Optimisation

Master 1 Statistique & Data Science, Ingénierie Mathématique, 2019-2020

Feuille de TD/TP n°6 : Algorithmes pour la minimisation (sous contraintes)

Exercice 1. On suppose donnés des couples (x_i, y_i), $1 \leqslant i \leqslant n$, $n \geqslant 2$, avec les x_i deux à deux distincts. On cherche la droite de régression linéaire approchant au mieux ces valeurs au sens des moindres carrés :

$$\text{Trouver } (a,b) \in \mathbb{R}^2 \text{ minimisant} \quad f(a,b) = \sum_{i=1}^{n} (y_i - a x_i - b)^2.$$

1. Montrer que f est une fonctionnelle quadratique et préciser la matrice A, le vecteur v et la constante c telle que, si l'on note $x = (a,b)^T$,

   $$f(a,b) = \langle Ax, x\rangle - \langle v, x\rangle + c$$

   à l'aide des notations

   $$S_x = \sum_{i=1}^{n} x_i \quad S_y = \sum_{i=1}^{n} y_i \quad S_{x^2} = \sum_{i=1}^{n} x_i^2 \quad S_{y^2} = \sum_{i=1}^{n} x_i^2$$

2. Justifier que le problème d'optimisation considéré admet une unique solution.
3. On considère maintenant le problème contraint à ordonnée à l'origine positive :

   $$\text{Trouver } (a,b) \in \mathbb{R} \times [0, +\infty[\text{ minimisant} \quad f(a,b) = \sum_{i=1}^{n} (y_i - a x_i - b)^2.$$

   Déterminer la solution de ce problème à l'aide de conditions de KKT.
4. Proposer un pseudo-code pour résoudre le problème contraint.
5. Ecrire deux fonctions

   `[a,b] = regression_lin(x,y)` et
   `[a,b] = regression_lin_bpos(x,y)`

   qui donnent les solutions des deux problèmes considérés.
6. Tester les deux fonctions avec les données :

x_i	1	2	3	4	5	6	7	8	9	10
y_i	0	-3	6	-3	6	3.8	5	-2	1.4	8

   Afficher sur un même graphe les deux droites de régressions ainsi que les points (x_i, y_i).
7. Implémenter une fonction

   `x = gradient_proj(x0,eps,f,graf,t,projC)`

   qui implémente l'algorithme de gradient projeté avec un pas de descente t et un opérateur de projection sur le convexe est donné par `projC` (à définir par l'utilisateur pour chaque problème).

8. Vérifier que l'algorithme de gradient projeté appliqué au problème contraint converge bien vers la solution donnée par la fonction `[a,b] = regression_lin_bpos(x,y)`.

Exercice 2.

On a vu que l'algorithme de descente de gradient à pas optimal permettait de résoudre de manière efficace un système de la forme

$$Ax = b$$

dans le cas où A est une matrice carrée définie positive en minimisant la fonctionnelle quadratique de matrice A et de vecteur b.

Le but de l'exercice est de montrer que l'on peut également utiliser cet algorithme pour résoudre le problème des moindres carrés. On considère donc A une matrice rectangulaire de $\mathcal{M}_{m,n}(\mathbb{R})$ et $b \in \mathbb{R}^m$, et on définit $f : \mathbb{R}^n \to \mathbb{R}$ par

$$f(x) = \frac{1}{2}\|Ax - b\|^2.$$

1. Montrer que la fonction f est quadratique et préciser le matrice A', le vecteur b' et la constante c' pour cette fonctionnelle.
2. Sous quelle condition sur A l'algorithme du gradient à pas optimal converge-t-il pour cette fonctionnelle f ?

Dans toute la suite de l'exercice on supposera que A est de rang n (et donc que $m \geqslant n$).

3. Dans le cas $m = n$, cette hypothèse revient à supposer $A \in GL_n(\mathbb{R})$ (c'est-à-dire A est une matrice inversible). Justifier que minimiser f revient à résoudre le système linéaire $Ax = b$. On peut donc résoudre tous les systèmes linéaires carrés en minimisant f.

Les problèmes de moindres carrés sont très souvent rencontrés en statistique et en traitement du signal. Cette question propose un cadre pour tester l'algorithme du gradient conjugué pour résoudre le problème des moindres carrés.

On se place sur l'intervalle $[0, 2\pi]$. On suppose que l'on dispose de m mesures $y_i \in \mathbb{R}$ prises en chaque point $t_i \in [0, 2\pi]$ placés aléatoirement. Ces mesures correspondent aux valeurs d'une fonctions $g(t_i)$ et sont corrompus par un bruit gaussien i.i.d. de loi $\mathcal{N}(0, \sigma^2)$.

$$y_i = g(t_i) + W_i, \quad \text{avec } W_i \simeq \mathcal{N}(0, \sigma^2)$$

On cherche alors à approcher la fonction g par un polynôme trigonométrique de degré inférieur ou égal à d

$$p(t) = a_0 + \sum_{k=1}^{d} a_k \cos(kt) + \sum_{k=1}^{d} b_k \sin(kt).$$

Pour déterminer les $n = 2d + 1$ coefficients $a_0, a_1, \dots, a_d, b_1, \dots, b_d$ on résout le problème des moindres carrés

$$\min \frac{1}{2}\|Ax - b\|^2$$

avec

$$b = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{R}^m, \quad x = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \\ b_1 \\ \vdots \\ b_d \end{pmatrix}, \quad A = \left(\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \quad C \quad S \right)$$

où

$$C = (C_{i,k})_{1\leqslant i\leqslant m, 1\leqslant k\leqslant n} = \big(\cos(kt_i)\big)$$

et

$$S = (S_{i,k})_{1\leqslant i\leqslant m, 1\leqslant k\leqslant n} = \big(\sin(kt_i)\big)$$

Ce problème peut s'interpréter comme un maximum de vraisemblance (car le bruit est gaussien).

4. Tester le script ci-dessous qui résout le problème des moindres carrés pour l'approximation de mesures bruités de la fonction $g(t) = t(2\pi - t)$ par un polynôme trigonométrique. Vérifier que l'algorithme du gradient à pas optimal appliqué à la fonctionnelle $f(x) = \frac{1}{2}\|Ax - b\|^2$ résout convenablement le problème. Tester plusieurs valeurs pour m et pour l'écart-type σ. Qu'observe-t-on pour $n = m$? Est-ce toujours intéressant de prendre m grand ?

```
function [Xk, FXk]=desc_gradient_fquadra(x0,eps,f,gradf,A)
    x = x0;
    Xk = x;
    FXk = f(x);
    gfx = gradf(x);
    normgfxsq = gfx'*gfx;
    k = 0;
    while(normgfxsq >  eps^2 & k < 10000)
        t = normgfxsq/((A*gfx)'*gfx);
        x = x - t*gfx;
        k = k+1;
        Xk = [Xk, x];
        FXk = [FXk, f(x)];
        gfx = gradf(x);
        normgfxsq = gfx'*gfx;
    end
endfunction

// simulation des mesures bruitees
m = 41; // m impaire
sigma = 0.5;
t = gsort(2*%pi*rand(m,1),'g','i');  // lieu des mesures : tri croissant
b = t.*(2*%pi-t) + sigma*grand(m, 1, "nor", 0, 1);

// Exemple : Interpolation par un polynÃt'me trigonometrique
d = 5;
n = 2*d+1; // n impaire

A = zeros(m,n);
A(:,1) = ones(m,1);
for k=1:d
    A(:,k+1) = cos(k*t);
```

```
    A(:,k+1+d) = sin(k*t);
end

Ap = A'*A;
bp = A'*b;

function y = f(x)
    y = 0.5*sum((Ap*x).*x,1) - bp'*x;
endfunction
function gfx = gradf(x)
    gfx = Ap*x -bp;
endfunction


// Resolution avec scilab :
// A\b fait crasher scilab pour n=100...
xstar = linsolve(Ap,-bp); // linsolve computes all the solutions to A*x+b

// Resolution avec escente gradient pas optimal :
x0 = zeros(bp);
eps = 10^(-5);
[Xk, FXk]=desc_gradient_fquadra(x0,eps,f,gradf,Ap)
x = Xk(:,$);
tt = linspace(0,2*%pi,10^3)';
N = length(tt);
D = zeros(N,n);
D(:,1) = ones(N,1);
for k=1:d
    D(:,k+1) = cos(k*tt);
    D(:,k+1+d) = sin(k*tt);
end
polstar = D*xstar;
pol = D*x;

// affichage
scf(1); clf()
plot(t,b,'r*');
plot(tt,polstar,'-m');
plot(tt,pol(:,$),'-.b');
```