TD – Calcul numérique

Introduction et objectifs

L'objectif de ce TD consacré au calcul numérique est le codage d'algorithmes simples pour les problèmes suivants :

- Approximation de dérivées.
- Résolution de l'équation $f(x)=0$.
- Approximation d'intégrales définies.

Approximation de dérivées

Exercice N°1 – Mise en œuvre

Ecrire une fonction Python `Delta` qui reçoit en arguments le nom `f` d'une fonction dérivable et un écart algébrique `h` (cf. le cours) et renvoie une fonction (on utilisera la définition à l'aide du mot-clé `lambda`) correspondant à l'approximation suivante de f' en tout réel x où f est dérivable :

$$f'(x) \simeq \frac{f(x+h)-f(x-h)}{2h}$$

On pourra tester la fonction `Delta` avec la fonction $x \mapsto \cos^3(x)$ dont on peut facilement, après linéarisation, calculer les dérivées d'ordre quelconque, $x = \frac{\pi}{6}$ et $h = 0,01$.

On pourra également utiliser : $P : x \mapsto x^{11} - x^{10} + 2x^7 - 3x^6 - 2x^5 + x^4 - 6$ qui donne :

$$P'(1) = -9,\ P''(1) = -14,\ P^{(3)}(1) = 234,\ P^{(4)}(1) = 3\,264 \text{ et } P^{(5)}(1) = 27\,840$$

On prendra cette fois : $h = 0,01$ puis $h = 0,001$.

Résolution de l'équation *f*(*x*) = 0

Exercice N°2 - Trichotomie

En vous inspirant de la méthode de dichotomie, écrire une fonction Python `Tricho` mettant en œuvre la méthode de trichotomie.
La fonction recevra comme arguments :

- Le nom `f` d'une fonction (qui sera définie par ailleurs dans le script).
- deux flottants `a` et `b` correspondant aux bornes du segment sur lequel la fonction s'annule.
- un flottant `eps` correspondant à la largeur de l'encadrement du zéro cherché.

La fonction renverra les valeurs des bornes du segment encadrant le zéro et le nombre d'itérations effectuées.

Par exemple, on pourra résoudre l'équation $f(x)=0$ avec la fonction :

$$f1 : x \mapsto \ln(x) - \arctan(x)$$

via l'appel :

```
Tricho(f1,1,5,1e-6)
```

Le script comportant :

```
def f1(x):
    return(log(x) – atan(x))
```

N'oubliez pas les importations !

Exercice N°3 – Méthode de Newton-Raphson SANS estimation de la dérivée

Ecrire une fonction Python `NewtonRaphson` qui reçoit comme arguments :

- Le nom `f` d'une fonction définie par ailleurs dans le script et dont on cherche un zéro.
- Le nom `df` d'une fonction définie par ailleurs dans le script et correspondant à la fonction dérivée f' de lka fonction *f*.
- Un flottant `x_init` correspondant à l'abscisse initiale x_0.
- Deux flottants `epsx` et `epsy` correspondant à la précision sur les abscisses et sur les ordonnées.
- Un entier nmax correspondant au nombre maximal d'itérations effectuées (typiquement on prendra `nmax=1000` comme valeur par défaut).

Exercice N°4 – Méthode de Newton-Raphson AVEC estimation de la dérivée

Dans cette version du code la fonction `NewtonRaphson2` demandée ne recevra pas l'argument `df` de la fonction `NewtonRaphson` précédente puisque la dérivée sera estimée.

On testera les deux fonctions précédentes en cherchant le zéro des fonctions précédentes (on donne des valeurs approchées entre parenthèses et la valeur de x_0) :

$x \mapsto x^3 - 3x + 3$	$2{,}103\,803\,402\,735$	$x_0 = 10$
$x \mapsto x^3 - x - 1$	$1{,}324\,717\,957\,244$	$x_0 = 10$
$x \mapsto \dfrac{1}{x} - \sin(x)$	$1{,}114\,157\,140\,871\,9$	$x_0 = 0{,}1$
$x \mapsto \ln(x) - \arctan(x)$	$3{,}692\,585\,685\,455$	$x_0 = 10$

Intégrales définies

On testera les fonctions des exercices de cette partie avec :

$$x \mapsto x^2 \text{ sur } [0\,;2] \text{ et } x \mapsto \ln(x) \text{ sur }]0\,;1]$$

Exercice N°5 – Méthode des rectangles

Ecrire deux fonctions Python `IntRectG` et `IntRectD` mettant respectivement en œuvre la méthode des rectangles à gauche et à droite pour l'approximation d'une intégrale définie.
Les fonctions recevront comme arguments :

- Le nom `f` d'une fonction (qui sera définie par ailleurs dans le script) à intégrer.
- deux flottants `a` et `b` correspondant aux bornes du segment sur lequel on intègre la fonction *f*.
- un entier `n` correspondant aux nombres d'intervalles (réguliers) de la subdivision du segment $[a\,;b]$.

Exercice N°6 – Méthode des rectangles (bis)

Ecrire une fonction Python `IntRect` mettant en œuvre la méthode des rectangles.
La fonction recevra les mêmes arguments que les fonctions précédentes mais renverra un tuple correspondant aux valeurs approchées de l'intégrale correspondant à la méthode des rectangles à gauche et à droite respectivement.

Exercice N°7 – Méthode des trapèzes

Ecrire une fonction Python `IntTrap` mettant en œuvre la méthode des trapèzes.

La fonction recevra comme arguments :

- Le nom `f` d'une fonction (qui sera définie par ailleurs dans le script) à intégrer.
- deux flottants `a` et `b` correspondant aux bornes du segment sur lequel on intègre la fonction *f*.
- un entier `n` correspondant aux nombres d'intervalles (réguliers) de la subdivision du segment $[a\,;b]$.

Exercice N°8 – Méthode de Simpson

Ecrire une fonction Python `IntSimp` mettant en œuvre la méthode de Simpson.

La fonction recevra comme arguments :

- Le nom `f` d'une fonction (qui sera définie par ailleurs dans le script) à intégrer.
- deux flottants `a` et `b` correspondant aux bornes du segment sur lequel on intègre la fonction *f*.
- un entier `n` correspondant aux nombres d'intervalles (réguliers) de la subdivision du segment $[a\,;b]$. La fonction testera la parité de la variable `n` et renverra un message d'erreur si n est impair.