EXERCICE
Introduction à Python Scientifique

Pr Marc BUFFAT
département mécanique,Université Claude Bernard Lyon 1

9 février 2015

In [23]:
```
%matplotlib inline
%autosave 300
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
from matplotlib import rcParams
rcParams['font.family'] = 'serif'
rcParams['font.size'] = 14
from IPython.core.display import HTML
from IPython.display import display
from matplotlib import animation
#from JSAnimation import IPython_display
css_file = 'style.css'
HTML(open(css_file, "r").read())
```

```
Autosaving every 300 seconds
```

Out[23]:
```
<IPython.core.display.HTML at 0x7f25ac9ad8d0>
```

1 Introduction à la programmation scientifique avec Python

from the course "Scientific Python Lectures" by Robert Johansson

1.1 Marc BUFFAT, dpt mécanique, Université Claude Bernard Lyon 1

2 Exigences du calcul scientifique

2.0.1 "Le logiciel (software) est une des pierres angulaires de la science moderne. Sans logiciel, la science du vingt et unième siècle serait impossible. Mais sans de meilleurs logiciels, la science ne peut pas progresser" [http ://sciencecodemanifesto.org/]

** Réplicabilité et de reproductibilité ** sont des pierres angulaires de la méthode scientifique. En ce qui concerne le travail numérique, se conformer à ces concepts a des implications pratiques suivantes :

- **Réplicable :** L'auteur d'un article scientifique qui implique des calculs numériques doit être en mesure de relancer les simulations et de reproduire les résultats sur demande. D'autres scientifiques devraient également être en mesure d'effectuer les mêmes calculs et d'obtenir les mêmes résultats, compte tenu des informations sur les méthodes utilisées dans une publication.
- **Reproductible :** Les résultats obtenus à partir de simulations numériques doivent être reproductibles avec une mise en oeuvre indépendante du procédé, ou en utilisant un procédé tout à fait différent.

En résumé : un résultat scientifique solide doit être reproductible, et une étude scientifique solide doit être réplicable.

— importance de la **documentation**
— importance de la **validation**
— importance de la **gestion de version** (git)

3 Langage de programmation Python

[Python] (http ://www.python.org/) est un langage de programmation moderne de haut niveau, orienté objet et d'usage général.

Caractéristiques générales de Python :

— ** Langage simple : ** facile à lire et à apprendre avec une syntaxe minimaliste.
— ** Langage concis et expressif : ** moins de lignes de code, moins de bugs, plus facile à maintenir.

Détails techniques :

— ** Typé dynamiquement : ** Pas besoin de définir le type des variables, les arguments ou le type des fonctions.
— ** La gestion automatique de la mémoire : ** Aucune nécessité d'allouer explicitement et désallouer la mémoire pour les variables et les tableaux de données. Aucun bug de fuite de mémoire.
— ** Interprété : ** Pas besoin de compiler le code. L'interpréteur Python lit et exécute le code python directement.

3.0.2 Avantages :

— Le principal avantage est la facilité de programmation, qui minimise le temps nécessaire pour développer, déboguer et maintenir le code.
— Langage bien conçu qui encouragent les bonnes pratiques de programmation :
— Modulaire et orientée objet, permet l'encapsulation et la réutilisation de code. Il en résulte souvent un code plus transparent, maintenable et sans bug.
— Documentation intégré avec le code.
— De nombreuses bibliothèques standards, et de nombreux packages add-on.

3.0.3 Inconvénients :

— Puisque Python est un langage de programmation interprété et typé dynamiquement, l'exécution de code python peut être lent par rapport à des langages de programmation compilés à typage statique, tels que C et Fortran.
— Un peu trop décentralisé, avec différents environnements, bibliothéques, et documentation répartis à différents endroits. Cela peut le rendre difficile pour commencer.

4 Python en calcul scientifique

— Python est largement utilisé en calcul scientifique :
  — Large communauté d'utilisateurs, aide et documentation facile à trouver.
— Large écosystème de bibliothèques et d'environnement scientifiques
  — Numpy : http ://numpy.scipy.org - Numerical Python
  — Scipy : http ://www.scipy.org - Python Scientifique
  — Matplotlib : http ://www.matplotlib.org - bibliothèque graphique
— Bonne performance grâce à l'utilisation de codes éprouvés et hautement optimisés écrits en C et Fortran :
  — Blas, altas blas, lapack, arpack, Intel MKL, ...
— Un bon support pour
  — Le traitement parallèle avec les processus et les threads

  - Communication parallèle (MPI)
  - GPU computing (OpenCL et CUDA)
- Facilement disponible sur les grands centres HPC
- Pas de coût de licence.

5 Scientific Python software stack

6 Environnement Python

6.1 Interpréteur Python

La méthode standard pour utiliser le langage de programmation Python est d'utiliser l'interpréteur Python pour exécuter du code python. L'interpréteur Python est un programme qui lire et exécuter le code python dans les fichiers qui lui sont passés comme arguments. À l'invite de commande, la commande `python` est utilisé pour invoquer l'interpréteur **Python**. Par exemple pour exécuter un fichier `mon-programme.py` qui contient le code python partir de l'invite de commande $, utilisez

```
buffat@p2chpd-visu2:~$ python mon-programme.py
```

Nous pouvons également commencer l'interprète en tapant simplement `python` à la ligne de commande, et de manière interactive taper le code Python dans l'interpréteur.

On peut aussi utiliser l'interpréteur `IPython` **Ipython** est une version plus `user-friendly`, qui apporte :

- Hystorique des commandes, qui peut être consulté avec les flèches haut et bas sur le clavier.
- Auto-complétion (avec Tab).
- Edition de code dans ligne.
- Introspection des objets, et extraction automatique des chaînes de documentation des objets python comme les classes et les fonctions.
- Bonne interaction avec le shell et le système d'exploitation.
- Support de calcul parallele (cloud computing).

6.2 Editeur de texte

Pour écrire des programmes Pythons on utilise un éditeur de texte, avec coloration syntaxique

- **vi** (vim) ou gedit

6.3 Environnement de développement Spyder

Spyder est un IDE à la MATLAB pour python scientifique

6.4 Ipython Notebook

IPython notebook est un environnement portable sous HTML pour Python, similaire à Mathematica ou maple. Il est basé sur un shell IPython, mais offre un environnement à base de cellules avec une grande interactivité, où les calculs peuvent être organisées documentée de manière structurée.

7 Installation d'un environnement Python scientifique

7.1 Linux

sous Ubuntu on utilise le gestionnaire de paquet debian

```
$ sudo apt-get install python ipython ipython-notebook

$ sudo apt-get install python-numpy python-scipy python-matplotlib python-sympy
```

```
$ sudo apt-get install spyder
```

7.2 Windows

Windows manque d'un bon système de gestion de packages, de sorte que le moyen le plus facile pour mettre en place un environnement Python est d'installer une distribution pré-packagée. Quelques bonnes alternatives sont

— Enthought Python Distribution. EPD commercial product but available free for academic use.
— Anaconda CE. Anaconda Pro commercial product, but Anaconda Community Edition is free.
— Python(x,y). Fully open source.

8 Documents et sites Web sur Python

— Python. The official Python web site.
— Python tutorials. The official Python tutorials.
— Think Python. ''How to Think Like a Computer Scientist" by Allen B. Downey (free book).
— MMOC InProS ''Introduction à la Programmation Scientifique"

9 Version de Python et des bibliothéques

```
In [24]: import sys
         print "Version de Python:\t",sys.version
         import IPython
         print "Version de IPython:\t",IPython.__version__
         import numpy
         print "Version de numpy:\t",numpy.version.version
         import scipy
         print "Version de scipy:\t",scipy.version.version
         import matplotlib
         print "Version de matplotlib:\t",matplotlib.__version__
```

```
Version de Python:        2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2]
Version de IPython:       2.2.0
Version de numpy:        1.8.2
Version de scipy:        0.13.3
Version de matplotlib:        1.3.1
```

10 Fin de la leçon