Formulaire SysRF

Antonin Kenzi 26 décembre 2023

Polynôme d'interpolation : 1

Forme de lagrange :

$$P_n(x) = \sum_{i=0}^n y_i l_i(x) \qquad l_i = \prod_{j=0, j\neq i}^n \frac{x - x_j}{x - x_i}$$

$$L(x) = \prod_{i=0}^n (x - x_i) \qquad \lambda_i^{-1} = \prod_{j=0, j\neq i}^n (x_i - x_j)$$

Evaluation :

$$P_n = L(x) \sum_{i=0}^n \frac{\lambda_i}{x - x_i} y_i$$

Formule barycentrique :

$$P_n = \frac{\sum_{i=0}^n \frac{\lambda_i}{x - x_i} y_i}{\sum_{i=0}^n \frac{\lambda_i}{x - x_i}}$$

forme de Newton :

$$P_n(x) = c_0 + c_n \prod_{i=0}^{n-1} (x - x_i)$$

$$f[x_0] = y_0 = c_0 \qquad f[x_n] = y_n$$

$$f[x_0, x_1] = \frac{y_1 - y_0}{x_1 - x_0}$$

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$$

$$f[x_0, x_1, ..., x_n] = \frac{f[x_1, x_2, ..., x_n] - f[x_0, x_1, ..., x_{n-1}]}{x_n - x_0}$$

Schéma de newtown :

$$\begin{tikzcd}
x_0 & f[x_0] \arrow[dr, "-"] & & \\
& & f[x_0, x_1] \arrow[dr, "-"] & \\
x_1 & f[x_1] \arrow[ur, "+"] \arrow[dr, "-"] & & f[x_0, x_1, x_2] \\
& & f[x_1, x_2] \arrow[ur, "+"] & \\
x_2 & f[x_2] \arrow[ur, "+"] & &
\end{tikzcd}$$

Erreur d'interpolation :

$$|f(x) - P_n(x)| \leq \frac{M_{n+1}}{(n+1)!} \left|\prod_{i=0}^n (x - x_i)\right|$$

$$M_{n+1} := max_{\xi \in [a,b]} \left|f^{(n+1)}(\xi)\right|$$

Splines : 2

Calcul d'une spline cubique :

$$S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i$$

$$a_i = \frac{1}{6h}(y''_{i+1} - y''_i) \qquad b_i = \frac{1}{2}(y''_i)$$

$$c_i = \frac{y_{i+1} - y_i}{h} - \frac{h}{6}(y''_{i+1} + 2y''_i) \qquad d_i = y_i$$

Calcul des dérivées secondes :

$$y''_{i-1} + 4y''_i + y''_{i+1} = \frac{6}{h^2}(y_{i+1} - 2y_i + y_{i-1})$$

avec $i = 1, ..., n-1$

Matrice tridiagonale :

$$\begin{bmatrix} 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} y''_1 \\ y''_2 \\ y''_3 \\ y''_4 \end{bmatrix} = \frac{6}{h^2} \begin{bmatrix} y_2 - 2y_1 + y_0 \\ y_3 - 2y_2 + y_1 \\ y_4 - 2y_3 + y_2 \\ y_5 - 2y_4 + y_3 \end{bmatrix} - \begin{bmatrix} y''_0 \\ 0 \\ 0 \\ y''_5 \end{bmatrix}$$

Sur la calculatrice :

- Matrice tridiagonale sto->(ctrl + var) -> A
- menu -> Matrice et vecteur(7) -> simultané(6)
- A,vecteur colonne de droite -> :=

Intégration : 3

Quadrature interpolation :

formule interpolatoire :

$$Q_n = \int_a^b P_n(x)dx = \sum_{i=0}^n f(x_k) \int_a^b L_k(x)dx = \sum_{i=0}^n f(x_k) w_k$$

avec :

$$w_k = \int_a^b L_k(x)dx = \int_a^b \prod_{j=0, j\neq k}^n \frac{x - x_j}{x_k - x_j} dx$$

- Q_n est une formule d'intégration de degré au moins n
- x_k sont les points de quadrature
- w_k sont les poids de quadrature

$$E[f] = Q - I = \sum_{i=0}^n f(x_k) w_k - \int_a^b f(x)dx$$

Formule de Newton-Cotes : 4

Les formules interpolatoires polynomiales avec noeuds équidistants sont appelées des formules de Newton-Cotes

Fermées (car a et b en font partie) :

$$x = a + (b - a)t \quad t \in [0, 1] \Leftrightarrow dx = (b - a)dt$$

$$w_k = (b - a) \int_0^1 \prod_{j=0, j\neq k}^n \frac{nt - j}{k - j} dt$$

$$f_0 = f(a)$$

Trapezes (degré de precision 1) :

$Q_1 = \frac{h}{2}(f_0 + f_1) \quad h = b - a$

Simpson (degré de precision 3) :

$Q_2 = \frac{h}{3}(f_0 + 4f_1 + f_2) \quad h = \dfrac{b-a}{2}$

3/8 de Newtown (degré de precision 3) :

$Q_3 = \frac{3h}{8}(f_0 + 3f_1 + 3f_2 + f_3) \quad h = \dfrac{b-a}{3} \quad f_0 = f(a)$

Boole (degré de precision 5) :

$Q_4 = \frac{2h}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) \quad h = \dfrac{b-a}{4}$

degré de precision :

- Prendre nombre de point impaire.
- subdiviser l'intervalle en sous intervalle de taille égale.

Regles des trapezes composées :

$$T(h) = h\left[\frac{1}{2}f(a) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}f(b)\right]$$

avec

$$h = \frac{b-a}{n}$$

Regle du point milieu

$$M(h) = h\sum_{i=1}^{n} f(x_{k+\frac{1}{2}}) \quad x_{k+\frac{1}{2}} = a + (k + \frac{1}{2})h$$

Lien avec la regle des trapezes :

$$T\left(\frac{h}{2}\right) = \frac{1}{2}\left[T(h) + M(h)\right]$$

exemple :

- $T(h) = h/2(f_0 + f_4)$
- M(h) = $h(f_2)$
- $T(h/2) = 1/2(T(h) + M(h))$
- $M(h/2) = h/2(f_1 + f_3)$
- $T(h/4) = 1/2(T(h/2) + M(h/2))$

Erreur (trapezes composées) : convergence lente sauf cas optimal :

- f(x) est périodique
- f(x) infiniement dérivable
- Intergrale sur une période

$$\left|\int_a^b f(x)dx - T(h)\right| \leq \frac{h^2(b-a)}{12} max_{x\in[a,b]} \left|f''(x)\right|$$

Regles des Simpson composées :

Sous 4 interval de taille h = (b-a)/4

$$S_c = \frac{h}{3}\left[f_0 + 4f_1 + 2f_2 + 4f_3 + f_4\right]$$

Paire : 2n sous intervalle de taille h = (b-a)/2n

$$S_c = \frac{h}{3}\left[f(a) + 4f(x_1) + f(b) + 2\sum_{i=1}^{n-1} f(x_{2i}) + 2f(x_{2i+1})\right]$$

Erreur (Simpson composées) :

$$\left|\int_a^b f(x)dx - S_c\right| \leq \frac{h^4(b-a)}{180} max_{x\in[a,b]} \left|f^{(4)}(x)\right|$$

Intégration de Romberg 5

Principe :

$$T_{i,0} = T(h_i) \quad h_i = \frac{b-a}{2^i}$$

$$T_{i,j} = \frac{4^j T_{i,j-1} - T_{i-1,j-1}}{4^j - 1}$$

$$\begin{tikzcd}
T_{0,0} \arrow[dr, "-"] & & & \\
T_{1,0} \arrow[r, "+4^1"'] & T_{1,1} \arrow[dr, "-"] & & \\
T_{2,0} & T_{2,1} \arrow[r, "+4^2"'] & T_{2,2} \arrow[dr, "-"] & \\
T_{3,0} & T_{3,1} & T_{3,2} \arrow[r, "+4^3"'] & T_{3,3}
\end{tikzcd}$$

Attention :

Si fonction a une singularité, amélioration de la précision pas significative. Si fonction est périodique et infiniement dérivable, Trapezes composées est optimal.

Intégration de Gauss 6

$$\int_a^b f(x)dx = \sum_{i=1}^{n} w_i f(x_i)$$

$$w_i = \int_a^b \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} dx = \int_a^b \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} dx$$

Polynomes de Legendre :

$$P_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}\left[(x^2-1)^n\right]$$

$$P_0(x) = 1 \quad P_1(x) = x \quad P_2(x) = \frac{1}{2}(3x^2 - 1)$$

$$P_3(x) = \frac{1}{2}(5x^3 - 3x) \quad P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3)$$

Equations non linéaires : 7

Méthode de la bissection :

Principe :

- 2 point a et b tels que :
  - f(x) continue sur [a,b]
  - f(a)f(b) < 0
  - f(x) contient au moins un zéro sur [a,b]

Algorithme :

1. $x_1 = \frac{a+b}{2}$ et $f(x_1)$
2. Si $f(x_1) = 0$ alors x_1 est la solution
3. Si $f(a)f(x_1) < 0$ alors $x_2 = \frac{a+x_1}{2}$ et $f(x_2)$
4. sinon a = x1
5. on recommence 1

Borne à priori : $|x_k - r| \leq \frac{b-a}{2^k} \quad r => f(r) = 0$

Avantage et inconvénient :

- Avantage :
  - Convergence garantie
  - Facile à mettre en oeuvre
- Inconvénient :
  - Convergence lente

Méthode de la sécante :

Principe : On remplace f (x) par une fonction plus simple et itère

algorithme :

1. Choisir x_0 et x_1
2. Calculer f(x_0) et f(x_1)
3. Remplacer f(x) par la droite passant par les points (x_0, f(x_0)) et (x_1, f(x_1))
4. determiner le zero de la droite
5. $x_0 = x_1$ et $x_1 = x_2$
6. on recommence 2

formule :

$$x_{k+1} = x_{k-1} - f(x_{k-1})\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$$

Avantage et inconvénient :

- Avantage :
  - Convergence superlinéaire $p = \frac{1+\sqrt{5}}{2}$
- Inconvénient :
  - Convergence non garantie

Méthode de Newton :

Principe : On remplace f (x) par la tangente en x_k

Hypotheses : f(x) est continument dérivable

formule :

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Avantage et inconvénient :

- Avantage :
  - Convergence quadratique
  - Facile à mettre en oeuvre
  - un seul point de départ
  - Généralisable à plusieurs variables
- Inconvénient :
  - Convergence non garantie
  - Nécessite le calcul de la dérivée et la diférentabilité de f
  - deux evaluations de f par itération

Ordre de convergence :

$$F'(x_k) = 1 - \frac{f'(x)^2 - f''(x)f(x)}{f'(x)^2} = \frac{f''(x)f(x)}{f'(x)^2}$$

$$\lim_{x \to r} F'(x) = \frac{f''(r)f(r)}{f'(r)^2}$$

$$e_{k+1} = \frac{f''(r)f(r)}{2f'(r)^2}e_k^2$$

avec

$$\frac{F''(r)}{2} = \frac{f''(r)}{2f'(r)}$$

Equations non linéaires à plusieurs variables : 8

Principe :

on cherche r tel que F(r) = r :

Iteration du point fixe :

Algorithme :

1. Choisir x_0
2. $x_{k+1} = G(x_k)$
3. on recommence 1

$$||x_k - r|| \leq \frac{L^{k-l}}{1-L}||x_{l+1} - x_l|| \quad 0 < l < 1$$

Estimation à priori :

$$||x_k - r|| \leq \frac{L^k}{1-L}||x_1 - x_0|| \quad l = 0$$

Estimation à posteriori :

$$||x_k - r|| \leq \frac{L}{1-L}||x_k - x_{k-1}|| \quad l = k-1$$

$$L := max_{x\in[a,b]}||J(x)||$$

Jacobienne de F :

$$J(x) = \begin{pmatrix} \frac{\partial F_1}{\partial x_1} & \cdots & \frac{\partial F_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n}{\partial x_1} & \cdots & \frac{\partial F_n}{\partial x_n} \end{pmatrix}$$

Norme de $||J(x)||$:
sous la forme de la norme de frobenius

$$||J(x)|| = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}\left|\frac{\partial F_i}{\partial x_j}\right|^2}$$

Avantage et inconvénient :

- Avantage :
  - Résultat global
- Inconvénient :
  - Difficile de trouver le Domaine

Ordre de convergence :

$$||e_{k+1}|| \leq ||J(r)||||e_k||$$

Generalement linéaire
Convertion quadratique si $||J(r)|| = 0$
Méthode de Newton :
Principe :
On considère le système d'équation non linéaire :

$$f(x, y) = 0 \quad g(x, y) = 0$$

Algorithme :

1. Choisir $(x_0,\ y_0)$ proche de (r,s)
2. remplace $(x_0, y_0, f(x_0, y_0))$ par le plan tangent
$$T_f(x, y) = f(x_0, y_0) + (x - x_0) f_x(x_0, y_0) + (y - y_0) f_y(x_0, y_0)$$
$$T_g(x, y) = g(x_0, y_0) + (x - x_0) g_x(x_0, y_0) + (y - y_0) g_y(x_0, y_0)$$
3. determiner l'intersection des plans avec le plan Oxy (cela donne une droite)
$$T_f(x, y) = 0 \quad T_g(x, y) = 0$$
4. determiner le point d'intersection des droites
5. intersection des droites : $(x_1,\ y_1)$ nouvelle approximation de (r,s)
6. on recommence 2 avec $(x_1,\ y_1)$

la méthode de Newton(-Raphson)

$$\begin{aligned} x_{k+1} &= x_k - \xi_k \\ &= x_k - \frac{g(x_k, y_k) f_y(x_k, y_k) - f(x_k, y_k) g_y(x_k, y_k)}{f_x(x_k, y_k) g_y(x_k, y_k) - f_y(x_k, y_k) g_x(x_k, y_k)} \\ y_{k+1} &= y_k - \eta_k \\ &= y_k - \frac{f(x_k, y_k) g_x(x_k, y_k) - g(x_k, y_k) f_x(x_k, y_k)}{f_x(x_k, y_k) g_y(x_k, y_k) - f_y(x_k, y_k) g_x(x_k, y_k)} \end{aligned}$$

r = x_k et s = y_k
Ordre de convergence :
Au moins égal à 2
Généralisation :

$$F(r) = 0 \quad F = \begin{bmatrix} f_1(x_1, x_2, \ldots, x_n) \\ f_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ f_n(x_1, x_2, \ldots, x_n) \end{bmatrix}$$

algorithme :

1. Choisir $x_0 = (x_{1,0}, x_{2,0}, \ldots, x_{n,0})$
2. $x_{k+1} = x_k - J(x_k)^{-1} F(x_k)$
3. on recommence 1

Ajouter peut-être Banach :

Courbes de Bézier : **9**

Algorithme de De Casteljau :
Ordre 1 :
$P_0 = \binom{1}{1} \quad P_1 = \binom{2}{3}$
alors
$P(t) = (1-t)P_0 + tP_1 = \binom{1}{1}(1-t) + \binom{2}{3}t = \binom{1-t}{1+2t}$
Ordre 2 :
On construit d'abord deux courbes de Bézier d'ordre 1
$P_0 = \binom{1}{1} \quad P_1 = \binom{2}{3} \quad P_2 = \binom{4}{2}$

$$\begin{aligned} P(t) &= (1-t)Q_0(t) + tQ_1(t) \\ &= (1-t)((1-t)P_0 + tP_1) + t((1-t)P_1 + tP_2) \\ &= (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2 \end{aligned}$$

Ordre 3 :
On construit d'abord deux courbes de Bézier d'ordre 2

$$
\begin{aligned}
P(t) &= (1-t)R_0(t) + tR_1(t) \\
&= \dots \\
&= (1-t)^3P_0 + 3t(1-t)^2P_1 + 3t^2(1-t)P_2 + t^3P_3
\end{aligned}
$$

Propriétés :

- points initialement donnés = points de contrôle
- Les courbes de Bézier passent par le premier et le dernier point de contrôle
- Les courbes de Bézier sont contenues dans l'enveloppe convexe des points de contrôle
- Les courbes de Bézier sont stables sous transformations affines (translation, rotation, homothétie)

Polynôme de Bernstein :

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i} \quad i = 0, 1, \dots, n$$

Relation avec les courbes de Bézier :

$$P(t) = \sum_{i=0}^{n} B_i^n(t) P_i$$

Propriétés :

$$P'(t) = n \sum_{i=0}^{n-1} (P_{i+1} - P_i) B_i^{n-1}(t)$$

en t = 0 et t = 1 :

$$P'(0) = n(P_1 - P_0) \quad P'(1) = n(P_n - P_{n-1})$$

la direction de la tangente en P_0 est identique à celle de la droite P_0P_1 et la direction de la tangente en P_n est identique à celle de la droite $P_{n-1}P_n$

Résolition des systèmes linéaires 10

resoudre $A\vec{x} = \vec{b}$ pour trouver $\vec{x} \in \mathbb{R}^n$.

hypotheses

- A est une matrice $n \times n$.
- A est inversible.

methodes

- **méthode directe** : A^{-1} est calculé explicitement.
  Exemple : Elimination de Gauss, LU, LL^t, QR
- **méthode iterative** : $\vec{x}$ est calculé par une suite de vecteurs $\vec{x}_0, \vec{x}_1, \vec{x}_2, \dots$.

méthode iterative

Normes vecteurs

Classique :

$$||v||_p = \left(\sum_{i=1}^{n} |v_i|^p \right)^{1/p}$$

- $p = 1$: norme L_1 (norme de Manhattan)

$$||v||_1 = \sum_{i=1}^{n} |v_i|$$

- $p = 2$: norme L_2 (norme euclidienne)

$$||v||_2 = \sqrt{\sum_{i=1}^{n} |v_i|^2}$$

- $p = \infty$: norme L_∞ (norme du maximum)

$$||v||_\infty = \max_{1 \le i \le n} |v_i|$$

Normes matricielles

$$||A\vec{v}||_p \le ||A||_p \, ||\vec{v}||_p \quad \forall \vec{v} \in \mathbb{R}^n \iff ||A||_p = \max_{\vec{v} \neq 0} \frac{||A\vec{v}||_p}{||\vec{v}||_p}$$

- $p = 1$: norme L_1

$$||A||_1 = \max_j \sum_{i=1}^{n} |a_{ij}|$$

- $p = 2$: norme L_2

$$||A||_2 = max|\lambda(A^TA)|$$

- $p = \infty$: norme L_∞

$$||A||_\infty = \left|\left|A^T\right|\right|_1 = \max_i \sum_{j=1}^{n} |a_{ij}|$$

Lien entre $||\vec{x} - \vec{x}_k||$ et $||\vec{b} - A\vec{x}_k||$

$$\underbrace{\frac{||\vec{x} - \vec{x}_k||_p}{||\vec{x}||_p}}_{\text{erreur relative}} \le \underbrace{||A||_p ||A^{-1}||_p}_{\kappa_p(A)} \underbrace{\frac{||\vec{b} - A\vec{x}_k||_p}{||\vec{b}||_p}}_{\text{résidu relatif}}$$

$\kappa_p(A)$ est le conditionnement de A. Si $\kappa_p(A)$ est grand, alors A est mal conditionné.

Methode directe 11

Elimination de Gauss

1. Transformer la matrice A en forme echelonnee.

2. Substituer les valeurs trouvees dans les equations.

Factorisation LU

1. Transformer la matrice A en forme echelonnee. $A = LU$
2. Substituer en avant $L\vec{y} = \vec{b}$. avec L la matrice triangulaire inferieure.
3. Substituer en arriere $U\vec{x} = \vec{y}$. avec U la matrice triangulaire superieure.

voir slide 9.

méthode de Doolittle

$$u_{km} = a_{km} - \sum_{j=1}^{k-1} l_{kj} u_{jm} \quad m = k, k+1, \ldots, n$$

$$l_{ik} = \frac{1}{u_{kk}} \left(a_{ik} - \sum_{j=1}^{k-1} l_{ij} u_{jk} \right) \quad m = k+1, k+2, \ldots, n$$

Besoin de pivotement

Elim. de Gauss/LU sans pivotement peut echouer si $a_{kk} = 0$.

Permuter les lignes de A pour avoir $a_{kk} \neq 0$. Mettre le plus grand element en valeur absolue en haut.

$$LU = PA$$

P est une matrice de permutation.

Initialiser P comme la matrice identite I.

Si A subit une permutation, P doit subir la meme permutation.

Transformations de Householder

— Choisir H_1 pour que $H_1 A$ ait des zeros dans la premiere colonne.

$$A_1 = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{bmatrix} = H_1 A$$

— Choisir H_2 pour que $H_2 H_1 A$ ait des zeros dans la deuxieme colonne.

$$A_2 = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & H^{(3)} \end{bmatrix} H_1 A = H_2 H_1 A$$

— Choisir H_3 pour que $H_3 H_2 H_1 A$ ait des zeros dans la troisieme colonne.

$$A_3 = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & H^{(2)} \end{bmatrix} H_2 H_1 A = H_3 H_2 H_1 A$$

Algorithme :

1. $\vec{x}$ donné par la colonne j de A.
2. $$\vec{y} = \begin{bmatrix} ||\vec{x}||_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$
3. $\vec{u} = \vec{x} - \vec{y}$
4. $$H = I - 2\frac{\vec{u}\vec{u}^T}{||\vec{u}||_2^2}$$
5. $$H_1 = \begin{bmatrix} I & 0 \\ 0 & H \end{bmatrix}$$

Méthodes itératives

Principe commun :

1. Init : $x^{(0)}$ à 0
2. Itération : $x^{(k)} \leftarrow x^{(k-1)}$
3. Jusqu'à condition d'arrêt

Méthodes de Jacobi

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)} \right) \quad i = 1, \ldots, n$$

Méthodes de Gauss-Seidel

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) \quad i = 1, \ldots, n$$

Exemple :

$$\begin{bmatrix} 4 & -1 & 1 \\ -1 & 4 & -2 \\ 1 & -2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 12 \\ -1 \\ 5 \end{bmatrix}$$

Jacobi :

$$\vec{x}^{(1)} = \begin{bmatrix} 3 \\ -0.25 \\ 1.25 \end{bmatrix} \Rightarrow \vec{x}^{(2)} = \begin{bmatrix} 2.625 \\ 1.125 \\ 0.375 \end{bmatrix} \Rightarrow \vec{x}^{(3)} = \begin{bmatrix} 3.187 \\ 0.593 \\ 1.156 \end{bmatrix}$$

Gauss-Seidel :

$$\vec{x}^{(1)} = \begin{bmatrix} 3 \\ 0.50 \\ 0.75 \end{bmatrix} \Rightarrow \vec{x}^{(2)} = \begin{bmatrix} 2.9375 \\ 0.8594 \\ 0.9453 \end{bmatrix} \Rightarrow \vec{x}^{(3)} = \begin{bmatrix} 2.9785 \\ 0.9678 \\ 0.9890 \end{bmatrix}$$

Méthodes de gradient

Algorithme de la plus grande pente :

1. Init : $x^{(0)}$ dans $\mathbb{R}^n$
2. Itération :
   a) $\vec{d}_k = -\nabla G(x^{(k)}) = \vec{b} - A\vec{x}_k = \vec{r}_k$
   b) $a_k = \frac{\vec{d}_k^T \vec{r}_k}{\vec{d}_k^T A \vec{d}_k} = \frac{\vec{r}_k^T \vec{r}_k}{\vec{r}_k^T A \vec{r}_k}$
   c) $x^{(k+1)} = x^{(k)} + a_k \vec{d}_k$

Méthodes de gradient conjugué

1. Init : $x^{(0)}$ dans $\mathbb{R}^n$
2. initialiser $\vec{r}_0 = b - Ax^{(0)}$ et $\vec{d}_0 = \vec{r}_0$
3. Itération :
   a) $\vec{d}_k = -\nabla G(x^{(k)}) = \vec{b} - A\vec{x}_k = \vec{r}_k$
   b) $a_k = \frac{\vec{d}_k^T \vec{r}_k}{\vec{d}_k^T A \vec{d}_k}$
   c) $x^{(k+1)} = x^{(k)} + a_k \vec{d}_k$
   d) $\vec{r}_{k+1} = \vec{r}_k + a_k A \vec{d}_k$
   e) $\beta_{k+1} = \frac{\vec{r}_{k+1}^T \vec{r}_{k+1}}{\vec{r}_k^T \vec{r}_k}$
   f) $\vec{d}_{k+1} = -\vec{r}_{k+1} + \beta_{k+1} \vec{d}_k$

Resolition numerique des EDOs

modéle exponentiel **12**

$$\frac{d\vec{x}}{dt} = \lambda \vec{x}$$
$$\vec{x}(0) = \vec{x}_0$$

modéle logistique **13**

$$\frac{dx}{dt} = kx(t)(M - x(t))$$
$$x(0) = x_0$$

modéle de Lotka-Volterra **14**

$$\frac{dx}{dt} = \alpha x(t) - \beta x(t) y(t)$$
$$\frac{dy}{dt} = \gamma x(t) y(t) - \delta y(t)$$
$$x(0) = x_0$$
$$y(0) = y_0$$

Problème de Cauchy **15**

$$\frac{d\vec{u}}{dt} = \vec{f}(t, \vec{u}(t))$$
$$\vec{u}(t_0) = \vec{u}_0$$

Si ED n'est pas sous la forme de Cauchy, on peut faire une reduction de l'ordre.
Transformation de l'EDO d'ordre $n \leq 1$

$$\frac{d^n y}{dt^n} = y^{(n)} = f(t, y, y', y'', \ldots, y^{(n-1)} = f(t, y_1, y_2, \ldots, y_{n(t)})$$

$$\frac{d\vec{u}(t)}{dt} = \begin{bmatrix} y1(t) \\ y2(t) \\ \vdots \\ y_{n-1}(t) \\ y_n(t) \end{bmatrix} = \begin{bmatrix} y2(t) \\ y3(t) \\ \vdots \\ y_n(t) \\ f(t, y_1, y_2, \ldots, y_{n(t)}) \end{bmatrix} = \vec{f}(t, \vec{u}(t))$$

+ condition initiale $\vec{u}(t_0) = \vec{u}_0 = \begin{bmatrix} y_1(t_0) \\ y_2(t_0) \\ \vdots \\ y_n(t_0) \end{bmatrix}$

Méthode d'Euler explicite **16**

$$\frac{du}{dt} = f(t, u(t)) \quad u(t_0) = u_0$$

Discrétisation de l'intervalle $[t_0, t_0 + T]$ en N sous-intervalles de longueur $h = \frac{T}{N}$

$$t_{k+1} = t_k + h \quad k = 0, 1, \ldots, N-1$$

$$u_{k+1} = u_k + hf(t_k, u_k) \quad k = 0, 1, \ldots, N-1$$

Méthode d'Euler modifiée (Heun) **17**

$$\vec{s}_1 = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s}_2 = \vec{f}(t_k + h, \vec{u}_k + h\vec{s}_1)$$
$$\vec{u}_{k+1} = \vec{u}_k + \frac{h}{2}(\vec{s}_1 + \vec{s}_2)$$

Méthode de point milieu **18**

$$\vec{s_1} = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s_2} = \vec{f}(t_k + \frac{h}{2}, \vec{u}_k + \frac{h}{2}\vec{s_1})$$
$$\vec{u}_{k+1} = \vec{u}_k + h\vec{s_2}$$

Méthode de optimale **19**

$$\vec{s_1} = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s_2} = \vec{f}(t_k + \frac{h}{2}, \vec{u}_k + \frac{h}{2}\vec{s_1})$$
$$\vec{u}_{k+1} = \vec{u}_k + h(\frac{1}{4}\vec{s_1} + \frac{3}{4}\vec{s_2})$$

Méthode de Runge-Kutta à 2 étages **20**

$$\vec{s_1} = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s_2} = \vec{f}(t_k + c_2 h, \vec{u}_k + a_{21} h\vec{s_1})$$
$$\vec{u}_{k+1} = \vec{u}_k + h(w_1\vec{s_1} + w_2\vec{s_2})$$

$$w_1 = 1 - \frac{1}{2c_2} \quad w_2 = \frac{1}{2c_2}$$

Tableau de Butcher **21**

$$\begin{array}{c|cc} 0 & & \\ c_2 & a_{21} & \\ \hline & w_1 & w_2 \end{array}$$

Méthode de Runge-Kutta à 3 étages **22**

$$\vec{s_1} = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s_2} = \vec{f}(t_k + c_2 h, \vec{u}_k + a_{21} h\vec{s_1})$$
$$\vec{s_3} = \vec{f}(t_k + c_3 h, \vec{u}_k + a_{31} h\vec{s_1} + a_{32} h\vec{s_2})$$
$$\vec{u}_{k+1} = \vec{u}_k + h(w_1\vec{s_1} + w_2\vec{s_2} + w_3\vec{s_3})$$

$$w_1 = 1 - \frac{1}{2c_2} - \frac{1}{2c_3} \quad w_2 = \frac{1}{2c_2} \quad w_3 = \frac{1}{2c_3}$$

$$\begin{array}{c|ccc} 0 & & & \\ c_2 & a_{21} & & \\ c_3 & a_{31} & a_{32} & \\ \hline & w_1 & w_2 & w_3 \end{array}$$

Méthode de Runge-Kutta à 4 étages **23**

$$\vec{s_1} = \vec{f}(t_k, \vec{u}_k)$$
$$\vec{s_2} = \vec{f}(t_k + c_2 h, \vec{u}_k + a_{21} h\vec{s_1})$$
$$\vec{s_3} = \vec{f}(t_k + c_3 h, \vec{u}_k + a_{31} h\vec{s_1} + a_{32} h\vec{s_2})$$
$$\vec{s_4} = \vec{f}(t_k + c_4 h, \vec{u}_k + a_{41} h\vec{s_1} + a_{42} h\vec{s_2} + a_{43} h\vec{s_3})$$
$$\vec{u}_{k+1} = \vec{u}_k + h(w_1\vec{s_1} + w_2\vec{s_2} + w_3\vec{s_3} + w_4\vec{s_4})$$

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

Estimation de l'erreur **24**

$$\vec{u}_{k+1} = \vec{u}_k + h\Phi(t_k, \vec{u}_k, h)$$

Type d'erreur :

- Globale : commis dans l'intervalle $[t_0, t_0 + T]$
  $E(T) = \vec{u}(T) - \vec{u}_n$
- locale : commis à chaque étape k
- $e_{k+1} = \vec{u}(t_{k+1}) - \vec{u}_{k+1} = \vec{u}(t_{k+1}) - \vec{u}(t_k) - h\Phi(t_k, \vec{u}_k, h)$
- arrondi : approximations numériques des calculs
  $R_{k+1} = \vec{u}_{k+1} - \vec{U}_{k+1}$
- total : $|u(T) - U_n| \leq |E(T)| + |R_n|$