Analyse Numérique I

Sup'Galilée, Ingénieurs MACS 1ère année & L3 MIM

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

2021/10/24

Chapitre IV

Résolution de systèmes linéaires

Plan

Plan

Méthodes itératives pour la résolution du système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

Trouver une **matrice d'itération** $\mathbb{B}$ et d'un vecteur $\boldsymbol{c}$ telles que

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c},\ \ k \geqslant 0,\ \boldsymbol{x}^{[0]} \text{ arbitraire}$$

vérifie

$$\lim_{k\to\infty} \boldsymbol{x}^{[k]} = \tilde{\boldsymbol{x}} \ \ \text{avec } \tilde{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$$

Plan

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ une matrice régulière, avec $\forall i \in [\![1, n]\!],\ \mathrm{A}_{i,i} \neq 0$

$$\mathbb{A} = \begin{pmatrix} \mathrm{A}_{1,1} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{A}_{n,n}\end{pmatrix} + \begin{pmatrix} 0 & \cdots & \cdots & 0\\ \mathrm{A}_{2,1} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathrm{A}_{n,1} & \cdots & \mathrm{A}_{n,n-1} & 0\end{pmatrix} + \begin{pmatrix} 0 & \mathrm{A}_{1,2} & \cdots & \mathrm{A}_{1,n} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \mathrm{A}_{n-1,n} \\ 0 & \cdots & \cdots & 0\end{pmatrix}$$

$$= \mathbb{D} - \mathbb{E} - \mathbb{F} = \begin{pmatrix} \ddots & & -\mathbb{F} \\ & \mathbb{D} & \\ -\mathbb{E} & & \ddots \end{pmatrix}$$

Plan

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Jacobi** :

$$b_i = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k]} + \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

ou encore

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Jacobi** :

$$b_i = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k]} + \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{b} = \mathbb{D}\boldsymbol{x}^{[k+1]} - \mathbb{E}\boldsymbol{x}^{[k]} - \mathbb{F}\boldsymbol{x}^{[k]}$$

ou encore

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{x}^{[k+1]} = \mathbb{D}^{-1}(\mathbb{E} + \mathbb{F})\boldsymbol{x}^{[k]} + \mathbb{D}^{-1}\boldsymbol{b}$$

Plan

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Gauss-Seidel** :

$$b_i = \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

ou encore

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j}x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j}x_j + \mathrm{A}_{i,i}x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j}x_j$$

La méthode itérative de **Gauss-Seidel** :

$$b_i = \mathrm{A}_{i,i}x_i^{[k+1]} + \sum_{j=1}^{i-1} \mathrm{A}_{i,j}x_j^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j}x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{b} = \mathbb{D}\boldsymbol{x}^{[k+1]} - \mathbb{E}\boldsymbol{x}^{[k+1]} - \mathbb{F}\boldsymbol{x}^{[k]}$$

ou encore

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij}x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij}x_j^{[k]}\right) \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{x}^{[k+1]} = (\mathbb{D} - \mathbb{E})^{-1}\mathbb{F}\boldsymbol{x}^{[k]} + (\mathbb{D} - \mathbb{E})^{-1}\boldsymbol{b}$$

Plan

Soit $w \in \mathbb{R}^*$.

$$x_i^{[k+1]} = w\hat{x}_i^{[k+1]} + (1 - w)x_i^{[k]}$$

où $\hat{x}_i^{[k+1]}$ est obtenu à partir de l'une des deux méthodes précédentes.
Avec la méthode de Gauss-Seidel : méthode S.O.R. (successive over relaxation)

$$x_i^{[k+1]} = \frac{w}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij}x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij}x_j^{[k]}\right) + (1 - w)x_i^{[k]} \quad \forall i \in [\![1, n]\!]$$

Exercice 3.1:

Déterminer la matrice d'itération $\mathbb{B}$ et le vecteur $\boldsymbol{c}$ tels que

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

en fonction de $\mathbb{D}$, $\mathbb{E}$, $\mathbb{F}$, et $\boldsymbol{b}$.

Proposition:

On pose $\mathbb{L} = \mathbb{D}^{-1}\mathbb{E}$ et $\mathbb{U} = \mathbb{D}^{-1}\mathbb{F}$.
La matrice d'itération de la méthode de Jacobi, notée $\mathbb{J}$, est donnée par

$$\mathbb{J} = \mathbb{D}^{-1}(\mathbb{E} + \mathbb{F}) = \mathbb{L} + \mathbb{U}, \tag{1}$$

La matrice d'itération de la méthode S.O.R., notée $\mathcal{L}_w$, est donnée par

$$\mathcal{L}_w = \left(\frac{\mathbb{D}}{w} - \mathbb{E}\right)^{-1}\left(\frac{1 - w}{w}\mathbb{D} + \mathbb{F}\right) = (\mathbb{I} - w\mathbb{L})^{-1}((1 - w)\mathbb{I} + w\mathbb{U}). \tag{2}$$

et elle vérifie

$$\rho(\mathcal{L}_w) \geqslant |w - 1|. \tag{3}$$

La matrice d'itération de Gauss-Seidel est $\mathcal{L}_1$ et elle correspond à

$$\mathcal{L}_1 = (\mathbb{D} - \mathbb{E})^{-1}\mathbb{F} = (\mathbb{I} - \mathbb{L})^{-1}\mathbb{U}. \tag{4}$$

Plan

Théorème:

Soit $\mathbb{A}$ une matrice régulière décomposée sous la forme $\mathbb{A} = \mathbb{M} - \mathbb{N}$ avec $\mathbb{M}$ régulière. On pose

$$\mathbb{B} = \mathbb{M}^{-1}\mathbb{N} \quad \text{et} \quad \boldsymbol{c} = \mathbb{M}^{-1}\boldsymbol{b}.$$

Alors la suite définie par

$$\boldsymbol{x}^{[0]} \in \mathbb{K}^n \quad \text{et} \quad \boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

converge vers $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$ quelque soit $\boldsymbol{x}^{[0]}$ si et seulement si $\rho(\mathbb{B}) < 1$.

Lien avec les méthodes de Jacobi, Gauss-Seidel, S.O.R.?

Corollaire:

Soit $\mathbb{A}$ une matrice vérifiant $\mathrm{A}_{i,i} \neq 0 \; \forall i$. Une condition nécessaire de convergence pour la méthode S.O.R. est que $0 < w < 2$.

Théorème: voir Lascaux-Théodor, vol.2, Théorème 19 et 20, pages 346 à 349

Soit $\mathbb{A}$ une matrice à diagonale strictement dominante ou une matrice inversible à diagonale fortement dominante alors

- la méthode de Jacobi est convergente,
- si $w \in]0,1]$ la méthode de Relaxation est convergente.

Théorème

Soit $\mathbb{A}$ une matrice hermitienne inversible en décomposée en $\mathbb{A} = \mathbb{M} - \mathbb{N}$ où $\mathbb{M}$ est inversible. Soit $\mathbb{B} = \mathbb{I} - \mathbb{M}^{-1}\mathbb{A}$, la matrice de l'itération. Supposons que $\mathbb{M}^* + \mathbb{N}$ (qui est hermitienne) soit définie positive. Alors $\rho(\mathbb{B}) < 1$ si et seulement si $\mathbb{A}$ est définie positive.

Preuve : voir exercice

Théorème

Soit $\mathbb{A}$ une matrice hermitienne définie positive, alors la méthode de relaxation converge si et seulement si $w \in]0,2[$.

Preuve : voir Lascaux-Théodor, *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, vol.2, Corollaire 24, page 351.

Plan

Principe de base

Résoudre :

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

Méthodes itératives :

$$\boldsymbol{x}^{[0]} \in \mathbb{K}^n \quad \text{et} \quad \boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

Algorithme :

$\boldsymbol{x}^{[0]}$ donné
Pour $k = 0,\ 1, \cdots$ **faire**
  $\boldsymbol{x}^{[k+1]} \leftarrow \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$
Fin Pour

Critère d'arrêt? Stockage de tous les $\boldsymbol{x}^{[k]}$?

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$\Longrightarrow$ boucle **Tantque**

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$\Longrightarrow$ boucle **Tantque**

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

Exemple de critère d'arrêt pour la convergence :
Soit $\boldsymbol{r}^{[k]} = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}^{[k]}$ le résidu.

$$\frac{\left\|\boldsymbol{r}^{[k]}\right\|}{\|\boldsymbol{b}\|} \leqslant \varepsilon$$

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$\Longrightarrow$ boucle **Tantque**

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

Exemple de critère d'arrêt pour la convergence :
Soit $\boldsymbol{r}^{[k]} = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}^{[k]}$ le résidu.

$$\frac{\|\boldsymbol{r}^{[k]}\|}{\|\boldsymbol{b}\|} \leqslant \varepsilon$$

Car dans ce cas, on a avec $\boldsymbol{e}^{[k]} = \bar{\boldsymbol{x}} - \boldsymbol{x}^{[k]}$

$$\frac{\|\boldsymbol{e}^{[k]}\|}{\|\bar{\boldsymbol{x}}\|} \leqslant \varepsilon \operatorname{cond}(\mathbb{A})$$

Algorithme 1 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :
$\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$,
$\boldsymbol{x}^0$: vecteur initial de $\mathbb{K}^n$,
ε : la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax : nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$
Résultat :
$\boldsymbol{x}^{\mathrm{tol}}$: un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$ ▷ $\boldsymbol{p}$ contient le vecteur précédent
  $\boldsymbol{x} \leftarrow$ calcul de l'itérée suivante en fonction de $\boldsymbol{p}$, $\mathbb{A}$, $\boldsymbol{b}$, ...
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Pour Jacobi , la suite des itérées est définie par

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right), \quad \forall i \in [\![1, n]\!].$$

Algorithme 2 $\boxed{\mathcal{R}_0}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  $\boxed{\boldsymbol{x} \leftarrow \text{calcul par Jacobi}}$
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 2 $\boxed{\mathcal{R}_1}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  Pour $i \leftarrow 1$ **à** n **faire**
    $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} \boldsymbol{p}_j \right)$
  Fin Pour
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 2 $\boxed{\mathcal{R}_1}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  Pour $i \leftarrow 1$ **à** n **faire**
    $\boxed{x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} \boldsymbol{p}_j \right)}$
  Fin Pour
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 2 $\boxed{\mathcal{R}_2}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  Pour $i \leftarrow 1$ **à** n **faire**
    $\mathrm{S} \leftarrow 0$
    Pour $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
      $\mathrm{S} \leftarrow \mathrm{S} + \mathrm{A}_{i,j} p_j$
    Fin Pour
    $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}}(b_i - \mathrm{S})$
  Fin Pour
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 2 Méthode itérative de Jacobi pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :
$\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$,
$\boldsymbol{x}^0$: vecteur initial de $\mathbb{K}^n$,
ε : la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax : nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$
Résultat :
$\boldsymbol{X}$: un vecteur de $\mathbb{K}^n$

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    Pour $i \leftarrow 1$ **à** n **faire**
      $\mathrm{S} \leftarrow 0$
      Pour $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
        $\mathrm{S} \leftarrow \mathrm{S} + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - \mathrm{S})/A(i,i)$
    Fin Pour
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Pour Gauss-Seidel , la suite des itérées est définie par

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

Algorithme 3 $\boxed{\mathcal{R}_0}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  $\boxed{\boldsymbol{x} \leftarrow \text{calcul par Gauss-Seidel}}$
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 3 $\boxed{\mathcal{R}_1}$

$k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
  $k \leftarrow k + 1$
  $\boldsymbol{p} \leftarrow \boldsymbol{x}$
  Pour $i \leftarrow 1$ **à** n **faire**
    $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j - \sum_{j=i+1}^{n} \mathrm{A}_{ij} \boldsymbol{p}_j \right)$
  Fin Pour
  $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
  $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
Fin Si

Algorithme 3 $\mathcal{R}_1$

```
k ← 0, x^tol ← ∅
x ← x^0, r ← b − 𝔸x,
tol ← ε(‖b‖ + 1)
Tantque ‖r‖ > tol et k ⩽ kmax faire
    k ← k + 1
    p ← x
    Pour i ← 1 à n faire

        x_i ← (1/A_ii) ( b_i − Σ_{j=1}^{i−1} A_ij x_j − Σ_{j=i+1}^{n} A_ij p_j )
    Fin Pour
    r ← b − 𝔸x,
Fin Tantque
Si ‖r‖ ⩽ tol alors
    x^tol ← x
Fin Si
```

$$\boxed{x_i \leftarrow \frac{1}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1}^{i-1}\mathrm{A}_{ij}x_j - \sum_{j=i+1}^{n}\mathrm{A}_{ij}p_j\right)}$$

Algorithme 3 $\mathcal{R}_2$

```
k ← 0, x^tol ← ∅
x ← x^0, r ← b − 𝔸x,
tol ← ε(‖b‖ + 1)
Tantque ‖r‖ > tol et k ⩽ kmax faire
    k ← k + 1
    p ← x
    Pour i ← 1 à n faire
        S ← 0
        Pour j ← 1 à i − 1 faire
            S ← S + A_{i,j} x_j
        Fin Pour
        Pour j ← i + 1 à n faire
            S ← S + A_{i,j} p_j
        Fin Pour
        x_i ← (1/A_ii) (b_i − S)
    Fin Pour
    r ← b − 𝔸x,
Fin Tantque
Si ‖r‖ ⩽ tol alors
    x^tol ← x
Fin Si
```


Algorithme 3 Méthode itérative de Gauss-Seidel pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :

- $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$,
- $\boldsymbol{b}$: vecteur de $\mathbb{K}^n$,
- $\boldsymbol{x}^0$: vecteur initial de $\mathbb{K}^n$,
- ε : la tolérence, $\varepsilon \in \mathbb{R}^+$,
- kmax : nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$

Résultat :

- $\boldsymbol{X}$: un vecteur de $\mathbb{K}^n$

```
Fonction X ← RSLGaussSeidel ( 𝔸, b, x^0, ε, kmax )
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à i − 1 faire
                S ← S + A(i, j) * x(j)
            Fin Pour
            Pour j ← i + 1 à n faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```



```
Fonction X ← RSLJacobi ( 𝔸, b, x^0, ε, kmax )
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à n (j ≠ i) faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```

```
Fonction X ← RSLGaussSeidel ( 𝔸, b, x^0, ε, kmax
)
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à i − 1 faire
                S ← S + A(i, j) * x(j)
            Fin Pour
            Pour j ← i + 1 à n faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```



```
Fonction X ← RSLJacobi ( 𝔸, b, x^0, ε, kmax )
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à n (j ≠ i) faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```

```
Fonction X ← RSLGaussSeidel ( 𝔸, b, x^0, ε, kmax
)
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à i − 1 faire
                S ← S + A(i, j) * x(j)
            Fin Pour
            Pour j ← i + 1 à n faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```


Même ossature puisque toutes deux basées sur l'Algorithme générique

Peut-on simplifier, clarifier et racourcir les codes?


```
Fonction X ← RSLJacobi ( 𝔸, b, x^0, ε, kmax )
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        Pour i ← 1 à n faire
            S ← 0
            Pour j ← 1 à n (j ≠ i) faire
                S ← S + A(i, j) * p(j)
            Fin Pour
            x(i) ← (b(i) − S)/A(i, i)
        Fin Pour
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```


Algorithme 4 Itération de Jacobi : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n}\mathrm{A}_{ij}y_j\right), \quad \forall i \in [\![1, n]\!].$$

Données :

- A : matrice de $\mathcal{M}_n(\mathbb{K})$,
- b : vecteur de $\mathbb{K}^n$,
- y : vecteur de $\mathbb{K}^n$,

Résultat :

- x : un vecteur de $\mathbb{K}^n$

```
Fonction x ← IterJacobi ( A, b, y )
    Pour i ← 1 à n faire
        S ← 0
        Pour j ← 1 à n (j ≠ i) faire
            S ← S + A(i, j) * y(j)
        Fin Pour
        x(i) ← (b(i) − S)/A(i, i)
    Fin Pour
Fin Fonction
```



```
Fonction X ← RSLJacobi2 ( 𝔸, b, x^0, ε, kmax )
    k ← 0, X ← ∅
    x ← x^0, r ← b − 𝔸 * x,
    tol ← ε(‖b‖ + 1)
    Tantque ‖r‖ > tol et k ⩽ kmax faire
        k ← k + 1
        p ← x
        x ← IterJacobi(𝔸, b, p)
        r ← b − 𝔸 * x,
    Fin Tantque
    Si ‖r‖ ⩽ tol alors
        X ← x
    Fin Si
Fin Fonction
```

Algorithme 5 Itération de Jacobi : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n}\mathrm{A}_{ij}y_j\right), \quad \forall i \in [\![1, n]\!].$$

Données :

- A : matrice de $\mathcal{M}_n(\mathbb{K})$,
- b : vecteur de $\mathbb{K}^n$,
- y : vecteur de $\mathbb{K}^n$,

Résultat :

- x : un vecteur de $\mathbb{K}^n$

```
Fonction x ← IterJacobi ( A, b, y )
    Pour i ← 1 à n faire
        S ← 0
        Pour j ← 1 à n (j ≠ i) faire
            S ← S + A(i, j) * y(j)
        Fin Pour
        x(i) ← (b(i) − S)/A(i, i)
    Fin Pour
Fin Fonction
```

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    Pour $i \leftarrow 1$ à n faire
      $S \leftarrow 0$
      Pour $j \leftarrow 1$ à $i-1$ faire
        $S \leftarrow S + A(i,j) * x(j)$
      Fin Pour
      Pour $j \leftarrow i+1$ à n faire
        $S \leftarrow S + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - S)/A(i,i)$
    Fin Pour
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Algorithme 6 Itération de Gauss-Seidel : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{A_{ii}}\left(b_i - \sum_{j=1}^{i-1} A_{i,j}x_j - \sum_{j=i+1}^{n} A_{i,j}y_j\right), \quad \forall i \in [\![1,n]\!].$$

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
Résultat :
x : un vecteur de $\mathbb{K}^n$

Fonction x $\leftarrow$ IterGaussSeidel (A, b, y)
  Pour $i \leftarrow 1$ à n faire
    $S \leftarrow 0$
    Pour $j \leftarrow 1$ à $i-1$ faire
      $S \leftarrow S + A(i,j) * x(j)$
    Fin Pour
    Pour $j \leftarrow i+1$ à n faire
      $S \leftarrow S + A(i,j) * y(j)$
    Fin Pour
    $x(i) \leftarrow (b(i) - S)/A(i,i)$
  Fin Pour
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel2 ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterGaussSeidel($\mathbb{A}, \boldsymbol{b}, \boldsymbol{p}$)
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Algorithme 7 Itération de Gauss-Seidel : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{A_{ii}}\left(b_i - \sum_{j=1}^{i-1} A_{i,j}x_j - \sum_{j=i+1}^{n} A_{i,j}y_j\right), \quad \forall i \in [\![1,n]\!].$$

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
Résultat :
x : un vecteur de $\mathbb{K}^n$

Fonction x $\leftarrow$ IterGaussSeidel (A, b, y)
  Pour $i \leftarrow 1$ à n faire
    $S \leftarrow 0$
    Pour $j \leftarrow 1$ à $i-1$ faire
      $S \leftarrow S + A(i,j) * x(j)$
    Fin Pour
    Pour $j \leftarrow i+1$ à n faire
      $S \leftarrow S + A(i,j) * y(j)$
    Fin Pour
    $x(i) \leftarrow (b(i) - S)/A(i,i)$
  Fin Pour
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel2 ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterGaussSeidel($\mathbb{A}, \boldsymbol{b}, \boldsymbol{p}$)
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi2 ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterJacobi($\mathbb{A}, \boldsymbol{b}, \boldsymbol{p}$)
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Les deux codes sont fortement similaires!

Peut-on éviter les copier/coller et gagner encore en lisibilité?

Ecriture Algorithme générique sous forme d'une fonction et on ajoute aux paramètres d'entrées une fonction formelle IterFonc calculant une itérée :

$$\boldsymbol{x} \leftarrow \mathrm{IterFonc}(\mathbb{A}, \boldsymbol{b}, \boldsymbol{y}).$$

Algorithme 7 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :
$\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$,
IterFonc : fonction de paramètres une matrice d'ordre n, et deux vecteurs de $\mathbb{K}^n$. retourne un vecteur de $\mathbb{K}^n$.
$\boldsymbol{x}^0$: vecteur initial de $\mathbb{K}^n$,
ε : la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax : nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$
Résultat :
$\boldsymbol{x}^{\mathrm{tol}}$: un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

Fonction $\boldsymbol{X} \leftarrow$ RSLMethIter ($\mathbb{A}, \boldsymbol{b}$, IterFonc, $\boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterFonc($\mathbb{A}, \boldsymbol{b}, \boldsymbol{p}$)
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi3 ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $\boldsymbol{X} \leftarrow$ RSLMethIter($\mathbb{A}, \boldsymbol{b}$, IterJacobi, $\boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel3 ($\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $\boldsymbol{X} \leftarrow$ RSLMethIter($\mathbb{A}, \boldsymbol{b}$, IterGaussSeidel, $\boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
Fin Fonction

Algorithme 7 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :
$\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$,
IterFonc : fonction de paramètres une matrice d'ordre n, et deux vecteurs de $\mathbb{K}^n$. retourne un vecteur de $\mathbb{K}^n$.
$\boldsymbol{x}^0$: vecteur initial de $\mathbb{K}^n$,
ε : la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax : nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$
Résultat :
$\boldsymbol{x}^{\mathrm{tol}}$: un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

Fonction $\boldsymbol{X} \leftarrow$ RSLMethIter ($\mathbb{A}, \boldsymbol{b}$, IterFonc, $\boldsymbol{x}^0, \varepsilon, \mathrm{kmax}$)
  $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ faire
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterFonc($\mathbb{A}, \boldsymbol{b}, \boldsymbol{p}$)
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ alors
    $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Méthode de relaxation utilisant Gauss-Seidel,avec $w \in \mathbb{R}^*$,

$$x_i^{[k+1]} = \frac{w}{A_{ii}}\left(b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{[k+1]} - \sum_{j=i+1}^{n} A_{ij}x_j^{[k]}\right) + (1-w)x_i^{[k]} \quad \forall i \in [\![1,n]\!]$$

Algorithme 8 Itération S.O.R.

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
w : réel non nul.
Résultat :
x : un vecteur de $\mathbb{K}^n$

Fonction x $\leftarrow$ IterSOR (A, b, y, w)
  Pour $i \leftarrow 1$ à n faire
    $S \leftarrow 0$
    Pour $j \leftarrow 1$ à $i-1$ faire
      $S \leftarrow S - A(i,j) * x(j)$
    Fin Pour
    Pour $j \leftarrow i+1$ à n faire
      $S \leftarrow S - A(i,j) * y(j)$
    Fin Pour
    $x(i) \leftarrow w * (b(i) - S)/A(i,i) + (1-w) * x(i)$
  Fin Pour
Fin Fonction

Paramètre w "en trop" dans l'appel de la fonction IterSOR pour pouvoir utiliser la fonction générique RSLMethIter !

Méthode de relaxation utilisant Gauss-Seidel,avec $w \in \mathbb{R}^*$,

$$x_i^{[k+1]} = \frac{w}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) + (1-w) x_i^{[k]} \quad \forall i \in [\![1, n]\!]$$

Algorithme 9 Itération S.O.R.

Données :
- A : matrice de $\mathcal{M}_n(\mathbb{K})$,
- b : vecteur de $\mathbb{K}^n$,
- y : vecteur de $\mathbb{K}^n$,
- w : réel non nul.

Résultat :
- x : un vecteur de $\mathbb{K}^n$

```
Fonction x ← ITERSOR ( A, b, y, w )
  Pour i ← 1 à n faire
    S ← 0
    Pour j ← 1 à i − 1 faire
      S ← S − A(i, j) * x(j)
    Fin Pour
    Pour j ← i + 1 à n faire
      S ← S − A(i, j) * y(j)
    Fin Pour
    x(i) ← w * (b(i) − S)/A(i, i) + (1 − w) * x(i)
  Fin Pour
Fin Fonction
```

Paramètre w "en trop" dans l'appel de la fonction ITERSOR pour pouvoir utiliser la fonction générique RSLMETHITER !

```
Fonction X ← RSLSOR3 ( 𝔸, b, w, x⁰, ε, kmax )
  ITERFUN ← ((𝕄, r, s) ↦ ITERSOR(𝕄, r, s, w))
  X ← RSLMETHITER(𝔸, b, ITERFUN, x⁰, ε, kmax)
Fin Fonction
```