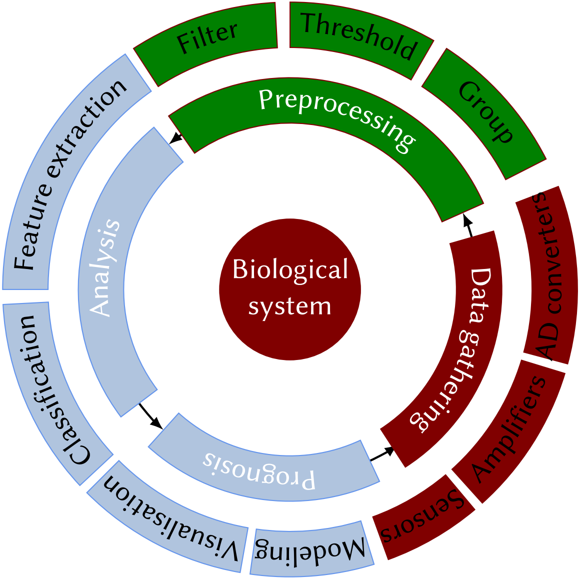
Biological data analysis (3)
Python

Aleksandras Voicikas
avoicikas@gmail.com

Programming

- Why?
- What language to choose?
- Tools?
- How?

Data analysis



Programming languages

- C
- Java
- JavaScript
- SQL
- Matlab
- R
- Python
- Julia

https://www.scriptol.com/programming/hello-world.php

Python installation

- Windows:
  - Chocolatey; choco install python
  - python -m pip install -U pip
- Linux:
  - sudo apt-get update; sudo apt-get install python3.8
- MAC:
  - Homebrew installation
  - brew install python
- Anaconda, canopy package managers (https://docs.anaconda.com/anaconda/install/)

Python package and environment management

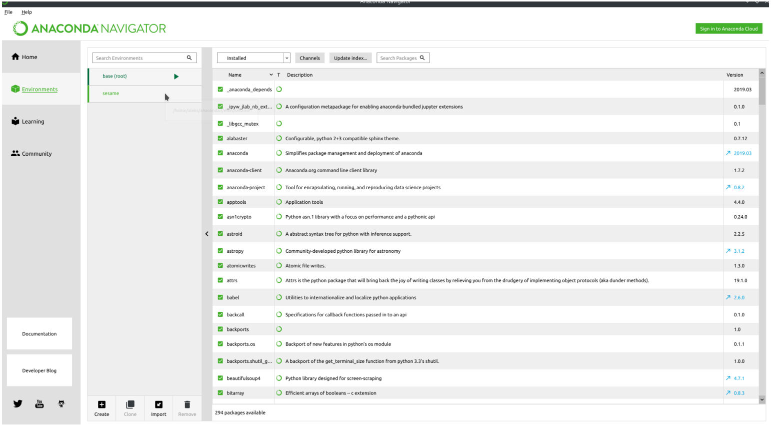
Pip:

- Python Package Index
- pip install virtualenv
- cd project; virtualenv venv; source venv/bin/activate;
- pip install -r requirements.txt
- pip freeze > requirements.txt; deactivate
- pip install virtualenvwrapper

Conda:

- conda create –name myenv
- conda activate myenv; conda list
- conda env export > environment.yml
- conda deactivate
- conda remove –name myenv –all
- conda create -f environment.yml

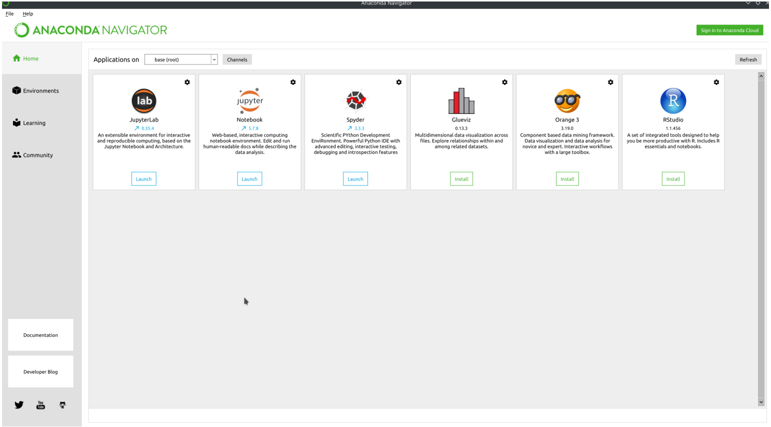
Anaconda-navigator: packages and environments



	Name	T	Description	Version
☑	_anaconda_depends	○		2019.03
☑	_ipyw_jlab_nb_ext…	○	A configuration metapackage for enabling anaconda-bundled jupyter extensions	0.1.0
☑	_libgcc_mutex	○		0.1
☑	alabaster	○	Configurable, python 2+3 compatible sphinx theme.	0.7.12
☑	anaconda	○	Simplifies package management and deployment of anaconda	↗ 2019.03
☑	anaconda-client	○	Anaconda.org command line client library	1.7.2
☑	anaconda-project	○	Tool for encapsulating, running, and reproducing data science projects	↗ 0.8.2
☑	apptools	○	Application tools	4.4.0
☑	asn1crypto	○	Python asn.1 library with a focus on performance and a pythonic api	0.24.0
☑	astroid	○	A abstract syntax tree for python with inference support.	2.2.5
☑	astropy	○	Community-developed python library for astronomy	↗ 3.1.2
☑	atomicwrites	○	Atomic file writes.	1.3.0
☑	attrs	○	Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	19.1.0
☑	babel	○	Utilities to internationalize and localize python applications	↗ 2.6.0
☑	backcall	○	Specifications for callback functions passed in to an api	0.1.0
☑	backports	○		1.0
☑	backports.os	○	Backport of new features in python's os module	0.1.1
☑	backports.shutil_g…	○	A backport of the get_terminal_size function from python 3.3's shutil.	1.0.0
☑	beautifulsoup4	○	Python library designed for screen-scraping	↗ 4.7.1
☑	bitarray	○	Efficient arrays of booleans -- c extension	↗ 0.8.3

294 packages available

Anaconda-navigator: editors



Applications on base (root)

JupyterLab
↗ 0.35.4
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook
↗ 5.7.8
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Spyder
↗ 3.3.3
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Glueviz
0.13.3
Multidimensional data visualization across files. Explore relationships within and among related datasets.

Orange 3
3.19.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Python versions

- python2
- python3
- IPython
- ...

- python script.py # Execute code written in script.py
- python -i script.py # interactive mode
- Launch text editor, terminal and console from JupyterLab

https://docs.python.org/3/tutorial/index.html

IPython

- jupyter.org
- ipython -> jupyter notebook -> jupyter lab
- Additional functions to basic python:
  - help(len) or len? to get help on len function
  - Code accessibility with ??
  - TAB for autocompletion
  - Previous commands (Ctrl-p/n/r)
  - Session control (Ctrl-d/c/l)
  - magic commands: %cpaste, %run, %lsmagic, %pip, %conda, %cd and other bash commands
  - output history: %history -n 1-4, print(_), Out[1], In[1]
  - debugging: %debug, %xmode Plain; %pdb on
  - etc...

IPython: magic functions

```
%cd ~/Documents/biod/duomenys
path = %pwd
directories = !find . -type d
direktories
['.', './csv', './img', './mat', './pdf', './t
xt', './xlsx']
%who
direktories
%reset
Once deleted, variables cannot be recovered. Proceed (y
/n)?
%who
Interactive namespace is empty.
%lsmagic
%magic
```

Bash commands in python

```
In [1]: alias
Total number of aliases: 16
Out[1]:
[('cat', 'cat'),
 ('clear', 'clear'),
 ('cp', 'cp'),
 ('ldir', 'ls -F -o --color %l | grep /$'),
 ('less', 'less'),
 ('lf', 'ls -F -o --color %l | grep ^-'),
 ('lk', 'ls -F -o --color %l | grep ^l'),
 ('ll', 'ls -F -o --color'),
 ('ls', 'ls -F --color'),
 ('lx', 'ls -F -o --color %l | grep ^-..x'),
 ('man', 'man'),
 ('mkdir', 'mkdir'),
 ('more', 'more'),
 ('mv', 'mv'),
 ('rm', 'rm'),
 ('rmdir', 'rmdir')]
```

IPython

- Ctrl-l clears text in terminal
- Ctrl-c aborts command execution
- Ctrl-d quit python

```
help(len)

Help on built-in function len in module builtins:

len(obj, /)
    Return the number of items in a container.
(END)

len?
Signature: len(obj, /)
Docstring: Return the number of items in a container.
Type:      builtin_function_or_method
```

IPython output

```
5+5
10

7+2
9

print(In)
['',5+5', '7+2', 'print(In)']

 Out[1]+Out[2]
 19
_
 19
_2
 9
_2;
42 in Out
 False
```

http://ipython.org/

Reserved keywords

```
False      await      else       import     pass
None       break      except     in         raise
True       class      finally    is         return
and        continue   for        lambda     try
as         def        from       nonlocal   while
assert     del        global     not        with
async      elif       if         or         yield
```

https://analyticsindiamag.com/top-10-python-open-source-projects-on-github-2019

Comments

```
#!/usr/bin/env python37
# -*- coding: utf-8 -*-

2+2 # Comment is ignored

def function_name():
    ””” docstring example

    This comment is accesible from .__doc__ atribute
    ”””

# help(function_name)
print(function_name.__doc__)
docstring

    This comment is accesible from .__doc__ atribute
```

https://www.python.org/dev/peps/pep-0257/

Data types

```
# Integers
10
10
0b10
2
# Float
type(3.14)
float
4e2
400.0
# Boolean
type(True)
bool
# Complex
type(3+4j)
complex
# Strings;
path = '~/Documents/bioa/data'
```

List

```
list()
# mutable, ordered objects
a = [1, 'text', True,'3.14', len, int]
[1, 'text', True, '3.14', <function len(obj, /)>, int]
a[0]
1
a[-1]
int
a[1:4]
['text', True, '3.14']
1 in a
True
b = ['nested',a]
['nested', [1, 'text', True, '3.14', <functio n len(obj, /)>, int]]
b[1][1]
'text'
```

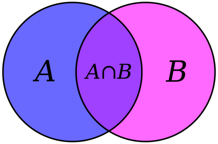
Tuple

```
tuple()
# immutable
t = ('a','b','c')
42, 'text'
(42, 'text')
a,b,c=t # unpack
a
'a'
```

Dictionary

```
dict()
# mutable, dynamic, nested, elements accessed via keys
human = {}
type(human)
dict
human['name'] = 'John'
human['gender]='male'
human['age']=40
human
{'name': 'John', 'gender': 'male', 'age': 40}
{(1,1):45, (1, 2):5}
{(1, 1): 45, (1, 2): 5}
# Usefull methods
human.keys(); human.clear(): human.popitem(); human.update()
```

Set



```
set()
s = 'quux'
list(s)
['q', 'u', 'u', 'x']
set(s)
{'x', 'u', 'q'}
```

Print

```
print('long senctence\
       can be divided into \
       parats using \\')
long senctence can be divided into parats using \

# with r
print(r'raw\nkeyward\tprints all symbols\\')
raw\nkeyward\tsprints all symbols\\

# without r
print('raw\nvkeyward\tprints all symbols\\')
raw
keyward         prints all symbols\
```

Print

```
name = 'John'
age = 22
”Name:, %s. Age: %s.” % (name, age)
”Name, {0}. Age {1}.”.format(name, age)
f”Name, {name}. Age {age}.”
'Name, John. Age 22.'

f”Name, {name.lower()}. age {age}.”
'Name, john. age 22.'
```

Packages

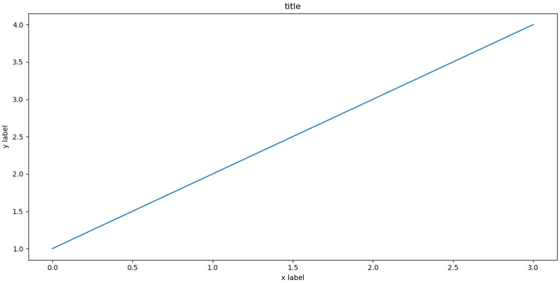
```python
import math

math.sqrt(4)

from math import sqrt
sqrt(4)
from math import *
sqrt(4)
```

Plotting

```
import matplotlib.pyplot as plt
plt.figure(figsize=(20, 3))
plt.plot([1,2,3,4])
plt.title(”title”)
plt.xlabel(”x label”)
plt.ylabel(”y label”)
```



Functions

```
Fahrenheit = 100
Celsius = (Fahrenheit - 32) * 5.0/9.0
Fahrenheit = 9.0/5.0 * Celsius + 32

def celcius_to_farenheit(celcius):
    fahrenheit = 9.0/5.0 * celcius + 32
    return fahrenheit, celcius

celcius_to_farenheit(29)
(84.2, 29)
```