INFORMATIQUE APPLIQUÉE AU CALCUL SCIENTIFIQUE

COURS 1

# Représentation des nombres en machine

1) Zéro et l'infini
2) Application à la résolution d'une équation
3) Suite et série géométrique
4) Erreurs d'arrondis
5) Signe, exposant, mantisse

François Dubois, 04 octobre 2004, édition 03 août 2006, 13 pages.

IACS ①

# Représentation des nombres en machine.

9/10/04
D

(CNAM Paris)

## ① Zéro et l'infini.

- Une expérience numérique simple avec le logiciel "octave" (clone américain du progiciel "Matlab", dans le domaine public, téléchargeable sur www.octave.org) consiste à prendre les puissances successives de $x = 0,1$. L'exponentiation $x^n$ se note x***n avec octave, et si $x^{300}$ est encore bien représenté en machine, le résultat de $x^{400}$ affiche simplement "zéro". Il existe donc un "plus petit nombre" non nul que la machine peut représenter ; il est compris entre $10^{-300}$ et $10^{-400}$ ; notons le ici $\eta$ :

(1) $\exists \eta > 0$, si $x \neq 0$ est représentable en machine, alors $|x| > \eta$.

- afin de mieux évaluer $\eta \in [10^{-400}, 10^{-300}]$ de manière expérimentale, on met en œuvre sur l'ordinateur un <u>algorithme de dichotomie</u>. Cet algorithme suppose qu'une propriété $P(m)$ paramétrée par $m$ (nombre entier ici pour fixer les idées) dans un intervalle $[\alpha, \beta]$ est telle que

(2) $P(\alpha)$ est vraie et $P(\beta)$ est fausse.

- Dans l'exemple de la recherche de $\eta$ (relation (1)), on a $\alpha = 300$, $\beta = 400$ et $P(m)$ donné par "le nombre $x = 10^{-m}$ est représentable en machine par un nombre $\neq 0$".

- Il est alors intuitif (modulo des hypothèses de monotonie et de continuité sur lesquelles nous reviendrons) qu'il existe alors $\gamma \in ]\alpha, \beta[$ de sorte que $P(m)$ est vraie si $m < \gamma$ et $P(m)$ est fausse si $m > \gamma$. L'objectif de l'algorithme de dichotomie est d'approcher cette valeur limite $\gamma$ où la propriété $P(m)$ change de valeur de vérité.

- L'algorithme de dichotomie consiste à changer $[\alpha, \beta]$ en $[\alpha', \beta']$ "deux fois plus petit", ie tel que

(3) $\beta' - \alpha' = \frac{1}{2}(\beta - \alpha)$.

L'idée est d'introduire un candidat $\gamma$ intermédiaire entre $\alpha$ et $\beta$, et chercher la valeur de vérité de $P(\gamma)$. L'algorithme de dichotomie propose

(4) $\gamma = \frac{1}{2}(\alpha + \beta)$.

Il y a alors deux éventualités

(i) Si $P(\gamma)$ est vraie alors la propriété $P(m)$ change de valeur de vérité entre $m = \gamma$ et $m = \beta$. On pose alors $\alpha' = \gamma$, $\beta' = \beta$ et on a bien la relation (3), compte tenu du choix (4):

$\beta' - \alpha' = \beta - \frac{1}{2}(\alpha+\beta) = \frac{1}{2}(\beta-\alpha)$

(ii) Si $P(\gamma)$ est fausse, la propriété change de valeur de vérité entre $m=\alpha$ et $m=\gamma$ cette fois. On pose alors $\alpha'=\gamma$, $\beta'=\beta$ et la relation (4) a encore lieu : $\beta'-\alpha' = \frac{1}{2}(\alpha+\beta)-\alpha = \frac{1}{2}(\beta-\alpha)$.

- On itère l'algorithme en posant $\alpha=\alpha'$, $\beta=\beta'$ à la fin du "branchement" précédent écrit aux étapes (i) et (ii). A chaque étape on dispose d'un intervalle $[\alpha_k, \beta_k]$ $(k \geq 1)$ tel que

(5) $\alpha_0=\alpha$, $\beta_0=\beta$, $\beta_{k+1}-\alpha_{k+1} = \frac{1}{2}(\beta_k-\alpha_k)$

on a alors $\beta_k-\alpha_k = \left(\frac{1}{2}\right)^k(\beta-\alpha)$ et la "taille" de l'intervalle de recherche du paramètre $\gamma$ tend vers 0 comme une suite géométrique de raison $\frac{1}{2}$, c'est à dire assez vite.

- L'algorithme précédent parvient en moins de 10 étapes de montrer que

(6) $\left(\frac{1}{10}\right)^{323} \neq 0$ en machine, alors que $\left(\frac{1}{10}\right)^{324}$ est représenté par "0".

Donc le nombre $\eta$ introduit à la relation (1) vérifie $10^{-324} < \eta < 10^{-323}$.

- on reprend l'étude précédente en partant cette fois de $y = 10$ au lieu de $x = \frac{1}{10}$, et on calcule les puissances successives de $y$ avec le logiciel octave. on trouve (en utilisant à nouveau l'algorithme de dichotomie) que $10^{308}$ est bien représenté en machine, mais qu'en revanche, quand on multiplie ce nombre par deux, octave répond "Inf", et considère $2.10^{308}$ comme "infiniment grand" :

(7) $\exists A > 0$, si $x \neq 0$ est représentable en machine, alors $|x| < A$.

## (2) application à la résolution d'une équation

L'algorithme de dichotomie peut être mis en œuvre pour résoudre une équation du type "$f(x) = 0$" (où $f$ est une fonction continue de $\mathbb{R}$ dans $\mathbb{R}$) grâce au théorème des valeurs intermédiaires.

**(Th) des valeurs intermédiaires**

Soient $\alpha < \beta$ deux nombres réels, $f : [\alpha, \beta] \to \mathbb{R}$ une fonction continue telle que

(8) $f(\alpha) \cdot f(\beta) < 0$.

Alors il existe $\gamma \in ]\alpha, \beta[$ tel que $f(\gamma) = 0$.

- Le point important dans le résultat précédent est la continuité de la fonction $f(\cdot)$. Le point délicat en pratique est de trouver $\alpha$ et $\beta$ de sorte que $f(\alpha) > 0$ et $f(\beta) < 0$ ou bien $f(\alpha) < 0$ et $f(\beta) > 0$, c'est à dire un premier encadrement du (des?) "zéro" de la fonction $f(\cdot)$.

- Nous pouvons utiliser l'algorithme précédent pour construire une approximation de $\sqrt{2}$. Il suffit de choisir

$$(9) \quad f(x) = x^2 - 2$$

qui vérifie $f(1) = -1 < 0$, $f(2) = 2 > 0$. Les itérations de l'algorithme fournissent les valeurs suivantes de $\gamma$

| $k=$ | $\gamma=$ |
|---|---|
| 0 | 1,5 |
| 1 | 1,25 |
| 2 | 1,375 |
| 3 | 1,4375 |
| 4 | 1,40625 |
| 5 | 1,421875 |
| 6 | 1,4140625 |
| 7 | 1,41479375 |

Nous invitons le lecteur à reconstituer le programme "octave" correspondant, une fois quelques rudiments de programmation acquis.

## (3) Suite et série géométriques

- Nous poursuivons les rappels de "mathématiques élémentaires" par l'étude des suites géométriques. Une suite géométrique $(u_n)_{n\in\mathbb{N}}$ à valeurs dans $\mathbb{R}$ ou $\mathbb{C}$ (nous nous bornerons ici à $\mathbb{R}$ pour simplifier l'exposé) vérifie

(10) $\quad u_{n+1} = q\,u_n \quad , \forall n \in \mathbb{N}.$

on vérifie facilement (preuve par récurrence laissée au lecteur !) qu'on a alors

(11) $\quad u_n = q^n u_0 \quad , n \in \mathbb{N}.$

Une question naturelle est de savoir "ce que devient" $u_n$ si $n$ tend vers l'infini, quelle est la limite de la suite $u_n$ si $n \nearrow \infty$.

- Si $q = 1$ (la raison $q$ vaut 1), la suite $(u_n)_{n\in\mathbb{N}}$ est <u>constante</u>, donc "tend vers" $u_0$ si $n \to \infty$.

* Si $0 \leq q < 1$, la suite $u_n$ tend vers 0 si $n \to \infty$

* Si $q > 1$, $u_n \to \pm\infty$ et le signe est $+$ si $u_0 > 0$, $-$ si $u_0 < 0$.

* Si $q = -1$, la suite <u>oscille</u> : $u_0, u_1 = -u_0$, $u_2 = u_0, u_3 = -u_0$, etc. Elle n'a <u>pas</u> de limite, mais la "suite extraite" avec les entiers pairs $(u_{2k})_{k\in\mathbb{N}}$ est constante égale à $u_0$, celle avec les entiers impairs $(u_{2k+1})_{k\geq 0}$ est constante égale à $-u_0$.

* Si $-1<q<0$, la suite $(u_n)_{n\in\mathbb{N}}$ change de signe à chaque étape, et tend vers 0 si $n\to\infty$.
* Si $q<-1$, la suite $(u_n)_{n\in\mathbb{N}}$ change toujours de signe à chaque étape, et de plus, $|u_n|\nearrow\infty$ si $n\to\infty$.

• On forme maintenant la série $(S_n)_{n\in\mathbb{N}}$ définie en sommant les "n" premiers termes de la suite $u_n$. On pose

(12) $S_0 = u_0$, $S_{k+1} = S_k + u_{k+1}$, $k\in\mathbb{N}$

et on a facilement (par récurrence laissée au lecteur) $S_k = \sum_{j=0}^{k} u_j$. Si on pose $u_0 = 1$, on se propose de donner une autre expression pour

(13) $S_n = 1+q+q^2+\cdots+q^n \equiv \sum_{j=0}^{n} q^j$.

* Si $q=1$, on a simplement $S_n = (n+1)$ et il n'y a rien de plus à dire.
* Si $q\neq 1$, on multiple $S_n$ par $(1-q)$. Il vient

$$(1-q)S_n = (1+q+\cdots+q^{n-1}+q^n) - (q+q^2+\cdots+q^n+q^{n+1})$$
$$= 1-q^{n+1} \quad \text{Donc}$$

(14) $$S_n = \frac{1-q^{n+1}}{1-q} = \frac{q^{n+1}-1}{q-1}$$

* Dans le cas $q=2$ et $n=9$, on a $S_9 = 2^{10}-1 = 1023$

## (4) Erreurs d'arrondis

- Un ordinateur calcule toujours faux !
  Lors d'une opération arithmétique simple comme l'addition, il effectue des erreurs qu'on peut mettre en évidence en calculant l'expression

  (15) $\quad z = 1+\varepsilon$

  avec $\varepsilon$ nombre de plus en plus petit. L'expérience avec 'octave' montre que

  $1+10^{-14}$ est différent de 1 en machine
  $1+10^{-16}$ est identifié à 1 dans la mémoire de l'ordinateur.

- Grosso-modo, l'ordinateur fait les calculs avec 15 chiffres significatifs. En pratique, on a peu de garantie de qualité des chiffres fournis par 'octave' au delà du douzième.

- Pour avoir une meilleure précision que celle fournie par la "double précision" d'une machine qui possède un processeur qui effectue des calculs élémentaires avec 32 bits, il faut reprogrammer les opérations élementaires avec une arithmétique dont on se donne a priori la précision. C'est un autre travail que celui auquel on s'intéresse dans ce cours IACS, où l'on considère a priori que la précision des calculs qu'effectue l'ordinateur avec son arithmétique usuelle est suffisante.

## (5) Signe, exposant, mantisse

- les nombres sont stockés en machine sur 64 bits avec le logiciel 'octave'. Ainsi, zero se note $\underbrace{0\ldots0}_{64\text{ "zéros"}}$, un se note $00\underbrace{1\ldots1}_{10\text{ "un"}}\underbrace{0\ldots0}_{52\text{ "zéros"}}$ ce qui est déjà relativement complexe. Notons que la notation binaire de "2" (le 'format bit' du logiciel 'octave') est plus simple, puisque "deux" se note $01\underbrace{0\ldots0}_{62\text{ "zéros"}}$. Le nombre "4" se note en machine sous la forme $01\underbrace{0\ldots01}_{9\text{ "0"}}\underbrace{0\ldots0}_{52\text{ "0"}}$
- Ce stockage correspond à la norme "IEEE 754". Il s'agit d'une norme* proposée en 1985 par l'association professionnelle des ingénieurs électroniciens américains (IEEE = Institute for Eletrical and Electronics Engeneers) et adoptée comme une norme "ANSI".

* Standard for Binary Floating Point Arithmetic. Elle est de plus en plus répandue dans la plupart des machines électroniques.

- Un nombre "flottant" $x$ est représenté par un triplet $(s, e, m)$, où $s$ est le signe, $e$ l'exposant et $m$ la mantisse du nombre $x$. Par convention $(s, e, m)$ sont des nombres entiers stockés en base deux, donc uniquement à l'aide des symboles "0" et "1".

  * le signe $s$ est codé sur 1 bit : $0 \leq s \leq 1$
  * l'exposant $e$ est codé sur 11 bits : $0 \leq e \leq 2046$
    Car effet, $1 + 2 + \cdots + 2^{10} = 2^{11} - 1 = 2046 + 1$
  * la mantisse $m$ est codée sur 52 bits : $0 \leq m \leq 2^{52} - 1$

Nous retenons

(16) $\quad 0 \leq s \leq 1, \quad 0 \leq e \leq 2047, \quad 0 \leq m \leq 2^{52} - 1$

- La règle de codage du nombre $x$ avec la norme IEEE 754 est la suivante. Si $y$ est un entier tel que $0 \leq y \leq 2^{64} - 1$, on décompose $y$ sous la forme

(17) $\quad y = s\, 2^{63} + e\, 2^{52} + m$

et en numération binaire, on lit le nombre $y$, codé avec 64 symboles, sous la forme

(18) $\quad y \leftrightarrow (s, e, m) \begin{cases} s & \text{sur } 1 \text{ bit} \\ e & 11 \text{ bits} \\ m & 52 \text{ bits} \end{cases}$

Une fois le "format bit" identifié, le nombre $x$ se calcule grâce aux relations suivantes.

On introduit un "biais" b qui avec l'octave vaut $1023 = 2^{10} - 1$ :

$$b = 1023 = 2^{10} - 1 \tag{19}$$

On pose alors, avec y donné en (17)(18) :

$$x = \begin{cases} (-1)^s \left(1 + m \cdot 2^{-52}\right) 2^{e-b} & \text{si } e \geq 1 \\ (-1)^s \, m \cdot 2^{-52} \, 2^{-b} & \text{si } e = 0 \end{cases} \tag{20}$$

Cette définition semble un peu complexe, mais c'est la plus fidèle que nous avons pu identifier à l'aide de nos expériences avec l'Octave

- Si $s = e = m = 0$, i.e. $y = 0$, on tire de (20) que $x = 0$. Le nombre "0" est bien représenté par 64 "zéros" avec la norme IEEE.

* Le nombre "1" s'écrit $1 = (-1)^0 (1+0) \, 2^{b-b}$, on le représente par la chaîne

0 0 1 --- 1 0 ——— 0

dix "un" 52 "zéros"

ce qui indique bien $s = 0$, $e = 1 + 2 + \cdots + 2^9$
$e = 2^{10} - 1 = 1024 - 1 = 1023 = \text{biais}$, $m = 0$.
C'est un peu compliqué, mais on s'y retrouve

* Plus simplement, $2 = (-1)^0 (1+0) \, e^{b+1-b}$
donc $s = 0$, $e = b + 1 = 1024 = 2^{10}$, $m = 0$

qu'on lit en format 'bit' sous la forme

"2" ↔ 0 | 1 0 — — 0 | 0 — — — 0 |

signe — 10 "zéros" — 52 "zéros"

11 bits d'exposant ;
$e = 2^{10}$.

* Pour le nombre "4", on a : $4 = (-1)^0 (1+0)\, 2^{b+2-b}$

Donc $s = 0$, $m = 0$, $e = 1023 + 2 = 2^{10} + 1$

et se code donc sous la forme

"4" ↔ 0 | 1 0 — — 0 1 | 0 — — — — 0 |

neuf "zéros — 52 "zéros"

- Si on cherche le plus petit nombre $\neq 0$ représentable en machine (le nombre $\eta$ de la relation (17)), on a compté tenu des relations (16) et (20)

(21) $|x| \geqslant 2^{-1074}$ $\{1074 = 1023 + 51\}$

et $\eta = 2^{-1074} \simeq 4{,}94\ 10^{-324}$ ainsi que nous l'avions découvert expérimentalement (relation (6)) à l'aide de l'algorithme de dichotomie.

- Pour le plus grand nombre en valeur absolue, on maximise d'abord l'exposant, avec $e-b=1023$. Puis on prend une mantisse dont tous les chiffres valent 1, soit $m = 1+2+\cdots+2^{-51} = 2^{52}-1$, on trouve

$$|x| \leqslant 2^{1023}\left(2-2^{-52}\right) = 2^{1024}\left(1-2^{-53}\right)$$

Donc

$$(22) \quad |x| \leqslant 2^{1024} \simeq 1,7976\,9313\,10^{308}$$

et le nombre A de la relation (7) est maintenant correctement déterminé. Au delà, l'ordinateur indique "+ Inf".

- Ce qui concerne les erreurs d'arrondis, le plus petit nombre $\varepsilon$ qu'on peut ajouter à 1 de sorte que $1+\varepsilon > 1$ en machine, ie

$$(23) \quad \varepsilon = \inf\{x>0,\ 1+x>1 \text{ en machine}\}$$

s'obtient en prenant la mantisse minimale, pour soit $m=1$, ie $x = 2^{-52}$ compte tenu de la relation (20). On a effectivement $1+2^{-52} > 1$ alors que $1+2^{-53}$ est représenté par "1" dans la mémoire de l'ordinateur! On a donc

$$(24) \quad \varepsilon = 2^{-52} \simeq 2,22\times 10^{-16}$$

ce qui indique la taille relative des erreurs d'arrondis.