Systèmes linéaires: méthodes itératives et relaxation (SOR)

CONTENU

Le **plan** de la session est le suivant:

- Méthodes de décomposition / splitting: le principe et les questions de convergence,
- Méthodes de Jacobi et Gauss-Siedel,
- Méthode de relaxation (ou SOR),
- Méthode de relaxation dans le cas de matrices définies par blocs.

Méthodes itératives: principe

On cherche à résoudre efficacement le système linéaire

$$Ax = b$$

avec A matrice à coefficients réels, carrée $n \times n$, inversible et b vecteur de $\mathbb{R}^n$ donné.

Outre les méthodes dites directes telles que les méthodes de Gauss ou Cholesky, on peut envisager la résolution de tels systèmes par des méthodes *itératives*, tout particulièrement dans les cas où n est grand et A est *creuse.*

(A creuse signifie que A comporte très peu de coefficients non nuls, typiquement moins voire beaucoup moins que 1%, ce qui est fréquent en pratique).

Le principe de base des méthodes itératives est de construire une suite de "vecteurs solutions" $(x^{(n)})_{n\geq 0}$ telle que: $\lim_{n\to\infty} x^{(n)} = x^*$ où x^* est l'unique solution du système linéaire.

Les calculs ne pouvant être menés à l'infini le *test d'arrêt*: $\|x^{(n)} - x^*\| \leq \varepsilon$doit être imposé, mais sachant que la solution exacte x^* n'est pas connue, on peut considérer le critère de stationnarité suivant: $\frac{\|x^{(n)} - x^{(n-1)}\|}{\|x^{(n-1)}\|} \leq \varepsilon$.

Nous allons développer ici les méthodes dites de Jacobi, Gauss-Siedel, pour finalement arriver à la méthode de relaxation SOR (Successive Over Relaxation), qui sont des méthodes dites de décomposition (*splitting* en anglais).

Méthodes de décomposition: principe et convergence

Le principe est de décomposer A sous la forme $A = M - N$, ce qui donne l'équation: $Mx = Nx + b$, et de définir la méthode itérative suivante:

$$\boxed{x_0 \text{ donné }, Mx^{(n+1)} = Nx^{(n)} + b}$$

La question va être de définir la décomposition, donc M, telle que:

i) M soit relativement peu chère à "inverser" à chaque itération,

ii) l'algorithme converge (rapidement de préférence).

A noter que cette idée de splitting est relativement fréquente en analyse / méthodes numériques ("on sépare / on explicite certaines difficultés du problème").

En termes de convergence, on va s'appuyer sur le résultat suivant.

Proposition. *Soit M une matrice à coefficients réels, carrée $n \times n$. Les quatre propriétés suivantes sont équivalentes:*

i) $\lim_{n\to\infty} \|Mx\| = 0$, $\forall x \in \mathbb{R}^n$,

ii) $\lim_{n\to\infty} \|M\| = 0$,

iii) Le rayon spectral de M, $\rho(M) = \max_k |\lambda_k(A)| < 1$,

iv) $(I - M)$ *est inversible et* $(I - M)^{-1} = \sum_{k=0}^{\infty} M^k$.

Application à la décomposition $M - N$**.** Sur la base de décomposition $A = M - N$, soit l'équation $Mx = Nx + b$, en passant à la limite dans l'équation, on obtient: $(I - M^{-1}N)x^{(\infty)} = M^{-1}b$.

Une *condition nécessaire et suffisante* de convergence des méthodes itératives basée sur cette décomposition s'écrit alors: $\rho(M^{-1}N) < 1$.

Décomposition $D - E - F$

Pour chacune des méthodes à venir, la décomposition de A va se baser sur l'écriture suivante: $A = D - E - F$, avec D la diagonale de A, $(-E)$ la partie inférieure stricte et $(-F)$ la partie supérieure stricte.

$$A = \begin{bmatrix} D & -F & -F \\ -E & D & -F \\ -E & -E & D \end{bmatrix}$$

soit: $D = diag(a_{ii})$; $E_{ij} = -a_{ij}$ si $i > j, 0$ sinon; et $F_{ij} = -a_{ij}$ si $i < j, 0$ sinon.

Méthode de Jacobi

La *méthode de Jacobi* consiste à effectuer la décomposition suivante: $M = D$, $N = E + F$.
Ce qui donne les itérations:

$$Dx^{(n+1)} = (E + F)x^{(n)} + b$$

Ce qui s'écrit en termes de composantes:

$$x_i^{(n+1)} = \frac{1}{a_{ii}}(b_i - \sum_{j \neq i} a_{ij} x_j^{(n)})$$

pour $a_{ii} \neq 0$.
Cette méthode est triviale à implémenter, très peu chère à chaque itération, par contre sa convergence est très lente.

Méthode de Gauss-Siedel

En observant bien les itérés de la méthode de Jacobi, on s'aperçoit que l'on peut introduire les valeurs de x_j fraichement calculées sans surcoût de calcul. Cela revient à effectuer les itérations suivantes:

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

Ou encore cela revient à faire la décomposition suivante: $M = D - E$, $N = F$, ce qui donne:

$$(D - E)x^{(n+1)} = Fx^{(n)} + b$$

Il s'agit de la *méthode dite de Gauss-Siedel.* Cette dernière converge plus vite que Jacobi (si convergence).
Elle ne coûte pas plus cher que Jacobi; il n'y a d'ailleurs toujours pas de système linéaire à résoudre à chaque itération.
Le coût par itération est tout simplement: $\mathcal{O}(cn)$ où c est nombre moyen d'éléments non nuls par lignes.

Méthode de relaxation

Une généralisation de la précédente méthode est la méthode dite *SOR (Successive Over Relaxation).*
Son principe est d'introduire un "équilibre" de la décomposition diagonale entre la partie explicite et la partie implicite, à savoir: $M = \frac{1}{\omega} D - E$, $N = \frac{1-\omega}{\omega} D + F$.

Cela donne les itérés suivants:

$$(\frac{1}{\omega} D - E)x^{(n+1)} = (\frac{1-\omega}{\omega} D + F)x^{(n)} + b$$

ou encore en termes de composantes:

$$x_i^{(n+1)} = \frac{\omega}{a_{ii}}[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(n+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(n)}] + (1 - \omega)x_i^{(n)}$$

Notons que la méthode de Gauss-Siedel n'est rien d'autre que la méthode SOR en posant $\omega = 1$.
Le coût par itération de SOR est toujours le même à savoir $\mathcal{O}(cn)$,
c le nombre moyen d'éléments non nuls par lignes.

Exercice. Vérifier que si l'algorithme SOR converge, on obtient bien la solution recherchée.

Méthode de relaxation: convergence

Quel est l'intérêt de l'introduction de cette "relaxation" de D au travers du paramètre ω ?
Se donner des chances de tenter d'accélerer l'algorithme... Et en pratique on y arrive !
La pratqiue montre que l'on arrive à gagner jusqu'à un facteur 10.

Comment choisir ω ? Dans le cas A symétrique définie positive, une analyse mathématique donne la valeur optimale de ω; cette valeur optimale est d'ailleurs supérieure à 1 d'où la terminologie *over*-relaxation.

On peut démontrer le résultat suivant.
Théoreme de convergence. *Pour A matrice à coefficients réels, carrée $n \times n$, inversible, "quelconque", une condition nécessaire de convergence de la méthode SOR est: $\omega \in]0, 2[$.*
Dans le cas où A est symétrique définie positive, cette condition est suffisante.
Dans le cas où A est à diagonale dominante, une condition suffisante est $\omega \in]0, 1]$.

Méthode de relaxation: cas de matrices par blocs

Dans le cas où A possède une structure par blocs particulière, la méthode SOR peut naturellement s'adapter.
Considérons A de la forme suivante.

Notons qu'une telle forme se rencontre couramment, par exemple dans un contexte de résolution de modèles numériques d'équations aux dérivées partielles sur maillages réguliers (par exemple en mécanique des fluides ou structures).

La méthode de relaxation par blocs s'écrit alors:

$$A_{ii}x_i^{(n+1)} = \omega(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(n+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(n)}) + (1-\omega)A_{ii}x_i^{(n)}$$

A chaque itération n, un système linéaire de taille $m \times m$ doit donc être résolu:

$$A_{ii}x_i^{(n+1)} = c_i^n \quad 1 \leq i \leq n$$

Typiquement, on pourra calculer au préalable une factorisation LU des matrices A_{ii}.