Université Mohammed V - Agdal
Faculté des Sciences
Département de Mathématiques et Informatique
Avenue Ibn Batouta, B.P. 1014
Rabat, Maroc

Filière :
Sciences Mathématiques et Informatique (SMI)
et
Sciences Mathématiques (SM)

Module Analyse Numérique I :

Par

Année 2003-2004

Plan du cours

Introduction

Représentation des nombres en machine

Résolution de f(x)=0
- **Méthodes directes**
- **Méthodes itératives**

Résolution de systèmes linéaires
- **Par la méthode de Gauss**
- **Par la décomposition LU**

Interpolation polynomiale

Intégration - Dérivation

Equations différentielles

Chapitre I :

Représentation des nombres en machine

0.1

Arithmétique des calculateurs et Sources d'erreurs

Si sophistiqué qu'il soit , un calculateur ne peut fournir que des réponses approximatives. Les approximations utilisées dépendent à la fois des contraintes physiques (espace mémoire, vitesse de l'horloge...) et du choix des méthodes retenues par le concepteur du programme . (pour plus de détails sur le fonctionnement d'un ordinateur et la terminologie de base voir par exemple la page web htttp://www.commentcamarche.com

Le but de ce chapitre est de prendre connaissance de l'impact de ces contraintes et de ces choix méthologiques. Dans certains cas il doit être pris en compte dans l'analyse des résultats dont une utilisation erronée pourrait être coûteuse.

La première contrainte est que le système numérique de l'ordinateur est discret, c'est à dire qu'il ne comporte qu'un nombre fini de nombres; Il en découle que tous les calculs sont entachés d'erreurs.

0.1.1 Evaluation de l'erreur

Rappelons d'abord quelques notion de base ;

Si X est une quantité à calculer et X^* la valeur calculée, on dit que :

1. $X - X^*$ est l'erreur et $\mid E \mid = \mid X - X^* \mid$est l'erreur absolue.

   Exemple :

   Si $X = 2.224$ *et* $X^* = 2.223$ *alors l'erreur absolue* $\mid E \mid = \mid X - X^* \mid = 2.224 - 2.223 = 0.001$

2. $E_r = \left|\frac{X-X^*}{X_r}\right|$ est l'erreur relative, $X_r \neq 0$. X_r est une valeur de référence pour X. En général ,on prend $X_r = X$.

   Exemple :

   Si $X = 2.224$ *et* $X^* = 2.223$ *alors , si on prend* $X_r = X$ *, l'erreur relative* $E_r = \left|\frac{X-X^*}{X_r}\right| = \frac{|X-X^*|}{|X|} = \frac{0.001}{2.224} = 4.496 \times 10^{-4}$

   Cependant, si X est la valeur d'une fonction $F(t)$ avec $a \leq t \leq b$, on choisira parfois une valeur de référence globale pour toutes les valeurs de t.

   Exemple :

   Si $X = \sin(t)$ *avec* $0 \leq t \leq \frac{\pi}{4}$, *on pourra prendre* $X = \frac{\sqrt{2}}{2} = \sup_{0 \leq t \leq \frac{\pi}{4}} \sin(t)$.

En général , on ne connait pas le signe de l'erreur de sorte que l'on considère les erreurs absolues et les erreurs relatives absolues.

Les opérations élémentaires propagent des erreurs.

Dans la pratique, on considère que :

1) L'erreur absolue sur une somme est la somme des erreurs absolues.

2) L'erreur relative sur un produit ou un quotient est la somme des ereurs relatives.

On peut estimer l'effet d'une erreur E sur l'argument x d'une fonction $f(x)$ au moyen de la dérivée de $f(x)$. En effet $f(x+E) \simeq f(x) + Ef'(x)$

Exemple :

Calculer la valeur de $(11111111)^2$

La valeur fournie par une petite calculatrice à cinq chiffres est $1,2345x10^{14}$

Mais la réponse exacte est 123456787654321.

La machine a donc tronqué le résultat à 5 chiffres et l'erreur absolue est de $6 * 19^9$.

L'erreur relative est de 0.0005% .

Cet exemple montre qu'il faut établir clairement l'objectif visé.

Cet objectif est double ;

1) Nous voulons un bon ordre de grandeur (ici 10^{14}) et avoir le maximum de décimales exactes,

2) Ce maximum ne peut excéder la longueur des mots permis par la machine et dépend donc de la machine

0.1.2 La mémoire de l'ordinateur : le stockage des nombres

La mémoire d'un ordinateur est formée d'un certain nombre d'unités adessables appelées OCTETS . Un ordinateur moderne contient des millions voir des milliards d'octets. Les nombres sont stockés dans un ordinateur comme ENTIERS ou REELS.

Les nombres entiers :

Les nombres entiers sont ceux que l'on utilise d'habitude sauf que le plus grand nombre représentable dépend du nombre d'octets utilisés:

-avec deux (2) octets, on peut représenter les entiers compris entre

$$-32768 \text{ et } 32767$$

-avec quatre (4) octets on peut représenterr les entiers compris entre

$$-2147483648 \text{ et } 2147483647$$

Les nombres réels

Dans la mémoire d'un ordinateur, les nombres réels sont représentés en notation flottante.

Cette notation a été introduite pour garder une erreur relative à peu prés constante; quelque soit l'ordre de gandeur du nombre qu'on manipule.

En notation flottante, un nombre a la forme:

$$x = \pm Y \times b^e$$

b est la base du système numérique utilisé

Y est la mantisse : une suite de s entier $y_1 y_2 ... y_s$ avec $y_1 \neq 0$ si $x \neq 0$ et $0 \leq y_i \leq (b-1)$

e est l'exposant(un nombre entier relatif)

La norme choisie est celle où la mantisse est comprise entre 0 et 1 et où le premier chiffre après la virgule est différent de zéro.

Calcul de l'erreur

Nous terminons ce chapitre en définissant les notions de troncature et d'arrondie.

Exemple :

En base 10, $x = 1/15 = 0.066666666.....$

Dans le cas d'une représentation tronquée nous aurons, pour $s = 5$, $fl(x) = 0.66666 * 10^{-1}$.

Remarquez comment nous avons modifié l'exposant afin de respecter la règle qui veut que le premier chiffre de la mantisse ne soit pas nul .

Dans ce cas, l'erreur absolue $X - fl(X)$ est de $6 \times 10^{-7\cdot}$. L'erreur relative est de l'ordre de 10^{-5}

Dans une représentation tronquée à s chiffres, l'erreur relative maximale est de l'ordre de 10^{-s}

Dans une représentation arrondie, lorsque la première décimale négligée est supérieure à 5, on ajoute 1 à la dernière décimale conservée.

Exemple :

x = 1/15 = 0.066666666.
Nous écrirons $fl(x) = 0.66667 \times 10^{-1}$
L'erreur absolue serait alors 3.333×10^{-7} *et l'erreur relative serait* 5×10^{-6}

En général, l'erreur relative dans une représentation arrondie à s chiffres est de $5 \times 10^{-(s+1)}$ soit la moitié de celle d'une représentation tronquée.

0.2 Les régles de base du modèle

Pour effectuer une opération sur deux nombres réels, on effectue l'opération sur leurs représentations flottantes et on prend ensuite la représentation flottante du résultat.

l'addition flottante

$$x \oplus y = fl(fl(x) + fl(y))$$

la soustraction flottante

$$x \ominus y = fl((x) - fl(y))$$

la multiplication flottante

$$x \otimes y = fl(fl(x) \times fl(y))$$

la division flottante

$$x \div y = fl(fl(x)/fl(y))$$

Chaque opération intermédiaire dans un calcul introduit une nouvelle erreur d'arrrondi ou de troncature.

Dans la pratique, il faudra se souvenir du fait que deux expressions algébriquement équivalentes peuvent fournir des résultats différents et que l'ordre des opérations peut changer les résultats.

Pour l'addition et la soustraction on ne peut effectuer ces 2 opérations que si les exposants sont les mêmes. On transforme le plus petit exposant et

donc on ne respecte plus la régle voulant que le premier chiffre de la mantisse ne soit pas nul.

Quelques remarques sur ce modèle:

On constate une déviation importante par rapport aux lois habituelles de l'arithmétique.

$x + (y + z)$ peut être différent de $(x + y) + z$.

Exemple :

Pour 4 chiffres significatifs ($s = 4$) on a :

$$(1 + 0.0005) + 0.0005 = 1.000$$

car

$$0.1 \times 10^1 + 0.5. \times 10^{-3} = 0.1. \times 10^1 + 0.00005. \times 10^1 = 0.1 \times 10^1 + 0.0000. \times 10^1 = 0.1 \times 10^1$$

et

$$1 + (0.0005 + 0.0005) = 1.001$$

Ainsi, l'addition flottante n'est pas associative .(TD:Sommation d'une série à termes positifs)

On constate aussi que si y est trés petit par rapport à x, l'addition de x et y donnera seulement x.

Exemple :

L'équation $1 + x = x$ *a* $x = 0$ *comme unique solution. Mais dans un système à* 10 *chiffres significatifs, elle aura une infinité de solutions* (*il suffit de prendre* $\mid x \mid \prec 5 \times 10^{-11}$)

La distributivité de la multiplication par rapport à l'addition.

Exemple :

Considérons l'opération

$$122 \times (333 + 695) = (122 \times 333) + (122 \times 695) = 125416$$

Si nous effectuons ces deux calculs en arithmétique à 3 chiffres ($s = 3$) et arrondi, nous obtenons:

$$122 \times (333 + 695) = fl(122) \times fl(1028)$$
$$= 122 \times 103 \times 10^1 = fl(125660) = 126 \times 10^3$$
$$(122 \times 333) + (122 \times 695) = fl(40626) + fl(84790)$$
$$406 \times 10^2 + 848 \times 10^2 = fl(406 + 848) \times 10^2 = fl(1254 \times 10^2) = 125 \times 10^3$$

Donc la distributivité de la multiplication par rapport à l'addition n'est pas respectée en arithmétique flottante.

0.3 Propagation des erreurs.

Une étude de la propagation des erreurs d'arrondi permattra d'expliquer ce phénomène.

Soit à calculer e^x à l'aide de son développement en série qui est convergent pour tout x :

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + ...$$

Il est evident que dans la pratique il est impossible d'effectuer la sommation d'une infinité de termes. On arrêtera donc lorsque le terme général $\frac{x^k}{k!}$ devient inférieur à 10^{-t}(on a t digits). Pour x négatif on sait que le reste de la serie est inférieur au premier terme négligé donc à 10^{-t}(puisque la serie est altérnée).

Les calculs suivant sont fait sur ordinateur pour $t = 14$.

x	e^x	S
-10	$4.54.10^{-5}$	$4.54.10^{-5}$
-15	$3.06.10^{-7}$	$3.05.10^{-7}$
-20	$2.06.10^{-9}$	$-1.55.10^{-7}$
-25	$1.39.10^{-11}$	$1.87.10^{-5}$
-30	$9.36.10^{-14}$	$6.25.10^{-4}$

On voit que pour $x \leq 20$ les résultats obtenus sont dépourvus de sens. L'explication de ce phénomène est la suivante: pour $x = -30$ les termes de la serie vont en croissant jusqu'à $\frac{x^{30}}{30!} = 8.10^{11}$ puis ils décroissent et $\frac{x^{107}}{107!} \tilde{} - 9.19.10^{-15}$.

L'erreur absolue sur le terme maximal est de $8.10^{11}.10^{-15} = 8.10^{-4}$. Ainsi le résultat obtenu pour S représente uniquement l'accumulation des erreurs d'arrondi sur les termes de plus grand module de développement en serie.

La propagation des erreurs est un des principaux problèmes en calcul numérique.

Considérons le cas d'une somme :

Dans l'addition, les erreurs absolues s'additionnent. Soit en effet ϵ_1 et $\epsilon_2$les erreurs absolues sur x_1 et x_2 .

On peut écrire:

$$(x_1 \pm \epsilon_1) + (x_2 \pm \epsilon_2) = (x_1 + x_2) \pm (\epsilon_1 + \epsilon_2)$$

En arithmétique flottante, l'erreur relative δ est à peu près constante et les erreurs absolues peuvent être approximativement explicités par :

$$\epsilon_1 =\mid x_1 \mid \times\delta, \epsilon_2 =\mid x_2 \mid \times\delta$$

Si les nombres en présence ont le même signe, l'erreur relative reste la même que celle qu'on avait x_1 et x_2. En effet

$$\frac{\epsilon_1 + \epsilon_2}{x_1 + x_2} = \frac{(\mid x_1 \mid + \mid x_2 \mid) \times \delta}{x_1 + x_2} = \pm\delta$$

Si par contre les nombres sont de signes différents, l'erreur relative peut être amplifiée de façon spectaculaire.

Dans la multiplication, les erreurs relatives s'additionnent.

En effet,soient x_1 et x_2 ,on a :

$$(x_1 + \epsilon_1) \times (x_2 + \epsilon_2) = x_1x_2 + x_1\epsilon_2 + x_2\epsilon_1 + \epsilon_1\epsilon_2$$

Si de plus les erreurs s'écrivent

$$\epsilon_1 = |x_1| \times \delta$$
$$\epsilon_2 = |x_2| \times \delta$$

on a alors en négligeant certains termes:

$$\frac{|(x_1 + \epsilon_1) \times (x_2 + \epsilon_2) - x_1x_2|}{x_1x_2} = \frac{|x_1\epsilon_2 + x_2\epsilon_1|}{x_1x_2} = \frac{\epsilon_1}{x_1} + \frac{\epsilon_2}{x_2} = 2\delta$$

Des formules équivalentes peuvent donner des résultats différents; on peut améliorer le résultant en utilisant une formule mathématique équivalente nécessitant des opérations différentes.

Exemple :

Si on considère les nombres $\sqrt{7001}$*et* $.\sqrt{7000}$.

En arirthmétique flottante à 8 *chiffres , on a :*

$$\sqrt{7001} = 0.83671979 \times 10^2$$
$$\sqrt{7000} = 0.83666003 \times 10^2$$

Donc

$$\sqrt{7001} - \sqrt{7000} = fl((0.83671979 - 0.83666003) \times 10^2) = 0.59760000 \times 10^{-2}$$

On peut obtenir un résultat plus précis en utilisant l'identité suivante:

$$\sqrt{x} - \sqrt{y} = (\sqrt{x} - \sqrt{y}) \times \frac{\sqrt{x} + \sqrt{y}}{\sqrt{x} + \sqrt{y}} = \frac{x - y}{\sqrt{x} + \sqrt{y}}$$

On obtient alors

$$\frac{1}{\sqrt{7001} + \sqrt{7000}} = \frac{1}{0.16733798 \times 10^3} = 0.59759297 \times 10^{-2}$$

0.4 Conditionnement et stabilité numérique.

Le fait que certains nombres ne soient pas représentés de façon exacte dans un ordinateur entraine que l'introduction même de donnée d'un problème en machine modifie quelque peu le problème initial; Il se peut que cette petite variation des données entraine une variation importante des résultats. C'est la notion de conditionnement d'un problème.

On dit qu'un problème est bien (ou mal) conditionné, si une petite variation des données entraine une petite (une grande) variation sur les résultats.

Cette notion de conditionnement est liée au problème mathématique lui même et est indépendante de la méthode utilisée pour le résoudre.

Une autre notion importante en pratique est celle de stabilité numérique. Un problème peut être bien conditionné et la méthode utilisée pour le résoudre peut être sujette à une propagation importante des erreurs numériques.

Ces notions de conditionnement d'un problème et de stabilité numérique d'une méthode de résolution sont fondamentales en analyse numérique. Si un problème est mal conditionné alors la solution exacte du problème tronqué ou arrondi à t digits pourra être très différente de la solution exacte du problème initial. Aucune methode ne pourra rien; il faudra essayer de donner une autre formulation au problème.

0.5 Instabilité numérique :

Si les erreurs introduites dans les étapes intermédiaires ont un effet négligeable sur le résultat final,on dira que le calcul ou l'algorithme est numériquement stable.si des petits changements sur les données entrainent des petits changements sur le résultat. Sinon , on dira que l'algorithme est numériquement instable.

Exemple :

On veut calculer la valeur de

$$I_n = \int_0^1 \frac{x^n}{a+x}\,dx$$

où a est une constante plus grande que 1*, pour plusieurs valeurs de n. pour ce faire, nous allons exprimer I_n récursivement, i.e. nous allons exprimer I_n en fonction de n et I_{n-1}.*

$$\begin{aligned} I_n &= \int_0^1 \frac{x^{n-1}(x+a-a)}{a+x}\,dx \\ &= \int_0^1 x^{n-1}\,dx - a\int_0^1 \frac{x^{n-1}}{a+x}\,dx \\ &= \frac{1}{n} - aI_{n-1} \\ &= \sum_{i=0}^{n-1} \frac{(-a)^i}{n-i} + (-a)^n I_0 \end{aligned}$$

Comme

$$I_0 = ln(\tfrac{1+a}{a})$$

On peut calculer I_n pour toutes les valeurs de n.

Mais l'algorithme est numériquement instable car toute erreur dans le calcul de$I_0 = ln(\frac{1+a}{a})$ va se propager.

En effet si on note par I_0^* la valeur approchée de I_0 et si $I_0^* = I_0 + \epsilon$ alors

$$\begin{aligned} I_n^* &= \sum_{i=0}^{n-1} \frac{(-a)^i}{n-i} + (-a)^n I_0^* \\ &= \sum_{i=0}^{n-1} \frac{(-a)^i}{n-i} + (-a)^n (I_0 + \epsilon) \end{aligned}$$

donc $|I_n - I_n^*| \geq a^n \epsilon$.

Remarques:

Il y a en fait différentes sources d'erreur. Nous pouvons les classer en 3 catégories:

- les erreurs liées à l'imprécision des mesures physiques ou au résultat d'un calcul approché

- les erreurs liées à l'algorithme utilisé

- les erreurs de calcul liées à la machine

En général, pour l'objet de notre cours, si le premier chapitre met l'accent sur les erreurs liées à la machine, nous nous interresserons beaucoup plus aux erreurs liées aux méthodes ou encore aux algorithmes utilisés.

Série de Travaux Dirigés N I

Exercice 1 : En arithmétique flottante avec 3 chiffres significatifs et arrondi, illustrer la non-validité des lois d'associativité et de distributivité.

(On pourra prendre : $x = 854$, $y = 251$ et $z = 852$)

Exercice 2 : Soit p une fonction polynôme de degré n définie par

$$p(x) = \sum_{i=0}^{n} a_i x^i$$

Entrée:

$n, (a_i)_{0 \leq i \leq n}, t$

Sortie:

$val = p(t)$

Description du corps de l'algorithme:

$a \longleftarrow a_n$

$\left\{ \begin{array}{l} pour\ i = n-1 \text{ à } 0 \text{ faire} \\ \quad a \longleftarrow a_i + t \times a \\ \quad\quad fin \end{array} \right.$

$val \longleftarrow a$

Expliquer cet algorithme.

Exercice 3 : En arithmétique flottante, avec $s = 3$, calculer $\sum_{i=1}^{10} \frac{1}{i^2}$.

1) En calculant : $\frac{1}{1} + \frac{1}{4} + ... \frac{1}{100}$

2) En calculant : $\frac{1}{100} + \frac{1}{81} + ... + \frac{1}{1}$

Quel résultat est le plus précis et pourquoi?

Exercice 4 : Résolution d'équations du second degré

Soit l'équation du second degré avec c et $b \succ 0$

$$x^2 + bx + c = 0$$

On suppose que le discriminant $\Delta \succ 0$ est proche de b^2

1) Donner une expression de x_1 et $x_2 (x_1 \prec x_2)$

2) Montrer que la racine x est évaluée avec plus de précision en utilisant $x_1 = \dfrac{c}{x_2}$

3) Vérifier que pour $b = 160$ et $c = 1$, Δ est strictement positif et proche numériquement de b^2

Exercice 5 : On considère le polynôme $ax^2 + bx + c = 0$ $(a \neq 0)$. On suppose que le discriminant $\Delta \succ 0$. On sait que

$$(1) \qquad x_1 = \frac{-b+\sqrt{b^2-4ac}}{2a} \text{ et } x_2 = \frac{-b-\sqrt{b^2-4ac}}{2a}$$

1) Vérifier que $x_1 + x_2 = -\frac{b}{a}$ et $x_1 * x_2 = \frac{c}{a}$.

2) Utiliser ce résultat pour montrer que ces racines peuvent aussi s'écrire sous la forme

$$(2) \qquad x_1 = \frac{-2c}{b+\sqrt{b^2-4ac}} \text{ et } x_2 = \frac{-2c}{b-\sqrt{b^2-4ac}}$$

3) Pour $s = 4$ et trouver les racines de $x^2 + 53.1x + 1 = 0$ en calculant

i) x_1 à partir de (1) et la relation $x_1 * x_2 = \frac{c}{a}$.

ii) x_1 à partir de (2) et la relation $x_1 * x_2 = \frac{c}{a}$.

Quel calcul donne le meilleur résultat et pourquoi?

iii) Si on clacule d'abord x_2, laquelle des formules (1) et (2) serait-il préférable de choisir et pourquoi?

Exercice 6 : Résolvez ces deux systèmes linéaires:

$$(1) \qquad \begin{cases} x + y = 2 \\ x + 1.01y = 2.01 \end{cases}$$

$$(2) \qquad \begin{cases} x + y = 2 \\ x + 1.01y = 2.02 \end{cases}$$

Que remarquez-vous?

Exercice 7 : On cherche les racines de

$$p(x) = (x - 1)(x - 2)(x - 10)$$

Elles sont évidentes.

Si on développe ce polynome en une valeur approchée pour l'une des racines par exemple 10.1

p(x) devient

$$p(x) = (x - 10.1)(x^2 + bx + c)$$

Calculer b et c .

En déduire les racines du polynomes du second degré. Que remarquez-vous?

Exercice 8:

Soient ϵ_1 et ϵ_2 les erreurs absolues sur ;x_1 et x_2 .

On peut écrire:

$$(x_1 \pm \epsilon_1) + (x_2 \pm \epsilon_2) = (x_1 + x_2) \pm (\epsilon_1 + \epsilon_2)$$

En arithmétique flottante, l'erreur relative δ est à peu près constante et les erreurs absolues peuvent être approximativement explicités par :

$\epsilon_1 =\mid x_1 \mid \times\delta, \epsilon_2 =\mid x_2 \mid \times\delta$

Que peut-on en conclure pour l'erreur relative obtenue pour la multiplication et l'addition?

Chapter 1

Résolution de f(x)=0

Introduction
Soit f une fonction numérique d'une variable réelle.
On cherche les racines simples de l'équation

$$(1) \qquad f(x) = 0$$

La première étape consiste à isoler les racines, c'est à dire trouver un intervalle $[a, b]$ dans lequel α est l'unique racine réelle de (1). On supposera que f est continue et dérivable autant de fois que nécessaire dans cet intervalle.

Pour trouver cet intervalle on aura besoin de quelques calculs préliminaires en utilisant soit le graphe des fonctions, soit (si la fonction f est continue dans $[a, b]$) le théorème des valeurs intermédiaires en calculant $f(a)$ et $f(b)$

Si $f(a) * f(b) \prec 0$ f admet un nombre impair de racines dans $[a, b]$

Si $f(a) * f(b) \succ 0$ f admet un nombre pair de racines

Exemple :

Soit. la fonction $f(x) = x - \frac{e^x}{e^x - 2}$

and Settings/ANO2005/Bureau/An-Num-deug1/graphics/ANum1-chap1-old

1.png

$f(x) = x - \frac{e^x}{e^x - 2}$

La fonction n'est pas définie pour $x = ln(2)$ et on a $f'(x) = 1 + \frac{2e^x}{(e^x - 2)^2}$

donc $f'(x) \succ 0$ pour tout x

L'équation a donc 2 racines simples situées de chaque côté de $ln(2)$.

On vérifie sans problème qu'une première racine appartient à $[-1, 0]$ et la deuxième à $[1, 2]$

On supposera donc désormais avoir trouvé un intervalle $[a, b]$ où f admet une unique racine simple et on supposera que f est définie, continue, et autant de fois continument dérivable que nécessaire.

Nous allons à présent définir la notion d'algorithme.

Nous appellerons algoritnme toute méthode de résolution d'un problème donné.

Pour tout problème, nous avons des données et des résultats. Les données sont appelées paramètres d'entrée (input) et les résultats paramètres de sortie (output). Ils constituent l'interface de l'algorithme (ou encore la partie visible de l'algorithme).

Dans ce chapitre, nous désignerons par$\{p_n\}$ une suite de nombres réels .

Il y a plusieurs façons de générer les termes d'une suite. En analyse numérique, on construit les suites à l'aide d'un procédé itératif appelé algorithme.

Les algorithmes classiques que nous allons étudier sont les suivants :

-Méthode de la bissection

-Méthode de Newton-Raphson

-Méthode de la sécante

-Méthode du point fixe

Méthode de la bissection

Considérons une fonction $f(x)$ quelconque, continue et cherchons p tel que $f(p) = 0$

Nous supposons qu'on a localisé par tatonnement un intervalle $[a, b]$ dans lequel la fonction change de signe (c.à.d. $f(a) * f(b) \prec 0$) on pose $c = \frac{a+b}{2}$, si $f(a) * f(c) \prec 0$ on remplace b par c sinon on remplace a par c, et on continue cette operation jusqu'à ce qu'on trouve p avec la précision demandée.

Algorithme de bissection (ou de dichotomie)

But : Donner une fonction continue $f(x)$ et un intervalle $[a, b]$ pour lequel $f(a)$ et $f(b)$ sont de signes contraires, trouver une solution de $f(x) = 0$ dans cet intervalle.

Entrées : a, b les extrémités de l'intervalle
ϵ la précision désirée
N_0 le nombre maximal d'itérations

Sortie : la valeur approchée de la solution de $f(p) = 0$

choix

Si $f(a) * f(b) > 0$: $\rightarrow$'pas de changement de signe'

Si $f(a) * f(b) \leq 0$: $n \longleftarrow 1$

iteration **arrêt-si** $n > N_0$ ou $\mid b-a \mid < \epsilon$ ou $f(a).f(b) = 0$

$p \longleftarrow \frac{a+b}{2}$

choix

Si $f(a) * f(p) \leq 0 :$ $b \longleftarrow p$

Si $f(a) * f(p) > 0 :$ $a \longleftarrow p$

fin-choix

$n \longleftarrow n + 1$

fin-iteration

choix

Si $f(a) * f(b) = 0 :$ a ou b sont solutions

Si $\mid b - a \mid < \epsilon$: $\rightarrow a$ 'solution à ϵ près'

Sinon : 'le nombre maximum d'itération est atteit'

fin-choix

fin-choix

Cet algorithme peut aussi s'écrire sous la forme :

Etape 0: Si f(a)=0 imprimer la solution est a ,aller à l'étape 9
Si f(b)=0 imprimer la solution est b,aller à l'étape 9

Etape 1:
si f(b)f(a)
alors imprimer (il n'y a pas de changement de signe)
aller à l'étape 9

Etape 2:poser n=1

Etape 3:
Tant que N$\leq N_0$,faire les étapes 4 à 7
Etape 4:poser p=$\frac{a+b}{2}$
Etape 5:Si f(p)=0 ou $\frac{b-a}{2} \leq \epsilon$
Alors imprimer p
Fin
Etape 6: poser n=n+1
Etape 7 Si f(a)*f(p)$\succ$ 0
alors poser a=p
sinon poser b=p

Etape 8: Imprimer aprés N_0 itérations l'approximation obtenue est p et l'erreur maximale est $\frac{b-a}{2}$

Etape 9: Fin

Méthode de Newton-Raphson:
INSERER GRAPHE
But: Trouver une solution de $f(x) = 0$
Entrées: une approximation initiale p_0
ε(la précision désirée)
N_0 (le nombre maximum d'itérations)
Sortie: valeur approchée de p ou un message d'échec
Etape 3:poser$p = p_0 - \frac{f(p_0)}{f'(p_0)}$
Etape 4:Si$\mid p - p_0 \mid \leq \varepsilon$ alors imprimer p

Méthode de la sécante

La méthode de Newton -Raphson suppose le calcul de $f'(p)$ à chaque étape. Il se peut qu'on ne dispose pas d'un programme permettant de calculer systématiquement f' .

L'algorithme suivant peut être considéré comme une approximation de la méthode de Newton.

Au lieu d'utiliser la tangente au point p_n nous allons utiliser la sécante passant par les points d'abscisses p_n et p_{n-1} pour en déduire p_{n+1}.

L'équation de la sécante s'écrit :
s(x)=f(p_n)+(x-p_n)$\frac{f(p_n)-f(p_{n-1})}{p_n-p_{n-1}}$
Si s(p_{n+1})=0 , on en déduit:
p_{n+1}=p_n-f(p_n)$\frac{p_n-p_{n-1}}{f(p_n)-f(p_{n-1})}$

INSERER GRAPHE
Algorithme de la sécante:

But: Trouver une solution de $f(x) = 0$
Entrées: deux approximations initiales p_0 et p_1
ε(la précision désirée)
N_0 (le nombre maximum d'itérations)
Sortie:la valeur approchée de p ou un message d'échec

Etape 1: poser N=2
q_0=f(p_0)
q_1=f(p_1)
Etape 2: Tant que N$\leq N_0 + 1$,faire les étapes 3 à 6
Etape 3:poser $p = p_1 - q_1\frac{(p_1-p_0)}{q_1-q_0}$

Etape 4:Si $\mid p - p_1 \mid \leq \varepsilon$ alors imprimer p
Fin
Etape 5: Poser N=N+1
Etape 6:Poser p_0=p_1
q_0=q_1
p_1=p
q_1=f(p)
Etape 7:Imprimer la méthode a échoué aprés N_0 itérations
Etape 8:Fin

Méthode du point fixe

Nous pouvons observer que la méthode de Newton peut s'interpréter comme $p_{n+1} = g(p_n)$ où

$g(x) = x - (\frac{f(x)}{f'(x)})$. Maintenant , si la fonction $g(x)$ est continue et si l'algorithme converge (c.à.d. $p_n \to p$),

on tire de $p_{n+1} = g(p_n)$ que p satisfait l'équation $p = g(p)$; on dit que p est un point fixe de g.

On peut toujours transformer un problème du type $f(x) = 0$ en un problème de la forme $x = g(x)$ et ce d'une infinité de façons.

Par exemple $x^2 - 2 = 0$ ou $x = 2/x$

ou $x = x^2 + x - 2$

ou $x = \alpha(x^2 - 2) + x$

Il faut toutefois noter que ce type de transformations introduisent des solutions 'parasites'.

Par exemple : résoudre $1/x = a$ ou encore $x = 2x - ax^2$

On voit que 0 est racine de la deuxième équation mais pas de la première.

Algorithme du point fixe

But: trouver une solution de $g(x) = x$

Entrées: une approximation initiale p_0
ε(la précision désirée)

N_0 le nombre maximale d'itérations

Sortie: valeur approchée de p ou un message d'échec
Etape 1: poser N=1

Etape 2 Tant que N$\leq N_0$,faire les étapes 3 à 6
Etape 3:poser p=g(p_0)
Etape 4 Si $\mid p - p_0 \mid \leq \varepsilon$
alors imprimer p
Fin
Etape 5:poser n=n+1
Etape 6:poser p_0=p

Etape 7: Imprimer (la méthode a échoué aprrès N_0 itérations)

Convergence des algorithmes
Ordre de convergence
Etude de la convergence des méthodes itératives à un pas

Ordre de convergence
Considérons une suite $\{p_n\}$ convergeanr vers p et posons $e_n = p_n - p$.
On dit dans le cas où $\left\{\left|\frac{e_n}{e_{n-1}}\right|\right\}$ converge, que la suite p_n converge linéairement vers p ou encore que la méthode est du premier ordre.
Si on a $\left\{\left|\frac{e_n}{(e_{n-1})^k}\right|\right\}$ converge, alors la convergence est dite d'ordre k
Par exemple la suite $p_n = \frac{1}{n}$est d'ordre 1

Série d'exercices n°2

Exercices 1
Résoudre à l'aide de la méthode de bisection $tanx - x = 0$ dans l'intervalle $[4; 4.7]$.
Exercice 2
On considère l'équation
(1) $e^x - 4x = 0$
1)Déterminer le nombre et la position approximative des racines de (1) situées dans $.x \geq 0$
2)Utiliser l'algorithme de bissection pour déterminer la plus petite de ces racines à ε près.(par exemple 10^{-7})

3)Sans faire d'itérations, déterminer combien vous devriez en faire pour calculer la plus grande racine à l'aide de la bissection avec une précision de 10^{-8}, si l'intervalle de départ est $[2; 2, 5]$

Exercice 3

Écrire un algorithme pour calculer par la méthode de Newton la racine K-ième d'un nombre.

Quelle est la valeur de $s = \sqrt{2 + \sqrt{2 + \sqrt{2 + ...}}}$?

Suggestion: écrire .p_{n+1}=G(p_n),p_0=0 Quel est le taux de convergence ?

Exercice 4

Écrire 3 méthodes itératives pour la résolution de $x^3 - x - 1 = 0$ et vérifier expérimentalement leur convergence avec $x_0 = 1, 5$. Trouver à 10^{-6}près la racine comprise entre 1 et 2. Connaissant la valeur de cette racine, calculer le taux de convergence de vos 3 méthodes. Ce résultat coincide-t-il avec l'expérience?

Exercice 5

Résoudre x^2-1=0 en utilisant la méthode de la sécante avec $x_0 = -3$ et $x_1 = 5/3$. Qu'arrivera-t-il si on choisit et $x_0 = 5/3$ et $x_1 = -3$? Expliquez.