DESCAMPS Simon

Tuteur : François Boulier

Rapport Final :
Tutorat Calcul Numérique

Année universitaire 2018-2019

Sommaire :

1. Introduction
2. Cahier des charges
3. La fonction logistique
4. La méthode des carrés linéaires
5. Résolution par la méthode des carrés linéaires
6. La méthode des carrés non linéaires
7. Résolution par la méthode des carrés non linéaires
8. Les différents algorithmes et actions utilisés
9. Fonctionnement du programme
10. Application
11. Difficultés rencontrées
12. Pistes d'amélioration et ressenti personnel

1. Introduction:

Le but du projet est d'écrire en FORTRAN un programme qui permet de lire les coordonnées de m points formant une courbe sigmoïdale et qui calcule par la suite les valeurs de paramètres grâce à différentes méthodes. Pour cela, nous utiliserons deux méthodes : La méthode des carrés linéaires et la méthode des carrés non linéaires.

2. Cahier des charges

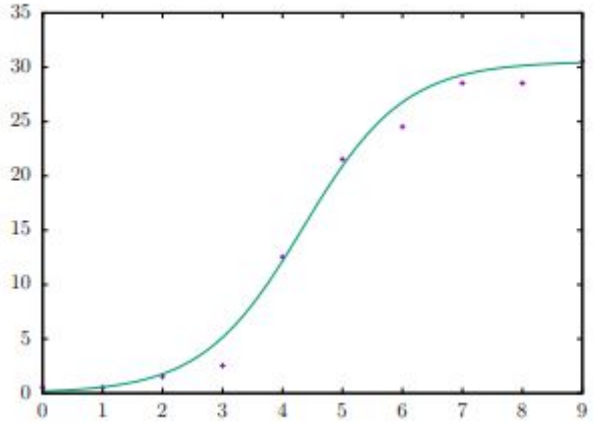
Nous disposons d'un jeu de données comportant m = 10 mesures. Ces mesures correspondent à la quantité d'eau présente dans les oeufs de Locustana pardalina en formation à une température de 35 degrés.
On cherche à estimer les paramètres de la fonction logistique à l'aide du jeu de données ci-dessous :

nb. jours x_i	0	1	2	3	4	5	6	7	8	9
qté eau y_i	.53	.53	1.53	2.53	12.53	21.53	24.53	28.53	28.53	30.53

3. La fonction logistique:

La fonction logistique est la suivante : $$y(x) = \frac{\kappa}{1 + e^{\alpha - \rho x}}.$$

La variable x représentant le temps, et les trois lettres grecques κ, α, ρ des paramètres. La courbe représentative de la fonction a la forme d'une sigmoïde ayant pour asymptotes horizontales les droites y = 0 et y = κ. Ci-dessous, on peut observer une représentation graphique de la fonction logistique.



4. La méthode des carrés linéaires:

En introduisant la fonction logit définie par : $\text{logit}(p) = \ln\left(\frac{p}{1-p}\right)$

On trouve à partir de la fonction logistique logit(y/κ) = ρx - α
En sachant que κ correspond à l'asymptote horizontale vers laquelle tend la fonction logit, on peut supposer κ connu. On prendra dans ce cas κ = 30.54

Par application, on en revient à vouloir résoudre le système suivant :

$$\begin{pmatrix} 0 & -1 \\ 1 & -1 \\ \vdots & \vdots \\ 9 & -1 \end{pmatrix} \begin{pmatrix} \rho \\ \alpha \end{pmatrix} = \begin{pmatrix} logit(\frac{y(0)}{k}) \\ logit(\frac{y(1)}{k}) \\ \vdots \\ logit(\frac{y(9)}{k}) \end{pmatrix}$$

5. Résolution par la méthode des carrés linéaires:

Nous avons défini en cours que nous pouvions effectuer une factorisation de Cholesky afin de résoudre facilement le système.
Dans notre cas, on a la matrice de départ que l'on nommera A avec dim(A) = (mx2). Et on veut appliquer l'algorithme de Cholesky prenant en entrée la matrice A et donnant en sortie la matrice L triangulaire inférieure telle que :

$$LL^t = A$$

Une fois la matrice L obtenue, et en notant :

$$A = \begin{pmatrix} 0 & -1 \\ 1 & -1 \\ \vdots & \vdots \\ 9 & -1 \end{pmatrix}, x = \begin{pmatrix} \rho \\ \alpha \end{pmatrix}, b = \begin{pmatrix} logit(\frac{y(0)}{k}) \\ logit(\frac{y(1)}{k}) \\ \vdots \\ logit(\frac{y(0)}{k}) \end{pmatrix}$$

Il ne reste donc qu'à résoudre le système suivant :

$$Ax = b \Leftrightarrow LL^t = A^t b$$

En décomposant, on obtient le système suivant à résoudre :

$$\begin{cases} Lv = A^t b \\ L^t x = v \end{cases}$$

La première équation du système nous permettant de fournir v en faisant une descente sur la matrice triangulaire inférieure grâce à un algorithme.
La seconde équation nous permet de fournir x sachant v connu. Pour résoudre ce système, on effectue une remontée puisque la transposée d'une matrice triangulaire inférieure est une matrice triangulaire supérieure. Il s'agit d'un algorithme similaire à celui de la descente.

Le vecteur x fourni correspond à l'estimation des paramètres ρ et α.

6. La méthode des carrés non linéaires:

La méthode des carrés non linéaires est une variante de la méthode de Gauss-Newton. Celle-ci est plus sophistiquée que la méthode des carrés linéaires mais est plus optimale. On cherche à construire une suite de vecteurs

$$v_i = \begin{pmatrix} \kappa_i \\ \alpha_i \\ \rho_i \end{pmatrix}$$

qui converge vers des valeurs qui minimisent la somme des carrés des écarts verticaux entre la courbe et les points.
On introduit pour cela la fonction sigmoïde suivante :

$$s(\kappa, \alpha, \rho, x) = \frac{\kappa}{1 + e^{\alpha - \rho x}}$$

Et on considère la fonction résidu suivante :

$$r(\kappa, \alpha, \rho) = \begin{pmatrix} s(\kappa, \alpha, \rho, x_1) - y_1 \\ s(\kappa, \alpha, \rho, x_2) - y_2 \\ \vdots \\ s(\kappa, \alpha, \rho, x_m) - y_m \end{pmatrix}.$$

Finalement, nous notons la matrice J correspondant à la matrice Jacobienne de r.
En nous plaçant à une itération l, la méthode de Gauss-Newton consiste à construire le vecteur r et la matrice J et à résoudre le système d'équation suivant :

$$(J^T J)\, w = -J^T r$$

Ce vecteur correspond à une estimation de la variation des paramètres corrigeant l'écart entre les points de la sigmoïde et les points expérimentaux. Cela nous permet de construire le vecteur :

$$v_{\ell+1} = v_\ell + w\,.$$

7. Résolution par la méthode des carrés non linéaires:

Pour obtenir une estimation de nos paramètres par cette méthode, il est nécessaire d'entrer le vecteur initial. Ensuite, nous calculons itérativement les m itérations. Pour chaque itération, on calcule alors la matrice jacobienne et le vecteur résidu afin de calculer w.

Comme expliqué précédemment, on cherche à résoudre le système suivant afin d'obtenir w à chaque itération :

$$(J^T J)\, w \;=\; -J^T r$$

Le produit $J^t J$ donnant une matrice symétrique définie positive, on peut donc suivre la même méthode que pour les carrés linéaires, c'est à dire en effectuant une factorisation de Cholesky puis en effectuant une descente et une remontée. Une fois la dernière itération effectuée, on peut donc fournir le dernier vecteur v correspondant théoriquement à la plus proche estimation des paramètres de notre suite de vecteurs.

8. Les différents algorithmes et actions utilisés:

CALCUL_ATB : Action permettant de calculer un vecteur de taille N correspondant au produit d'une matrice A(MxN) et d'un vecteur B de taille M l'ajout du paramètre scal permet de multiplier le résultat par un scalaire, ce qui est nécessaire pour calculer $-J^t r$.

PROD_AT_A : Action permettant de calculer la matrice $A^t A$.

SIGMOIDE : Action permettant de donner le résultat de la fonction sigmoïde appliquée aux paramètres k, alpha et p.

DESCENTE : Permet de fournir V, un vecteur de taille N en résolvant le système LV = ATB. En appliquant une descente sur L une matrice triangulaire inférieure.

MONTEE : Permet de fournir X en appliquant une montée sur une matrice triangulaire supérieure. Le vecteur X correspond à une estimation de nos paramètres.

LOGIT : Action permettant de calculer le logit de chaque éléments d'un vecteur.

CHOLESKY : Action permettant de construire une matrice triangulaire inférieure à partir d'une matrice définie positive symétrique.

BUILD_W, BUILD_R, BUILD_JACOB : Actions permettant de construire respectivement les vecteurs w, r et la matrice jacobienne du vecteur r.

METH_CARRE_NLIN et METH_CARRE_LIN : Actions permettant d'appliquer respectivement la méthode des carrés non linéaires et la méthode des carrées linéaires aux données passées dans le programme.

Voici en exemple le pseudo-code de l'algorithme descente :

```
Action DESCENTE ( N, L, ATB, V, LDA)
   Données : N, LDA Entiers
                    ATB (LDA) vecteur
                    L (LDA,*) matrice
   Variables Locales
       I,J : Entiers
       somme : Réel
   Donnée Résultat :
       V ( LDA ) vecteur

   Pour I de 1 à N Faire :
     somme <- 0
     Pour J de 1 à I-1 Faire
        somme <- somme + V[J]*L[I][J]
     Fin Pour
     V[I} <- ( ATB[I] - somme )/L[I][I]
   Fin Pour

Fin Action
```

9. Fonctionnement du programme :

Lors de l'exécution du programme, il vous sera dans un premier lieu demandé d'entrer une matrice A. Cette matrice correspond à une matrice à m lignes et 2 colonnes, les lignes de la première colonne sont l'indice de la variable dont chaque observation dépend (Dans notre cas il s'agit du temps). La deuxième colonne est quand à elle une colonne remplie de -1.

$$\begin{pmatrix} 0 & -1 \\ 1 & -1 \\ \vdots & \vdots \\ 9 & -1 \end{pmatrix}$$

Il vous sera ensuite demandé d'entrer un vecteur B. Ce vecteur correspond au résultat de chaque observation.

Une fois ces deux données rentrées, le programme effectuera les étapes suivantes afin d'appliquer la méthode des carrés linéaires:

- Calcul du logit de chaque élément du vecteur B
- Calcul de A^tA
- Factorisation de Cholesky de la matrice A^tA
- Calcul du vecteur A^tb
- Descente pour résoudre $Lv = A^tb$
- Montée pour résoudre $L^tx = v$
- Affichage du vecteur v correspondant aux paramètres.

Le programme demande ensuite d'entrer le vecteur initial v_0 correspondant à

$$v_0 = \begin{pmatrix} \kappa \\ \alpha \\ \rho \end{pmatrix}$$

Une fois cela fait, le programme calculera les m itérations du vecteur v en effectuant à chaque étape :

- Calcul du vecteur résidu
- Calcul de la matrice Jacobienne
- Calcul du vecteur w
- Calcul du vecteur V pour l'étape suivante

Le programme se termine par l'affichage de la dernière itération du vecteur v, correspondant à l'estimation des paramètres κ, α, ρ.

10. Application :

Nous testons à présent le programme en passant en entrée le fichier donnees.dat à l'aide de la commande $./a.out < donnees.dat$.
On prend donc pour départ $\kappa = 30.54$.

On obtient par la méthode des carrés linéaires l'estimation suivante :
.

$$v = \begin{pmatrix} \rho = 1.188 \\ \alpha = 5.163 \end{pmatrix}$$

Les résultats correspondent bien aux résultats présentés dans l'énoncé.

Pour la méthode des carrés non linéaires, le programme affiche à chaque itération la Jacobienne du résidu ainsi que le vecteur w. En prenant au départ le même κ et les estimations de α et ρ. On devrait obtenir les mêmes résultats que dans l'énoncé.

Voici le contenu du fichier donnees.dat :
A := <<0 | -1>, <1 | -1>, <2 | -1>, <3 | -1>, <4 | -1>, <5 | -1>, <6 | -1>, <7 | -1>, <8 | -1>, <9 | -1>>
B := <<0.53>,<0.53>,<1.53>,<2.53>,<12.53>,<21.53>,<24.53>,<28.53>,<28.53>,<30.53>>
Vn := <<30.54>,<5.163>,<1.188>>
Si vous souhaitez le modifier, les matrices et vecteurs sont entrées lignes par lignes entre les < >. Par exemple : A := < <element_ligne1_1 | element_ligne1_2 > , < …| …> >.

11. Difficultés rencontrées:

N'ayant pas trouvé de binôme avec qui je m'entends, j'ai choisi de travailler seul pour ne pas mâcher le travail de quelqu'un d'autre comme cela a pu être le cas dans un autre projet. Cela m'a bien entendu porté préjudice puisque j'ai été seul à travailler sur le code FORTRAN et surtout seul pour produire le rapport dans des délais limités. En effet, mon stage a commencé aussitôt l'année scolaire terminée. Cela implique donc que je travaille la semaine, ce à quoi j'ajoute une activité de sportif de niveau national. Il est donc évident que je n'ai pas eu beaucoup de temps pour travailler sur le rapport ni pour améliorer le code.
Finalement, n'ayant pas accès à un compilateur, il est très difficile de compléter la partie utilisation qui devrait normalement expliquer comment compiler, exécuter et passer un fichier en entrée. Cette même partie devrait également montrer la sortie du programme pour un exemple avec des captures d'écran.

12. Pistes d'amélioration et ressenti personnel;

Bien que le projet soit très intéressant, il a été difficile de gérer les différents projets qui nous ont été données en cette fin d'année ainsi que les examens. Je pense personnellement que si le projet nous avait été donné plus tôt nous aurions pu passer plus de temps en dehors des cours. De plus, pour la raison expliquée précédemment, je ne peux me rendre à Polytech pour pouvoir avancer sur le code et je ne pense pas être le seul dans cette situation.

Également, nous avons découvert Fortran lors de ce projet. Ne connaissant pas bien le fonctionnement, j'ai commis beaucoup d'erreurs dont je me suis rendu compte plus tard. En outre, dans la plupart des subroutines que j'ai créées, j'ai passé en paramètres inutilement des variables car ce sont des variables locales. Cela rend le code beaucoup

moins esthétique mais également beaucoup moins compréhensible. J'aurais eu besoin d'avantage de temps pour pouvoir installer un compilateur sur mon ordinateur et ainsi corriger ces erreurs.

Finalement, j'ai codé la méthode des carrés non linéaires mais n'ai pas pu tester si elles fonctionnait correctement ou non pour les mêmes raisons. Je trouve cela regrettable car je ne suis pas capable de dire si mon projet a abouti à une solution ou non. Il est regrettable de ne pas voir si les efforts effectués ont porté leurs fruits ou non.

Pour ce qui est de la fiche tutorat, cela est à mon goût une bonne idée mais certaines choses seraient à perfectionner. En effet, j'avais personnellement complètement oublié cette fiche et ne l'ai ressortie que deux semaines avant la fin des cours. Il serait peut-être mieux si les professeurs nous rappelaient de remplir cette fiche lorsque leur cours traite un des éléments de celle-ci.