Département de Mathématiques 2017/2018
Module: Analyse numérique 1
Fiche de T.P. N°5

Modules et importations

Les modules sont des fichiers qui regroupent des ensembles de fonctions. En effet, il peut être commode de découper un programme important en plusieurs fichiers de taille modeste pour en faciliter la maintenance et le travail collectif. Une application Python typique sera alors constituée d'un programme principal accompagné de un ou plusieurs modules contenant chacun les définitions d'un certain nombre de fonctions accessoires.
Il existe un grand nombre de modules pré-programmés qui sont fournis d'office avec Python. Souvent on essaie de regrouper dans un même module des ensembles de fonctions apparentées que l'on appelle des bibliothèques.
Un module peut être appelé depuis plusieurs programmes, il s'agit d'un fichier .py commençant par son identité (qui ne contient pas de point).
N'importe quel fichier .py peut donc être appelé depuis un autre comme un module. Il peut contenir :
du script
des fonctions
...
Par exemple, nous allons créer un fichier nommé equation_non_lineaire.py qui va définir 3 fonctions : bissection(), point_fixe() et Newton(). Un tel fichier est appelé un module et il va pouvoir être importé dans un autre fichier, et en particulier dans le fichier qui contient le programme principal.

```
def bissection(f,a,b,tol,nmax):
    ......
    .....
    return resultat
def point_fixe(f,a,b,tol,nmax):
    ......
    .....
    return resultat
def Newton(f,a,b,tol,nmax):
    ......
    .....
    return resultat
```

Il est maintenant possible d'utiliser dans un programme principal les fonctions qui ont été définies dans le module equation_non_lineaire.py. Pour cela, il faut importer les fonctions à partir du module.

Importer un module

Pour utiliser des fonctions de modules dans un programme, il faut au début du fichier importer ceux-ci.

Pour ce faire, utiliser la commande "**import**" :

On importe une seule fonction

Exemple

```
from equation_non_lineaire import bissection
a,b=1,2
f=lambda x:x**2-2
x=bissection(f,1,2,1e-6,100)
```

On importe le module

Exemple

```
import equation_non_lineaire
a,b=1,2
f=lambda x:x**2-2
x=equation_non_lineaire.bissection(f,1,2,1e-6,100)
```

On importe le module et on lui donne un alias

Exemple

```
import equation_non_lineaire as en
a,b=1,2
f=lambda x:x**2-2
x=en.bissection(f,1,2,1e-6,100)
```

On importe une fonction d'un module et on lui donne un alias

Exemple

```
from equation_non_lineaire import bissection as bis
a,b=1,2
f=lambda x:x**2-2
x=bis(f,1,2,1e-6,100)
```

Exécution d'un module en tant que script

À l'intérieur d'un module, son propre nom est accessible dans la variable __name__. Quand on utilise un fichier en tant que script, la variable globale __name__ prend pour valeur '__main__'. Ceci permet d'avoir dans le fichier un bloc d'instructions qui sera exécuté uniquement lorsque le fichier est lancé en tant que script.

Par exemple

```
if __name__ == "__main__":
    a,b=1,2
    f=lambda x:x**2-2
    x=bissection(f,1,2,1e-6,100)
    print('la solution est',x)
```