Projet - Analyse Numérique Matricielle - Modèles d'antennes

(Ce projet est adapté du texte "public2017-B1" de modélisation posé au concours de l'agrégation externe de mathématiques).

Nous nous intéressons dans ce projet à un modèle simple de réseau et d'antennes. Une antenne est un dispositif émetteur d'ondes électromagnétiques. On appelle puissance nominale de l'antenne la valeur x de la puissance émise par l'antenne au point où elle se trouve. La puissance reçue par un récepteur situé à une distance r de l'antenne est alors donnée par $xV(r)$ où $V(r)$ est une fonction normalisée (i.e. $V(0) = 1$) qui mesure la perte de puissance du signal en fonction de la distance. On supposera dans la suite qu'en première approximation cette perte de signal est bien décrite en prenant

$$V(r) = \frac{1}{1 + \left(\frac{r}{r_0}\right)^2} \tag{1}$$

où $r_0 > 0$ est une distance caractéristique du modèle.

Dans le modèle unidimensionnel de réseau que nous proposons d'étudier, nous supposons que n antennes comme ci-dessus sont situées sur une droite et que deux antennes successives sont distantes d'une longueur d donnée et fixée. Les positions des antennes sont repérées par les abscisses $a_i = id$ pour $i \in \{1, \cdots, n\}$.

Si on note x_i la puissance nominale de l'antenne numéro i, la puissance effective reçue par un récepteur situé à la position $s \in \mathbb{R}$ est donnée par :

$$\Pi(s) = \sum_{j=1}^{n} x_j V(|s - a_j|). \tag{2}$$

On s'intéresse plus particulièrement à la puissance effective du signal aux pieds des antennes. Ainsi, pour un vecteur $x = (x_i)_{1 \le i \le n}$ des puissances nominales donné, le vecteur $\pi = (\pi_i)_{1 \le i \le n}$ des puissances effectives au pied de chaque antenne est donné par :

$$\pi_i = \Pi(a_i) = \sum_{j=1}^{n} x_j V(|a_i - a_j|) = \sum_{j=1}^{n} \frac{x_j}{1 + \frac{d^2}{r_0^2}|i - j|^2}. \tag{3}$$

1. Le problème du réseau complet

1.1. Position du problème. — On souhaite maintenant déterminer les valeurs que l'on doit donner aux puissances nominales $x = (x_i)_{1 \le i \le n}$ des différentes antennes pour que les puissances effectives $\pi = (\pi_i)_{1 \le i \le n}$ au pied des antennes soient toutes égales à la même valeur, que l'on prendra égale à 1 pour fixer les idées.

On note donc $\pi = {}^t(1, \cdots, 1)$ et on cherche un vecteur de réels positifs $x \in \mathbb{R}^n$ tel que :

$$\pi = Ax \quad \text{avec} \quad A = \begin{pmatrix} 1 & V(d) & V(2d) & \cdots & V((n-1)d) \\ V(d) & 1 & V(d) & \ddots & \vdots \\ V(2d) & \ddots & \ddots & \ddots & V(2d) \\ \vdots & \ddots & \ddots & 1 & V(d) \\ V((n-1)d) & \cdots & V(2d) & V(d) & 1 \end{pmatrix} \tag{4}$$

Pour tout $\gamma > 0$ on pose :

$$f(\gamma) = \sum_{m=1}^{+\infty} \frac{1}{1 + \gamma^2 m^2}. \tag{5}$$

On note γ_0 et γ_1 les uniques réels positifs tels que $f(\gamma_0) = \frac{1}{2}$ et $f(\gamma_1) = \frac{1}{4}$. (On justifie aisément que f, γ_0 et γ_1 sont bien définis.)

Théorème 1.1.

Si $\frac{d}{r_0} > \gamma_0$, *alors la matrice* $A \in \mathcal{M}_n(\mathbb{R})$ *est symétrique définie positive pour toute valeur de n et dans ce cas, l'unique solution x de (4) vérifie :*

$$\sup_{1\leq i\leq n} |x_i| \leq \frac{1}{1-2f(\frac{d}{r_0})} \tag{6}$$

Question 1.

Démontrer le théorème 1.1. En suggestion, on pourra par exemple :

1. pour montrer l'inversibilité de A, s'aider d'un résultat vu en cours et td ;
2. utiliser le résultat classique suivant en le démontrant : soit $B = (b_{i,j})$ une matrice à diagonale strictement dominante (sur les lignes) : on pose $l(B) = \min_i\{l_i(B)\} > 0$ où $l_i(B) = |b_{i,i}| - \sum_{j=1, j\neq i}^{n} |b_{i,j}|$. Alors, $\|B^{-1}\|_\infty \leq \frac{1}{l(B)}$.

Théorème 1.2.

Si $\frac{d}{r_0} \geq \gamma_1$, *alors l'unique solution x de (4) est positive. Plus précisément, on a :*

$$\inf_{1\leq i\leq n} x_i \geq \frac{1-4f(\frac{d}{r_0})}{1-2f(\frac{d}{r_0})} \tag{7}$$

Question 2.

Démontrer le théorème 1.2. On pourra pour cela expliciter l'équation (4) et utiliser le résultat du théorème 1.1 (en justifiant son emploi).

Sous de bonnes hypothèses qu'on **supposera réalisées dans toute la suite de ce projet**, le problème (4) admet donc une unique solution positive.

1.2. Solveur direct. —

1.2.1. Méthode LU. —

Question 3.

1. Ecrire une fonction python `MatriceA(n,tau)` avec en entrée l'entier naturel n et le réel $\tau = \dfrac{d}{r_0}$. En sortie, la matrice A définie par (4).
2. Avec $n = 20$ et $\tau = 1$, résoudre numériquement le problème (4) par la méthode LU. Vous utiliserez vos scripts python établis en TP.
3. Question de cours : rappeler le nombre d'opérations (en fonction de n) effectuées avec la méthode LU.

1.2.2. Méthode adaptée au contexte. — Avec la méthode LU, la complexité numérique devient déraisonnable pour n grand. Par ailleurs, cette méthode n'utilise pas les propriétés particulières de la matrice A : elle est Toeplitz symétrique, au sens suivant.

Définition 1.3.

Une matrice carrée T d'ordre n à coefficients réels est dite de **Toeplitz symétrique** ssi il existe un vecteur $t = (t_k)_{k\in\{0,\cdots,n-1\}} \in \mathbb{R}^n$ tel que : pour tout (i, j) dans $\{1, \cdots, n\}^2$ tel que $|j - i| = k$, on a $T_{i,j} = t_k$.

Notez que le vecteur t est la première colonne de la matrice $T = \begin{pmatrix} t_0 & t_1 & t_2 & \cdots & t_{n-1} \\ t_1 & t_0 & t_1 & \ddots & \vdots \\ t_2 & t_1 & t_0 & \ddots & t_2 \\ \vdots & \ddots & \ddots & \ddots & t_1 \\ t_{n-1} & \cdots & t_2 & t_1 & t_0 \end{pmatrix}$.

On présente maintenant une méthode directe en deux étapes pour résoudre un système linéaire dont **la matrice T est Toeplitz symétrique définie positive**. Pour tout $k \in \{1, \cdots, n\}$, on note T_k la matrice carrée de taille k obtenue en conservant seulement les k premières lignes et k premières colonnes de T. On note S_k l'opérateur d'inversion des indices pour les vecteurs de $\mathbb{R}^k$ défini par : $\forall y \in \mathbb{R}^k$, $\forall i \in \{1, \cdots, k\}$, $(S_k y)_i = y_{k+1-i}$. Autrement dit : $S_k \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix} = \begin{pmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_1 \end{pmatrix}$

1. **Première étape** : On construit une famille de vecteurs $f_k \in \mathbb{R}^k$ pour tout $k = 1, \cdots, n$ de la manière suivante :
   (a) On pose $f_1 = (\dfrac{1}{t_0}) \in \mathbb{R}$.
   (b) Pour $k = 2, \cdots, n$ on détermine $\alpha_k, \beta_k \in \mathbb{R}$ tels que le vecteur
   $$f_k = \alpha_k \begin{pmatrix} S_{k-1} f_{k-1} \\ 0 \end{pmatrix} + \beta_k \begin{pmatrix} 0 \\ f_{k-1} \end{pmatrix} \text{ vérifie } T_k f_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

2. **Deuxième étape** : Soit à résoudre $Tx = b$. On construit une famille de vecteurs x_k de taille k pour tout $k = 1, \cdots, n$ vérifiant $T_k x_k = {}^t(b_1, \cdots, b_k)$. Pour cela on effectue la récurrence suivante :
   (a) On pose $x_1 = (\frac{b_1}{t_0})$
   (b) Pour $k = 2, \cdots, n$, on cherche x_k sous la forme $x_k = \begin{pmatrix} x_{k-1} \\ 0 \end{pmatrix} + \theta_k f_k$ où $\theta_k \in \mathbb{R}$ est déterminé pour que l'équation $T_k x_k = {}^t(b_1, \cdots, b_k)$ soit vérifiée.

Cet algorithme nécessite $O(n^2)$ opérations, il est donc plus rapide que la méthode du pivot de Gauss pour des grandes valeurs de n.

Question 4.

1. Montrer que l'algorithme décrit précédemment, sous l'hypothèse où la matrice T est Toeplitz symétrique définie positive, fournit bien la solution recherchée et qu'il s'effectue en $O(n^2)$ opérations (en comptabilisant première et deuxième étape).

   Suggestions pour la **Première étape**. On pourra par exemple :
   (a) Montrer que T_k est inversible, pour tout $k = 1, \cdots, n$;
   (b) Montrer que $T_k f_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$ implique que $T_k(S_k f_k) = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, pour tout $k = 1, \cdots, n-1$;
   (c) Montrer que pour tout $k = 2, \cdots, n$, le vecteur $f_{k-1} = \begin{pmatrix} f_{1,(k-1)} \\ \vdots \\ f_{(k-1),(k-1)} \end{pmatrix}$ vérifie $f_{(k-1),(k-1)} > 0$, puis montrer que les vecteurs $T_k \begin{pmatrix} S_{k-1} f_{k-1} \\ 0 \end{pmatrix}$ et $T_k \begin{pmatrix} 0 \\ f_{k-1} \end{pmatrix}$ de $\mathbb{R}^k$ sont linéairement indépendants ;
   (d) Montrer que pour tout $k = 2, \cdots, n$, les conditions $f_k = \alpha_k \begin{pmatrix} S_{k-1} f_{k-1} \\ 0 \end{pmatrix} + \beta_k \begin{pmatrix} 0 \\ f_{k-1} \end{pmatrix}$ et $T_k f_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$ sont équivalentes à la résolution d'une équation du type $\begin{pmatrix} 1 & \delta_k \\ \delta_k & 1 \end{pmatrix} \begin{pmatrix} \alpha_k \\ \beta_k \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ où $\delta_k \neq \pm 1$ s'exprime explicitement en fonction des coordonnées de f_{k-1} et de celles du vecteur $t = (t_0, \cdots, t_n)$.
   (e) Montrer que le calcul de la suite (f_k) nécessite $O(n^2)$ opérations.

   Pas de suggestion pour la **Deuxième étape** (qu'il est nécessaire de traiter !).
2. Ecrire une fonction python `Etape1(t)` avec en entrée le vecteur $t = (t_0, \cdots, t_{n-1})$ caractérisant la matrice Toeplitz T. En sortie, la suite des vecteurs $(f_1, \cdots, f_n)$.
3. Ecrire une fonction python `Etape2(t,b)` avec en entrée le vecteur $t = (t_0, \cdots, t_{n-1})$ caractérisant la matrice Toeplitz T et le vecteur b. En sortie, la solution x de l'équation $Tx = b$.

4. Appliquer vos fonctions précédentes à la résolution du problème (4) avec $n = 20$ et $\tau = 1$. Comparer votre résultat à celui obtenu à la question 3.

2. Le problème du réseau partiel

2.1. Position du problème. — On s'intéresse à la situation dans laquelle un certain nombre d'antennes du réseau sont en panne. On notera I l'ensemble des indices $i \in \{1, \cdots, n\}$ correspondant aux antennes qui fonctionnent et $P = (p_{i,j})$ la matrice de projection sur cet ensemble I, c'est à dire $p_{i,j} = 1$ si $i = j \in I$ et $p_{i,j} = 0$ sinon.

On souhaiterait donc résoudre l'équation (4) sous la contrainte $Px = x$, ce qui n'est bien sûr pas possible en général. Il nous faut donc relaxer le problème. Nous proposons deux façons de le faire.

2.1.1. Première approche. — On demande que la puissance effective soit exactement égale à 1 uniquement au pied des antennes qui fonctionnent. On se propose donc de résoudre :

$$\begin{cases} PAPx = P\pi \\ \text{avec } Px = x \end{cases} \tag{8}$$

Question 5.

1. Montrer que le problème (8) admet une unique solution et qu'elle répond bien à la réponse souhaitée dans cette première approche.
2. En adaptant vos algorithmes Python de la question 3, calculer la solution x du problème (8) avec $n = 20$, $d = r_0$ quand les antennes numéros 7, 8, 15, 16, 17, 18 sont en panne, ainsi que les puissances effectives $\pi = Ax$.
3. Visualisation : en utilisant `matplotlib`, afficher dans une même fenêtre les valeurs de x et de π calculé.

2.1.2. Deuxième approche. — On demande que les puissances effectives au pied de toutes les antennes soient en moyenne les plus proches de 1. Pour cela on va chercher un vecteur x qui réalise le problème d'optimisation avec contraintes suivant :

$$\inf_{x \in \mathbb{R}^n, Px = x} \|Ax - \pi\|_2^2 \tag{9}$$

où $\|.\|_2$ est la norme euclidienne usuelle de $\mathbb{R}^n$. On vérifie que ceci revient à chercher l'unique solution du problème linéaire suivant :

$$\begin{cases} P\,{}^tAAPx = P\,{}^tA\pi \\ \text{avec } Px = x \end{cases} \tag{10}$$

Question 6.

1. Montrer que le problème (10) admet une unique solution.
2. Montrer que le problème (10) est équivalent au problème (9). (On pourra se référer au problème des moindres carrés vu dans le cours, en adaptant le raisonnement menant à l'équation normale).

3. En adaptant vos algorithmes Python de la question 3, calculer la solution x du problème (10) avec $n = 20$, $d = r_0$ quand les antennes numéros 7, 8, 15, 16, 17, 18 sont en panne, ainsi que les puissances effectives $\pi = Ax$.
4. Visualisation : en utilisant `matplotlib`, afficher dans une même fenêtre les valeurs de x et de π calculé.

2.2. Solveur itératif. — Dans le cadre du réseau partiel, les matrices qui interviennent sont PAP ou $P{}^tAAP$ qui, malheureusement, n'ont plus la structure Toeplitz symétrique de la matrice A. On ne peut donc pas utiliser de façon immédiate le solveur direct proposé en §1.2.
Pour résoudre un problème du type $PMPx = Pb$ avec $Px = x$ où M est une matrice symétrique définie positive et P la projection définie plus haut, on propose d'utiliser une version adaptée de l'algorithme du gradient conjugué qui s'écrit :

1. Initialisation : Choisir x_0 tel que $Px_0 = x_0$. Calculer le résidu $\rho_0 = Pb - PMPx_0$. Poser $g_0 = \rho_0$.
2. Pour $k = 0, \cdots$ et si $\rho_k \neq 0$, faire :
$$\alpha_k = \frac{\langle \rho_k, \rho_k \rangle}{\langle PMPg_k, g_k \rangle}, \quad x_{k+1} = x_k + \alpha_k g_k, \quad \rho_{k+1} = \rho_k - \alpha_k PMPg_k,$$
$$\beta_k = \frac{\langle \rho_{k+1}, \rho_{k+1} \rangle}{\langle \rho_k, \rho_k \rangle}, \quad g_{k+1} = \rho_{k+1} + \beta_k g_k$$
où $\langle ., . \rangle$ désigne le produit scalaire euclidien.

Théorème 2.1.

L'algorithme proposé converge en au plus n itérations vers l'unique solution x du problème $PMPx = Pb$, $Px = x$.

Chaque itération de l'algorithme nécessite un certain nombre de calculs de produits scalaires mais surtout un calcul d'un produit matrice/vecteur de la forme My. Dans le cadre qui nous intéresse, la matrice M (égale à A ou ${}^tAA$) est pleine et le calcul d'un tel produit matrice/vecteur coûte donc a priori $O(n^2)$ opérations. Au final, le solveur itératif proposé va coûter, au pire, $O(n^3)$ opérations.

Question 7.

1. Démontrer le théorème 2.1.
2. Ecrire une fonction python `Grad(M,P,b)` avec en entrée une matrice M symétrique définie positive, une matrice de projection P, un vecteur b. En sortie, la solution x du problème $PMPx = Pb$, $Px = x$ par la méthode adaptée de l'algorithme du gradient conjugué.
3. Appliquer l'algorihme au cas du réseau partiel d'antennes avec $n = 20$, $d = r_0$ et où les antennes numéros 7, 8, 15, 16, 17, 18 sont en panne, pour les 2 approches décrites au §2.1.