Pandas

Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

- Pandas cheat cheet
- Official pandas website
- Video tutorial Series
- 8 ressources for data analysis with Pandas

LPP dev guide

Efficient scientific coding

You will find here material to help you get started with your coding. Depending on whether you mostly do data analysis or numerical modeling, you'll need slightly different tools and methods. the advice we give you will help to be efficient, rigorous and to write code than you'll be able to use, maintain and share in the long run. Now remember, **If you're a PhD or a post-doc**, following these advice will not only help you improving the quality and reproducibility of your science, but also will make all your coding efforts **reusable** for your future you and by people in the lab once you're gone. More importantly, it will give you the basic knowledge you need to legitimately claim for a data science / computing science position in the private sector.

- The commandments of programming
- Common tools and methodology you'll need to get started.

Now, what are **you** doing?

- I'm mostly coding for data analysis
- I'm mostly coding for numerical modeling

Code review and analysis

Review

Code Review for Teams Too Busy to Review Code (youtube video)
Nice tutorial on code review with Rhodecode

Analysis

CPP Check

Performance

Videos

Modern C++

Optimizations

Instruction tables

Setting up a clean Python environment

- Python and virtualenv

C++ development

- The C++ language
- Design Pattern in general and in C++
- C++ Gurus
- patterns in C++

Courses

These are useful links to check out regularly

PRACE training: https://events.prace-ri.eu/category/2/

Catalogue of courses: http://formation-calcul.fr/

Formation IDRIS: https://cours.idris.fr/php-plan/affiche_planning.php?total

Code Design and Architecture

- The S.O.L.I.D. principles

Writing code

- Code review
- Useful resources

Documentation

- [[hyb-par: Documentationtools| Documentation Tools]]

Dev Dej games

1 - show me your snippet
2 - explique à ton voisin
3 - montre nous ton blog préféré
4 - pull me quizz
5 - sell me (some features of) your editor
6 - critic my code
7 - show me a youtube video
8 - optimize my snippet
9 - translate my code
10 - Ze lib of Ze Week

Code review

- Code Review for Teams Too Busy to Review Code (youtube video)
- Nice tutorial on code review with Rhodecode

C++ Gurus

- Bjarne Stroustrup, is the inventor of C++ (personal website)
- Scott Meyers (personal blog)
- https://herbsutter.com blog and web site

Programming with C++

Books

Here are some references on C++:

- The C++ Programming language, Fourth edition, Bjarne Stroustrup is C++11/14 is the exhaustive bible/reference from B. Stroutrup, the inventor of C++
- Programming Principles and Practice Using C++ 2nd edition (C++11/14) is a book for new comers to C++, from basics to more advanced topics it has nice explanations and comes with exercices (solutions here)
- A tour of C++ Pocket version presenting modern C++ by B. Stroutrup.
- Effective C++: 55 Specific Ways to Improve Your Programs and Designs Nice book written by Scott Meyers. Is pre-C++11 but still very useful.
- Effective Modern C++ 42 ways to improve Your Use of C++11 and C++14, from Scott Meyers, highly recommended to understand the new way of developing after C++11. Is for intermediate to advanced C++ programmers.
- Scientific and Engineering C++: An Introduction with Advanced Techniques and Examples Pre-C++11, very good to learn C++ (for C and Fortran programmers) and presents design techniques.

Videos

- Modern C++
- Effective C++11/14 Nice video from Scott Meyers (authors of the book effective C++11/14). very good explanation of std::move/std::forward
- Programming using C++ : Video Tutorials

FAQ

C++ FAQ
Don't be Afraid of Returning by Value, Know the Return Value Optimization
Copy elision

Smart pointers

- When should I use raw pointers over smart pointers?
- Which kind of pointer do I use and when?

Data analysis FAQ

General Data Analysis

Space Plasmas Analysis

General Data Analysis

Which library should I use to handle data series?

You will want to use the python Pandas (check here for Pandas tips) for analyzing your data. It is in particular very well suited for time series analysis and enable you to very easily manipulate dates, missing data, etc.

Is there a library for plotting statistical data?

Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface for drawing attractive statistical graphics. Check here for seaborn tips

Where can I learn machine learning?

Check out this page to find many links on machine learning.

Space Plasmas Data Analysis

How can I open a CDF file?

You should install Spacepy on your machine. Then import the module spacepy.pycdf to load CDF files. Documentation for PYCDF is here

How can I change coordinates systems in planetary systems other than Earth?

We will need to download and load some **SPICE** kernels (that indicate the position of the satellite, its orientation ...etc) from the NAIF servers.

The SPICE kernels file contents are summarized below:

S- Spacecraft ephemeris, given as a function of time. (SPK)
P- Planet, satellite, comet, or asteroid ephemerides, or more generally, location of any target body, given as a function of time. (also SPK)
The P kernel also logically includes certain physical, dynamical and cartographic constants for target bodies, such as size and shape specifications, and orientation of the spin axis and prime meridian. (PCK)
I- Instrument description kernel, containing descriptive data peculiar to a particular scientific instrument, such as field-of-view size, shape and orientation parameters. (IK)
C- Pointing kernel, containing a transformation, traditionally called the "C-matrix," which provides time-tagged pointing (orientation) angles for a spacecraft bus or a spacecraft structure upon which science instruments are mounted. A C-kernel may also include angular rate data for that structure. (CK)
E- Events kernel, summarizing mission activities - both planned and unanticipated. Events data are contained in the SPICE EK file set, which consists of three components: Science Plans, Sequences, and Notes. (EK)

Some additional data products are also important components of the SPICE system, even if not contained in the "SPICE" acronym.

A "frames kernel" **(FK)** contains specifications for the assortment of reference frames that are typically used by flight projects. This file also includes mounting alignment information for instruments, antennas and perhaps other structures of interest.

Spacecraft clock **(SCLK)** and leap seconds **(LSK)** kernels are also part of SPICE; these are used in converting time tags between various time measurement systems.

Under development is a digital shape model kernel (DSK) for both small, irregularly shaped bodies such as asteroids and comet nuclei, and for large, more uniformly shaped bodies such as the moon, earth and Mars. Other kernel types can be added as requirements arise and time permits.

For more information, please consult the NAIF Homepage

What languages support the SPICE kernels toolkit?

C, FORTRAN, IDL, MATLAB

and

Spiceypy for PYTHON

How can I get the CASSINI spacecraft axes in the SSE (same as KSO) coordinate systems?

We should visit the CASSINI Spacecraft Attitude tool.
We choose the "Time Range", "Time Interval", and **SSE Spacecraft Axes (SSE)** for the "Attitude Type".

Machine learning

Tutorials

- 10 videos tutorial of scikit-learn and machine learning.
- 28 videos on machine learning with Scikit-Learn and Python.

Seaborn

- official seaborn website

Design Patterns

References and books

- Source Making is a very nice website gathering design patterns, anti-patterns and refactoring rules
- Wikipedia list of usual design patterns
- Design Pattern explained simply
- Head First Design patterns
- WikiBook design patterns

C++ Patterns and code tricks

- Copy constructor for class having abstract class as attribute
    - http://stackoverflow.com/questions/4507565/problem-with-copy-constructor-with-class-with-polymorphic-pointers
- Secrets of good OO design
    - http://stackoverflow.com/questions/3758244/secret-to-achieve-good-oo-design
    - Separation of concerns : https://en.wikipedia.org/wiki/Separation_of_concerns
    - D.R.Y. (Don't Repeat Yourself) https://en.wikipedia.org/wiki/Don%27t_repeat_yourself
    - YAGNI (you arn't gonna need it) : https://en.wikipedia.org/wiki/You_aren%27t_gonna_need_it

Common tips

Python

- Where can I get notebook examples? ?

C++

- Classical C/C++ pitfalls explained by INTEL
- The definitive C++ book guide and list

Debugging

- What tools to find bugs in C and C++ ?

Python

Where can I get notebook examples?

check the notebook gallery : Links to the best IPython and Jupyter Notebooks.

Debugging

What tools to find bugs in C and C++ ?

Use google sanitizers, here is a video that explains what they are and here is the github access.

Read Documentation you will need

- Do you know devdocs ? Whether you're coding in C++, C, python, etc. there's a documentation for you. Ok there's none for IDL and poor FORTRAN...

Version Control

- git cheat sheet

Numerical simulations

High Performance Computing

General Data Analysis

What is the effect of the different cache on performance?

You can have very good information on CPU caches on this blog page.

What should I know about memory?

There is a **very** good and complete review on computer memories here. In particular the section 6 is good to know for performance programming.

Python and virtualenv

Why?

Usually with Python you may ear from your colleagues "Hey, you may use package X to do this it's much better than Y and easier" then you discover that package X isn't packaged in your distribution or you install it and it mess up your system. For example Jupyter isn't packaged yet in Fedora and installing it directly with pip may break your already installed version of IPython-3x.
To use last version of your favourite Python tools such as Jupyter notebooks or Pandas on your system without messing up your computer, virtualenv is a solution.
It will allow you to install any python package with pip in an isolated environment so your system will not see your packages until you activate it.
You will be able to have as many virtual environments as you want with different packages and different versions in each environment.

How?

1. First get virtualenv

On Fedora:

```
sudo dnf install python*-virtualenv  python-qt5-devel python3-qt5-devel
```

On Mac OS with port:

```
sudo port install py35-virtualenv
sudo port install py27-virtualenv
```

Will install both Python2 and Python3 versions of virtualenv.

1. Create your environments

```
sudo mkdir /opt/Py2Venv /opt/Py3Venv
sudo chown -R <yourLogin> /opt/Py2Venv /opt/Py3Venv
virtualenv-2.7 --system-site-packages /opt/Py2Venv
virtualenv-3.4 --system-site-packages /opt/Py3Venv
```

Now you have two basic virtual environments for Python 2 and 3. Note that you can remove the --system-site-packages flag to tell that you want your environment to ignore system-wide packages it may protect you from some local and global packages incompatibilities. In most cases you may create it with --system-site-packages flag.

1. Use it

Your environment is ready to play, to use it you have to **activate** it.

```
source /opt/Py3Venv/bin/activate
```

Then it will override your system pip command by your current virtualenv one and all packages installed with pip while your environment is activated will be installed in your environment.
If the activate command worked you may see you shell prompt like this:

```
(Py3Venv)[adminlpp@pc-instru opt]$
```

Note the **(Py3Venv)** this says that Py3Venv is activated. To quit/deactivate it just use the command **deactivate**.

Let's install Pandas, Jupyter, numpy...

```
(Py3Venv)[adminlpp@pc-instru opt]$ pip install pandas
(Py3Venv)[adminlpp@pc-instru opt]$ pip install jupyter
```

Note that some python packages depends on system libraries, you may need to install them plus the devel packages

The SOLID principles

S : Separation of concerns principle
O : Open-Closed principle
L : Liskov substitution Principle
I : Interface segregation Principle
D : Dependency Inversion Principle

Videos

- Short videos for each of the principles (youtube)
- Bob Martin on SOLID principles (youtube)

Useful resources

Random Dev Blogs

A Random Walk Through Geek-Space
I Like Big Bits

Algorithms

Log Structured Merge Trees
Algebraic patterns - Semigroup
EXACT STRING MATCHING ALGORITHMS
A Memory Allocator
LL and LR Parsing Demystified

Tips

Falsehoods programmers believe about time
GCC optimizations for embedded software

Python programming

Useful libraries for data science in Python · GitHub
Example Google Style Python Docstrings
Welcome to Bokeh

Jupyter Notebook gallery

Notebook Gallery
A gallery of interesting IPython Notebooks

Videos

SciPy 2016: "Data Science is Software" tutorial

C programming

Bit Twiddling Hacks
The Lost Art of C Structure Packing
Memory management in C programs
Const and Optimization in C
You Can't Always Hash Pointers in C
Four Ways to Compile C for Windows