Méthode de Kacmarz

Pierron Théo

1^er juin 2014

On se place dans $\mathbb{R}^N$ muni de sa structure d'ev euclidien canonique. Soit $A \in GL_N(\mathbb{R})$ et b un vecteur colonne de taille N. On présente ici une méthode d'approximation itérative de l'unique $\overline{x}$ solution de $AX = B$.

Soient $a_1, \dots, a_N$ des vecteurs tels que a_i^T soit la i-ème ligne de A. On note aussi

$$u_i = \frac{a_i}{\|a_i\|} \text{ et } \beta_i = \frac{b_i}{\|a_i\|}$$

Notons finalement H_i l'hyperplan

$$\{z, u_i^T z = \beta_i\} = \beta_i u_i + \mathrm{Vect}\,\{u_i\}^\perp$$

On remarque d'abord le lemme suivant :

Lemme 1
$\{\overline{x}\} = \bigcap_{i=1}^{n} H_i$.

Démonstration.

$$z \in \bigcap_{i=1}^{n} H_i \quad \text{ssi} \quad \forall i, u_i^T z = \beta_i \quad \text{ssi} \quad \forall i, a_i^T z = b_i \quad \text{ssi} \quad Az = b \quad \text{ssi} \quad z = \overline{x} \qquad \blacksquare$$

La méthode de Kacmarz consiste, étant donné x_0 le point initial, à :

- projeter x_0 sur H_1 pour obtenir x_1
- projeter x_1 sur H_2 pour obtenir x_2
- ...
- projeter x_{N-1} sur H_N pour obtenir x_N
- recommencer à partir de x_N

faire un dessin !

Vu qu'il y a des projections partout, on va avoir besoin du lemme suivant :

Lemme 2
Soit $u \in \mathbb{R}^N$ de norme 1. La projection orthogonale sur $\mathrm{Vect}\,\{u\}^\perp$ a pour matrice $M := I - uu^T$.

De plus, $\|M\| = 1$ et pour tout $x \notin \mathrm{Vect}\,\{u\}^\perp$, $\|Mx\| < \|x\|$.

Démonstration.

- Si $x \in \mathrm{Vect}\,\{u\}^\perp$, alors
$$(I - uu^T)(x) = x - u\langle u, x\rangle = x$$
Si $x \in \mathrm{Vect}\,\{u\}$, alors $x = \lambda u$ donc
$$(I - uu^T)(x) = \lambda u - \lambda u \langle u, u\rangle = 0$$
Donc M est bien la matrice de la projection orthogonale sur $\mathrm{Vect}\,\{u\}^\perp$.

- Pour tout x, Mx est orthogonal à $(I-M)x$ donc par Pythagore :

$$\|x\|^2 = \|Mx\|^2 + \|(I-M)x\|^2 \geqslant \|Mx\|^2$$

D'où $\|M\| \leqslant 1$. On remarque aussi que si $x \notin \text{Vect}\,\{u\}^\perp$, $\|(I-M)x\| \neq 0$ donc l'inégalité est stricte.
Si $x \in \text{Vect}\,\{u\}^\perp$, $\|Mx\| = \|x\|$ donc $\|M\| = 1$. ■

Notons alors M_i la matrice de projection sur $\text{Vect}\,\{u_i\}^\perp$ et $T = M_N \dots M_1$. La suite $(x_n)_n$ vérifie alors

$$x_{n+1} = M_{(n+1) \mod N} x_n + \beta_i u_i$$

Notons $\varepsilon_n = x_n - \overline{x}$ l'erreur. Le but est de prouver que $\|\varepsilon_n\|$ tend vers 0. On remarque d'abord que

$$\varepsilon_{n+1} = M_{(n+1) \mod N} \varepsilon_n$$

donc, comme $\|M_i\| = 1$, on a $\|\varepsilon_{n+1}\| \leqslant \|\varepsilon_n\|$. On a donc affaire à une suite décroissante minorée qui converge donc vers un réel l. De plus,

$$\|\varepsilon_{kN}\| = \|T\|^k \|\varepsilon_0\|$$

donc on aura gagné dès qu'on saura que $\|T\| < 1$. Soit $x \in \mathbb{R}^N$, $x \neq 0$. On veut montrer que $\|Tx\| < \|x\|$. On a deux cas :

- il existe i tel que $M_i \dots M_1 x \notin \text{Vect}\,\{u_{i+1}\}^\perp$. Par le lemme, on sait que

$$\|M_{i+1}(M_i \dots M_1 x)\| < \|M_i \dots M_1 x\| \leqslant \|x\|$$

Alors

$$\|Tx\| \leqslant \|M_N \dots M_{i+2}\| \, \|M_{i+1}(M_i \dots M_1 x)\| < \|x\|$$

et c'est gagné.

- pour tout i, $M_i \dots M_1 x \in \text{Vect}\,\{u_{i+1}\}^\perp$. Alors $\langle x, u_1 \rangle = 0$ donc $M_1 x = x$ et $a_1^T x = 0$.

$$0 = \langle M_1 x, u_2 \rangle = \langle x, u_2 \rangle$$

donc $M_2 x = x$ et $a_2^T x = 0$. Par récurrence, on a donc $M_i x = x$ et $a_i^T x = 0$ pour tout i. D'où $Ax = 0$ donc $x = 0$, ce qui est absurde.

Finalement, $\|T\| < 1$ donc on a bien convergence (exponentielle !) de la méthode.

À chaque itération, on calcule

$$x_{n+1} = M_{(n+1) \mod N} x_n + \beta_i u_i$$

On doit donc faire N^2 multiplications et N soustractions pour calculer $M_{(n+1) \mod N}$. La multiplication par x_n coûte alors $N(N-1)$ additions et N^2 multiplications. On fait N multiplications pour calculer $\beta_i u_i$ et N additions pour x_{n+1}. On fait donc $3N^2 + 2N$ opérations.

En précalculant les M_i et les $\beta_i u_i$, on a un prétraitement en $N^3 + 2N^2$ mais chaque itération coûte alors $2N^2$ opérations.