Analyse Numérique - TP n° 1
Calcul des Zéros des Fonctions

Groupe C06

Christian Ingouff
Pierre-Alexandre Tyndal

EISTI

2013/2014
Semestre 2

Table des matières

Introduction

Le travail présenté dans ce rapport[2] traite du calcul de la valeur du minimum d'une fonction à une variable. Dans l'intérêt de notre étude, les fonctions manipulées dans le cadre de cette réalisation sont non linéaires et réelles.

Parmi les méthodes existantes, nous présenterons la méthode de Newton-Raphson pour le calcul des racines des fonctions. Ainsi, ce rapport présente le principe, les conditions, l'implémentation et le résultat de l'application d'une telle méthode. Ceci sera conclu par un récapitulatif critique de la méthode, présentant ses avantages et inconvénients.

Partie 1

Description mathématique : Méthode de Newton-Raphson

La méthode de Newton-Raphson[4] est utilisée pour calculer les racines d'une fonction non linéaire. Pour cela, elle se fonde sur le développement de Taylor au premier ordre.

Soit $f : D \to \mathbb{R}; D \subset \mathbb{R}, f \in C^2$. On cherche la racine $\xi \in D$ de la fonction f. Pour cela, on prend $x_0 \in D$, pas trop éloigné de ξ. La méthode consiste à obtenir $x_1 \in D$, qui devrait être plus proche de la racine, en résolvant l'équation suivante :

$$\begin{aligned} & f(x_0) + f'(x_0)(x_1 - x_0) = 0 \\ \Leftrightarrow \quad & x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}; f'(x_0) \neq 0 \end{aligned}$$

En répétant la résolution plusieurs fois, on peut ainsi approcher la racine ξ au fur et à mesure. Le problème peut alors se modéliser sous la forme suivante :

$$\begin{cases} x_0 \in D \\ x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}; n \in \mathbb{N}, \forall n \in \mathbb{N}, f'(x_n) \neq 0 \end{cases}$$

De cette manière, la logique de la méthode de Newton-Raphson dicterait que :

$$\lim_{n \to +\infty} x_n = \xi$$

Pour adapter la méthode à la minimisation d'une fonction, on utilise la propriété suivante :

$$x^* \text{ est un minimum} \Rightarrow f'(x^*) = 0$$

En récapitulatif, les conditions imposées à la méthode de Newton-Raphson sont les suivantes :

- $f \in C^2$
- $x_0 \in D$, pas trop éloigné de la racine ξ
- $\forall n \in \mathbb{N}, f'(x_n) \neq 0$

Partie 2

Réalisation informatique

2.1 Description

La réalisation informatique se concentre d'abord sur la recherche des racines d'une fonction. Pour cela, elle utilise la méthode de Newton-Raphson décrite précédemment.

Afin d'appliquer la méthode, il faut cependant aussi intégrer la notion de dérivées dans le programme. Nous proposons ainsi deux méthodes : la première utilise simplement la formule analytique (c'est-à-dire calculée au préalable), alors que la deuxième fera usage de l'approximation de Taylor au premier ordre :

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

Il s'agit désormais de répéter l'opération décrite précédemment pour pouvoir approcher la racine que l'on cherche. On sait que l'on approche d'une racine quand la différence entre les termes successifs tend vers 0, c'est-à-dire que l'on cherche n sous la condition suivante :

$$\forall \varepsilon > 0, \exists n \in \mathbb{N}, |x_{n+1} - x_n| < \varepsilon$$

Dans le cas où il n'y a pas de racines, la suite $(x_n)_{n\in\mathbb{N}}$ diverge. Il faut donc penser à mettre un nombre $n_{\max}$ d'itérations (nous prenons ici arbitrairement $n_{\max} = 10000$).

2.2 Algorithme

Par conséquent, en définissant une précision ε pour la condition d'arrêt, l'algorithme de la méthode Newton-Raphson pour le calcul de racines se présente ainsi :

```
Fonction NewtonRaphson(f : fonction, x0 : réel, eps : réel) : réel
// f : fonction à analyser
// x0 : x de départ
// eps : précision
var
   X : tableau de réels // représente la suite (xn)
   i : entier           // itération
   nmax : entier        // maximum d'itérations
Début
   X(0) <- x0
   i <- 0
   Tant que |X(i+1) - X(i)| > eps et i < nmax
      X[i+1] <- intersection(f, X(i))
      i <- i+1
   FinTantQue
   Si i < nmax alors
      Retourner X(i)
   Sinon
      Retourner "Aucune racine"
   FinSi
Fin
```

Pour le calcul du minimum d'une fonction, il suffit d'appliquer le calcul de l'intersection avec la dérivée de la fonction plutôt qu'avec la fonction elle-même.

Partie 3

Etude des résultats

Dans l'ensemble de l'étude, le programme[1] a été lancé avec les données suivantes :

- Précision : $\varepsilon = 0.000001$
- Approximation de Taylor : $h = 0.0001$
- Newton-Raphson : $n_{\max} = 10000$

3.1 Calcul de racines

3.1.1 Phase de calcul

Soient les fonctions réelles :

$$\begin{cases} f_1(x) = x^2 - a & a > 0 \\ f_2(x) = x^2 + 17 & (x_0 = 1) \\ f_3(x) = x - 2\sin(x) & (x_0 = 1.1) \end{cases}$$

Pour f_1, nous testerons pour les valeurs

$$\begin{cases} a = 2 \\ x_0 = 0.5 \end{cases}$$

ainsi que

$$\begin{cases} a = 42 \\ x_0 = 42 \end{cases}$$

Les résultats escomptés[3] sont les suivants :

$$\begin{cases} f_1, a = 2 & \pm\sqrt{2} \simeq \pm 1.41421356237 \\ f_1, a = 42 & \pm\sqrt{42} \simeq \pm 6.48074069841 \\ f_2 & \text{Pas de racines} \\ f_3 & 0, \pm 1.89549426703 \end{cases}$$

Ci-dessous se trouvent les tableaux des itérations de chaque calcul :

f_1 avec $a = 2, x_0 = 0.5$ (formule analytique)	
n	x_n
0	0.5000000000
1	2.2500000000
2	1.5694444444
3	1.4218904041
4	1.4142342902
5	1.4142135527
6	1.4142135151

f_1 avec $a = 2, x_0 = 0.5$ (approximation Taylor)	
n	x_n
0	0.5000000000
1	2.2498250175
2	1.5694066541
3	1.4218914694
4	1.4142345307
5	1.4142135940
6	1.4142135564

f_1 avec $a = 42, x_0 = 42$ (formule analytique)	
n	x_n
0	42.0000000000
1	21.5000000000
2	11.7267441860
3	7.6541505982
4	6.5706847589
5	6.4813562660
6	6.4807406937
7	6.4807406921

f_1 avec $a = 42, x_0 = 42$ (approximation Taylor)	
n	x_n
0	42.0000000000
1	21.5000244054
2	11.7267782296
3	7.6541798141
4	6.5706959860
5	6.4813571252
6	6.4807407576
7	6.4807406560

f_2 avec $x_0 = 1$ (formule analytique)	
n	x_n
0	1.0000000000
1	-8.0000000000
2	-2.9375000000
3	1.4248670213
4	-5.2530355365
5	-1.0084057445
6	7.9249442236
7	2.8899093666
8	-1.4963140112
9	4.9324688467
...	...

f_2 avec $x_0 = 1$ (approximation Taylor)	
n	x_n
0	1.0000000000
1	-7.9995500225
2	-2.9371836119
3	1.4254111812
4	-5.2502517864
5	-1.0061154978
6	7.9457213356
7	2.9031342819
8	-1.4762275043
9	5.0200263783
...	...

f_3 avec $x_0 = 1.1$ (formule analytique)	
n	x_n
0	1.1000000000
1	8.4529922615
2	5.2564136940
3	203.3835760907
4	17.9874660259
5	-67.5943030253
...	...
43	266148.3335503269
44	-1911692.7225781712
45	-1002233.9088852162
46	-666267.7999207005
47	2367090.7485990403
...	...
76	1.1855588632
77	3.8737537227
78	1.7789645934
79	1.9048070075
80	1.8955438144
81	1.8954942829
82	1.8954942499

f_3 avec $x_0 = 1.1$ (approximation Taylor)	
n	x_n
0	1.1000000000
1	8.4459380395
2	5.2389139065
3	1358.1506668365
4	13744.2343976542
5	9078.3409039940
...	...
108	-51471.9504141341
109	-103421.2473611150
110	-206882.9976709758
111	-135790.3976192583
112	-42537.5714124546
...	...
147	420.2291872214
148	1314.7551891458
149	-1.3574862063
150	-2.3933483933
151	-1.9745456696
152	-1.8987429629
153	-1.8955001225
154	-1.8954942892
155	-1.8954942562

3.1.2 Interprétation des résultats

Pour f_1, avec les deux couples de valeurs, les résultats coïncident avec un résultat escompté selon la précision donnée, que ce soit avec la formule analytique ou l'approximation de la dérivée. En effet, la racine obtenue est à chaque fois la plus proche du point x_0 initial. Les deux formules nous permettent d'obtenir les résultats en 7 ou 8 itérations toutes les deux.

On remarque cependant tout de suite la précision perdue avec l'approximation en comparant par exemple, pour le premier couple de valeurs :

$$n = 1 \Rightarrow x_{1_{\text{analytique}}} = 2.25 \text{ et } x_{1_{\text{Taylor}}} = 2.2498250175$$

Ceci dénotera une convergence plus directe avec la formule analytique qu'avec l'approximation.

f_2 présente un cas de non solution : par conséquent, la suite x_n diverge, ce qui est bien représenté par les données obtenues. Il n'y aura aucune combinaison successive de $(x_n, x_{n+1}$ dont la différence conviendra à la précision donnée, ce qui explique le résultat.

On obtient bel et bien une solution escomptée pour le cas de f_3. Cependant, le caractère de la fonction fait faire au programme des itérations qui éloignent beaucoup x d'une racine envisagée, expliquant la centaine d'itérations nécessaires afin d'aboutir au résultat.

En effet, l'utilisation de l'intersection de la tangente dans des fonctions incluant la fonction sin nuit à l'aboutissement de la méthode Newton-Raphson. La méthode veut parcourir une pente croissante ou décroissante afin d'arriver à un zéro, mais comme f_3 est successivement croissante et décroissante sur $\mathbb{R}$, l'allure alternante des tangentes amènera l'algorithme dans une fausse direction.

En première conclusion, la méthode de Newton-Raphson n'opérera bien que si la fonction est au moins localement monotone autour des racines à chercher, ce qui constitue une limite déjà contraignante dans l'univers des fonctions non linéaires. Elle s'avère toutefois relativement satisfaisante pour, par exemple, des fonctions polynômiales ou pseudo-polynômiales.

3.2 Calcul de minima

3.2.1 Phase de calcul

Soient les fonctions suivantes :

$$\begin{cases} f_4(x) = \sin(3x) + x^2 - 2 & x \in [-3, 3], x_0 = 1 \\ f_5(x) = x\cos(x) & x \in [-2\pi, 2\pi], x_0 = 0.1 \text{ et } x_0 = 2.5 \end{cases}$$

Les résultats escomptés[3] sont les suivants :

$$\begin{cases} f_4, x \in [-3, 3] & x_{\min} = -0.427307846 \\ f_5, x \in [-2\pi, 2\pi] & x_{\min} = -2\pi \simeq -6.283185307 \end{cases}$$

L'approximation de Taylor ne donne pas de résultats exploitables : elle ne s'approche pas assez pour f_4 et diverge trop pour f_5, probablement à cause d'erreurs trop grandes. Par conséquent, l'ensemble des résultats ci-dessous utilise la formule analytique de la dérivée pour les calculs.

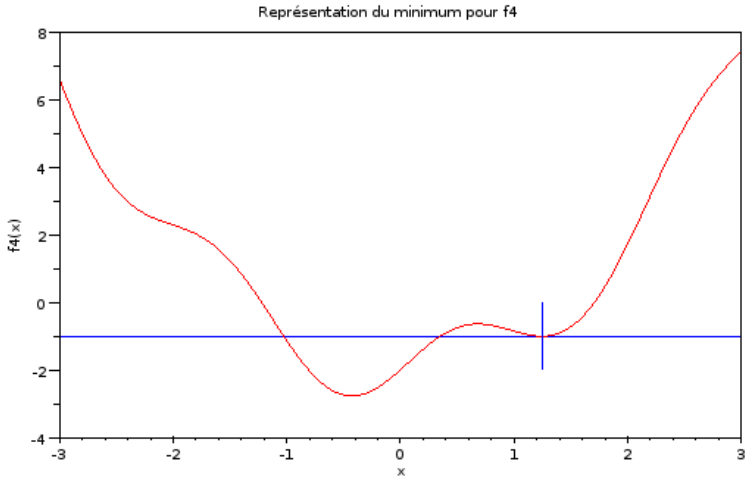

FIGURE 3.1 – f_4 : $x_{min} = 1.2445900098$

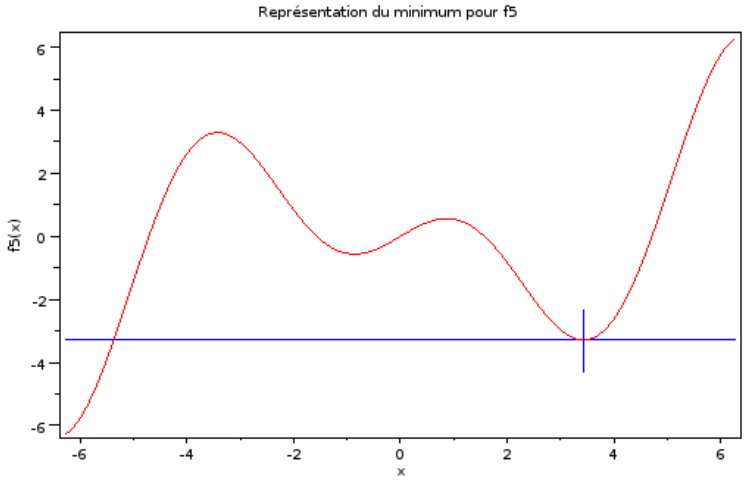

FIGURE 3.2 – $f_5, x_0 = 0.1 : x_{min} = 3.4256184171$

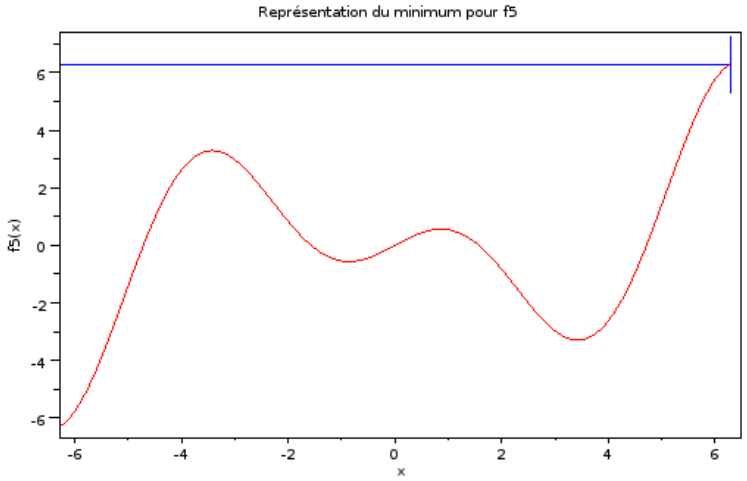

FIGURE 3.3 – $f_5, x_0 = 0.1 : x_{min} = 2\pi$ (collision avec la borne supérieure)

3.2.2 Interprétation des résultats

Les résultats obtenus avec la formule analytique correspondent à des extrema locaux, voisins à x_0. En effet, les résultats obtenus pour f_4 et $f_5, x_0 = 0.1$ correspondent à des minima locaux, mais non globaux. Le résultat obtenu avec $f_5, x_0 = 2.5$ correspond à un maximum local plutôt qu'à un minimum : le fait est que même si

$$x^* \text{ est un minimum} \Rightarrow f'(x^*) = 0$$

la réciproque n'est vraie qu'en partie, et inclut les maxima et les points selles dans l'implication.

Ceci constitue de premières limites qui restreignent beaucoup le potentiel de la méthode dans la recherche de minima globaux. On relève cependant que la méthode fonctionne pour chercher des extrema locaux, et il est possible d'implémenter une vérification de la positivité ou non de la dérivée seconde autour de l'extremum afin de différencier les minima des maxima.

L'échec de l'implémentation de la méthode avec l'approximation de Taylor peut être due à l'approximation induite par la première dérivation effectuée pour l'adapter à la recherche de minima. La première dérivation se fait car on cherche les zéros dans la fonction dérivée. Une hypothèse plausible est que l'approximation n'est pas localement monotone autour de la racine à chercher, ce qui déroute la méthode. Cette hypothèse expliquerait aussi pourquoi la méthode ne fonctionne pas quelque soit le h spécifié pour l'approximation de la dérivée.

Il faut donc connaitre exactement la dérivée première de la fonction à analyser, ce qui constitue une énorme limite à la méthode Newton-Raphson.

Conclusion

La méthode de Newton-Raphson a réussi à présenter des résultats cohérents sous une certaine mesure. Dans notre étude, nous avons notamment réussi à déterminer des racines pour des fonctions polynômiales ou pseudo-polynômiales.

La détection de l'absence de racines est aussi bien en place, comme nous l'a montré l'étude de f_2. Ceci démontre la validité de la méthode.

De plus, elle réussit à obtenir ces racines de façon rapide la plupart du temps, ce qui est intéressant dans le cadre de l'analyse numérique de fonctions non linéaires.

Cependant, comme il a été vu, cette méthode comporte de nombreuses limites :

- Elle ne fonctionne que pour la recherche de racines de fonctions dont la monotonie est certaine autour de la racine recherchée.
- Elle nécessite l'expression exacte de la dérivée de la fonction à analyser pour une utilisation optimale.
- Son utilisation pour la recherche d'extrema se limite à un cadre local

En conclusion, la méthode de Newton-Raphson comporte sa part de validité, mais ses limites l'empêchent d'être potentiellement utilisable dans un cadre plus poussé.

Références

[1] Ce programme a été conçu dans Scilab 5.4.1. Les graphiques proposés proviennent de la fonction "plot" de Scilab.
http ://www.scilab.org/

[2] Ce rapport a été rédigé en LaTeX
http ://www.latex-project.org/

[3] Il a été fait usage de WolframAlpha pour l'étude des fonctions et le calcul des résultats escomptés.
http ://www.wolframalpha.com/

[4] Les notions mathématiques introduites dans ce rapport proviennent de cours fournis à l'EISTI.
http ://arel.eisti.fr/
http ://sifoci.eisti.fr/

Annexes

Annexe A : Organigramme du programme

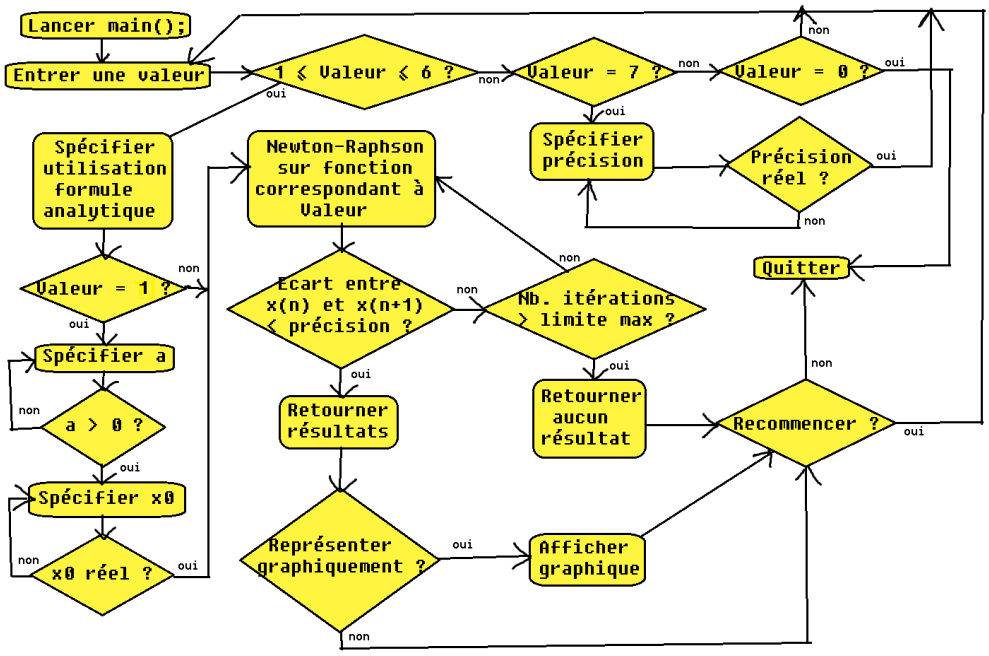

FIGURE 3.4 – Organigramme du programme Scilab

Annexe B : Guide d'utilisation du programme

Exécution du programme

Vous trouverez joint à ce rapport, dans le dossier `src`, le fichier `code.sce` qui peut être executé sous Scilab. Pour ceci, il faut accéder depuis la console Scilab au dossier contenant `code.sce` (par exemple : `cd ananu-tp1/src/;`), puis exécuter avec `exec("code.sce");`.

Le code ainsi executé, il suffit de lancer `main();` pour accéder au programme.

Utilisation du programme

Le programme ainsi lancé propose à l'utilisateur plusieurs options :

- Choisir une fonction pour la recherche de racines
  (choisir un entier de 1 à 3)
- Choisir une fonction pour la recherche de minima
  (choisir un entier de 4 à 6)
- Modifier la précision (choisir 7)
- Quitter le programme (choisir 0)

L'utilisateur choisit son option en entrant un entier au clavier et en validant avec `Entrée`.

La modification de la précision se fait de manière analogue : il sera demandé à l'utilisateur d'entrer un réel correspondant à la nouvelle précision et de valider avec `Entrée`.

Il est parfois demandé à l'utilisateur de répondre à des questions oui ou non, dénotées par `(o/n)`. L'utilisateur répond avec le clavier par un "o" pour oui ou par un "n" pour non et valide avec `Entrée`.

Utilisation de Newton-Raphson

Pour tous les calculs, il est demandé de spécifier si l'utilisateur veut utiliser la formule analytique pour les calculs (avec une question oui ou non). Le cas échéant, le calcul se fait avec l'approximation de Taylor.

Pour le calcul avec f_1, il est demandé de spécifier les valeurs de a et de x_0. Pour cela, quand la demande vient, l'utilisateur entre des réels avec le clavier et valide avec `Entrée`.

A l'issue de ces vérifications, le programme opère la recherche de racine ou de minimum et retourne une solution accompagnée du nombre d'itérations. Il fournit aussi la liste des valeurs de la suite x_n dans `src/log_iteration.txt`.

S'il y a une solution valide, le programme propose à l'utilisateur de l'afficher par rapport à la fonction dans un graphique (via une question oui ou non). La validation de cette demande affiche une graphique généré grâce à `plot2D` représentant la solution graphiquement.

A l'issue de l'ensemble de ces demandes, l'utilisateur sera invité à relancer le programme depuis le choix de la fonction. Si l'utilisateur refuse de recommencer, le programme se termine.