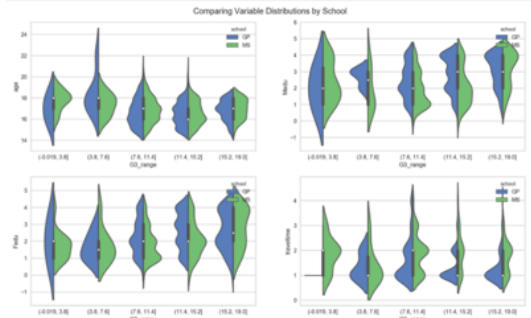
```
In [6]: # Splitting G3 into ranges to get a cleaner visual
student_perf['G3_range'] = pd.cut(student_perf.G3, 5, rebins = True)[0]

# Creating the plots
plt.figure(figsize=(16, 36))
sns.set(style="whitegrid", palette="muted", color_codes=True)

plt.subplots_adjust(top=0.97)
plt.suptitle('Comparing Variable Distributions by School')

sns.despine()
for column_index, column in enumerate(['age', 'Medu', 'Fedu', 'traveltime', 'studytme',
    'freetime', 'failures',
    'absences', 'famel', 'goout', 'Dalc', 'Walc', 'he
    alth', 'G1', 'G2']):
    if column == 'G3_range':
        continue
    plt.subplot(6, 2, column_index + 1)
    sns.violinplot(x='G3_range', y=column, hue = 'school', split = True, data=student_per
    f)

plt.savefig('fig/distrbyschool.png');
```



Code and tests

Branch: master - project-3-p2-ka-jo-ta / p3functions.py

s-johnson Add two_way function

1 contributor

41 lines (34 sloc) | 1.38 KB

```
1 import pandas as pd
2 import numpy as np
3
4 def make_indicators(df, names):
5     """Make indicator columns in dataframe df of whether existing columns are
6     equal to given values.
7
8     Args:
9         df (pandas.DataFrame): Dataframe to be modified.
10        names (dict) : Dictionary containing:
11            - Keys: Desired indicator column names
12            - Values: Two item tuple containing:
13                - Original dataframe column
14                - Value to compare to column
15
16    Returns:
17        void: Dataframe df is modified in place.
18
19    """
20    for k,v in names.items():
21        df[k] = 1*(df[v[0]] == v[1])
22    ...
```

Branch: master - project-3-p2-ka-jo-ta / tests.py

s-johnson Add two_way function

1 contributor

31 lines (24 sloc) | 906 Bytes

```
1 import pandas as pd
2 import numpy as np
3 import numpy.testing as npt
4
5 from p3functions import *
6
7
8 def test_make_indicators():
9     d = {'col1': [1, 2], 'col2': [3, 4]}
10    df = pd.DataFrame(data=d)
11    names = {'ind1': ('col1', 2), 'ind2': ('col2', 3)}
12    make_indicators(df, names)
13    exp_d = {'col1': [1, 2], 'col2': [3, 4], 'ind1': [0, 1], 'ind2': [1, 0]}
14    exp = pd.DataFrame(data=exp_d)
15    obs = df
16    assert obs.equals(exp)
17    ...
```

Project1-Main-Narrative

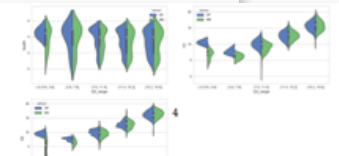
January 4, 2018

1 The effects of studying on high school students

Authors: Nadav Tadelis, Sarah Johnson, Chitwan Kaudan

1.1 Abstract

As students, a large part of our daily life is taken up by school. We often make unformed guesses about how much an extra hour of study is worth. Specifically, if imperfect information causes us to skip school, then we make poor decisions about how much to study (effectively resulting in a loss of utility). If we had more information about our own learning, then we could calibrate our investment in study time more accurately. Using a naive OLS, then addressing endogeneity by using an instrumental variable, we find that a marginal increase in study time per week can lead to a 0.15 point increase in grades.



1.2 Exploratory Data Analysis

The data being used are from the public archive collected by Paulo Cortez of the University of Minho. Below is a list of all included variables:

in our regression.

Now that we have established our data are clean we can move on to trying to answer our question regarding the marginal effect of studying on grades.

1 We need to make the additional assumption that in secondary school (where parents are notified when students are absent), absences are only caused by illnesses and emergencies (which are independent of study time). Without this assumption it would be plausible that students are skipping school because they value leisure over studying, implying a negative correlation between study time and absences.

1.3 Initial Naive OLS fit

The first step is to build a model and make some assumptions to define the relationship between grades and studying. Let an individual's grade be G_i and weekly hours of studying be S_i and their "ability" be A_i . Then we can write:

$$G_i = \beta_0 + \beta_1 S_i + \beta_2 A_i + U_i$$

1.5.1 References

Card, D., & Krueger, A. (1992). Does School Quality Matter? Returns to Education and the Characteristics of Public Schools in the United States. *Journal of Political Economy*, 100(1), 1-40.

Cortez, P. and Silva, A. Using Data Mining to Predict Secondary School Student Performance. In A. Brito and J. Teixeira Eds., *Proceedings of 5th Future Business Technology Conference (FUTURTEC 2008)* pp. 5-12. Porto, Portugal, April, 2008. EUROSS, ISBN 978-9077381-39-7.

Greene, W. H. (2000). *Econometric analysis*. Upper Saddle River, NJ: Prentice Hall.

MacKinnon, J.G. and H. White. (1985). Some heteroskedasticity consistent covariance matrix estimators with improved finite sample properties. *Journal of Econometrics*, 29, 53-57.

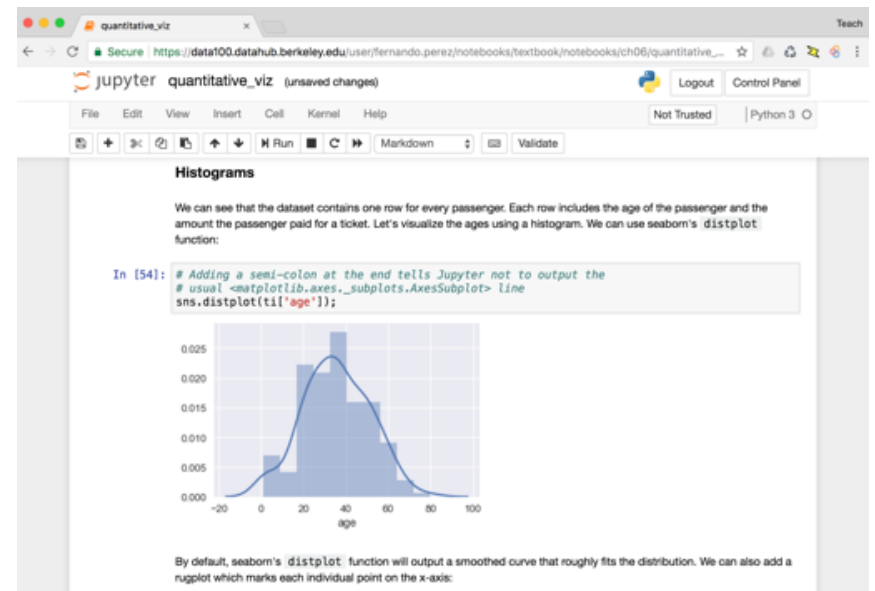
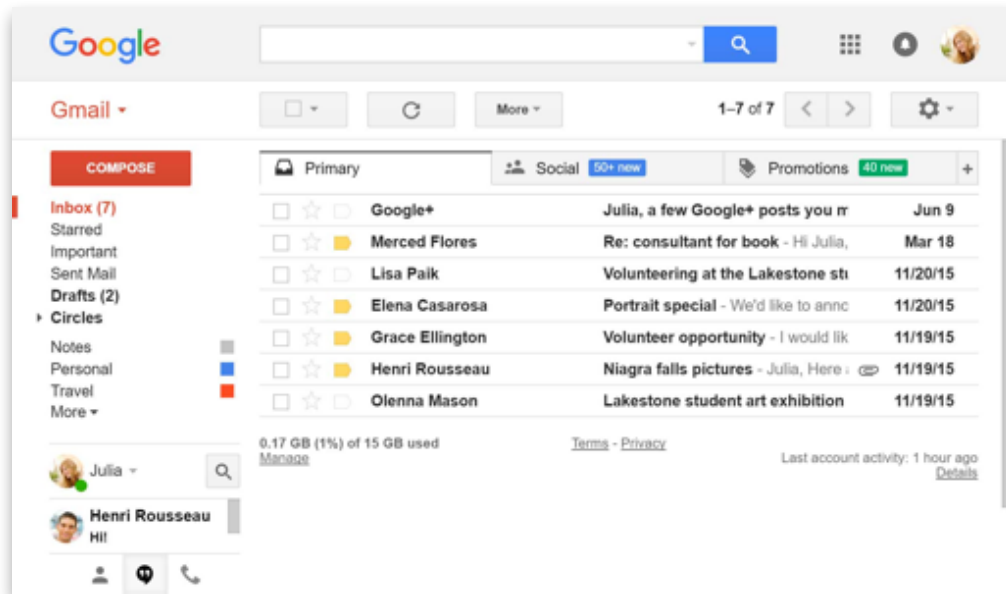
1.5.2 Author Contributions

- Nadav Tadelis: Had idea from a project he did in Econ 142, worked to pick right instruments to improve the 2SLS model, wrote analysis in main.ipynb, created visualizations, and wrote/coded model fitting notebooks.
- Sarah Johnson: Helped brainstorm instruments to improve 2SLS, wrote analysis in main.ipynb, created functions and tests, and integrated testing through Travis.
- Chitwan Kaudan: Helped brainstorm instruments to improve 2SLS, wrote analysis in main.ipynb, worked on reproducibility aspects, created environment and makefile, and structured notebooks.

With these tools, we provide:

- ❖ **Broad disciplinary reach** and impact of statistical thinking.
- ❖ Drastically **lowered barriers** to student access - intellectual and economic.
- ❖ Lowered barriers for **faculty*** to engage with statistical and computational ideas.
- ❖ (*) typically from non computational/statistical domains)

Berkeley in a few years...



In the words of a Berkeley scientist

We are witnessing a monumental phase shift in data science knowledge on campus - undergrads are extremely well trained.

[...]

There is a **knowledge gap** between the trained data science **undergrads** on campus and the **upper level scientists and PIs** who need their expertise for their projects. The labs are ill equipped on reproducible data science methods and therefore are **incapable of knowing how to manage the full potential of undergrad help.**



Ciera Martinez, @CieraReports

Postdoctoral Fellow, Molecular and Cell Biology & BIDS Fellow

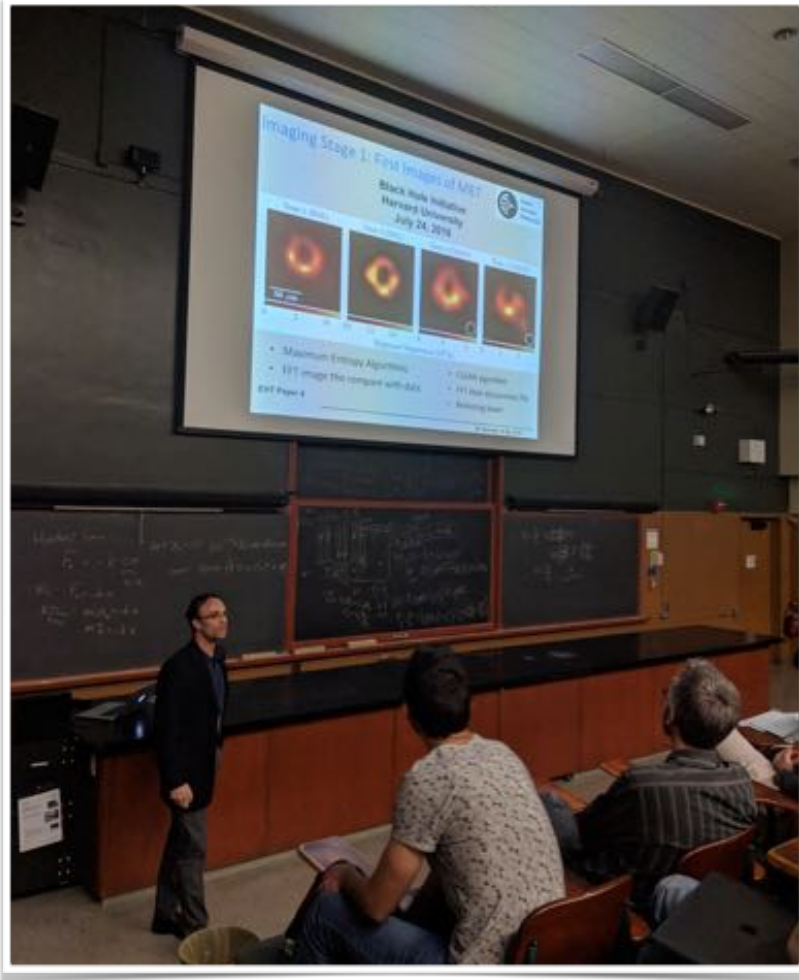
Transforming research



Event Horizon Telescope









April 18, 2019: Shep Doeleman



THE ASTROPHYSICAL JOURNAL LETTERS

OPEN ACCESS

First M87 Event Horizon Telescope Results. III. Data Processing and Calibration

The Event Horizon Telescope Collaboration, Kazunori Akiyama^{1,2,3,4} , Antxon Alberdi⁵ , Walter Alef⁶, Keiichi Asada⁷, Rebecca Azulay^{8,9,6} , Anne-Kathrin Baczko⁶ , David Ball¹⁰, Mislav Baloković^{4,11} , John Barrett²  [+ Show full author list](#)

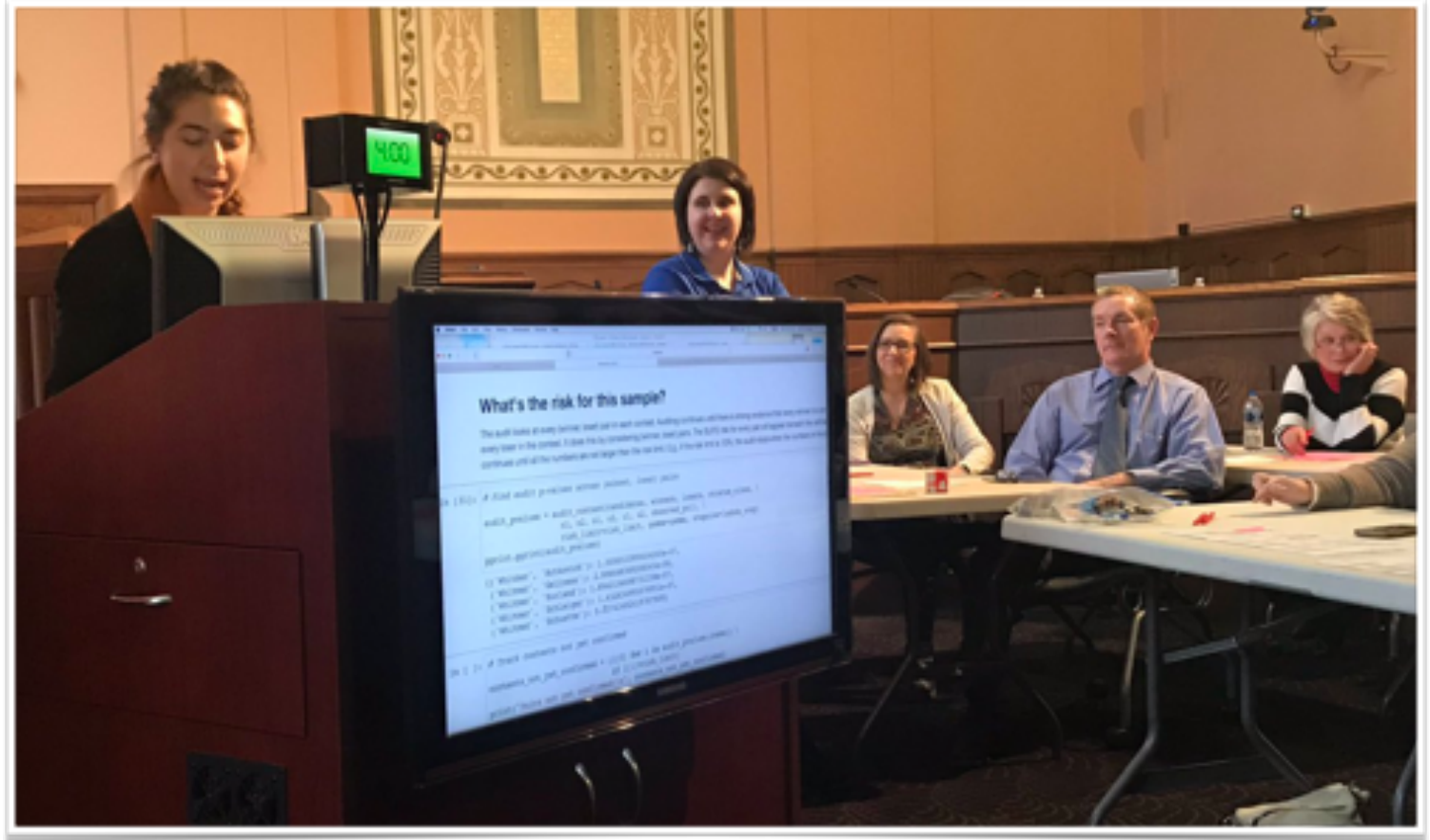
Published 2019 April 10 • © 2019. The American Astronomical Society.

[The Astrophysical Journal Letters, Volume 875, Number 1](#)

Software: DiFX (Deller et al. [2011](#)), CALC, PolConvert (Marti-Vidal et al. [2016](#)), HOPS (Whitney et al. [2004](#)), CASA (McMullin et al. [2007](#)), AIPS (Greisen [2003](#)), ParselTongue (Kettenis et al. [2006](#)), GNU Parallel (Tange [2011](#)), GILDAS, eht-imaging (Chael et al. [2016](#), [2018](#)), Numpy (van der Walt et al. [2011](#)), Scipy (Jones et al. [2001](#)), Pandas (McKinney [2010](#)), Astropy (The Astropy Collaboration et al. [2013](#), [2018](#)), Jupyter (Kluyver et al. [2016](#)), Matplotlib (Hunter [2007](#)).

K. Ottoboni/P. Stark: Defenders of Democracy

Michigan election audit, Dec. 2018



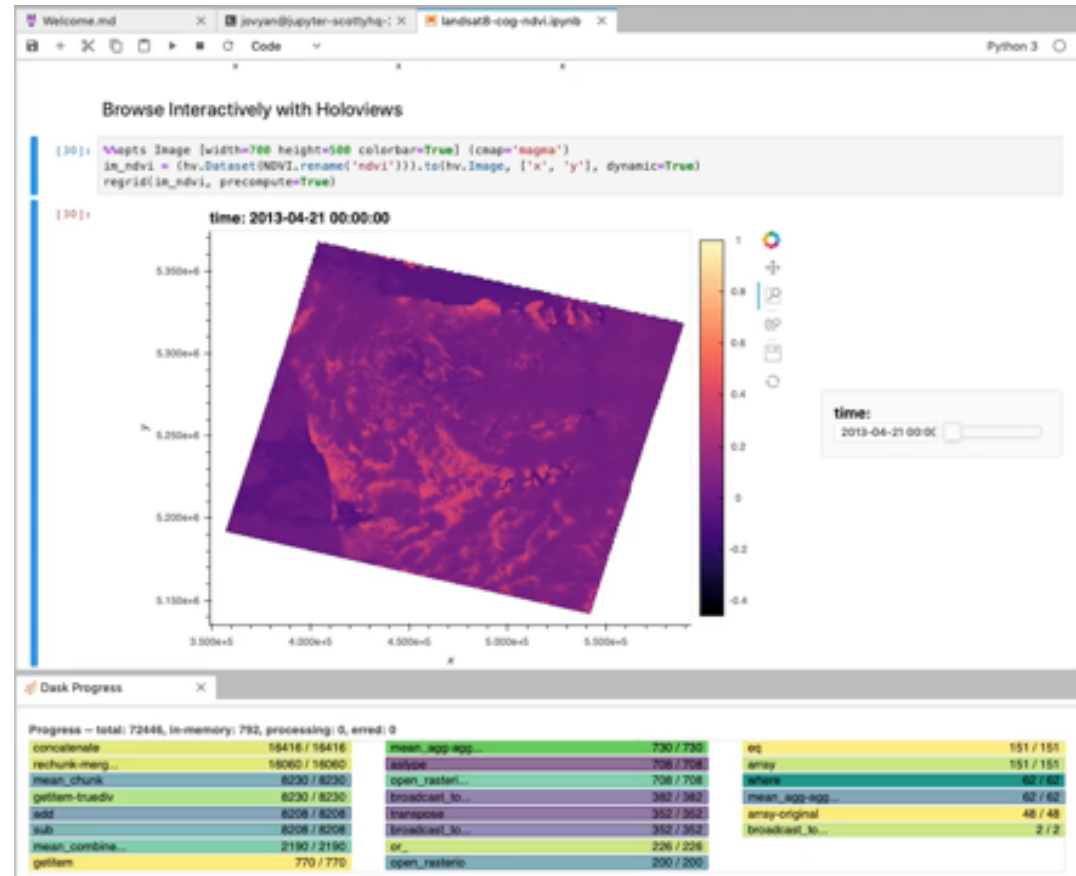
Credit: @ginvdr

Pangeo: open geosciences (and more!)



Harnessing the power of cloud computing to study the whole earth interactively.

<https://pangeo.io>



Scott Henderson [Follow](#)

Research geophysicist at University of Washington eScience Institute
Oct 1 · 7 min read

The Economist



Evan Hensleigh [Follow](#)

Visual journalist at The Economist; accidental Londoner.

Oct 9 · 3 min read

Peeling back the curtain

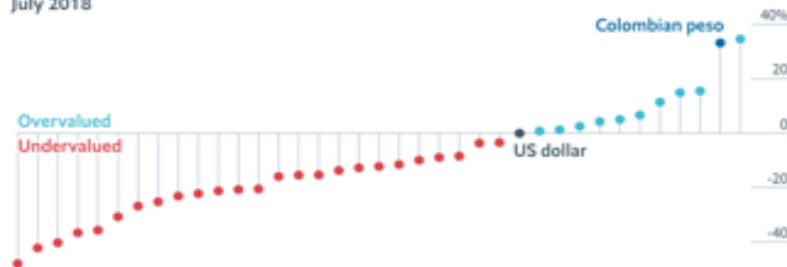
How the Economist is opening the data behind our reporting

We started calculating the Big Mac index in 1986, and until this year it has been compiled and calculated manually. There are still a lot of hands-on parts to the process—in particular, compiling the list of prices—but we have now converted much of the calculation into code. **We published these calculations in a Jupyter notebook**, a tidy format for breaking scripts into small blocks and annotating them.

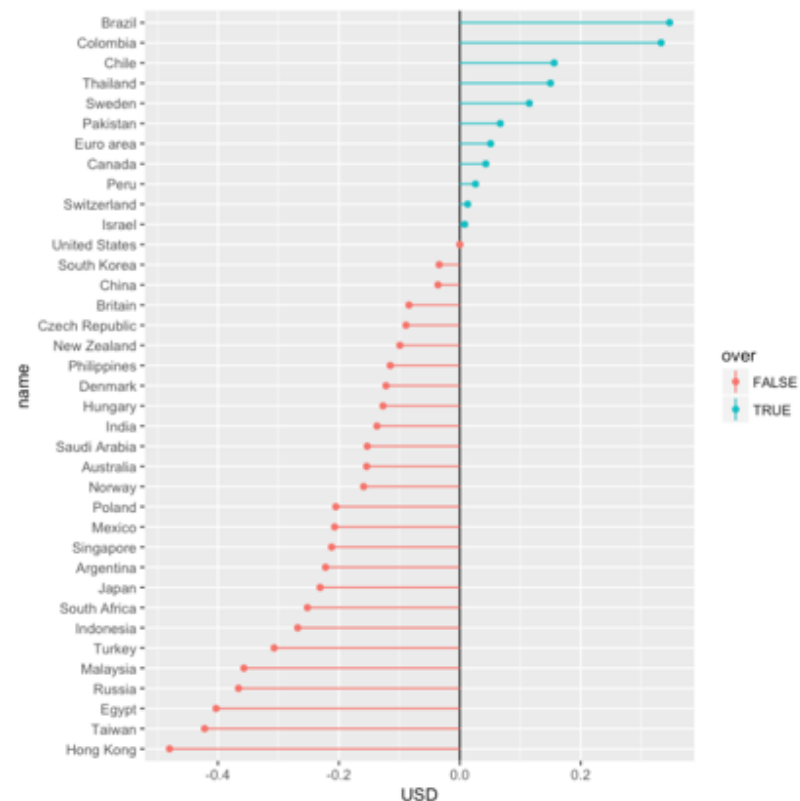


The Colombian peso is 33% overvalued against the US dollar

July 2018



```
In [16]: to_plot = big_mac_adj_index[date == latest_date]
to_plot$name = factor(to_plot$name, levels=to_plot$name[order(to_plot$USD)])
ggplot(to_plot[, over := USD > 0], aes(x=name, y=USD, color=over)) +
  geom_hline(yintercept = 0) +
  geom_linerange(aes(ymin=0, ymax=USD)) +
  geom_point() +
  coord_flip()
```





National infrastructure, from K-12 to HPC

cybera

CYBERA > NETWORK > SERVICES > PROJECTS > NEWS & EVENTS > MEMBERSHIP > CONTACT US >

Jupyter 'All-in-One' Science Platform

Learning and sharing in a flexible, collaborative and interactive way.

jupyter

Jupyter is an integrative application that incorporates math, science and engineering tools, along with communication and visualization resources, in one web-based platform. Simply put: It enables a broad suite of computing capabilities on any device that has an internet connection. For free.

Cybera and the Pacific Institute for the Mathematical Sciences (PIMS) have teamed up to increase access to, and awareness of, Jupyter. Cybera is hosting the platform on its Rapid Access Cloud, and is offering free access (and advice on how to get started) to Canada's public and innovation sectors.

Who is Jupyter Useful For?


- Teachers**
 - Create "interactive textbooks" that allow students to actively work on math or data science problems.
- Students**
 - Learn and practice multiple programming languages, and log experiments, all in one place.

Use Cases

- K-12**
 - An Alaska high school teacher used Jupyter to change the way "Introduction to Programming" is taught
- Post-Secondary**
 - A professor at George Washington University used Jupyter to teach *Aerodynamics-Hydrodynamics* (see [github notes](#))

compute canada | calcul canada

15/03/2017 Featured News



compute canada and pims launch jupyter service for researchers

Home

About

Renewing Canada's Academic Research Computing Infrastructure

Research Data Management

News

SYZYG.Y.CA

ABOUT PARTNERS INTRO **LAUNCH** EN FR

SYZYG.Y.CA

Launch Jupyter at your university, school or company?

Your name

Your email

Your message

SEND



Pacific Institute *for the* Mathematical Sciences



Microsoft Azure Notebooks [PREVIEW]
Overview Libraries FAQ/Support What's New Sign In

Jupyter

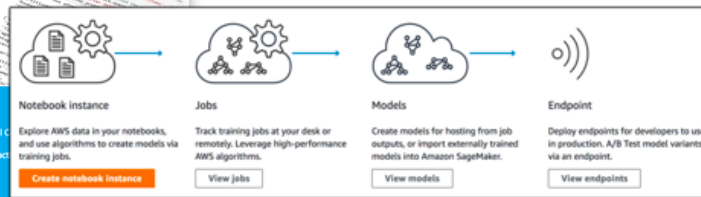
Notebooks hosted on Microsoft Azure

Go to my Notebook Server
Show me some samples

WHAT IS JUPYTER?
Interactive Notebooks for Data Science and Technical Computing
Browser-based REPL with Markdown and Inline Interactions
Support for Python 2, Python 3 and R

ABOUT THIS SERVICE

Amazon SageMaker



Google Cloud Platform

CLOUD DATALAB BETA

An easy to use interactive tool for large-scale data exploration, analysis, and visualization.

[TRY IT FREE](#)

Powerful Data Exploration

Cloud Datalab is a powerful interactive tool created to explore, analyze and visualize data with a single click on Google Cloud Platform. It runs on Google App Engine and orchestrates multiple services automatically so you can focus on exploring your data.



A Growing Set of Data Science Tools

- Jupyter Notebooks**
Create and collaborate on Python, R, and Scala notebooks that contain rich interactive outputs.
- RStudio**
Jumpstart your R experience with a free, open-source RStudio tool.
- Machine Learning (Coming Soon)**
Create, train and deploy machine learning models.



Overview of Collaboratory Features

Table of contents: Code snippets, Cells, Text cells, Adding and moving cells, Working with python, System aliases, Magics, Tab-completion and exploring code, Rich, interactive outputs, Integration with Drive, Commenting on a cell, SECTION

Rich, interactive outputs

Until now all of the generated outputs have been text, but they can be more interesting, like the chart below.

```
import numpy as np
from matplotlib import pyplot as plt
ye = 200 + np.random.randn(100)
x = [x for x in range(len(ye))]
plt.plot(x, ye, '-')
plt.fill_between(x, ye, 195, where=(ye > 195), facecolor='g', alpha=0.4)
plt.title("Fill and Alpha Example")
plt.show()
```

Integration with Drive

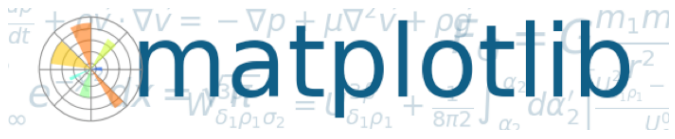
Collaboratory is integrated with Google Drive. It allows you to share, comment, and collaborate on the same document with multiple people:

- The **SHARE** button (top-right of the toolbar) allows you to share the notebook and control permissions set on it.
- File-Make a Copy** creates a copy of the notebook in Drive.
- File-Save** saves the File to Drive. **File-Save and checkpoint** pins the version so it doesn't get deleted from the revision history.
- File-Revision History** shows the notebook's revision history.
- Multiple people can **collaboratively edit** the same notebook at the same time. Like Google Docs, you can see collaborators both within the document (top right, left of the comments button) and within a cell (right of the cell).



So you want to build Data Science tools
in academia...

Career paths?



BrighamYoung.University



ANACONDA.



Should I Resign From My Full Professor Job To Work Fulltime On Cocalc?

William Stein • Apr 12, 2019 •

Nearly 3 years ago, I gave [a talk](#) at a Harvard mathematics conference announcing that “I am leaving academia to build a company”. What I really did is go on *unpaid leave* for three years from my tenured Full Professor position. No further extensions of that leave is possible, so I finally have to decide whether or not to go back to academia or resign.



CoCalc Blog

My unpaid leave is up – what am I going to do?

My third year of unpaid leave from UW is up. I have to decide whether to return to UW or resign. If I return, it turns out that I would have to have [at least a 50% appointment](#). I currently have 50% of one year of teaching in “credits”, which means I wouldn't be required to teach for the first year I go back as a 50% appointment. Moreover, the current department chair (John Palmieri) understands and appreciates Sage – he is among the [top 10 all time contributors to the source code of Sage!](#)

[I have decided to resign](#). I'm worried about issues of intellectual property; it would be extremely unfair to my employees, investors and customers if I took a 50% UW

An awkward space for academics

- ❖ These problems sit at the **interface of research and engineering**.
- ❖ Some go to the heart of statistical thinking.
- ❖ Yet they may require heavy-duty software engineering.
- ❖ This type of work is extremely **hard to fund and reward**.
- ❖ **Industry** alone won't do what we need.
- ❖ **Sustainability** of these efforts is today extremely challenging
 - ❖ (even funded ones like Jupyter)

Jupyter - funding and resources



**ALFRED P. SLOAN
FOUNDATION**

GORDON AND BETTY
MOORE
FOUNDATION

THE LEONA M. AND HARRY B.
HELMSLEY
CHARITABLE TRUST



U.S. DEPARTMENT OF
ENERGY



SIMONS FOUNDATION

NETFLIX

POWERED BY
rackspace
the open cloud company



ANACONDA

IBM



Microsoft

ENTHOUGHT
SCIENTIFIC COMPUTING SOLUTIONS

Google

Bloomberg