Un algorithme de calcul de racine carrée en précision augmentée

Olivier Marty
Sous la direction de Mioara Joldes, MAC/LAAS-CNRS

École Normale Supérieure de Cachan

Vendredi 5 septembre 2014

LAAS-CNRS

Le capitole

Banyuls

Banyuls-sur-Mer

Plan

Plan

Expression mathématique

Definition

- $\beta \in \mathbb{N}^{\geq 2}$ la base
- $p \in \mathbb{N}^{\geq 1}$ la précision
- $e_{min}, e_{max} \in \mathbb{Z}$ les bornes de l'exposant

Les nombres à virgule flottante s'écrivent

$$(-1)^s \cdot m \cdot \beta^e$$

où

- $s \in \{0, 1\}$ le signe
- m le significande ($1 \leq m < \beta$ et m a un chiffre avant la virgule et au plus $p - 1$ après)
- e l'exposant ($e_{min} \leq e \leq e_{max}$)

Expression mathématique

Definition

- $\beta \in \mathbb{N}^{\geq 2}$ la base
- $p \in \mathbb{N}^{\geq 1}$ la précision
- $e_{min}, e_{max} \in \mathbb{Z}$ les bornes de l'exposant

Les nombres à virgule flottante s'écrivent

$$(-1)^s \cdot m \cdot \beta^e$$

où

- $s \in \{0, 1\}$ le signe
- m le significande ($1 \leq m < \beta$ et m a un chiffre avant la virgule et au plus $p - 1$ après)
- e l'exposant ($e_{min} \leq e \leq e_{max}$)

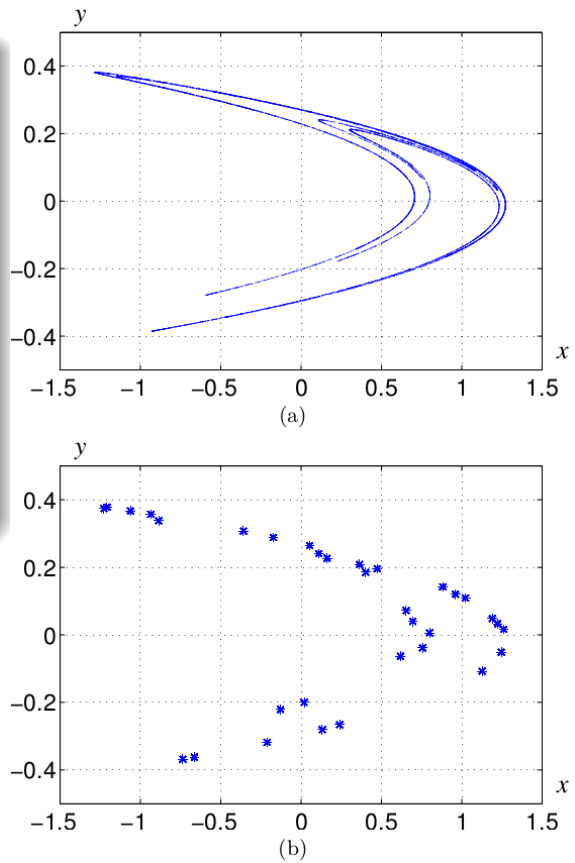
Limites

Exemples

- $\begin{cases} u_0 &= 2 \\ u_1 &= -4 \\ u_n &= 111 - \frac{1130}{u_{n-1}} + \frac{3000}{u_{n-1}u_{n-2}} \end{cases}$
  Tend vers 100 au lieu de 6
- Stabilité à long terme de systèmes

Tout de même...

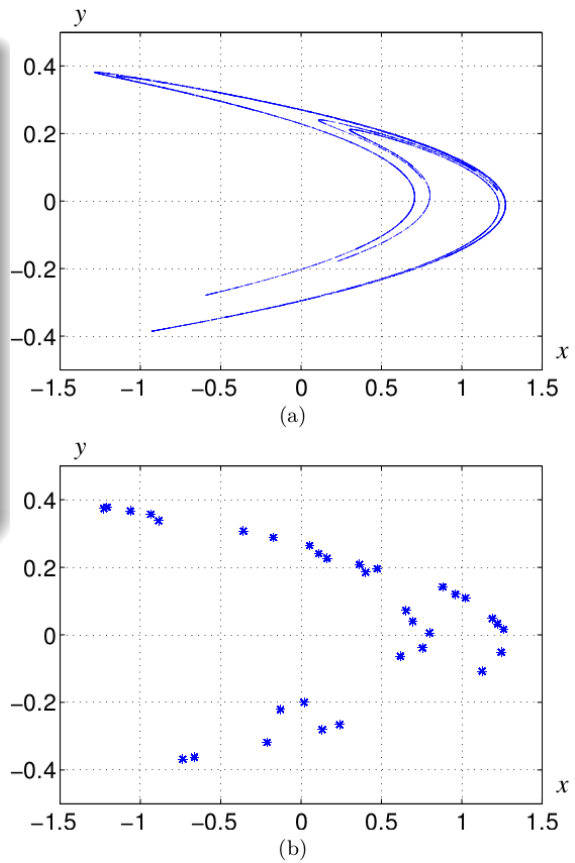
- distance Terre-Lune à 1mm près
- peu de constantes physiques très précises



Limites

Exemples

- $\begin{cases} u_0 &= 2 \\ u_1 &= -4 \\ u_n &= 111 - \frac{1130}{u_{n-1}} + \frac{3000}{u_{n-1}u_{n-2}} \end{cases}$
  Tend vers 100 au lieu de 6
- Stabilité à long terme de systèmes



Tout de même...

- distance Terre-Lune à 1mm près
- peu de constantes physiques très précises

2SUM

2SUM(a,b)

$$
\begin{array}{lcl}
s & \leftarrow & \mathrm{RN}(a+b) \\
b' & \leftarrow & \mathrm{RN}(s-a) \\
a' & \leftarrow & \mathrm{RN}(s-b') \\
\delta_b & \leftarrow & \mathrm{RN}(b-b') \\
\delta_a & \leftarrow & \mathrm{RN}(a-a') \\
t & \leftarrow & \mathrm{RN}(\delta_a+\delta_b)
\end{array}
$$

return (s,t)

Fast2SUM, Produits

Fast2SUM(a,b) si $a \geq b$

$$
\begin{aligned}
s &\leftarrow \mathrm{RN}(a+b) \\
z &\leftarrow \mathrm{RN}(s-a) \\
t &\leftarrow \mathrm{RN}(b-z)
\end{aligned}
$$

return (s,t)

Ces algorithmes sont minimaux en nombre d'opérations, et en profondeur des dépendances.

Produits

- Produit de Dekker (17 opérations)
- Instruction Fused Multiply Add :

$$
\begin{aligned}
r_1 &\leftarrow \mathrm{RN}(a \times b) \\
r_2 &\leftarrow \mathrm{RN}(a \times b - r_1)
\end{aligned}
$$

return (r_1, r_2)

Fast2SUM, Produits

Fast2SUM(a,b) si $a \geq b$

$s \leftarrow \mathrm{RN}(a+b)$
$z \leftarrow \mathrm{RN}(s-a)$
$t \leftarrow \mathrm{RN}(b-z)$
return (s,t)

Ces algorithmes sont minimaux en nombre d'opérations, et en profondeur des dépendances.

Produits

- Produit de Dekker (17 opérations)
- Instruction Fused Multiply Add :

  $r_1 \leftarrow \mathrm{RN}(a \times b)$
  $r_2 \leftarrow \mathrm{RN}(a \times b - r_1)$
  return (r_1, r_2)

QD et MPFR

QD

- Calcule avec deux/trois/quatre FP
- Incorrecte mais très rapide

GNU MPFR

- Utilise plusieurs entiers : significande et exposant
- Précision arbitraire
- Tous les modes d'arrondis supportés

QD et MPFR

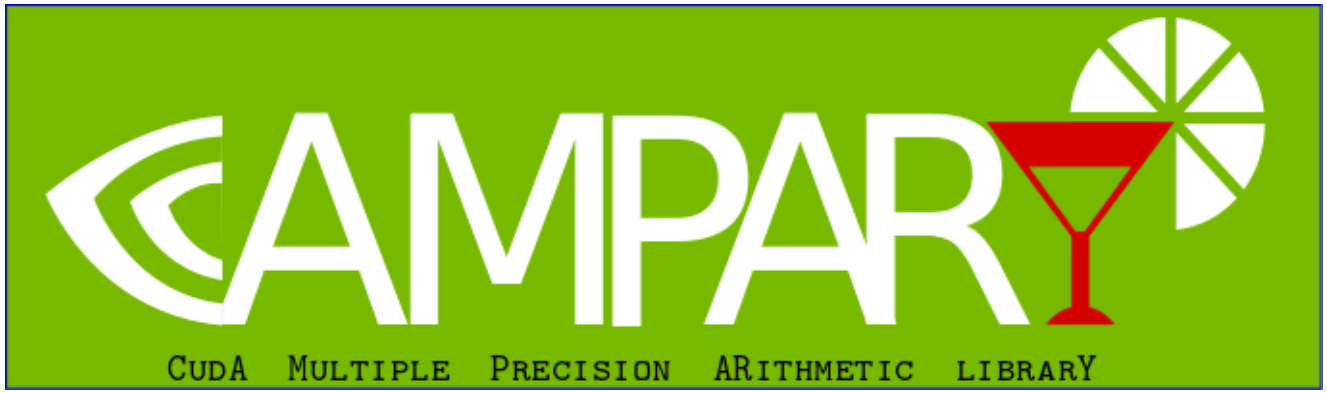
QD

- Calcule avec deux/trois/quatre FP
- Incorrecte mais très rapide

GNU MPFR

- Utilise plusieurs entiers : significande et exposant
- Précision arbitraire
- Tous les modes d'arrondis supportés

Campary



- Ciblée GPU
- Calcul haute performance
- Précision arbitraire
- Plus rapide que MPFR

Nombres manipulés

Definition (Floating Point Expansion)

$a = a_0 + \cdots + a_k$

Definition (Non chevauchement selon Priest)

e_x et e_y sont les exposants de x et y : $|e_x - e_y| \geq p$.

Definition (Non chevauchement selon Bailey)

Si $x < y$: $|x| \leq \frac{1}{2}\text{ulp}(y)$.

Nombres manipulés

Definition (Floating Point Expansion)

$a = a_0 + \cdots + a_k$

Definition (Non chevauchement selon Priest)

e_x et e_y sont les exposants de x et y : $|e_x - e_y| \geq p$.

Definition (Non chevauchement selon Bailey)

Si $x < y$: $|x| \leq \frac{1}{2}\text{ulp}(y)$.

Nombres manipulés

Definition (Floating Point Expansion)

$a = a_0 + \cdots + a_k$

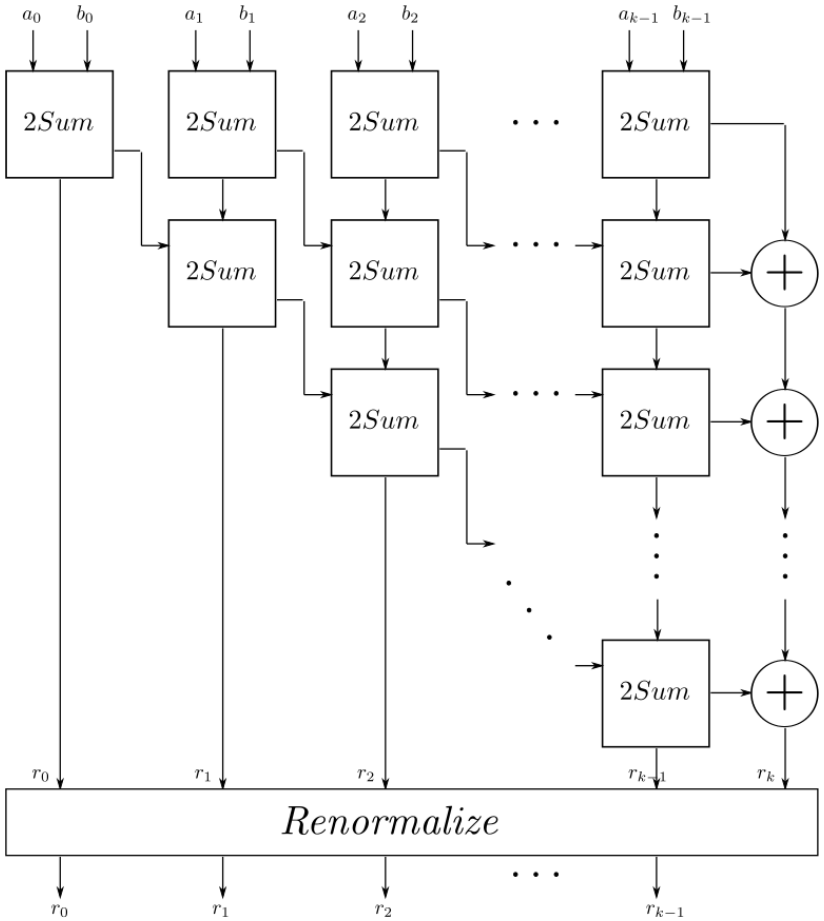
Definition (Non chevauchement selon Priest)

e_x et e_y sont les exposants de x et y : $|e_x - e_y| \geq p$.

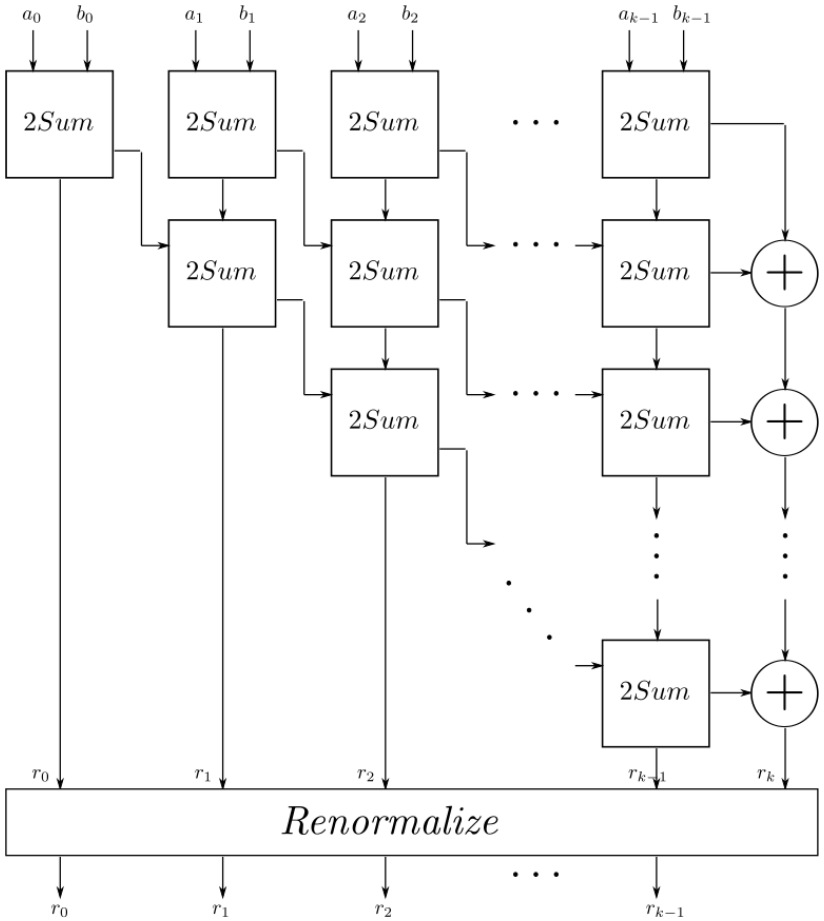
Definition (Non chevauchement selon Bailey)

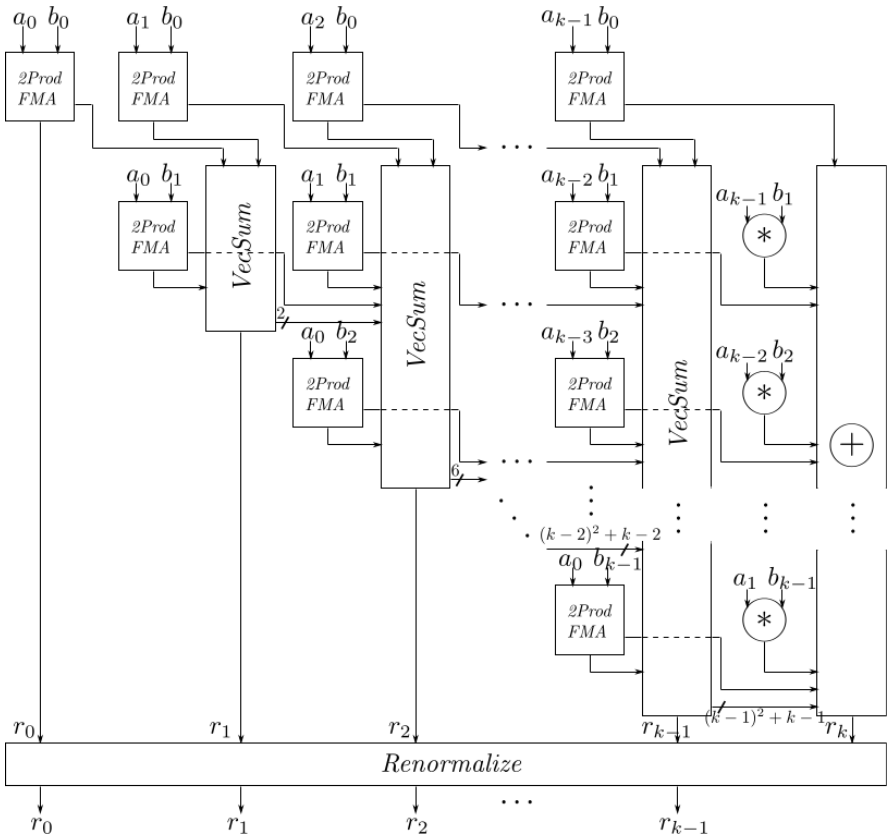
Si $x < y$: $|x| \leq \frac{1}{2}\text{ulp}(y)$.

Addition et multiplication



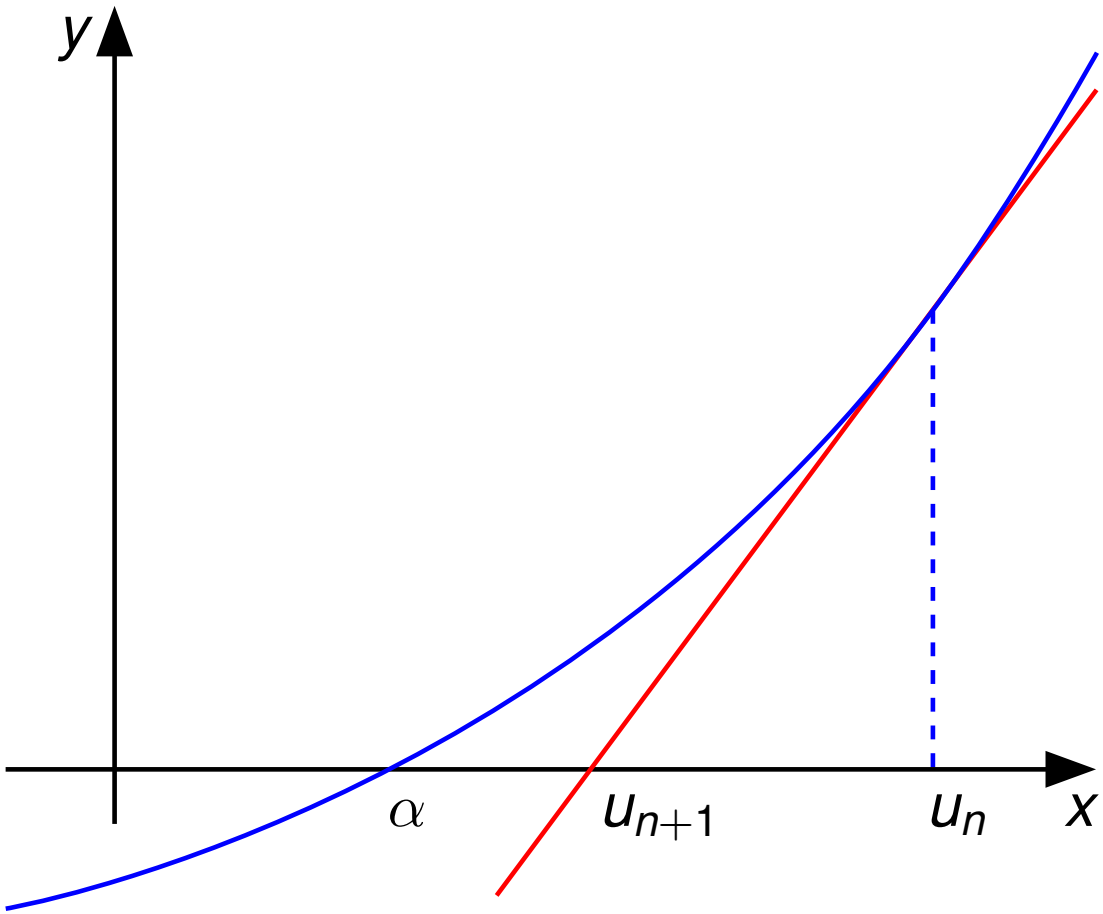
Addition et multiplication





Plan

L'itération de Newton-Raphson



$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}$$

Suites obtenues

$1/a$	$x \mapsto \frac{1}{x} - a$	$u_{n+1} = u_n(2 - au_n)$
$\sqrt{a}$	$x \mapsto x^2 - a$	$u_{n+1} = \frac{1}{2}\left(u_n + \frac{a}{u_n}\right)$ (suite de Babylone)
$1/\sqrt{a}$	$x \mapsto \frac{1}{x^2} - a$	$u_{n+1} = \frac{1}{2}u_n\left(3 - au_n^2\right)$

Entrée : $a = a_0 + \cdots + a_{2^k-1}$ et la précision cible 2^q termes.
Sortie : $x = x_0 + \cdots + x_{2^q-1}$ tel que

$$\left| x - \frac{1}{\sqrt{a}} \right| \leq \frac{2^{-2^q(p-3)-1}}{\sqrt{a}}$$

Algorithme

$x_0 \leftarrow 1/\sqrt{a_0}$
for $i \leftarrow 1$ to q do
  $\hat{v}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{a}^{(2^i)}$
  $\hat{w}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{v}^{(2^i)}$
  $\hat{y}^{(2^i)} \leftarrow 3 - \hat{w}^{(2^i)}$
  $\hat{z}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{y}^{(2^i)}$
  $x^{(2^i)} \leftarrow \hat{z}^{(2^i)} / 2$
end for
return $x = x_0 + \cdots + x_{2^q-1}$

Entrée : $a = a_0 + \cdots + a_{2^k-1}$ et la précision cible 2^q termes.
Sortie : $x = x_0 + \cdots + x_{2^q-1}$ tel que

$$\left| x - \frac{1}{\sqrt{a}} \right| \leq \frac{2^{-2^q(p-3)-1}}{\sqrt{a}}$$

Algorithme

$x_0 \leftarrow 1/\sqrt{a_0}$
for $i \leftarrow 1$ to q do
  $\hat{v}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{a}^{(2^i)}$
  $\hat{w}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{v}^{(2^i)}$
  $\hat{y}^{(2^i)} \leftarrow 3 - \hat{w}^{(2^i)}$
  $\hat{z}^{(2^i)} \leftarrow x^{(2^{i-1})} \times \hat{y}^{(2^i)}$
  $x^{(2^i)} \leftarrow \hat{z}^{(2^i)} / 2$
end for
return $x = x_0 + \cdots + x_{2^q-1}$

Démonstration

Prendre en compte les erreurs de méthode et de calcul.

Idées

Lemme : $|u_i| \leq 2^{-ip}|u_0|$

Avec $\eta = \sum_{j=0}^{+\infty} 2^{-p(j+1)} = \frac{1}{2^p-1}$

on a $|u - u^{(i+1)}| \leq 2^{-ip}|u|\frac{\eta}{1-\eta}$

Borne de ε_0 : RN

Borne de ε_{i+1} :

- inégalités triangulaires à chaque calcul
- décroissance quadratique de la suite

Démonstration

Prendre en compte les erreurs de méthode et de calcul.

Idées

Lemme : $|u_i| \leq 2^{-ip}|u_0|$

Avec $\eta = \sum_{j=0}^{+\infty} 2^{-p(j+1)} = \frac{1}{2^p-1}$

on a $|u - u^{(i+1)}| \leq 2^{-ip}|u|\frac{\eta}{1-\eta}$

Borne de ε_0 : RN

Borne de ε_{i+1} :

- inégalités triangulaires à chaque calcul
- décroissance quadratique de la suite

Démonstration

Prendre en compte les erreurs de méthode et de calcul.

Idées

Lemme : $|u_i| \leq 2^{-ip}|u_0|$

Avec $\eta = \sum_{j=0}^{+\infty} 2^{-p(j+1)} = \frac{1}{2^p-1}$

on a $|u - u^{(i+1)}| \leq 2^{-ip}|u|\frac{\eta}{1-\eta}$

Borne de ε_0 : RN

Borne de ε_{i+1} :

- inégalités triangulaires à chaque calcul
- décroissance quadratique de la suite

Démonstration

Prendre en compte les erreurs de méthode et de calcul.

Idées

Lemme : $|u_i| \leq 2^{-ip}|u_0|$

Avec $\eta = \sum_{j=0}^{+\infty} 2^{-p(j+1)} = \frac{1}{2^p-1}$

on a $|u - u^{(i+1)}| \leq 2^{-ip}|u|\frac{\eta}{1-\eta}$

Borne de ε_0 : RN

Borne de ε_{i+1} :

- inégalités triangulaires à chaque calcul
- décroissance quadratique de la suite

Architecture GPU

Contraintes

- Peu de branchements conditionnels
- Déroulement des boucles
- Taille des tableaux fixée

Défaut

- Taille du binaire produit

Architecture GPU

Contraintes

- Peu de branchements conditionnels
- Déroulement des boucles
- Taille des tableaux fixée

Défaut

- Taille du binaire produit

Conclusion

- Équipe très sympa
- Algorithme intégré dans la bibliothèque
- Travail assez pragmatique mais utile
- Beaucoup inspiré d'un papier

References

Mioara Joldes, Jean-Michel Muller, and Valentina Popescu.
On the computation of the reciprocal of floating point expansions using an adapted Newton-Raphson iteration.
In *Proceedings of 25th IEEE International Conference on Application-specific Systems, Architectures and Processors*, page to appear, Zurich, Suisse, 2014.

Jean-Michel Muller, Nicolas Brisebarre, Florent de Dinechin, Claude-Pierre Jeannerod, Vincent Lefèvre, Guillaume Melquiond, Nathalie Revol, Damien Stehlé, and Serge Torres.
Handbook of Floating-Point Arithmetic.
Birkhäuser Boston, 2010.