Analyse Numérique Licence L3 de Mathématique 2015-2016, TP3

Soit $A \in \mathcal{M}_n(\mathbb{R})$ une matrice SDP (Symétrique Définie Positive). On considère $b \in \mathbb{R}^n$ et le système linéaire $Ax = b$ de solution $x \in \mathbb{R}^n$.

$$\text{On pose } e^{(k)} = x - x^{(k)},\ r^{(k)} = Ae^{(k)} = b - Ax^{(k)},$$

et on considère l'algorithme itératif du gradient conjugué pour résoudre le système $Ax = b$: $x^{(1)}$ donné, $r^{(1)} = p^{(1)} = b - Ax^{(1)}$ et pour $k \in \mathbb{N}$

$$\begin{cases} x^{(k+1)} = x^{(k)} + \alpha^{(k)}\ p^{(k)}, \\ r^{(k+1)} = r^{(k)} - \alpha^{(k)}\ Ap^{(k)}, \\ p^{(k+1)} = r^{(k+1)} + \beta^{(k)}\ p^{(k)}, \\ \alpha^{(k)} = \frac{(r^{(k)}, r^{(k)})}{(p^{(k)}, Ap^{(k)})}, \\ \beta^{(k)} = \frac{(r^{(k+1)}, r^{(k+1)})}{(r^{(k)}, r^{(k)})}. \end{cases}$$

On va comparer la convergence de cet algorithme avec celle de l'algorithme de Richardson.

(1) Soient $n_x \in \mathbb{N}^*$, $n_y \in \mathbb{N}^*$, $n = n_x n_y$, $\Delta x = \frac{1}{n_x+1}$, $\Delta y = \frac{1}{n_y+1}$. Programmer dans scilab la matrice pentadiagonale $A \in \mathcal{M}_n(\mathbb{R})$ telle que
   $A_{i,i} = \frac{2}{\Delta x^2} + \frac{2}{\Delta y^2}$ pour $i = 1, \cdots, n$,
   $A_{i,i+1} = -\frac{1}{\Delta x^2}$ pour $i = 1, \cdots, n-1$,
   $A_{i,i-1} = -\frac{1}{\Delta x^2}$ pour $i = 2, \cdots, n$,
   $A_{i,i+n_x} = -\frac{1}{\Delta y^2}$ pour $i = 1, \cdots, n - n_x$,
   $A_{i,i-n_x} = -\frac{1}{\Delta y^2}$ pour $i = n_x + 1, \cdots, n$,
   $A_{i,j} = 0$ pour tous les autres indices.
   On utilisera la commande $diag(u)$ et $diag(u_x, \pm 1)$, $diag(u_y, \pm n_x)$ où u et u_x, u_y sont des vecteurs.

(2) Programmer dans scilab le vecteur colonne $y \in \mathbb{R}^n$ tel que

$$y(i_x + (i_y - 1)n_x) = \sin(100 i_x \Delta x) + \sin(10 i_y \Delta y),\ i_x = 1, \cdots, n_x,\ i_y = 1, \cdots, n_y,$$

   et calculer $b = Ay$.

   Par la suite on va résoudre le système $Ax = b$ de solution y par l'algorithme du gradient conjugué.

(3) Programmer dans scilab l'algorithme du gradient conjugué suivant :
   — Choix de la précision $\epsilon = 10^{-10}$ sur le résidu relatif
   — Initialisation : $x^{(1)} = 0$ (dans $\mathbb{R}^n$), $r^{(1)} = b - Ax^{(1)}, p^{(1)} = r^{(1)}$, $nr^{(1)} = \|r^{(1)}\|_2$, $k = 1$
   — Tant Que $\frac{nr^{(k)}}{nr^{(1)}} > \epsilon$ Faire
      $q^{(k)} = Ap^{(k)}$
      $\alpha^{(k)} = \frac{(r^{(k)}, r^{(k)})}{(p^{(k)}, q^{(k)})}$
      $x^{(k+1)} = x^{(k)} + \alpha^{(k)} r^{(k)}$
      $r^{(k+1)} = r^{(k)} - \alpha^{(k)} q^{(k)}$
      $\beta^{(k)} = \frac{(r^{(k+1)}, r^{(k+1)})}{(r^{(k)}, r^{(k)})}$
      $p^{(k+1)} = r^{(k+1)} + \beta^{(k)} p^{(k)}$
      $nr^{(k+1)} = \|r^{(k+1)}\|_2$

$k = k + 1$

— End Tant Que

— $nit = k$

La forme programmée dans scilab ne stokera que l'itéré courant de p, q, x, r. On utilisera la boucle *while condition*; *end*, le produit scalaire $z' * z$. Un seul produit matrice vecteur est autorisé par itération.

(4) Tests numériques :
Tracer la courbe de la norme du résidu $nr^{(k)}$ fonction de k ainsi que la courbe de la norme de l'erreur relative $(Ae^{(k)}, e^{(k)})/(Ae^{(1)}, e^{(1)})$ fonction de k où $e^{(k)} = y - x^{(k)}$. Vérifier que l'algorithme converge vers la solution exacte (à la précision machine près) en au plus n itération. Regarder la dépendance du nombre d'itérations à précision ϵ fixée en fonction de n. Vérifier que $(Ae^{(k)}, e^{(k)})$ décroit strictement alors que le comportement de $nr^{(k)}$ n'est pas monotone.

(5) Comparer (sur la même figure) la convergence de l'algorithme du gradient conjugué avec celle de l'algorithme de Richardon à pas variable (à reprendre du TP2).