Arithmétique à virgule flottante

Pablo de Oliveira (pablo.oliveira@uvsq.fr)

2022-2023

M1 Calcul Haute Performance Simulation, Calcul Numérique

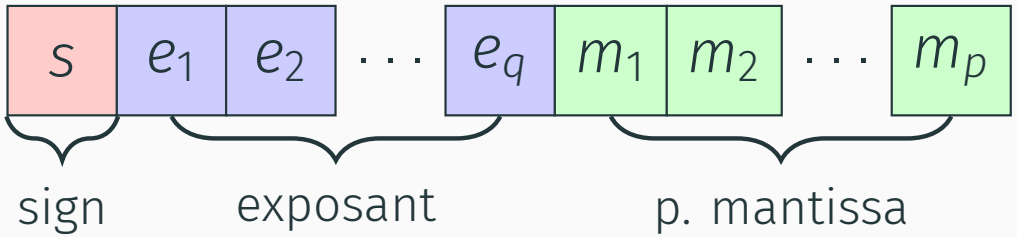
Sommaire

Standard IEEE-754

Représentation des nombres flottants

IEEE-754 définit une représentation flottante standardisée.

$$f = s \times 2^e \times m$$



Exemple

$$\begin{aligned}(1.1001 \times 2^0)_2 =& (1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4})_{10} \\ =& (1 + 0.5 + 0.125)_{10} = 1.625_{10}\end{aligned}$$

$$(1.0001 \times 2^{10})_2 = (1 \times 2^{10} + 1 \times 2^6)_{10} = (1088.0)_{10}$$

Standard IEEE-754

Formats classiques : binary32 (float), binary64 (double)

Pour un double:

- 11 bits pour l'exposant, $e \in [-1022; 1023]$
- 52 bits de pseudo-mantisse (epsilon machine $\epsilon = 2^{-52}$)

Pseudo-mantisse car le premier bit est implicite

- 1 pour les normaux
- 0 pour les dénormaux

Encodage de l'exposant

L'exposant est codé avec biais, il faut soustraire 1023 ($= 2^{q-1}$) à la valeur binaire $E = e_1 \dots e_q$

- pour l'exposant 2^1, on stockera $E = 1024$ en binaire
- pour l'exposant le plus petit 2^{-1022}, on stockera $E = 1$ en binaire
- pour l'exposant le plus grand 2^{1023}, on stockera $E = 2046$ en binaire

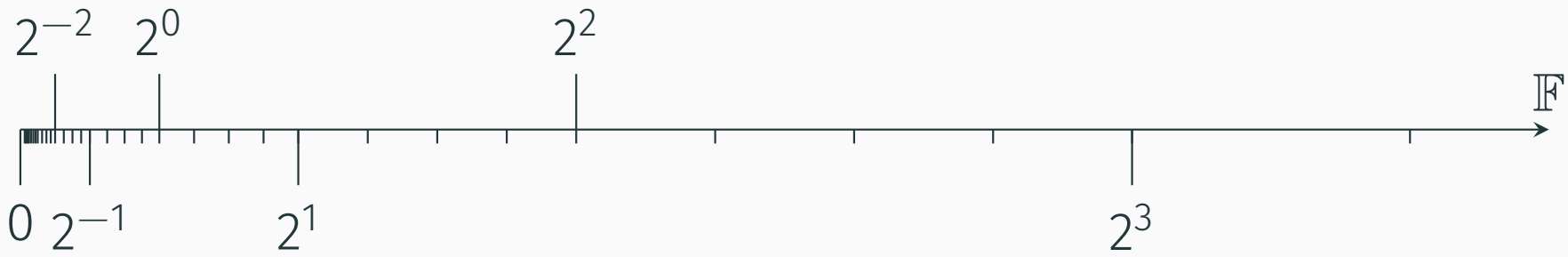
Valeur spéciales

- $E = 2047$ représentent $+\infty$, $-\infty$, NaN (en fonction de s et m)
- $E = 0$ et $m = 0$ représentent +0 et -0
- $E = 0$ et $m \neq 0$ représente un dénormal

Underflow et dénormaux

- Le plus petit double normal (en valeur absolue): $m = 1.0 \times 2^{-1022}$
- $\frac{m}{4}$ est non représentable car son exposant est trop petit; c'est un *underflow*.
- Retourner la valeur 0 à la place ? Pour ne pas perdre complètement la précision on utilise des dénormaux.
- On encode $\frac{m}{4} = 0.01 \times 2^{-1022}$. Pour les dénormaux le bit implicite est remplacé par un zéro. Ici on perds deux bits de précision dans la mantisse.

Répartition sur la droite des réels



Plus on se rapproche de zéro plus les flottants sont proches entre eux. Ici on a utilisé une précision $p = 2$. Donc 4 valeurs possibles entre deux puissances de deux (1.00, 1.01, 1.10, 1.11).

Arrondis

Si le résultat d'un calcul x n'est pas représentable, il est arrondi.

- Mode au plus près: la valeur la plus proche est choisie (en cas d'équidistance, par défaut la mantisse paire est choisie).
- Vers $\pm\infty$ ou vers 0: on arrondit vers $+\infty$, $-\infty$ ou 0.

La norme IEEE-754 garantit l'arrondi correct pour + - $\times$ / $\sqrt{}$

Lors d'un arrondi on commet une erreur maximale d' 1/2 ulp (unit in the last place). Soit pour un exposant 2^0, on commet une erreur maximale de $\frac{\epsilon}{2} = u = 2^{-53}$.

Analyse d’Erreurs

Modèle standard (Higham)

$$fl(x \circ y) = (x \circ y)(1+\delta) \qquad \text{avec } |\delta| \leqslant u = 2^{-53}$$

Le terme $(1+\delta)$ capture l'erreur commise.

Erreurs numériques: Représentation

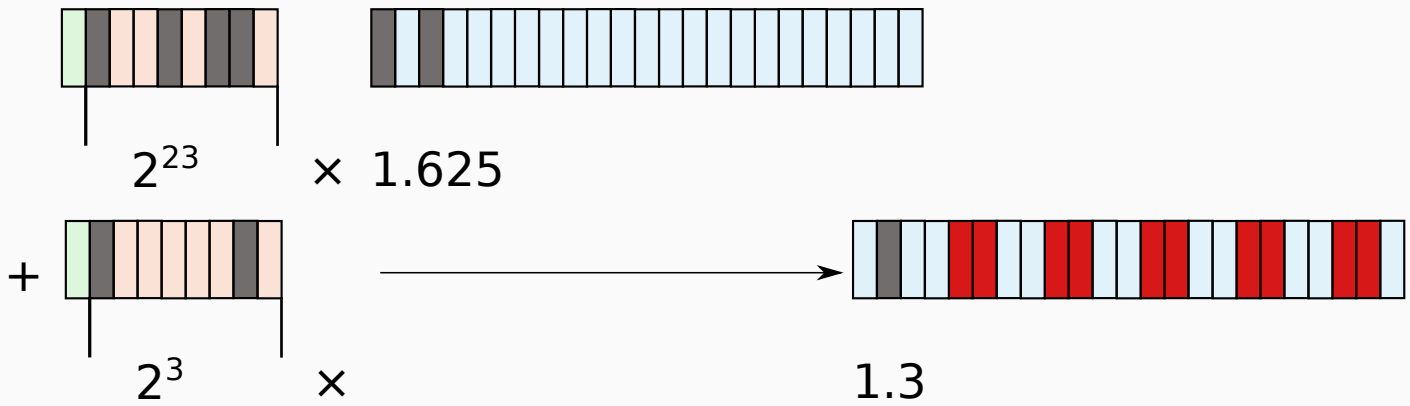
- Exemple: $0.1_{10} = 0.0001100110011\ldots_2$ a une mantisse infinie.
- La mantisse est tronquée en double précision.

Bug du missile Patriot (1991)

- Utilisation d'un registre de 24 bits pour multiplier 0.1 par l'horloge interne du système (pas de 0.1s).
- Après 100 heures, erreur d'approximation de 0.34s, soit 500m à la vitesse du missile Scud iraqien.
- Non interception du missile Scud, 34 soldats américains morts.

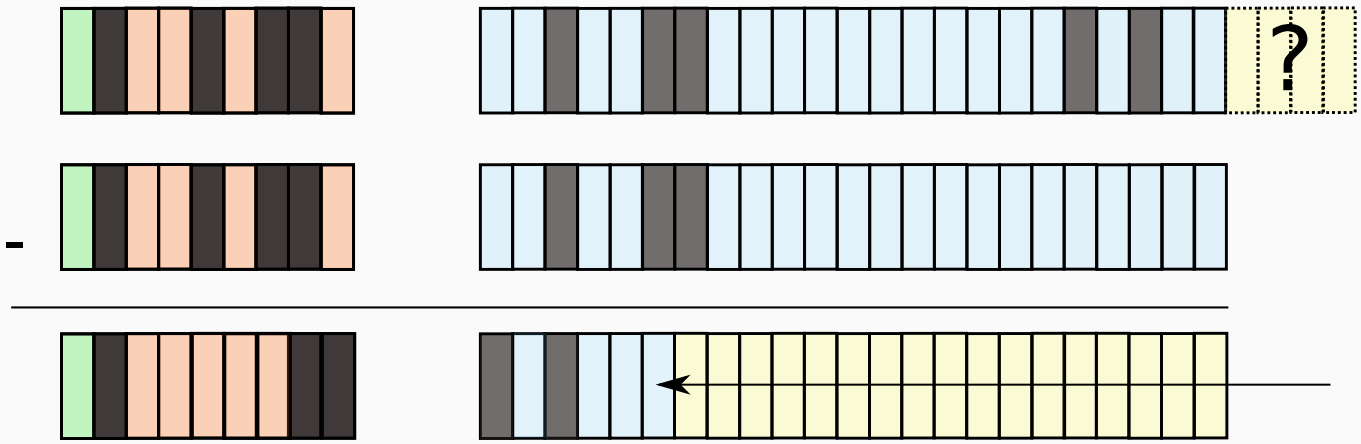
Erreurs numériques: Absorption

- Lors d'une addition ou soustraction, on renormalise le résultat.
- Les chiffres les plus à droite de la mantisse peuvent être perdus.



Erreurs numériques: Cancellation catastrophique

- Lorsque l'on soustrait deux valeurs proches, une partie de la mantisse s'annule (en anglais on parle de *cancellation*)
  - $9633812.0 - 9633792.0 = 20.000000$



- On complète la mantisse avec des zéros.
- Si les deux opérandes contenaient des erreurs sur les derniers bits (erreur d'arrondi sur les opérations précédentes), c'est une *cancellation catastrophique*.
- En effet, **l'erreur est promue en début de mantisse**.

Perte d'associativité

- En raison des erreurs d'arrondi, absorption, cancellation, l'arithmétique IEEE-754 **n'est pas associative**.

$$a + (b + c) \neq (a + b) + c$$

- L'ordre des opérations est donc important.

Sources classiques d’erreur dans les codes de calcul

- Sommations: produits scalaires, moyennes, réductions.
- Accumulation d’erreurs sur le temps: intégration méthodes explicites.
- Calcul de gradient sur des quantités proches.
- Non reproductibilité:
  - Parallélisme (change l’ordre des opérations)
  - Vectorisation et optimisations agressives
  - Branchements instables

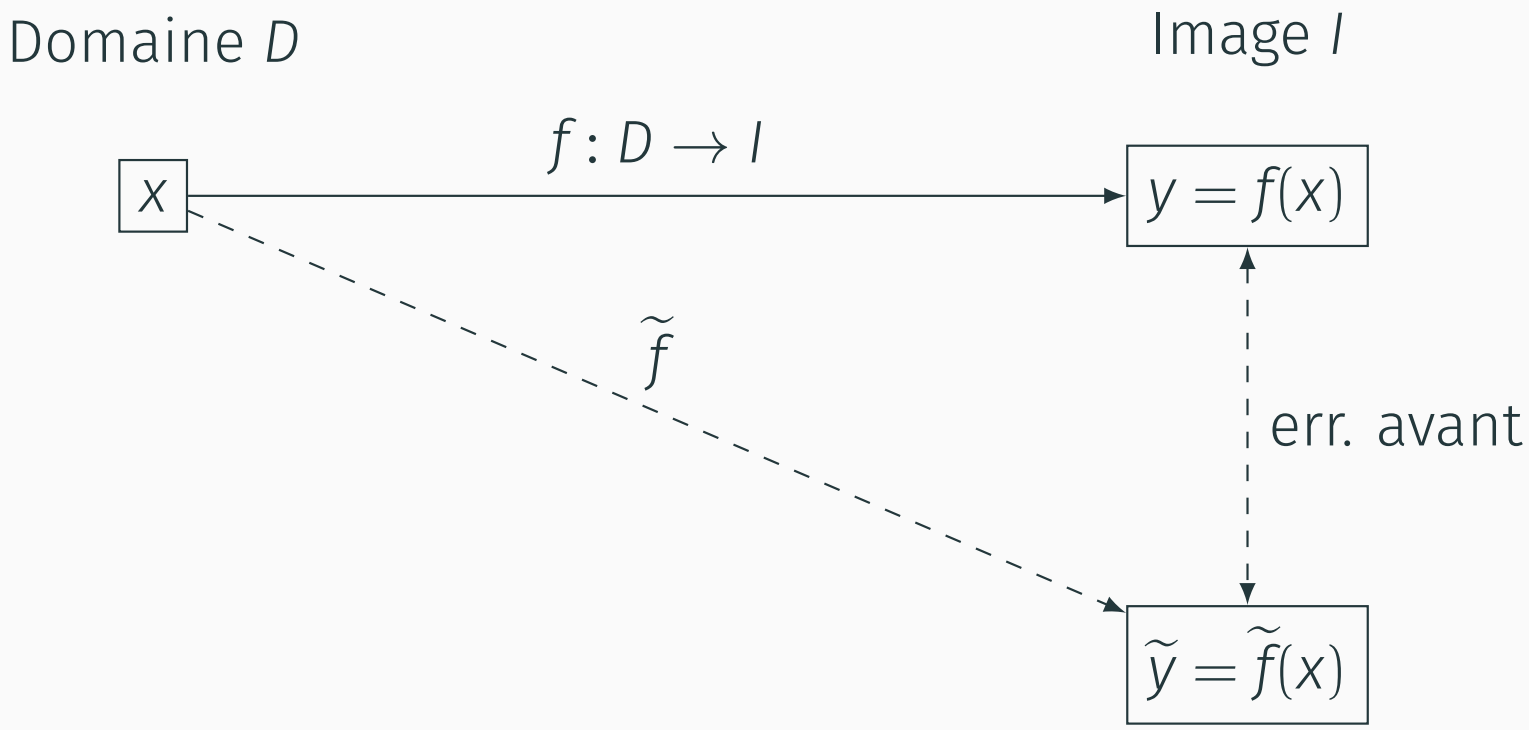
Quelques techniques pour atténuer les problèmes

- Réécrire une expression pour éviter des cancellations catastrophiques ou absorptions.

$$\frac{\sqrt{x^2+1}-1}{x} = \frac{x}{\sqrt{x^2+1}+1} \qquad \text{cancellation pour } x \to 0$$

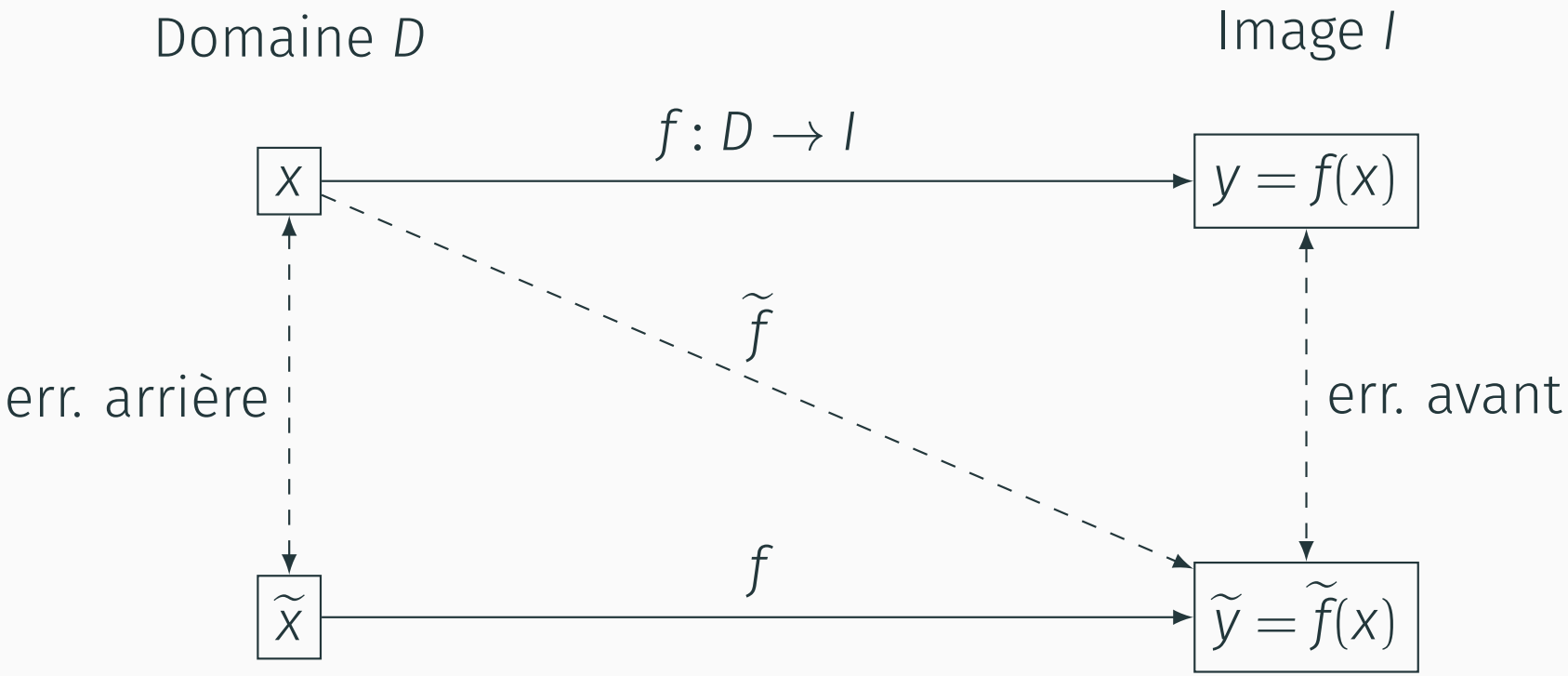
- Remplacer une formule par une approximation qui se comporte correctement.
- Utiliser les bons facteurs d'échelle pour éviter les dépassements.
- Augmenter le précision.
- Utiliser des algorithmes compensés.

Analyse d'erreur en avant et en arrière



Erreur avant $= \widetilde{y} - y$

Analyse d'erreur en avant et en arrière



- Erreur arrière: on voit $\widetilde{y}$ comme l'image d'une entrée perturbée.

$$\widetilde{y} = f(\widetilde{x}) = f(x + \underbrace{\delta x}_{\text{err. arrière}})$$

Étude de la somme

Exemple de la somme naïve

```
sum = 0
for i in range(n):
    sum += x[i]
```

$$f(\mathbf{x}) = S_n = \sum_{i=1}^{n} x_i$$

Erreur en avant (Wilkinson, Higham)

$$\widetilde{S}_k = (\widetilde{S}_{k-1} + x_k)(1+\delta_k) \qquad |\delta_k| \leqslant u$$

$$\widetilde{S}_n = \sum_{i=1}^{n} x_i \prod_{k=i}^{n} (1+\delta_k) \qquad \text{récursivement avec } \delta_1 = 0$$

$$\left| \prod_{k=1}^{n} (1+\delta_k) \right| \leqslant 1 + \sum_{k=1}^{n} |\delta_k| + O(u^2) \leqslant 1 + n.u + O(u^2)$$

$$\left| S_n - \widetilde{S}_n \right| \leqslant nu \sum_{i=1}^{n} |x_i| + O(u^2)$$

Erreur relative et Conditionnement de la somme

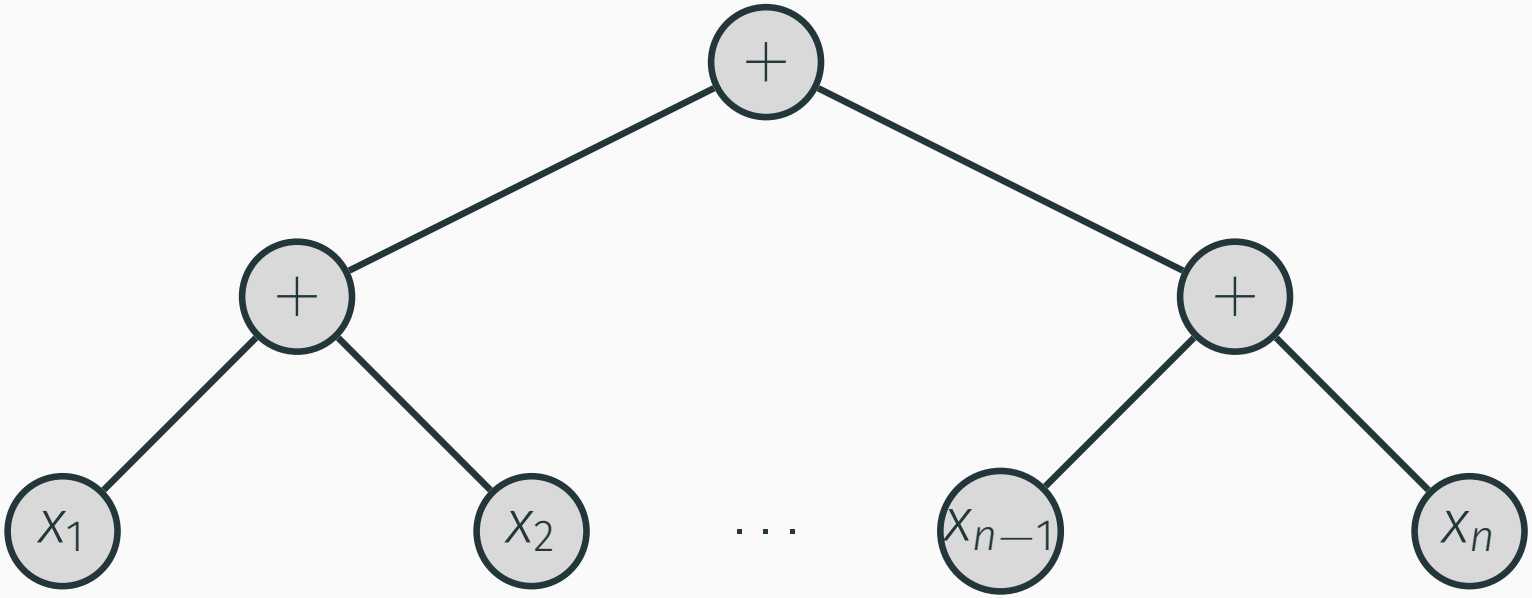
$$\left|S_n - \widetilde{S}_n\right| \leqslant nu \sum_{i=1}^{n} |x_i| + O(u^2)$$

$$\frac{\left|S_n - \widetilde{S}_n\right|}{|S_n|} \leqslant nu \underbrace{\left[\frac{\sum_{i=1}^{n} |x_i|}{\left|\sum_{i=1}^{n} x_i\right|}\right]}_{\textit{Conditionnement}} + O(u^2)$$

- Si tous les termes sont de même signe, le conditionnement de la somme est 1. L'erreur numérique augmente linéairement avec le nombre d'opérations.
- En pratique, souvent les erreurs tendent à se compenser et l'augmentation de l'erreur est en $O(\sqrt{n})$.

Pairwise summation

Plutôt que de sommer les termes en série, il est possible de les sommer en arbre.



- Erreur majorée par $O(log(n))$ et en pratique $O(\sqrt{log(n)})$
- D'autres algorithmes encore plus précis (Somme de Kahan)

Arithmétique stochastique

Arithmétique Stochastique (Stott Parker)

Problème: il est souvent difficile de faire une analyse formelle d'erreur pour un algorithme compliqué. Méthode empirique de mesure d'erreur ?

$$fl(x \circ y) = (x \circ y)(1 + \delta)$$

- On remplace δ par une variable aléatoire.
- On simule la distribution des erreurs arithmétiques d'un programme à l'aide de plusieurs tirages stochastiques.
- On choisit δ comme un bruit uniforme de magnitude u.

Arithmétique Stochastique (Stott Parker)

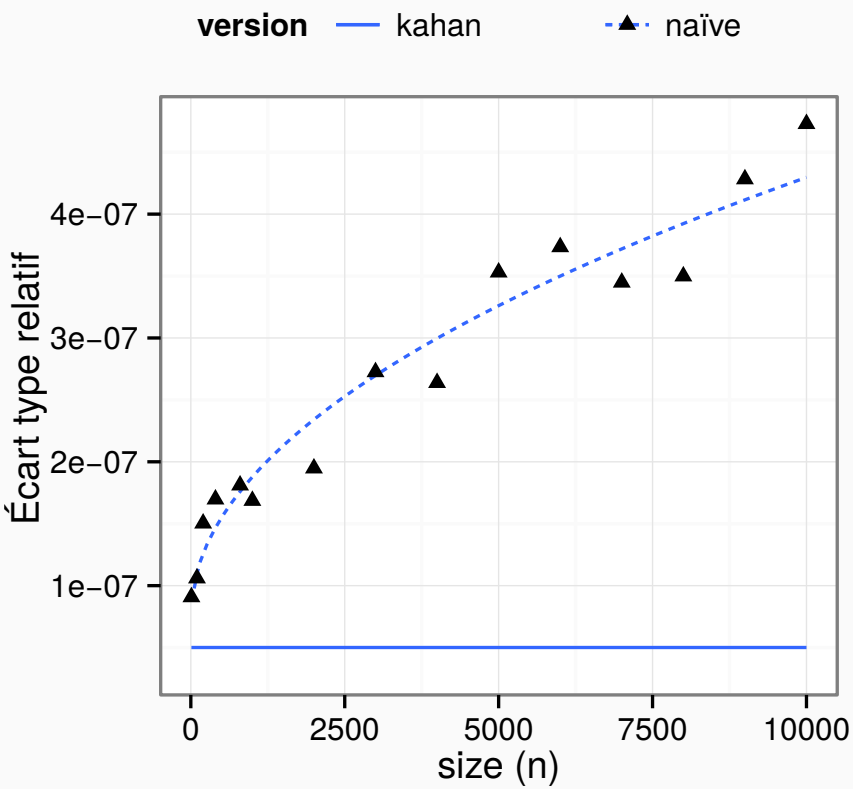


Figure 1: Erreur relative ($\frac{\sigma}{\mu}$). L'erreur pour la version naïve évolue en $O(\sqrt{n})$; la méthode compensée de Kahan est résistante au bruit numérique. Résultats obtenus avec le logiciel `https://github.com/verificarlo/verificarlo`

Références

- GAO report – Patriot Missile Defense, `https://www-users.cse.umn.edu/~arnold/disasters/GAO-IMTEC-92-96.pdf`
- The accuracy of floating-point summations, Nicholas Higham, 1993.