

L'implantation d'une fonction correctement arrondie

ou : CRLibm for dummies

Présentation pour le projet ANR TaMaDi

Christoph Quirin Lauter

LIP6 - PEQUAN - UPMC Paris VI - CNRS

Lyon, 28 octobre 2010



Généralités sur l'arrondi correct et notations

Généralités sur l'arrondi correct et notations

L'implantation d'une fonction

Défis de l'arrondi correct

Conclusions et extensions

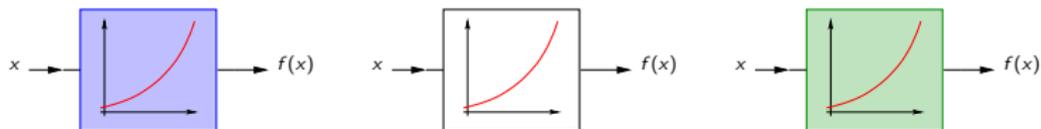
Fonctions mathématiques sur ordinateur

- Réalisation de fonctions mathématiques sur ordinateur
 - Programmes calculant la valeur de $f(x)$ pour un x donné



Fonctions mathématiques sur ordinateur

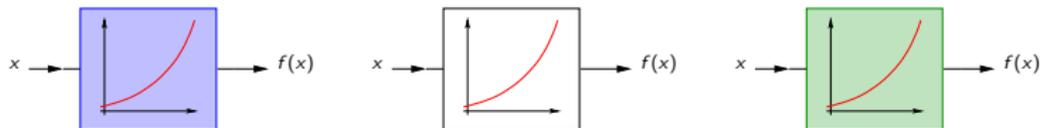
- Réalisation de fonctions mathématiques sur ordinateur
 - Programmes calculant la valeur de $f(x)$ pour un x donné



- Différentes réalisations d'une fonction f

Fonctions mathématiques sur ordinateur

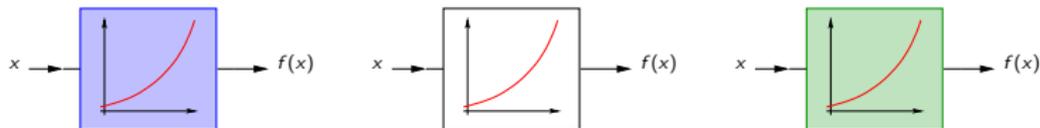
- Réalisation de fonctions mathématiques sur ordinateur
 - Programmes calculant la valeur de $f(x)$ pour un x donné



- Différentes réalisations d'une fonction f
- Les résultats du calcul doivent être les mêmes partout.
 - Des résultats déterministes
 - Des ordinateurs qui se contrôlent l'un l'autre
 - Le montant à payer et celui à percevoir qui sont les mêmes

Fonctions mathématiques sur ordinateur

- Réalisation de fonctions mathématiques sur ordinateur
 - Programmes calculant la valeur de $f(x)$ pour un x donné



- Différentes réalisations d'une fonction f
- Les résultats du calcul doivent être les mêmes partout.
 - Des résultats déterministes
 - Des ordinateurs qui se contrôlent l'un l'autre
 - Le montant à payer et celui à percevoir qui sont les mêmes
- \Rightarrow L'arrondi correct des fonctions mathématiques

L'arrondi

- Fonctions mathématiques réelles transcendentes :
Nombre infini de chiffres dans le développement de $f(x)$:
p.ex. $\exp(1) = 2.718281828459045235360287471 \dots$

L'arrondi

- Fonctions mathématiques réelles transcendentes :
Nombre infini de chiffres dans le développement de $f(x)$:

p.ex. $\exp(1) = 2.718281828459045235360287471 \dots$

- Mémoires finies dans nos ordinateurs
Solution : n'écrire que le début du développement

p.ex. $\exp(1) \approx 2.7182818284590459$

L'arrondi

- Fonctions mathématiques réelles transcendentes :
Nombre infini de chiffres dans le développement de $f(x)$:

$$\text{p.ex. } \exp(1) = 2.718281828459045235360287471 \dots$$

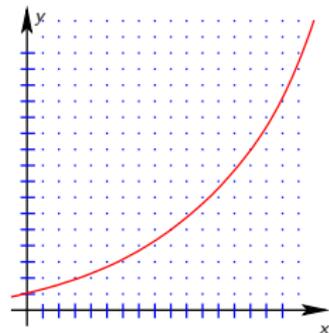
- Mémoires finies dans nos ordinateurs
Solution : n'écrire que le début du développement

$$\text{p.ex. } \exp(1) \approx 2.7182818284590459$$

- Il s'agit là d'un arrondi

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

où \mathbb{F} est une représentation des réels



La virgule flottante

- La *virgule flottante* est une représentation des réels

La virgule flottante

- La *virgule flottante* est une représentation des réels
- Il s'agit d'un système semi-logarithmique
 - Un exposant E donne l'ordre de grandeur.
 - Un significatif m indique les chiffres début du développement.

La virgule flottante

- La **virgule flottante** est une représentation des réels
- Il s'agit d'un système semi-logarithmique
 - Un exposant E donne l'ordre de grandeur.
 - Un significand m indique les chiffres début du développement.
- Exemples :

$$\begin{aligned}105.415 &= 10^3 \cdot 1.05415 = 10^E \cdot m \\0.000000124565 &= 10^{-7} \cdot 1.24565 = 10^E \cdot m \\62 &= 2^5 \cdot 1.11110_2 = 2^E \cdot m\end{aligned}$$

La virgule flottante

- La **virgule flottante** est une représentation des réels
- Il s'agit d'un système semi-logarithmique
 - Un exposant E donne l'ordre de grandeur.
 - Un significand m indique les chiffres début du développement.
- Exemples :

$$\begin{aligned}105.415 &= 10^3 \cdot 1.05415 = 10^E \cdot m \\0.000000124565 &= 10^{-7} \cdot 1.24565 = 10^E \cdot m \\62 &= 2^5 \cdot 1.11110_2 = 2^E \cdot m\end{aligned}$$

- La représentation se fait dans une base β ; ici $\beta = 2$.
- Le nombre de chiffres indiqués est la précision k .

La virgule flottante

- La **virgule flottante** est une représentation des réels
- Il s'agit d'un système semi-logarithmique
 - Un exposant E donne l'ordre de grandeur.
 - Un significand m indique les chiffres début du développement.
- Exemples :

$$\begin{aligned}105.415 &= 10^3 \cdot 1.05415 = 10^E \cdot m \\0.000000124565 &= 10^{-7} \cdot 1.24565 = 10^E \cdot m \\62 &= 2^5 \cdot 1.11110_2 = 2^E \cdot m\end{aligned}$$

- La représentation se fait dans une base β ; ici $\beta = 2$.
- Le nombre de chiffres indiqués est la précision k .
- L'ensemble des nombres flottants en précision k se note \mathbb{F}_k .

La norme IEEE 754

- Calcul en virgule flottante régi par la norme IEEE 754-2008
- Plusieurs formats \mathbb{F}_k spécifiés

Précision double \mathbb{F}_{53}

\pm	exposant	signifiant
	11 bits	52 + 1 bits

La norme IEEE 754

- Calcul en virgule flottante régi par la norme IEEE 754-2008
- Plusieurs formats \mathbb{F}_k spécifiés

Précision double \mathbb{F}_{53}



- Un arrondi IEEE 754 \circ_k est une application $\mathbb{R} \rightarrow \mathbb{F}_k$:

$$x \mapsto \circ_k(x)$$

La norme IEEE 754

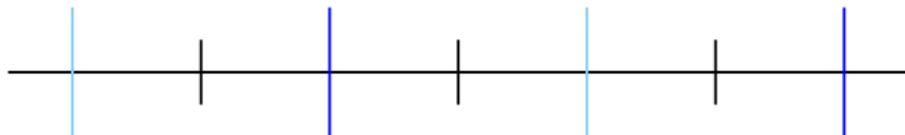
- Calcul en virgule flottante régi par la norme IEEE 754-2008
- Plusieurs formats \mathbb{F}_k spécifiés

Précision double \mathbb{F}_{53}



- Un arrondi IEEE 754 \circ_k est une application $\mathbb{R} \rightarrow \mathbb{F}_k$:

$$x \mapsto \circ_k(x)$$



La norme IEEE 754

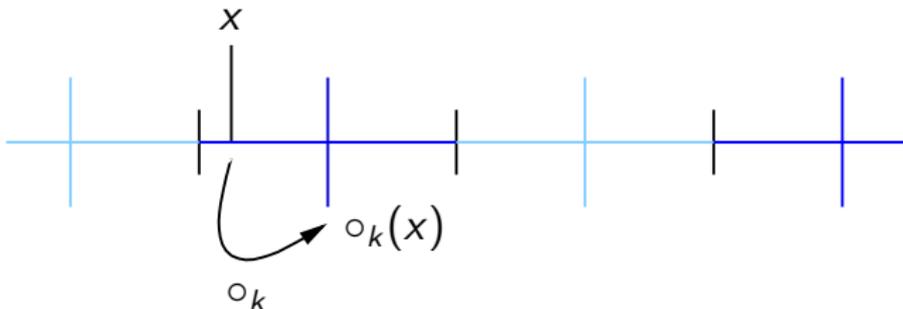
- Calcul en virgule flottante régi par la norme IEEE 754-2008
- Plusieurs formats \mathbb{F}_k spécifiés

Précision double \mathbb{F}_{53}



- Un arrondi IEEE 754 \circ_k est une application $\mathbb{R} \rightarrow \mathbb{F}_k$:

$$x \mapsto \circ_k(x)$$



La norme IEEE 754

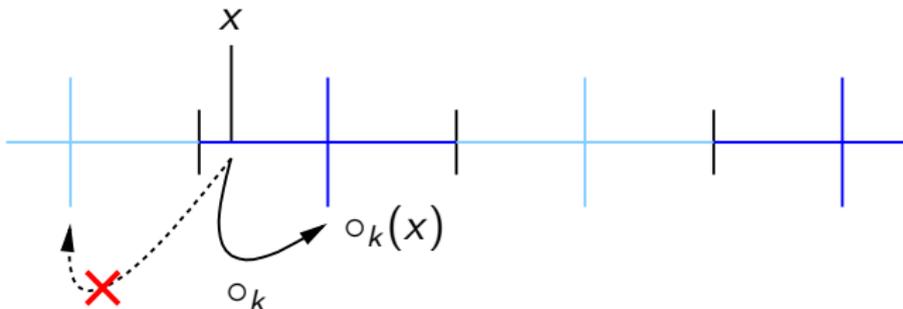
- Calcul en virgule flottante régi par la norme IEEE 754-2008
- Plusieurs formats \mathbb{F}_k spécifiés

Précision double \mathbb{F}_{53}

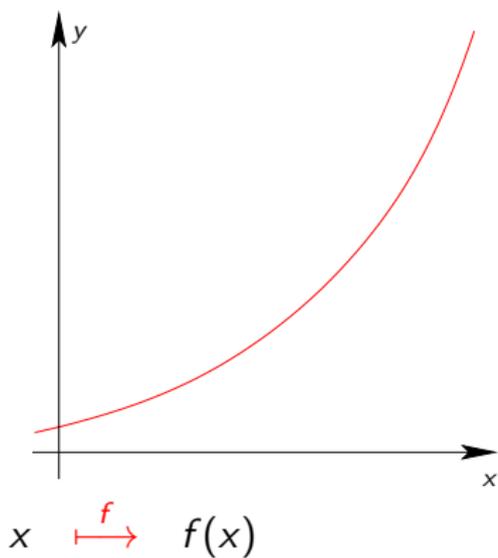


- Un arrondi IEEE 754 \circ_k est une application $\mathbb{R} \rightarrow \mathbb{F}_k$:

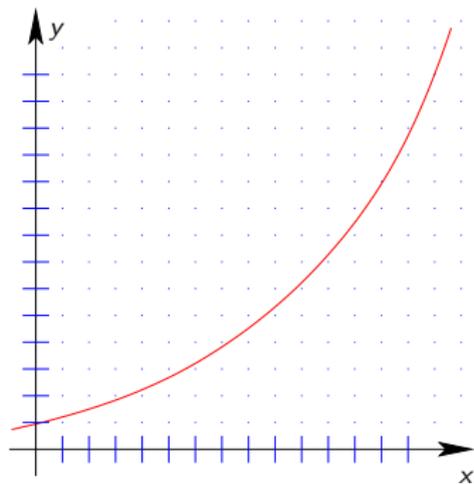
$$x \mapsto \circ_k(x)$$



L'arrondi correct d'une fonction f

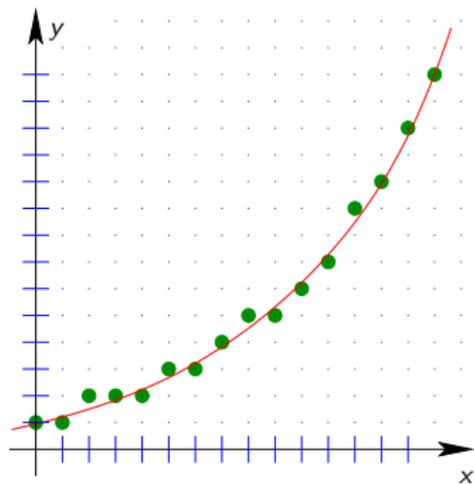


L'arrondi correct d'une fonction f



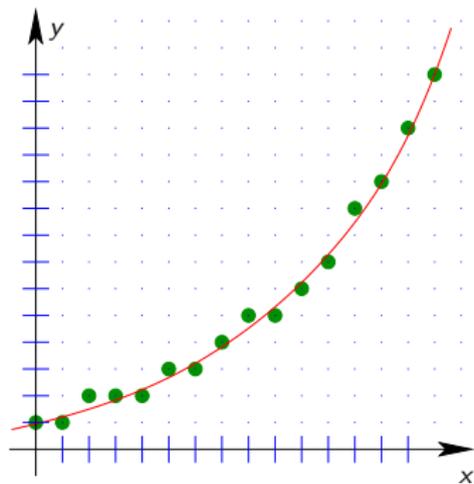
$$\forall x \in \mathbb{F}_k, \quad x \xrightarrow{f} f(x) \xrightarrow{\circ_k}$$

L'arrondi correct d'une fonction f



$$\forall x \in \mathbb{F}_k, \quad x \xrightarrow{f} f(x) \xrightarrow{\circ_k} \circ_k(f(x))$$

L'arrondi correct d'une fonction f



$$\forall x \in \mathbb{F}_k, \quad x \xrightarrow{f} f(x) \xrightarrow{\circ_k} \circ_k(f(x))$$

Définition

Soit $f : \mathbb{R}^n \rightarrow \mathbb{R}$ une fonction. Soit $\circ_k : \mathbb{R} \rightarrow \mathbb{F}_k$ un arrondi.
La fonction $F : \mathbb{F}_k^n \rightarrow \mathbb{F}_k$ est l'arrondi correct de f ssi

$$\forall x \in \mathbb{F}_k^n, \quad F(x) = \circ_k(f(x))$$

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

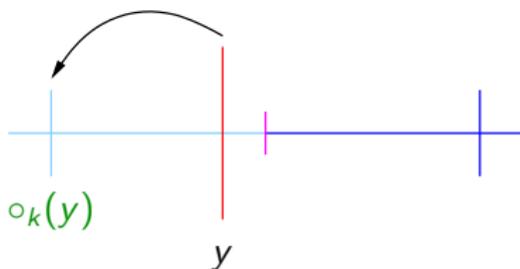


- La valeur $y = f(x)$ ne peut être calculée exactement.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

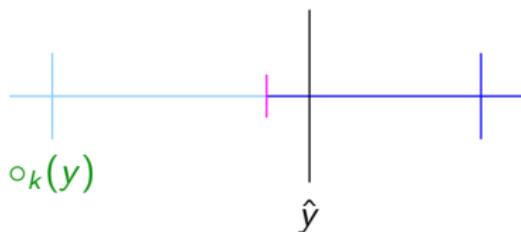


- La valeur $y = f(x)$ ne peut être calculée exactement.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

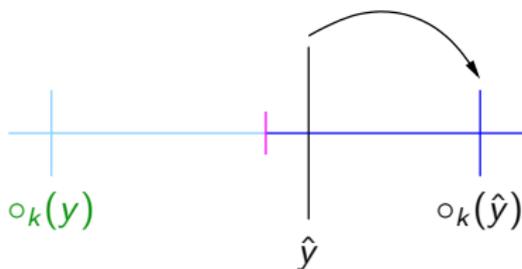


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

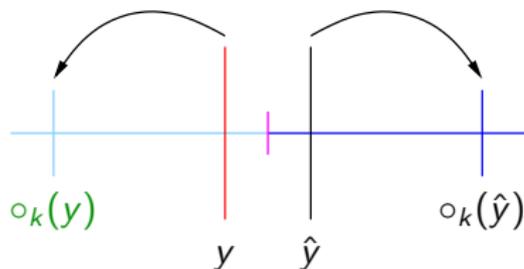


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

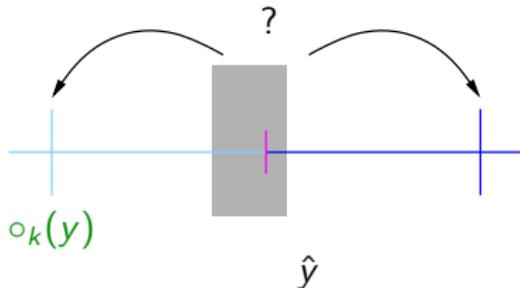


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

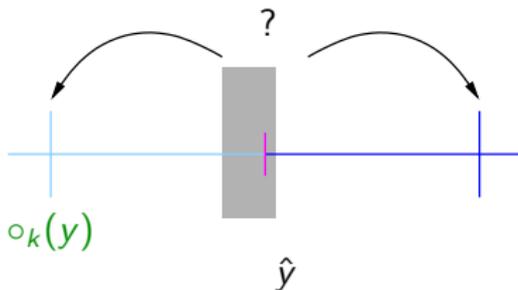


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

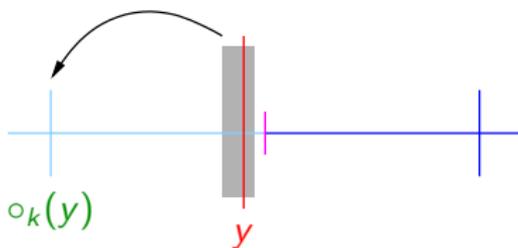


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.

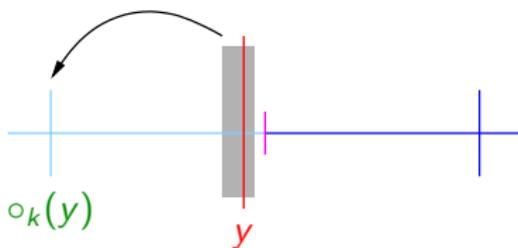


- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.

Voir assez net à l'endroit décisif

Arrondi correct $\circ_k(f(x))$:

- Combinaison de deux phénomènes
 - La valeur de $f(x)$, transcendante, ne peut être qu'approchée.
 - L'arrondi \circ_k change brusquement aux frontières d'arrondi.



- La valeur $y = f(x)$ ne peut être calculée exactement.
- Une approximation $\hat{y} = f(x) \cdot (1 + \varepsilon)$ peut être calculée.
- La précision $\bar{\varepsilon}$ nécessaire au pire cas est inconnue.
- C'est le dilemme du fabricant de tables.

L'implantation d'une fonction

Généralités sur l'arrondi correct et notations

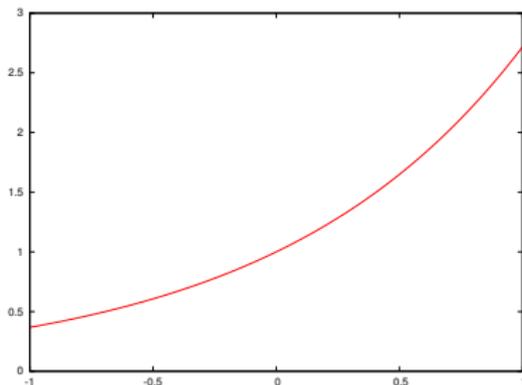
L'implantation d'une fonction

Défis de l'arrondi correct

Conclusions et extensions

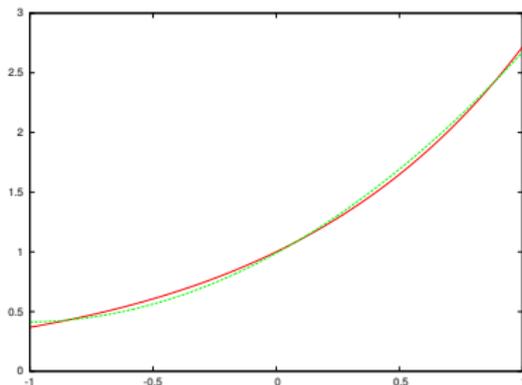
Approximation polynomiale

- Comment donc implanter p.ex. $f = \exp$...
- ... sur un domaine restreint d'abord, p.ex. $I = [-1; 1]$?



Approximation polynomiale

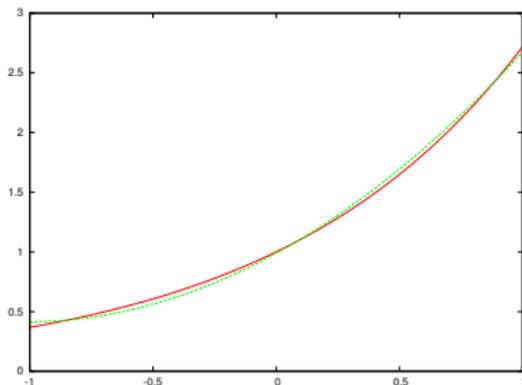
- Comment donc implanter p.ex. $f = \exp$...
- ... sur un domaine restreint d'abord, p.ex. $I = [-1; 1]$?



- Solution : remplacer f par un **polynôme d'approximation** p

Approximation polynomiale

- Comment donc implanter p.ex. $f = \exp$...
- ... sur un domaine restreint d'abord, p.ex. $I = [-1; 1]$?



- Solution : remplacer f par un **polynôme d'approximation p**
- Types de polynômes :
 - Développement de Taylor
 - Polynômes **minimisant l'erreur maximale** sur le domaine
 - **Polynômes quasi-minimax à coefficients flottants** (Chevillard)

Pourquoi des polynômes ?

- Pourquoi des polynômes et non d'autres approximations ?
 - Pourquoi pas de **fractions continues** (tronquées) ?
 - Pourquoi pas d'approximation avec d'autres fonctions de base ?
 - Algorithmes CORDIC etc ?

Pourquoi des polynômes ?

- Pourquoi des polynômes et non d'autres approximations ?
 - Pourquoi pas de **fractions continues** (tronquées) ?
 - Pourquoi pas d'approximation avec d'autres fonctions de base ?
 - Algorithmes CORDIC etc ?
- Une réponse (presque) purement technologique
 - **Matériel très rapide pour +, × et FMA**
 - Performance bien moindre pour la division (19 cycles sur i5/i7)
 - Possibilité de calculer l'erreur pour + et × (et /)...
... mais pas pour sin et log

Pourquoi des polynômes ?

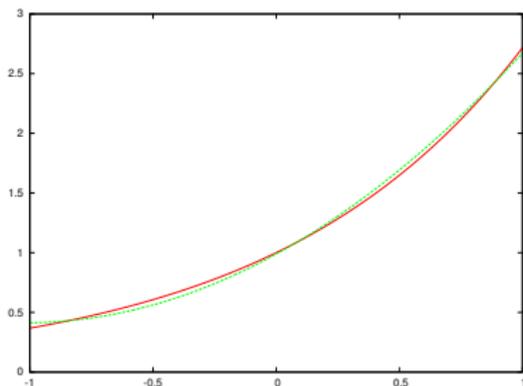
- Pourquoi des polynômes et non d'autres approximations ?
 - Pourquoi pas de **fractions continues** (tronquées) ?
 - Pourquoi pas d'approximation avec d'autres fonctions de base ?
 - Algorithmes CORDIC etc ?
- Une réponse (presque) purement technologique
 - **Matériel très rapide pour +, × et FMA**
 - Performance bien moindre pour la division (19 cycles sur i5/i7)
 - Possibilité de calculer l'erreur pour + et × (et /)...
... mais pas pour sin et log
- Une réponse pas catégorique
 - Certaines fonctions **s'approchent mal par des polynômes** : asin
 - CRLibm, c'est un logiciel ; autre chose sur FPGA ou ASIC
 - La chaîne d'outil n'est pas aussi riche pour d'autres approches

Nécessité de la réduction d'argument

- Nécessité d'implanter f sur tout le domaine des flottants
- Pour la double, $f = \exp$ est définie sur $I = [-744.5; 709]$

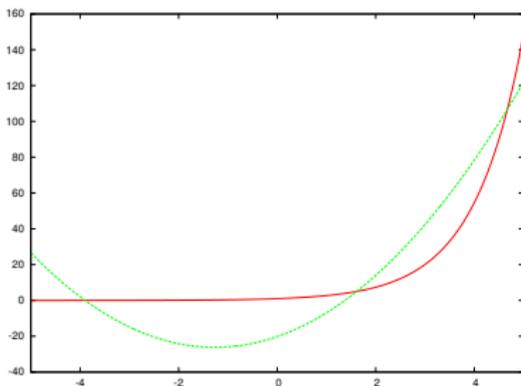
Nécessité de la réduction d'argument

- Nécessité d'implanter f sur tout le domaine des flottants
- Pour la double, $f = \exp$ est définie sur $I = [-744.5; 709]$
- L'approximation polynomiale seule ne suffit pas :



Nécessité de la réduction d'argument

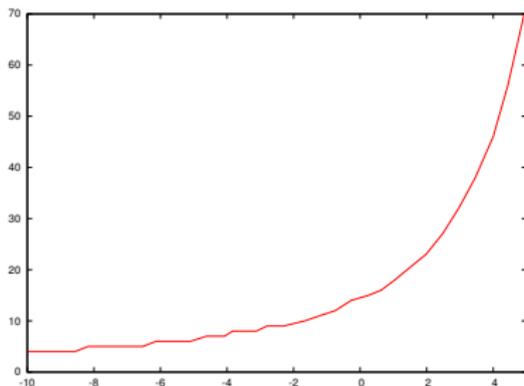
- Nécessité d'implanter f sur tout le domaine des flottants
- Pour la double, $f = \exp$ est définie sur $I = [-744.5; 709]$
- L'approximation polynomiale seule ne suffit pas :



- Quand le domaine est large, l'erreur explose pour un degré

Nécessité de la réduction d'argument

- Nécessité d'implanter f sur tout le domaine des flottants
- Pour la double, $f = \exp$ est définie sur $I = [-744.5; 709]$
- L'approximation polynomiale seule ne suffit pas :



- Quand le domaine est large, l'erreur explose pour un degré
- ou bien : il faut compenser par un degré immense

Réduction et tabulation

- La réduction d'argument ramène tout à un petit domaine
 - Propriétés algébriques de la fonction
 - ▶ Périodicité : \sin
 - ▶ Autosimilarité : \exp
 - ▶ Symétrie : asin
 - Caractère semi-logarithmique du format flottant
 - ▶ Vu de loin, `convertToInteger` et `exp` se ressemblent
 - Si rien d'autre : **découpage** du domaine

Réduction et tabulation

- La réduction d'argument ramène tout à un petit domaine
 - Propriétés algébriques de la fonction
 - ▶ Périodicité : sin
 - ▶ Autosimilarité : exp
 - ▶ Symétrie : asin
 - Caractère semi-logarithmique du format flottant
 - ▶ Vu de loin, convertToInteger et exp se ressemblent
 - Si rien d'autre : **découpage** du domaine
- Exemple pour exp :

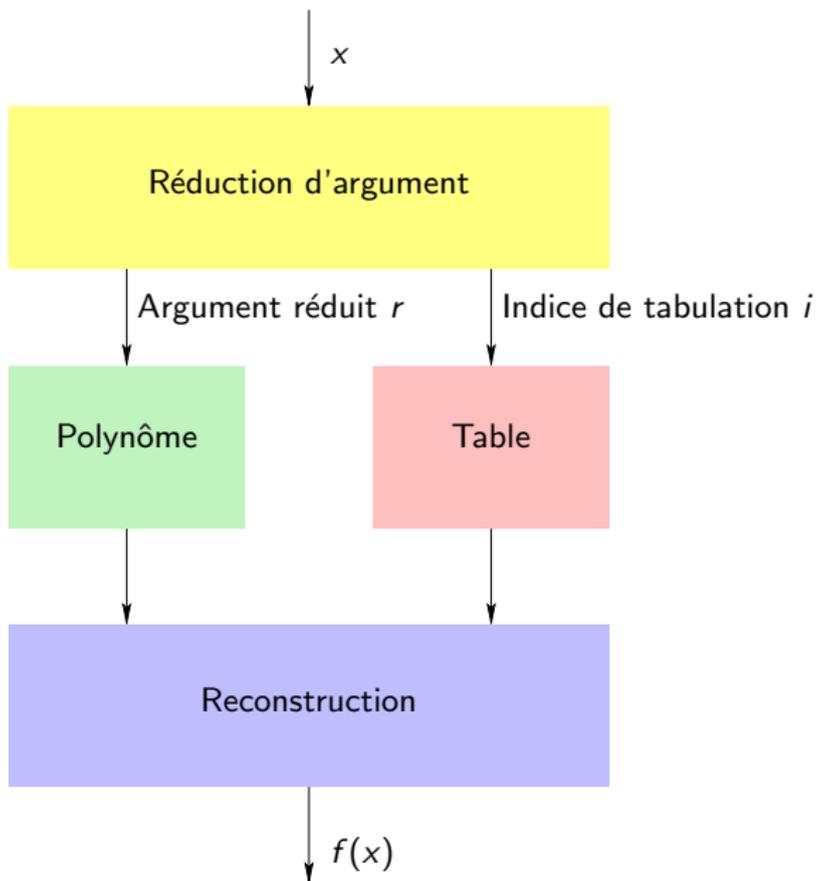
$$e^x = 2^{\frac{x}{\log 2}} = 2^{\lfloor \frac{x}{\log 2} \rfloor} \cdot 2^{\frac{x}{\log 2} - \lfloor \frac{x}{\log 2} \rfloor} = 2^E \cdot e^{x - E \log 2} = 2^E \cdot e^r$$

- La réduction d'argument ramène tout à un petit domaine
 - Propriétés algébriques de la fonction
 - ▶ Périodicité : sin
 - ▶ Autosimilarité : exp
 - ▶ Symétrie : asin
 - Caractère semi-logarithmique du format flottant
 - ▶ Vu de loin, convertToInteger et exp se ressemblent
 - Si rien d'autre : **découpage** du domaine
- Exemple pour exp :

$$e^x = 2^{\frac{x}{\log 2}} = 2^{\lfloor \frac{x}{\log 2} \rfloor} \cdot 2^{\frac{x}{\log 2} - \lfloor \frac{x}{\log 2} \rfloor} = 2^E \cdot e^{x - E \log 2} = 2^E \cdot e^r$$

- La **réduction** est souvent couplée à une **tabulation**
 - Tabulation : **précalcul d'une fonction g à des endroits discrets**
 - Réduction : le polynôme p doit être valide seulement entre ces endroits discrets

Schéma de calcul d'une fonction



Une exponentielle jouet

```
// A very crude "toy" implementation of exp(x)
//
// About 45 bits of accuracy. No checks for NaN, Inf whatsoever.
//
double Exp(double x) {
    double z, n, t, r, P, tbl, y;
    uint32_t E, idx, N;
    doubleCaster shiftedN, twoE;

    // Argument reduction
    z = x * TWO_4.RCP_LN_2;           // z = x * 2^4 * 1/ln(2)
    shiftedN.d = z + TWO_52.P_51;     // shiftedN.d = nearestint(z) + 2^52 + 2^51
    n = shiftedN.d - TWO_52.P_51;     // n = nearestint(z) as double
    N = shiftedN.i[LO];               // N = nearestint(z) as integer
    E = N >> 4;                       // E = floor(2^-4 * N)
    idx = N & 0x0f;                   // idx = N - E * 2^4
    t = n * TWO_M_4.LN_2;             // t = n * 2^-4 * ln(2)
    r = x - t;                        // r = x - t

    // Polynomial approximation      p(r) approximates exp(r)
    P = c0 + r * (c1 + r * (c2 + r * (c3 + r * (c4 + r * c5))));

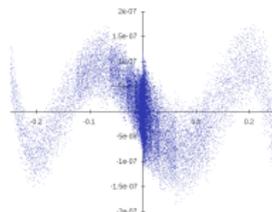
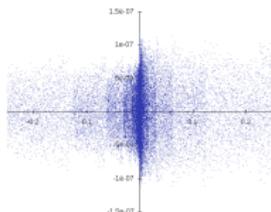
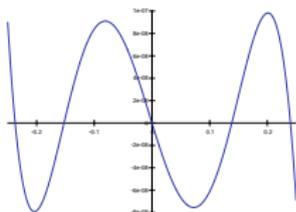
    // Table access
    tbl = table[idx];                 // tbl = 2^(2^-4 * idx)

    // Reconstruction
    twoE.i[HI] = (E + 1023) << 20;
    twoE.i[LO] = 0;                   // twoE.d = 2^E
    y = twoE.d * (tbl * P);           // y = 2^E * tbl * P

    return y;
}
```

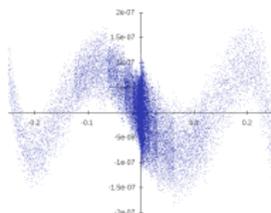
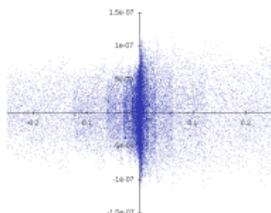
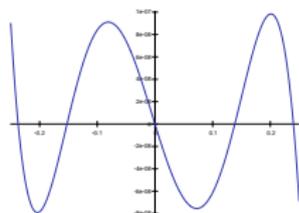
Sources d'erreur

- Des approximations en flottant, entachées d'erreurs



Sources d'erreur

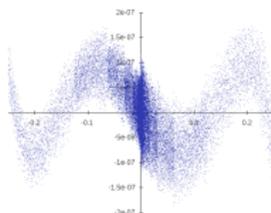
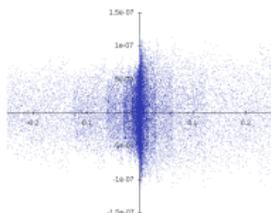
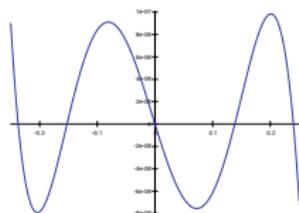
- Des approximations en flottant, entachées d'erreurs



- Cinq sources d'erreur principales :
 - Erreur de réduction d'argument
 - Erreur dans les entrées de table
 - Erreur d'approximation $\|p/f - 1\|_{\infty}$
 - Erreur d'évaluation du polynôme $|P(x)/p(x) - 1|$
 - Erreur de reconstruction

Sources d'erreur

- Des approximations en flottant, entachées d'erreurs



- Cinq sources d'erreur principales :

- Erreur de réduction d'argument
- Erreur dans les entrées de table
- Erreur d'approximation $\|p/f - 1\|_\infty$
- Erreur d'évaluation du polynôme $|P(x)/p(x) - 1|$
- Erreur de reconstruction

- Combinaison des erreurs pour donner l'erreur globale

- Analyse manuelle
 - Réduction d'argument, tabulation
 - Très fastidieuse
 - Très dangereuse

1.

- Analyse manuelle
 - Réduction d'argument, tabulation
 - Très fastidieuse
 - Très dangereuse
- L'outil Gappa (Melquiond)
 - Évaluation et reconstruction
 - Relativement simple (quasi-automatique)
 - Génération d'une preuve Coq possible

- Analyse manuelle
 - Réduction d'argument, tabulation
 - Très fastidieuse
 - Très dangereuse
- L'outil Gappa (Melquiond)
 - Évaluation et reconstruction
 - Relativement simple (quasi-automatique)
 - Génération d'une preuve Coq possible
- Algorithmes de calcul de norme infini validée¹
 - Erreur d'approximation
 - Automatique et rapide
 - Résultats validés par arithmétique d'intervalle
 - Génération d'une preuve HOL est sujet de recherche

1. Chevillard, Harrison, Joldeş et Lauter. *Efficient and accurate computation of upper bounds of approximation errors*, TCS, à paraître

Arithmétiques double-double et multi-double

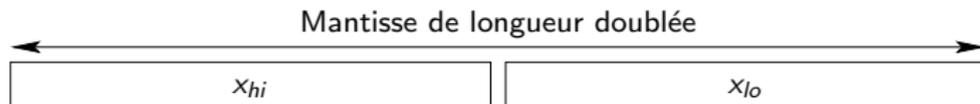
- Précision maximale disponible en matériel : double précision
- But : Arrondi fidèle ou correct en double précision

Arithmétiques double-double et multi-double

- Précision maximale disponible en matériel : double précision
- But : Arrondi fidèle ou correct en double précision
 - *Bits de garde* nécessaires au-delà de la double précision
 - Précisions d'environ 120-130 bits pour l'arrondi correct

Arithmétiques double-double et multi-double

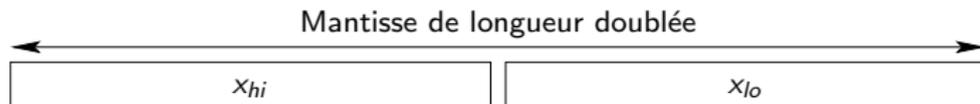
- Précision maximale disponible en matériel : double précision
- But : Arrondi fidèle ou correct en double précision
 - *Bits de garde* nécessaires au-delà de la double précision
 - Précisions d'environ 120-130 bits pour l'arrondi correct
- Arithmétique double-double :
 - Représenter x par une somme $x_{hi} + x_{lo}$ de deux doubles



- Base : errorfree transformations
 - ▶ Calcul de l'erreur des opérations flottantes

Arithmétiques double-double et multi-double

- Précision maximale disponible en matériel : double précision
- But : Arrondi fidèle ou correct en double précision
 - *Bits de garde* nécessaires au-delà de la double précision
 - Précisions d'environ 120-130 bits pour l'arrondi correct
- Arithmétique double-double :
 - Représenter x par une somme $x_{hi} + x_{lo}$ de deux doubles



- Base : errorfree transformations
 - ▶ Calcul de l'erreur des opérations flottantes
- Un concept extensible : arithmétique multi-double
 - Sommes de 3, 4... doubles
 - Triple-double suffisante pour l'arrondi correct en double
 - Un facteur 10 plus rapide que l'émulation logicielle

Défis de l'arrondi correct

Généralités sur l'arrondi correct et notations

L'implantation d'une fonction

Défis de l'arrondi correct

Conclusions et extensions

Précision au pire cas

- Pas de calcul *direct* de l'arrondi correct $F(x) = \circ(f(x))$
- Calcul d'une approximation $f(x) \cdot (1 + \varepsilon)$ plus précise que...
... la distance relative du pire cas

Précision au pire cas

- Pas de calcul *direct* de l'arrondi correct $F(x) = \circ(f(x))$
- Calcul d'une *approximation* $f(x) \cdot (1 + \varepsilon)$ *plus précise* que...
... la distance relative du *pire cas*
- Utilisation d'un lemme comme le suivant :

Lemme

Soit $f = \exp$. Soit $\circ_{53} : \mathbb{R} \rightarrow \mathbb{F}_{53}$ un arrondi double précision.

Soit $\mathbb{D} = \mathbb{F}_{53} \cap [-744.5; 709]$ le domaine de définition double de f .

Alors pour tout $x \in \mathbb{D} \setminus \{0\}$ et tout $|\varepsilon| \leq 2^{-159}$,

$$\circ_{53}(f(x) \cdot (1 + \varepsilon)) = \circ_{53}(f(x)).$$

Précision au pire cas

- Pas de calcul *direct* de l'arrondi correct $F(x) = \circ(f(x))$
- Calcul d'une approximation $f(x) \cdot (1 + \varepsilon)$ plus précise que...
... la distance relative du *pire cas*
- Utilisation d'un lemme comme le suivant :

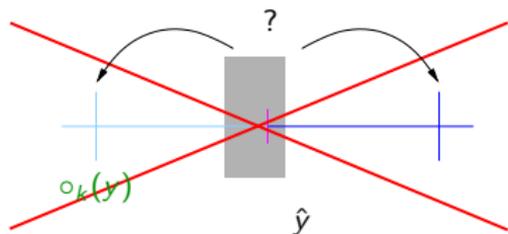
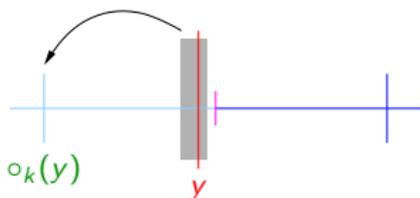
Lemme

Soit $f = \exp$. Soit $\circ_{53} : \mathbb{R} \rightarrow \mathbb{F}_{53}$ un arrondi double précision.

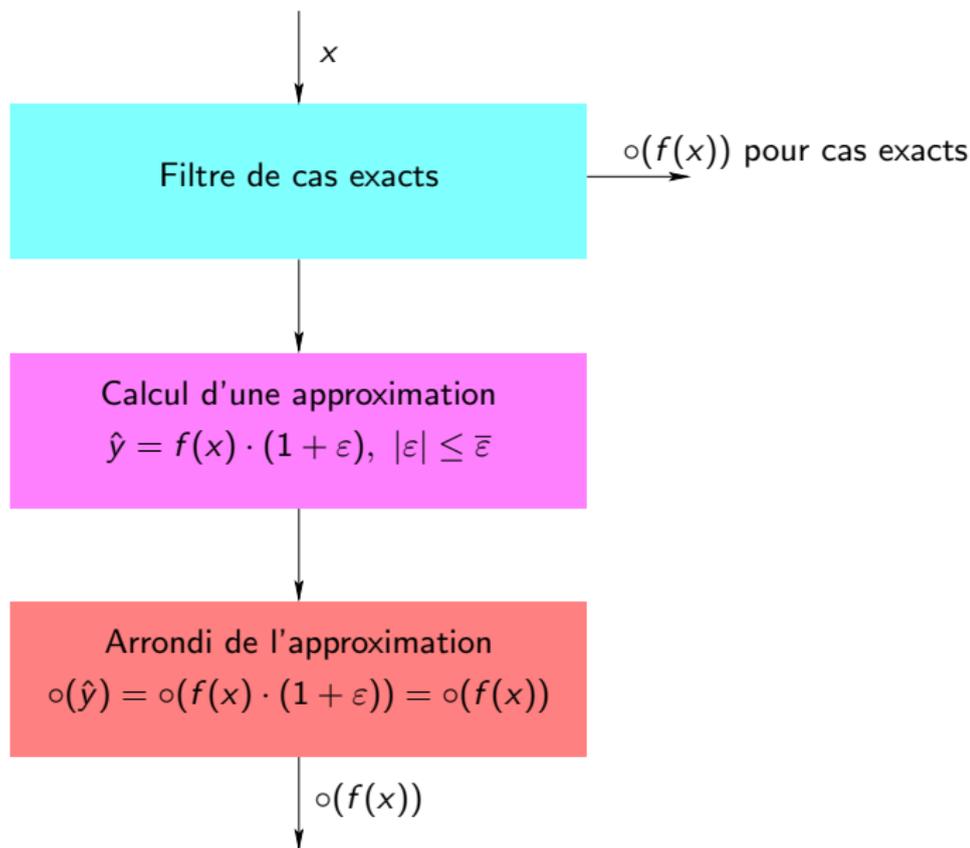
Soit $\mathbb{D} = \mathbb{F}_{53} \cap [-744.5; 709]$ le domaine de définition double de f .

Alors pour tout $x \in \mathbb{D} \setminus \{0\}$ et tout $|\varepsilon| \leq 2^{-159}$,

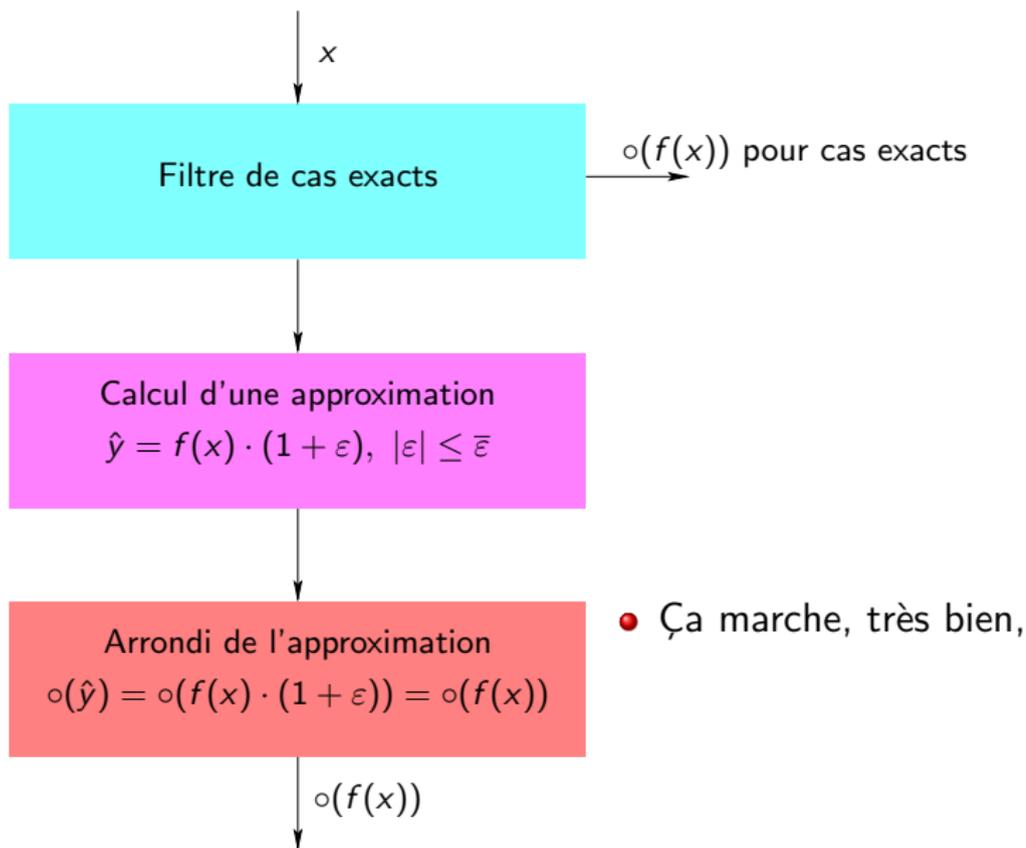
$$\circ_{53}(f(x) \cdot (1 + \varepsilon)) = \circ_{53}(f(x)).$$



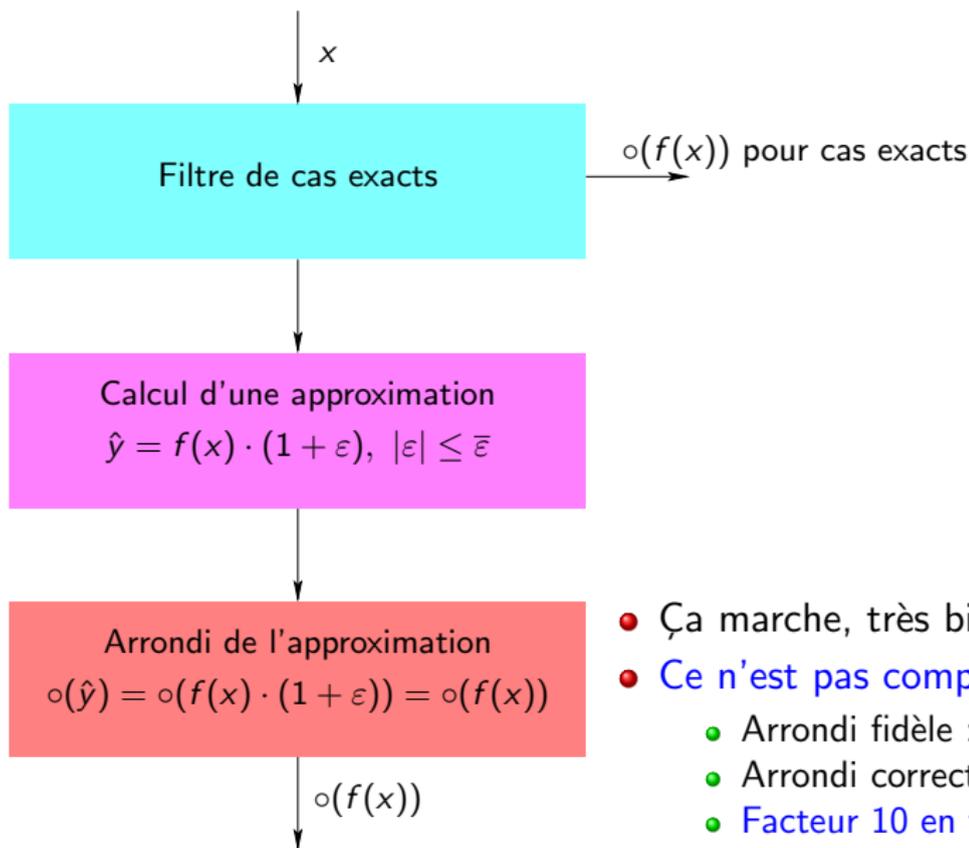
Un schéma simple pour l'arrondi correct



Un schéma simple pour l'arrondi correct

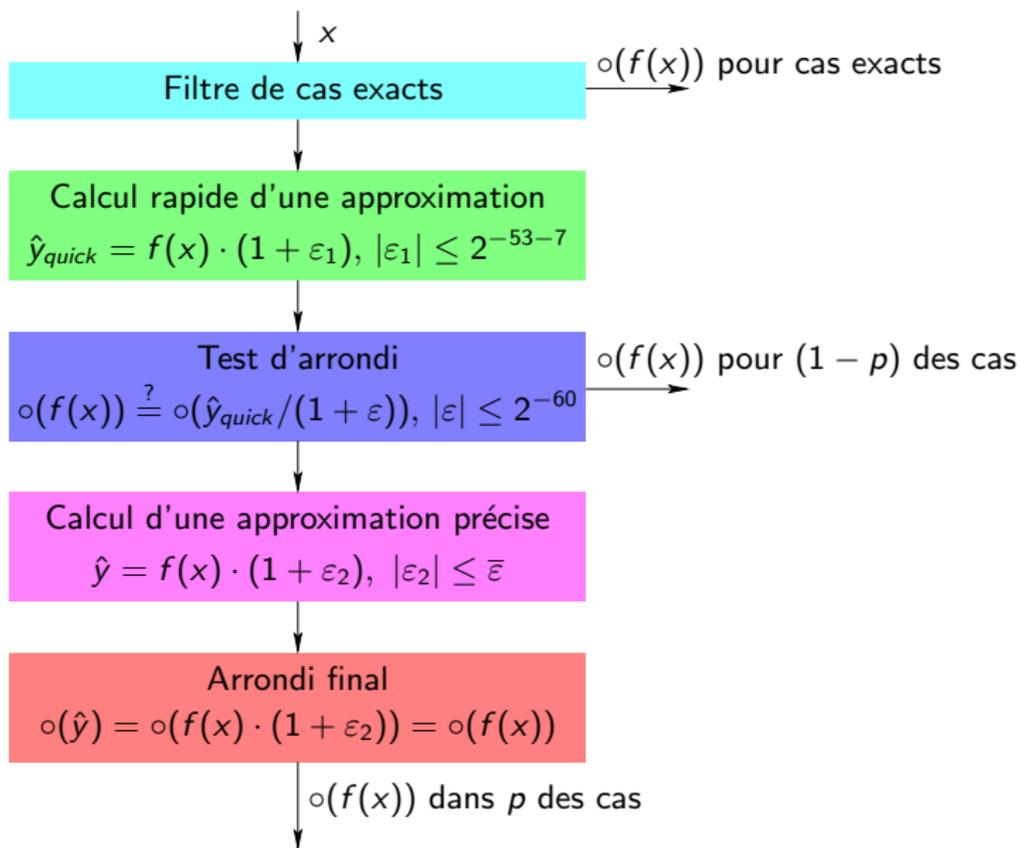


Un schéma simple pour l'arrondi correct

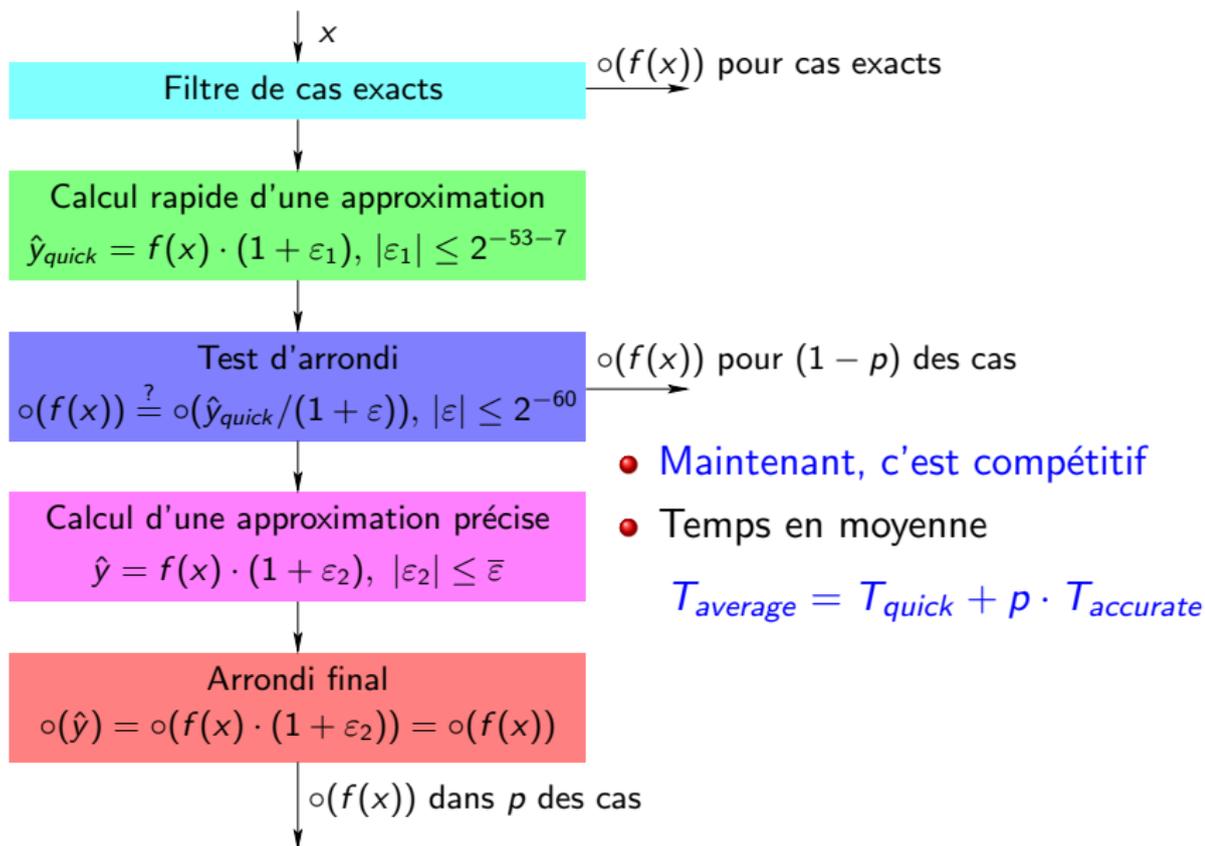


- Ça marche, très bien, mais...
- Ce n'est pas compétitif du tout
 - Arrondi fidèle : 53 + 5 bits
 - Arrondi correct : 159 bits
 - Facteur 10 en temps de calcul

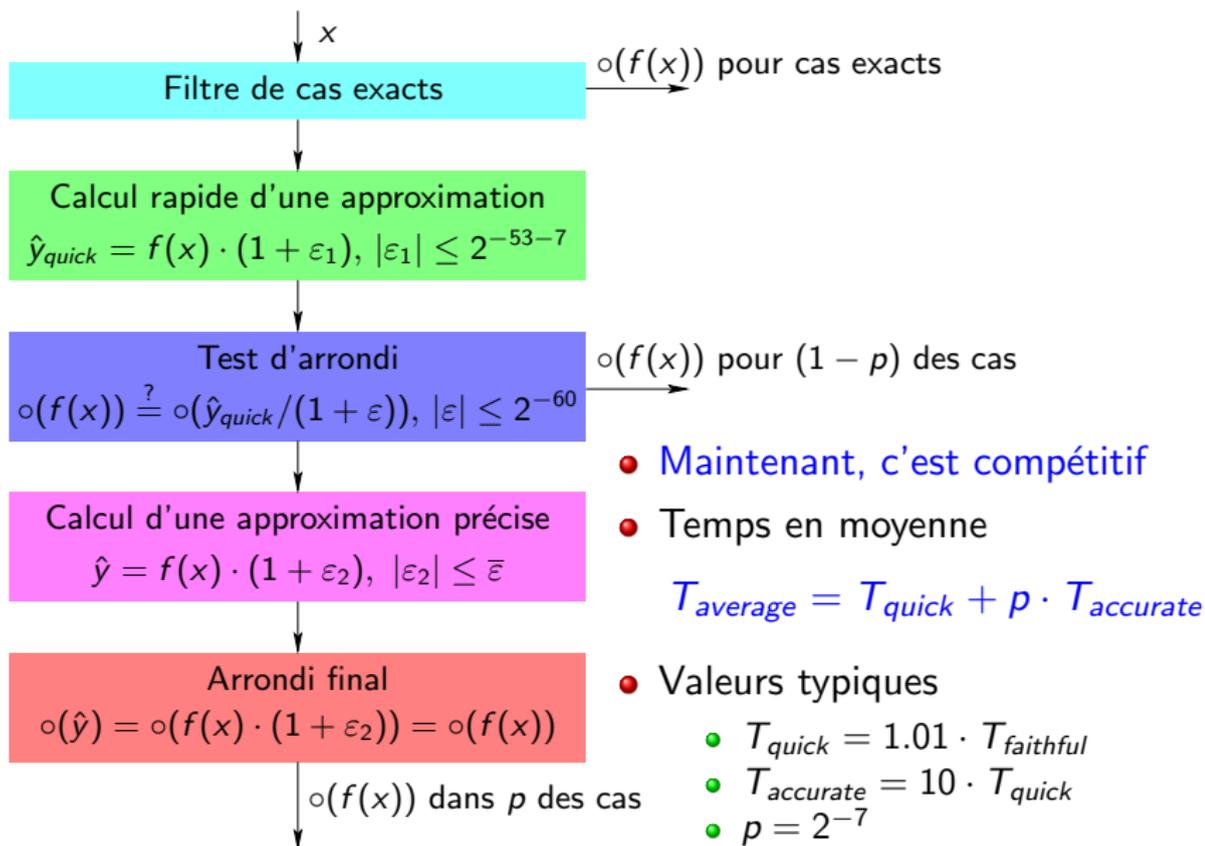
L'approche en deux étapes



L'approche en deux étapes



L'approche en deux étapes



Performances CRLibm

Opteron (cycles)	moyenne	pire
CRLibm en double-étendue	118	862
<i>libm par défaut (sans arrondi correct)</i>	189	8050
Pentium 4 (cycles)	moyenne	pire
CRLibm en double-étendue	339	4824
<i>libm par défaut (sans arrondi correct)</i>	332	8424
Pentium 3 (cycles)	moyenne	pire
CRLibm en double-étendue	150	891
<i>libm par défaut (sans arrondi correct)</i>	172	1286
Power 5 (unités de mesure)	moyenne	pire
CRLibm (sans FMA)	50	259
<i>libm par défaut (avec arrondi correct)</i>	52	28881
Itanium (cycles)	moyenne	pire
CRLibm en double-étendue avec FMA	73	2150
<i>libm par défaut (sans arrondi correct)</i>	54	8077

- Preuve de correction d'une fonction CRLibm
 - Preuve : analyse fine de l'erreur
 - ▶ Réduction d'argument
 - ▶ Erreur d'approximation
 - ▶ Erreur d'arrondi
 - ▶ etc.
 - Pour chacune des phases à part :
 - ▶ Phase rapide : hypothèses du test d'arrondi vérifiées
 - ▶ Phase précise : utilisation du lemme pire cas

Preuves de correction

- Preuve de correction d'une fonction CRLibm
 - Preuve : analyse fine de l'erreur
 - ▶ Réduction d'argument
 - ▶ Erreur d'approximation
 - ▶ Erreur d'arrondi
 - ▶ etc.
 - Pour chacune des phases à part :
 - ▶ Phase rapide : hypothèses du test d'arrondi vérifiées
 - ▶ Phase précise : utilisation du lemme pire cas
- Structure générale de la preuve d'arrondi correct

$$\frac{\text{Gamma} \quad \text{Norme inf}}{\hat{y} = f(x)(1 + \varepsilon), |\varepsilon| \leq \bar{\varepsilon} \quad \begin{array}{l} |\varepsilon| \leq \bar{\varepsilon} \Rightarrow o(f(x)) = o(f(x) \cdot (1 + \varepsilon)) \\ o(\hat{y}) = o(f(x)) \end{array}}$$

Le lemme clef qui manque...

- Méthodologie générale de preuve – 1 thèse
- **Gappa** : analyse formelle des erreurs d'arrondi – 1 thèse
- Normes infinies rigoureuses et validées – 1 thèse

Le lemme clef qui manque...

- Méthodologie générale de preuve – 1 thèse
- **Gappa** : analyse formelle des erreurs d'arrondi – 1 thèse
- Normes infinies rigoureuses et validées – 1 thèse
- Rien de formel disponible pour les lemmes clefs :

Lemme

Soit $f = \exp$. Soit $\circ_{53} : \mathbb{R} \rightarrow \mathbb{F}_{53}$ un arrondi double précision.

Soit $\mathbb{D} = \mathbb{F}_{53} \cap [-744.5; 709]$ le domaine de définition double de f .

Alors pour tout $x \in \mathbb{D} \setminus \{0\}$ et tout $|\varepsilon| \leq 2^{-159}$,

$$\circ_{53}(f(x) \cdot (1 + \varepsilon)) = \circ_{53}(f(x)).$$

Le lemme clef qui manque...

- Méthodologie générale de preuve – 1 thèse
- **Gappa** : analyse formelle des erreurs d'arrondi – 1 thèse
- Normes infinies rigoureuses et validées – 1 thèse
- Rien de formel disponible pour les lemmes clefs :

Lemme

Soit $f = \exp$. Soit $\circ_{53} : \mathbb{R} \rightarrow \mathbb{F}_{53}$ un arrondi double précision.

Soit $\mathbb{D} = \mathbb{F}_{53} \cap [-744.5; 709]$ le domaine de définition double de f .

Alors pour tout $x \in \mathbb{D} \setminus \{0\}$ et tout $|\varepsilon| \leq 2^{-159}$,

$$\circ_{53}(f(x) \cdot (1 + \varepsilon)) = \circ_{53}(f(x)).$$

- Ben, bon, c'est la tâche de l'ANR TaMaDi...

Conclusions et extensions

Généralités sur l'arrondi correct et notations

L'implantation d'une fonction

Défis de l'arrondi correct

Conclusions et extensions

Conclusion et extensions

- L'implantation d'une fonction correctement arrondie...
 - utilise des algorithmes classiques
 - se fait en deux étapes (rapide et précise)
 - est conditionnée par le pire cas de la fonction.

Conclusion et extensions

- L'implantation d'une fonction correctement arrondie...
 - utilise des algorithmes classiques
 - se fait en deux étapes (rapide et précise)
 - est conditionnée par le pire cas de la fonction.
- La preuve de correction...
 - passe par une analyse d'erreur fine
 - est plus ou moins fastidieuse
 - plus ou moins validée et sûre (Gappa et norme infinie)
 - est à jamais améliorable.

Conclusion et extensions

- L'implantation d'une fonction correctement arrondie...
 - utilise des algorithmes classiques
 - se fait en deux étapes (rapide et précise)
 - est conditionnée par le pire cas de la fonction.
- La preuve de correction...
 - passe par une analyse d'erreur fine
 - est plus ou moins fastidieuse
 - plus ou moins validée et sûre (Gappa et norme infinie)
 - est à jamais améliorable.
- La validation des calcul de pire cas est nécessaire :
 - Sinon, il restera toujours un trou dans la preuve finale.
 - C'est la clef pour l'industrialisation de l'IEEE 754-2008 9.2.

Conclusion et extensions

- L'implantation d'une fonction correctement arrondie...
 - utilise des algorithmes classiques
 - se fait en deux étapes (rapide et précise)
 - est conditionnée par le pire cas de la fonction.
- La preuve de correction...
 - passe par une analyse d'erreur fine
 - est plus ou moins fastidieuse
 - plus ou moins validée et sûre (Gappa et norme infinie)
 - est à jamais améliorable.
- La validation des calcul de pire cas est nécessaire :
 - Sinon, il restera toujours un trou dans la preuve finale.
 - C'est la clef pour l'industrialisation de l'IEEE 754-2008 9.2.
 - Mais c'est pour ça que nous sommes ici.

Merci !

Merci pour votre attention !

Des questions ?