

TP N° 6 : MÉTHODE DE RÉGULA-FALSI (Méthodes numériques d'optimisation)

1 But

L'objectif de ce TP est de comparer différentes méthodes de résolution d'équations non linéaires du type $f(x) = 0$.

On considère un intervalle $[a, b]$ et une fonction f continue de $[a, b]$ dans \mathbb{R} . On suppose que $f(a) * f(b) < 0$ et que l'équation $f(x) = 0$ admet une unique solution α sur l'intervalle $[a, b]$.

2 Méthode de Regula-Falsi (fausse position)

Le principe de cette méthode est identique à celle de la méthode de Dichotomie. Au lieu d'utiliser le point médian, on utilise le point d'intersection avec l'axe des abscisses de la droite :

$$x(0) = a, \quad x(1) = b.$$

Ce point est donné par l'équation suivante qui se vérifie très simplement :

$$x(k) = a - f(a) * [(b - a) / (f(b) - f(a))]$$

2.1 Méthode de Dichotomie

La méthode de Dichotomie consiste à construire une suite (x_n) qui converge vers la racine α de la manière suivante :

Principe : on prend pour x_0 le milieu de l'intervalle $[a, b]$, $(x_0 = \frac{a+b}{2})$. La racine se trouve alors dans l'un des deux intervalles $]a, x_0[$ ou $]x_0, b[$ ou bien elle est égale à x_0 .

1. Si $f(x_0) = 0$, c'est la racine de f et le problème est résolu.
2. Si $f(x_0) \neq 0$, nous regardons le signe de $f(a) * f(x_0)$
 - a). Si $f(a) * f(x_0) < 0$, alors $\alpha \in]a, x_0[$
 - b). Si $f(x_0) * f(b) < 0$, alors $\alpha \in]x_0, b[$

On recommence le processus en prenant l'intervalle $[a, x_0]$ au lieu de $[a, b]$ dans le premier cas, et l'intervalle $[x_0, b]$ au lieu de $[a, b]$ dans le second cas.

De cette manière, on construit par récurrence sur n trois suites (a_n) , (b_n) et (x_{0n}) telles que $a_1 = a$, $b_1 = b$ et telles que pour tout $n \geq 0$,

- si $f(a) * f(x_0) < 0$, alors $\alpha \in]a, x_0[$. On pose $a_1 = a$, $b_1 = x_0$.
- si $f(a) * f(x_0) = 0$, alors $\alpha = x_0$.

- si $f(a) * f(x_0) > 0$, alors $\alpha \in]x_0, b[$. On pose $a_1 = x_0$, $b_1 = b$.

On prend alors pour x_1 le milieu de $[a_1, b_1]$.

On construit ainsi une suite $x_0 = (a + b)/2, x_1 = (a_1 + b_1)/2, \dots, x_n = (a_n + b_n)/2$; telle que $|\alpha - x_n| \leq \frac{(b-a)}{2^{n+1}}$. Etant donné une précision ϵ , cette méthode permet d'approcher α à un nombre prévisible d'itérations. L'algorithme ci-dessus s'appelle l'algorithme de Dichotomie.

2.2 Algorithme (Organigramme) de la méthode de Dichotomie

Entrées

Saisir a , la borne de gauche de l'intervalle

Saisir b , la borne de droite de l'intervalle ($a < b$)

Saisir la précision ϵ souhaitée

Traitement et sorties

Si $f(a)$ et $f(b)$ sont du même signe alors

| Afficher "On ne peut pas faire de Dichotomie!"

Sinon

| Tant que l'écart entre a et b est supérieur à la précision ϵ répéter

| | c prend comme valeur la moyenne des nombres a et b

| | si $f(a)$ et $f(c)$ sont de signes contraires alors

| | | Affecter la valeur c à b

| | sinon

| | | Affecter la valeur c à a

| | Fin de si

| Fin de Tant que

| | Afficher "La solution est comprise entre a et b "

Fin de si

3 Jeux de données

On travaillera avec les fonctions et les intervalles suivants :

$$\begin{cases} f_1(x) = x - e^{\sin(x)} \\ [a \ b] = [1 \ 10] \end{cases} \quad (1)$$

$$\begin{cases} f_2(x) = x^3 - 12x^2 - 60x + 46 \\ [a \ b] = [0 \ 1] \end{cases} \quad (2)$$

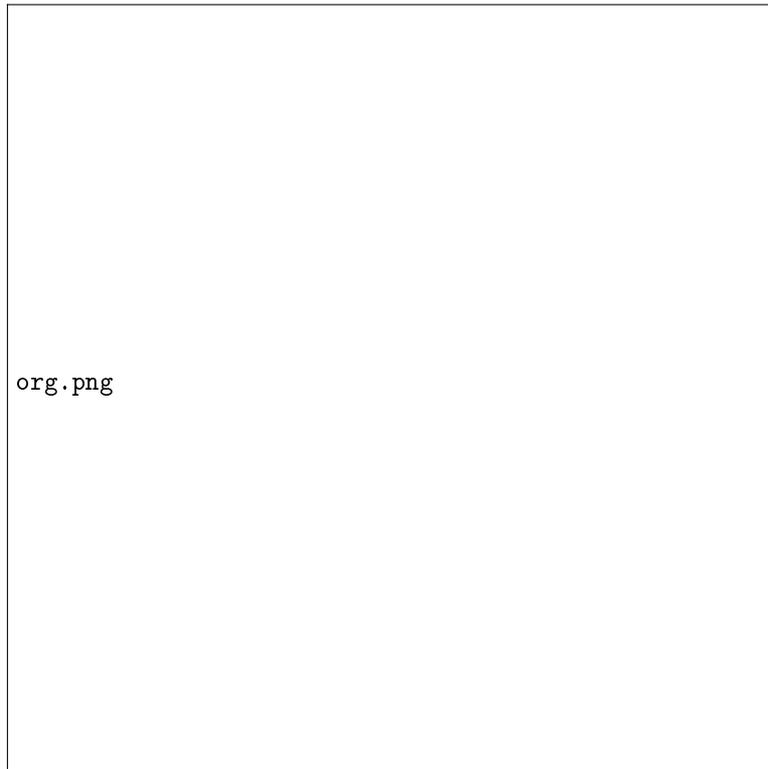


FIGURE 1 – Organigramme de la méthode de Dichotomie.

$$\begin{cases} f_3(x) = x^3 - 12x^2 + 1 \\ [a \ b] = [0 \ 1] \end{cases} \quad (3)$$

$$\begin{cases} f_4(x) = \cos(x) - x^3 \\ [a \ b] = [0 \ 1] \end{cases} \quad (4)$$

4 Travail à réaliser

Programmation de la méthode de Dichotomie :

Écrire les programmes en MATLAB permettant d'appliquer la méthode en question aux fonctions précédemment définies.

On prendra un test d'arrêt de la forme $|x_{n+1} - x_n| < \epsilon$ et on prendra soin de prévoir un compteur d'itérations qui permettra d'interrompre le traitement dès que N_{max} d'itérations sont effectuées sans que la précision ϵ ne soit atteinte. On pourra prendre par exemple $N_{max} = 100$.

- Les données seront a, b et ϵ, N_{max} et la fonction utilisée ; Les résultats seront la racine obtenue ainsi que son image par la fonction utilisée, le nombre d'itérations effectuées, et l'erreur de calcul. - Tester sur les fonctions f_1, f_2, f_2 et f_4 pour $\epsilon = 10e - 3, 10e - 6, 10e - 9$ et $10e - 12$.

5 Programme en Matlab

5.1 Boucle 'for ... end'

En prenant l'exemple de la solution d'une fonction $f(x) = x - e^{\sin(x)}$:

```
clc
```

```

clear all
x=1 :0.1 :10 ;
f=inline('x-exp(sin(x))');
plot(x,f(x)), grid
a=1 ; fa=f(a);
b=10 ; fb=f(b);
i=0 ; Nmax=20 ;
eps=1.0e-3 ;
if ((fa*fb)<0)
    for (i=0 :Nmax)
        x=(a+b)/2 ;
        i=i+1 ;
        if (sign(f(x)) == sign(fa))
            a=x ; fa=f(x) ;
        else
            b=x ; fb=f(x) ;
        end
        fprintf('dans l'ititation i=%d \t la solution est x0=f \t f(x0)= %f \n',i,x,f(x))
    end ;
    fprintf('La solution finale est x0 = %f \n',x)
else
    disp('On ne peut pas faire de Dichotomie dans cet intervalle!!')
end

```

5.2 Boucle 'while ... end'

On prend le même exemple $f(x) = x - e^{\sin(x)}$:

```

clc
clear all
x=1 :0.1 :10 ;
f=inline('x-exp(sin(x))');
plot(x,f(x)), grid
a=1 ; fa=f(a);
b=10 ; fb=f(b);
i=0 ; eps=1.0e-3 ;
if ((fa*fb)<0)
    while (b-a)>eps
        x=(a+b)/2 ;
        i=i+1 ;
        if (sign(f(x)) == sign(fa))

```

```

    a=x; fa=f(x);
else
    b=x; fb=f(x);
end
fprintf('dans l'ititation i=%d \t la solution est x0=f \t f(x0)= f \n',i,x,f(x))
end
fprintf('La solution finale est x0 = %f \n ',x)
else
    disp('On ne peut pas faire de Dichotomie dans cet intervalle!!')
end

```