École Polytechnique de Tunisie 2017/2018

PROJET DU MODULE TP OPTIMISATION

Classe : $1^{\text{ère}}$ année Y. Gati & W. Ben Romdhane

L'objectif de ce projet est d'appliquer les différentes méthodes d'optimisation pour résoudre des problèmes de régression et de voir quelle méthode est plus adaptée pour résoudre un problème donné. Ce projet est accompagné de données à utiliser avec les codes Matlab qui seront implémentés. Ces données sont contenues dans les fichiers `partie1.mat`, `partie2.mat` et `partie3.mat` qui peuvent être chargés dans Matlab en utilisant la fonction `load`.

1 Optimisation sans contrainte : La régression linéaire simple

1.1 Étude du problème

Étant donné un nuage de points $(x_i, y_i)_{i=1,\ldots,n}$. On voudrait faire passer une droite "le plus proche possible d'un maximum de points de ce nuage" (voir figure ci-dessous). Soit $D : y = a_1 x + a_0$ l'équation de cette droite. Soit $\hat{y}_i = a_1 x_i + a_0$ le point de D qui a pour abscisse x_i. Trouver cette droite revient à minimiser la fonction d'erreur suivante :

$$E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

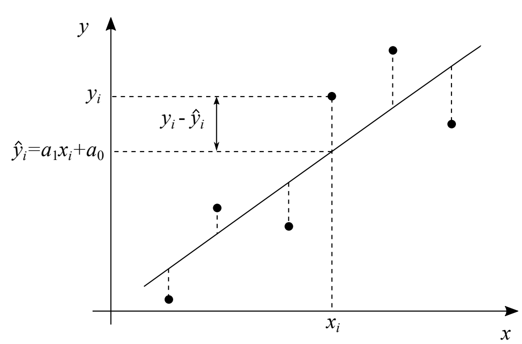

FIGURE 1 – Illustration d'une droite de régression linéaire

1. De quelles variables la fonction d'erreur dépend-elle ?
2. Montrer que la fonction d'erreur peut s'écrire sous la forme $E(a) = \|M\, a - m\|_2^2$ où a est un vecteur colonne contenant les variables de la fonction d'erreur et M et m sont respectivement une matrice et un vecteur que l'on précisera.
3. En utilisant les données fournies :

(a) Écrire une fonction Matlab qui calcule l'erreur $E(a)$ pour un vecteur a donné

(b) Visualiser cette fonction d'erreur sur Matlab, d'abord en 3D, puis en utilisant la fonction `contour`. Commenter.

(c) En utilisant les fonctions `gradient` et `quiver` de Matlab, calculer puis superposer le champ de gradient de la fonction d'erreur sur la représentation en contours. Commenter.

4. Montrer que minimiser E revient à minimiser une fonction $F(a) = \frac{1}{2}\langle Aa, a\rangle - \langle b, a\rangle$ où A et b sont respectivement une matrice et un vecteur que l'on précisera.

5. Calculer $\nabla F(a)$ et $\nabla^2 F(a)$

6. En quel point F est elle minimale ?

1.2 Algorithmes de résolution

1. Pour trouver le minimum de F numériquement, implémenter les méthodes suivantes :

(a) Méthode de Newton

On choisit a^0 au voisinage de la solution
Itérer pour $n \geq 0$: $a^{n+1} = a^n - \left[\nabla^2 F(a^n)\right]^{-1} \nabla F(a^n)$
S'arrêter si $\left|a^{n+1} - a^n\right| < \varepsilon$

(b) Méthode de gradient à pas constant

On choisit a^0 un vecteur de $\mathbb{R}^N$ et $\rho > 0$ un pas fixe.
Itérer pour $n \geq 0$: $a^{n+1} = a^n - \rho \nabla F(a^n)$
S'arrêter si $\left|a^{n+1} - a^n\right| < \varepsilon$

(c) Méthode de gradient à pas optimal

On choisit a^0 un vecteur de $\mathbb{R}^N$.
Itérer pour $n \geq 0$: $a^{n+1} = x^n - \rho_n \nabla F(a^n)$ avec $\rho_n = \dfrac{(r^n, r^n)}{(Ar^n, r^n)}$ et $r^n = Aa^n - b$

(d) Méthode du gradient conjugué

On choisit a^0 un vecteur de $\mathbb{R}^N$ et on pose $r^0 = Aa^0 - b, \quad d^0 = r^0$
Itérer pour $n \geq 0$: $a^{n+1} = a^n - \rho_n d^n$ avec $\rho_n = \dfrac{(r^n, r^n)}{(Ad^n, d^n)}$, et $r^n = Aa^n - b$

$$d^{n+1} = r^{n+1} + \beta_n d^n \quad \text{avec} \quad r^{n+1} = Aa^{n+1} - b \quad \text{et} \quad \beta_n = \frac{\left\|r^{n+1}\right\|^2}{\left\|r^n\right\|^2}$$

2. Comparaison des algorithmes : pour chacune des méthodes implémentés :

(a) Tracer la trajectoire des itérations sur la figure représentant les courbes de niveau. Refaire la même chose pour plusieurs points de départ.

(b) Tracer la courbe du nombre d'itération n nécessaire pour obtenir une précision ε donnée en fonction de $-\log_{10}(\varepsilon)$. Tracer plusieurs courbes en faisant varier le point de départ.

(c) Interpréter les résultats obtenus

3. Tracer le nuage de point et la droite de régression optimale sur un même graphique

2 Optimisation avec contrainte : la régression linéaire multiple

2.1 Étude du problème

Dans le cas précédent, nous avons exprimé la variable y comme fonction linéaire d'une seule variable. Lorsque y s'exprime comme fonction linéaire de plusieurs variables, on parle de régression linéaire multiple. La variable$y$ est alors estimée par l'expression suivante :

$$\hat{y} = a_0 + a_1 x_1 + a_2 x_2 + ... + a_n x_n \tag{1}$$

On dit que $x_1, x_2, \ldots, x_n$ sont les variables indépendantes (ou explicatives) et que y est la variable dépendante (ou à expliquer). On note :

— $X = [x_0, x_1, x_2, \ldots, x_n]$ la matrice des données relatives aux variables explicatives contenant $n + 1$ vecteurs colonnes $[x_j] = [x_{i,j}]$. Remarquer que X commence avec un vecteur x_0 qui n'apparait pas dans l'équation 1. C'est un vecteur colonne rempli de 1.

— $y = [y_i]$ le vecteur colonne contenant les observations relatives à la variable à expliquer

— $a = [a_0, a_1, a_2, \ldots, a_n]^T$ le vecteur colonne contenant les coefficients

1. Écrire $\hat{y}$ en fonction de X et de a
2. Écrire la fonction d'erreur quadratique $E(a)$. Calculer son gradient et sa hessienne et montrer qu'elle est convexe.
3. Pour des observations données X et y, déterminer la formule qui permet d'obtenir les paramètres qui minimisent l'erreur
4. Utiliser Matlab pour calculer ces paramètres avec la formule de la question précédente. Que remarquez-vous ?

2.2 Méthode de pénalisation

Le problème constaté lors de la question précédente est appelé "explosion des coefficients". Pour éviter ce problème, on pénalise la norme du vecteur a en ajoutant un terme à la fonction d'erreur. Le nouveau problème d'optimisation s'écrit alors :

$$\min_a \|Xa - y\|_2^2 + \lambda \|a\|_2^2 \tag{2}$$

où λ est un réel positif.

1. Expliquer comment la pénalisation permet d'éviter l'explosion des coefficients
2. Parmi les méthodes utilisées dans la première partie, laquelle est la mieux adaptée pour résoudre le problème (2) ? Écrire les fonctions Matlab qui permettent de l'implémenter

3. Tracer la variation des valeurs des coefficients en fonction de λ. Commenter.
4. Soit $\hat{y}_{opt}(\lambda)$ l'estimation de y obtenue en utilisant les paramètres optimaux pour une valeur donnée de λ. Tracer l'évolution des variables y et $\hat{y}_{opt}(\lambda)$ sur un même graphique en faisant varier la valeur de λ.

3 Méthode de gradient accélérés

3.1 Système du premier ordre

Un système dynamique linéaire du premier ordre est un système dont la réponse indicielle est du type : $y(t) = a\left(1 - e^{bt}\right)$. L'identification des coefficients a et b est facile en théorie. Mais souvent, dans la pratique, les mesures sont discrètes (réalisées à des instants t_i, $i = 1, \ldots, n$) et sont entachées d'erreurs. L'identification des coefficients se fait alors en minimisant le logarithme de l'erreur quadratique (voir figure) :

$$E = \log\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right)$$

avec :

y_i : la valeur mesurée de y à l'instant t_i

$\hat{y}_i$: la valeur estimée par le modèle $\hat{y}_i = a\left(1 - e^{bt_i}\right)$

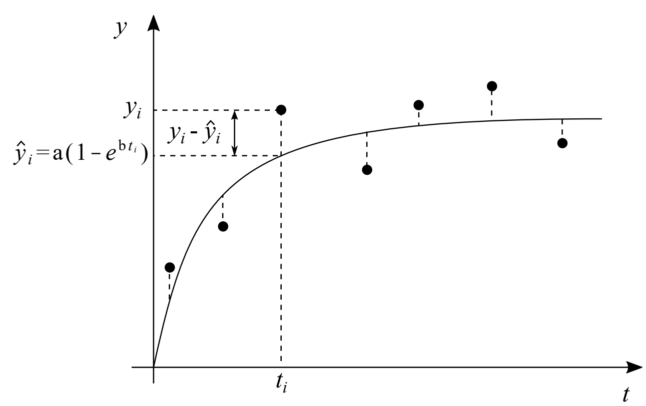

FIGURE 2 – Illustration de la réponse indicielle d'un système linéaire du premier ordre

1. Calculer le gradient de la fonction d'erreur
2. Écrire une fonction `erreur.m` qui calcule l'erreur E pour des paramètres données et une fonction `grad_erreur.m` qui calcule son gradient
3. Avec les données fournies, visualiser cette fonction d'erreur sur Matlab, d'abord en 3D, puis en utilisant la fonction `contour`
4. Cette fonction est-elle convexe ?
5. Pour trouver les paramètres qui minimisent la fonction d'erreur, nous allons utiliser des méthodes qui font appel au gradient uniquement car la hessienne est difficile à calculer. La méthode de

Newton et celle du gradient à pas optimal sont donc à éliminer. Les algorithmes à implémenter sont les suivants :

(a) Méthode de gradient à pas fixe

$$\left|\begin{array}{l}\text{On choisit } x^0 \in \mathbb{R}^2 \text{ et } \rho > 0 \text{ un pas fixe} \\ \text{Itérer pour } n \geq 0, \quad x^{n+1} = x^n - \rho \nabla E\left(x^n\right) \\ \text{S'arrêter si } \left|x^{n+1} - x^n\right| < \varepsilon\end{array}\right.$$

(b) Méthode d'inertie

$$\left|\begin{array}{l}\text{On choisit } x^0, v^0 \in \mathbb{R}^2 \text{ et } \eta, \rho \in \mathbb{R} \\ \text{Itérer pour } n \geq 0\ :\ v^{n+1} = \eta v^n + \rho \nabla E\left(x^n\right) \text{ et } x^{n+1} = x^n - v^{n+1} \\ \text{S'arrêter si } \left|x^{n+1} - x^n\right| < \varepsilon\end{array}\right.$$

(c) Méthode du gradient accéléré de Nesterov

$$\left|\begin{array}{l}\text{On choisit } x^0, v^0 \in \mathbb{R}^2 \text{ et } \eta, \rho \in \mathbb{R} \\ \text{Itérer pour } n \geq 0\ :\ v^{n+1} = \eta v^n + \rho \nabla E\left(x^n - \eta v^n\right) \text{ et } x^{n+1} = x^n - v^{n+1} \\ \text{S'arrêter si } \left|x^{n+1} - x^n\right| < \varepsilon\end{array}\right.$$

6. Pour le point de départ $x^0 = [1; 0]$,

   (a) Chercher des paramètres qui permettent à chacun des algorithmes de converger

   (b) Pour chacun des algorithmes implémentés :

      i. Tracer la trajectoire des itérations sur la figure représentant les courbes de niveau.

      ii. Tracer la courbe du nombre d'itérations n nécessaire pour obtenir une précision ε donnée en fonction de $-\log_{10}(\varepsilon)$. Commenter.

7. Pour le point de départ $x^0 = [-2; -1]$, $\rho = 0,0004$, $\eta = 0.09$ et $\varepsilon = 10^{-6}$

   (a) Déterminer l'algorithme le plus rapide

   (b) Interpréter le comportement de chacun des algorithmes au voisinage de la solution (utiliser le zoom)

   (c) En utilisant la méthode la mieux adaptée, créer une animation qui affiche le nuage de points et la courbe approchée à chaque itération de l'algorithme d'optimisation.