Analyse Numérique

Analyse des erreurs

Laurence Lamoulie

EISTI

Ecole Internationale des Sciences du Traitement de l'Information

Infos

- Site du cours : http ://sifoci.eisti.fr
- sifoci : Systèmes Informatiques FOrmels et Intelligents
- Vous y trouverez
  - Le planning du cours réactualisé chaque semaine.
  - Le poly du cours et les transparents de Pau et de Cergy.
  - Des livres et des articles sous forme électronique que vous êtes fortement encouragés à consulter.
  - Des informations de dernière minute, le cas échéant.

Menu

- 13 séances : 10 cours 1h00, 10 TD 1h30,2 TP 2h30 Examen : 3h00.
- Pendant les TD on travaillera essentiellement sur les exercices du poly.
- Les TP seront notés. Pour avoir une note > 0 il faut que vous soyez présents. Il n'y aura pas de rattrapage du TP. Leur moyenne compte pour 25% de la note de la matière.
- La note de l'évaluation de Scilab compte pour 10% de la note d'Analyse numérique
- CALCUL DE LA NOTE FINALE : 10% de la note Scilab , 25% de la note du TP, 65% de la note de l'examen

Infos

Faire des calculs, surtout avec un ordinateur, n'est pas simple : le calcul "à la main" n'est pas transposable "à la machine" et réciproquement. L'analyse numérique est

- une branche des mathématiques, et
- un ensemble des techniques informatiques qui vous permettront de mieux vous débrouiller avec les calculs sur ordinateur.

Elle requiert donc de comprendre les maths ET l'informatique

Infos

L'erreur du calcul numérique est inévitable.

Stockage d'un nombre réel dans un registre avec un nombre fini de bits.
Soit M l'ensemble fini des nombres qu'un ordinateur peut stocker.
Pour un $x \in R$ on n'a pas obligatoirement $x \in M$. Par exemple $\sqrt{2} \in R$ mais $\sqrt{2} \notin M$.
Ainsi $x \in R$ devient dans un ordinateur $fl(x) \in M$, où $fl(x)$ sera appelé nombre-machine, avec $x = fl(x)$.
Exemple : Si on rajoute 10 fois la valeur $0,1$ le résultat obtenu avec Scilab est $9,9999999999999989.10^{-1}$, soit une erreur de $1,11.10^{-16}$

Vocabulaire

Notation scientifique normalisée en décimal

Forme d'un nombre en notation scientifique normalisée : $0,a_1a_2a_3...a_p.10^n$, où $1 \leq a_1 \leq 9$, $0 \leq a_i \leq 9$ pour $1 < i \leq p$ et $n \in Z$.

Notation scientifique normalisée en binaire

Forme d'un nombre en notation scientifique normalisée binaire : $1,a_1a_2a_3...a_p.10^n$, où $0 \leq a_i \leq 1$ pour $0 < i \leq p$ et $n \in Z$.

Attention : la base de numération est toujours notée 10 indépendamment du système utilisé (binaire, décimal, hexadécimal...). Le nombre p est fixé d'avance, il est appelé **mantisse**.

Calculs

Problème 1

Le zéro $0.0.10^0$ ne peut pas être représenté en écriture scientifique normalisée !

Problème 2

Multiplication ou division avec deux digits décimaux :
$x_1 \times x_2 = (a_1 \times a_2) \times 10^{n_1+n_2}$
$x1/x2 = (a1/a2) \times 10^{n_1-n_2}$
Exemple :
$(0.567 \times 10^{-4}) \times (0.234 \times 10^3) = 13.33 \times 10^{-3} = 0.133 \times 10^{-1}$
au lieu de 0.132678×10^{-1}

Exemple

Rappels : Passage du décimal au binaire

- Pour la partie entière
  $13/2 = 6$ reste 1
  $6/2 = 3$ reste 0
  $3/2 = 1$ reste 1
  donc on obtient : $13_{10} = 1101_2$
- Pour la partie décimale
  $0.8125 * 2 = 1.625$ de partie entière 1
  $0.625 * 2 = 1.25$ de partie entière 1
  $0.25 * 2 = 0.5$ de partie entière 0
  $0.5 * 2 = 1$ de partie entière 1
  $0.0 * 2 = 0$ de partie entière 0
  donc obtient : $0.8125_{10} = 0.1101_2$
- Au final on a par exemple : $13.8125_{10} = 1101.1101_2$

Exemple (suite)

Passage en machine en simple précision

- Nombre positif : bit de signe $= 0$
- On passe à la notation avec exposant :
  $13.8125_{10} = 1101.1101_2 = (1.1011101 * 10^{11})_2$
- Exposant : il vaut $11_2 = 3_{10}$
  Exposants disponibles : codage sur 8 bits donc variant de 0 à 255, soit 127 valeurs pour les exposants négatifs et 127 pour les exposants positifs.
  Conséquence : l'exposant est codé par $3 + 127 = 130 = 10000010_2$

Exemple (suite)

Passage en machine en simple précision

- Mantisse : on complète pour obtenir 23 bits et on a donc : 1(.)10111010000000000000000
- Comme on a toujours un premier chiffre de mantisse (à gauche du . qui est omis) égal à 1, on ne l'écrit pas (bit caché) et on a la représentation définitive :
  0 10000010 10111010000000000000000

Certains nombres ne peuvent pas s'accomoder de ces conventions :

- 0 est codé par convention
  0 00000000 00000000000000000000000
  ce qui devrait correspondre à $(1.0 * 10^0)_2 = 1_{10}$

On est donc obligé de réserver l'exposant 00000000

Conséquences sur les nombres représentables

Plus grand et plus petit nombres normalisés

- Plus petit nombre normalisé :
  0 00000001 00000000000000000000000
  Il correspond à $1.0 * 2^{1-127} = 2^{-126} \approx 1.2 * 10^{-38}$ donc les exposants varient entre -126 et 127 et non pas -127 et 127.
- Plus grand nombre normalisé :
  0 11111110 11111111111111111111111
  Il correspond à
  $(1.11...1)_2 * 2^{254-127} = (2 - 2^{-23} * 2^{127} \approx 3.4 * 10^{38}$

Conséquences sur les nombres représentables

Plus grand et plus petit nombres normalisés

- Nombres spéciaux :utilisables par l'ordinateur seulement
  - Nombres sous normalisés
    Le plus petit nombre non nul stockable est :
    0 00000000 00000000000000000000001
    Il correspond $0.00\ldots01 * 2^{-127} = 2^{-149}$
    En cas de résultat égal à un nombre sous-normalisé : underflow
  - Nombres sur normalisés : idem et overflow

  Si la mantisse est nulle : on a représenté $\pm\infty$ sinon on a un non-nombre noté NaN

Arithmétique arrondie : Situations possibles

Un nombre x peut être :

- égal à un nombre normalisé
- entre le plus petit et le plus grand nombres normalisés mais pas égal à un nombre normalisé
- un nombre sous-normalisé
- une valeur infinie ($+\infty$ ou $-\infty$)
- NaN

Les normes définissent ce qui doit être fait dans chaque cas.

Définitions

Définition 1.3.1

La précision eps d'un ordinateur est le plus petit nombre positif normalisé tel que

$$1 + \text{eps} = 1$$

Fait 1.1

La précision eps d'une machine simple précision qui suit la norme IEEE-754 est

$$\text{eps} = 2^{-23} \approx 1.192 * 10^{-7}$$

Conséquences

Résultat

Si x est un nombre à représenter selon le standard IEEE754 alors on le remplacera soit par x_- soit par x_+. le nombre ainsi choisi est m(x).
On a

$$|m(x) - x| \leq \frac{1}{2} eps.2^E$$

Eléments théoriques

Définitions

- Erreur (resp. erreur absolue)de représentation : Δx(resp. $|\Delta x|$)

$$\Delta x = m(x) - x \qquad (\text{resp. } |\Delta x| = |m(x) - x|)$$

- Erreur relative de représentation : $\iota(x)$

$$\iota(x) = \frac{\Delta x}{m(x)} = \frac{m(x) - x}{m(x)}$$

Eléments théoriques

Définitions - suite

- Erreur relative (resp. erreur relative absolue) de précision : $\eta(x)$ (resp. $|\eta(x)|$)

$$\eta(x) = \frac{\Delta x}{x} = \frac{m(x) - x}{x} = \frac{m(x)}{x} - 1$$

$$(\text{resp. } |\eta(x)| = \frac{|m(x) - x|}{|x|})$$

Conséquence

L'erreur relative de précision permet d'exprimer le nombre de précision machine **eps** par :

$$m(x) = x(1 + \eta(x))$$

Eléments théoriques

Théorème 1.4.1 : Erreur de précision relative

Sur un calculateur en base β avec une mantisse de p chiffres, l'erreur de précision relative est bornée pour tout $x \in R$ par :

$$|\eta(x)| \leq \begin{cases} \beta^{1-p} & \text{si approximation par troncature} \\ \frac{\beta^{1-p}}{2} & \text{si approximation par arrondi} \end{cases}$$

Fait 1.2

L'erreur de précision relative selon le standard IEEE-754 est

$$|\eta(x)| \leq \begin{cases} 2^{1-p} & \text{si approximation par troncature} \\ 2^{-p} & \text{si approximation par arrondi} \end{cases}$$

si on utilise une représentation binaire