Traitements Numériques pour les Systèmes Embarqués

1) Arithmétique flottante et analyse d'erreur

Matthieu Martel

UPVD - LAMPS

matthieu.martel@univ-perp.fr.fr

Introduction

Flottants utilisés à la place des réels pour les calculs sur ordinateurs

Arithmétiques très différentes :

- Opérations non associatives, non distributives, non inversibles
  - $\sqrt{2.0} = 1.414\ldots = x \quad x^2 = 1.9999\ldots$
  - dans les doubles :
    $x + 16.0 = 16.0$ ssi $x \in$
    $[-8.88178419700125232e^{-16}, 1.77635683940025046e^{-15}]$
- Nombre fini de décimales
  - Exceptions : overflow, underflow, NaN. `if (x<>0) y = 1/(x*x)`
  - Pertes de précision : erreurs "subjectives" ne déclenchent pas d'exception/interruption

Bugs connus

- Missile Patriot (1991) : $\frac{1}{10} = \frac{1}{2^4} + \frac{1}{2^5} + \frac{1}{2^8} + \ldots = 0.00011001\underline{1100}$. Codé sur 24 bits : erreur de $9.5 \cdot 10^{-8}$. Au bout de 100 heures, $100 \times 3600 \times 10 \times 9.5 \cdot 10^{-8} = 0.34$ secondes. Vitesse du Scud : $1676 ms^{-1}$. Erreur > 500 mètres.
- Autres exemples : bourse de Vancouver, elections allemandes, plateforme pétrolière
- Flottants de + en + utilisés dans des systèmes embarqués critiques (avions, voitures, centrales nucléaires, etc.)
- Besoin de techniques formelles de validation $\neq$ méthodes classiques des numériciens (tests/comparaison à des résultats obtenus analytiquement ou expérimentalement)

Généralités sur les flottants

1. Norme IEEE 754
2. Mesure des erreurs

La norme IEEE 754

$$\begin{aligned} f &= \pm d_0.d_1 d_2 \ldots d_{p-1} \beta^e \\ &= \pm \left(d_0 + d_1 \beta^{-1} + \ldots d_{p-1} \beta^{-(p-1)} \right) \beta^e \end{aligned} \tag{1}$$

- β : base, e : exposant ($e_{min} \leq e \leq e_{max}$), p : précision, $d_0.d_1 \ldots d_{p-1}$ mantisse avec $0 \leq d_i < \beta$
- Nombres *flottants* : nombres représentables par (1)
- Nombres *normalisés* : nombres flottants t.q. $d_0 \neq 0$ $\Rightarrow$ unicité
- Nombres non-représentables :
  - Hors des bornes : $x < \beta^{e_{min}}$ ou $x > \beta^{e_{max}}$
  - Précision : π, 0.1 en base 2

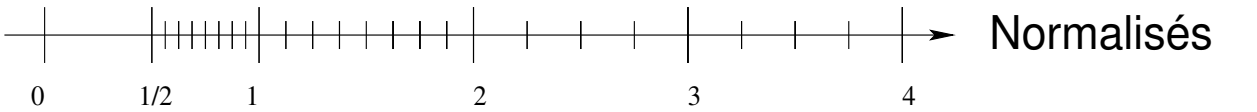
Norme IEEE 754 : formats

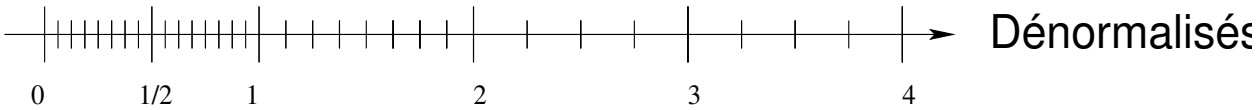
Type	Mantisse (bits)	e_{max}	e_{min}	Exposant (bits)
Single	23+1	+127	-126	8
Single Extended	$\geq$ 32	$\geq$+1023	$\leq$-1022	$\geq$11
Double	52+1	+1023	-1022	11
Double Extended	$\geq$64	$>$16383	$\leq$-16382	$\geq$15

- Rayon d'un proton : $1.2 \times 10^{-15} m$
- Masse d'un electron : $9.1 \times 10^{-31} kg$
- Masse de la voie lactée : $2.2 \times 10^{41} kg$
- Age de l'univers : $4.17 \times 10^{17} s$

Valeurs particulières

Valeur	Signification	Exposant	Mantisse
$\pm\infty$	Dépacement	$e_{max}+1$	0
NaN	Not a Number	e_{max}	$\neq 0$
± 0	Zéro signé	$-e_{max}$	0
$0.dd\ldots$	Dénormalisés	$-e_{max}$	$\neq 0$





$\beta = 2,\ p = 3,\ -1 \leq e \leq 1$

Opérations élémentaires sur les flottants

- Entre valeurs particulières :

$$-\infty \times -\infty = +\infty, \quad \pm 0 \div \pm\infty = \pm 0, \quad \pm\infty \div \pm\infty = \mathit{NaN}, \ \mathit{etc.}$$

- Pour les valeurs “classiques” $\rightarrow$ garantie de l’arrondi
- 4 modes d’arrondi : vers 0, vers $+\infty$, au plus près, vers $-\infty$

$$\uparrow_\circ : \mathbb{R} \rightarrow \mathbb{F}$$

$$\text{Pour } \lozenge \in \{+, -, \times, \div, \sqrt{}\}, \ x \lozenge_{\mathbb{F}} y = \uparrow_\circ (x \lozenge_{\mathbb{R}} y)$$

- Obtenu en utilisant des bits de garde (bits supplémentaires)
- Donne une sémantique précise aux opérations sur les flottants

Mesure des erreurs

- Erreurs déclenchant une exception : Overflow Underflow NaN
- Pertes de précision (ne déclenchent pas d'exception) :
  - Absorption $x + y = x$ si $x \gg y$ (cf. $x + 16 = 16$)
  - Branchement / Test instable
  - Elimination catastrophique (cancellation), $x - y$ avec $x \approx y$

$$A = \sqrt{s(s-a)(s-b)(s-c)} \quad \text{avec } s = \frac{a+b+c}{2}$$

Si le triangle est plat, $a \approx b + c$ et $s \approx a$

Si $a = 9.00$, $b = c = 4.53$:

$s_{\mathbb{R}}$	$A_{\mathbb{R}}$	$s_{\mathbb{F}}$	$A_{\mathbb{F}}$	Erreur
9.03	2.34	9.04	2.71	15%

Ulp, erreurs relatives

- *ulp* : **U**nit in the **L**ast **P**lace, ordre de grandeur du plus petit chiffre significatif d'un nombre
- ex : si $p = 3$, $\beta = 10$, $x = 1.23 \cdot 10^4$, $ulp(x) = 1 \cdot 10^2$
- Erreur relative : $e_r = \left| \frac{r_{\text{exact}} - r_{\text{approché}}}{r_{\text{exact}}} \right|$
- Elimination catastrophique $\Rightarrow$ erreur relative importante
- aire du triangle : dans $\mathbb{R}$, $9.03 - 9.0 = 0.03$; dans $\mathbb{F}$, $9.04 - 9.0 = 0.04$; $e_r = \left| \frac{0.03 - 0.04}{0.03} \right| = 33\%$

ϵ-machine, Erreurs d'ordre supérieur

- ϵ-machine : maximum de l'erreur relative due à un arrondi au voisinage d'un point x.

$$\epsilon = ulp(1) \qquad x = 1.23 \cdot 10^4,\ ulp(x) = 1 \cdot 10^2 = 10^4\epsilon$$

- Erreurs d'ordre supérieur :

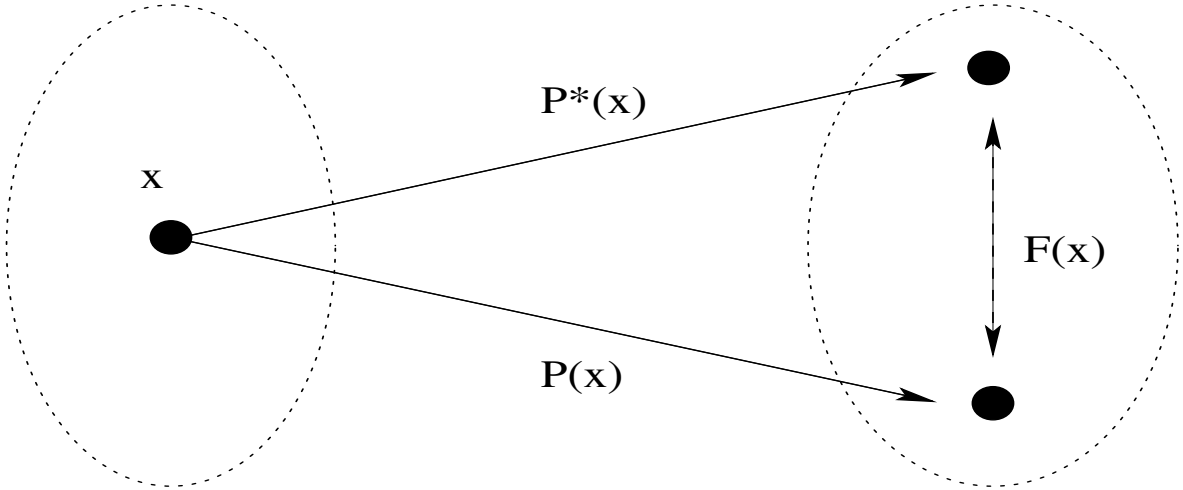
$$x_1 = f_1 + \epsilon_1,\ x_2 = f_2 + \epsilon_2, \quad x_1 \times x_2 = \underbrace{f_1 f_2}_{resultat} + \underbrace{f_1\epsilon_2 + f_2\epsilon_1}_{1er\ ordre} + \underbrace{\epsilon_1\epsilon_2}_{2d\ ordre}$$

- plus généralement, des erreurs d'ordre n peuvent apparaitre au cours d'un calcul.
- les erreurs d'ordre supérieur sont généralement négligeables par rapport à celles d'ordre 1 mais $\exists$ des exceptions

Erreur en avant

$F(x) = \text{dist}(p(x), p^*(x))$

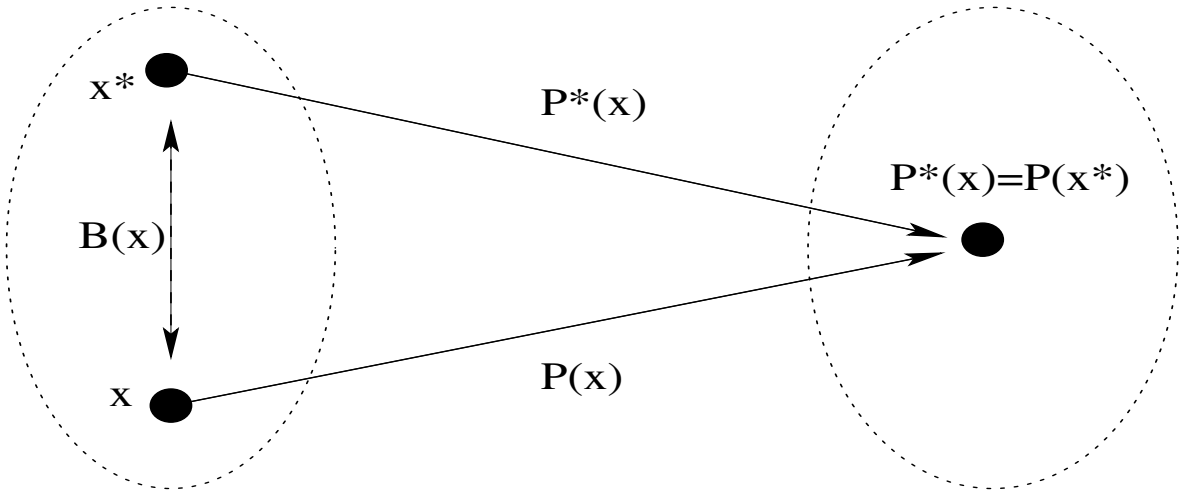
$$\begin{cases} p & : \text{ calcul exact} \\ p^* & : \text{ calcul approché} \end{cases}$$



Estime la précision de la solution en fonction de la précision d'une entrée particulière

Erreur en arrière

$$B(x) = \text{Inf}\left\{\text{dist}(x, x^*) \ : \ x^* = p^{-1}(p^*(x))\right\}$$



Détermine si une solution approchée est égale à la solution exacte d'un problème voisin

Références

- Goldberg, D., What Every Computer Scientist Should Know About Floating-Point Arithmetic, ACM Computing Surveys, 23 :1 pp5-48, 1991
- Knuth, D. The Art of Computer Programming - Seminumerical Algorithms, Chapter 4, ISBN 0-201-89684-2, Addison Wesley, troisième édition, 1997
- Handbook of Floating-Point Arithmetic, Jean-Michel Muller (coordinator), Birkhauser Boston, dec. 2009, ISBN : 978-0-8176-4704-9

Analyse d’erreur

Sémantiques des Nombres Flottants

1. Sémantique de référence $[\![.]\!]_{\mathbb{E}}$: flottant + erreur exacte
2. Sémantique pour les intervalles $[\![.]\!]_{\mathbb{I}}$
3. Sémantique pour l'arithmétique stochastique $[\![.]\!]_{\mathbb{S}}$
4. Sémantique pour la différentiation automatique $[\![.]\!]_{\mathbb{D}}$

$[\![.]\!]_{\mathbb{E}}$: flottant + erreur exacte

- $[\![.]\!]_{\mathbb{E}}$ sémantique de référence :
  - Théorique : utilise des réels, non implémentable
  - Utilisée pour comparer les propriétés calculées par les autres méthodes
- Principe : déterminer l'erreur exacte entre le résultat d'un calcul dans les flottants et dans les réels
  - e expression arithmétique,
  - $[\![e]\!]_{\mathbb{R}}$ résultat de l'évaluation dans les réels
  - $[\![e]\!]_{\mathbb{F}}$ résultat de l'évaluation dans les flottants
  - $[\![e]\!]_{\mathbb{E}}$ résultat flottant et écart entre $[\![.]\!]_{\mathbb{R}}$ et $[\![.]\!]_{\mathbb{F}}$

Utilisation de la Norme IEEE 754

- Arrondi des nombres réels (vers 0, $-\infty$, $+\infty$ et au plus près) :

$$\uparrow_\circ : \mathbb{R} \to \mathbb{F}$$

- Résultats des opérations : pour $\Diamond \in \{+, -, \times, \div, \sqrt{}\}$ on a :

$$x \Diamond_{\mathbb{F}} y = \uparrow_\circ (x \Diamond_{\mathbb{R}} y)$$

- Erreur d'arrondi :

$$\downarrow_\circ : \mathbb{R} \to \mathbb{R} \qquad \downarrow_\circ (r) = r - \uparrow_\circ (r)$$

Valeurs de $[\![.]\!]_{\mathbb{E}}$

- Valeurs de $[\![.]\!]_{\mathbb{E}}$:

$$v = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e, \quad f \in \mathbb{F},\ e \in \mathbb{R}$$

- Exemple : $\frac{1}{3}$ représenté par

$$\begin{aligned} v &= \uparrow_{\circ}\left(\tfrac{1}{3}\right)\vec{\varepsilon}_f + \left(\tfrac{1}{3} - \downarrow_{\circ}\left(\tfrac{1}{3}\right)\vec{\varepsilon}_e \\ &= 0.333333\vec{\varepsilon}_f + \left(\tfrac{1}{3} - 0.333333\right)\vec{\varepsilon}_e \end{aligned}$$

- Interprétation d'une constante d dans $[\![.]\!]_{\mathbb{E}}$:

$$[\![d]\!]_{\mathbb{E}} = \uparrow_{\circ}(d)\vec{\varepsilon}_f + \downarrow_{\circ}(d)\vec{\varepsilon}_e$$

Sémantique des opérations élémentaires dans $[\![.]\!]_{\mathbb{E}}$

$$x_1 = f_1\vec{\varepsilon}_f + e_1\vec{\varepsilon}_e \quad \text{et} \quad x_2 = f_2\vec{\varepsilon}_f + e_2\vec{\varepsilon}_e$$

$$[\![x_1 + x_2]\!]_{\mathbb{E}} = \uparrow_\circ (f_1 + f_2)\vec{\varepsilon}_f + [e_1 + e_2 + \downarrow_\circ (f_1 + f_2)]\,\vec{\varepsilon}_e$$

$$[\![x_1 - x_2]\!]_{\mathbb{E}} = \uparrow_\circ (f_1 - f_2)\vec{\varepsilon}_f + [e_1 - e_2 + \downarrow_\circ (f_1 - f_2)]\,\vec{\varepsilon}_e$$

$$[\![x_1 \times x_2]\!]_{\mathbb{E}} = \uparrow_\circ (f_1 \times f_2)\vec{\varepsilon}_f + [e_1 f_2 + e_2 f_1 + e_1 e_2 + \downarrow_\circ (f_1 \times f_2)]\,\vec{\varepsilon}_e$$

$$[\![\frac{1}{x_1}]\!]_{\mathbb{E}} = \uparrow_\circ \left(\frac{1}{f_1}\right)\vec{\varepsilon}_f + \left[\downarrow_\circ \left(\frac{1}{f_1}\right) + \sum_{n\geq 1}(-1)^n \frac{e_1^n}{f_1^{n+1}}\right]\vec{\varepsilon}_e$$

Cas de la division

$$\frac{1}{1+x} = \sum_{n\geq 0}(-1)^n x^n \text{ pour tout } x \text{ t.q. } -1 < x < 1$$

Nous avons :

$$\frac{1}{f+e} = \frac{1}{f} \times \frac{1}{1+\frac{e}{f}} = \frac{1}{f} \times \sum_{n\geq 0}(-1)^n \frac{e^n}{f^n}$$

et :

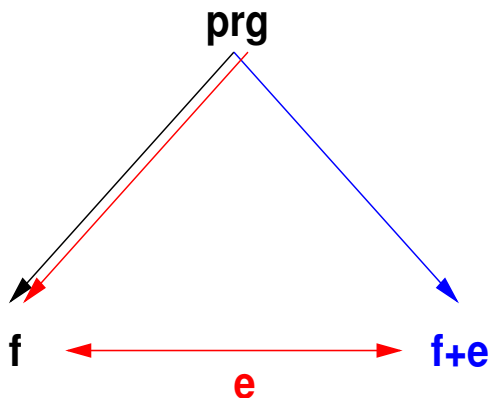
$$[\![\frac{1}{x}]\!]_{\mathbb{E}} = \uparrow_{\circ}\left(\frac{1}{f}\right)\vec{\varepsilon}_f + \left[\downarrow_{\circ}\left(\frac{1}{f}\right) + \sum_{n\geq 1}(-1)^n \frac{e^n}{f^{n+1}}\right]\vec{\varepsilon}_e$$

Valable pour $-1 < \frac{e}{f} < 1$ ou pour $|e| \leq |f|$, c.à.d. tant que l'erreur est inférieure au flottant.

Propriété de $[\![.]\!]_{\mathbb{E}}$

Propriété

Soit a une expression arithmétique. Si $[\![a]\!]_{\mathbb{E}} = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e$ *alors* $[\![a]\!]_{\mathbb{F}} = f$ *et* $[\![a]\!]_{\mathbb{R}} = f + e$.



[prg] F

[prg] R

[prg] E

$[\![.]\!]_{\mathbb{I}}$: les intervalles

- Interprétation d'une constante d :

$$[\![d]\!]_{\mathbb{I}} = [\uparrow_{-\infty}(d), \uparrow_{+\infty}(d)]$$

- Sémantique des opérations : $x_1 = [\underline{x_1}, \overline{x_1}]$ et $x_2 = [\underline{x_2}, \overline{x_2}] \in \mathbb{F} \times \mathbb{F}$; $i = [\underline{i}, \bar{i}] \in \mathbb{R} \times \mathbb{R}$ défini par $i = x_1 \Diamond x_2$.

$$[\![x_1 \Diamond x_2]\!]_{\mathbb{I}} = [\uparrow_{-\infty}(\underline{i}), \uparrow_{+\infty}(\bar{i})]$$

Remarques

- Les intervalles ont tendance à grossir rapidement
- Wrapping effect :

$$P(x) = x - x^2$$

  - Calcul direct : $P([0,1]) \subseteq [0,1] - [0,1] \times [0,1] = [-1,1]$
  - Par Horner : $P([0,1]) \subseteq [0,1]$
  - Exactement : $P([0,1]) = [0, \frac{1}{4}]$
- On utilise souvent des flottants de grande précision pour représenter les bornes (ex : MPFI)

Précision des bornes des intervalles

- Les intervalles sont utilisés pour encadrer le résultat réel d'un calcul
- Deux possibilités :
  - Bornes de même précision que celle utilisée pour le calcul :

$$\forall x \in \mathbb{R},\ x \in [\uparrow_{-\infty}(x), \uparrow_{+\infty}(x)] \quad \text{et} \quad \forall x \in \mathbb{R},\ \forall \circ,\ \uparrow_{\circ}(x) \in [\uparrow_{-\infty}(x), \uparrow_{+\infty}(x)$$

  - Bornes de précision p_1 supérieure à celle p_2 du calcul :

$$x \in [\uparrow_{-\infty}(x), \uparrow_{+\infty}(x)] \quad \text{mais } \exists x \in \mathbb{R} : \uparrow_{\circ}^{p_2}(x) \notin [\uparrow_{-\infty}^{p_1}(x), \uparrow_{+\infty}^{p_1}(x)]$$

Exemple

```
float x=1.0 ;
float y=1.0e-8 ;
for(int i=0 ;i<1e8 ;i++) {
    x=x-y ;
}
```

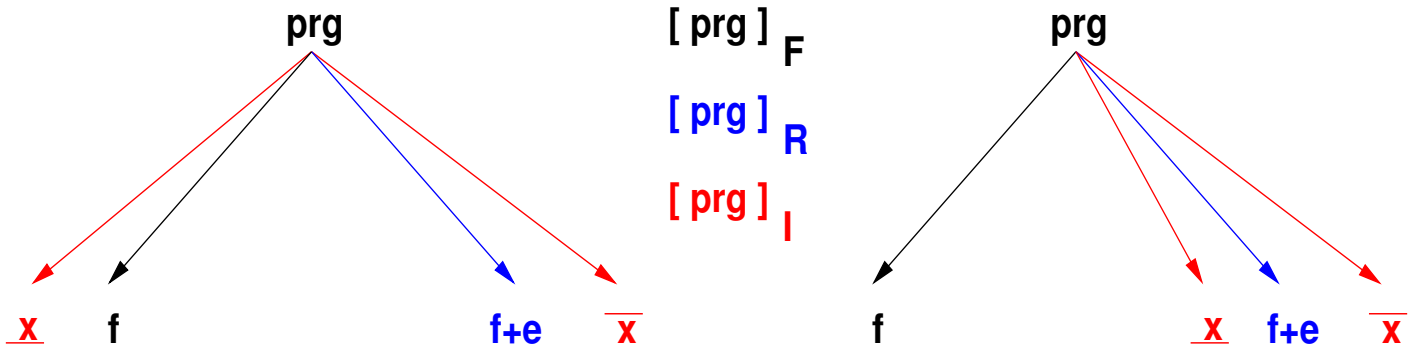
- Dans les réels : $[\![p]\!]_{\mathbb{R}} = 0$
- Dans les flottants : $[\![p]\!]_{\mathbb{F}} = 1$
- Dans la sémantique avec erreur exacte : $[\![p]\!]_{\mathbb{E}} = 1\vec{\varepsilon}_f - 1\vec{\varepsilon}_e$
- Dans les intervalles de flottants : $[\![p]\!]_{\mathbb{I}} \supseteq [0, 1]$
- Dans les intervalles grande précision : $[\![p]\!]_{\mathbb{I}} \subseteq [-\epsilon, \epsilon]$, $\epsilon > 0$ petit

Propriétés de $[\![.]\!]_{\mathbb{I}}$

Propriété

Soit a une expression arithmétique telle que $[\![a]\!]_{\mathbb{E}} = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e$ *et* $[\![a]\!]_{\mathbb{I}} = [\underline{x}, \overline{x}]$. *Alors :*

- $f + e \in [\underline{x}, \overline{x}]$.
- *De plus, si les bornes des intervalles sont des flottants IEEE,* $f \in [\underline{x}, \overline{x}]$.

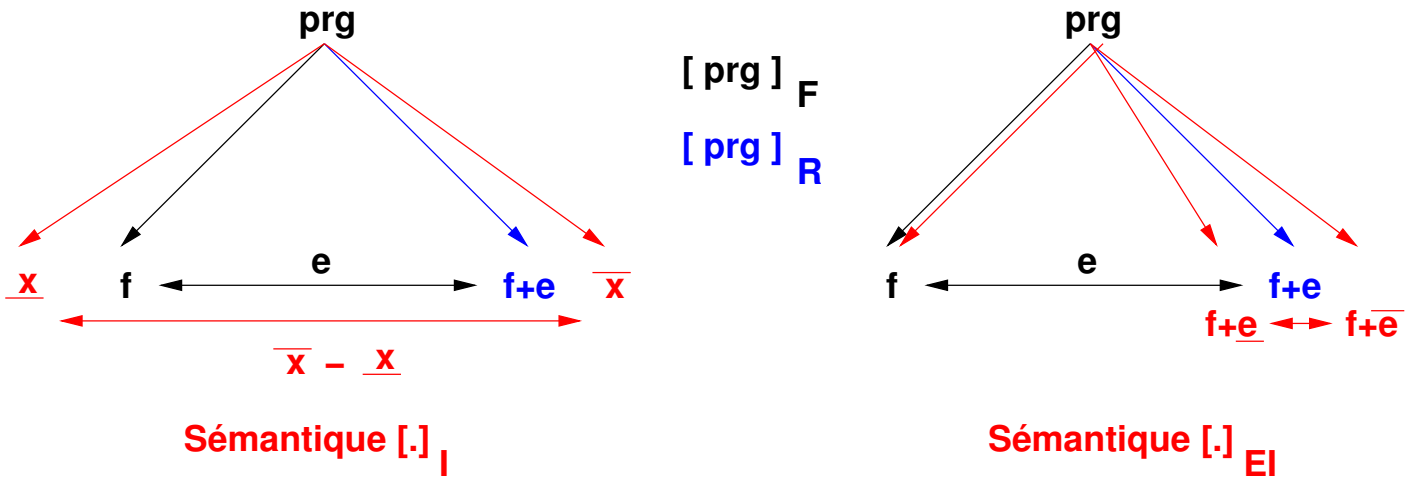


Intervalles de flottants

Intervalles grande précision

Remarques concernant $[\![.]\!]_{\mathbb{I}}$

- Si $[\![.]\!]_{\mathbb{I}}$ utilise des bornes flottantes, si $[\![a]\!]_{\mathbb{I}} = [\underline{x}, \overline{x}]$ alors $[\![a]\!]_{\mathbb{E}} = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e$ et $e \leq \overline{x} - \underline{x}$.
- $[\![.]\!]_{\mathbb{I}}$ donne un moyen d'implémenter $[\![.]\!]_{\mathbb{E}}$: nouvelle sémantique $[\![.]\!]_{\mathbb{EI}}$ dont les valeurs sont de la forme $f\vec{\varepsilon}_f + [\underline{e}, \overline{e}]\vec{\varepsilon}_e$.



$[\![.]\!]_{\mathbb{S}}$: arithmétique stochastique

- Principe :
  - Arrondir le résultat de chaque opération aléatoirement
  - Exécuter plusieurs fois le programme (de fa con synchrone)
  - sous certaines hypothèses, la moyenne des résulats donne le résultat réel du calcul.
- But : contrairement aux intervalles, prendre en compte le fait que les erreurs d'arrondi se compensent
- Méthode CESTAC, implémentation : logiciel CADNA

$[\![.]\!]_{\mathbb{S}}$: définition de la sémantique

- On introduit le mode d'arrondi aléatoire :

$$\uparrow_? (d) = \begin{cases} \text{soit } \uparrow_{-\infty} (d) \\ \text{ou } \uparrow_{+\infty} (d) \end{cases} \quad \text{avec probabilité } \frac{1}{2}$$

- Interprétation d'une constante d :

$$[\![d]\!]_{\mathbb{S}} = (\uparrow_? (d), \ldots, \uparrow_? (d)) \in \mathbb{F}^n$$

- Sémantique d'une opération élémentaire $\Diamond$:

$$[\![x \Diamond x']\!]_{\mathbb{S}} = (\uparrow_? (x_1 \Diamond x'_1), \ldots, \uparrow_? (x_n \Diamond x'_n))$$

Distribution

- Soit a expression telle que $[\![a]\!]_{\mathbb{S}} = x = (x_1, \ldots, x_n)$ et $[\![a]\!]_{\mathbb{E}} = f\vec{\varepsilon_f} + e\vec{\varepsilon_e}$.
- Notation : $x_{\mathbb{R}} = f + e$, $E(x) = \overline{x}$ espérance de x, variance $\sigma = E[(x - E(x))^2]$

- Hypothèses :
  1. Indépendance des erreurs d'arrondi commises au cours du calcul
  2. Les erreurs d'ordre supérieur sont négligeables
- Propriétés
  1. $\overline{x}$ est le résultat exact $x_{\mathbb{R}} = f + e$ du calcul, à probabilité près
  2. La distribution de x est quasi-gaussienne (distribution binomiale)

Nombre de tirages

- Utilisation du test de Student :
  - Estimation de l'espérance d'une gaussienne à partir d'un échantillon, à probabilité près
  - Utilisé pour estimer $\overline{x}$ à partir d'un échantillon $(x_1, \ldots, x_n)$:

$$\forall \beta \in [0,1],\ \exists \tau_\beta \in \mathbb{R}\ t.q.\ P(|\overline{x} - x_{\mathbb{R}}| \leq \frac{\tau_\beta \sigma}{\sqrt{n}}) = \beta$$

- $C_{x,f}$ nombre de chiffres communs à x et f

$$|f - x| \approx \left|\frac{f+x}{2}\right| .10^{C_{x,f}} \Leftrightarrow C_{x,f} \approx \log_{10} \left|\frac{x+f}{2(x-f)}\right|$$

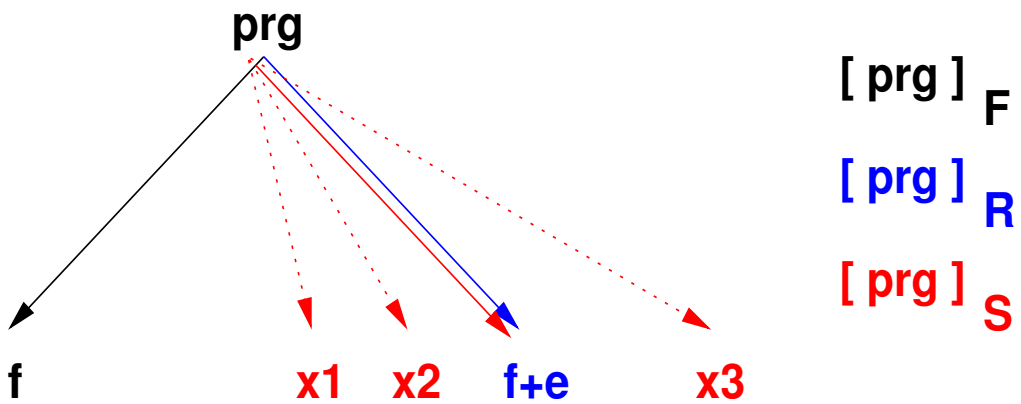
- Dans notre cas : $C_{x_{\mathbb{R}},\overline{x}} = \log_{10}\left(\frac{\sqrt{n}|\overline{x}|}{\sigma\tau_\beta}\right)$
- Pour $n = 3$ et $\beta = 0.95$, $\tau_\beta = 4.303$

Propriétés de $[\![.]\!]_{\mathbb{S}}$

$C_{x,y}$ nombre de chiffres communs à x et y.

Propriété

Soit a une expression arithmétique telle que $[\![a]\!]_{\mathbb{S}} = (x_1, \ldots, x_n)$ et $[\![a]\!]_{\mathbb{E}} = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e$. Alors, avec probabilité β, $\overline{x}$ et $f + e$ ont $C_{\overline{x},f+e}$ chiffres en communs



Remarques concernant $[\![.]\!]_{\mathbb{S}}$

- La correction de $[\![.]\!]_{\mathbb{S}}$ repose sur 2 hypothèses :
  - Les erreurs sont indépendantes les unes aux autres : souvent faux dans les boucles
  - Les erreurs du premier ordre dominent
- $[\![.]\!]_{\mathbb{S}}$ peut être utilisée pour implémenter $[\![.]\!]_{\mathbb{E}}$: nouvelle sémantique $[\![.]\!]_{\mathbb{ES}}$
  - Valeurs de $[\![.]\!]_{\mathbb{ES}}$: $f\vec{\varepsilon_f} + e\vec{\varepsilon_e}$ avec f flottant et e nombre stochastique

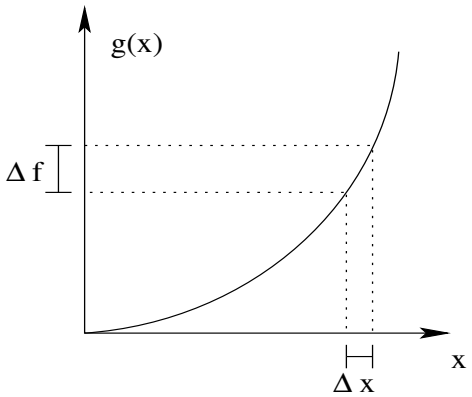
$[\![.]\!]_{\mathbb{D}}$: différentiation automatique

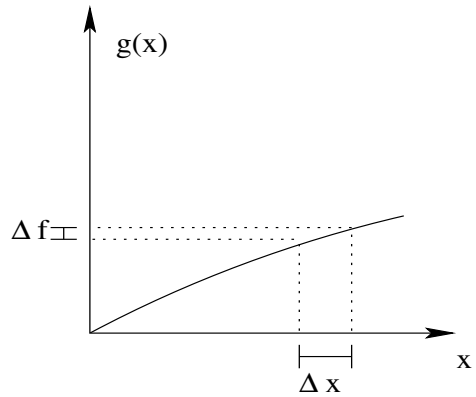
- Programme vu comme une fonction :

$$\begin{pmatrix} v_1 \\ \vdots \\ v_m \end{pmatrix} = \begin{pmatrix} g_1(d_1, \ldots, d_n) \\ \vdots \\ g_m(d_1, \ldots, d_n) \end{pmatrix}$$

- Calcul (numérique) des dérivées partielles
- Analyse de *sensibilité* aux variations des données : une faible variation des entrées implique-t-elle une variation importante des résultats ?

Différentiation automatique : principe





- Calcul des dérivées :

$$g'(x) \approx \frac{g(x+\Delta x) - g(x)}{\Delta x} \qquad \text{imprécis}$$

on utilisera : $(f \circ g)'(x) = \left[f(g(x))\right]' = g'(x) \times f'(g(x))$

$[\![.]\!]_{\mathbb{D}}$: sémantique

- Valeurs : $(n+1)$-uplet $(f, \delta_1, \ldots, \delta_n) \in \mathbb{F}^{n+1}$
- Initialement : $[\![d_i]\!]_{\mathbb{D}} = \big(d_i, \delta_1 = 0, \ldots, \delta_i = 1, \ldots, \delta_n = 0\big)$
- En cours de calcul : v résultat intermédiaire, $v = h_1(d_1, \ldots, d_n)$ t.q.

$$g((d_1, \ldots, d_n)) = h_2 \circ h_1(d_1, \ldots, d_n)$$

- v représente :

$$v = (f, \delta_1, \ldots, \delta_n) = \left(h_1(d_1, \ldots, d_n), \frac{\partial h_1}{\partial d_1}(d_1, \ldots, d_n), \ldots, \frac{\partial h_1}{\partial d_n}(d_1, \ldots, d_n)\right)$$

$[\![.]\!]_{\mathbb{D}}$: sémantique des opérations

$$[\![d_i]\!]_{\mathbb{D}} = (d_i, \delta_1 = 0, \ldots, \delta_i = 1, \ldots, \delta_n = 0)$$

$$v_1 = (f_1, \delta_1, \ldots, \delta_n) \quad \text{et} \quad v_2 = (f_2, \eta_1, \ldots, \eta_n)$$

$$[\![v_1 + v_2]\!]_{\mathbb{D}} = (f_1 + f_2, \delta_1 + \eta_1, \delta_2 + \eta_2, \ldots, \delta_n + \eta_n)$$

$$[\![v_1 - v_2]\!]_{\mathbb{D}} = (f_1 - f_2, \delta_1 - \eta_1, \delta_2 - \eta_2, \ldots, \delta_n - \eta_n)$$

$$[\![v_1 \times v_2]\!]_{\mathbb{D}} = (f_1 \times f_2, f_1\eta_1 + f_2\delta_1, f_1\eta_2 + f_2\delta_2, \ldots, f_1\eta_n + f_2\delta_n)$$

$$[\![\frac{x_1}{x_2}]\!]_{\mathbb{D}} = \left(\frac{v_1}{v_2}, \frac{f_2\delta_1 - f_1\eta_1}{f_2}, \frac{f_2\delta_2 - f_1\eta_2}{f_2}, \ldots, \frac{f_2\delta_n - f_1\eta_n}{f_2}\right)$$

Exemple

$g(x, y) = sin\frac{x}{y}$, sensibilité à x au vois. de (1.5 ;0.5) ?

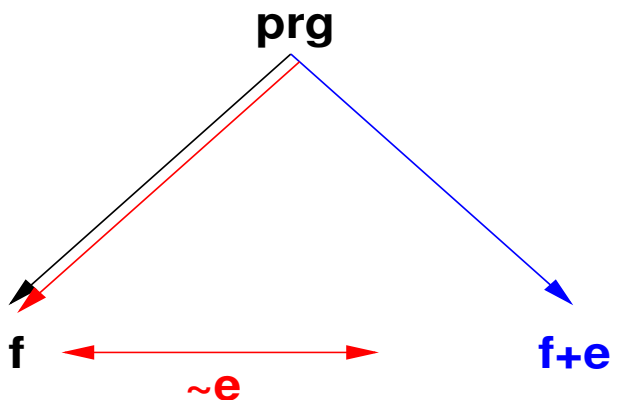
$$
\begin{array}{c|c}
v_0 = x = 1.5 & \dot{v}_0 = \dot{x} = 1 \\
v_1 = y = 0.5 & \dot{v}_1 = \dot{y} = 0 \\
v_2 = v_0 \div v_1 = 1.5 \div 0.5 = 3 & \dot{v}_2 = \frac{v_1 \dot{v}_0 - \dot{v}_1 v_0}{v_1^2} = 2 \\
v_3 = \sin v_2 = \sin 3 = 0.1411 & \dot{v}_3 = \dot{v}_2 \cos v_1 = 2 \times -0.99 = -1.98
\end{array}
$$

Propriétés de $[\![.]\!]_{\mathbb{D}}$

Propriété

Si $[\![g(d_1)]\!]_{\mathbb{E}} = x_r = f_r\vec{\varepsilon}_f + e_r\vec{\varepsilon}_e$ *alors par approximation linéaire*

$x_r \approx g(d_1)\vec{\varepsilon}_f + \left(e_{d_1} \times \frac{dg}{dd_1}(d_1)\right)\vec{\varepsilon}_e.$



[prg] F

[prg] R

[prg] D

Remarques concernant $[\![.]\!]_{\mathbb{D}}$

```
float g(float x,int n) {
   y=x ;
   for (int i=0 ;i<n ;i++) { y=y*x ; } ;
   return y ;
}
```

- Pour $x = f\vec{\varepsilon}_f + e\vec{\varepsilon}_e$ tel que $f < 1$ et $f + e > 1$, dans $\mathbb{F}$, $g(x) \to 0$ alors que dans $\mathbb{R}$, $g(x) = g(f + e) \to \infty$ quand $n \to \infty$
- Valeur retournée par $[\![.]\!]_{\mathbb{D}}$: $(f^{n+1}, (n+1)f^n)$. Pour $x = 0.95\vec{\varepsilon}_f + 0.1\vec{\varepsilon}_e$:

$$[\![g(x,n)]\!]_{\mathbb{E}} \to 0\vec{\varepsilon}_f + \infty\vec{\varepsilon}_e \qquad [\![g(x,n)]\!]_{\mathbb{D}} \to (0,0) \qquad \text{quand } n \to \infty$$

- Autre approximation : erreurs d'arrondi :

$$[\![x_1 + x_2]\!]_{\mathbb{E}} = \uparrow_\circ (f_1 + f_2)\vec{\varepsilon}_f + [e_1 + e_2 + \downarrow_\circ (f_1 + f_2)]\,\vec{\varepsilon}_e$$

Exemples

Essais réalisés en utilisant des implémentation des méthodes présentées dans cette section :

- MPFI : librairie d'intervalles multi-précision
- CADNA : implémentation de l'arithmétique stochastique
- ADOL-C : librairie de différentiation automatique
- Fluctuat : implémentation de $[\![.]\!]_{\mathbb{EI}}$

Exemple 1

- Calcul stable mais contenant des erreurs d'arrondi
- Inspiré de l'exemple du Patriot

```
float x = 1.0 ;
float y = 1.0e-8 ;
for (int i=0 ;i<1e8 ;i++) {
   x = x-y ;
}
```

Exemple 1 : résultats expérimentaux

	Result	Comment	Interpretation
$[\![.]\!]_{\mathbb{E}}$	x=$1.0\vec{\varepsilon}_f - 1.0\vec{\varepsilon}_e$	$[\![.]\!]_{\mathbb{E}}$ is the theoretical, non implementable semantics.	An absorption arises at each iteration. The float result is 1.0 and the error w.r.t. to the real solution exactly is -1.0.
MPFI	x=[-1.244141e1,1.000000]	x is initialized by `mpfi_init2(x,24)` to simulate the IEEE 754 Standard simple precision mode.	The real solution as well as the float solution belong to the given interval. The error may be as large as the interval width, i.e. 11.44141.
MPFI	x=[-4.11312855843084e-9,3.28835822230294e-9]	x is initialized by `mpfi_init_set_d` to obtain a highly accurate result.	The real solution is very close to zero. There is no unstability in this example.
CADNA	x=-0.3808, cestac= 4	x and y are declared as `single_st` numbers.	With high probability, the real solution is $x = -0.3808$ and the first four significant digits seem correct.
ADOL-C	$x = 1.0$, $\frac{\partial x}{\partial y} = -1.0e8$	Since ADOL only has double precision numbers, this test has been carried out using `adouble` numbers and $y = 1.0e-22$. Results have been transposed to our example.	The float result is 1.0 but the sensitivity to y is high. By linear approximation, the real solution is $1.0 + \frac{\partial x}{\partial y} \times \Delta y$ where Δy is the initial error on y.
Fluctuat	x=$1.0\vec{\varepsilon}_f + [-\infty$,-1.0e-8$]\vec{\varepsilon}_e$	No instrumentation of the code. Fluctuat does not unroll the loop (5 iterations are carried out).	Fluctuat detects that there is possibly (but not surely) a large error on the result.

Exemple 2

Calcul de la puissance n^{ime} u_n du nombre d'or par récurrence :

$$u_1 = \frac{\sqrt{5}-1}{2} \quad \text{et} \quad u_{n+2} = u_n - u_{n+1}$$

```
double x = 1.0 ;
double y = 0.618034 ;
for (i=0 ;i<=100 ;i++) {
    z=x ;
    x=y ;
    y=z-y ;
} ;
```

Exemple 2 : résultats expérimentaux

	Result	Comment	Interpretation
MPFI	y = [-9.37805732496113e14, -1.04330402763639e13]	y initialized with mpfi_interv_d.	The real solution belong to the given interval. The error may be as large as the interval width, i.e. approximatively 9.27e14. The computation is unstable.
CADNA	y=-0.474119386437716e+15 cestac=15	x, y, z have double_st type. y was initialized to the median value of the interval.	With high probability, the real solution is y≈-0.47e+15 and the first fifteen significant digits are correct. This computation seems stable.
ADOL-C	y=-4.74119e+14, *grad*(y)=-3.54225e+20	x, y, z have adouble type. y is initialized to the median value of the interval.	The gradient indicates that this computation is unstable.
Fluctuat	y=[-9.37805732496110e14,-1.04330402763639e13]$\vec{\varepsilon}_f$+[-25363.4,25363.4]$\vec{\varepsilon}_e$	An assertion is used to initialize y to [0.618034,0.618035]$\vec{\varepsilon}_f$ + 0$\vec{\varepsilon}_e$. Fluctuat is asked to unroll the loop.	The results belong to a large interval but the errors never are greater than 25363.4. If the initial error on y is null, the computation is unstable but roundoff errors are negligible.
Fluctuat	y=[-1.04330402763639e13,-1.04330402763639e13]$\vec{\varepsilon}_f$+[-9.27373e+14,-0.00149523]$\vec{\varepsilon}_e$	An assertion is used to initialize y to 0.618034$\vec{\varepsilon}_f$+[0,0,02]$\vec{\varepsilon}_e$. Fluctuat is asked to unroll the loop.	This program is very sensitive to the initial value of y.

Exemple 3

- Calcul de fonction non-linéaire
- Les erreurs d'ordre supérieur dominent

```
x = 1.0 ;
y = 0.99 ;
for (int i=0 ;i<1000 ;i++) {
   x = x*y ;
}
```

Exemple 3 : résultats expérimentaux

	Result	Comment	Interpretation
$[\![.]\!]_{\mathbb{E}}$	x= 0.43e-4$\vec{\varepsilon}_f - \omega\vec{\varepsilon}_e$	ω is tiny if initially y=0.99$\vec{\varepsilon}_f + \omega'\vec{\varepsilon}_e$ with $\omega' < 0.01$. ω is large otherwise.	This computation is sensitive to y.
MPFI	x=[4.31712474106544e-5,4.31712474106612e-5]	x and y initialized by mpfi_init_set_d.	The float solution is close to the real one since the interval width is small.
CADNA	x=0.43171247410657e-4, cestac =14	x and y are declared as double_st numbers.	With high probability, the real solution is $x \approx 0.43e-4$ and the first fourteen significant digits are correct.
ADOL-C	x=4.31712e-5, $\frac{\partial x}{\partial y}$=0.0436073	The experiment has been carried out using adouble numbers.	The derivative is small. This computation seems not much sensitive to y.
Fluctuat	x=4.31712474106578e-5$\vec{\varepsilon}_f$+[8.22526e-21,6.17629e-20]$\vec{\varepsilon}_e$	No instrumentation of the code. Fluctuat is asked to fully unroll the loop.	Assuming that initially y is exact, no numerical error arises in this execution.
Fluctuat	x=4.31712474106578e-5$\vec{\varepsilon}_f$+[-1.42622e-12,20959.2]$\vec{\varepsilon}_e$	An assertion states that initially $y = 0.99\vec{\varepsilon}_f + [0.0, 0.02]\vec{\varepsilon}_e$. Fluctuat is asked to fully unroll the loop. The tool states that the dominant error is a higher order error.	The sensitivity to y is detected.

Exemple 4

$$(S_1) : \begin{cases} 2x + y = \frac{5}{3} \\ x + 3y = \frac{5}{2} \end{cases} \qquad (S_2) : \begin{cases} x_{n+1} = \frac{5}{6} - \frac{1}{2} y_n \\ y_{n+1} = \frac{5}{6} - \frac{1}{3} x_n \end{cases}$$

$$(S_3) : \begin{cases} x_{n+1} = [0.80, 0.85] - [0.4, 0.6] y_n \\ y_{n+1} = [0.80, 0.85] - [0.30, 0.35] x_n \end{cases}$$

```
int i ;
double x1,y1 ;
double a = [0.8,0.85] ; double b = [0.4,0.6] ;
double c = [0.8,0.85] ; double d = [0.3,0.35] ;
double x2 = [2.0,3.0] ; double y2 = [2.0,3.0] ;
for(i=0 ;i<1000 ;i++) {
   x1 = x2 ; y1 = y2 ;
   x2 = a-b*y1 ;
   y2 = c-d*x1 ;
}
```

Exemple 4 : résultats expérimentaux

	Result	Comment	Interpretation
MPFI	x=[3.536585365853 65e-1,6.16279069767442e-1], y=[5.843023255813 95e-1,7.43902439024391e-1]	x and y initialized by mpfi_interv_d.	The real solution as well as the float solution belong to the given intervals. The errors never are larger than the interval widths, i.e. about 0.26 for x and 0.15 for y.
CADNA	x=0.49253731343283, cestac=15 ; y=0.664925373134328, cestac=15	a, b, c, d, x2 and y2 are initialized to the median values of the intervals using the double_st type.	The large number of common digits indicates that the program, executed with the chosen parameters, is stable, with high probability.
ADOL-C	x=0.492537, *grad*(x)=0.0971263, y=0.664925, *grad*(y)=0.475897	a, b, c, d, x2 and y2 are initialized to the median values of the intervals using the adouble type.	The gradients indicate that this computation is stable in the neighbour of the chosen parameters.
Fluctuat	x=[3.536585365853 66e-1,6.16279069767442e-1]$\vec{\varepsilon}_f$+[-1.58101e-16,1.58101e-16]$\vec{\varepsilon}_e$ y=[5.843023255813 95e-1,7.43902439024390e-1]$\vec{\varepsilon}_f$+[-1.24724e-16,1.24724e-16]$\vec{\varepsilon}_e$	Assertions are used to initialize the identifiers. Fluctuat is asked to unroll the loop.	Fluctuat states that the errors on x and y never are larger than, approximatively, $1.0e-16$ for any execution.

Références

- Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics, by Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter, Springer-Verlag, London, 2001, ISBN 1-85233-219-0.
- Fabienne Jézéquel, Jean Marie Chesneaux : CADNA : a library for estimating round-off error propagation. Computer Physics Communications 178(12) : 933-955 (2008)
- Griewank, A., Evaluating Derivatives, Principles and Techniques of Algorithmic Differentiation, SIAM (Publisher), ISBN 0-89871-451-6, 2000