Préconditionnement de systèmes linéaires

Gaëtan Hello

Université d'Evry Val d'Essonne
UFR S&T - Laboratoire de Mécanique et d'Energétique d'Evry
gaetan.hello@ufrst.univ-evry.fr

HPC CS93 - 2013

1. Notion de conditionnement
   - Contexte Eléments-Finis
   - Méthodes de résolution
   - Conditionnement

Contexte Eléments-Finis

- La méthode des éléments-finis est une méthode numérique permettant la résolution approchée d'équations aux dérivées-partielles.
- Elle consiste à transformer un problème continu initial en un problème algébrique associé pouvant être résolu par des techniques automatisables/programmables.
- On se ramène ainsi à la résolution de :

$$K \cdot u = f \tag{1}$$

avec

- $K \in \mathbb{R}^{n \times n}$ matrice de rigidité symétrique et définie positive,
- $u \in \mathbb{R}^n$ vecteur des inconnues nodales,
- $f \in \mathbb{R}^n$ vecteur des efforts nodaux.

Méthodes de résolution

Pour résoudre un système linéaire, il existe deux grandes familles de méthodes :

- les méthodes directes (nombre fini d'opérations) :
  - pivot de Gauss,
  - décomposition LU : $A = LU$ (L lower triangular, U upper triangular),
  - décomposition QR : $A = QR$ (Q orthogonale, R upper tri.),
  - décomposition de Cholesky : $A = LDL^T$ (A p.d., L lower tri., D diag.),
  - ...
- les méthodes itératives (nombre d'opérations lié à la convergence) :
  - Gauss-Seidel,
  - SOR (Successive Over Relaxation),
  - gradient conjugué (A p.d.),
  - GMRES (Generalized Minimal Residual Method)
  - ...

Conditionnement

- Les performances des méthodes numériques de résolution des systèmes linéaires sont fortement influencées par le conditionnement du problème considéré,
- Le conditionnement d'un système linéaire $A \cdot x = b$ ne dépend que de A et est noté $\kappa(A)$,
- $\kappa(A)$ permet de majorer l'erreur relative sur la solution induite par des variations sur les données A et b,
- Pour les méthodes de résolution itératives, $\kappa(A)$ influence également le nombre d'itérations de convergence : plus $\kappa(A)$ est grand et plus généralement le nombre d'itérations nécessaires est important.

Conditionnement

- variations de A :

$$\frac{||\Delta x||}{||x + \Delta x||} \leq \kappa(A) \cdot \frac{||\Delta A||}{||A||} \tag{2}$$

- variations de b :

$$\frac{||\Delta x||}{||x||} \leq \kappa(A) \cdot \frac{||\Delta b||}{||b||} \tag{3}$$

- avec une norme matricielle devant vérifier $||.|| : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}$:

$$\begin{aligned} &||A|| \geq 0, \ ||A|| = 0 \Leftrightarrow A = 0, \\ &||\alpha \cdot A|| = |\alpha| \cdot ||A||, \ \forall \alpha \in \mathbb{R}, \\ &||A + B|| \leq ||A|| + ||B||, \\ &||A \cdot B|| \leq ||A|| \cdot ||B||. \end{aligned} \tag{4}$$

Conditionnement

normes vectorielles :

- $||v||_1 = \sum_{i=1}^{n} |v_i|$
- $||v||_2 = \sqrt{\sum_{i=1}^{n} v_i^2}$
- $||v||_\infty = \max_{1 \leq i \leq n} |v_i|$

normes matricielles :

- $||A||_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}|$
- $||A||_2 = \sqrt{\rho(A^T \cdot A)}$, avec $\rho(M)$ le rayon spectral de M (ie sa plus grande valeur propre)
- $||A||_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|$

Conditionnement

- Le conditionnement d'une matrice vérifie :

$$1 \leq \kappa(A) \leq +\infty \tag{5}$$

- Le cas $\kappa(A) = 1$ correspond à celui d'une matrice identité,
- Le cas $\kappa(A) = +\infty$ correspond à celui d'une matrice singulière (ie non inversible),
- La définition du conditionnement pour la norme n est :

$$\kappa_n(A) = ||A^{-1}||_n \cdot ||A||_n \tag{6}$$

- Dans le cas d'une matrice normale ($A^T \cdot A = A \cdot A^T$) et donc d'une matrice symétrique définie positive, on a :

$$\kappa(A) = \left| \frac{\lambda_{max}(A)}{\lambda_{min}(A)} \right| \tag{7}$$

avec $\lambda_{max}(A)$ et $\lambda_{min}(A)$ les valeurs propres max et min de A

Conditionnement

Exemple de bon conditionnement :

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \kappa(A) = 1,\ b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 1.01 & 0.01 \\ 0.01 & 1.01 \end{bmatrix} \cdot \begin{bmatrix} 0.9804 \\ 1.9804 \end{bmatrix} = \begin{bmatrix} 1.01 \\ 2.01 \end{bmatrix}$$

Une faible variation de $+0.01$ sur l'ensemble des coefficients de A et b n'induit qu'une faible variation sur la solution.

Conditionnement

Exemple de mauvais conditionnement :

$$A = \begin{bmatrix} 1 & -0.99 \\ -0.99 & 1 \end{bmatrix}, \ \kappa(A) = 199, \ b = \begin{bmatrix} -0.98 \\ 1.01 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -0.99 \\ -0.99 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -0.98 \\ 1.01 \end{bmatrix}$$

$$\begin{bmatrix} 1.01 & -0.98 \\ -0.98 & 1.01 \end{bmatrix} \cdot \begin{bmatrix} 0.333 \\ 1.333 \end{bmatrix} = \begin{bmatrix} -0.97 \\ 1.02 \end{bmatrix}$$

Une faible variation de $+0.01$ sur l'ensemble des coefficients de A et b induit une forte variation sur la solution.

2 Préconditionnement

- Principe
- Quelques précontionneurs
- Gradient-conjugué préconditionné
- Exemples numériques

Principe

- La méthode de résolution numérique du système linéaire $A \cdot x = b$ verra sa précision, sa stabilité et sa vitesse de convergence (pour les méthodes itératives) influencées par $\kappa(A)$ ainsi que la distribution des valeurs propres de A.
- A partir du système initial, on peut chercher à résoudre un système équivalent (ayant la même solution) mais qui proposera un meilleur conditionnement et/ou des valeurs propres plus agrégées.
- Par exemple, on peut chercher à résoudre le système $P^{-1} \cdot A \cdot x = P^{-1} \cdot b$ où l'on a :

$$\kappa(P^{-1} \cdot A) < \kappa(A) \tag{8}$$

- L'opérateur P constitue un préconditionneur.

Principe

- Pour les méthodes itératives, l'usage d'un précontionneur augmente le coût de calcul par itération (opérations+mémoire).
- La réduction du nombre d'itérations de convergence pour le système préconditionné tend au final à réduire le coût de calcul global.
- Pour des méthodes de résolution n'exigeant pas une matrice symétrique, il est possible d'utiliser un préconditionnement à gauche :

$$P^{-1} \cdot A \cdot x = P^{-1} \cdot b \tag{9}$$

- Pour les méthodes où la matrice doit être symétrique (gradient-conjugué), il faut alors utiliser un préconditionnement à gauche et à droite pour garantir la symétrie de la nouvelle matrice :

$$\begin{aligned} (P^{-1} \cdot A \cdot P^{-1}) \cdot (P \cdot x) &= (P^{-1} \cdot b) \\ \tilde{A} \cdot \tilde{x} &= \tilde{b} \end{aligned} \tag{10}$$

avec P symétrique définie positive.

Préconditionneurs triviaux

- $P = I$: on résoud alors $I^{-1} \cdot A \cdot x = I^{-1} \cdot b$ qui est directement équivalent au système de départ. Ce préconditionnement est certes le moins onéreux de tous, mais également le moins efficace.
- $P = A$: on résoud alors $A^{-1} \cdot A \cdot x = A^{-1} \cdot b$ où l'expression de x est donnée par un simple produit matrice $\times$ vecteur. Il s'agit du préconditionneur le plus efficace ($\kappa(P^{-1} \cdot A) = 1$) mais également du plus onéreux car l'obtention de A^{-1} réclame autant d'opérations qu'une résolution directe du système initial ($o(n^3)$).
- En pratique, on cherchera un préconditionneur avec le meilleur rapport efficacité/coût.

Préconditionneurs pratiques

On partitionne A selon :

- $A = L + D + U$, si A est non symétrique
- $A = L + D + L^T$, si A est symétrique

avec L matrice strictement inférieure, D matrice diagonale et U matrice strictement supérieure.

- Jacobi :
  - $P = D$, efficace pour les matrices à diagonale dominante,
- Gauss-Seidel :
  - $P = D + L$, (non-symétrique)
  - $P = (D + L)D^{-1}(D + L^T)$, (symétrique)
- Symmetric Successive Over-Relaxation (SSOR)
  - $P = (\frac{D}{\omega} + L)\frac{\omega}{2-\omega}D^{-1}(\frac{D}{\omega} + L^T)$, avec $\omega \in]0, 2[$

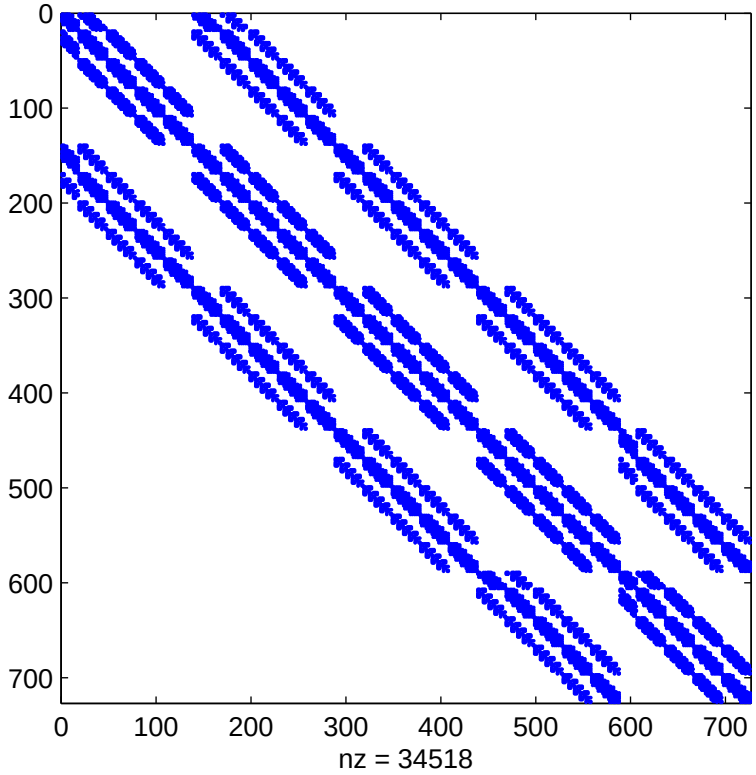
Gradient-conjugué préconditionné

Algorithme du gradient-conjugué préconditionné [Golub-1996]

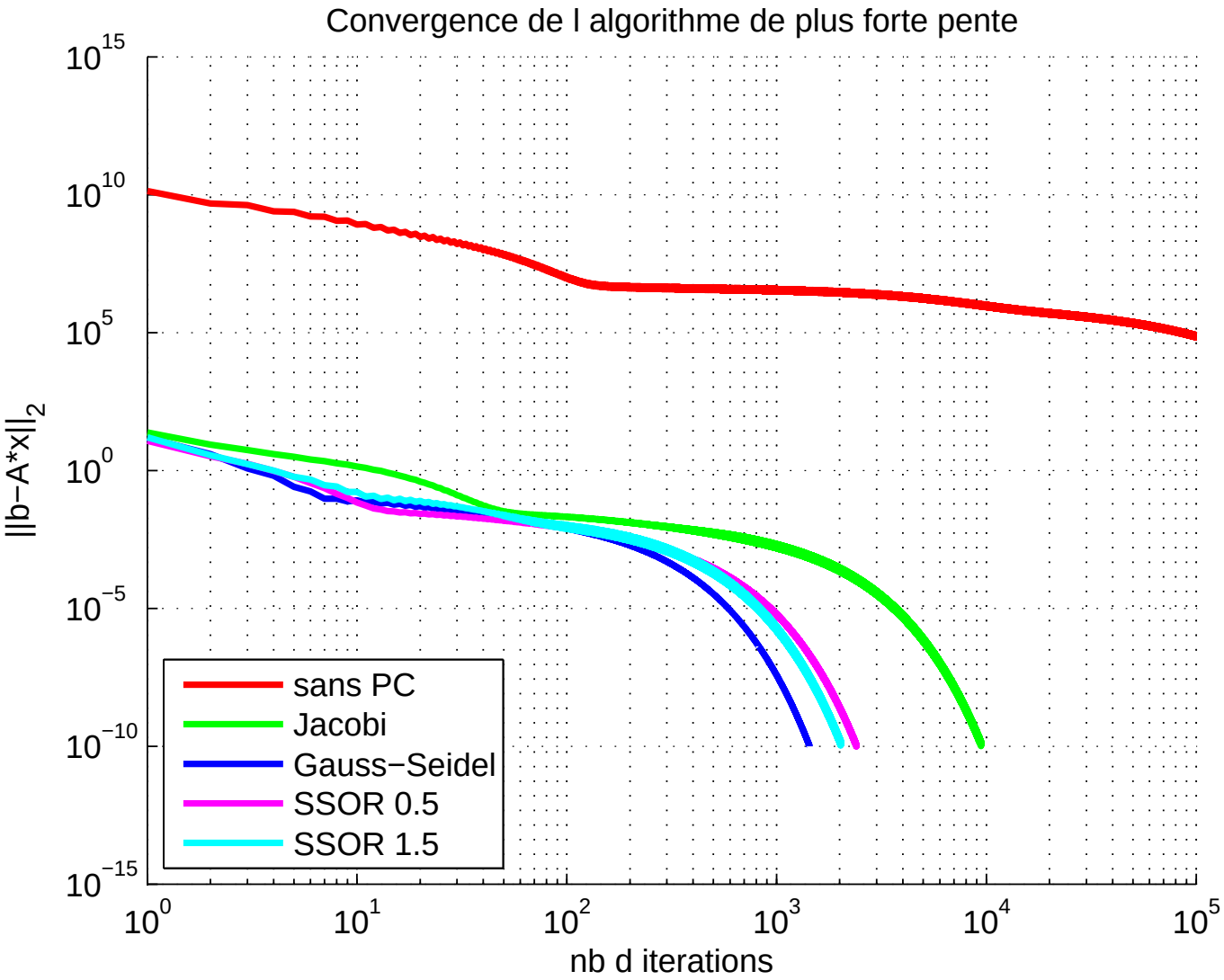
x_0=initial guess
$r_0 = b - A \cdot x_0$
$k = 0$
while $\|r_k\|_2 > \varepsilon$ **do**
  solve $P \cdot z_k = r_k$
  $k = k + 1$
  if k=1 **then**
    $p_1 = z_0$
  else
    $\beta_k = \frac{r_{k-1}^T \cdot z_{k-1}}{r_{k-2}^T \cdot z_{k-2}}$
    $p_k = z_{k-1} + \beta_k \cdot p_{k-1}$
  end if
  $\alpha_k = \frac{r_{k-1}^T \cdot z_{k-1}}{p_k^T \cdot A \cdot p_k}$
  $x_k = x_{k-1} + \alpha_k \cdot p_k$
  $r_k = r_{k-1} - \alpha_k \cdot A \cdot p_k$
end while

Exemple 1

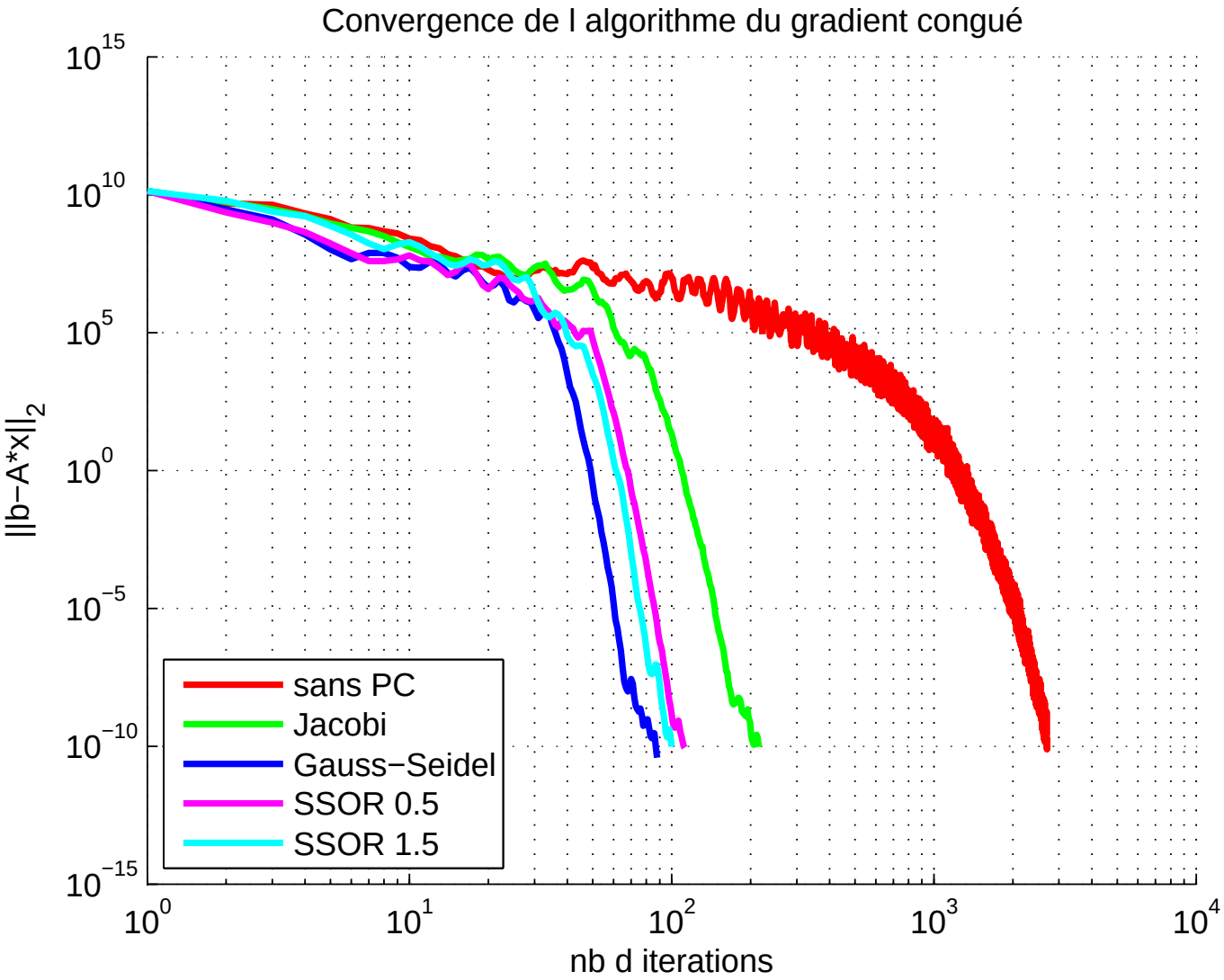
Matrice *msc00726*
(http ://www.cise.ufl.edu/research/sparse/matrices/index.html)



Exemple 1


Convergence de l algorithme de plus forte pente
10^{15}
10^{10}
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{-15}
$||b-A*x||_2$
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
10^{5}
nb d iterations
sans PC
Jacobi
Gauss−Seidel
SSOR 0.5
SSOR 1.5

Exemple 1


Convergence de l algorithme du gradient congué
10^{15}
10^{10}
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{-15}
$||b-A*x||_2$
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
nb d iterations
sans PC
Jacobi
Gauss−Seidel
SSOR 0.5
SSOR 1.5

Exemple 1

Précond.	I	Jacobi	Gauss-Seidel	SSOR 0.5	SSOR 1.5
κ	417817	1468	433	358	1550
n_{iter} SD	$> 10^5$	9456	1438	2400	2030
n_{iter} CG	2699	245	88	112	100