A Bioinformaticians' Guide to Python

Bioinformatics Practicum

HELLO, WORLD!

I am Trenton Beckendorff

I am a Second-Year CS Major, and Peer Mentor for the *Big Data in Biology* FRI Stream.

You can contact me at trentonbeckendorff@utexas.edu

Preparing for Launch

http://bit.ly/2kgkYfm

Installing Anaconda

A Python distribution for Data Science.

1

DOWNLOAD ANACONDA NOW

Download for

GET SUPERPOWERS WITH ANACONDA

Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages for data science.

Additionally, you'll have access to over 720 packages that can easily be installed with conda, our renowned package, dependency and environment manager, that is included in Anaconda. See the packages included with Anaconda and the Anaconda changelog

Which version should I download and install?
With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.

If you don't have time or disk space for the entire distribution, try Miniconda which contains only conda and Python. Then install just the individual packages you want through the conda command.

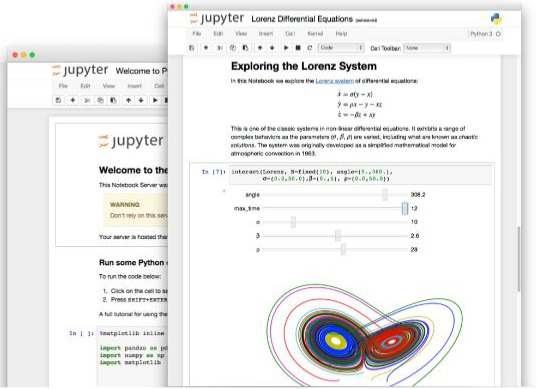
Running Jupyter

An interactive notebook for Python.

2

Ready to get started?

Try it in your browser | Install the Notebook



The Jupyter Notebook

The Jupyter Notebook is a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, machine learning and much more.

Language of choice

The Notebook has support for over 40 programming languages, including those popular in Data Science such as Python, R, Julia and Scala.

Share notebooks

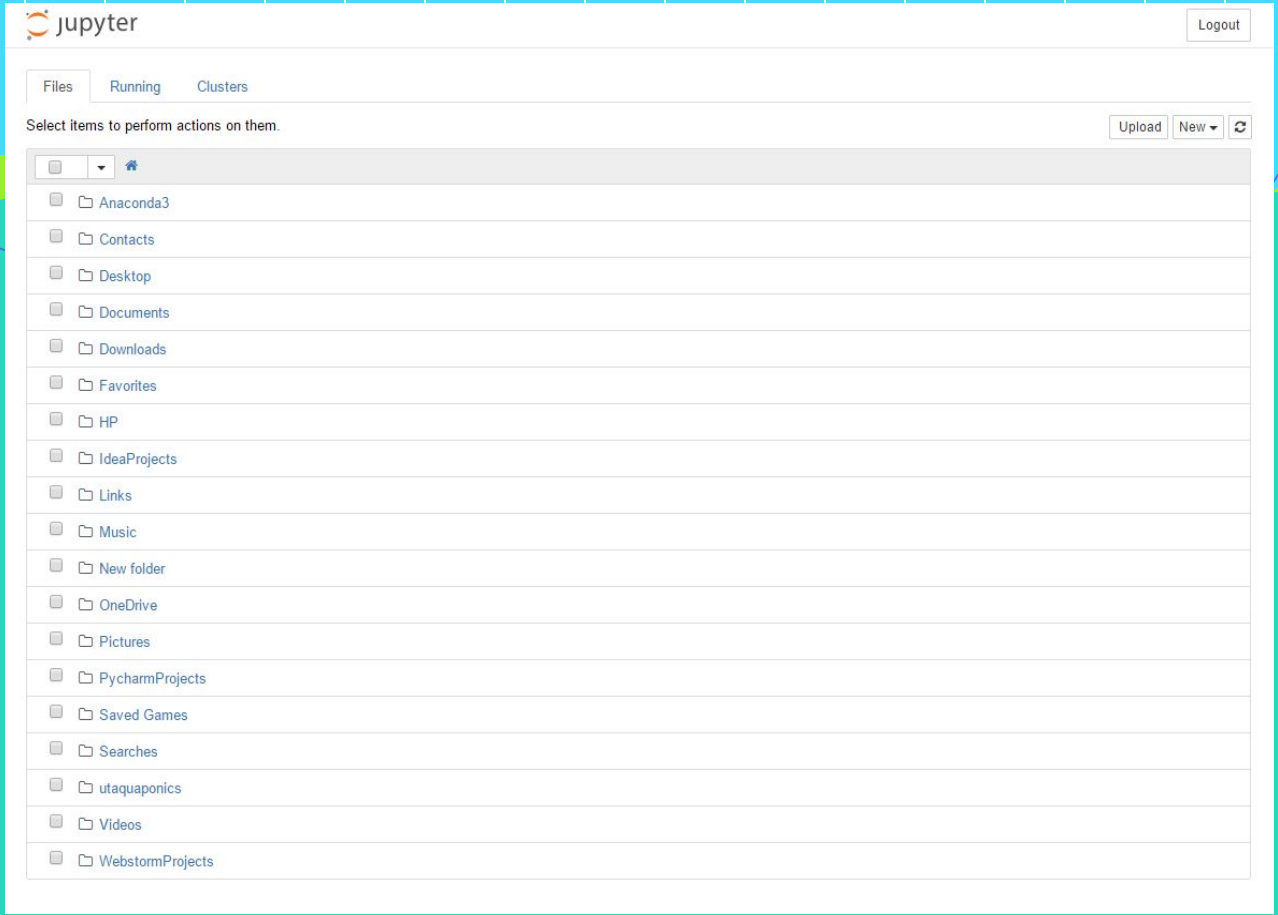
Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer.

Interactive widgets

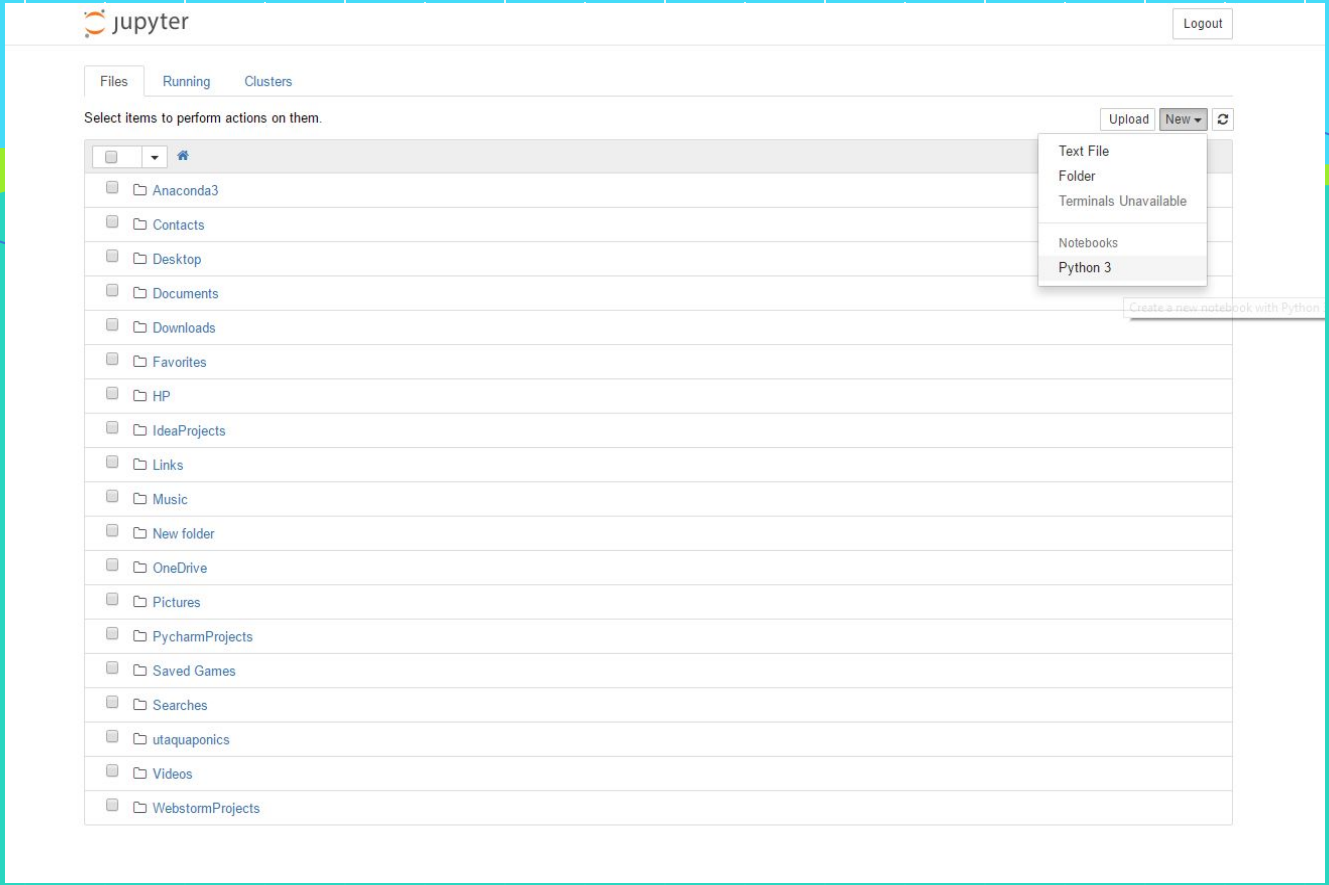
Code can produce rich output such as images, videos, LaTeX, and JavaScript. Interactive widgets can be used to manipulate and visualize data in realtime.

Big data integration

Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, dplyr, etc.




```
trenton@linux:~$ jupyter notebook
```



Creating a new Jupyter Notebook for Python 3.

jupyter Untitled Last Checkpoint: a few seconds ago (unsaved changes)

Logout

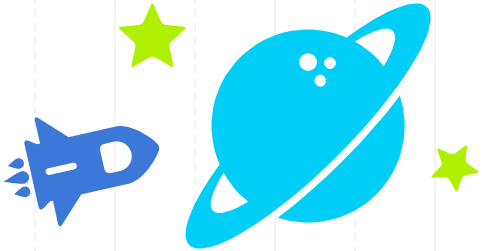
File Edit View Insert Cell Kernel Widgets Help

Python 3

Code

CellToolbar

```
In [ ]:
```



What is Python?

A fast, powerful, open, object-oriented, readable, all-in-one programming language.

Understanding the Syntax

3

Indentation, Comments, Operators, Variables, Types, Control Flow, Functions, File I/O, Imports

```python
#!/usr/bin/env python

def main():
    print("hello, world!")

if __name__ == "__main__":
    main()
```

Example 1: Hello World

Indentation

3.1

```python
#!/usr/bin/env python

def main():
    print("hello")
    print("hello again")

if __name__ == "__main__":
    main()
```

Example 2: Indention

Comments

3.2

```python
#!/usr/bin/env python

def main:
    print("hello")
    print("hello again") # prints hello again

if __name__ == "__main__":
    main()
```

Example 3: In-Line Comments

```python
#!/usr/bin/env python


def main:
    print("hello")
    print("hello again")


'''
Run main() if executed from command line.
'''
if __name__ == "__main__":
    main()
```

Example 4: Multi-Line Comments

Operators 3.3

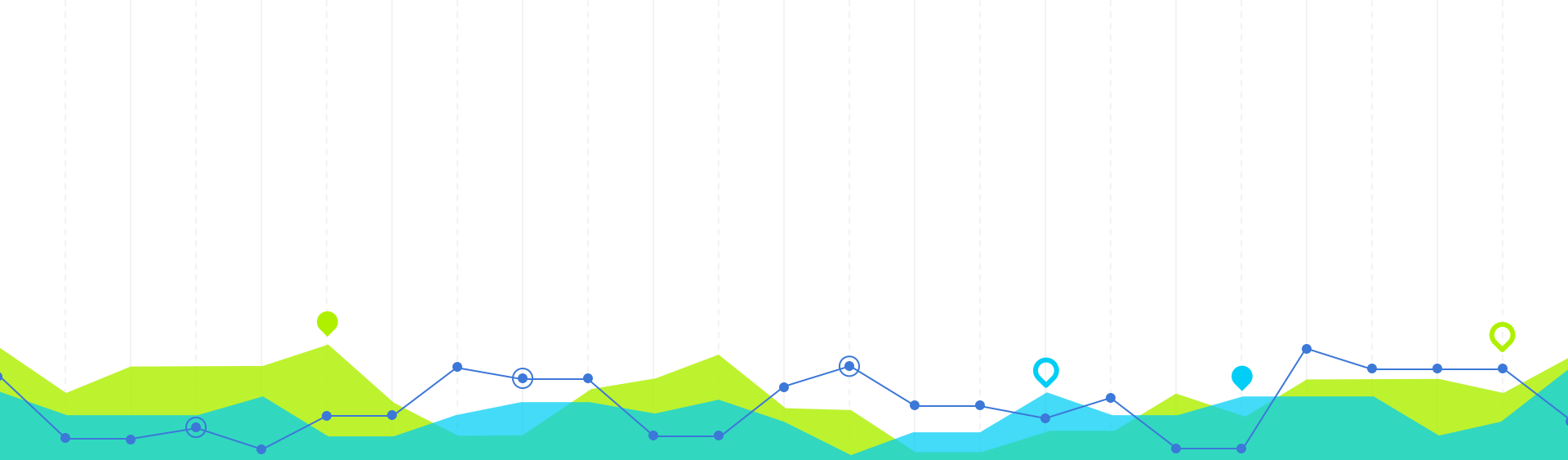
```python
#!/usr/bin/env python


def main:
    print(5 + 5)        # Addition
    print(27 / 3)       # Floating-Point Division
    print(5 * 5)        # Multiplication
    print(5**5)         # Exponentiation
    print(100 // 3)     # Integer Division
    print(5 % 4)        # Remainder (modulo)
    print(13.5 - 14.5) # Subtraction
    print(True + 1)     # ????


if __name__ == "__main__":
    main()
```

Example 5: Operators

Variables 3.4

```python
#!/usr/bin/env python

def main:
    x = 5
    y = 6
    z = 7

    print(x - y)
    print(y - x)
    print(x + y + z)
    print(x == y)

if __name__ == "__main__":
    main()
```

Example 6: Variables



Types 3.5

```python
#!/usr/bin/env python

def main:
    dna_sequence = 'ACTGACTGACTGTATATATATATATA'
    dna_complement = 'TGACTGACTGACATATATATATATAT'
    rna_sequence = 'ACUGACUGACUGUAUAUAUAUAUAUA'
    amino_acid = 'MetLeuSerTyrTyrGluSerIleMetLeuSerTyrTyrGluSerIle'

    x = 'A'
    x = x + 'C'
    print(x)
    print(x[0])
    print(x[1])

if __name__ == "__main__":
    main()
```

Example 7: Strings

```python
#!/usr/bin/env python

def main:
    cancer_genes = ['ABI1', 'ABL1', 'ABL2', 'ACKR3', 'ACSL3', 'ACSL6', 'AVCR1', 'AFF1']

    first_gene = cancer_genes[0]
    print("The first gene is: " + first_gene)

    last_gene = cancer_genes[len(cancer_genes) - 1]
    print("The last gene is: " + last_gene)

    cancer_genes.insert(2, 'BRCA1')
    print("Cancer genes: " + str(cancer_genes))

    cancer_genes.append('BRCA1')
    print("Cancer genes: " + str(cancer_genes))

    cancer_genes.remove('ABI1')
    print("Cancer genes: " + str(cancer_genes))

    print("The number of BRCA1 genes is: " + str(cancer_genes.count('BRCA1')))
    print("The index of BRCA1 is:" + str(cancer_genes.index('BRCA1')))

    cancer_genes.reverse()
    print("Cancer genes: " + str(cancer_genes))

if __name__ == "__main__":
    main()
```

Example 8: Lists

```python
#!/usr/bin/env python

def main:
    start_codon_loc = 0
    stop_codon_loc = 5
    exon_loc = (start_codon_loc, stop_codon_loc + 1)
    print(exon_loc)
    print(exon_loc[0])
    print(exon_loc[1])
    print(exon_loc * 2)

if __name__ == "__main__":
    main()
```

Example 8: Tuples

```python
#!/usr/bin/env python

def main:
    dna_to_rna = {'A':'A', 'C':'C', 'G':'G'}
    dna_to_rna['T'] = 'U'

    print(dna_to_rna['A'])
    print(list(dna_to_rna.keys()))
    print('Y' in dna_to_rna)
    print(list(dna_to_rna.values()))

if __name__ == "__main__":
    main()
```

Example 9: Dictionaries

Control Flow

3.6

```python
#!/usr/bin/env python

def main():
    if(True):
        print("hello")
        print("hello again")
    else:
        print("goodbye")

if __name__ == "__main__":
    main()
```

Example 10: If / Else

```python
#!/usr/bin/env python

def main:
    dna_sequence = 'ACTGACGTCTATATA'
    for base in dna_sequence:
        print(base)
    print("\n")
    for i in range(len(dna_sequence)):
        print(dna_sequence[i])

if __name__ == "__main__":
    main()
```

Example 11: For

```python
#!/usr/bin/env python

def main:
    dna_sequence = 'AAAAAACTGA'
    i = 0
    while(dna_sequence[i] == 'A' and i < len(dna_sequence)):
        print(dna_sequence[i])
        i = i + 1

if __name__ == "__main__":
    main()
```

Example 12: While

Functions

3.7

```python
#!/usr/bin/env python

def transcribe(dna):
    rna = ''
    for base in dna:
        if (base == 'T'):
            rna = rna + 'U'
        else: rna = rna + base
    return rna

def main():
    dna_sequence = 'ACTGATCAGATGCA'
    rna = transcribe(dna_sequence)
    print(rna)

if __name__ == "__main__":
    main()
```

Example 13: Functions

File I/O

3.8

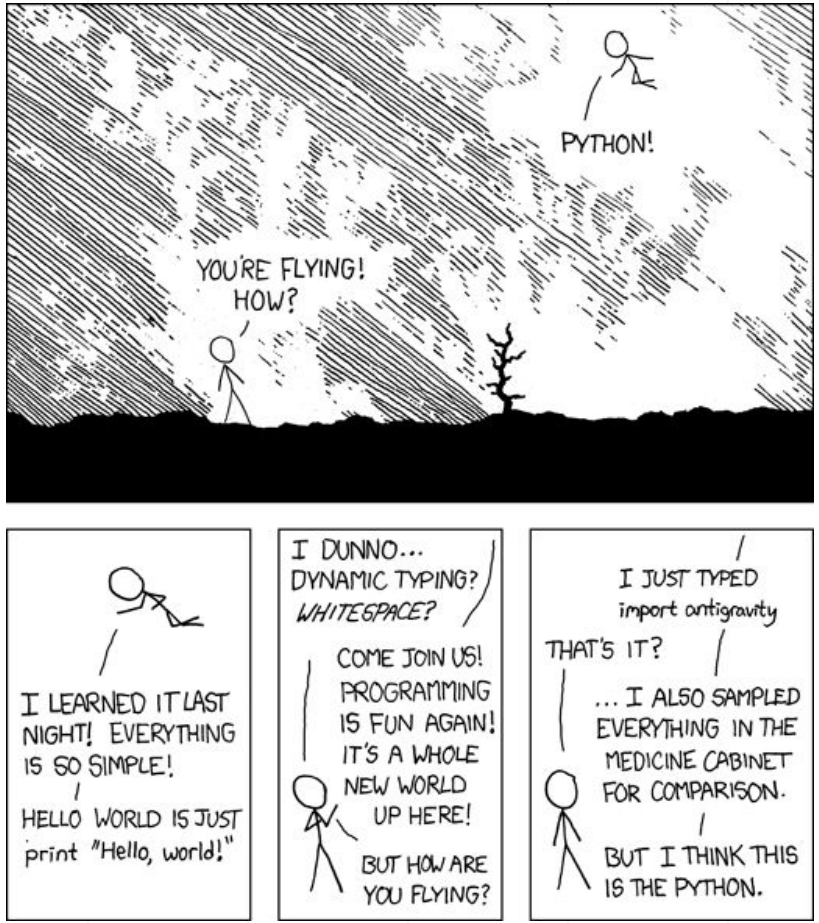
```python
#!/usr/bin/env python

def transcribe(dna):
    rna = ''
    for base in dna:
        if (base == 'T'):
            rna = rna + 'U'
        else: rna = rna + base
    return rna

def main():
    f = open('dna_sample.txt', 'r') # Read the dna sample file.
    dna_sample = f.read()
    print(dna_sample)
    print("\n")
    rna_sample = transcribe(dna_sample)
    print(rna_sample)

    w = open('rna_sample.txt', 'w')
    w.write(rna_sample)

if __name__ == '__main__':
    main()
```

Example 14: File I/O

Simple, yet Powerful

Next Time: Applying Python to Bioinformatics

4


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

THANKS!

Any questions?

You can find me at

trentonbeckendorff@utexas.edu

Citations

1. Startup icon made by Maxim Basinski from www.flaticon.com
2. Presentation slide template by SlidesCarnival
3. Python comic by XKCD

PEP8 Style Guide: *Key Concepts*

Indentation:
- use 4 spaces per indentation level
- ‘hanging indentations’ should be distinguishable

Tabs or Spaces?
- spaces > tabs
- Python 3: no mixing of tabs and spaces

Line Length:
- maximum of 79 characters
- limit docstrings / comments to 72 characters

Blank Lines:
- use in functions (sparingly) to separate logical sections
- 2 blank lines b/t functions

Source File Encoding:
- UTF-8

Imports:
- use separate line for each import
- always at the top of the file
  - standard library imports
  - third party imports
  - local imports

Quotes:
- double or single

Whitespace:
- avoid extraneous whitespace
- always surround operators

Comments:
- should be complete sentences
- should be updated when modifying code
- inline comments: use sparingly