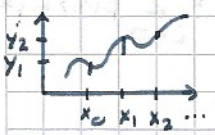
le 20.09.18

Analyse Num:

CR 1. Interpolation polynomiale.

1.1 Les polynômes de Lagrange et formule de Newton:

Soit $f : [a,b] \longrightarrow \mathbb{R}$ connue en certain points $x_0, \dots, x_n \in [a,b]$ avec des valeurs $y_i = f(x_i)$.



Question: Trouver un polynôme $p(x)$ qui passe par les $(x_i; y_i)$, $i = 0, \dots, n$

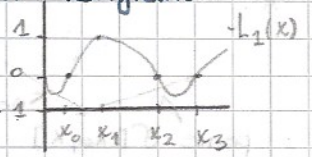
TRm: Soit $n+1$ points $(x_0, y_0) \dots (x_n, y_n)$, avec les x_i différent 2 à 2, alors $\exists!$ polynôme de degré $\leq n$, appelé polynôme d'interpolation (de Lagrange) tq : $p(x_i) = y_i$, $i = 0, \dots, n$

Preuve: On définit les polynômes de Lagrange :

- pour $n = 0$, $L_0(x) = 1$
- pour $n > 0$, $L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$, pour $i = 0, \dots, n$

dont le degré de chacun est n vérifiant :

$$L_i(x_k) = \begin{cases} 1 & \text{si } i = k \\ 0 & \text{si } i \neq k \end{cases}$$



alors le polynôme $p_n(x) = \sum_{i=0}^{n} y_i L_i(x)$ est de degré $\leq n$ et passe bien par les points $(x_j; y_j)$ $j = 0, 1, \dots, n$

Pour l'unicité ; soient p et q 2 quandidats. On pose $d(x) = q(x) - p(x)$, le polynôme $d(x)$ est de degré $\leq n$ et il vérifie $d(x_i) = q(x_i) - p(x_i) = 0$ $\forall i = 0, \dots$

Il possède donc $n+1$ racines au moins, donc $d = 0$ car un polynôme non-nul possède exactement autant de racine que son degré. □

On a montré la formule de Lagrange :

$$p_n(x) = \sum_{i=0}^{n} y_i L_i(x) \text{ , avec } L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

Rem: On a en fait que $\mathbb{P}_n$, l'espace vectoriel des polynômes de degré $\leq n$ a pour base $\{L_i,\ i = 0 \dots n\}$

En effet, pour $p \in \mathbb{P}_n$, on a la décomposition $p(x) = \sum_{i=0}^{n} p(x_i) L_i(x)$

Notation: $\mathbb{P}_n = \text{Vect}\{L_0, \dots L_n\}$

$\{L_0, \dots, L_n\}$ est une base différente de la base $\{1, x, \dots, x^n\}$

C'est la base de Lagrange de $\mathbb{P}_n$.

Une autre approche par les différences divisées :

Déf: Soit (x_i, y_i), $i = 0, \dots, n$ tq $x_i \neq x_j$ si $i \neq j$. On définit

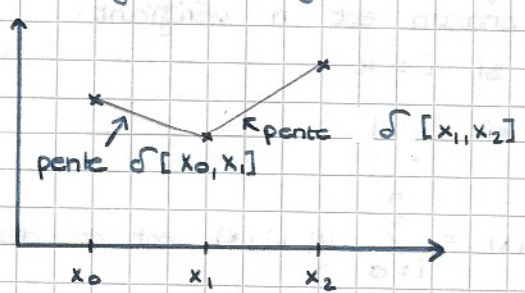
$$\delta y[x_i; x_{i+1}] = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad \text{et pour } k = 2, 3, \dots$$

$$\delta^k y[x_i, \dots; x_{i+k}] = \frac{\delta^{k-1} y[x_{i+1} \dots x_{i+k}] - \delta^{k-1} y[x_i, \dots, x_{k-1}]}{x_{i+k} - x_i}$$

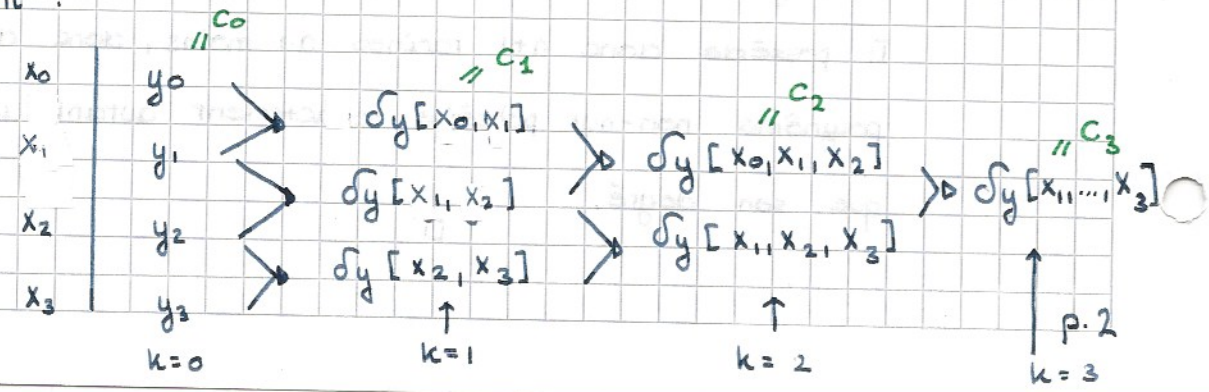
On peut aussi pour simplifier les notations :

$$\delta^0 y[x_i] = y_i \qquad \delta^1 y[x_i; x_{i+1}] = \delta y[x_i; x_{i+1}]$$

Ex :



Methode: Pour calculer les différences divisées, on utilise le schéma suivant :



Analyse Num: le 20.09.18

Thm: (Formule de Newton)

Soit $p_n(x) = c_0 + c_1(x - x_0) + c_2(x-x_0)(x-x_1) + \dots + c_n(x-x_0)\dots(x-x_{n-1})$

L'unique polynôme de degré $\leq n$ passant par $(x_i; y_i)$ $i = 0, \dots n$

(x_i distincts 2 à 2). Alors : $c_k = \delta^k y\,[x_0, \dots, x_k]$ $k = 0, \dots, n$

Remarque: La formule de Newton s'appuie sur la base de Newton de $\mathbb{P}_n$:

$\{1;\ (x - x_0);\ (x-x_0)(x-x_1);\ \dots\dots;\ (x-x_0)\dots(x-x_{n-1})\}$

Preuve: Par récurrence

Pour $n = 0$ (évident) $p_n(x) = y_0$

On suppose le résultat vrai au rang $n-1$

On considère p_{n-1}, le polynôme de degré $\leq n-1$ passant par $(x_i; y_i)$ $i = 1, \dots, n-1$

Par hypothèse de récurrence, $p_{n-1}(x) = \delta y^0[x_0] + \delta y[x_0; x_1](x-x_0) +$

$+ \dots + \delta^{n-1} y\,[x_0, \dots, x_{n-1}](x-x_0)\dots(x-x_{n-1})$

On remarque qu'il existe α tq : $p_n(x) = p_{n-1}(x) + \alpha(x-x_0)\dots(x-x_{n-1})$

En effet α est choisi tq $p_n(x_n) = y_n$

Il reste à mq $\alpha = c_n = \delta^n y\,[x_0, \dots, x_n]$

On considère q_{n-1} de degré $\leq n-1$ qui passe par (x_i, y_i) $i = 1, \dots, n$

$q_{n-1}(x) = y[x_1] + \delta y[x_1, x_2](x-x_1) + \dots\dots + \delta^{n-1} y\,[x_1 \dots x_n](x-x_1)\dots(x-x_{n-1})$

On remarque que p_n peut s'écrire comme :

$$p_n(x) = \frac{x_n - x}{x_n - x_0}\, p_{n-1}(x) + \frac{x - x_0}{x_n - x_0}\, q_{n-1}(x)$$

Puisque α est le therme le + élevé : $p_n(x) = \alpha x^n + r_{n-1}(x)$
avec $r_{n-1} \in \mathbb{P}_{n-1}$

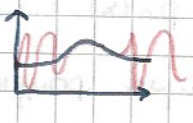
On obtient :

$$\alpha = -\frac{\delta^{n-1} y\,[x_0, \dots, x_{n-1}]}{x_n - x_0} + \frac{\delta^{n-1} y\,[x_1; \dots; x_n]}{x_n - x_0} = c_n$$

1.2 Erreur de l'interpolation:

le 27.09.18

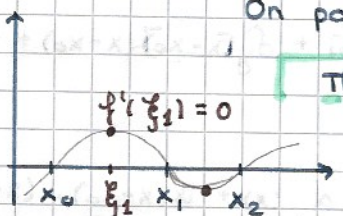
Observation:

	$f(x) = \sin(x)$	$f(x) = \frac{1}{1+x^2}$
subdivision uniforme		
subdivision Chebychev		

Lemme: Soit $f: [a,b] \longrightarrow \mathbb{R}$ n fois dérivable et $y_i = f(x_i)$ $x_0, \dots x_n \in [a,b]$ distincts 2 à 2. Alors $\exists\, \xi \in [a,b]$ tq $\delta y^n[x_0, \dots x_n] = \frac{f^{(n)}(\xi)}{n!}$

Preuve: Soit $p(x) = c_0 + c_1(x - x_0) + \dots + c_n(x - x_0) \cdots (x - x_{n-1})$ le polynôme d'interpolation de degré $\leq n$ passant par (x_i, y_i) $i \in [0,n]$

On pose $d(x) = f(x) - p(x)$, alors $d(x)$ s'annule en x_i, $i = 0, \dots n$



Thm de Rolle:

Si $f: [a,b] \longrightarrow \mathbb{R}$ est continue sur $[a,b]$ et dérivable sur (a,b) avec $f(a) = f(b)$, alors $\exists\, \xi \in (a,b)$ tq $f'(\xi) = 0$

en appliquant le thm de Rolle n fois à la fonction $d(x)$, on obtient que $d'(x)$ possède au moins n racines distinctes dans $[a,b]$

$d''(x)$ possède au moins $n-1$ racines, ..., $d^{(n)}(x)$ possède au moins 1 racine qu'on note ξ. $d^{(n)}(\xi) = 0 \Rightarrow f^{(n)}(\xi) = n! \cdot c_n = n!\, \delta f^n[x_0 \dots]$ □

Thm: Soit $f: [a,b] \to \mathbb{R}$ $n+1$ dérivable et p le polynôme d'interp. de deg $\leq n$ passant par les $(x_i; f(x_i))$ $i = 1, \dots, n$ avec les x_i distincts 2 à 2. Alors $\exists\, \xi \in (a,b)$ (qui dépend de x) tq $f(x) - p(x) = (x - x_0)(x - x_1) \dots (x - x_n) \cdot \frac{f^{(n+1)}(\xi)}{(n+1)!}$

Preuve:

La formule est vraie pour $x = x_i$ $i = 0, \dots, n$

On considère x distinct de x_i $i = 0, \dots, n$

On considère $\overline{p}$ le poly d'interpolation de deg $\leq n+1$ passant par les x_i $i = 0, \dots, n$ et par x.

Par la formule de Newton on a: $\overline{p}(y) = p(y) + (y - x_0) \cdots (y - x_n) \cdot \delta_y^{n+1}[x_0, \dots x_n, x]$.

Par le lemme, $\exists\, \xi \in (a,b)$ tq $\delta_y^{n+1}[x_0, \dots x_n, x] = \frac{f^{(n+1)}(\xi)}{(n+1)!}$.

On obtient $f(y) = p(y) + (y - x_0) \cdots (y - x_n) \cdot \frac{f^{(n+1)}(\xi)}{(n+1)!}$

et on fait $y = x$

Retour sur l'expérience numérique:

Si on considère $f = \sin$ avec les pt $0, 1, \dots, 6$, on obtient,

$$|p_6(x) - f(x)| \leq |x(x-1)\dots(x-6)| \frac{1}{7!}|$$

$$\max_{x \in [0,6]} |p_6(x) - \sin(x)| \leq 95{,}85 \cdot \frac{1}{7!} \leq 0{,}019 \quad \text{(petit)}$$

En revanche si $f(x) = \frac{1}{1+x^2}$ $\quad f^{(7)} = -40'320\left(\frac{x(x^6 - 7x^4 + 7x^2 - 1)}{(1+x^2)^8}\right)$

$$\max_{x \in [0,6]} |f^{(7)}(x)| \leq 4'392$$

$$\max_{x \in [0,6]} \left|p_7(x) - \frac{1}{1+x^2}\right| \leq \dots \leq 2'984$$

La série de Taylor associée à $f(x)$ ne peut pas converger à ∞ car $f^{(i)} = \infty$ elle converge sur S_1 $x \in [0,6]$

1.3 Polynômes de Chebyshev:

Un bon choix de subdivision est de minimiser $\max_{x \in [a,b]} |(x-x_0)\dots(x-x_n)|$

La réponse est donnée par les racines des poly. de Chebyshev.

Déf: les polynômes de Chebyshev $T_n(x)$ sont définit pour $n = 0, \dots$ par

$T_n(x) = \cos(n \cdot \arccos(x))$ pour tout $x \in [-1,1]$

Remarque: • De manière équivalente, pour $x = \cos\theta$ $T_n(x) = \cos(n\theta)$ $\forall \theta \in \mathbb{R}$

• T_n est en fait un poly de degré n définit sur $\mathbb{R}$.

Thm: a) T_n est un poly de deg n de la forme $T_n(x) = 2^{n-1}x^n + \dots$

b) T_n satisfait la formule de récurrence suivante $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$

$\forall x \in \mathbb{R}$, $\forall n \geq 1$ et $T_0(x) = 1$ et $T_1(x) = x$

d) $T_n\left(\cos\left((2k+1)\frac{\pi}{2n}\right)\right) = 0$ $k = 0, \dots, n-1$ n racines

c) $|T_n(x)| \leq 1$ $\forall x \in [-1,1]$

e) $T_n\left(\cos\frac{k\pi}{n}\right) = (-1)^k$ $k = 0, \dots, n$ extréma locaux.

Preuve: b) $\cos((n+1)\alpha) + \cos((n-1)\alpha) = 2\cos(\alpha)\cos(n\alpha)$

avec $\alpha = \arccos(x)$ pour $x \in [-1,1]$, on obtient

$T_{n+1}(x) = \cos((n+1)\alpha) = 2\cos\alpha\cos n\alpha - \cos((n-1)\alpha)$

$= 2xT_n(x) - T_{n-1}(x)$ □

Par récurence sur n, on obtient que $T_n(x)$ est un poly de deg n de la forme $T_n(x) = 2^{n-1}x^n + \dots$

Ceci montre a) et $T_n(x)$ est mtn définit pour $x \in \mathbb{R}$.

Lemme: Soit $n \geq 1$ et $q(x) = 2^{n-1}x^n + \dots$ avec $q(x) \not\equiv T_n(x)$

(ie: $\exists$ au moins 1 point où elles sont différentes)

alors : $\max_{x \in [-1,1]} |q(x)| > \max_{x \in [-1,1]} |T_n(x)| = 1$

le 27.09.18

Analyse Num:

Preuve: Mq si un polynôme $q(x) = 2^{n-1}x^n + \dots$ vérifie

$\max_{x \in [-1,1]} q(x) \leq 1$, alors $q \equiv T_n$ (q est le poly T_n)

On considère la différence $d = T_n - q$

$d(\cos\frac{k\pi}{n}) = (-1)^k - \underbrace{q(\cos\frac{k\pi}{n})}_{|\dots| \leq 1} = \begin{cases} \leq 0 & k \text{ impaire} \\ \geq 0 & k \text{ paire} \end{cases}$

Alors d s'annule au moins 1 fois dans chacun des intervalles (fermés) $\left[\cos\frac{(k+1)\pi}{n}, \cos\frac{k\pi}{n}\right]$ $k = 0, \dots, n-1$

S'il existe $\xi_j \in (-1,1)$ racine de d, à l'extrémité d'un intervalle alors $T_n(\xi_j) = q(\xi_j) = \pm 1$ et donc ξ_j est un extremum de T_n et de q, donc $T_n'(\xi_j) = q'(\xi_j) = 0$ et donc $d'(\xi_j) = 0$ et ξ_j est une racine de multiplicité 2 au moins.

En comptant les autres racines, d possède n racine (avec multipli)

Or d est de deg $\leq n-1$, donc $d \equiv 0$

la multiplicité d'une racine se trouve en regardant combien de dérivée n-ième s'annulent pour la racine.

Ce Lemme mq on arrive à minimiser $\max_{x \in [-1,1]} |(x-x_0)\cdots(x-x_n)|$

$\Leftrightarrow (x-x_0)\cdots(x-x_n) = \frac{1}{2^{n-1}} T_{n+1}(x)$ soit de manière équivalente,

(les x_i sont les racines de $T_{n+1}(x)$) $x_k = \cos\frac{(2k+1)\pi}{2n+2}$ (racine de T_{n+1})

le 04.10.18

On a mq $\max_{x \in [-1,1]} |(x-x_0)(x-x_1)\dots(x-x_n)| \longrightarrow \min$

si et seulement si $x_k = \cos\left(\frac{2k+1}{2n+2}\pi\right)$ $k = 0,\dots n$

et le minimum vaut alors : $1/2^n$

Pour un résultat analogue sur un intervalle $[a,b]$, on considère la transformation affine qui transforme $[-1,1]$ en $[a,b]$

$x \longmapsto \frac{a+b}{2} + \frac{b-a}{2} x$

Thm : L'expression $\max_{x \in [a,b]} |(x-x_0)\dots(x-x_n)|$ est minimale si et seulement si $x_k = \frac{a+b}{2} + \frac{b-a}{2}\cos\left(\frac{2k+1}{2n+2}\pi\right)$ $k = 0 \dots n$

Thm : Soit $f \in \mathcal{C}^2$ sur $[a,b]$ et p_n le polynôme d'interpolation passant par $x_k = \frac{a+b}{2} + \frac{b-a}{2}\cos\left(\frac{2k+1}{2n+2}\pi\right)$ $k = 0 \dots n$

p_n de degré $\leq n$.

Alors $\max_{x \in [a,b]} |f(x) - p_n(x)| \xrightarrow{n \to \infty} 0$

On dit que la suite $(p_n)_{n \in \mathbb{N}}$ converge uniformément vers f sur $[a,b]$.

⚠ À ne pas confondre avec la convergence ponctuelle ⚠

Ex : $f_n(x) = x^n$ définie sur $[0,1]$

$f_n(x) \xrightarrow[n \to \infty]{} f(x) = \begin{cases} 0 & \text{si } x < 1 \\ 1 & \text{si } x = 1 \end{cases}$

mais on peut mq $\max_{x \in [0,1]} |f_n(x) - f(x)| \not\longrightarrow 0$

1.4 Formules de barycentres :

On peut utiliser la formule de la grange pour évaluer numériquement le polynôme d'interpolation.
Une implémentation naïve coûte $O(n^2)$ opérations. On peut faire mieux avec seulement $O(n)$ opération.
(voir série (pour résumé))

1.5 Interpolation de Hermitte :

L'interp. de H. est une généralisation de l'interpolation de Lagrange, on fixe une subdivision $x_0, x_1 \dots x_n$ de $[a,b]$, (avec les x_i distincts 2 à 2) et on cherche un polynôme tel que :

$p(x_i) = y_i$ et $p'(x_i) = z_i$ $\quad i = 0, \dots, n$

Comme il y a $2n+2$ conditions, on peut supposer qu'il existe un unique polynôme de degré $\leq 2n+1$.

Thm : Soit $x_0, \dots x_n$ 2 à 2 distincts, et $y_0, \dots y_n$ et $z_0, \dots z_n$ fixés
Alors il existe un unique polynôme de deg $\leq 2n+1$ tel que

$p(x_i) = y_i$ et $p'(x_i) = z_i$

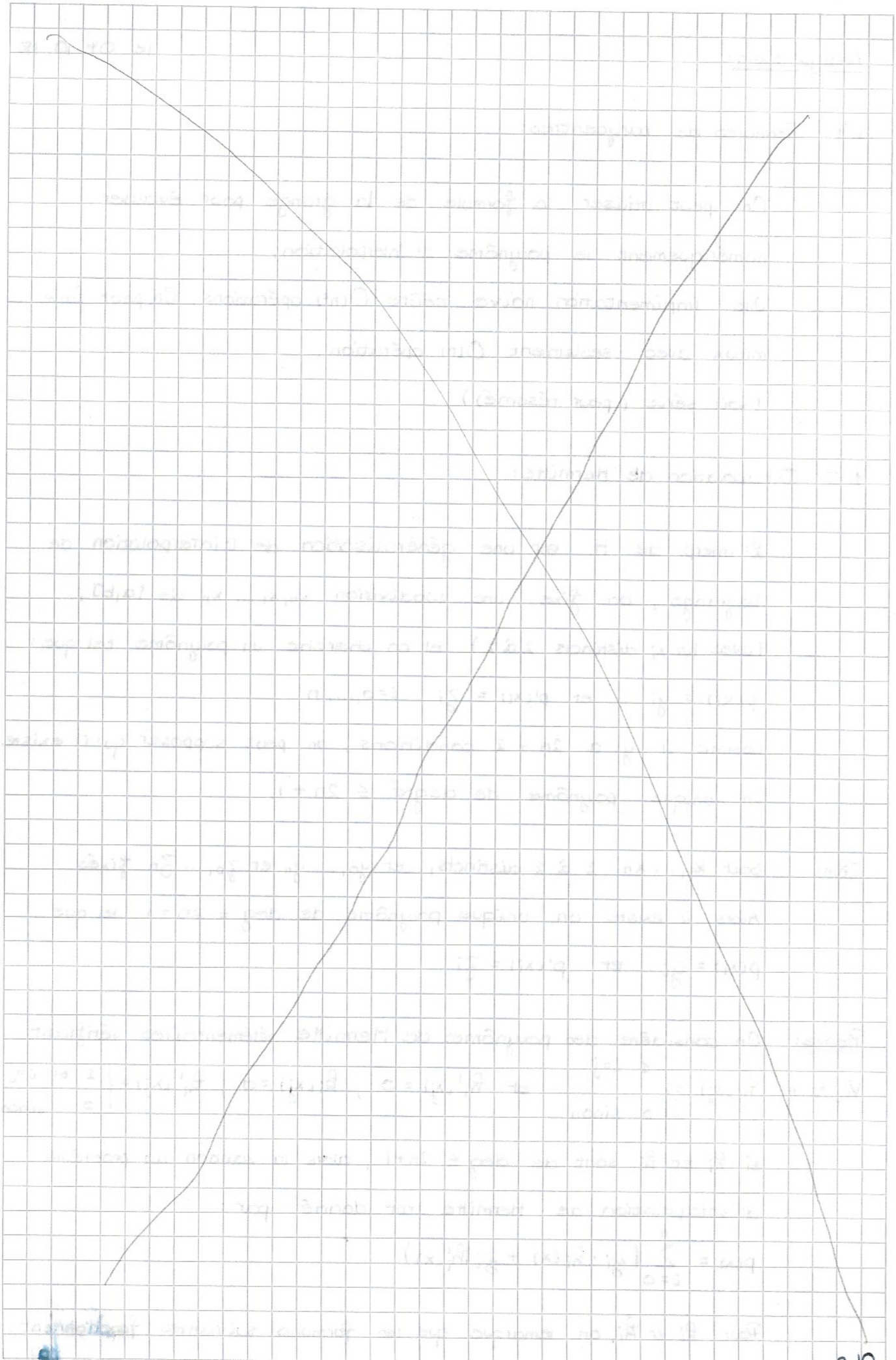
Preuve : On considère des polynômes de Hermitte élémentaires vérifiant

$\forall i,j = 0..n \quad h_i(x_j) = \begin{cases} 1 & i=j \\ 0 & \text{sinon} \end{cases}$, $\quad h_i'(x_j) = 0$, $\quad k_i(x_j) = 0$, $\quad k_i'(x_j) = \begin{cases} 1 & \text{si } i=j \\ 0 & \text{sinon} \end{cases}$

si h_i et k_i sont de deg $\leq 2n+1$, alors la solution au problème d'interpolation de Hermitte est donné par :

$$p(x) = \sum_{i=0}^{n} \left(y_i \cdot h_i(x) + z_i k_i(x) \right)$$

Pour h_i et k_i, on remarque que les formules suivantes fonctionnent :



Preuve : (suite)

$$h_i(x) := (l_i(x))^2 (1 - 2 L_i'(x_i)(x - x_i))$$

$$\tilde{h}_i(x) := (l_i(x))^2 (x - x_i)$$

Ceci montre l'existence.

Pour démontrer l'unicité, soit p et q 2 poly. d'interp. et on considère la différence $d(x) = p(x) - q(x)$ de deg $\leq 2n+1$ et $(n+1) \cdot 2$ racines au moins, avec multiplicité.

($d'(x_i) = d(x_i) = 0$ x_i est de multiplicité ≥ 2)

$\Rightarrow$ d est le polynôme nul.

Rem :
- Les h_i et $\tilde{h}_i$ (poly $\leq 2n+1$) dans la preuve précédente sont unique, on peut leur appliquer le thm.
- On peut utiliser / généraliser la formule de Newton pour calculer le polynôme d'interp. de Hermite.

Idée : On va construire $p(x)$ comme la limite pour $\varepsilon \to 0$ du polynôme d'interpolation de Lagrange de deg $\leq 2n+1$ pour les points

$x_0, x_0 + \varepsilon, x_1, x_1 + \varepsilon, \ldots, x_n, x_n + \varepsilon$

deux à deux distincts $\forall \varepsilon$ assez petit, $\varepsilon < \min_{i \neq j} |x_i - x_j|$

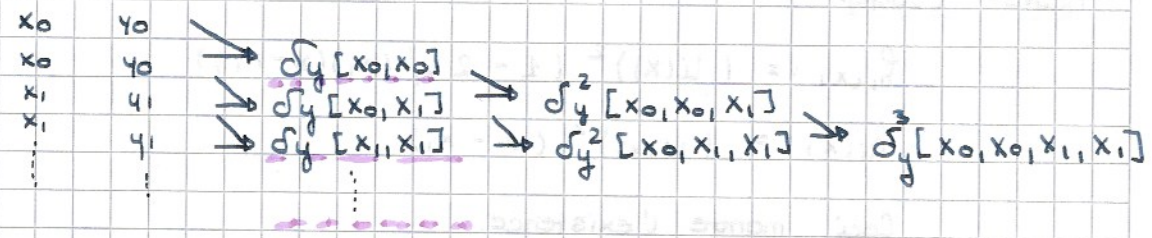
On note $q_\varepsilon(x)$ ce polynôme vérifiant :

$$q_\varepsilon(x_i) = y_i \quad , \quad q_\varepsilon(x_i + \varepsilon) = y_i + \varepsilon \cdot z_i$$

On commence par considérer la formule de Newton pour $\varepsilon \neq 0$

x_0	$\underbrace{y_0}_{= y}$		
$x_0 + \varepsilon$	$y_0 + \varepsilon \cdot z_0$	$\delta y[x_0, x_0 + \varepsilon]$	
x_1	y_1	$\delta y[x_1, x_0 + \varepsilon]$	$\delta^2 y[x_0, x_0 + \varepsilon, x_1]$
$x_1 + \varepsilon$	$y_1 + \varepsilon z_1$		
$\vdots$	$\vdots$	$\vdots$	
x_n	y_n		
$x_n + \varepsilon$	$y_n + \varepsilon z_n$		

afin d'obtenir $p(x)$, on prend la limite $\varepsilon \to 0$



Comme les x_i sont distincts 2 à 2 toutes les diff. divisées ainsi écrites ont un sens sauf celle soulignée

$$\delta_y[x_i, x_i+\varepsilon] = \frac{y_i + \varepsilon z_i - y_i}{x_i + \varepsilon - x_i} = z_i$$

On pose donc $\delta_y[x_i,x_i] = \lim_{\varepsilon \to 0} \delta_y[x_i, x_i+\varepsilon] = z_i$

On note $q_*(x) = q \lim_{\varepsilon \to 0} q_\varepsilon(x)$ le polynôme limite ainsi obtenu en faisant $\varepsilon \to 0$ dans la formule de Newton pour l'interpol. de Lagrange.

En utilisant la formule de Taylor,

$$q_\varepsilon'(x_i) = \frac{q_\varepsilon(x_i+\varepsilon) - q_\varepsilon(x_i)}{\varepsilon} - \frac{\varepsilon}{2} q_\varepsilon''(\xi_\varepsilon)$$

$$q_\varepsilon(x_i+\varepsilon) = q_\varepsilon(x_i) + \varepsilon q_\varepsilon'(x_i) + \frac{\varepsilon^2}{2} q_\varepsilon''(\xi_\varepsilon) \text{ avec } \xi_\varepsilon \in [x_i; x_i+\varepsilon]$$

vérifions que $q_*(x)$ vérifie les conditions de l'interp. de H.:

$$q_*(x_i) = \lim_{\varepsilon \to 0} q_\varepsilon(x_i) = y_i$$

$$q_*'(x_i) = \lim_{\varepsilon \to 0} q_\varepsilon'(x_i) = \lim_{\varepsilon \to 0} \frac{q_\varepsilon(x_i+\varepsilon) - q_\varepsilon(x_i)}{\varepsilon} - \frac{\varepsilon}{2} q_\varepsilon''(\xi_\varepsilon)$$

$$= z_i - \lim_{\varepsilon \to 0} \frac{\varepsilon}{2} q_*''(x_i) = z_i$$

car $\xi_\varepsilon \xrightarrow{\varepsilon \to 0} x_i$

$\deg(q_*) \leq 2n+1$ (Par construction)

On a donc généralisé la formule de Newton pour l'interpol. de Hermite. On déduit également :

Thm : Soient $x_0, \dots x_n$ 2 à 2 distincts dans $[a,b]$, soit $f:[a,b] \to \mathbb{R}$ $2n+2$ fois dérivable et soit $p(x)$ le poly. d'inter de Hermite de degré $\leq 2n+1$ vérifiant $p(x_i) = f(x_i)$ et $p'(x_i) = f'(x_i)$

Alors pour $x \in [a,b]$, $\exists\, \xi \in [a,b]$ dépendant de x tq

$$f(x) - p(x) = (x-x_0)^2 (x-x_1)^2 \dots (x-x_n)^2 \frac{f^{(2n+2)}(\xi)}{(2n+2)!}$$

Analyse Num: le 04.10.18

Preuve: On considère $q_\varepsilon(x)$ poly. d'inter. de Lagrange de deg $\leq 2n+1$

avec $q_\varepsilon(x_i) = f(x_i)$ et $q_\varepsilon(x_i+\varepsilon) = f(x_i+\varepsilon)$

On applique le thm sur l'erreur d'interpolation de Lagrange:

$$f(x) - q_\varepsilon(x) = (x-x_0)(x-x_0-\varepsilon)\dots(x-x_n-\varepsilon)\frac{f^{(2n+2)}(\xi_j)}{(2n+2)}$$

On fait $\varepsilon \to 0$:

$$f(x) - p(x) = (x-x_0)^2 \dots (x-x_n)^2 \frac{f^{(2n+2)}(\xi_j)}{(2n+2)}$$

avec $p(x) = \lim_{\varepsilon\to 0} q_\varepsilon(x)$

En effet, $\lim_{\varepsilon\to 0} \frac{q_\varepsilon(x_i+\varepsilon) - q_\varepsilon(x_i)}{\varepsilon} = \lim_{\varepsilon\to 0} \frac{f(x_i+\varepsilon) - f(x_i)}{\varepsilon} = f'(x_i)$

---

le 11.10.18

Ch. 2. Analyse d'erreurs d'arrondi

2.1 Ex: **Calcul de π:**

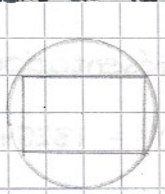
Un problème ancien, déjà étudié par les grecs :

la quadrature du cercle : (1) aire π [$\sqrt{\pi}$] aire π

En 1775, l'académie de Paris se fâche (les sciences)

1761, $\pi \notin \mathbb{Q}$ Jean-Henri Lambert

1837 Théorie de la règle et du compas Pierre-Laurent Wenñqel.





aire F_n

Approximation de π avec la méthode d'Archimède (260 av. JC)

aire du triangle avec $\alpha = \frac{2\pi}{n}$, $F_n = \cos\frac{\alpha_n}{2}\sin\frac{\alpha_n}{2} = \frac{1}{2}\sin\alpha_n$

l'aire du polygône est $nF_n = \frac{n}{2}\sin\alpha_n = \frac{n}{2}\sin\frac{2\pi}{n}$

On connait $\sin(\frac{2\pi}{n})$ pour certain n : $\sin(\frac{2\pi}{6}) = \sin(\frac{\pi}{3}) = \frac{\sqrt{3}}{2}$

En utilisant $\cos\alpha_n = 1 - \sin^2(\alpha_n/2)$ on peut calculer

$\sin(\alpha_n/2)$ à partir de $\sin(\alpha_n)$ p.13

si u / α_{2n})

$$\sin(\alpha_{2n}) = \sin(\alpha_n/2) = \sqrt{\frac{1-\cos\alpha_n}{2}}$$

$$\sin(\alpha_{2n}) = \sqrt{\frac{1-\sqrt{1-\sin\alpha_n^2}}{2}}$$

On peut donc calculer $A_5, A_{10}, A_{20} \ldots$

Observation: Deux algorithmes corrects pour calculer A_n, $(A_n \xrightarrow{n\to\infty} \pi)$ peuvent se comporter différement, ce qui est dû aux erreurs d'arrondi

2.2 Les nombres machines et la norme IEEE:

Chaque réel est exprimé en virgule flottante :

$$x = \pm m \times B^e$$ (similaire à la not. scientifique)

le signe ↗

m est la mantisse, $m = DD\,DDD$,

B est la base de représent ($B = 2$) $\in \mathbb{N}^*$

e est l'exposant $e = \pm DDD\ldots \in \mathbb{Z}$

Rem:
- Si on enlève le terme B^e, on obtient une représent. en virgule fixe $x = \pm m$ (repres. dosdite.
- Pour la mantisse en général, la virgule est fixée souvent juste après le premier chiffre : $m = D.DD$
- Si la virgule est flottante, la représentation n'est pas unique:

$$13.254 = 13.254 \cdot 10^0 = 1.3254 \cdot 10^1 = 13254 \cdot 10^{-3}$$

Pour rendre cette représentation unique, il faut la normaliser

Un choix souvent pour $x \neq 0$: $m = D.DD\ldots$

Ex: $\pi^4 = 97,409 = 9.7 \cdot 10^1$

(97,409 : dénormalisé ; $9.7 \cdot 10^1$: normalisé)

le 11.10.18

Analyse Num:

On note $\mathbb{M}$ l'ensemble des réels représentés en machine (virg. flottante) $\mathbb{M}$ est fini.

Ex: On considère une calculatrice, avec une base $10 = B$, avec 5 chiffres diff. dans la machine, 2 chiffres dans l'exposant

$x = \pm D.D\ldots \times 10^{\pm DD}$ $\quad D \in \{0,1,\ldots 9\}$

le + grd nbr machine : $+\ 9.9999 \cdot 10^{99}$

le + petit : $-9.9\ldots\ 10^{99}$

le + petit normal : $1.0\ldots\ 10^{-99}$

le + petit non-normal : $0,0\ldots 01 \cdot 10^{-99}$

Combien de nbr machine tot ?

Normalisé : $2 \times 9 \times 10^4 \times 199$ (signe × 1er chiffres × autres chiffres)

Non-Normalisé : $2 \times 10^4 - 1 = 19'999$.

Déf: Si $x \in \mathbb{R} \setminus \mathbb{M}$ (x réel non représentable en machine), il faut l'arrondir au plus près avec $\tilde{x} \in \mathbb{M}$. Ceci définit la fonction d'arrondi en nombre machine : $f_L : \begin{array}{ccc} \mathbb{R} & \to & \mathbb{M} \\ x & \mapsto & \tilde{x} = f_L(x) \end{array}$

On note ε_{mach} l'écart entre 1 et le + petit nbr machine > 1.

ε_{mach} s'appelle la précision de la machine, ou l'epsilon de la machine. Pour les réels (de magnitude raisonable) on a : $\frac{|x - f_L(x)|}{|x|} \leq \varepsilon_{mach}$

Précisément: Pour $x \in \mathbb{R}$ avec $\delta_u < |x| < \delta_o$ (underflow, overflow), il existe ε_x avec

$|\varepsilon_x| \leq \varepsilon_{mach}$ tq $f_l(x) = x(1+\varepsilon_x)$

$\Rightarrow \quad \frac{|f_l(x) - x|}{|x|} = \varepsilon_x \leq \varepsilon_{mach}$.

Rem : $fl(1 + \varepsilon_{mach}) = 1 + \varepsilon_{mach}$ — Peut-être 1 exemple

Déf : La norme IEEE 754 (exemple en simple précision)

Base : 2 sur 32 bits (binary digits)

S	E .... E	M ..... M
↑	1 8	9 31
signe	↓ exposant	mantisse
1 bt	8 bt	23 bt

Nombre normalisé : Si $1 \leq e \leq 254 = 2^8 - 2$, on obtient

$$x = (-1)^s \times 1.m \times 2^{e-127}$$

Rem :
- le 1er bits 1 de la mantisse n'est pas stocké en mémoire.
- le + petit nbr machine > 1 est : $1.\underbrace{00\ldots01}_{22 \cdot 0}$ et $\varepsilon_{mach} = 2^{-23}$ ou $1{,}2 \cdot 10^{-7}$
- Il n'y a pas de signe mais un biais dans l'exposant
- le + grd nbr machine : $1.1\ldots1 \times 2^{254-127} \hat{=} 3.4028 \cdot 10^{38}$

Nombre dénormalisé : On pose $x = (-1)^s \times 0.m \times 2^{-126}$ $(e = 0)$

ce sont deo nbr + proche de 0

Exceptions : On peut représenter $+\infty, -\infty$, NaN, $+0, -0$.

Avec $e = 255$ et $m \neq 0$, on a un nbr non-valide et on note $\hat{x} = NaN$

$e = 255$, $s = 0$, $m = 0$, on a $x = +\infty$ (Inf)

$e = 255$, $s = 1$, $m = 0$, on a $x = -\infty$

$e = 0$, $s = \begin{cases} 0 \\ 1 \end{cases}$, $m = 0$ on a $x = \begin{cases} 0 \\ -0 \end{cases}$

Dans Matlab les nbr sont stockés en double précision.

Déf : Norme IEEE double précision

S	E . . . E	M .... M
1 bt	11 bits	52 bits

le 18.10.18

Analyse Num :

2.3 Arithmétique flottante :

Ex : Calculatrice avec 5 chiffres significatifs en base 10

Soit $\tilde{a} = 1,2345 \cdot 10^2 \in \mathbb{M}$ et $\tilde{b} = 6,7890 \cdot 10^4 \in \mathbb{M}$

$c = \tilde{a} \cdot \tilde{b} = \underbrace{8,38102050}_{\notin \mathbb{M}} \cdot 10^6$ la calculatrice arrondi

$f_L(c) = \tilde{c} = 8.3810 \cdot 10^6 = c(1+\varepsilon)$

$|\varepsilon| \leq \varepsilon_{mach}$ $\varepsilon \approx 2.05 \cdot 10^{-5}$ $\varepsilon_{mach} = 10^{-4}$

Déf : Axiome fondamental d'arithmétique flottante :

Soit $\circledast \in \{+, -, \cdot, \div\}$, pour tout $\tilde{a}, \tilde{b} \in \mathbb{M}$, $\exists \varepsilon$ avec $|\varepsilon| \leq \varepsilon_{mach}$ tq $f_L(\tilde{a} \circledast \tilde{b}) = (\tilde{a} \circledast \tilde{b})(1+\varepsilon)$

Ex : Calculer $a+b$ avec $a, b \in \mathbb{R}$, $f_L(f_L(a) + f_L(b)) = f_L(a(1+\varepsilon_A) + b(1+\varepsilon_B))$ [$\exists \varepsilon_A, \varepsilon_B$ avec $|\varepsilon_A| < \varepsilon_{mach}$ et $|\varepsilon_B| < \varepsilon_{mach}$]

$= (a(1+\varepsilon_A) + b(1+\varepsilon_B))(1+\varepsilon_c) = a + b + a\varepsilon_a + b\varepsilon_B + a\varepsilon_c + b\varepsilon_c + a\varepsilon_a\varepsilon_c + b\varepsilon_b\varepsilon_c \neq (a+b)(1+\varepsilon_d)$

2.4 Annulation de chiffre :

Ex : Calculatrice base 10 à 3 chiffres significatifs et 2 dans l'exposant.

Calculer $a^2 - b^2$ $a = 4,25$ $b = 4,23$

	Arithm. correcte	Arithm flottante	
a	4.25	4.25	$f_L(a) = a$
b	4.23	4.23	$f_L(b) = b$
a^2	18,0625	$1,81 \cdot 10^1$	$f_L(a^2) \neq a^2$
b^2	17,8929	$1,79 \cdot 10^1$	$f_L(b^2) \neq b^2$
a^2-b^2	0,1696	$2,00 \cdot 10^{-1}$	$f_L(a^2-b^2) = (a^2-b^2)(1+\varepsilon)$ où $\varepsilon = 0,18$

L'erreur relative $\varepsilon \approx 18\%$ est beaucoup + grande que la précision machine $\varepsilon_{mach} = 10^{-2} = 1\%$

Lorsqu'on soustrait 2 nombres proches, le résultat subit une perte de précision.

Ex : (Suite)

Un autre algorithme $a^2 - b^2 = (a-b)(a+b)$

	Arithm exacte	Arithm flottante	
$a-b$	0.02	$2.00 \cdot 10^{-2}$	$fl(a-b) = a-b$
$a+b$	8.48	$8.48 \cdot 10^{0}$	$fl(a+b) = a+b$
$(a-b)(a+b)$	0,1696	$1,70 \cdot 10^{-1}$	$fl(a^2-b^2) = 0,1696(1+\varepsilon)$

$\varepsilon = 0.2\%$

Ici on a une bonne précision $\varepsilon \simeq 0.2\% < \varepsilon_{mach} = 1\%$.

Conclusion : Éviter de soustraire des nbr de même taille

Ex : Retour sur la formule de calcul de π.

$$\sin\frac{\alpha_n}{2} = \sqrt{\frac{1-\sqrt{1-\sin^2(\alpha_n)}}{2}} \qquad \text{pour } n \to \infty,\ \sqrt{1-(\sin\alpha_n)^2} \longrightarrow 1$$

On calcul au numérateur une soustraction de nombres de même taille: $1 - \sqrt{1-\sin^2(\alpha_n)}$

Pour éviter cette soustraction, on peut utiliser la formule :

$$\left(\sin\frac{\alpha_n}{2}\right)^2 = \frac{1-\sqrt{1-\sin^2(\alpha_n)}}{2} \cdot \frac{1+\sqrt{1-\sin^2\alpha_n}}{1+\sqrt{1-\sin^2\alpha_n}}$$

$$= \frac{\sin^2\alpha_n}{2\left(1+\sqrt{1-(\sin\alpha_n)^2}\right)}$$

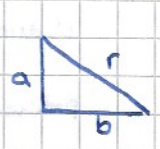
↑ on ajoute 2 nbr de m̂ signe.

2.6 Overflow et Underflow:

Déf : Overflow : on fait intervenir des nbr trop grands en val. abs

Underflow : " ——— " ——— " petit " ——— "

Ex : Calculer : $r = \sqrt{a^2+b^2}$ $a = 3 \cdot 10^{52}$ $b = 4 \cdot 10^{52}$



	exacte	flottante
a^2	$9 \cdot 10^{104}$	$9,99 \cdot 10^{99}$
b^2	$1.6 \cdot 10^{105}$	$9,99 \cdot 10^{99}$
a^2+b^2	$2.5 \cdot 10^{105}$	$9,99 \cdot 10^{99}$
$\sqrt{a^2+b^2}$	$5 \cdot 10^{52}$	$9.99 \cdot 10^{45}$

le 18.10.18

Analyse Num:

Ex: (Suite)

Calculer pwsfôt $r = |b|\sqrt{\left(\frac{a}{b}\right)^2 + 1}$ avec $|b| = \max\{|a|, |b|\}$

	Arithm exacte		flottante	
a/b	0.75		$7.5 \cdot 10^{-1}$	
$(a/b)^2$	0,5625		$5,63 \cdot 10^{-1}$	
$(a/b)^2 + 1$	1,5625		$1,56 \cdot 10^{0}$	
$\sqrt{\ }$	1.25	=	$1.25 \cdot 10^{0}$	l'égalité est une chance
$\lvert b\rvert \cdot \sqrt{\ }$	$5 \cdot 10^{52}$		$5 \cdot 10^{52}$	$\Rightarrow \varepsilon = 0$

Chapitre 3 Condition et stabilité

Rappel: Des familles de normes p.

Pour $\underline{x} \in \mathbb{R}^n$, $\|\underline{x}\|_p = (|x_1|^p + \dots + |x_n|^p)^{1/p}$ $1 \leq p < +\infty$

$\|\underline{x}\|_2$ = norme euclidienne

$\|\underline{x}\|_\infty = \max\{|x_1|, \dots, |x_n|\} = \max_{1 \leq j \leq n} |x_j|$

But: étant donné un problème $x \longmapsto f(x)$, étudier l'influence (des ~~erreurs d'arrondis~~) des perturbations de x dans le résultat $f(x)$

Déf: Soit $\underline{x} \in \mathbb{R}^n$ et $f: \mathbb{R}^n \to \mathbb{R}^m$ avec $\underline{x} \neq 0$ et $f(\underline{x}) \neq 0$.

La condition $K = K(f(x))$ du problème f en $\underline{x}$ est le + petit nbr tq: $\forall \varepsilon > 0$, $\forall \underline{y} \neq \underline{x}$, $\frac{\|y - x\|_p}{\|x\|_p} \leq \varepsilon \Rightarrow \frac{\|f(y) - f(x)\|_p}{\|f(x)\|_p} \leq K \cdot \varepsilon + o(\varepsilon)$

On dit que le problème est bien conditioné si K n'est "pas trop grand" sinon on dit qu'il est mal conditionné.

Rappel: $o(\varepsilon)$ signifie une quantité de la forme $\alpha(\varepsilon) \cdot \varepsilon$ avec $\alpha(\varepsilon) \xrightarrow{\varepsilon \to 0} 0$ $(\varepsilon^2, 100\,\varepsilon^{3/2}, \varepsilon^3, 10^4 \cdot \varepsilon^{12})$

$O(\varepsilon)$ signifie une quantité de la forme $\Gamma(\varepsilon) \cdot \varepsilon$ avec $\Gamma(\varepsilon)$ bornée.

Rem: ⚠ K dépend de f, x et du choix de normes

Retour sur le comportement avec l'arithm. en virgule flottante:

Déf: Soit $\underline{x} \in \mathbb{R}^n$ où $\underline{x} = (x_1 \dots x_n)$

Pour un algorithme, on arrondi $\underline{x}$ en vecteurs de nombres machines $\tilde{\underline{x}} = (\tilde{x}_1, \dots, \tilde{x}_n)$. On a $\|\underline{x} - \tilde{\underline{x}}\|_1 = \sum_{i=1}^{n} |x_i - \tilde{x}_i| = \sum_{i=1}^{n} |x_i - f_L(x_i)|$

$= \sum_{i=1}^{n} |x_i||\varepsilon_i| \leq \varepsilon_{mach} \cdot \sum_{i=1}^{n} |x_i| = \varepsilon_{mach} \cdot \|\underline{x}\|_1$

Déf: Ensuite, pour le résultat du problème f:

$$f(\tilde{x}) = f(x) + \|f(x)\|_1 \cdot \delta \quad \text{avec} \quad \delta = \frac{f(\tilde{x}) - f(x)}{\|f(x)\|_1} \in \mathbb{R}$$

où δ est vecteur des erreurs relatives

tq $\|\delta\|_1 \simeq K\,\varepsilon_{mach} + o(\varepsilon_{mach})$ négligable en général

Si $K \simeq 1$ bien conditionné avec un résultat à la précision de la machine.

Si K est grand: on perd en précision, pbl mal conditionné.

Ex: la condition pour +

Soit $f(x_1, x_2) = x_1 + x_2$ où $x_1, x_2 \in \mathbb{R}$

Soit $y = (y_1, y_2)$ avec $|x_i - y_i| \leq \varepsilon \cdot \|x\|_\infty \ \forall i$ (ie $\|x - y\|_\infty \leq \varepsilon \|x\|_\infty$)

On a $\left| \frac{(\overbrace{y_1 + y_2}^{\text{perturbé}}) - (\overbrace{x_1 + x_2}^{\text{exacte}})}{(x_1 + x_2)} \right| = \left| \frac{y_1 - x_1 + y_2 - x_2}{(x_1 + x_2)} \right| \leq \frac{2\varepsilon \|x\|_\infty}{|x_1 + x_2|}$

Par le choix de norme $\|.\|_\infty$ dans $\mathbb{R}^2$ et $|.|$ pour $\mathbb{R}$ ($f: \mathbb{R}^2 \to \mathbb{R}$)

on obtient $K_x = \frac{2\|\underline{x}\|_\infty}{|x_1 + x_2|}$

Si le signe de x_1 = celui de x_2 alors $K = 2$ bien conditionner

Si x_1 et x_2 sont de signes opposés et proches en val. absolues, alors $|x_1 - x_2| \simeq 0$ et $K \gg 1$ (mauvaise condition).

le 25.10.18

Analyse Numérique:

Rappel: Condition du problème f en x $\kappa(f(x))$ le + petit nbr tq

$$\forall \varepsilon > 0,\ \forall y \neq x \quad \frac{\|x-y\|}{\|x\|} \leq \varepsilon \Rightarrow \frac{\|f(x)-f(y)\|}{\|f(x)\|} \leq \kappa \cdot \varepsilon + O(\varepsilon)$$

3.2 Conditions de problèmes différentiables :

Rappel: Diffentiabilité d'une fonction

On dit que $f: \mathbb{R}^n \to \mathbb{R}^m$ est différentiable en x si $\exists$ une application linéaire $f'(x): \mathbb{R}^n \to \mathbb{R}^m$ (la différentielle)

tq : $f(y) = f(x) + f'(x)(y-x) + o(\|y-x\|)$ pour $\|y-x\| \to 0$

ou $f(x+h) = f(x) + f'(x)\cdot h + o(\|h\|)$ pour $\|h\| \to 0$

$f'(x): \mathbb{R}^n \to \mathbb{R}^m$ app. linéaire représentée par la matrice jacobienne souvent notée comme $f'(x)$

$$f'(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & & \frac{\partial f_m}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{m\times n}$$

m lignes et n colonnes.

$\uparrow \partial/\partial x_1$

avec $f = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix}$ et $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$

⚠ ne pas confondre $\nabla f(x) = f'(x)^T$ le gradient

Rappel: $O(\|h\|)$ signifie une quantité de la forme $\varepsilon(h)\cdot\|h\|$

avec $\varepsilon(h) \xrightarrow{h\to 0} 0$

Déf: Norme d'opérateur.

pour une matrice $A = \begin{pmatrix} a_{11} & & a_{1n} \\ & & \\ a_{m1} & & a_{mn} \end{pmatrix} \in \mathbb{R}^{m\times n}$

On définit pour $A \in \mathbb{R}^{m\times n}$ $\|A\|_{p\to q} = \max_{\|x\|_p = 1} \|Ax\|_q = \max_{x\neq 0} \frac{\|Ax\|_q}{\|x\|_p}$

↑ norme p vers norme q

Rem : • $\|A\|_{p\to q}$ est le + petit réel tel que : $\forall \underline{x} \in \mathbb{R}^n$, $\|A\underline{x}\|_q \leq \|A\|_{p\to q}\|\underline{x}\|_p$

En effet : $\forall x \neq 0$ on a $\dfrac{\|A\underline{x}\|_q}{\|\underline{x}\|_p} \leq \|A\|_{p\to q}$

• $\|A\|_{p\to q}$ dépend du choix des normes :

Si $p = q$ on note $\|A\|_p = \|A\|_{p\to p}$

Thm : Propriété des normes d'opérateurs

Soit $A \in \mathbb{R}^{m\times n}$, alors :

1) $\|A\| \geq 0$ 2) $\|A\| = 0 \iff A = 0$

3) $\|A+B\| \leq \|A\| + \|B\|$

4) $\|\alpha A\| = |\alpha|\,\|A\|$ $\forall \alpha \in \mathbb{R}$

5) $\|A\underline{x}\| \leq \|A\|\cdot\|\underline{x}\|$ $\forall x \in \mathbb{R}^m$ (prop d'une norme d'opération)

6) $\|AB\| \leq \|A\|\cdot\|B\|$ (Sous multiplicité) $\forall B \in \mathbb{R}^{m\times p}$ et $\|B\|$ norme d'opérateur.

Preuve 6) : $\forall \underline{x} \in \mathbb{R}^n$ $\|AB\underline{x}\| \leq \|A\|\|B\underline{x}\| \leq \|A\|\|B\|\|\underline{x}\|$

comme $\|AB\|$ est la meilleur cst tq $\|AB\underline{x}\| \leq \|AB\|\,\|\underline{x}\|$

on obtient $\|AB\| \leq \|A\|\cdot\|B\|$

⚠ Il existe des normes sur l'esp. vect de $\mathbb{R}^{m\times n}$ qui ne vérifient pas 5) et 6).

Il y a deo cas particullier important où il existe une forme explicite pour calculer $\|A\|$.

- $\|A\|_2 = \|A\|_{2\to 2} = \sqrt{\text{+ grande val. propre de } A^T.A}$
- $\|A\|_1 = \|A\|_{1\to 1} = \max\limits_{j=1\dots n} \left(\sum\limits_{i=1\dots m} |a_{ij}|\right)$
- $\|A\|_\infty = \|A\|_{\infty\to\infty} = \max\limits_{i=1\dots m} \left(\sum\limits_{j=1\dots n} |a_{ij}|\right)$

Rem : • $\|A\|_1 = \|A^T\|_\infty$

• $\|A\|_2 = \|A^T\|_2$

Preuve : Pour la formulle de $\|A\|_\infty$

Pour $\underline{x} \in \mathbb{R}^n$, avec $\|\underline{x}\|_\infty = 1$, on calcule

$$|(Ax)_i| = \left|\sum_{j=1}^{m} a_{ij}x_j\right| \leq \sum_{j=1}^{m} |a_{ij}|\,|x_j| \leq \sum_{j=1}^{m} |a_{ij}|$$

Ainsi $\|Ax\|_\infty = \max_{i=1\dots n} |(Ax)_i| \leq \max_{i=1\dots n} \left(\sum_{j=1}^{m} |a_{ij}|\right)$

Ce qui mq $\|A\| \leq \max_{i=1\dots n} \left(\sum_{j=1}^{m} |a_{ij}|\right)$

Il reste à montrer l'égalité.

Soit $i = I$ la ligne où $\sum_{j=1}^{m} |a_{ij}|$ est maximale

alors $\max_{i=1\dots n} \sum_{j=1}^{m} |a_{ij}| = \sum_{j=1}^{m} |a_{Ij}|$

On pose $v = \begin{pmatrix} \pm 1 \\ \vdots \\ \pm 1 \end{pmatrix}$ avec $v_j = \text{signe}(a_{Ij})$

Alors $\|\underline{v}\|_\infty = 1$ et $(A\underline{v})_I = \sum_{j=1}^{m} a_{Ij} v_j = \sum_{j=1}^{n} |a_{Ij}|$

et pour $i \neq j$ $|(Av)_i| = |\sum_{j=1}^{m} a_{ij}v_j| \leq \sum_{j=1}^{m} |a_{ij}|$

et donc $\|A\|_\infty = \max_{i=1,\dots n} \sum_{j=1}^{m} |a_{ij}| = \|A\underline{v}\|_\infty$

Thm : Soit $f : \mathbb{R}^n \to \mathbb{R}^m$ différentiable, alors la condition K de f en $\underline{x} \in \mathbb{R}^n$ vérifie pour $\underline{x} \neq 0$ et $f(\underline{x}) \neq 0$:

$$K = \lim_{\varepsilon \to 0} \sup \left\{ \frac{\|f(y) - f(x)\|}{\|f(x)\|} ; \frac{\|y - x\|}{\|x\|} \leq \varepsilon \right\}$$

$$= \frac{\|f'(x)\|}{\|f(x)\| / \|x\|} = \frac{\|f'(x)\| \cdot \|x\|}{\|f(x)\|}$$

Preuve: • Pour la 1ère égalité, avec la déf de K on a

$$\lim_{\varepsilon \to 0} \sup \left\{ \frac{\| f(y) - f(x) \|}{\| f(x) \|} ; \frac{\| y - x \|}{\| x \|} \leq \varepsilon \right\} = \lim_{\varepsilon \to 0} \frac{K\varepsilon + o(\varepsilon)}{\varepsilon}$$

$$= \lim_{\varepsilon \to 0} K + o(1) = K.$$

• Pour la 2ème (par déf de diff)

$$f(y) - f(x) = f'(x)(y - x) + o(\| x - y \|) \quad y \to x$$

$$\| f(y) - f(x) \| = \| f'(x)(x - y) \| + o(\| x - y \|) \quad y \to x$$

$$\sup_{0 < \| y - x \| \leq \varepsilon \| x \|} \| f'(x)(x - y) \| = \sup_{0 < \| y - x \| < \varepsilon \| x \|} \frac{\| f'(x)(x - y) \|}{\| x - y \|} \cdot \| x - y \| = \| f'(x) \| \cdot \| x - y \|$$

Pour $\varepsilon \to 0$, on a $\| y - x \| \leq \| x \| \cdot \varepsilon \longrightarrow 0$

et $\lim_{\varepsilon \to 0} \frac{o(\| y - x \|)}{\varepsilon} \leq \lim_{\| y - x \| \to 0} \| x \| \cdot \frac{o(\| y - x \|)}{\| y - x \|} \xrightarrow[\varepsilon \to 0]{} 0$

On a ainsi $K = \lim_{\varepsilon \to 0} \sup \left\{ \frac{\| f'(x)(y - x) \| + o(\| y - x \|)}{\varepsilon \cdot \| f(x) \|} ; \frac{\| y - x \|}{\| x \|} \leq \varepsilon \right\}$

$$= \lim_{\varepsilon \to 0} \frac{\| f'(x) \| \cdot \| x \| \cdot \varepsilon}{\varepsilon \| f(x) \|} = \frac{\| f'(x) \| \cdot \| x \|}{\| f(x) \|}$$

Ex: Addition: $f(x_1, x_2) = x_1 + x_2 \quad f'(x_1, x_2) = (1 \quad 1) \in \mathbb{R}^{1 \times 2}$

Pour la norme 1 $\| f'(x_1, x_2) \| = 1$ et

$$K = \frac{\| f'(x_1, x_2) \|}{\| f(x_1, x_2) \| / \| (x_1, x_2) \|} = \frac{|x_1| + |x_2|}{|x_1 + x_2|}$$

$K = 2$ si signe x_1 = signe x_2 et $K \gg 1$ si $x_1 \simeq -x_2$

Ex: Calcul de π :

On a considéré $f(x) = 1 - \sqrt{1 - (\sin x)^2}$

On a vu que l'évaluation de $f(x)$ pour $x \simeq 0$ pose problème avec virgule flottante avec un algorithme naïf.

Calculons le conditionement de $f(x) = 1 - \cos(x)$

$$K(f(x)) = \frac{|f'(x)||x|}{|f(x)|} = \frac{|\sin x||x|}{|1 - \cos(x)|} = \frac{x \sin x}{1 - \cos x} \qquad x \geq 0 \quad x > 0$$

A la limite : $\lim_{x \to 0^+} K(f(x)) = \lim_{x \to 0^+} \frac{(x + o(x^3))x}{1 - (1 - \frac{x^2}{2} + o(x^4))} = \ldots = 2$

↑ le problème est bien conditioné pour x petit

On peut borner le conditionement d'une compasition de problèmes par le produit des conditionnements.

Cor: Soit $f_1, \dots f_n$ differentiables, alors $f = f_n \circ \dots \circ f_1$ vérifie $\kappa(f) \leq \prod_{k=1}^{n} \kappa(f_k)$. $= \kappa(f_n) \cdot \dots \cdot \kappa(f_1)$.

Preuve: $n = 2$

$f = f_2 \circ f_1 \qquad f'(x) = f_2'(f_1(x)) \cdot f_1'(x)$

$\| f'(x) \| \leq \| f_2'(f_1(x)) \| \cdot \| f_1'(x) \|$

$\kappa(f) \leq \dfrac{\| f_2'(f_1(x)) \| \cdot \| f_1'(x) \|}{\| f_2(f_1(x)) \|} \cdot \dfrac{\| f_1'(x) \|}{\| f_1(x) \|} = \kappa(f_1) \cdot \kappa(f_2)$

⚠ Ne pas confondre le conditionement d'un problème avec la stabilité d'un algorithme utilisé pour le résoudre, un même problème f peut posséder un algorithme stable et un autre instable.

---

le 01.11.18

3.3 Stabilité des algorithmes:

On considère un problème f avec une condition $\kappa(f)$. On considère un algorithme représenté par $\tilde{f} = \tilde{f}_n \circ \dots \circ \tilde{f}_1$ correspond à une succésion d'opération pour calculer f

Idéalement, on amerait un bon algo avec la précision de la machine

$\dfrac{\| \tilde{f}(x) - f(x) \|}{\| f(x) \|} \leq C \cdot \varepsilon_{mach} \qquad \forall x$ tq $f(x) \neq 0$ avec C cst petite

Malheureusement, une telle propriété est trop exigeante, car certain problèmes (-) sont mal conditionnés.

Il faut enfait aussi tenir compte de la sensibilité par rapport aux données mesurées par le $\kappa(f, x)$

Déf: Un algo $\tilde{f}$ pour évaluer un problème f est dit **forwardstable** (Stable au sens direct) si $\forall x$ (tq $f(x) \neq 0$) $\dfrac{\| \tilde{f}(x) - f(x) \|}{\| f(x) \|} \leq C \cdot K(f,x) \cdot \varepsilon_{mach}$ + $\mathcal{O}(\varepsilon_{mach})$ avec C indépendant de x pas trop grand.

⚠ Ne pas confondre la **condition** $K(f,x)$ avec **la stabilité** d'un **algorithme**.

Il existe plusieurs algorithmes pour 1 problème, certains forwardstables et d'autre instables

Ex: Considérons l'évaluation de $f(x) = \dfrac{1}{x(1+x)}$ $x > 0$

On étudie la stabilité de 2 algo différents.

1) $x \xrightarrow{f_1} x \;\; x \xrightarrow{f_2} x+1 \;\; (x, x+1) \xrightarrow{f_3} x(x+1) \xrightarrow{f_4} \dfrac{1}{x(x+1)}$

$f_1(x) = x$ $\quad K(f_1) = 1$

$f_2(x) = x+1$ $\quad K(f_2) = 1$

$f_3(x,y) = xy$ $\quad K(f_3) = 2$

$f_4(x) = 1/x$ $\quad K(f_4) = 1$

$\max(K(f_1)K(f_2)) \cdot K(f_3) \cdot K(f_4) = 2 \leq C \cdot K(f)$ avec $K(f) = 2$

$$K(f) = \left| \frac{x f'(x)}{f(x)} \right| = x \cdot x(x+1) \left[-\frac{1}{x^2} + \frac{1}{(x+1)^2} \right] = \frac{2x+1}{x+1} < 2$$

$K(f) \xrightarrow{x \to \infty} 2$

$C \geq 1 \Rightarrow$ l'algo est forwardstable

2) Un autre algorithme pour $f(x) = \dfrac{1}{x(x+1)} = \dfrac{1}{x} - \dfrac{1}{x+1}$

$x \xrightarrow{g_1} \frac{1}{x}$ $\quad x \xrightarrow{g_2} x+1 \xrightarrow{g_3} \frac{1}{x+1}$ $\quad \frac{1}{x}, \frac{1}{x+1} \xrightarrow{g_4} \frac{1}{x} - \frac{1}{x+1} = f(x)$

$K(g_1) = K(g_2) = K(g_3) = 1$ mais $K(g_4)$ est différent

$g_4(z,y) = z - y$ $\quad K(g_4) = \dfrac{|y| + |z|}{|y - z|} = \dfrac{\frac{1}{x} + \frac{1}{x+1}}{\frac{1}{x} - \frac{1}{x+1}} = 2x+1$

$\max(K(g_1), K(g_2), K(g_3)) \cdot K(g_4) = 2x+1 \leq C \cdot K(f) = C \cdot 2$

C est grand pour x grand $\Rightarrow$ algo non forwardstable

Ex : Retour sur les algorithmes pour évaluer π

Calculons pour $0 < x \ll 1$ ($x \simeq 0$) la stabilité de

$f(x) = 1 - \sqrt{1 - \sin^2 x}$ évalué à l'aide de l'algo. suivant :

$$x \xrightarrow[f_1]{} \sin x \xrightarrow[f_2]{} \sin^2 x \xrightarrow[f_3]{} \sqrt{1-\sin^2 x} \xrightarrow[f_4]{} 1 - \sqrt{1-\sin^2 x}$$

$f_1(x) = \sin x \quad f_2(x) = x^2 \quad f_3(x) = \sqrt{1-x} \quad f_4(x) = 1 - x$

Avec $\sin x \simeq x$ et $\sqrt{1-x} \simeq 1 - \frac{x}{2}$ et

$(1 + m\varepsilon_1)(1 + n\varepsilon_2) = (1 + (m+n)\varepsilon_3)$ avec $|\varepsilon_i| \leq \varepsilon_{mach}$

On obtient :

$$x \xrightarrow{\tilde{f}_L} x(1+\varepsilon_1)$$
$$\xrightarrow{\tilde{f}_1} \sin(x(1+\varepsilon_1))(1+\varepsilon_2) \simeq x(1+\varepsilon_1)(1+\varepsilon_2) = x(1+2\varepsilon_3)$$
$$\xrightarrow{\tilde{f}_2} x^2(1+\Sigma\varepsilon_3)^2(1+\varepsilon_4) \simeq x^2(1+5\varepsilon_5)$$
$$\xrightarrow{\tilde{f}_3} \sqrt{1-x^2(1+5\varepsilon_5)}\,(1+\varepsilon_6) \simeq 1 + \varepsilon_6 - \tfrac{1}{2}x^2(1+6\varepsilon_7)$$
$$\xrightarrow{\tilde{f}_4} \left[1 - (1 + \varepsilon_6 - \tfrac{1}{2}x^2(1+6\varepsilon_7))\right](1+\varepsilon_8) \simeq -\varepsilon_6 + \tfrac{1}{2}x^2(1+7\varepsilon_9)$$

Donc l'erreur relative de $\tilde{f} = \tilde{f}_4 \circ \tilde{f}_3 \circ \tilde{f}_2 \circ \tilde{f}_1 \circ \tilde{f}_L$ est bornée

comme $$\left|\frac{\tilde{f}(x) - f(x)}{f(x)}\right| \simeq \frac{\left|1 - \varepsilon_6 + \frac{1}{2}x^2(1+7\varepsilon_9) - \frac{x^2}{2}\right|}{\frac{x^2}{2}} < \frac{\varepsilon_{mach}}{\frac{x^2}{2}} + 7\varepsilon_{mach}$$

On remarque que $\frac{1}{\frac{x^2}{2}} \xrightarrow[x \to 0]{} \infty$ et pourtant $\kappa(f) = 2$ si x petit

Ainsi cet algo n'est pas forwardstable.

En résumé, pour voir si $\tilde{f} = \tilde{f}_N \circ \dots \circ \tilde{f}_1 \circ \tilde{f}_L$ pour évaluer f

est forwardstable on regarde si :

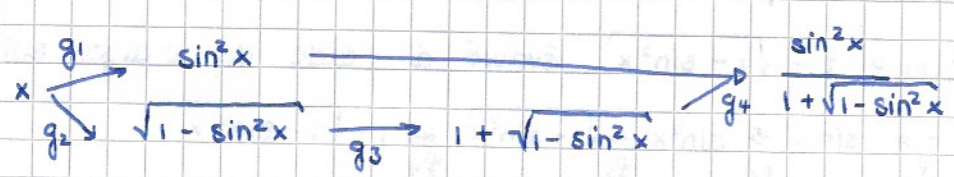
$$\kappa(\tilde{f}_N) \cdots \kappa(\tilde{f}_1) \leq C \cdot \kappa(f)$$

avec C pas trop grand.

Ex: (Suite)

Maintenant on va évaluer la stabilité de $f(x) = \dfrac{\sin^2 x}{1+\sqrt{1-\sin^2 x}} = 1 - \sqrt{1-\sin^2 x}$

pour $0 < x << 1$, maintenant évalué comme :



où $g_1 = f_2 \circ f_1$ $g_2 = f_3 \circ f_2 \circ f_1$ et $g_4(y,z) = \dfrac{y}{z}$

On obtient :

$$x \xrightarrow{\tilde{g}_1} x^2(1+5\varepsilon_1) = y$$

$$x \xrightarrow{\tilde{g}_2} 1 + \varepsilon_2 - \tfrac{1}{2}x^2(1+6\varepsilon_3) \xrightarrow{\tilde{g}_3} 2 + 3\varepsilon_4 - \tfrac{1}{2}x^2(1+7\varepsilon_5) = z$$

$$(y,z) \xrightarrow{\tilde{g}_4} \frac{x^2(1+5\varepsilon_1)}{2+3\varepsilon_4 - \frac{1}{2}x^2(1+7\varepsilon_5)} \cdot (1+\varepsilon_6) = \ldots =$$

$$= \tfrac{1}{2}x^2\left(1 + \underbrace{7.5\,\varepsilon_8}_{C}\right)$$

Donc l'erreur est bornée par 7.5 ε_{mach} pour $x \simeq 0$

Cet algo est donc forward stable pour évaluer f avec $K(f) \simeq 2$

En pratique, la forward stability est fastidieuse à étudier, une autre notion est celle de backward stability plus pratique

Déf: backward stability de Wilkinson

Un algo $\tilde{f}$ pour évaluer f est dit backward stable si $\forall x \neq 0$

$\exists \hat{x}$ tq $\tilde{f}(x) = f(\hat{x})$ et $\dfrac{\|\hat{x} - x\|}{\|x\|} \leq C \cdot \varepsilon_{mach} + O(\varepsilon_{mach})$

avec C indép de x et x pas trop grand.

Autrement dit, le résultat inexacte $\tilde{f}(x)$ donné par l'algo est en fait exacte pour des données perturbées $\tilde{x}$ $\tilde{f}(x) = f(\tilde{x}) = f(x+\delta)$ avec $\delta \leq C \cdot \varepsilon_{mach} \|x\|$

Analyse Numérique: le 01.11.18

Ex : Produit scalaire

$f(\underline{x}, \underline{y}) = x_1 y_1 + x_2 y_2 \qquad \underline{x} = (x_1, x_2) \quad \underline{y} = (y_1, y_2)$

$(x_1, x_2)(y_1, y_2) \longrightarrow \begin{matrix} x_1 y_1 \\ x_2 y_2 \end{matrix} \longrightarrow x_1 y_1 + x_2 y_2$

Résultat numérique :

$$\tilde{f}(\underline{x}, \underline{y}) = \Big(x_1(1+\varepsilon_1) \cdot y_1 (1+\varepsilon_2)(1+\varepsilon_3) \pm x_2(1+\varepsilon_4) \cdot y_2 (1+\varepsilon_5)(1+\varepsilon_6) \Big) \cdot (1+\varepsilon_7) = f(\tilde{\underline{x}}, \tilde{\underline{y}}) \qquad \tilde{\underline{x}} = (\tilde{x}_1, \tilde{x}_2) \quad \tilde{\underline{y}} = (\tilde{y}_1, \tilde{y}_2)$$

$$\begin{aligned} \tilde{x}_1 &= x_1(1+\varepsilon_1)(1+\varepsilon_7) & \tilde{y}_1 &= y_1(1+\varepsilon_2)(1+\varepsilon_3) \\ \tilde{x}_2 &= x_2(1+\varepsilon_4)(1+\varepsilon_7) & \tilde{y}_2 &= y_2(1+\varepsilon_5)(1+\varepsilon_6) \end{aligned}$$

En utilisant, $(1+\varepsilon_1)(1+\varepsilon_2) \leq 1 + 2\underbrace{\varepsilon_3}_{\max(|\varepsilon_1|, |\varepsilon_2|)} + \mathcal{O}(\varepsilon_{mach}) \quad |\varepsilon_i| \leq \varepsilon_{mach}$

On obtient $\dfrac{\|\tilde{x} - x\|}{\|x\|} \leq C \cdot \varepsilon_{mach} + O(\varepsilon_{mach})$ avec $C = 4 \cdot 2 = 8$

on considère la norme 1 pour avoir $C = 8$

Comme C petit, l'algo est backward stable

Rem: Avantage ici, pas besoin de calculer $K(f)$ pour étudier la backward stability.

Thm: Soit $\tilde{f}$ un algo backward stable pour un f problème alors $\tilde{f}$ est aussi forward stable.

Preuve: D'après la déf de back. stable, $\exists\, \tilde{x}$ tq $\tilde{f}(x) = f(\tilde{x})$

et $\dfrac{\|\tilde{x} - x\|}{\|x\|} \leq C \cdot \varepsilon_{mach} + O(\varepsilon_{mach})$

Soit K la condition de f en x, alors :

$$\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|} \leq K \cdot C \cdot \varepsilon_{mach} + \mathcal{O}(\varepsilon_{mach})$$

Puisque $f(\tilde{x}) = \tilde{f}(x)$ on obtient que $\tilde{f}$ est forward stable

Rem: On peut mq (série) que les normes en dim. finies (dans $\mathbb{R}^n$) sont équivalentes. Ainsi, si f est bien conditionnée pour 1 choix de norme, alors f l'est aussi pour les autres.
C'est vrai aussi pour les algo. qui peuvent être forward ou backward stable.

---

le 08.11.18

Ch 4. Intégration Numérique:

Étant donné une fonction $f: [a,b] \longrightarrow \mathbb{R}$, on cherche à approcher numériquement l'intégrale $\int_a^b f(x)\,dx$.

Rem: Il n'y a pas toujours de primitive avec une formule simple disponible (Ex: $f(x) = e^{-x^2}$)

Déf:
- Une formule de quadrature à s étages ($s \in \mathbb{N}^*$ fixé) est donnée par: $\int_0^1 g(t)\,dt \approx \sum_{i=1}^{s} b_i\, g(c_i)$, les c_i sont les noeuds 2 à 2 distincts et les b_i sont les poids.
- La formule de quadrature est dite d'ordre $p \in \mathbb{N}^*$ si elle exacte pour tous les polynômes de degré $\leq p-1$ c'est-à-dire: $\int_0^1 g(t)\,dt = \sum_{i=1}^{s} b_i g(c_i) \quad \forall g \in \mathbb{P}_{p-1}$

4.1 La formule de Newton-Cotes:

Déf: La formule de Newton-Cotes à s noeuds est obtenue avec:

$$c_i = \frac{i-1}{s-1}, \quad i = 1 \dots s$$



$$b_i = \int_0^1 \prod_{\substack{1 \leq j \leq s \\ j \neq i}} \frac{t - c_j}{c_i - c_j}\,dt = \int_0^1 l_i(t)\,dt$$

Analyse Numérique :

le 08.11.18

Thm : On considère une formule de quadrature $(b_i, c_i)_{i=1\dots s}$ à s étages.

La formule est d'ordre (au moins) p si et seulement si :

$$\sum_{i=1}^{s} b_i c_i^{q-1} = \frac{1}{q} \quad \forall q = 1, 2 \dots p$$

Preuve : $\Rightarrow$: Dans la définition de l'ordre au moins p, on choisit le cas particulier $g(t) = t^{q-1}$ avec $q = 1 \dots p$.

On obtient $\sum_{i=1}^{s} b_i c_i^{q-1} = \int_0^1 t^{q-1} dt = \left[\frac{t^q}{q}\right]_0^1 = \frac{1}{q}$

$\Leftarrow$: Réciproquement, si les monomes t^{q-1}, $q = 1 \dots p$ sont intégrés exactement, par linéarité de la formule de quadrature par rapport à g, la formule de quadrature est aussi exacte $\forall g \in \mathbb{P}_{p-1}$. En effet, $\{t^{q-1}, q = 1 \dots p\}$ est la base de monomes de $\mathbb{P}_{p-1}$.

Cor : Pour des noeuds c_i, $i = 1 \dots s$, fixés (et distincts 2 à 2), il existe une unique formule de quadrature d'ordre (au moins) p avec $p \geq s$.

C'est-à-dire $\exists !$ choix des poids b_i, $i = 1 \dots s$ pour avoir $p \geq s$.

De plus on a $b_i = \int_0^1 l_i(t) dt$ avec l_i, $i = 1 \dots s$ les polynômes de Lagrange associés aux noeuds $c_1, \dots c_s$.

Preuve : Pour $p = s$ dans le théorème précédent, on obtient

$$\sum_{i=1}^{s} b_i c_i^{q-1} = \frac{1}{q} \quad q = 1 \dots s \quad \text{soit matriciellement}$$

$$\begin{pmatrix} 1 & \cdots & 1 \\ c_1 & & c_s \\ c_1^{s-1} & & c_s^{s-1} \end{pmatrix} \begin{pmatrix} b_1 \\ \\ b_s \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{s} \end{pmatrix}$$

On reconnaît un système de Vander Monde, le déterminant de la matrice vaut $\prod_{1 \leq i < j \leq s} (c_j - c_i)$ et est donc non nul.

Preuve: (Suite)

Le système possède donc une unique solution.

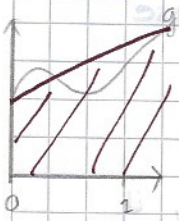
La formule de quadrature étant exacte pour $g(t) = l_i(t) \in \mathbb{P}_{s-1}$

on obtient $\int_0^1 l_j(t)\,dt = \sum_{i=1}^{s} b_i\, l_j(c_i) = b_j$

Ex: Parmis les formules de Newton - Cotes

	s	ordre	poids	nom	formule pour approcher $\int_0^1 g(t)\,dt$
$c_1 = 1/2$	1	2	$b_1 = 1$	point milieu	$g(1/2)$
	2	2	$b_1 = b_2 = 1/2$	trapèze	$(g(0) + g(1)) \cdot 1/2$
	3	4	$b_1 = b_3 = 1/6$ $b_2 = 2/3$	Simpson	$\frac{1}{6}(g(0) + 4g(1/2) + g(1))$

Ex: Formule de Simpson:



$\int_0^1 g(t)\,dt \approx \dfrac{g(0) + 4g(1/2) + g(1)}{6}$ est obtenue en interpolant $g(t)$ par une parabolle qui coincide avec $g(t)$ aux points $0, 1/2, 1$,

Calculons par exemple:

$$b_1 = \int_0^1 l_1(t)\,dt = \int_0^1 \frac{t - 1/2}{0 - 1/2}\,\frac{t - 1}{0 - 1}\,dt = \frac{1}{6} = b_3 \qquad b_2 = \frac{2}{3}$$

Thm: Soit $(b_i, c_i)_{i=1\dots s}$ une formule de quadrature supposée symétrique, c'est-à-dire avec $c_i = 1 - c_{s+1-i}$ $i = 1\dots s$ et $b_i = b_{s+1-i}$ $i = 1\dots s$

Alors son ordre est pair, c'est-à-dire la formule est d'ordre $p = 2m$ pour un certain $m \in \mathbb{N}$

Preuve: voir série # (19)

4.2 Erreur pour les formules de quadrature composées :

Pour approcher précisément une intégrale $\int_a^b f(t)\,dt$, on évite de faire tendre vers l'infini le nombre s de noeuds.

(Voir le phénomène de Runge pour l'interpolation de Lagrange)

On procède en décomposant l'intervalle $[a,b]$ avec une subdivision $a = x_0 < x_1 < \dots < x_N = b$ et on considère l'approximation suivante

$$\int_a^b f(t)\,dt = \sum_{j=0}^{N-1} \int_{x_j}^{x_{j+1}} f(t)\,dt \qquad \text{CDV : } t = x_j + \xi h_j \quad h_j = x_{j+1} - x_j$$

$$= \sum_{j=0}^{N-1} h_j \int_0^1 f(x_j + \xi h_j)\,d\xi \simeq \boxed{\sum_{j=0}^{N-1} h_j \sum_{i=1}^{s} b_i f(x_j + c_i h_j)}$$

C'est-à-dire, on applique la formule de quadrature N fois (sur chaque $[x_j, x_{j+1}]$) avec la fonction $g(\xi) = f(x_j + \xi h_j)$

Question : Quelle est l'erreur induite pour une formule de quadrature ainsi composée ?

On étudie l'erreur : $\text{err} = \left| \int_a^b f(t)\,dt - \sum_{j=0}^{N-1} h_j \sum_{i=1}^{s} b_i f(x_j + c_i h_j) \right|$

pour s fixé, et on fait $N \to +\infty$ avec également le pas

$$h = \max_{j=0\dots N-1} h_j \longrightarrow 0$$

Etudions d'abord l'erreur pour un seul sous-intervalle de longueur h :

$$E_s(f, x_0, h) = h \left(\int_0^1 f(x_0 + th)\,dt - \sum_{i=1}^{s} b_i f(x_0 + c_i h) \right)$$

Remarque : Dorénavant, on suppose que les poids de quadrature vérifient $b_i = \int_0^1 l_i(t)\,dt$ pour garantir un ordre au moins s.

Lemme : Considérons une formule de quadrature d'ordre au moins p.

Si $f : [x_0, x_0 + h]$ est C^p alors, $|E_s(f, x_0, h)| \le C \cdot h^{p+1} \max_{0 \le t \le 1} |f^{(p)}(x_0 + th)|$

avec C indépendant de f et h

Preuve : On montre en fait le résultat plus faible pour $h \to 0$

$$|E_s(f, x_0, h)| = C \cdot h^{p+1} \cdot |f^{(p)}(x_0)| + O(h^{p+2})$$

Si f est C^{p+1}, on peut développer $f(x_0 + \alpha h)$ en série de Taylor pour $\alpha \in [0,1]$, $f(x_0 + \alpha h) = \sum_{q=0}^{p} \frac{(\alpha h)^q}{q!} f^{(q)}(x_0) + \frac{(\alpha h)^{p+1}}{(p+1)!} f^{(p+1)}(\xi)$

où $\xi \in (x_0, x_0 + h)$

En remplacant $f(x_0 + th)$ et $f(x_0 + c_i h)$ par de telles séries dans la définition de $E(f, x_0, h)$, on obtient :

$$E_s(f, x_0, h) = h \left(\int_0^1 \sum_{q=0}^{p} \frac{(th)^q}{q!} f^{(q)}(x_0)\, dt - \sum_{i=1}^{s} \sum_{q=0}^{p} b_i \frac{(c_i h)^q}{q!} f^{(q)}(x_0) \right) + O(h^{p+2})$$

$$= \sum_{q=0}^{p} \frac{h^{q+1}}{q!} f^{(q)}(x_0) \underbrace{\left(\int_0^1 t^q\, dt - \sum_{i=1}^{s} b_i c_i^q \right)}_{E_s(t^q, 0, 1)} + O(h^{p+2})$$

Comme la formule de quadrature est d'ordre p, on a $E(t^q, 0, 1) = 0$ pour $0 \le q \le p-1$, on obtient :

$$E_s(f, x_0, h) = \frac{h^{p+1}}{p!} f^{(p)}(x_0) \underbrace{\left(\frac{1}{p+1} - \sum_{i=1}^{s} b_i c_i^p \right)}_{= C} + O(h^{p+2})$$

le 15.11.18

Thm : Erreur globale pour une formule de quadrature composée

Soit $f : [a,b] \to \mathbb{R}$ de classe C^p, on considère une formule de quadrature (b_i, c_i) $i = 1 \dots s$ d'ordre p.

Alors l'erreur $\text{err} = \left| \int_a^b f(x)\, dx - \sum_{j=0}^{N-1} h_j \sum_{i=1}^{s} b_i f(x_j + c_i h_j) \right|$ ($h_j = x_{j+1} - x_j$)

vérifie $\text{err} \le C \cdot h^p (b-a) \max_{a \le x \le b} |f^{(p)}(x)|$ avec $h = \max_{1 \le j \le N} h_j$

et C ne dépend pas de f et de h

Preuve : On remarque que err est la somme des err locales sur chaque ss-intervalle de $[a,b]$ pour $a = x_0 < x_1 \dots < x_N = b$

$$\text{err} \le \sum_{j=0}^{N-1} |E_s(f, x_j, h_j)| = \sum_{j=0}^{N-1} h_j \left| \int_0^1 f(x_j + th_j)\, dt - \sum_{i=1}^{s} b_i f(x_j + c_i h_j) \right|$$

$$\le \sum_{j=0}^{N-1} C h_j^{p+1} \max_{0 \le t \le 1} |f^{(p)}(x_j + th_j)| \le C \cdot h^p \underbrace{\sum_{j=0}^{N-1} h_j}_{(b-a)} \max_{a \le x \le b} |f^{(p)}(x)|$$

$$= C \cdot h^p \cdot (b-a) \cdot \max_{a \le x \le b} |f^{(p)}(x)|$$

4.3 Formules d'ordre supérieur:

On a vu avec les polynômes de Lagrange que pour choix de noeuds c_i, $i=1\dots s$, distincts 2 à 2, il existe un unique choix des poids b_i, $i=1\dots s$, pour atteindre un ordre (au moins) p

On a vu qu'on peut avoir $p>s$, en particulier pour s impair et une formule symétrique.

Question: Ordre maximal p pour s fixé ?

Lemme: Jacobi 1826

Soit (b_i, c_i) $i=1\dots s$ une formule de quad. d'ordre $p \geq s$ pour $[0,1]$. Alors l'ordre est au moins $s+m$ si et seulement si

$\forall q \in \mathbb{P}_{m-1}$, $\int_0^1 M(t)\,q(t)\,dt = 0$ où $M(t) = (t-c_1)\dots(t-c_s)$

Preuve: Soit f un polynôme de degré $\leq s+m-1$, on peut faire une division euclidienne de f par le poly. M, pour obtenir :

$f(t) = M(t)\,q(t) + r(t)$ avec $q \in \mathbb{P}_{m-1}$ $r \in \mathbb{P}_{s-1}$

Comme la formule de quad. est exacte pour r, on obtient :

$$\int_0^1 f(t)\,dt = \int_0^1 M(t)\,q(t)\,dt + \int_0^1 r(t)\,dt = \int_0^1 M(t)\,q(t)\,dt + \sum_{i=1}^{s} b_i\, r(c_i)$$

D'autre part, on a $\sum_{i=1}^{s} b_i f(c_i) = \sum_{i=1}^{s} b_i \left[\underbrace{M(c_i)}_{0}\, q(c_i) + r(c_i)\right]$

$= \sum_{i=1}^{s} b_i\, r(c_i)$

Ainsi $\int_0^1 f(t)\,dt = \int_0^1 M(t)\,q(t)\,dt + \sum_{i=1}^{s} b_i f(c_i)$

la formule de quad. est donc exacte $\forall f \in \mathbb{P}_{s+m-1}$

$\Leftrightarrow$ $\int_0^1 M(t)\,q(t)\,dt = 0$ $\forall q \in \mathbb{P}_{m-1}$

Question: Quelle est la valeur maximale de m dans le thm précédent tq l'implication se réalise ?

Thm : L'ordre d'une formule de quadrature à s étages est $\leq 2s$

Preuve : Par l'absurde, on suppose la formule d'ordre $\geq 2s+1$

par le thm précédent, $\int_0^1 M(t) q(t)\,dt = 0 \quad \forall q \in \mathbb{P}_s$

en particulier si $q = M$, on obtient : $\int_0^1 M^2(t)\,dt = 0$ ↯

↑ polynôme non-nul et positif.

Rem : On verra que $\exists$ une unique formule de quadrature avec $p = 2s$

Lemme : Soit $f \in C^0([a,b])$ avec $\forall x \in [a,b]\ f(x) \geq 0$, alors si

$\int_a^b f(t)\,dt = 0 \Rightarrow f \equiv 0$

Preuve : On va montrer la contraposée

Si $f \not\equiv 0 \Rightarrow \int_a^b f(t)\,dt \neq 0$

$\exists x_0 \in [a,b]$, tq $f(x_0) \neq 0$, comme est continue en x_0,

$\forall \varepsilon > 0, \exists \eta > 0, \forall y \in [a,b], |x_0 - y| \leq \eta \Rightarrow |f(x_0) - f(y)| \leq \varepsilon$

En choisissant $\varepsilon = \frac{|f(x_0)|}{2}$, $\exists \eta > 0$ tq $|f(x)| \geq \frac{|f(x_0)|}{2}\ \forall x \in]x_0 - \eta, x_0 + \eta[\cap [a,b] = I$. Alors $\int_a^b f(x)\,dx \geq \int_I f(x)\,dx$

$\geq \int_I \frac{|f(x_0)|}{2}\,dx = \frac{f(x_0)}{2}\,\text{longueur}(I) \neq 0$

4.4 Polynômes orthogonaux de Legendre :

Pour simplifier les calculs, on va travailler plutôt avec $[-1,1]$ au lieu de $[0,1]$ avec le C.D.V $\tau = 2t - 1$

But : Trouver des poly. $P_k(\tau)$ de degré exactement k tq $\forall q \in \mathbb{P}_{k-1}$, $\int_{-1}^1 P_k(\tau)\, q(\tau)\,d\tau = 0$

Ceci signifie géométriquement que P_k est $\perp$ à l'esp. vect $\mathbb{P}_{k-1}$ pour le prod. scalaire sur $\mathbb{P}$ déf par $\langle f, g \rangle = \int_{-1}^1 f(t) g(t)\,dt$

Ce sont les poly. de Legendre

Analyse Numérique :

le 15.11.18

On verra que ces poly sont uniques à cst multiplicative près, c-à-d αP_k est aussi un poly de Legendre de deg k $\forall \alpha \in \mathbb{R}^*$

Les P_k représentent un exemple d'une famille de poly. orthog. car $\langle P_k, P_l \rangle = 0 \quad \forall k \neq l$

On va voir que les P_k peuvent être construit avec l'algo de Graham-Schmidt.

Thm : Le polynôme de Legendre P_k existe $\forall k \in \mathbb{N}$ avec un degré exactement k. De plus $P_0 P_1 \dots P_k$ forment une base orthogonale de $\mathbb{P}_k$ pour le prod. scalaire $\langle \cdot, \cdot \rangle$

Preuve : Par récurrence sur k

Pour $k = 0$, rien à montrer / prouver

On suppose déjà construits les poly $P_0, \dots, P_k$

Construisons P_{k+1}

On considère $Q_{k+1}(x) = x P_k(x)$ de degré exactement $k+1$

On orthogonalise Q_{k+1} par la méthode de Graham-Schmidt et on pose : $P_{k+1} = Q_{k+1} - \sum_{j=0}^{k} \alpha_j P_j$ avec $\alpha_j = \dfrac{\langle Q_{k+1}, P_j \rangle}{\langle P_j, P_j \rangle}$

$\langle P_{k+1}, P_l \rangle = \langle Q_{k+1}, P_l \rangle - \sum_{j=0}^{k} \alpha_j \underbrace{\langle P_j, P_l \rangle}_{= 0 \text{ si } j \neq l} = \langle Q_{k+1}, P_l \rangle - \alpha_l \langle P_l, P_l \rangle$

$= 0$

Mq P_{k+1} vérifie toutes les propriétés demandées :

a) Le degré de Q_{k+1} est bien $k+1$ et on a bien le même degré pour P_{k+1} car on soustrait des poly $\leq k$ de degré

b) On a bien $\langle P_{k+1}, P_l \rangle = 0 \quad \forall l \neq k+1$

Et donc $P_0 \dots P_{k+1}$ forment une base de $\mathbb{P}_{k+1}$ car ils constituent $k+2$ "vecteurs" orthogonaux dans $\mathbb{P}_{k+1}$, esp vect de dim $k+2$

Preuve: (Suite)

c) En utilisant que $P_0 \dots P_k$ est une base orthog. de $\mathbb{P}_k$, chaque poly $g \in \mathbb{P}_k$ se décompose comme $g = \sum_{j=1}^{k} \beta_j P_j$ avec $\beta_j = \frac{\langle P_j, g \rangle}{\langle P_j, P_j \rangle}$, alors $\forall g \in \mathbb{P}_k$, on obtient :

$$\langle P_{k+1}, g \rangle = \sum_{j=0}^{k} \beta_j \underbrace{\langle P_{k+1}, P_j \rangle}_{0} = 0$$

---

le 22.11.18

Polynôme de Legendre : $P_0, P_1, P_2, \dots$

$$P_{k+1} = Q_{k+1} - \sum_{j=0}^{k} \frac{\langle Q_{k+1}, P_j \rangle}{\langle P_j, P_j \rangle} P_j$$

avec $Q_{k+1}(\tau) = \tau P_k(\tau)$

on remarque : $\langle Q_{k+1}, P_j \rangle = \int_{-1}^{1} \tau P_k(\tau) \underbrace{P_j(\tau)}_{\text{degré } j+1} d\tau = 0$ si $j+1 < k$ ($j \leq k-2$)

On déduit que la formule de Gramm-Schmidt se simplifie en une récurrence à 3 termes.

$$P_{k+1}(\tau) = \tau P_k(\tau) - \frac{\langle \tau P_k, P_k \rangle}{\langle P_k, P_k \rangle} P_k - \frac{\langle \tau P_k, P_{k-1} \rangle}{\langle P_{k-1}, P_{k-1} \rangle} P_{k-1}$$

On rappel que les poly. de Legendre sont définis à un facteur réel multiplicatif près.

En fixant $P_k(1) = 1 \quad \forall k \geq 0$ on a

Thm: Les polyn. de Legendre avec $P_k(1) = 1 \ \forall k$ vérifiant :

$$P_0(\tau) = 1, \quad P_1(\tau) = \tau, \quad (k+1) P_{k+1}(\tau) = (2k+1) \tau P_k(\tau) - k P_{k-1}(\tau) \quad \forall k \geq 1$$

Thm: Répartitions des racines des P_k

Toutes les racines de $P_k(\tau)$ $k \geq 0$ sont réelles, simples, dans $]-1,1[$

Pv: Soit $\tau_1, \dots \tau_r$ toutes les racines sans-répétitions de $P_k(\tau)$ qui sont réelles, dans $]-1,1[$ et où $P_k(\tau)$ change de signe localement autour de τ_j (multiplicité impaire)

But : Mq $r = k$, ainsi les racines seront simples.

Pr: (Suite)

On considère $q(\tau) = (\tau - \tau_1) \dots (\tau - \tau_r)$

On calcule: $\int_{-1}^{1} P_k(\tau)\, q(\tau)\, d\tau \neq 0$ (Polyn non nul tq signe change pas)

Or $P_k(\tau)$ est orthogonal aux polynômes de degré $< k$

donc $\deg q = k \Rightarrow n = k$. ∎

Rem: Le résultat précédent reste valable pour les polynômes orthogonaux avec un poids $w(x) \geq 0$ dans le produit scalaire $\langle p, q \rangle = \int_{-1}^{1} p(\tau) \cdot q(\tau) \cdot w(\tau)\, d\tau$

Ex: pour $w(x) = 1/\sqrt{1-x^2}$, on a les polyn. de Chebyshev.

4.5 Formule de quadrature de Gauss :

But: Construire une form. de quad d'ordre max $p = 2s$

Comme $P_s(\tau)$ est de degrés s avec racines simples dans $]-1,1[$ $c_1 \dots c_s$. On obtient $\exists C_s \neq 0$ tq $M(t) = (t - c_1) \dots (t - c_s)$ $= C_s P_s(2t-1)$

Par CDV: $\tau = 2t - 1 \Rightarrow \int_0^1 M(t)\, q(t)\, dt = \frac{C_s}{2} \int_{-1}^{1} P_s(\tau)\, q\left(\frac{\tau+1}{2}\right) d\tau$

$= 0 \quad \forall q \in \mathbb{P}_{s-1}$

Par le Lemme de Jacobi, on a

Thm: Formule de quadrature de Gauss

$\forall s \geq 1$, $\exists$ une formule de quadrature à s étages d'ordre $p = 2s$

Elle est donnée par $\int_0^1 g(t)\, dt \simeq \sum_{i=1}^{s} b_i\, g(c_i)$ avec

les noeuds $c_1, \dots, c_s \in]0,1[$, les s racines simples de $P_s(2t-1)$

les poids $b_i = \int_0^1 l_i(t)\, dt$ où $l_i(t)$ polynômes de Lagrange pour $c_1 \dots c_s$

Rem: La formule de Gauss est la seule formule de quadrature à s noeuds d'ordre maximal 2s.

En effet, le lemme de Jacobi est une équivalence.

Comme les polynômes de Legendre sont uniques (à facteur multiplic près) on obtient l'unicité de la formule de quadrature de Gauss d'ordre $p = 2s$

Chapitre 5 : Système d'équations linéaires :

On considère un système d'équations linéaires rectangle (pas autant d'équations que d'inconnues) : $\begin{cases} a_{11}x_1 + \dots + a_{1n}x_n = b_1 \\ a_{m1}x_1 + \dots + a_{mn}x_n = b_m \end{cases}$

En notation matricielle, on écrit $A\underline{x} = \underline{b}$ $\quad A = \begin{bmatrix} a_{11} & \dots \\ \vdots & a_{mn} \end{bmatrix}$ $\underline{x} = \begin{pmatrix} x_1 \\ x_m \end{pmatrix}$ $\underline{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$

Dans ce chapitre, on suppose $m = n$ (système carré) et on suppose A inversible (le système possède une solution et une seule)

5.1 Condition d'une matrice et résolution d'un système linéaire :

Soit $Ax = b$ un sys. lin. (A inversible)

Soit $\tilde{A}y = \tilde{b}$ un sys. perturbé

On aimerait estimer $\|x - y\|$ l'erreur due aux perturbations.

On suppose : $\tilde{a}_{ij} = a_{ij}(1 + \varepsilon_{ij}) \quad |\varepsilon_{ij}| \leq \varepsilon_A \quad \forall ij = 1 \dots n$

$\tilde{b}_i = b_i(1 + \varepsilon_i) \quad |\varepsilon_i| \leq \varepsilon_b \quad \forall i = 1 \dots n$

où ε_A et ε_b spécifient la précision des données

Déf: Soit A inversible. On appelle $K(A) = \|A\|\|A^{-1}\|$ la condition de A pour $\|.\|$

Rem : $K(A)$ dépend du choix de la norme. Si on choisit $\|.\|_{p \to p}$, on note $K_p(A)$. En pratique on utilise $K_1(A)$, $K_2(A)$, $K_\infty(A)$

Thm : Soit A inversible. Alors $\forall$ choix de norme d'opérateur $\|.\|$,

1) $K(A) \geq 1$

2) $K(\alpha A) = K(A) \quad \forall \alpha \neq 0$

3) $K(A) = \max_{\|y\|=1} \|Ay\| \Big/ \min_{\|z\|=1} \|Az\|$

Dém : 1) $1 = \|I\| = \|AA^{-1}\| \leq \|A\| \cdot \|A^{-1}\| = K(A)$

2) $(\alpha A)^{-1} = \frac{1}{\alpha} A^{-1}$

3) On remarque $\|A^{-1}\| = \max_{x \neq 0} \frac{\|A^{-1}x\|}{\|x\|} \underset{z = A^{-1}x}{=} \max_{z \neq 0} \frac{\|z\|}{\|Az\|}$

$= \left(\min_{z \neq 0} \frac{\|Az\|}{\|z\|} \right)^{-1} = \left(\min_{\|z\|=1} \|Az\| \right)^{-1}$

Ex : Si A est une matrice orthogonale, A préserve la norme 2 (isométrie) et alors $K_2(A) = 1$ (car $\|Ax\|_2 = \|x\|_2$)

ce qui mq les matrices orthogonales ont le meilleur conditionnement.

Ex : Les matrices de Hilbert H_n et Van der Monde V_n,

$$H_n = \begin{bmatrix} 1 & 1/2 & 1/3 & \cdots & 1/n \\ 1/2 & & & & 1/n+1 \\ \vdots & & & & \vdots \\ 1/n & \cdots & \cdots & \cdots & 1/2n+1 \end{bmatrix} \quad \text{et} \quad V_n = \begin{bmatrix} 1 & \cdots & \cdots & 1 \\ c_1 & & & c_n \\ c_1^2 & & & \vdots \\ c_1^{n-1} & & & c_1^{n-1} \end{bmatrix} \quad c_j = \frac{j}{n}$$

$$H_n = \left(\frac{1}{i+j-1} \right)_{i,j=1 \ldots n} \qquad V_n = \left(\left(\frac{j}{n} \right)^{i-1} \right)_{i,j=1 \ldots n}$$

sont très mal conditionnées.

le 29.11.18

Ex : Matrice bien conditionnée

$A = \frac{1}{h}\begin{bmatrix} 4 & 1 & & 0 \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & 4 \end{bmatrix} \in \mathbb{R}^{n\times n}$ c'est la matrice utilisée pour le calcul de spline.

Calculons $K_\infty(A)$, $\|A\|_\infty \leq \frac{1+4+1}{h} = \frac{6}{h}$

On remarque $A = \frac{4}{h}(I - N)$ avec $N = \begin{pmatrix} 0 & -1/4 & & 0 \\ -1/4 & 0 & \ddots & \\ & \ddots & \ddots & -1/4 \\ 0 & & -1/4 & 0 \end{pmatrix}$

$A^{-1} = \frac{h}{4}(I-N)^{-1} = \frac{h}{4}(I + N + N^2 + \dots)$

et la série $\sum N^k$ converge car $\|N\|_\infty < 1$

Donc $\|A^{-1}\|_\infty \leq \frac{h}{4}(1 + \|N\| + \|N^2\| + \dots) = \frac{h}{4} \cdot 2 = \frac{h}{2}$

Donc $K_\infty(A) \leq \frac{6}{h} \cdot \frac{h}{2} = 3$ bien conditionné.

Lemme : Soit N une matrice de norme d'opérateur vérifiant $\|N\| < 1$

Alors $I - N$ est inversible d'inverse $(I-N)^{-1} = \sum_{k\geq 0} N^k$

Preuve : Comme $\|N\| < 1$, la série $\sum \|N\|^k$ converge et $\sum N^k$ est convergente. On calcule $(I-N)\sum N^k = \sum N^k - \sum N^{k+1} = I$

donc $I - N$ est inversible d'inverse $\sum N^k$ □

Retour au problème à données perturbées.

$Ax = b$

$(A + \delta A)y = b + \delta b$ avec A carrée inversible

$\tilde{a}_{ij} = a_{ij}(1 + \varepsilon_{ij})$ $\quad |\varepsilon_{ij}| \leq \varepsilon_A$

$\tilde{b}_i = b_i(1 + \eta_i)$ $\quad |\eta_i| \leq \varepsilon_B$

Ces hypothèses impliquent pour les normes $\|\ \|_1$ et $\|\ \|_\infty$

$\|\tilde{A} - A\| \leq \varepsilon_A \|A\|$ et $\|\tilde{b} - b\| \leq \varepsilon_b \|b\|$

Thm : Soit A inversible et $x \neq 0$. Soit $y, b, \delta b$ tq $Ax = b$, $(A+\delta A)y = b + \delta b$ avec $\|\delta A\| \leq \varepsilon_A \|A\|$ et $\|\delta b\| \leq \varepsilon_b \|b\|$.

Si $\varepsilon_A K(A) < 1$, alors $A + \delta A$ est inversible et

$\frac{\|y-x\|}{\|x\|} \leq \frac{1}{1 - \varepsilon_A K(A)}\left(\varepsilon_A K(A) + \varepsilon_b \|A^{-1}\| \frac{\|b\|}{\|x\|}\right)$, de plus

$\frac{\|y-x\|}{\|x\|} \leq \frac{K(A)}{1 - \varepsilon_A K(A)}(\varepsilon_A + \varepsilon_b)$

Rem: Si $\varepsilon_A = \varepsilon_b = \varepsilon$, on obtient la borne $\frac{\|y-x\|}{\|x\|} \leq \frac{2\varepsilon K(A)}{1-\varepsilon K(A)} = 2\varepsilon K(A) + O(\varepsilon)$

Ainsi on peut voir que les perturbations sur les données sont amplifiée par un facteur proportionel à $K(A)$.

Preuve: $A(y-x) = \delta b - \delta A y = \delta b - \delta A(y-x) - \delta A x$

Ainsi, $y - x = A^{-1}(\delta b - \delta A(y-x) - \delta A x)$

On obtient : $\|y-x\| \leq \varepsilon_b \|A^{-1}\| \|b\| + \varepsilon_A \|A^{-1}\| \|A\| \|y-x\| + \varepsilon_A \|A\| \|A^{-1}\| \|x\|$

et aussi $(1 - K(A)\varepsilon_A)\|y-x\| \leq \varepsilon_b \|A^{-1}\| \|b\| + \varepsilon_A K(A) \|x\|$

On remarque que si $x = 0 = b = \delta b$ on obtient

$(1 - \varepsilon_A K(A))\|y-x\| \leq 0 + 0$ et donc $y = 0$

C'est à dire $(A + \delta A)y = 0 \Rightarrow y = 0$

$A + \delta A$ a un noyeau nul donc est inversible.

Les estimations s'obtiennent en divisant par $\|x\| \neq 0$

Rem: $K(A)$ mesure la sensibilité de la solution de $Ax = b$ aux petites perturbations de A et b.

Cette sensibilité correspond en fait à la notion de condition du ch. 3 pour résoudre $Ax = b$, avec A inversible et b donnés.

Déf: On considère le conditionement comme définit au ch. 3

$$K_{Ab} = \lim_{\varepsilon \to 0} \sup \left\{ \frac{\|\tilde{A}^{-1}\tilde{b} - A^{-1}b\|}{\|A^{-1}b\|} ; \frac{\|\tilde{A} - A\|}{\|A\|} \leq \varepsilon \text{ et } \frac{\|\tilde{b} - b\|}{\|b\|} \leq \varepsilon \right\}$$

Cor: La condition de résolution de $Ax = b$ vérifie

$$K_{Ab} = K(A) + \|A^{-1}\| \frac{\|b\|}{\|A^{-1}b\|} \leq 2K(A)$$

5.2 Elimination de Gauss :

La méthode (ou élimination) de Gauss est l'algorithme le plus naturel pour résoudre un système $A\underline{x} = \underline{b}$

$$* \quad \begin{cases} a_{11}x_1 + \dots + a_{1n}x_n = b_1 \\ \vdots \\ a_{n1}x_1 + \dots + a_{nn}x_n = b_n \end{cases}$$

Rem: Si on soustrait une ligne $\lambda \neq 0$ fois sur une autre, on obtient un système équivalent. En effet, cette opération sur les lignes est réversible, d'inverse "soustraire $-\lambda$ fois"

si $a_{11} \neq 0$, on choisit a_{11} comme pivot et on élimine la variable x_1 dans les lignes du systèmes de 2 à n.

On calcule $\ell_{i1} = \dfrac{a_{i1}}{a_{11}} \quad i = 2, \dots, n$

et on remplace la ligne $i = 2 \dots n$ par ligne i - ℓ_{i1} × ligne 1

$$\begin{cases} a_{11}x_1 + \dots + a_{1n}x_n = b_1 \\ 0 + a_{22}^{(1)}x_2 + \dots = b_2^{(1)} \\ \vdots \\ 0 + a_{n2}^{(1)}x_2 + \dots + a_{nn}^{(1)}x_n = b_n^{(1)} \end{cases}$$

avec $a_{ij}^{(1)} = a_{ij} - \ell_{i1} \cdot a_{1j}$

$b_i^{(1)} = b_i - l_{i1} \cdot b_1$ pour $ij = 2 \dots n$

on applique la même procédure au système $(n-1) \times (n-1)$ défini par les nouvelles équations pour $x_2 \dots x_n$, et ainsi de suite jusqu'à atteindre un système 1×1. à résoudre.

L'étape k s'écrit donc (le pivot $a_{kk}^{(k-1)} \neq 0$)

$$\ell_{ik} = \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}$$

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \ell_{ik} \cdot a_{kj}^{(k-1)}$$

$$b_i^{(k)} = b_i^{(k-1)} - \ell_{ik} \cdot b_k^{(k-1)} \qquad i, j = k+1, \dots n$$

Après $n-1$ étapes, on obtient le système triangulaire équivalent suivant.

$$\begin{cases} a_{11}x_1 + \dots + a_{1n}x_n = b_1 \\ 0 + a_{22}x_2 + \dots = b_2 \\ \vdots \\ 0 + 0 + 0 \quad a_{nn}x_n = b_n \end{cases}$$

Ce système se résout facilement :

$x_n = b_n / a_{nn}$

$\vdots$

$$x_i = \left(b_i - \sum_{j=i+1}^{n} a_{ij}x_j \right) / b_{ii}$$

où a et b sont différents de $*$

Analyse Numérique : le 29.11.18

Rem : On vient de décrire l'algorithme de Gauss sans recherche du pivot.
Si à l'étape i, $a_{ii}^{(i-1)} = 0$, on ne peut pas le choisir comme pivot. (voir plus loin dans le cours)

Coût de l'algorithme de Gauss : (opération élémentaire $+, -, /, \cdot$)

Calcul	Coût de l'étape			total
	$k=1$	k	$k=n-1$	
Lik	$n-1$	$n-k$	1	$(n-1)n/2 \sim 1/2\, n^2$
$a_{ij} \longrightarrow a_{ij}^{(k)}$	$2(n-1)^2$	$2(n-k)^2$	2	$\sim 2/3\, n^3$
$b_i \longrightarrow b_i^{(k)}$	$2(n-1)$	$2(n-k)$	2	$\sim n^2$

Total $\sim 2/3\, n^3$ opérations élém.

Rem : Pour des matrices particulières structurellement, le coût peut être nettement inférieur.
Par exemple, l'exemple au début de cours, le coût est $O(n)$

---

le 06.12.18

5.3 Factorisation LU :

Nous allons voir que l'algorithme du pivot de Gauss correspond en fait à la factorisation $LU = A$ d'une matrice A avec L triangulaire inférieur et U triang. sup.

Rep : Algo de Gauss avec $A \in \mathbb{R}^{4\times 4}$

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \xrightarrow{L_1} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix} \xrightarrow{L_2} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \xrightarrow{L_3} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}$$

$A = A^{(0)}$ $\quad A^{(1)} = L_1 A$ $\quad A^{(2)} = L_2 A^{(1)}$ $\quad A^{(3)} = L_3 A^{(2)}$

× ubr qui changent.

On peut vérifier que l'opération de l'algo de Gauss à l'étape k correspond en fait à multiplier à gauche par la matrice L_k triang. inf définie par

$$L_k := \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & -l_{k+1,k} & \ddots \\ & & \vdots & \ddots \\ & & -l_{nk} & & 1 \end{pmatrix} \quad \text{(colonne } k\text{)}$$

Par conséquent, l'algo du pivot de Gauss sans recherche du pivot équivaut à : $L_{n-1} \cdots L_2 \cdot L_1 \cdot A = \underbrace{U}_{A^{(n-1)}}$ avec U triang. supérieure.

En posant, $L^{-1} = L_{n-1} \cdot \ldots \cdot L_2 \cdot L_1$ on a $L^{-1} \cdot A = U$ et donc $A = LU$

Lemme: On considère les matrices L et L_k définies ci-dessus.

Alors $L_k^{-1} = \begin{pmatrix} 1 & & & \\ & \ddots & & 0 \\ & & 1 & \\ & & L_{i+1,k} & \ddots \\ & & \vdots & & 1 \\ & & L_{nk} & & \end{pmatrix}$ et $L = L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1} = \begin{pmatrix} 1 & & & \\ L_{21} & 1 & & \\ L_{31} & L_{32} & \ddots & 0 \\ \vdots & \vdots & \ddots & 1 \\ L_{n1} & & & \end{pmatrix} = (L_{n-1} \cdot \ldots \cdot L_1)^{-1}$

Preuve:

L'inverse de soustraire la ligne k, sur les lignes $i = k+1 \ldots n$ avec un facteurs L_{ik} est la même opération en remplacant L_{ik} par $-L_{ik}$.

Pour prouver la formule de L, on remarque que le produit $L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1}$ $= L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1} \cdot I$ correspond à effectuer les opérations L_{n-1}^{-1} puis L_{n-2}^{-1} etc aux lignes de I

Preuve: (Autre) On considère:

$e_k = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{pmatrix} \leftarrow k \in \mathbb{R}^n$, $\ell_k = \begin{pmatrix} 0 \\ \ell_{k+1,k} \\ \vdots \\ \ell_{nk} \end{pmatrix}$, alors $L_k = I - \ell_k \cdot e_k^T$

comme $e_k^T \cdot \ell_k = 0$, on obtient: $(I - \ell_k e_k^T)(I + \ell_k e_k^T) = I + \ell_k e_k^T \ell_k e_k^T$ $= I$, Ainsi $(I + \ell_k e_k^T) = (I - \ell_k e_k^T)^{-1}$

En pratique, on calcule directement L sans chercher ni les L_k ni les L_k^{-1}.

Le coût de la factorisation LU est donc le même que pour l'algo du pivot de Gauss: $\sim \frac{2}{3} n^3$

(ici pas de 2nd membre $\underline{b}$ comme dans $Ax = b$)

Rem: Une fois calculée la décomposition $L \cdot U = A$ (coût $\sim 2/3\, n^3$) on peut résoudre un ou plusieurs systèmes $A \cdot x^{(l)} = b^{(l)}$ $l = 1 \ldots$ avec un coût de seulement $O(n^2)$ pour chaque système. $L \underbrace{U x^{(l)}}_{y^{(l)}} = b^{(l)} \Leftrightarrow \begin{cases} L y^{(l)} = b^{(l)} \\ U x^{(l)} = y^{(l)} \end{cases}$

5.4 Changement de pivot :

Considérons $A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$, 0 ne peut pas être le pivot et pourtant A est bien inversible, $A^{-1} = \frac{1}{-1}\begin{bmatrix} 1 & -1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}$.

De plus, A est bien conditionnée :

$$K_1(A) = \|A\|_1 \|A^{-1}\|_1 = 2 \cdot 2 = 4$$

Ici, on peut bien sûre échanger les lignes de A pour trouver un pivot.

On perturbe légèrement A : $\hat{A} = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}$, $\hat{A}^{-1} = \frac{1}{10^{-20}-1}\begin{bmatrix} 1 & -1 \\ -1 & 10^{-20} \end{bmatrix}$

$$K(\hat{A}) = 2 \cdot {}^{2}/_{1-10^{-20}} \simeq 4$$

On peut maintenant appliquer l'algorithme de Gauss (sans recherche de pivot)

$$\begin{cases} 10^{-20} x_1 + x_2 = 1 \\ x_1 + x_2 = 2 \end{cases} \qquad \begin{matrix} 10^{-20} x_1 + x_2 = 1 \\ 0 \cdot x_1 + (1 - 10^{20}) x_2 = 2 - 10^{+20} \end{matrix} \qquad \begin{matrix} x_1 \simeq 0 \\ x_2 \simeq 1 \end{matrix}$$

En supposant le calcul effectué en double précision ($\varepsilon_{mach} \simeq 10^{-16}$), la résolution du système triangulaire devient : $\hat{x}_2 = \frac{fl(2 - 10^{10})}{fl(1 - 10^{20})} = \frac{10^{20}}{10^{20}} = 1$

$$\hat{x}_1 = \frac{fl(1 - \hat{x}_2)}{fl(10^{-20})} = 0 \quad \text{⚠ Erreur de 100\%}$$

Avec la recherche de pivot par colonne (recherche partielle) :

$$\begin{cases} x_1 + x_2 = 2 \\ 10^{-20} x_1 + x_2 = 1 \end{cases} \qquad \begin{cases} x_1 + x_2 = 2 \\ 0 \cdot x_1 + (1 - 10^{-20}) x_2 = 1 - 2 \cdot 10^{-20} \end{cases}$$

on obtient avec la double précision. $\tilde{x}_2 = \frac{fl(1 - 2 \cdot 10^{-20})}{fl(1 - 10^{-20})} = 1$

$$\hat{x}_1 = fl(2 - \tilde{x}_2) = 1$$

C'est la solution optimale en double précision.

L'erreur relative est inférieure à $\varepsilon_{mach} \sim 10^{-16}$

Rem : Si le facteur l_{21} est très grand (ici $l_{21} = 10^{20}$)

alors $a_{22}^{(1)} = a_{22} - l_{21} a_{12} \simeq - l_{21} a_{12}$

$$b_2^{(1)} = b_2 - l_{21} b_1 \simeq - l_{21} b_1$$

on obtient $x_2 = b_2^{(1)} / a_{22}^{(1)} \simeq b_1 / a_{12}$ assez précis

$$x_1 = (b_1 - a_{12} x_2) / a_{11}$$

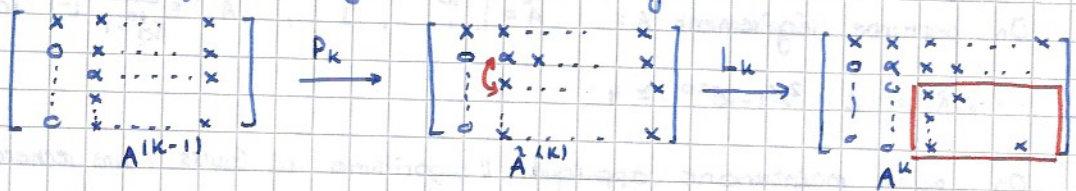
↑ mal conditionné $b_1 \simeq a_{12} x_2$

Rem: En pratique, il faut toujours faire une recherche de pivot, sauf dans le cas particulier où les points sur la diagonale sont déjà les meilleurs.

5.5 Factorisation LU avec changement de pivot :

A chaque étape k, on échange deux lignes pour obtenir le + grand pivot $\alpha = a_{ik}^{(k-1)}$ en valeur absolue (avec $i \geq k$).

Ainsi, on échange la ligne k avec la ligne i $(i > k)$



On peut interpréter l'échange des lignes k et i aussi matriciellement

$$\tilde{A}^{(k)} = P_k A^{(k-1)} \quad \text{avec} \quad P_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 0 & \cdots & 1 \\ & & \vdots & \ddots & \vdots \\ & & 1 & \cdots & 0 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow \text{ligne } k \\ \\ \leftarrow \text{ligne } i \end{matrix}$$

De plus $P^{-1} = P^T$ et P est orthogonale et $\det P = \pm 1$

P est une matrice de permutation.

Rem: Si $\alpha = 0$, tous les éléments de la colonne k sous la diagonale sont nuls et A n'est pas inversible. L'étape k est terminée et $P_k = L_k = I$.

Ainsi à chaque étape k on fait l'opération $A^{(k)} = L_k P_k A^{(k-1)}$

après $n-1$ étapes, on obtient : $L_{n-1} P_{n-1} \dots L_1 P_1 A = U$

En fait cette décomposition équivaut à $PA = LU$ avec P matrice de permutation.

Idée: Permuter les lignes de A au départ pour avoir déjà les pivots dans la bonne ligne.

Rap: Lorsqu'on parle de la décomposition LU d'une matrice, il s'agit en général de la décomposition "PLU" $PA = LU$

Cette décomposition a un coût $O(n^3)$ opérations élémentaires (en général)

Ensuite, pour résoudre $A\underline{x} = \underline{b}$, on remarque :

$$A\underline{x} = \underline{b} \iff PA\,\underline{x} = P\underline{b} \iff LU\underline{x} = P\underline{b}$$

Algo pour résoudre $Ax = b$:

1) Calculer $\underline{c} = bP\underline{b}$ (permut. des éléments coût $O(n)$)
2) Résoudre $L\underline{y} = \underline{c}$ (système triang. coût $O(n^2)$)
3) Résoudre $U\underline{x} = \underline{y}$ (——— " — ")

5.6 Implémentation :

$$u_{i,k:m} = (u_{ik} \dots u_{im})$$

Algo : Elimination de Gauss avec changement de pivot partielle

$U \leftarrow A$, $L \leftarrow I$, $P \leftarrow I$

for $k = 1 \dots n-1$ do

Chercher $i \geq k$ tq $|u_{ik}|$ soit maximal

if $u_{ik} \neq 0$ then

$u_{k,k:m} \longleftrightarrow u_{i,k:m}$
$L_{k,1:k-1} \longleftrightarrow L_{i,1:k-1}$
$P_{k,1:n} \longleftrightarrow P_{i,1:n}$
} échange des lignes i et k.

for $j = k+1 \dots n$ do

$L_{jk} \leftrightarrow u_{jk} / u_{kk}$

$u_{j,k:m} \longleftrightarrow u_{j,k:m} - l_{jk} \cdot u_{k,k:m}$

(ligne $j \leftarrow$ ligne j - l_{jk} ligne k)

Une autre implémentation :

On ignore maintenant le changement de pivot (+ simple)

On remarque que la décomp $A = LU$ équivaut à

$$a_{ij} = (LU)_{ij} = \sum_{r=1}^{\min(i,j)} L_{ir} \cdot u_{rj} \quad \text{car} \quad L_{ir} = u_{ri} = 0 \text{ si } r > i$$

En supposant connues les $k-1$ premières colones de L et U on remarque que : $a_{kj} = L_{k1} u_{1,j} + \dots + L_{k,k-1} u_{k-1,j} + L_{kk} u_{kj} \quad j = k \dots n$

$$= L_{k,1:k-1} u_{1:k-1,j} + u_{kj}$$

$$a_{ik} = L_{i,1} u_{1,k} + \dots + L_{i,k} u_{k,k} \quad i = k+1 \dots n$$

$$= L_{i,1:k-1}^{T} u_{1:k-1,k} + L_{ik} u_{kk}$$

Algo : Pivot de Gauss, version de Doolittle (sans recherche de pivot)

$U \longleftarrow A$, $L \leftarrow I$

for $k = 1 \dots n$

  for $j = k, \dots n$

    $u_{kj} \longleftarrow a_{kj} - L_{k,1:k-1}^{T} u_{1:k-1,j}$

  for $i = k+1 \dots n$

    $L_{ik} \longleftarrow (a_{ik} - L_{i,1:k-1}^{T} u_{1:k-1,k}) / u_{kk}$

On observe qu'il n'y a pas besoin de stocker les variables intermédiaires $a_{ij}^{(k)}$

5.4 Stabilité de la factorisation LU :

Comme échanger les lignes n'introduit pas d'erreur, on suppose les permutations nécessaires de la recherche des pivots déjà effectuées.

Ainsi, on a la décomposition exacte $A = LU$.

Soit $\tilde{L}$ et $\tilde{U}$ les matrices obtenues numériquement avec l'algo de Gauss pour la décomposition LU et $\tilde{A} = \tilde{L}\tilde{U}$

($\tilde{L}\tilde{U}$ est la décomposition LU exacte de $\tilde{A}$, c'est l'analyse rétrograde ou backward error analysis)

On va mq l'algo est backward stable ($\Rightarrow$ forward stable)

$$\frac{\|\tilde{A} - A\|}{\|A\|} \leq C \cdot \varepsilon_{mach} + o(\varepsilon_{mach}) \quad \text{avec } C \text{ pas trop grand.}$$

Thm : Wilkinson

Soit $A = LU$ et $\tilde{A} = \tilde{L}\tilde{U}$ avec $\tilde{L}\tilde{U}$ le résultat de l'algorithme de Gauss avec les erreurs d'arrondi. (pivots déjà en place)

Alors $|\tilde{a}_{ij} - a_{ij}| \leq 3\alpha \min(i-1, j)\, \varepsilon_{mach} + O(\varepsilon_{mach}^2)$

avec $\alpha = \max_{i,j,k} |a_{ij}^{(k)}|$

Preuve : A l'étape k, on calcule pour $i, j > k$, $\tilde{a}_{ij}^{(k)} = (\tilde{a}_{ij}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)}(1+\varepsilon_{ijk}))\cdot(1+\mu_{ijk})$ avec $|\varepsilon_{ijk}| \leq \varepsilon_{mach}$ et $|\mu_{ijk}| \leq \varepsilon_{mach}$

$$\tilde{a}_{ij}^{(k)} = \tilde{a}_{ij}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)}\varepsilon_{ijk} + (\tilde{a}_{ij}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)})\mu_{ijk} + O(\varepsilon_{mach}^2)$$

$$= \tilde{a}_{ij}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)} + \eta_{ijk} \quad (*)$$

De plus $|\eta_{ijk}| = |-\tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)}\varepsilon_{ijk} + (\tilde{a}_{ij}^{(k-1)} - \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)})\mu_{ijk}| + O(\varepsilon_{mach}^2)$

$$\leq |\tilde{l}_{ik}||\tilde{a}_{kj}^{(k-1)}||\varepsilon_{ijk}| + |\tilde{a}_{ij}^{(k)}||\mu_{ijk}| + O(\varepsilon_{mach}^2)$$

$$\leq 2\alpha\, \varepsilon_{mach} + O(\varepsilon_{mach}^2)$$

On remarque que $(*)$ est encore valide pour $i > j = k$ (partitionnement inférieur) car $\tilde{l}_{ij} = \frac{\tilde{a}_{ij}^{(j-1)}}{a_{jj}^{(j-1)}}(1+\varepsilon_{ijj}) \Rightarrow$

$$0 = \tilde{a}_{ij}^{(j)} = \tilde{a}_{ij}^{(j-1)} - \tilde{l}_{ij}a_{jj}^{(j-1)} + \mu_{ijj}$$

avec $|\mu_{ijj}| = |a_{ij}^{(j-1)}\varepsilon_{ijj}| \leq 2\alpha\, \varepsilon_{mach}$

La définition de $\tilde{A}$ implique : $\tilde{a}_{ij} = \sum_{k=1}^{\min(i,j)} \tilde{l}_{ik}\tilde{a}_{kj} = \sum_{k=1}^{\min(i,j)} \tilde{l}_{ik}\tilde{a}_{kj}^{(k-1)}$

et $(*)$ donne : $\tilde{l}_{ij}a_{kj}^{(k-1)} = \tilde{a}_{ij}^{(k-1)} - \tilde{a}_{ij}^{(k)} + \mu_{ijk}$ $\quad i > k,\ j \geq k$

Preuve : (Suite)

Premier cas : $i > j$ (Partie strictement inférieure)

$$\hat{a}_{ij} = \sum_{k=1}^{j} \tilde{L}_{ik}\, \tilde{a}_{kj}^{(k-1)} = \sum_{k=1}^{j} \left(a_{ij}^{(k-1)} - a_{ij}^{(k)} + \mu_{ijk}\right)$$

$$= a_{ij}^{(0)} - a_{ij}^{(j)} + \sum_{k=1}^{j} \mu_{ijk}$$

et ainsi $|\hat{a}_{ij} - a_{ij}| \leq 2\alpha j\, \varepsilon_{mach} + O(\varepsilon_{mach})$

Deuxième cas : $i \leq j$ (partie supérieure)

$$\hat{a}_{ij} = \sum_{k=1}^{i-1} \left(\hat{a}_{ij}^{(k-1)} - a_{ij}^{(k)} + \mu_{ijk}\right) + \tilde{L}_{ii}\, a_{ij}^{(i-1)}$$

$$= a_{ij} - a_{ij}^{(i-1)} + a_{ij}^{(i-1)} + \sum_{k=1}^{i-1} \mu_{ijk}$$

$$|\hat{a}_{ij} - a_{ij}| \leq 2\alpha (i-1)\, \varepsilon_{mach} + O(\varepsilon_{mach})$$

on obtient ainsi le résultat.

Rem : L'élimination de Gauss est backward stable si $\alpha / \max_{ij} |a_{ij}| \, \gamma(A)$ n'est pas trop grand.

$$|a_{ij}^{(k)}| = |a_{ij}^{(k-1)} - L_{ik}\, a_{kj}^{(k-1)}| \leq |a_{ij}^{(k-1)}| + |L_{ik}|\, |a_{kj}^{(k-1)}|$$

$$\leq 2 \max_{i,j} |a_{ij}^{(k-1)}| \leq 2^2 \max_{i,j} |a_{ij}^{(k-2)}| \leq \dots \leq 2^{k-1} \max_{ij} |a_{ij}|$$

ainsi $\dfrac{\alpha}{\max_{i,j} |a_{ij}|} \leq 2^{n-1}$

Il existe des matrices "pathologiques" où cette grande borne 2^{n-1} est atteintes. Mais pour la plus part des matrices on trouve une valeur modérée. $\gamma(A) \ll 2^{n-1}$

Ex : On a $\gamma(A) = 2^{n-1}$ pour la matrice $A = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ -1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -1 & \cdots & -1 & 1 \end{pmatrix}$

Ex : Pour une matrice "aléatoire" $\gamma(A) \simeq O(n^{0,7})$

Analyse Numérique :

le 13.12.18

Rem : Comme $\max_{i,j} |a_{ij}| \leq \|A\|_\infty$, on obtient $\frac{\|\tilde{A} - A\|_\infty}{\|A\|_\infty} \leq 3\,\gamma(A)\,n^2\,\varepsilon_{mach} + O(\varepsilon_{mach}^2)$

Rem : Concernant les recherches de pivots :

Autres stratégies :
- Recherche totale (on permute les lignes et les colonnes)
- Recherche avec seuil (on accepte comme pivot a_{ik} tq $|a_{ik}| \geq \alpha \max_{i,j} |a_{ij}|$ avec $\alpha \in (0,1)$ fixé)
- Pivotage diagonal (on cherche le + grand élément sur la diagonale en échangent lignes et colonnes)

Utile p. ex pour A tridiag. définie positive.

---

le 20.12.18

5.8 Factorisation de Cholesky :

On va étudier une variante de l'algo de Gauss dans le cas particulier important où A est symétrique. ($A = A^T$)

Déf : Une matrice A est dite symétrique définie positive (s.d.p) si $A = A^T$ et $\forall x \in \mathbb{R}^n \setminus \{0\}$ $x^T A x > 0$

Une matrice A symétrique est dite semi-définie positive si $\forall x \in \mathbb{R}^n$ $x^T A x \geq 0$

Rem : Si $A = A^T$, par le thm spectral, on peut diagonaliser A en base orthonormée, ie $\exists\, Q$ orthogonale ($Q^T Q = I$) tq $A = Q \Lambda Q^T$ où $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ est la matrice diagonale des valeurs propres de A. ($\lambda_i \in \mathbb{R}$)

- Pour $x \neq 0$, on a : $x^T A x = x^T Q \Lambda \underbrace{Q^T x}_{y} = y^T \Lambda y = \lambda_1 y_1^2 + \dots + \lambda_n y_n^2$
$= \sum_{i=1}^n \lambda_i y_i^2 \geq \lambda_{min} \|y\|_2^2$ avec $\lambda_{min} = \min_{i=1..n} \lambda_i$

Ainsi pour $A = A^T$, on a A s.d.p $\Leftrightarrow$ $\lambda_{min} > 0$

• $\Leftarrow$ Ok

$\Rightarrow$ on choisit $y = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$ ← position i_0 où $\lambda_{i_0} = \lambda_{min}$

Rem: $x = Q e_{i_0}$ = i_0-è colonne de Q = vect. prop. associé à λ_{min}

$0 < x^T A x = \lambda_{min}$

Ex: Soit $A = G^T G$ avec $G \in \mathbb{R}^{m \times n}$ $(m \geq n)$, alors A s.d.p $\Leftrightarrow$ rang $G = n$

En effet, pour $x \neq 0$

$$x^T A x = x^T G^T G x = (Gx)^T Gx = \|Gx\|_2^2 > 0$$

En particulier, pour G inversible, $A = G^T G$ est s.d.p

Thm: Propriétés de matrices s.d.p.

Soit $A \in \mathbb{R}^{n \times n}$ s.d.p, alors :

a) si $L \in \mathbb{R}^{m \times n}$ est de rang (m), alors $LAL^T \in \mathbb{R}^{m \times m}$ est s.d.p

Pv: On a :

$$y^T LAL^T y = (L^T y)^T A (L^T y) = x^T A x > 0$$

b) Les éléments de A sur la diagonale sont strictement positifs, $\forall i = 1 \ldots n$ $a_{ii} > 0$

Pv: on pose $x = e_i = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ alors $0 < e_i^T A e_i = a_{ii}$

c) Toutes les ss-matrices principales de A sont s.d.p.

c-à-d $\forall J \subset \{1 \ldots n\}$, $J \neq \emptyset$, $A_{J,J}$ est s.d.p.

Pv: On remarque que $A_{J,J}$ peut s'écrire : $A_{J,J} = L A L^T$ avec $L^T = [e_{j_1}, \ldots, e_{j_m}]$ où $J = \{j_1, \ldots, j_m\}$ (et $Rang(L^T) = m$)

on obtient le résultat en utilisant a)

d) $\det(A) > 0$ et A inversible

Pv: $\det(A) = \lambda_1 \ldots \lambda_n > \lambda_{min}^n > 0$

e) L'élément en val abs maximal de A se trouve toujours sur la diagonale.

Thm : (Suite)

Pv : On suppose par l'absurde que $|a_{ij}|$ est maximal pour $i \neq j$

alors la ss-matrice principale $\begin{pmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{pmatrix}$ est sdp et son déterminant est > 0.

$0 < a_{ii}a_{jj} - a_{ij}^2 \leq 0$ ↯

Pour des matrices s.d.p, ∃ une factorisation plus économique que LU :

Thm : Factorisation de Cholesky - Elimination de Gauss symétrique

Soit A s.d.p., alors ∃ une matrice triang. inférieure L, telle que $A = LL^T$

Rem : L est inversible car A est inversible

Rem : On peut montrer la ⩞ chose avec une matrice triang. supp.

Preuve : Soit $A = \begin{bmatrix} a_{11} & v^T \\ v & B \end{bmatrix}$ $a_{11} \in \mathbb{R}$ $v \in \mathbb{R}^{n-1}$ $B \in \mathbb{R}^{(n-1)\times(n-1)}$

On considère : $(a_{11} > 0)$ $L_1 = \begin{bmatrix} 1/\sqrt{a_{11}} & 0 \\ -v/\sqrt{a_{11}} & I \end{bmatrix}$

On calcule : $L_1 A L_1^T = \begin{bmatrix} 1 & 0 \\ 0 & \underbrace{B - \frac{vv^T}{a_{11}}}_{A^{(1)}} \end{bmatrix}$

On pose $A^{(1)} = B - \frac{vv^T}{a_{11}}$ qui est s.dp comme ss.-matrice principale de $L_1 A L_1^T$ (s.d.p car L_1 inversible)

On répète l'opération avec la matrice $A^{(1)}$ pour obtenir $\tilde{L}_2 A^{(1)} \tilde{L}_2^T$ de la forme $\tilde{L}_2 A \tilde{L}_2^T = \begin{pmatrix} 1 & 0 \\ 0 & A^{(2)} \end{pmatrix} \in \mathbb{R}^{(n-2)\times(n-1)}$ et on pose $L_2 = \begin{pmatrix} 1 & 0 \\ 0 & \tilde{L}_2 \end{pmatrix} \in \mathbb{R}^{n\times n}$ et ainsi de suite.

On obtient : $\underbrace{L_n \dots L_2 L_1}_{L^{-1}} A \underbrace{L_1^T L_2^T \dots L_n^T}_{L^{-T}} = I$

ce qui implique $A = LL^T$

Pour calculer L directement on peut modifier l'algorithme de Gauss.

On compare les coefficients dans l'égalité $A = LL^T$,

$i = k$, $\quad a_{kk} = L_{k1}^2 + L_{k2}^2 + \dots + L_{kn}^2$

$i > k$, $\quad a_{ik} = L_{i1} L_{k1} + \dots + L_{in} L_{kn}$

et on obtient l'algorithme suivant :

for $k = 1 \dots n$ do

$$L_{kk} \longleftarrow \left(a_{kk} - L_{k,1:k-1} L_{k,1:k-1}^T \right)^{1/2}$$

for $i = k+1 \dots n$ do

$$L_{ik} \longleftarrow \left(a_{ik} - L_{i,1:k-1} L_{k,1:k-1}^T \right) / L$$

Nombre d'opération :

En négligeant le coût des n racines ($O(n)$), le nombre d'opération est : $\sum_{k=1}^{n} 2(n-k)k \simeq 2 \int_0^n (n-x)x\,dx = n^3/3$

ce qui correspond à environ la moitié du coût de la décomp. LU ($\sim 2/3\, n^3$)

Rem :

- Pas de recherche de Pivot
- Cet algorithme peut-être adapté pour vérifier si une matrice symétrique est ou non s.d.p.

(On évite de calculer λ_{min}, coûteux en général en grande dimension)

Il suffit de tester au cours de l'algorithme que l'on a bien :

$$a_{kk} - L_{k,1:k-1} L_{k,1:k-1}^T > 0$$

sinon, la matrice n'est pas s.d.p

- La matrice L préserve la structure "par bande" de la matrice A

Ex : Si A est s.d.p tri-diag, alors L sera bi-diag et coût $O(n)$

- Lien entre décomp. de Cholesky et LU : $A = LL^T = \underbrace{L D^{-1}}_{L} \underbrace{D L^T}_{U}$

avec $D = \mathrm{diag}(\dots, 1/L_{ii}, \dots)$

le 20.12.18

Thm: Soit A s.d.p. Alors l'élimination de Gauss (symétrique) sans recherche de Pivot est "backward" stable.

---

le 21.02.19

Ch. 6 Méthode des moindres carrés :

On considère un système linéaire avec plus d'équations que d'inconnues (système surdéterminé)

Autrement dit, on considère le système : $A\underline{x} = \underline{b}$

avec $A \in \mathbb{R}^{m \times n}$ où $m \geq n$ (+ lignes que colonnes)

Ce système ne possède en général pas de solutions, car $b \notin \text{Im}(A)$ en général.

On considère le nouveau problème suivant, appelé méthode des moindres carrés :

$$\text{Trouver } \underline{x}^* \in \mathbb{R}^n \text{ tq } \|A\underline{x}^* - \underline{b}\|_2 = \min_{\underline{x} \in \mathbb{R}^n} \|A\underline{x} - \underline{b}\|_2$$

Rem: On minimise la somme des carrés des composantes du résidu

$$\underline{r} = A\underline{x} - \underline{b} \qquad \|\underline{r}\|_2^2 = \sum_{i=1}^{n} r_i^2$$

6.1 Les équations normales :

Thm: Soit $A \in \mathbb{R}^{m \times n}$ et $\underline{b} \in \mathbb{R}^m$ où $m \geq n$. Un vecteur $x^* \in \mathbb{R}^n$ minimise $\|A\underline{x} - \underline{b}\|_2 \iff$ il satisfait les équations normales : $A^T A\,\underline{x} = A^T \underline{b}$

La solution x^* est unique $\iff$ rang $A = n$

sinon, il y a un nbr ∞ de solution.

Rem: $A^T A \in \mathbb{R}^{n \times n}$ carrée et symétrique et est semi-définie positive

Preuve :

$\Rightarrow$: On suppose que x^* minimise $\|A\underline{x} - \underline{b}\|_2^2 = f(\underline{x})$

On calcul le gradient $\nabla f(\underline{x})$

$f(\underline{x}) = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j - b_i \right)^2$

$\partial f / \partial x_k \,(\underline{x}) = \sum_{i=1}^{m} \partial / \partial x_k \left(\sum_{j=1}^{n} a_{ij} x_j - b_i \right)^2$

$= \sum_{i=1}^{m} 2 \left(\sum_{j=1}^{n} a_{ij} x_j - b_j \right) a_{ik}$

$= 2 \sum_{i=1}^{m} a_{ik} \cdot r_i$ où $\underline{r} = A\underline{x} - \underline{b}$

Ainsi $\nabla f(x) = 2A^T r = 2A^T(A\underline{x} - b)$

Comme x^* minimise $f(\underline{x})$, $\nabla f(x^*) = 0$

et $2A^T(Ax^* - \underline{b}) = 0$ qui vérifie les équations normales.

$\Leftarrow$: $\underline{y} = \underline{x}^* + \underline{h}$ à montrer : $f(\underline{y}) \geq f(x^*)$

$f(\underline{y}) = f(x^* + \underline{h}) = (A(x^* + \underline{h}) - b)^T (A(x^* + \underline{h}) - b)$

$= f(x^*) + (A\underline{h})^T (Ax^* - b) + (Ax^* - b)^T (A\underline{h})$

$+ (A\underline{h})^T (A\underline{h})$

$= f(x^*) + \overbrace{2\underline{h}^T A^T (Ax^* - \underline{b})}^{0} + \|A\underline{h}\|_2^2$ ✻

Ainsi $f(\underline{y}) - f(x^*) = \|A\underline{h}\|_2^2 \geq 0$ pourtant $\underline{y} = x^* + \underline{h}$

et donc x^* est bien un minimiseur de $f(\underline{x})$

Rem : En utilisant la définition de $f'(x^*)$, on déduit de ✻,

$f'(x^*)\underline{h} = 2\underline{h}^T A^T (Ax^* - b) = \nabla f(x^*)^T \underline{h} = \underline{h}^T \nabla f(x^*)$

On retrouve, $\nabla f(x^*) = 2A^T(Ax^* - b)$

Rem : x^* solution des équations normales ($A^T A\underline{x} = A^T \underline{b}$)

$\Leftrightarrow \forall \underline{h} \in \mathbb{R}^n,\ \underline{h}^T A^T (A\underline{x}^* - \underline{b}) = 0$

$\Leftrightarrow \forall \underline{h} \in \mathbb{R}^n,\ (A\underline{h})^T (A\underline{x}^* - \underline{b}) = 0$

$\Leftrightarrow \underline{r} = A\underline{x}^* - \underline{b} \in (\mathrm{Im}\, A)^{\perp}$ ← Produit scalaire de $A\underline{h}$ et $A\underline{x}^* - \underline{b}$

d'où le nom " équations normales "