ANALYSE NUMÉRIQUE I

COURS 3

PROPAGATION DES ERREURS

Ch. Baskiotis

LAPI – EISTI

6 mars 2010

Ce que nous savons – I

Un nombre réel x et sa représentation en ordinateur $fl(x)$ avec $x \neq fl(x)$.

Ce que nous savons – I

Un nombre réel x et sa représentation en ordinateur $fl(x)$ avec $x \neq fl(x)$.

D'où erreur de représentation : $\delta x = fl(x) - x$.

Ce que nous savons – I

Un nombre réel x et sa représentation en ordinateur $fl(x)$ avec $x \neq fl(x)$.

D'où erreur de représentation : $\delta x = fl(x) - x$.

On utilise surtout l'errur relative de précision ou sa valeur absolue :

$$\eta(x) = \frac{fl(x) - x}{x}$$

Ce que nous savons – I

Un nombre réel x et sa représentation en ordinateur $fl(x)$ avec $x \neq fl(x)$.

D'où erreur de représentation : $\delta x = fl(x) - x$.

On utilise surtout l'errur relative de précision ou sa valeur absolue :

$$\eta(x) = \frac{fl(x) - x}{x}$$

d'où on a

$$fl(x) = x \cdot (1 + \eta(x))$$

Ce que nous savons – II

Stockage d'un réel x dans un ordinateur binaire :
Un bit de signe – p bits de mantisse – q bits d'exposant

Ce que nous savons – II

Stockage d'un réel x dans un ordinateur binaire :
Un bit de signe – p bits de mantisse – q bits d'exposant
x s'écrit :

$$x = (-1)^s \times w \times 2^n = (-1)^s \times \left(r + t \times 2^{-p}\right) \times 2^n;\ 2^{-1} \leq \mid r \mid < 1,\ 0 \leq \mid t \mid < 1$$

Ce que nous savons – II

Stockage d'un réel x dans un ordinateur binaire :
Un bit de signe – p bits de mantisse – q bits d'exposant
x s'écrit :

$$x = (-1)^s \times w \times 2^n = (-1)^s \times \left(r + t \times 2^{-p}\right) \times 2^n;\ 2^{-1} \leq \mid r \mid < 1,\ 0 \leq \mid t \mid < 1$$

Dans ce cas, l'erreur relative de précision est bornée et on a

$$\mid \eta(x) \mid \leq \begin{cases} 2^{1-p}, & \text{si approximation par troncature} \\ 0.5 \times 2^{1-p}, & \text{si approximation par arrondi} \end{cases} ;\ x \in \mathbb{R}$$

Ce que nous savons – III

Selon le standard IEEE-754, le nombre

$$x = (-1)^s \times w \times 2^n = (-1)^s \times \left(r + t \times 2^{-p}\right) \times 2^n;\ 2^{-1} \leq \mid r \mid < 1,\ 0 \leq \mid t \mid < 1$$

doit se coder comme suit

$$x = (-1)^s \times \left[\left(r + t \times 2^{-(p+1)}\right) \times 2^E\right]$$

Ce que nous savons – III

Selon le standard IEEE-754, le nombre

$$x = (-1)^s \times w \times 2^n = (-1)^s \times \left(r + t \times 2^{-p}\right) \times 2^n;\ 2^{-1} \leq\mid r \mid< 1,\ 0 \leq\mid t \mid< 1$$

doit se coder comme suit

$$x = (-1)^s \times \left[\left(r + t \times 2^{-(p+1)}\right) \times 2^E\right]$$

en tenant compte

- du bit caché qui permet de stocker un bit de plus, d'où l'exposant $p+1$ dans $2^{-(p+1)}$;

Ce que nous savons – III

Selon le standard IEEE-754, le nombre

$$x = (-1)^s \times w \times 2^n = (-1)^s \times \left(r + t \times 2^{-p}\right) \times 2^n;\ 2^{-1} \leq \mid r \mid < 1,\ 0 \leq \mid t \mid < 1$$

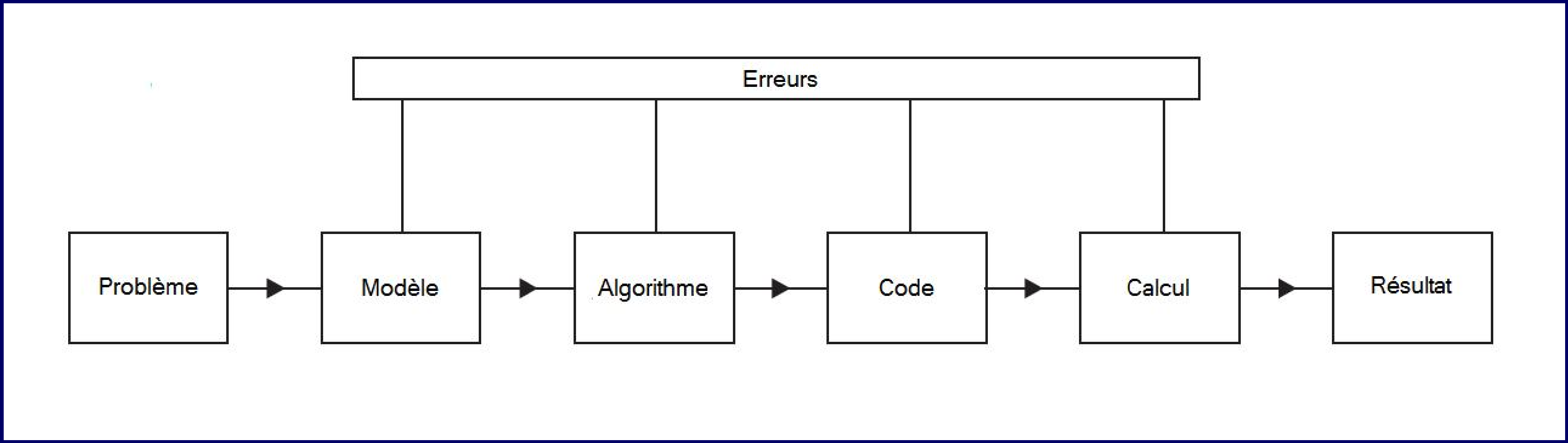
doit se coder comme suit

$$x = (-1)^s \times \left[\left(r + t \times 2^{-(p+1)}\right) \times 2^E\right]$$

en tenant compte

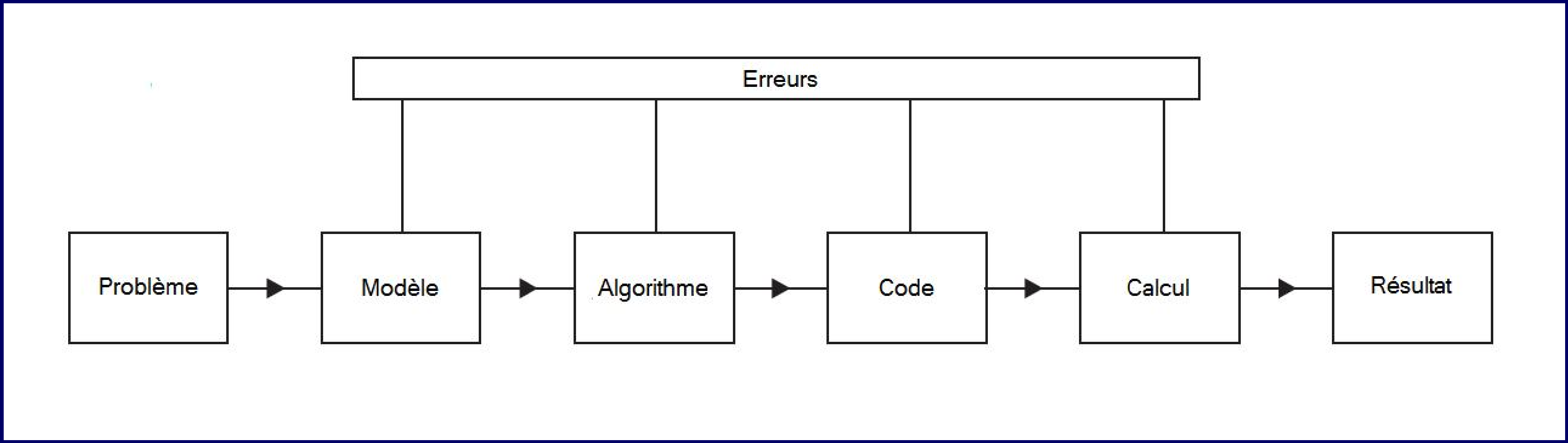
- du bit caché qui permet de stocker un bit de plus, d'où l'exposant $p+1$ dans $2^{-(p+1)}$;
- du biais $B = 2^{(q-1)} - 1$ dans l'exposant : $E = B + n$.

Sources des erreurs



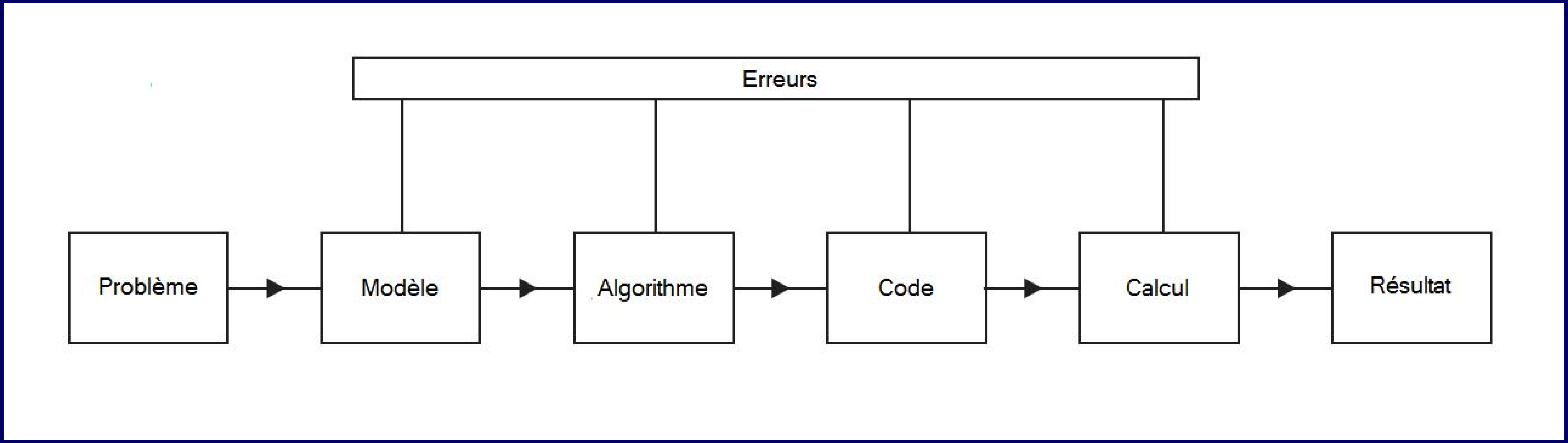
- Erreurs de la modélisation mathématique

Sources des erreurs



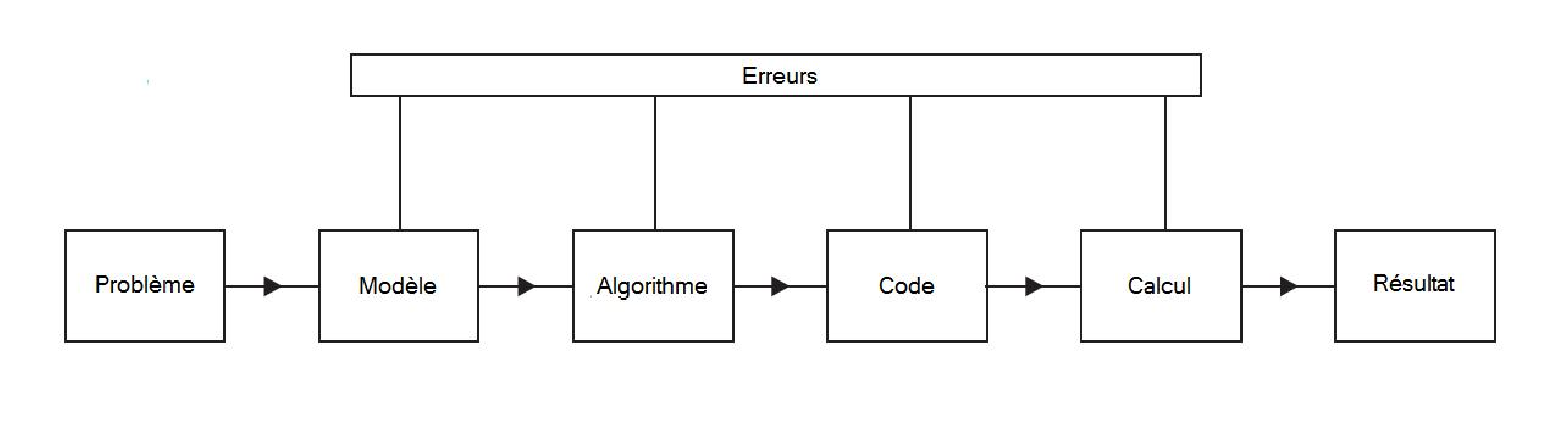
- Erreurs de la modélisation mathématique
- Erreurs d'approximation

Sources des erreurs



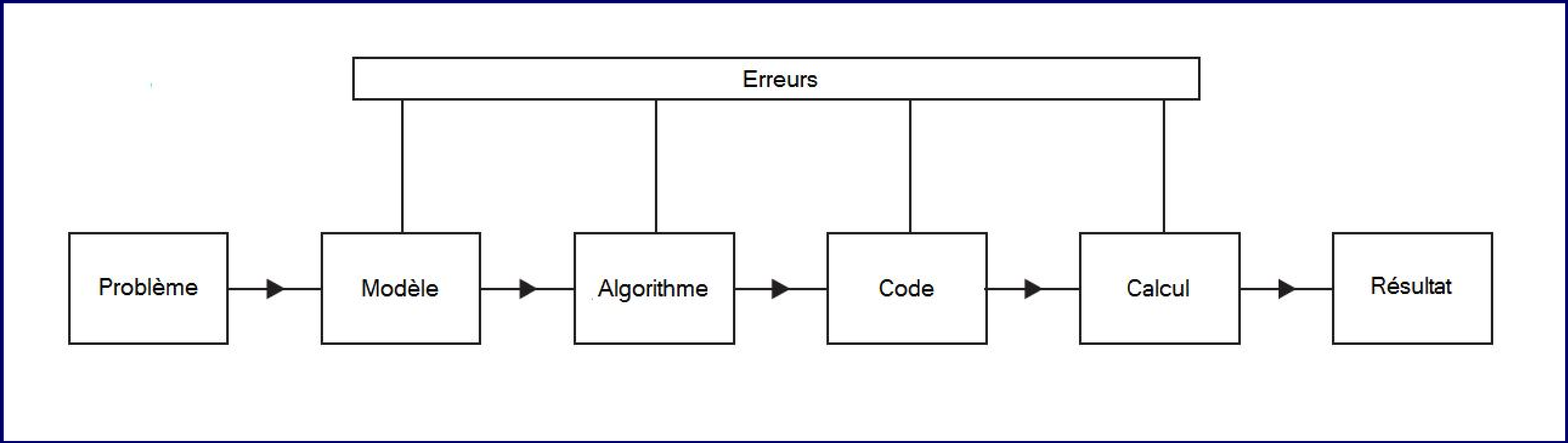
- Erreurs de la modélisation mathématique
- Erreurs d'approximation
- Erreurs de discrétisation

Sources des erreurs



- Erreurs de la modélisation mathématique
- Erreurs d'approximation
- Erreurs de discrétisation
- Erreurs du calcul numérique

Sources des erreurs



- Erreurs de la modélisation mathématique
- Erreurs d'approximation
- Erreurs de discrétisation
- Erreurs du calcul numérique

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{\ }$).

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).

$c = a \otimes b$ où c est le résultat de l'opération.

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).

$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).
$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

$$fl(c) = fl(fl(a) \otimes fl(b))$$

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).

$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

$$fl(c) = fl(fl(a) \otimes fl(b))$$

L'erreur de précision $\eta(c) = \dfrac{fl(c) - c}{c}$ peut se décomposer en deux parties :

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).

$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

$$fl(c) = fl(fl(a) \otimes fl(b))$$

L'erreur de précision $\eta(c) = \dfrac{fl(c) - c}{c}$ peut se décomposer en deux parties :

1. ***une erreur du calcul***, notée $\eta^C(c) = \eta^C(a \otimes b)$, qui est caractéristique de la machine et contre laquelle on ne peut rien faire (sauf à changer de machine),

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\surd$).

$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

$$fl(c) = fl(fl(a) \otimes fl(b))$$

L'erreur de précision $\eta(c) = \dfrac{fl(c) - c}{c}$ peut se décomposer en deux parties :

1. ***une erreur du calcul***, notée $\eta^C(c) = \eta^C(a \otimes b)$, qui est caractéristique de la machine et contre laquelle on ne peut rien faire (sauf à changer de machine), et
2. une ***erreur de l'entrée***, notée $\eta^I(c) = \eta^I(a \otimes b)$, due à la représentation par l'ordinateur du résultat de l'opération et qui dépend de l'ordre des calculs.

Opérations arithmétiques et erreur

$a, b \in \mathbb{R}$ et $\otimes$ opération arithmétique (c-à-d. $+$, $-$, $\times$, $/$, $\sqrt{}$).
$c = a \otimes b$ où c est le résultat de l'opération.

Sur un ordinateur, on a :

$$fl(c) = fl(fl(a) \otimes fl(b))$$

L'erreur de précision $\eta(c) = \dfrac{fl(c) - c}{c}$ peut se décomposer en deux parties :

1. ***une erreur du calcul***, notée $\eta^C(c) = \eta^C(a \otimes b)$, qui est caractéristique de la machine et contre laquelle on ne peut rien faire (sauf à changer de machine), et
2. une ***erreur de l'entrée***, notée $\eta^I(c) = \eta^I(a \otimes b)$, due à la représentation par l'ordinateur du résultat de l'opération et qui dépend de l'ordre des calculs.
   $\Rightarrow$ On change l'ordre de calculs $\Rightarrow$ on modifie l'erreur.

Opérations arithmétiques et erreur – Addition

THÉORÈME DE L'ERREUR POUR LES SOMMES .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

Opérations arithmétiques et erreur – Addition

THÉORÈME DE L'ERREUR POUR LES SOMMES .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

L'erreur absolue de l'entrée pour cette somme est

$$\Delta^I S = \Delta^I x_1 + \Delta^I x_2 + \cdots$$

Opérations arithmétiques et erreur – Addition

Théorème de l'Erreur pour les Sommes .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

L'erreur absolue de l'entrée pour cette somme est

$$\Delta^I S = \Delta^I x_1 + \Delta^I x_2 + \cdots$$

et l'erreur de précision de l'entrée est

$$\eta^I(S) = \tfrac{x_1}{S}\eta^I(x_1) + \tfrac{x_2}{S}\eta^I(x_2) + \cdots \text{ , avec } \left|\eta^I(x_1)\right| \leq \text{eps.}$$

Opérations arithmétiques et erreur – Addition

Théorème de l'Erreur pour les Sommes .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

L'erreur absolue de l'entrée pour cette somme est

$$\Delta^I S = \Delta^I x_1 + \Delta^I x_2 + \cdots$$

et l'erreur de précision de l'entrée est

$$\eta^I\left(S\right) = \frac{x_1}{S}\eta^I\left(x_1\right) + \frac{x_2}{S}\eta^I\left(x_2\right) + \cdots \text{ , avec } \left|\eta^I\left(x_1\right)\right| \leq \text{eps.}$$

$\Rightarrow$ Erreur maximale de la somme

$$\left|\eta^I\left(S\right)\right| \leq \left|\frac{x_1}{S}\right|\left|\eta^I\left(x_1\right)\right| + \left|\frac{x_2}{S}\right|\left|\eta^I\left(x_2\right)\right| + \cdots \leq \frac{1}{|S|}\left(|\, x_1\,| + |\, x_2\,| + \cdots\right) \cdot \text{eps}$$

Opérations arithmétiques et erreur – Addition

THÉORÈME DE L'ERREUR POUR LES SOMMES .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

L'erreur absolue de l'entrée pour cette somme est

$$\Delta^I S = \Delta^I x_1 + \Delta^I x_2 + \cdots$$

et l'erreur de précision de l'entrée est

$$\eta^I(S) = \frac{x_1}{S}\eta^I(x_1) + \frac{x_2}{S}\eta^I(x_2) + \cdots \text{, avec } \left|\eta^I(x_1)\right| \leq \text{eps.}$$

$\Rightarrow$ Erreur maximale de la somme

$$\left|\eta^I(S)\right| \leq \left|\frac{x_1}{S}\right| \left|\eta^I(x_1)\right| + \left|\frac{x_2}{S}\right| \left|\eta^I(x_2)\right| + \cdots \leq \frac{1}{|S|}(|\, x_1 \,| + |\, x_2 \,| + \cdots) \cdot \text{eps}$$

$$\Rightarrow \quad \left|\eta^I(S)\right| \leq \frac{\sum_i |x_i|}{\left|\sum_i x_i\right|} \times \text{eps}$$

Opérations arithmétiques et erreur – Addition

THÉORÈME DE L'ERREUR POUR LES SOMMES .- Soit la somme

$$S = x_1 + x_2 + \cdots$$

L'erreur absolue de l'entrée pour cette somme est

$$\Delta^I S = \Delta^I x_1 + \Delta^I x_2 + \cdots$$

et l'erreur de précision de l'entrée est

$$\eta^I(S) = \frac{x_1}{S}\eta^I(x_1) + \frac{x_2}{S}\eta^I(x_2) + \cdots \text{, avec } \left|\eta^I(x_1)\right| \leq \text{eps.}$$

$\Rightarrow$ Erreur maximale de la somme

$$\left|\eta^I(S)\right| \leq \left|\frac{x_1}{S}\right| \left|\eta^I(x_1)\right| + \left|\frac{x_2}{S}\right| \left|\eta^I(x_2)\right| + \cdots \leq \frac{1}{|S|}\left(|\, x_1 \,| + |\, x_2 \,| + \cdots\right) \cdot \text{eps}$$

$$\Rightarrow \quad \left|\eta^I(S)\right| \leq \frac{\sum_i |x_i|}{\left|\sum_i x_i\right|} \times \text{eps}$$

$\Rightarrow$ **Si** $\left|\sum_i x_i\right|$ **est petit par rapport à** $\sum_i |x_i| \times$ **eps, l'erreur de précision de l'entrée peut devenir importante.**

Opérations arithmétiques et erreur – Multiplication

THÉORÈME DE L'ERREUR POUR LES PRODUITS .- Soit le produit

$$P = x_1 \cdot x_2 \cdot \ldots \quad \text{avec } x_i \neq 0 \ \forall i$$

Opérations arithmétiques et erreur – Multiplication

Théorème de l'Erreur pour les Produits .- Soit le produit

$$P = x_1 \cdot x_2 \cdot \ldots \quad \text{avec } x_i \neq 0 \ \forall i$$

L'erreur absolue de l'entrée est

$$\Delta^I P = P \cdot \left(\frac{\Delta^I x_1}{x_1} + \frac{\Delta^I x_2}{x_2} + \cdots \right)$$

Opérations arithmétiques et erreur – Multiplication

THÉORÈME DE L'ERREUR POUR LES PRODUITS .- Soit le produit

$$P = x_1 \cdot x_2 \cdot \ldots \quad \text{avec } x_i \neq 0 \ \forall i$$

L'erreur absolue de l'entrée est

$$\Delta^I P = P \cdot \left(\frac{\Delta^I x_1}{x_1} + \frac{\Delta^I x_2}{x_2} + \cdots \right)$$

et l'erreur de précision de l'entrée est

$$\eta^I (P) = \eta^I (x_1) + \eta^I (x_2) + \cdots$$

Opérations arithmétiques et erreur – Multiplication

THÉORÈME DE L'ERREUR POUR LES PRODUITS .- Soit le produit

$$P = x_1 \cdot x_2 \cdot \ldots \quad \text{avec } x_i \neq 0 \ \forall i$$

L'erreur absolue de l'entrée est

$$\Delta^I P = P \cdot \left(\frac{\Delta^I x_1}{x_1} + \frac{\Delta^I x_2}{x_2} + \cdots \right)$$

et l'erreur de précision de l'entrée est

$$\eta^I (P) = \eta^I (x_1) + \eta^I (x_2) + \cdots$$

Donc la borne maximale pour l'erreur de précision pour la multiplication est $\eta^I (P) \leq N \cdot \text{eps}$, où N est le nombre de facteurs dans le produit P

Opérations arithmétiques et erreur – Division

THÉORÈME DE L'ERREUR POUR LA DIVISION .- Soit l'opération

$$Q = \frac{a}{b}\ ;\ b \neq 0$$

Opérations arithmétiques et erreur – Division

THÉORÈME DE L'ERREUR POUR LA DIVISION .- Soit l'opération

$$Q = \frac{a}{b} \; ; \; b \neq 0$$

L'erreur absolue de l'entrée est

$$\Delta^I Q = \frac{b\Delta^I a - a\Delta^I b}{b^2}$$

Opérations arithmétiques et erreur – Division

Théorème de l'Erreur pour la Division .- Soit l'opération

$$Q = \frac{a}{b} \; ; \; b \neq 0$$

L'erreur absolue de l'entrée est

$$\Delta^I Q = \frac{b\Delta^I a - a\Delta^I b}{b^2}$$

et l'erreur de précision à l'entrée est

$$\eta^I (Q) = \eta^I (a) - \eta^I (b)$$

Opérations arithmétiques et erreur – Division

THÉORÈME DE L'ERREUR POUR LA DIVISION .- Soit l'opération

$$Q = \frac{a}{b} \; ; \; b \neq 0$$

L'erreur absolue de l'entrée est

$$\Delta^I Q = \frac{b\Delta^I a - a\Delta^I b}{b^2}$$

et l'erreur de précision à l'entrée est

$$\eta^I(Q) = \eta^I(a) - \eta^I(b)$$

Donc la borne maximale pour l'erreur de précision pour la division est

$$\mid \eta^I(Q) \mid \leq 2 \cdot \text{eps}$$

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : NON

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : NON

Peut-on réduire l'erreur ? Réponse : OUI

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : <u>NON</u>

Peut-on réduire l'erreur ? Réponse : <u>OUI</u> sous certaines conditions

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : NON

Peut-on réduire l'erreur ? Réponse : OUI sous certaines conditions dont une essentielle : ***Programmer différemment***.

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : NON

Peut-on réduire l'erreur ? Réponse : OUI sous certaines conditions dont une essentielle : ***Programmer différemment***.

Pour ce faire, il faut comprendre

- l'influence des erreurs sur le résultat d'une suite d'opérations effectuées selon un algorithme ;

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : NON

Peut-on réduire l'erreur ? Réponse : OUI sous certaines conditions dont une essentielle : ***Programmer différemment***.

Pour ce faire, il faut comprendre

- l'influence des erreurs sur le résultat d'une suite d'opérations effectuées selon un algorithme ;
- qu'est-ce un algorithme du point de vue de l'analyse numérique ;

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***

Peut-on éviter l'erreur ? Réponse : <u>NON</u>

Peut-on réduire l'erreur ? Réponse : <u>OUI</u> sous certaines conditions dont une essentielle : ***Programmer différemment***.

Pour ce faire, il faut comprendre

- l'influence des erreurs sur le résultat d'une suite d'opérations effectuées selon un algorithme ;
- qu'est-ce un algorithme du point de vue de l'analyse numérique ;
- de quelle manière une erreur de calcul se propage lors du déroulement d'un algorithme,

Programmer différemment

CONCLUSION ***Les calculs sur ordinateur donnent des résultats erronés !***
Peut-on éviter l'erreur ? Réponse : NON
Peut-on réduire l'erreur ? Réponse : OUI sous certaines conditions dont une essentielle : ***Programmer différemment***.

Pour ce faire, il faut comprendre

- l'influence des erreurs sur le résultat d'une suite d'opérations effectuées selon un algorithme ;
- qu'est-ce un algorithme du point de vue de l'analyse numérique ;
- de quelle manière une erreur de calcul se propage lors du déroulement d'un algorithme, et
- en conclusion, pourquoi un algorithme est meilleur, du point de vue numérique qu'un autre.

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Soit la fonction $y = f(x)$ calculée pour un réel x et soit $fl(x) = x + \Delta x$ le nombre-machine correspondant à x.

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Soit la fonction $y = f(x)$ calculée pour un réel x et soit $fl(x) = x + \Delta x$ le nombre-machine correspondant à x.
En utilisant la formule de Taylor nous avons

$$fl(y) - y = f\left(x + \Delta x\right) - f\left(x\right) = f'\left(x\right)\Delta x + \mathcal{O}\left(\left(\Delta x\right)^2\right) \ , \qquad \xi \in \left]0,1\right[$$

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Soit la fonction $y = f(x)$ calculée pour un réel x et soit $fl(x) = x + \Delta x$ le nombre-machine correspondant à x.

En utilisant la formule de Taylor nous avons

$$fl(y) - y = f\left(x + \Delta x\right) - f\left(x\right) = f'\left(x\right)\Delta x + \mathcal{O}\left(\left(\Delta x\right)^2\right) \ , \qquad \xi \in \left]0, 1\right[$$

d'où nous obtenons

$$\frac{fl(y)-y}{y} = \frac{xf'(x)}{f(x)} \times \frac{\Delta x}{x} + \mathcal{O}\left(\left(\Delta x\right)^2\right)$$

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Soit la fonction $y = f(x)$ calculée pour un réel x et soit $fl(x) = x + \Delta x$ le nombre-machine correspondant à x.
En utilisant la formule de Taylor nous avons

$$fl(y) - y = f\left(x + \Delta x\right) - f\left(x\right) = f'\left(x\right)\Delta x + \mathcal{O}\left(\left(\Delta x\right)^2\right) \quad , \qquad \xi \in \left]0, 1\right[$$

d'où nous obtenons

$$\frac{fl(y)-y}{y} = \frac{xf'(x)}{f(x)} \times \frac{\Delta x}{x} + \mathcal{O}\left(\left(\Delta x\right)^2\right)$$

$$\Rightarrow \quad \boxed{\text{nombre-condition} = \frac{xf'(x)}{f(x)}}$$

Nombre-condition

On utilisera deux notions : le ***nombre-condition*** et la ***stabilité***.

Soit la fonction $y = f(x)$ calculée pour un réel x et soit $fl(x) = x + \Delta x$ le nombre-machine correspondant à x.
En utilisant la formule de Taylor nous avons
$$fl(y) - y = f\left(x + \Delta x\right) - f\left(x\right) = f'\left(x\right)\Delta x + \mathcal{O}\left(\left(\Delta x\right)^2\right) \quad , \qquad \xi \in \left]0,1\right[$$
d'où nous obtenons
$$\frac{fl(y)-y}{y} = \frac{xf'(x)}{f(x)} \times \frac{\Delta x}{x} + \mathcal{O}\left(\left(\Delta x\right)^2\right)$$

$$\Rightarrow \quad \boxed{\text{nombre-condition} = \frac{xf'(x)}{f(x)}}$$

dont la valeur conditionne la valeur de l'erreur relative du résultat $\dfrac{fl\left(y\right) - y}{y}$ et qui est indépendante de la méthode utilisée pour calculer f.

Stabilité

La ***stabilité*** d'un algorithme exprime le fait que pour des petites variations des valeurs des entrées de l'algorithme, les valeurs de sortie de cet algorithme présentent aussi des petites variations.

Stabilité

La ***stabilité*** d'un algorithme exprime le fait que pour des petites variations des valeurs des entrées de l'algorithme, les valeurs de sortie de cet algorithme présentent aussi des petites variations.

Pour son étude il faut décomposer l'algorithme ϕ pour le calcul de la fonction f, en transformations élémentaires $\phi^{(k)}; k =, 1, \ldots, r$, c'est-à-dire $\phi = \phi^{(r)} \circ \phi^{(r-1)} \circ \cdots \circ \phi^{(1)}$ et évaluer le nombre-condition pour chaque transformation. S'il n'y pas de transformation mal conditionnée, l'algorithme est stable.

Stabilité

La ***stabilité*** d'un algorithme exprime le fait que pour des petites variations des valeurs des entrées de l'algorithme, les valeurs de sortie de cet algorithme présentent aussi des petites variations.

Pour son étude il faut décomposer l'algorithme ϕ pour le calcul de la fonction f, en transformations élémentaires $\phi^{(k)}; k =, 1, \ldots, r$, c'est-à-dire $\phi = \phi^{(r)} \circ \phi^{(r-1)} \circ \cdots \circ \phi^{(1)}$ et évaluer le nombre-condition pour chaque transformation. S'il n'y pas de transformation mal conditionnée, l'algorithme est stable.

Notons qu'une transformation élémentaire est une transformation qui se limite à une seule opération arithmétique. Par exemple si nous avons à calculer la fonction $f(x) = ax + b$, nous avons deux transformations élémentaires :
$\phi^{(1)}$: $t_1 \leftarrow a \times x$ $\phi^{(2)}$: $t_2 \leftarrow t_1 + b$ de sorte que $f = \phi^{(2)} \circ \phi^{(1)}$.

Stabilité

La ***stabilité*** d'un algorithme exprime le fait que pour des petites variations des valeurs des entrées de l'algorithme, les valeurs de sortie de cet algorithme présentent aussi des petites variations.

Pour son étude il faut décomposer l'algorithme ϕ pour le calcul de la fonction f, en transformations élémentaires $\phi^{(k)}; k =, 1, \ldots, r$, c'est-à-dire $\phi = \phi^{(r)} \circ \phi^{(r-1)} \circ \cdots \circ \phi^{(1)}$ et évaluer le nombre-condition pour chaque transformation. S'il n'y pas de transformation mal conditionnée, l'algorithme est stable.

Notons qu'une transformation élémentaire est une transformation qui se limite à une seule opération arithmétique. Par exemple si nous avons à calculer la fonction $f(x) = ax + b$, nous avons deux transformations élémentaires :
$\phi^{(1)} : t_1 \leftarrow a \times x \quad \phi^{(2)} : t_2 \leftarrow t_1 + b$ de sorte que $f = \phi^{(2)} \circ \phi^{(1)}$.
La stabilité d'un algorithme dépend de l'algorithme pour le calcul de f, c'est-à-dire de la méthode utilisée pour son calcul.

Qu'est-ce un algorithme ?

Qu'est-ce un algorithme ?

Établissons l'algorithme pour le calcul de la valeur $y = a + b + a \times b$.

Qu'est-ce un algorithme ?

Établissons l'algorithme pour le calcul de la valeur $y = a + b + a \times b$.

Un algorithme est la transformation d'un ensemble d'entrées en un ensemble de résultats.

Qu'est-ce un algorithme ?

Établissons l'algorithme pour le calcul de la valeur $y = a + b + a \times b$.

Un algorithme est la transformation d'un ensemble d'entrées en un ensemble de résultats.

Entrées : Deux valeurs $a \in [0, 1]$ et $b \in \mathbb{R}$.

Qu'est-ce un algorithme ?

Établissons l'algorithme pour le calcul de la valeur $y = a + b + a \times b$.

Un algorithme est la transformation d'un ensemble d'entrées en un ensemble de résultats.

Entrées : Deux valeurs $a \in [0, 1]$ et $b \in \mathbb{R}$.

Résultat : $y = a + b + a \times b$.

Qu'est-ce un algorithme ?

Établissons l'algorithme pour le calcul de la valeur $y = a + b + a \times b$.

Un algorithme est la transformation d'un ensemble d'entrées en un ensemble de résultats.

Entrées : Deux valeurs $a \in [0, 1]$ et $b \in \mathbb{R}$.

Résultat : $y = a + b + a \times b$.

Algorithme de calcul :

```
s <- a + b ; t <- a * b ; y <- s + t
```

Analyse de l'algorithme I

`s <- a + b; t <- a * b; y <- s + t`

Analyse de l'algorithme I

`s <- a + b;` `t <- a * b; y <- s + t`

Les entrées forment un vecteur $\mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \in D = D_0 = \begin{bmatrix} [0,1] \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^2$

Analyse de l'algorithme I

`s <- a + b; t <- a * b; y <- s + t`

Les entrées forment un vecteur $\mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \in D = D_0 = \begin{bmatrix} [0,1] \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^2$

1e étape : `s <- a + b;`

Analyse de l'algorithme I

`s <- a + b;` `t <- a * b; y <- s + t`

Les entrées forment un vecteur $\mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \in D = D_0 = \begin{bmatrix} [0,1] \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^2$

1e étape : `s <- a + b;` À la fin de cette étape on se retrouve avec le vecteur $\mathbf{x}^{(1)} = \begin{bmatrix} a \\ b \\ s \end{bmatrix} \in D_1 = \begin{bmatrix} [0,1] \\ \mathbb{R} \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^3$

Analyse de l'algorithme I

`s <- a + b;` `t <- a * b; y <- s + t`

Les entrées forment un vecteur $\mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \in D = D_0 = \begin{bmatrix} [0,1] \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^2$

1e étape : `s <- a + b;` À la fin de cette étape on se retrouve avec le vecteur $\mathbf{x}^{(1)} = \begin{bmatrix} a \\ b \\ s \end{bmatrix} \in D_1 = \begin{bmatrix} [0,1] \\ \mathbb{R} \\ \mathbb{R} \end{bmatrix} \subset \mathbb{R}^3$

$\Rightarrow$ Cette étape réalise donc une *application élémentaire*

$$\phi^{(1)} : D_0 \to D_1 \text{ telle que } \phi^{(1)}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(1)}$$

Analyse de l'algorithme II

```
s <- a + b;  t <- a * b; y <- s + t
```

Analyse de l'algorithme II

`s <- a + b;  t <- a * b; y <- s + t`

2e étape : `t <- a * b;`

Analyse de l'algorithme II

`s <- a + b;` <u>`t <- a * b;`</u> `y <- s + t`

2e étape : `t <- a * b;`
À la fin de cette étape on se retrouve avec le vecteur

$$\mathbf{x}^{(2)} = \begin{bmatrix} s \\ t \end{bmatrix} \in D_2 = \begin{bmatrix} \mathbb{R} \\ \mathbb{R} \end{bmatrix} = \mathbb{R}^2$$

Analyse de l'algorithme II

`s <- a + b ;` <u>`t <- a * b ;`</u> `y <- s + t`

2e étape : `t <- a * b ;`
À la fin de cette étape on se retrouve avec le vecteur

$$\mathbf{x}^{(2)} = \begin{bmatrix} s \\ t \end{bmatrix} \in D_2 = \begin{bmatrix} \mathbb{R} \\ \mathbb{R} \end{bmatrix} = \mathbb{R}^2$$

$\Rightarrow$ On a effectué l'application élémentaire

$$\phi^{(2)} : D_1 \longrightarrow D_2 \text{ telle que } \phi^{(2)}\left(\mathbf{x}^{(1)}\right) = \mathbf{x}^{(2)}$$

Analyse de l'algorithme III

`s <- a + b; t <- a * b; y <- s + t`

Analyse de l'algorithme III

```
s <- a + b; t <- a * b;  y <- s + t
```

3e étape : `y <- s + t`

Analyse de l'algorithme III

`s <- a + b; t <- a * b;` <u>`y <- s + t`</u>

3e étape : `y <- s + t`
À la fin de cette étape on se retrouve avec le vecteur

$$\mathbf{x}^{(3)} = [y] \in \mathbb{R}$$

Analyse de l'algorithme III

`s <- a + b; t <- a * b; y <- s + t`

3e étape : `y <- s + t`
À la fin de cette étape on se retrouve avec le vecteur

$$\mathbf{x}^{(3)} = [y] \in \mathbb{R}$$

$\Rightarrow$ On a effectué l'application élémentaire

$$\phi^{(3)} : D_2 \longrightarrow D_3 \text{ telle que } \phi^{(3)}\left(\mathbf{x}^{(2)}\right) = \mathbf{x}^{(3)}$$

Analyse de l'algorithme IV

Récapitulons : ***L'algorithme se resume à la succession des transformations selon le schéma suivant***

$$D_0 \ni \mathbf{x} = \mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\phi^{(1)}} \mathbf{x}^{(1)} = \begin{bmatrix} a \\ b \\ s \end{bmatrix} \xrightarrow{\phi^{(2)}} \mathbf{x}^{(2)} = \begin{bmatrix} s \\ t \end{bmatrix} \xrightarrow{\phi^{(3)}} \mathbf{x}^{(3)} = \mathbf{y} = [y] \in \mathbb{R}$$

Analyse de l'algorithme IV

Récapitulons : ***L'algorithme se resume à la succession des transformations selon le schéma suivant***

$$D_0 \ni \mathbf{x} = \mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\phi^{(1)}} \mathbf{x}^{(1)} = \begin{bmatrix} a \\ b \\ s \end{bmatrix} \xrightarrow{\phi^{(2)}} \mathbf{x}^{(2)} = \begin{bmatrix} s \\ t \end{bmatrix} \xrightarrow{\phi^{(3)}} \mathbf{x}^{(3)} = \mathbf{y} = [y] \in \mathbb{R}$$

qu'on peut écrire

$$\boldsymbol{\phi} : D_0 \to \mathbb{R} \text{ définie par } D_0 \ni \mathbf{x} \mapsto \boldsymbol{\phi}(\mathbf{x}) = \mathbf{y} \in \mathbb{R}$$

Analyse de l'algorithme IV

Récapitulons : ***L'algorithme se resume à la succession des transformations selon le schéma suivant***

$$D_0 \ni \mathbf{x} = \mathbf{x}^{(0)} = \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\phi^{(1)}} \mathbf{x}^{(1)} = \begin{bmatrix} a \\ b \\ s \end{bmatrix} \xrightarrow{\phi^{(2)}} \mathbf{x}^{(2)} = \begin{bmatrix} s \\ t \end{bmatrix} \xrightarrow{\phi^{(3)}} \mathbf{x}^{(3)} = \mathbf{y} = [y] \in \mathbb{R}$$

qu'on peut écrire

$$\boldsymbol{\phi} : D_0 \longrightarrow \mathbb{R} \text{ définie par } D_0 \ni \mathbf{x} \longmapsto \boldsymbol{\phi}\left(\mathbf{x}\right) = \mathbf{y} \in \mathbb{R}$$

ou, en généralisant

$$\mathbb{R}^n \supseteq D_0 \ni \mathbf{x} = \mathbf{x}^{(0)} \longrightarrow \boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right) = \mathbf{x}^{(1)} \longrightarrow \cdots \longrightarrow \boldsymbol{\phi}^{(r)}\left(\mathbf{x}^{(r-1)}\right) = \mathbf{x}^{(r)} = \mathbf{y} \in D_r \subseteq \mathbb{R}^m$$

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.
Et si « *les voyages forment la jeunesse* »,

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.

Et si « *les voyages forment la jeunesse* », on peut aussi dire que

« les applications élémentaires forment l'erreur ».

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.
Et si « *les voyages forment la jeunesse* », on peut aussi dire que
« ***les applications élémentaires forment l'erreur*** ».
Mais si Léo Ferré, dans « Madre de Dios », ajoutait
« *T'en fais pas mon ami j'vieillirai* »,

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.
Et si « *les voyages forment la jeunesse* », on peut aussi dire que
« les applications élémentaires forment l'erreur ».
Mais si Léo Ferré, dans « Madre de Dios », ajoutait
« *T'en fais pas mon ami j'vieillirai* », l'erreur pourrait nous dire
« T'en fais pas mon ami je t'enquiquinerai ».

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.

Et si « *les voyages forment la jeunesse* », on peut aussi dire que

« les applications élémentaires forment l'erreur ».

Mais si Léo Ferré, dans « Madre de Dios », ajoutait

« *T'en fais pas mon ami j'vieillirai* », l'erreur pourrait nous dire

« T'en fais pas mon ami je t'enquiquinerai ».

En effet, lors d'une application élémentaire $\boldsymbol{\phi}^{(k)}$, nous avons, à la place de la valeur $\mathbf{x}^{(k+1)} = \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$,

L'erreur qui voyage...

Dans cette succession d'applications élémentaires l'erreur de calcul, et aussi de représentation, voyage d'application en application laissant sa trace un peu partout.

Et si « *les voyages forment la jeunesse* », on peut aussi dire que

« les applications élémentaires forment l'erreur ».

Mais si Léo Ferré, dans « Madre de Dios », ajoutait

« *T'en fais pas mon ami j'vieillirai* », l'erreur pourrait nous dire

« T'en fais pas mon ami je t'enquiquinerai ».

En effet, lors d'une application élémentaire $\boldsymbol{\phi}^{(k)}$, nous avons, à la place de la valeur $\mathbf{x}^{(k+1)} = \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$, la valeur

$$\mathbf{fl}\left(\mathbf{x}^{(k+1)}\right) = \mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$$

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta \mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\boldsymbol{\Delta}\mathbf{x}^{(k+1)} = \mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta\mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\boldsymbol{\Delta}\mathbf{x}^{(k+1)} = \mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de

$$\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$$

Apprivoiser cette erreur sauvageonne I

L'erreur $\mathbf{\Delta x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\mathbf{\Delta x}^{(k+1)} = \mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de

$\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$.

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta \mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\mathbf{\Delta x}^{(k+1)} = \mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de $\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$.

⚠ **Danger :** *Risque de sombrer dans le chaos.*

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta\mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\boldsymbol{\Delta}\mathbf{x}^{(k+1)} = \mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de

$$\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] \text{ à } \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right).$$

⚠ **Danger :** *Risque de sombrer dans le chaos.*

Astuce : Au lieu d'un saut, en faire deux plus petits, à savoir pour aller de $\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$,

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta\mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\boldsymbol{\Delta}\mathbf{x}^{(k+1)} = \mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de

$\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$.

Danger : *Risque de sombrer dans le chaos.*

Astuce : Au lieu d'un saut, en faire deux plus petits, à savoir pour aller de $\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)$,

aller d'abord de $\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]$

Apprivoiser cette erreur sauvageonne I

L'erreur $\Delta\mathbf{x}^{(k+1)} = \mathbf{fl}(\mathbf{x}^{(k+1)}) - \mathbf{x}^{(k+1)}$ devient

$$\boldsymbol{\Delta}\mathbf{x}^{(k+1)} = \mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$$

Le 2e membre de cette relation représente un très grand saut : on doit passer de

$\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$.

Danger : *Risque de sombrer dans le chaos.*

Astuce : Au lieu d'un saut, en faire deux plus petits, à savoir pour aller de $\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$,

aller d'abord de $\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right]$ à $\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]$

et ensuite de $\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]$ à $\phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)$

Apprivoiser cette erreur sauvageonne II

Nous devons donc calculer la quantité :

$$\begin{aligned}\boldsymbol{\Delta}\mathbf{x}^{(k+1)} &= \left[\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] + \\ &\quad \left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)\right]\end{aligned}$$

Apprivoiser cette erreur sauvageonne II

Nous devons donc calculer la quantité :

$$\begin{aligned}\boldsymbol{\Delta}\mathbf{x}^{(k+1)} &= \left[\mathbf{fl}\left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] + \\ &\quad \left[\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)\right]\end{aligned}$$

$$\Rightarrow\ \mathbf{fl}\left(\phi^{(k+1)}\left(\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right)\right) - \phi^{(k+1)}\left(\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right) = \mathbf{H}_{k+1}\cdot\mathbf{x}^{(k+1)}$$

et

$$\phi^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right] - \phi^{(k+1)}\left(\mathbf{x}^{(k)}\right) \simeq J\phi^{(k+1)}\left(\mathbf{x}^{(k)}\right)\boldsymbol{\Delta}\mathbf{x}^{(k)}$$

où $\mathbf{H}_{k+1}\cdot\mathbf{x}^{(k+1)}$ est l'erreur absolue entre le résultat de la $(k+1)$-ième application élémentaire et sa représentation machine.Les éléments diagonaux de $\mathbf{H}_{k+1}$ représentent les erreurs relatives correspondantes. $J\phi^{(k+1)}$ est le jacobien de ϕ.

Apprivoiser cette erreur sauvageonne II

Nous devons donc calculer la quantité :

$$\begin{aligned}\mathbf{\Delta x}^{(k+1)} &= \left[\mathbf{fl}\left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] - \boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right]\right] + \\ &\quad \left[\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right] - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)\right]\end{aligned}$$

$$\Rightarrow\ \mathbf{fl}\left(\boldsymbol{\phi}^{(k+1)}\left(\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right)\right) - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right) = \mathbf{H}_{k+1}\cdot\mathbf{x}^{(k+1)}$$

et

$$\boldsymbol{\phi}^{(k+1)}\left[\mathbf{fl}\left(\mathbf{x}^{(k)}\right)\right] - \boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right) \simeq J\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)\mathbf{\Delta x}^{(k)}$$

où $\mathbf{H}_{k+1}\cdot\mathbf{x}^{(k+1)}$ est l'erreur absolue entre le résultat de la $(k+1)$-ième application élémentaire et sa représentation machine.Les éléments diagonaux de $\mathbf{H}_{k+1}$ représentent les erreurs relatives correspondantes. $J\boldsymbol{\phi}^{(k+1)}$ est le jacobien de $\boldsymbol{\phi}$.

Donc finalement

$$\mathbf{\Delta x}^{(k+1)} \simeq J\boldsymbol{\phi}^{(k+1)}\left(\mathbf{x}^{(k)}\right)\cdot\mathbf{\Delta x}^{(k)} + \mathbf{H}_{k+1}\cdot\mathbf{x}^{(k+1)} \quad ; \quad \mathbf{\Delta x}^{(0)} = \mathbf{\Delta}x$$

Apprivoiser cette erreur sauvageonne III

$$\mathbf{\Delta x}^{(0)} = \mathbf{\Delta} x$$

Apprivoiser cette erreur sauvageonne III

$$\begin{aligned}
\mathbf{\Delta x}^{(0)} &= \mathbf{\Delta} x \\
\mathbf{\Delta x}^{(1)} &\simeq \mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right) \cdot \mathbf{\Delta x}^{(0)} + \mathbf{H}_1 \cdot \mathbf{x}^{(1)}
\end{aligned}$$

Apprivoiser cette erreur sauvageonne III

$$\begin{array}{lcl}
\boldsymbol{\Delta}\mathbf{x}^{(0)} & = & \boldsymbol{\Delta}x \\
\boldsymbol{\Delta}\mathbf{x}^{(1)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(0)} + \mathbf{H}_1\cdot\mathbf{x}^{(1)} \\
\boldsymbol{\Delta}\mathbf{x}^{(2)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(0)} + \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\boldsymbol{\Delta}\mathbf{x} + \mathbf{J}\boldsymbol{\psi}^{(1)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
\vdots & \vdots & \vdots
\end{array}$$

Apprivoiser cette erreur sauvageonne III

$$
\begin{array}{lcl}
\mathbf{\Delta x}^{(0)} & = & \mathbf{\Delta} x \\
\mathbf{\Delta x}^{(1)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\mathbf{H}_1\cdot\mathbf{x}^{(1)} \\
\mathbf{\Delta x}^{(2)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{\Delta x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\mathbf{\Delta x}+\mathbf{J}\boldsymbol{\psi}^{(1)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
\vdots & \vdots & \vdots \\
\mathbf{\Delta x}^{(r)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(r)}\left(\mathbf{x}^{(r-1)}\right)\cdot\mathbf{\Delta x}^{(r-1)}+\mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)}+\mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\mathbf{\Delta x}+\sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)}+\mathbf{H}_r\cdot\mathbf{y}
\end{array}
$$

Apprivoiser cette erreur sauvageonne III

$$
\begin{array}{lcl}
\boldsymbol{\Delta}\mathbf{x}^{(0)} & = & \boldsymbol{\Delta}x \\
\boldsymbol{\Delta}\mathbf{x}^{(1)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(0)} + \mathbf{H}_1\cdot\mathbf{x}^{(1)} \\
\boldsymbol{\Delta}\mathbf{x}^{(2)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(0)} + \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\boldsymbol{\Delta}\mathbf{x} + \mathbf{J}\boldsymbol{\psi}^{(1)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)} + \mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
\vdots & \vdots & \vdots \\
\boldsymbol{\Delta}\mathbf{x}^{(r)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(r)}\left(\mathbf{x}^{(r-1)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(r-1)} + \mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}^{(0)}\right)\cdot\boldsymbol{\Delta}\mathbf{x}^{(0)} + \sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)} + \mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\boldsymbol{\Delta}\mathbf{x} + \sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)} + \mathbf{H}_r\cdot\mathbf{y}
\end{array}
$$

où on a noté $\psi^{(i)}\left(\mathbf{x}^{(i)}\right) = \phi^{(r)}\circ\phi^{(r-1)}\circ\cdots\circ\phi^{(i+1)}\left(\mathbf{x}^{(i)}\right)$

Apprivoiser cette erreur sauvageonne III

$$
\begin{array}{lcl}
\mathbf{\Delta x}^{(0)} & = & \mathbf{\Delta} x \\
\mathbf{\Delta x}^{(1)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\mathbf{H}_1\cdot\mathbf{x}^{(1)} \\
\mathbf{\Delta x}^{(2)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{\Delta x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\cdot\mathbf{J}\boldsymbol{\phi}^{(1)}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\mathbf{J}\boldsymbol{\phi}^{(2)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\mathbf{\Delta x}+\mathbf{J}\boldsymbol{\psi}^{(1)}\left(\mathbf{x}^{(1)}\right)\mathbf{H}_1\cdot\mathbf{x}^{(1)}+\mathbf{H}_2\cdot\mathbf{x}^{(2)} \\
\vdots & \vdots & \vdots \\
\mathbf{\Delta x}^{(r)} & \simeq & \mathbf{J}\boldsymbol{\phi}^{(r)}\left(\mathbf{x}^{(r-1)}\right)\cdot\mathbf{\Delta x}^{(r-1)}+\mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & \simeq & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}^{(0)}\right)\cdot\mathbf{\Delta x}^{(0)}+\sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)}+\mathbf{H}_r\cdot\mathbf{x}^{(r)} \\
 & = & \mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\mathbf{\Delta x}+\sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)}+\mathbf{H}_r\cdot\mathbf{y}
\end{array}
$$

où on a noté $\psi^{(i)}\left(\mathbf{x}^{(i)}\right)=\phi^{(r)}\circ\phi^{(r-1)}\circ\cdots\circ\phi^{(i+1)}\left(\mathbf{x}^{(i)}\right)$Finalement on a

$$
\mathbf{\Delta y}=\mathbf{J}\boldsymbol{\phi}\left(\mathbf{x}\right)\cdot\mathbf{\Delta x}+\sum_{i=1}^{r-1}\mathbf{J}\boldsymbol{\psi}^{(i)}\left(\mathbf{x}^{(i)}\right)\mathbf{H}_i\cdot\mathbf{x}^{(i)}+\mathbf{H}_r\cdot\mathbf{y}
$$

Apprivoiser cette erreur sauvageonne IV

On arrive, enfin, à enfermer l'erreur totale d'un algorithme donné A dans une formule

$$E_r(A,\mathbf{x}) = J\boldsymbol{\psi}^{(1)}(\mathbf{x}^{(1)})\cdot\mathbf{H}_1\cdot\mathbf{x}^{(1)} + \cdots + J\boldsymbol{\psi}^{(r-1)}\left(\mathbf{x}^{(r-1)}\right)\cdot\mathbf{H}_{r-1}\cdot\mathbf{x}^{(r-1)} + \mathbf{H}_r y$$

Apprivoiser cette erreur sauvageonne IV

On arrive, enfin, à enfermer l'erreur totale d'un algorithme donné A dans une formule

$$E_r(A, \mathbf{x}) = J\boldsymbol{\psi}^{(1)}(\mathbf{x}^{(1)}) \cdot \mathbf{H}_1 \cdot \mathbf{x}^{(1)} + \cdots + J\boldsymbol{\psi}^{(r-1)}\left(\mathbf{x}^{(r-1)}\right) \cdot \mathbf{H}_{r-1} \cdot \mathbf{x}^{(r-1)} + \mathbf{H}_r y$$

Cette formule est utile pour la comparaison des erreurs de deux algorithmes différents.