I- Introduction

Année 2023 - 2024

Tous les documents sont disponibles sur l'ENT

I-1 Environnement

En quoi consiste ce module ?

Organisation du module

- **Durée :** 6 semaines
- **Outils :** notions, langages, TPs, supports, etc.
- **Projets :** par binôme à partir de la 3ème semaine
- **Evaluation :** code source du projet, un rapport en pdf ou des notebooks détaillées et présentation orale

Langage de programmation : pourquoi Python ?

Python est un langage de programmation moderne, polyvalent, orienté objet et de haut niveau. Il ne cesse de gagner de popularité. Plus utilisé par les acteurs majeurs dans le domaine tels que google, amazon, netflix, fecabook, youtube, la NASA, etc.

Caractéristiques de Python :

- **langage simple :** Le code est facile à lire et intuitif, la syntaxe minimaliste facilite l'apprentissage ainsi que la maintenance ce qui lui donne une flexibilité d'adaptation à la taille des projets.
- **Prise en main:**
  La prise en main est relativement facile avec moins de lignes de code, moins de bugs et une gestion simple. C'est le cas dans ce module. Les curieux auront l'opportunité d'aller plus loin ...

Détails techniques :

- **Typage dynamique :** Pas besoin de définir le type de variables, d'arguments de fonction ou de types de retour.

- **Gestion automatique de la mémoire :** Pas besoin d'allouer et de libérer explicitement de la mémoire pour les variables et les tableaux de données. Aucun bug qui sera dû à la fuite de mémoire.
- **interprété :** Pas besoin de compiler le code. L'interpréteur Python lit et exécute le code python directement.

Avantages :

- Le principal avantage reste la facilité de coder. Ce qui minimise le temps nécessaire au développement, au débogage et à la maintenance du code.
- Un langage bien conçu qui encourage beaucoup de bonnes pratiques de programmation :
- Programmation orientée objet qui peut amener à la réutilisation du code.
- Documentation intégrée au code. Il suffit d'utiliser le point d'interrogation suivi de la commande
- A vrai dire, si Python gagne en popularité c'est qu'il a permis à plusieurs codeurs de différents domaines de développer leur packages (bibliothèques), qui à leur tour ont permis d'en créer d'autres, etc...
- **Plateformes :** Il existe, aujourd'hui, des plateformes qui offrent un environement riche.

Inconvénients :

- L'exécution est plus lente que d'autres langages de typage statique (C/C++) mais mieux que d'autres !!
- A force de chercher on trouvera les différentes packages liés à une spécialité. Mais il est possible que la documentation ne soit pas toujours exploitable !!

Un bon choix pour les sciences de données & IA

- Python occupe une bonne place en calcul scientifique :
  - L'agrandissement de la communauté d'utilisateurs facilite le développement, le débogage et documentation.
- Un large choix de librairies. Entre autres on verra :
  - numpy : http://numpy.scipy.org
  - scipy : http://www.scipy.org
  - matplotlib : http://www.matplotlib.org
  - Pandas : https://pandas.pydata.org

    - Sklearn : https://scikit-learn.org/stable/index.html
- Meilleures performances grâce à une intégration étroite avec des codes éprouvés et hautement optimisés écrits en C et en Fortran.
- Calcul parallèle ou distribué
    - Processus et threads (multi-processeurs)
    - Communication Inter-process
    - GPU (OpenCL and CUDA) et plateformes cloud (Nvidia est leader dans le domaine)
- Peut être Facilement adapté à une utilisation sur des clusters informatiques ou datacenter pour les hautes performances (SSH)
- Plusieurs plateformes gratuites et payantes avec des environnements variés.

I-2 Langage

Langage interprétable

Python n'est pas seulement un langage de programmation, il fait également référence à la mise en oeuvre standard de l'interpréteur (techniquement appelé CPython) qui exécute le code python sur un ordinateur.

Il existe également de nombreux environnements différents dans lesquels l'interpréteur python peut être utilisé. Chaque environnement présente des avantages différents et convient à différents cas de travail. Nous allons donc commencer par un bref aperçu des environnements python utiles au calcul scientifique dans le cadre de ce module.

On peut le lancer dans un terminal avec la commande **$ python** où **$** représente le shell. Par exemple, nous pouvons écrire un script qui contient les commandes et le sauvegarder avec une extension `.py` . Imaginons un script `mon-programe.py` qui contient un code on peut l'exécuter avec :

```
$  python my-program.py
```

In [ ]:
```
# python3 mon_programme.py
```

In [ ]:
```
#ceci est un commentaire
# mon-programme
#créer un fichier qui contient la ligne suivante et sauvegarder sous
#le nom mon-programme.py

#!/usr/bin/python3
#-*- coding: Utf-8
print('Bienvenue dans hello hello  ce module  !!! ')
a=100.01
print(a)
```

```
#exécuter le programme dans un terminal avec la commande :  python3 mon-prog
#vous devez avoir le résultat suivant :
```

C'est souvent ainsi que nous souhaitons travailler lors du développement d'applications scientifiques ou lors de petits calculs. Mais l'interpréteur Python standard n'est pas très pratique pour ce type de travail.

IPython notebook

IPython notebook est un environnement basé sur HTML pour Python. Il est basé sur le shell IPython, mais fournit un environnement basé sur les **cellules** avec une grande interactivité, dans lequel les calculs peuvent être organisés et documentés de manière structurée.

In [ ]:
```
jupyter notebook

print('dans une cellule de code 2*3 =', 2*3)
```

---> et dans une cellule Markdown

I-3 Quelques exemples

On lance une notebook à l'aide de la commande **jupyter notebook** (https://ipython.org/).

Une fenêtre s'ouvre sur le navigateur par défaut avec le **home** (répertoire courant).

On clique **new** et on choisit un feuille (notebook) python 3

On peut lui donner un nom ou la renommer plus tard. Elle apparaitra automatiquement dans le home

In [ ]:
```
# Déclaration simple de la variable a
a=10
# Déclaration simple de la variable b
b=20
# Affichage de la somme
print('la somme de ',a,'et',b,'est :',(a+b))
print(type(a))
```

In [ ]:
```
x=0
y=1
x, y = y, x

print('x:',x)
print('y:',y)
```

II Installation

Pour installer Python, il suffit de télécharger la version 3 qui correspond à votre système d'exploitation à l'adresse : http://www.Python.org/

En particulier, Python est installé par défaut dans Linux et Mac os. Il reste à vérifier que vous avez Python3 sinon il faut mettre à jour.

Pour vérifier : ouvrez un terminal et taper la commande "python -- version"

ou dans une cellule :

pip ou conda install matplotlib

- Sur Linux, on peut utiliser les commandes suivantes comme administrateur :
- $ sudo apt-get install python ipython ipython-notebook
- $ sudo apt-get install spyder (ou pycharm)
- Il est conseillé de travailler dans un environnement virtuel. On expliquera par la suite le comment et les avantages

Pour l'instant on se contentera de deux gestionnaires de paquets utilisés pour installer et gérer des paquets écrits en Python.

PIP

Sans doute le gestionnaire de paquets le plus utilisé pour le langage Python. Comme Python utilise un grand nombre de librairies, plusieurs développeurs ont décidé de créer ces gestionnaires de paquets pour faciliter la gestion.

Pour vérifier qu'il est déjà installé : on tape la commande $ pip --version

Si ce n'est pas le cas, vous pouvez trouver la dernière version ainsi que tous les détails nécessaires sur le site officiel :

https://pypi.org/project/pip/

Conda

La grande différence entre conda et le gestionnaire de paquets pip réside dans la façon dont les dépendances sont gérées, ce qui représente une motivation/choix de travailler avec l'un ou l'autre. Un tel choix peut être important pour les grands projets ou pour un système d'exploitation particulier. Dans le cadre de ce cours, on travaillera avec pip pour

linux/macos qui est largement au dessus de nos exigences. Pour windows, conda semble être le mieux !!

Pour vérifier qu'il est déjà installé : avec la commande $ conda --version

Si ce n'est pas le cas, vous pouvez trouver la dernière version ainsi que tous les détails nécessaires sur le site officiel :

https://docs.conda.io/en/latest/

Questions diverses

1. C'est quoi un compilateur ?
2. C'est quoi un interpréteur ?
3. Quelles sont les différences et les similitudes ?
4. De quoi a t on besoin pour exécuter un programme Python ?
5. Citer quelques avantages et inconvinients de programmer avec Python ?