# TP4 : Conditionnement et méthodes de gradient

Vous êtes invités à créer un répertoire `TP4` et un fichier `.m` pour chacune des parties ci-dessous (comportant les lignes de code correspondant à la résolution de l'exercice).

En cas de blocage, commencez toujours par regarder l'aide !!

## Présentation et indications

Le but de ce TP est d'étudier l'influence du conditionnement sur la résolution de systèmes linéaires par des méthodes itératives. Dans tout le TP, les matrices des systèmes linéaires seront symétriques définies positives et les méthodes itératives étudiées seront la méthode de gradient à pas constant avec choix optimal du pas et la méthode de gradient à pas optimal.

## 1. Conditionnement de matrices particulières

1.1. **Matrice de Hilbert.** On s'intéresse à la matrice de Hilbert $H \in \mathcal{M}_n(\mathbb{R})$ définie par

$$H_{ij} = \frac{1}{i+j-1}, \qquad 1 \leq i, j \leq n.$$

1. Montrer que $H$ est symétrique définie positive (*Indication :* $H_{i,j} = \int_0^1 t^{i+j-2} dt$ *pour* $1 \leq i, j \leq n$).
2. A l'aide de la fonction `hilb` et de la fonction `cond` de Matlab, étudier l'évolution du conditionnement en norme 2 de la matrice de Hilbert en fonction de $n$. On pourra notamment représenter graphiquement le logarithme du conditionnement de $H$ en fonction de $n$ (`semilogy`), pour $n$ variant de 1 à 10.
3. Que conjecturez-vous sur l'expression du conditionnement de $H$ en fonction de $n$ ?

1.2. **Matrice de diffusion.** On considère maintenant $n = 2p+1$ impair et pour $\delta_1$ et $\delta_2$ deux réels strictement positifs fixés on définit la matrice de diffusion $D$ par

$$D_{ij} = \begin{cases} 2\delta_1 & \text{si} \quad i = j \leq p \\ -\delta_1 & \text{si} \quad |i-j| = 1, \quad |i+j| \leq 2p+1 \\ \delta_1 + \delta_2 & \text{si} \quad i = j = p+1 \\ 2\delta_2 & \text{si} \quad i = j \geq p+2 \\ -\delta_2 & \text{si} \quad |i-j| = 1, \quad |i+j| \geq 2p+3 \end{cases}$$

1. Que vaut la matrice $D$ si $\delta_1 = \delta_2 = 1$ ?

2. Montrer que $D$ est symétrique définie positive.

3. Écrire une fonction `[D]= matdiff(p,d1,d2)` qui calcule la matrice $D$ pour les valeurs de $p$, $\delta_1$ et $\delta_2$ données.

4. On suppose tout d'abord $\delta_1 = \delta_2 = 1$. Que sait-on de $\kappa_2(A)$ ? Mettre en évidence graphiquement le comportement de $\kappa_2(A)$ (on pourra prendre comme valeurs de $p$ : $p = 2^i$ pour $i = 1$ à 8).

5. On suppose désormais que $p = 20$ et $\delta_1 = 1$. Étudier numériquement l'évolution de $\kappa_2(D)$ en fonction de $\delta_2$. On pourra tester les valeurs suivantes pour $\delta_2$ : 1, 10, 100, 1000, 10000, 100000. Représenter graphiquement le résultat obtenu.

## 2. Méthodes de gradient

Soit $A \in \mathcal{M}_n(\mathbb{R})$ une matrice symétrique définie positive et $b \in \mathbb{R}^n$. On rappelle la méthode de gradient à pas constant et la méthode de gradient à pas optimal vues en cours.

**Méthode de gradient à pas constant $\alpha$**

$$\left\{ \begin{array}{l} x_0 \quad \text{donné} \\ x^{(k+1)} = x^{(k)} + \alpha r^{(k)} \\ \text{avec} \quad r^{(k)} = b - Ax^{(k)} \end{array} \right\} \quad \forall k \geq 0$$

On rappelle que le choix optimal de $\alpha$ est $\alpha_{pot} = \dfrac{2}{\lambda_1 + \lambda_n}$ où $\lambda_1$ et $\lambda_n$ sont respectivement la plus petite et la plus grande valeur propre de $A$.

**Méthode de gradient à pas optimal**

$$\left\{ \begin{array}{l} x_0 \quad \text{donné} \\ x^{(k+1)} = x^{(k)} + \alpha_k r^{(k)} \\ \text{avec} \quad r^{(k)} = b - Ax^{(k)} \quad \text{et} \quad \alpha_k = \dfrac{(r^{(k)}, r^{(k)})}{(Ar^{(k)}, r^{(k)})} \end{array} \right\} \quad \forall k \geq 0$$

Il s'agit de méthodes itératives. En pratique, on arrêtera les itérations quand la norme euclidienne du résidu $r^{(k)}$ est plus petite qu'une tolérance fixée `errmax` ou quand on a atteint un nombre d'itérations maximal fixé `nbitmax`. Pour ce TP on pose `nbitmax=1e4` et `errmax=1e-3`.

1. Écrire une fonction `[x,nbiter]=grad_pas_constant_opt(A,b,errmax,nbitmax)` qui résout le système $Ax = b$ par la méthode de gradient à pas constant, avec le choix optimal de $\alpha$ (calculé grâce à la fonction `eig` de Matlab).

2. Écrire une fonction `[x,nbiter]=grad_pas_opt(A,b,errmax,nbitmax)` qui résout le système $Ax = b$ par la méthode de gradient à pas optimal.

3. Tester vos fonctions sur la matrice de diffusion avec $\delta_1 = \delta_2 = 1$ et $p = 10$. On pourra prendre comme second membre $b = (1, \ldots, 1)^t$.

4. Résoudre le système linéaire $Hx = b$ pour $n = 1$ à 10 par les deux méthodes (on prendre également $b = (1, \ldots, 1)^t$). Représenter graphiquement le nombre d'itérations en fonction de $n$. Comparer avec la courbe obtenue pour le conditionnement.

5. Même question pour la résolution du système $Dx = b$ pour les deux cas :

   a) $\delta_1 = 1$, $\delta_2 = 5$ et $p = 3i$ pour $i = 1, 2, \ldots, 10$,

   b) $\delta_1 = 1$, $p = 20$ et $\delta_2 = 1, 2, \ldots, 10$.

   Représenter graphiquement le nombre d'itérations en fonction de $p$ ou en fonction de $\kappa_2$. Comparer avec les courbes obtenues pour le conditionnement.

## 3. Préconditionnement

La matrice $A$ du système linéaire $Ax = b$ pouvant être, comme on l'a vu, très mal conditionnée, on cherche à transformer le système en un système équivalent, de type $M^{-1}Ax = M^{-1}b$, avec une matrice $M^{-1}A$ mieux conditionnée. La matrice $M$ est appelée "préconditionneur" du système linéaire ; elle doit être facile à inverser (par exemple diagonale ou triangulaire). Remarquons que le préconditionneur "idéal" serait bien sûr $A$ elle-même, mais qu'il est inutilisable en pratique.

**Méthode de gradient à pas optimal avec préconditionneur**

$$\left\{ \begin{array}{l} x^{(0)} \quad \text{donné} \quad \text{et} \quad r^{(0)} = b - Ax^{(0)} \\ \\ x^{(k+1)} = x^{(k)} + \alpha_k z^{(k)}, \qquad r^{(k+1)} = r^{(k)} - \alpha_k A z^{(k)} \\ \\ \text{avec} \quad Mz^{(k)} = r^{(k)} \quad \text{et} \quad \alpha_k = \dfrac{\left(r^{(k)}, z^{(k)}\right)}{\left(Az^{(k)}, z^{(k)}\right)} \end{array} \right\} \quad \forall k \geq 0$$

1. Écrire une fonction `[x,nbiter]=grad_precond_pas_opt(A,M,b,errmax,nbitmax)` qui résout le système $Ax = b$ par la méthode de gradient à pas optimal avec préconditionneur $M$.

2. Tester l'efficacité d'un préconditionnement diagonal (`M=diag(diag(A))`), puis d'un préconditionnement triangulaire (`M=tril(A)`) sur la matrice du laplacien discret. Commenter.

3. Tester l'efficacité d'un préconditionnement triangulaire dans les cas des matrices à diffusion et de Hilbert. Pour cela on peut reprendre les codes des questions 4 et 5 de la partie précédente en remplaçant l'un des méthodes.