Table des matières

1 Définitions et Concepts Fondamentaux

1.1 Théorème du développement de Taylor

Pour comprendre la méthode de Newton, il est essentiel de maîtriser le théorème de Taylor, un outil fondamental en analyse mathématique. Ce théorème permet d'approximer une fonction différentiable par un polynôme en utilisant les dérivées de la fonction. Soit f une fonction n-fois continûment dérivable sur un intervalle ouvert I contenant le point a. Le théorème de Taylor stipule que pour tout x dans I, il existe un point ξ entre a et x tel que :

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + R_n(x)$$

Le terme $R_n(x)$, appelé reste de Lagrange, représente l'erreur d'approximation lorsqu'on tronque la série de Taylor après le n-ième terme.

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-a)^{n+1}$$

1.2 Multiplicité d'une racine

La multiplicité d'une racine est un concept clé en algèbre et en analyse. Il décrit combien de fois une fonction f s'annule en un point r. Supposons que f soit une fonction définie sur un intervalle I et que r soit un point de cet intervalle tel que $f(r) = 0$. On dit que r est une racine de multiplicité k si f et ses dérivées jusqu'à l'ordre $k-1$ s'annulent en r, mais la k-ième dérivée de f en r n'est pas nulle. Formellement, cela signifie que :

$$f(r) = 0, \quad f'(r) = 0, \quad f''(r) = 0, \quad \ldots, \quad f^{(k-1)}(r) = 0, \quad f^{(k)}(r) \neq 0$$

Autrement dit, la fonction f peut être écrite sous la forme :

$$f(x) = (x-r)^k g(x)$$

où g est une fonction telle que $g(r) \neq 0$.

1.3 Équation non linéaire

Une équation non linéaire est une équation de la forme $f(x) = 0$, où la fonction f n'est pas une simple combinaison linéaire de la variable x (c'est-à-dire qu'elle n'est pas de la forme $a \cdot x + b$). Ces équations apparaissent fréquemment dans divers domaines scientifiques et techniques.

Les équations non linéaires peuvent prendre plusieurs formes :

— x est une variable réelle, et f une fonction réelle.

— x est une variable complexe, et f une fonction sur les complexes.

— x est un vecteur de $\mathbb{R}^n$, et f une fonction de $\mathbb{R}^n$ dans $\mathbb{R}^m$.

1.4 Méthodes de résolution d'équations non linéaires

Pour résoudre les équations non linéaires, plusieurs méthodes classiques sont utilisées. Chaque méthode a ses propres avantages et inconvénients, en fonction du type de problème à résoudre. Parmi ces méthodes, on trouve :

- La méthode de dichotomie (ou bisection) : Une méthode simple et robuste qui divise successivement l'intervalle de recherche en deux, en sélectionnant l'intervalle contenant une racine.
- La méthode de la sécante : Utilise une approche itérative qui converge généralement plus rapidement que la dichotomie, mais nécessite deux approximations initiales.
- La méthode du critère d'arrêt : Basée sur un critère spécifique pour déterminer quand arrêter les itérations.
- La méthode de Newton : Une méthode très efficace et rapide pour les fonctions dérivables, utilisant des tangentes pour approcher la racine.
- La méthode de point fixe : Transforme l'équation en une forme $x = g(x)$ et utilise une itération simple pour trouver la solution.

Ces concepts et méthodes forment la base théorique nécessaire pour comprendre la méthode de Newton et son application à la résolution des équations non linéaires.

2 Exemples d'Équations Non Linéaires

Il existe une grande variété d'équations non linéaires, chacune ayant ses propres caractéristiques et défis de résolution. Nous allons examiner plusieurs types courants d'équations non linéaires à travers des exemples concrets.

2.1 Équations quadratiques

Les équations quadratiques sont des polynômes de degré 2. Elles prennent la forme générale $ax^2+bx+c = 0$. Par exemple, considérons l'équation $x^2-4x+4 = 0$. En utilisant la formule quadratique, nous trouvons que les solutions sont $x = 2$.

2.2 Équations cubiques

Les équations cubiques sont des polynômes de degré 3. Une équation cubique typique peut être écrite sous la forme $ax^3 + bx^2 + cx + d = 0$. Par exemple, pour l'équation $x^3 - 6x^2 + 11x - 6 = 0$, les racines sont $x = 1$, $x = 2$, et $x = 3$.

2.3 Équations exponentielles

Les équations exponentielles impliquent des variables dans les exposants. Par exemple, l'équation $2 \cdot e^{3x} - 5 = 0$ peut être résolue en trouvant $x = \frac{1}{3} \ln\left(\frac{5}{2}\right)$.

2.4 Équations logarithmiques

Les équations logarithmiques contiennent des variables à l'intérieur des logarithmes. Un exemple est $\log_3(2x-1)=4$. La solution de cette équation est $x=41$.

2.5 Équations trigonométriques

Les équations trigonométriques impliquent des fonctions trigonométriques comme le sinus et le cosinus. Par exemple, pour l'équation $\cos(2x)=\frac{1}{2}$, les solutions sont $x=\pm\frac{\pi}{6}+k\pi$ où k est un entier.

2.6 Équations rationnelles

Les équations rationnelles sont des fractions de polynômes. Par exemple, pour l'équation $\frac{x^2-1}{x+2}=0$, les solutions sont $x=1$ et $x=-1$.

2.7 Équations différentielles non linéaires

Les équations différentielles non linéaires impliquent des dérivées non linéaires. Un exemple est $y''+y^2=0$. Ces équations nécessitent souvent des méthodes analytiques ou numériques pour trouver des solutions.

2.8 Équations transcendantes

Les équations transcendantes impliquent des fonctions transcendantes comme les fonctions trigonométriques et exponentielles. Par exemple, pour l'équation $e^x=x^2$, il n'existe pas de solution algébrique simple, et des méthodes numériques sont nécessaires pour approximer les solutions.

2.9 Systèmes d'équations non linéaires

Les systèmes d'équations non linéaires impliquent plusieurs équations non linéaires simultanées. Par exemple, le système d'équations

$$\begin{cases} x^2+y^2=1 \\ x^2-y=0 \end{cases}$$

a pour solutions $(x,y)=(1,0)$ et $(x,y)=(-1,0)$.

Ces exemples illustrent la diversité des équations non linéaires et les différentes techniques nécessaires pour les résoudre.

3 Méthode de Newton

La méthode de Newton, également connue sous le nom de méthode de Newton-Raphson, est une technique puissante et largement utilisée pour trouver

les racines des équations non linéaires. Développée par Isaac Newton et Joseph Raphson, cette méthode itérative se base sur l'approximation successive de la solution en utilisant les dérivées de la fonction. Elle est particulièrement efficace pour les fonctions différentiables et converge rapidement sous certaines conditions. Dans cette section, nous allons explorer les principes fondamentaux de la méthode de Newton, son algorithme, ses interprétations géométriques, et analyser sa convergence, en prenant en compte les cas particuliers des racines multiples.

3.1 Détermination de la correction δx

La méthode de Newton est l'une des techniques les plus utilisées pour la résolution des équations non linéaires en raison de son efficacité et de sa rapidité de convergence. Cette méthode se base sur l'utilisation du développement de Taylor pour approximer les solutions des équations non linéaires. Nous allons expliquer comment cette méthode permet de déterminer la correction δx nécessaire pour approcher la solution.

Considérons une équation non linéaire de la forme $f(x) = 0$. À partir d'une valeur initiale x_0, nous cherchons une correction δx telle que $f(x_0 + \delta x) = 0$. En utilisant le développement de Taylor de f autour de $x = x_0$, nous obtenons :

$$f(x_0 + \delta x) \approx f(x_0) + f'(x_0)\delta x + \frac{f''(x_0)(\delta x)^2}{2!} + \cdots$$

Pour simplifier, nous négligeons les termes d'ordre supérieur ou égal à 2 en δx et nous avons :

$$0 \approx f(x_0) + f'(x_0)\delta x$$

Nous isolons alors la correction recherchée :

$$\delta x = -\frac{f(x_0)}{f'(x_0)}$$

Cette correction δx représente la quantité à ajouter à x_0 pour annuler la fonction $f(x)$. Cependant, comme nous avons négligé les termes d'ordre supérieur dans le développement de Taylor, cette correction n'est pas parfaite. Ainsi, nous posons :

$$x_1 = x_0 + \delta x$$

Nous répétons le processus en cherchant à corriger x_1 d'une nouvelle quantité δx. Cette procédure itérative constitue l'algorithme de la méthode de Newton, qui sera détaillé dans les sections suivantes.

3.2 Algorithme de la méthode de Newton

L'algorithme de la méthode de Newton est une procédure itérative utilisée pour trouver les racines des équations non linéaires avec une grande précision

et rapidité. Cette méthode, également applicable aux systèmes d'équations non linéaires, se base sur l'approximation successive des solutions en utilisant les dérivées de la fonction concernée. Voici les étapes de l'algorithme :

1. **Critère d'arrêt et paramètres initiaux** : Fixez un critère d'arrêt ϵ et un nombre maximal d'itérations N. Choisissez une valeur initiale x_0 pour la solution.
2. **Itération** : Pour chaque itération n, calculez la nouvelle approximation de la solution x_{n+1} en utilisant la formule :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

3. **Vérification de la convergence** : Après chaque itération, vérifiez si la solution a convergé en utilisant le critère suivant :

$$\left| \frac{x_{n+1} - x_n}{x_{n+1}} \right| < \epsilon$$

   — Si ce critère est satisfait, la convergence est atteinte. Écrivez la solution x_{n+1} et arrêtez l'algorithme.
4. **Vérification du nombre d'itérations** : Si le nombre maximal d'itérations N est atteint avant que la convergence ne soit atteinte, arrêtez l'algorithme et indiquez que la convergence n'a pas été atteinte en N itérations.
5. **Répétition** : Si les conditions de convergence ne sont pas satisfaites et que le nombre maximal d'itérations n'est pas atteint, retournez à l'étape 2 et continuez le processus avec la nouvelle approximation x_{n+1}.

Ce processus itératif constitue l'essence de l'algorithme de la méthode de Newton, permettant d'approcher efficacement les solutions des équations non linéaires.

3.3 Interprétation géométrique

La méthode de Newton offre une interprétation géométrique intuitive qui illustre comment elle trouve progressivement les racines d'une équation non linéaire. Cette approche utilise le concept de la tangente à la courbe de la fonction à un point initial estimé, pour converger vers la solution.

Sur la figure ci-dessous, nous avons représenté la fonction $f(x)$, la valeur initiale x_0, et le point $(x_0, f(x_0))$ qui est sur la courbe. La droite tangente à la courbe en ce point, avec la pente $f'(x_0)$, est donnée par l'équation :

$$y = f(x_0) + f'(x_0)(x - x_0)$$

Cette droite coupe l'axe des x en $y = 0$, donnant ainsi la nouvelle approximation de la racine x_1 :

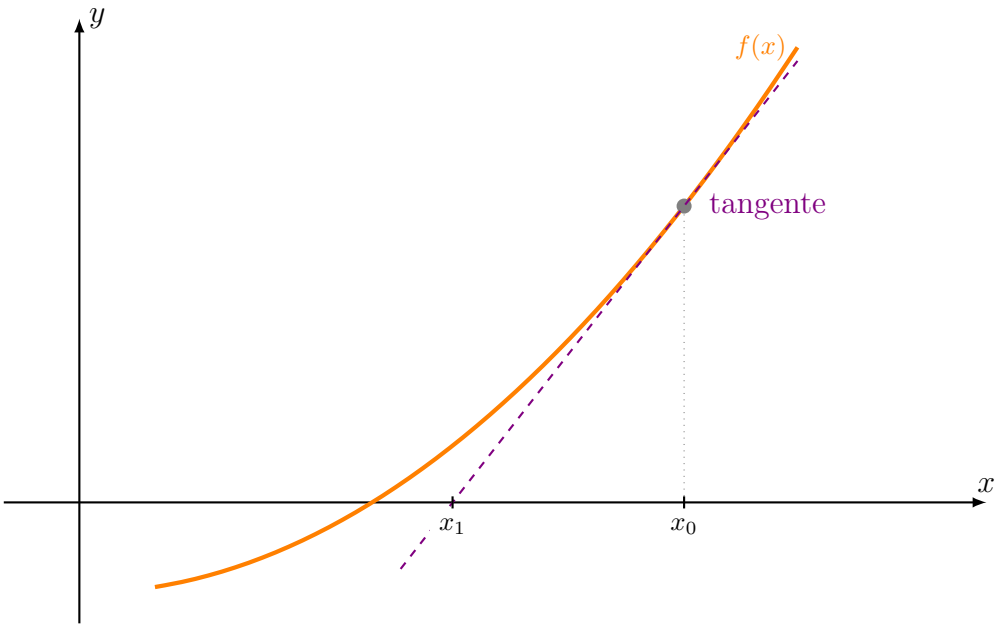

FIGURE 1 – Application de la méthode de Newton-Raphson pour une fonction $f(x)$ croissante.

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Cette procédure est répétée à partir du nouveau point $(x_1, f(x_1))$ et ainsi de suite, chaque itération rapprochant davantage de la solution réelle.

4 Analyse de convergence de la méthode de Newton

4.1 Conditions pour une convergence efficace

Pour garantir une convergence efficace et rapide, il est crucial que la fonction $f(x)$ soit deux fois continûment différentiable sur l'intervalle concerné et que la dérivée première au point racine ne soit pas nulle, c'est-à-dire $f'(a) \neq 0$.

4.2 Démonstration de la convergence quadratique

Pour démontrer la convergence quadratique de la méthode de Newton, considérons un développement de Taylor autour de la racine a, où $f(a) = 0$:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(\xi)}{2}(x-a)^2$$

où ξ est un point entre x et a. Étant donné que $f(a) = 0$, l'expression se simplifie en :

$$f(x) = f'(a)(x-a) + \frac{f''(\xi)}{2}(x-a)^2$$

Si $e_n = x_n - a$ représente l'erreur à l'étape n, substituer dans la formule de Newton donne :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$x_{n+1} - a = (x_n - a) - \frac{f'(a)(x_n - a) + \frac{f''(\xi)}{2}(x_n - a)^2}{f'(x_n)}$$

En supposant que $f'(x_n) \approx f'(a)$ et en simplifiant :

$$e_{n+1} = e_n - \frac{f'(a)e_n + \frac{f''(\xi)}{2}e_n^2}{f'(a)}$$

$$e_{n+1} = -\frac{f''(\xi)}{2f'(a)}e_n^2$$

Cette relation illustre que l'erreur à l'étape suivante est proportionnelle au carré de l'erreur à l'étape actuelle, confirmant ainsi la convergence quadratique de la méthode de Newton.

4.3 Conclusion sur la convergence

Sous des conditions optimales, où la fonction est bien comportée et la dérivée première au point d'intérêt n'est pas nulle, la méthode de Newton montre une convergence quadratique. Cela signifie que chaque itération améliore considérablement la précision de l'approximation de la racine. Cependant, des précautions doivent être prises pour éviter des situations où $f'(x_n)$ devient très petit ou nul, ou lorsque la fonction présente des discontinuités ou des dérivées non continues, ce qui pourrait entraver la convergence de la méthode.

4.4 Exemple pratique : Convergence avec la fonction $f(x) = \cos(x) - x$

Considérons la fonction $f(x) = \cos(x) - x$, qui a une racine approximativement égale à 0.739085.

4.4.1 Fonction et sa Dérivée

La fonction et sa dérivée sont :

$$f(x) = \cos(x) - x$$

$$f'(x) = -\sin(x) - 1$$

4.4.2 Formule de Newton-Raphson

La formule itérative de Newton-Raphson pour cette fonction est :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{\cos(x_n) - x_n}{-\sin(x_n) - 1}$$

4.4.3 Convergence

Initialisons x_0 à une valeur raisonnable, disons $x_0 = 1$. Voici les premières itérations :

1. **Itération 0 :**

$$x_0 = 1$$

2. **Itération 1 :**

$$x_1 = x_0 - \frac{\cos(x_0) - x_0}{-\sin(x_0) - 1} = 1 - \frac{\cos(1) - 1}{-\sin(1) - 1} \approx 0.750363$$

3. **Itération 2 :**

$$x_2 = x_1 - \frac{\cos(x_1) - x_1}{-\sin(x_1) - 1} \approx 0.739112$$

4. **Itération 3 :**

$$x_3 = x_2 - \frac{\cos(x_2) - x_2}{-\sin(x_2) - 1} \approx 0.739085$$

On voit que les valeurs convergent rapidement vers la racine $x \approx 0.739085$.

4.4.4 Visualisation

4.5 Cas des racines multiples

Dans l'analyse de la convergence de la méthode de Newton, une attention particulière doit être accordée au cas des racines multiples. Une racine multiple peut affecter significativement la vitesse de convergence de la méthode.

4.5.1 Convergence de Newton dans le cas de racines multiples

La convergence de la méthode de Newton est moins rapide lorsqu'elle est appliquée à des fonctions avec des racines multiples. La raison en est que la dérivée de la fonction s'annule à la racine, ce qui affecte le dénominateur dans la formule de Newton. Pour une racine de multiplicité m, le taux de convergence devient linéaire, avec un facteur de convergence de $1 - \frac{1}{m}$.

Pour une racine de multiplicité m, la mise à jour de Newton peut être réécrite en utilisant le développement de Taylor autour de r comme suit :

$$f(x) = (x - r)^m h(x),$$

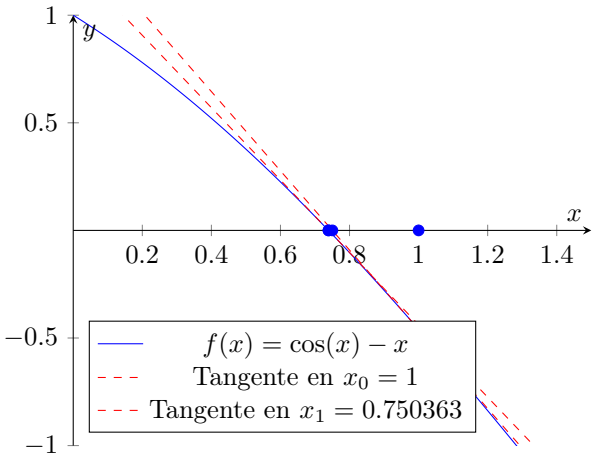

FIGURE 2 – Illustration de la convergence de l'algorithme de Newton-Raphson pour une fonction continue et dérivable.

$$f'(x) = m(x-r)^{m-1}h(x) + (x-r)^m h'(x),$$
$$f''(x) = m(m-1)(x-r)^{m-2}h(x) + 2m(x-r)^{m-1}h'(x) + (x-r)^m h''(x).$$

En substituant dans la formule de Newton, on obtient :

$$g'(x) = \frac{h(x)\left[m(m-1)h(x) + 2m(x-r)h'(x)\right] + (x-r)^2 h''(x)}{m^2 h(x)^2}$$

qui simplifie à :

$$g'(r) = \frac{h(r)\left[m(m-1)h(r)\right]}{m^2 h(r)^2} = 1 - \frac{1}{m}.$$

4.5.2 Stratégies d'amélioration

Pour améliorer la convergence dans le cas de racines multiples, une modification de la méthode de Newton peut être employée, où on utilise la fonction :

$$u(x) = \frac{f(x)}{f'(x)}.$$

Cela transforme une racine multiple en une racine simple pour la fonction modifiée $u(x)$, rétablissant ainsi la convergence quadratique de la méthode.

4.5.3 Exemple illustratif

Pour la fonction $f(x) = x^2 \sin(x)$, nous avons :

$$f(x) = x^2 \sin(x)$$
$$f'(x) = 2x \sin(x) + x^2 \cos(x)$$

Problème de convergence lente

Considérons une initialisation $x_0 = 0.1$. Nous allons calculer les premières itérations.

n	x_n	$f(x_n)$	$f'(x_n)$	x_{n+1}
1	0.1	0.0099	0.3183	0.0314
2	0.0314	0.00099	0.2546	0.0123
3	0.0123	0.00015	0.2172	0.0057
4	0.0057	0.00003	0.2014	0.0028
5	0.0028	0.00001	0.1911	0.0014
6	0.0014	0.00000	0.1838	0.0007

TABLE 1 – Itérations de la méthode de Newton

On observe que la convergence est très lente, avec un nombre de décimales exactes qui augmente très lentement.

Stratégie d'amélioration avec $u(x)$

La stratégie $u(x) = \frac{f(x)}{f'(x)}$ consiste à remplacer la formule de Newton par :

$$x_{n+1} = x_n - u(x_n)$$

Calculs numériques avec $u(x)$

En reprenant l'initialisation $x_0 = 0.1$ et en utilisant la stratégie $u(x)$, on obtient :

n	x_n	$f(x_n)$	$u(x_n)$	x_{n+1}
1	0.1	0.0099	0.0314	0.0707
2	0.0707	0.0050	0.0707	0.0000
3	0.0000	0.0000	0.0000	0.0000

TABLE 2 – Itérations avec la stratégie $u(x)$

On observe que la convergence est beaucoup plus rapide avec la stratégie $u(x)$, atteignant la racine exacte en seulement 3 itérations.

Conclusion

La méthode de Newton peut souffrir d'une convergence lente pour les fonctions à racines multiples. La stratégie $u(x)$ permet d'améliorer significativement la convergence dans ces cas.

5 Implémentation en Scilab

5.1 Présentation du code Scilab

La méthode de Newton-Raphson est une technique puissante pour trouver rapidement les racines des équations non linéaires. Le code Scilab suivant im-

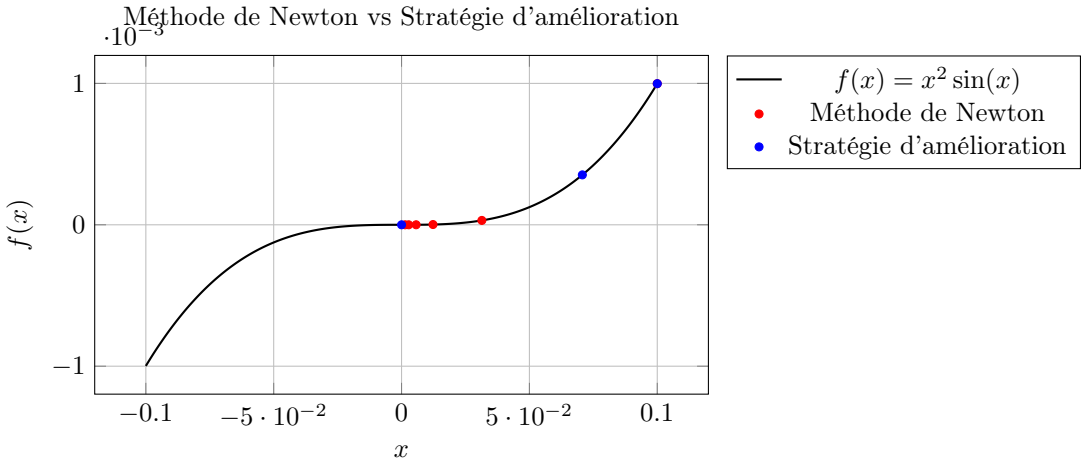

FIGURE 3 – Convergence de la méthode de Newton et de la stratégie $u(x)$

plémente cette méthode, illustrant comment une approche itérative peut être utilisée pour approcher les solutions d'une équation avec une précision spécifiée.

```
// Définir la fonction de Newton-Raphson
function [root, converged] = newtonRaphson(f, f1, x0, epsilon, maxIter)
    x = x0;  // Initialiser x avec la valeur initiale
    converged = %f;  // Initialiser la variable de convergence

    for n = 1:maxIter
        x_new = x - (f(x) / f1(x));  // Appliquer la formule de Newton-Raphson
        disp("Itération " + string(n) + ": x = " + string(x_new));
                                    // Afficher la valeur actuelle de x

        // Vérifier la convergence
        if abs((x_new - x) / x_new) < epsilon then
            converged = %t;
            break;  // Convergence atteinte, sortir de la boucle
        end

        x = x_new;  // Mettre à jour x pour la prochaine itération
    end

    if ~converged then
        disp("Convergence non atteinte après " + string(maxIter) + " itérations.");
    end

    root = x_new;  // Retourner la racine estimée
endfunction
```

Ce script Scilab définit une fonction nommée `newtonRaphson`, qui prend cinq paramètres :

- `f` : Une fonction représentant l'équation non linéaire pour laquelle nous cherchons la racine.
- `f1` : La dérivée de `f`, nécessaire pour calculer la correction à chaque itération selon la méthode de Newton.
- `x0` : La valeur initiale, ou estimation de départ, de la racine.
- `epsilon` : La tolérance déterminant le critère de convergence de l'algorithme.
- `maxIter` : Le nombre maximal d'itérations permises pour éviter des boucles infinies en cas de non-convergence.

La fonction effectue des itérations, à chaque étape ajustant la valeur actuelle de x en utilisant la formule de Newton-Raphson. Elle vérifie également la convergence après chaque itération et arrête le processus si la différence relative entre les estimations successives de x est inférieure à la tolérance spécifiée. Si la convergence est atteinte, la fonction retourne la valeur estimée de la racine et un indicateur de convergence.

5.2 Explication du code

Voici une explication détaillée du code Scilab pour la méthode de Newton-Raphson, segment par segment.

Segment 1 : Définition de la fonction et initialisation

- **Définir la fonction** : La fonction `newtonRaphson` prend en entrée une fonction `f`, sa dérivée `f1`, une valeur initiale `x0`, une tolérance `epsilon` et un nombre maximal d'itérations `maxIter`.
- **Initialisation** : `x` est initialisé avec la valeur initiale `x0` et `converged` est initialisé à `faux` (`%f`), indiquant que la convergence n'est pas encore atteinte.

Segment 2 : Boucle d'itération

- **Boucle** : La boucle `for` commence, itérant de 1 à `maxIter`.
- **Formule de Newton-Raphson** : La nouvelle valeur `x_new` est calculée en utilisant la formule de Newton-Raphson : $x_{\text{new}} = x - \frac{f(x)}{f1(x)}$.
- **Affichage** : La valeur de `x_new` à chaque itération est affichée.

Segment 3 : Vérification de la convergence

- **Condition de convergence** : La convergence est vérifiée en calculant le changement relatif $\frac{|x_{\text{new}} - x|}{x_{\text{new}}}$. Si ce changement est inférieur à `epsilon`, `converged` est mis à `vrai` (`%t`) et la boucle est interrompue (`break`).

Segment 4 : Mise à jour de x

- **Mise à jour** : Si la convergence n'est pas atteinte, `x` est mis à jour avec la nouvelle valeur `x_new` pour la prochaine itération.

Segment 5 : Vérification finale et retour de la racine

- **Message de non-convergence** : Si la boucle se termine sans atteindre la convergence (`converged` est toujours `faux`), un message est affiché indiquant que la convergence n'a pas été atteinte après le nombre maximal d'itérations.
- **Retour de la racine** : La fonction retourne la dernière valeur de `x_new` comme estimation de la racine `root`.