

J. P. ASSELIN DE BEAUVILLE

A. DOLLA

**Étude par simulation d'un estimateur de
Huber pour la protection de la régression
linéaire et polynomiale**

Les cahiers de l'analyse des données, tome 4, n° 2 (1979),
p. 147-158

http://www.numdam.org/item?id=CAD_1979__4_2_147_0

© Les cahiers de l'analyse des données, Dunod, 1979, tous droits réservés.

L'accès aux archives de la revue « Les cahiers de l'analyse des données » implique l'accord avec les conditions générales d'utilisation (<http://www.numdam.org/conditions>). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques

<http://www.numdam.org/>

ÉTUDE PAR SIMULATION D'UN ESTIMATEUR
DE HUBER POUR LA PROTECTION
DE LA RÉGRESSION LINÉAIRE ET POLYNÔMIALE
[RÉGRESSION ROBUSTE]

par J. P. Asselin de Beauville
et A. Dolla (1)

1 Introduction

Le modèle linéaire simple s'exprime par l'équation $Y_j = a + bx_j + E_j$. Le plus souvent, les coefficients de régression a et b sont estimés par la méthode des moindres carrés. On sait que, si les erreurs E_j sont normales ou si les conditions de Gauss-Markov sont remplies, cette méthode conduit aux meilleurs estimateurs linéaires non biaisés (B.L.U.E.) de a et b .

Lorsque les erreurs ne sont pas normales et plus particulièrement si la distribution des erreurs est asymétrique, ou si elle est plus étalée que la distribution normale, la méthode des moindres carrés donne alors de mauvais résultats. Pratiquement cela implique que les estimateurs des moindres carrés \hat{a} et \hat{b} sont très sensibles à la présence de points aberrants dans l'échantillon étudié. On peut montrer que si la loi des erreurs est normale contaminée par une loi normale d'écart-type supérieur, par exemple :

$$(1 - \epsilon)N(m, 1) + \epsilon N(m, 4), \quad 0 \leq \epsilon \leq 1$$

l'efficacité de l'estimateur \hat{b} n'est plus que de 0,363 pour $\epsilon = 0,25$.

Cette inefficacité de la méthode est d'ailleurs en liaison directe avec la faible robustesse de la moyenne arithmétique en tant qu'estimateur de position.

Pour pallier cet inconvénient de nombreuses méthodes de régression, plus ou moins robustes, ont été proposées. Le but recherché étant de définir une procédure d'estimation des paramètres a et b qui soit à la fois :

- peu sensible à la présence de points aberrants dans l'échantillon (grosses erreurs).
- relativement efficace pour une grande variété de distribution des erreurs.
- presque aussi efficaces que la méthode des moindres carrés lorsque les erreurs sont normales.

On trouvera une description détaillée de la plupart de ces procédures dans (1).

(1) Laboratoire d'Informatique Appliquée. U.E.R. A.G.I., Parc de Grandmont 37200 Tours.

Parmi les méthodes proposées, la famille des M-estimateurs de Huber a été très étudiée. Les coefficients a et b sont estimés en minimisant $\sum_{i=1}^n \rho((y_i - a - b x_i)/S)$ où $\rho(z)$ est une fonction généralement convexe et S une estimation de la dispersion des résidus de l'échantillon de taille n. De nombreuses fonctions ont été proposées pour $\rho(z)$, une des plus simples est celle de Huber :

$$\rho(z) = \begin{cases} z^2/2 & \text{pour } |z| < C \\ C|z| - C^2/2 & \text{pour } |z| \geq C. \end{cases}$$

On peut alors estimer a et b à partir du système d'équations :

$$\begin{aligned} \sum_{i=1}^n \psi[(y_i - a - b x_i)/S] x_i &= 0 \\ \text{et } \sum_{i=1}^n \psi[(y_i - a - b x_i)/S] &= 0 \\ \text{où } \psi(z) &= \max(-C, \min(C, z)). \end{aligned}$$

Ce système d'équations peut être résolu par un processus itératif plus ou moins sophistiqué.

Dans cette note on présente une solution "naïve" de ce système d'équations qui a l'avantage d'être très facilement programmable. (Nous avons programmé cette solution en FORTRAN : programme REGROB). On teste la robustesse de cet algorithme à l'aide d'échantillons artificiels et on compare les résultats obtenus avec ceux fournis par d'autres méthodes dans des conditions voisines. Pour terminer, on étudie par simulation, la robustesse de l'estimateur de Huber dans le cas du modèle polynomial :

$$Y_j = \delta_0 + \delta_1 x_j + \delta_2 x_j^2 + \dots + \delta_p x_j^p + E_j.$$

Les estimations des coefficients δ_i ($i = 0, 1, 2, \dots, p$) sont alors

$$\begin{aligned} \sum_{i=1}^n \psi[(y_i - \sum_{j=0}^p \delta_j x_i^j)/S] &= 0 \\ \sum_{i=1}^n \psi[(y_i - \sum_{j=0}^p \delta_j x_i^j)/S] x_i &= 0 \\ &\vdots \\ \sum_{i=1}^n \psi[(y_i - \sum_{j=0}^p \delta_j x_i^j)/S] x_i^p &= 0. \end{aligned}$$

Nous avons écrit un sous-programme en FORTRAN (sous-programme PULROB) pour effectuer ces calculs.

Les résultats concernant le modèle polynomial sont intéressants car il ne semble pas que l'emploi de l'estimateur de Huber pour ce type de modèle ait été très étudié jusqu'à ce jour.

Dans cette note on supposera qu'il n'y a pas d'erreur sur les coefficients x_j . Pour d'autres conditions relatives aux variables explicatives (en particulier lorsque, dans un modèle linéaire multidimensionnel ces variables sont très corrélées) on pourra consulter (2).

2 Les algorithmes

a) cas du modèle linéaire

Soit un échantillon (x_i, y_i) de taille n . Les différentes étapes de l'algorithme sont les suivantes :

1. $k = 1$

2. Calculer les estimations \hat{a}_k et \hat{b}_k des moindres carrés pour l'échantillon (x_i, y_i)

3. Calculer une estimation robuste S_k de l'écart-type des résidus.

4. Tester chaque résidu $d_i = y_i - \hat{a}_k - \hat{b}_k x_i$ par rapport à la quantité $h \cdot S_k$ où h est une constante positive donnée :

+ Si $d_i \geq 0$ et si $d_i > h S_k$ poser

$$y'_i = \hat{a}_k + \hat{b}_k x_i + h S_k.$$

+ Si $d_i < 0$ et si $|d_i| > h S_k$ poser

$$y'_i = \hat{a}_k + \hat{b}_k x_i - h S_k.$$

5. Si tous les d_i appartiennent à l'intervalle

aller en 6. $[\hat{a}_k + \hat{b}_k x_i - h S_k, \hat{a}_k + \hat{b}_k x_i + h S_k]$ poser $a^* = \hat{a}_k$ et $b^* = \hat{b}_k$ et

Si non :

+ Si $k = 1$ aller en 2.

+ Si $k > 1$, tester les variations relatives de la pente et de l'ordonnée à l'origine :

+ Si $|\hat{a}_k - \hat{a}_{k-1}| / |\hat{a}_{k-1}| \leq \alpha$ et si

$$|\hat{b}_k - \hat{b}_{k-1}| / |\hat{b}_{k-1}| \leq \alpha \text{ (où } \alpha \text{ est un nombre donné tel que}$$

$0 < \alpha < 1$) poser $a^* = \hat{a}_k$ et $b^* = \hat{b}_k$ et aller en 6.

+ Si non, faire $k = k + 1$ et aller en 2.

6. Arrêter la procédure.

Les estimations obtenues par cet algorithme sont notées a^* et b^* . On peut résumer cette méthode par le schéma suivant : on calcule les résidus d_i pour l'échantillon original (x_i, y_i) à partir de la droite des moindres carrés ($y = \hat{a} + \hat{b}x$) puis, ayant calculé S , on recalcule la droite des moindres carrés pour l'échantillon modifié (x_i, y'_i) où

$$y'_i = \begin{cases} \hat{a} + \hat{b}x_i + hS & \text{si } d_i > hS \\ y_i & \text{si } |d_i| \leq hS \\ \hat{a} + \hat{b}x_i - hS & \text{si } d_i < -hS \end{cases}$$

On continue le même processus jusqu'à ce que les critères d'arrêt définis à l'étape 5 ci-dessus soient vérifiés.

Dans la suite on a utilisé les valeurs $h = 2$ et $\alpha = 0,001$. $h = 2$ correspond à un choix qui tend à optimiser l'algorithme pour des erreurs normales. En effet, dans ce cas, la probabilité de trouver un résidu à l'extérieur d'un intervalle de largeur $4S$ centré sur la moyenne est égale à 4,56%. On verra dans la suite, que cette valeur conduit, en outre, à de bons résultats lorsque les erreurs ne sont pas normales. Les valeurs de h les plus utilisées sont : 1 ; 1,5 ; 2 et 2,1.

La valeur de α doit être choisie de façon à établir un compromis entre la précision recherchée et le temps de calcul. Il est clair qu'une faible valeur de α augmente le temps de calcul puisque le nombre d'itérations croît.

Le but recherché étant la robustesse de l'algorithme, l'estimateur S_k doit être robuste. Plusieurs propositions ont été étudiées.

D'abord :

$$S' = 2,1 \operatorname{med}\{|d_i|\} \quad \text{où } \operatorname{med}(x_i) = \text{médiane des } x_i.$$

À l'expérience il est apparu que S' pouvait présenter de brusques changements de valeur au cours des itérations. Pour remédier à cet inconvénient, Ramsay a employé :

$$S'' = (1/(n-4)) \sum_{i=3}^{n-2} |d_{(i)}| \quad \text{où } d_{(i)} \text{ est la } i^{\text{ème}} \text{ valeur de } d_i \text{ classé en ordre non décroissant.}$$

Bickel a proposé l'écart interquartile normalisé des résidus :

$$S''' = (1/1,35) (d_{0,75} - d_{0,25}) \quad \text{où } d_\beta \text{ est l'estimation obtenue, à partir des résidus classés en ordre non décroissant du percentile } \beta (0 < \beta < 1) \text{ de la distribution des erreurs.}$$

Après plusieurs essais, nous avons retenu l'estimateur robuste S_k de Fama et Roll, c'est-à-dire l'écart normalisé entre les percentiles 0,72 et 0,28 des résidus :

$$S_k = (1/1,166) (d_{0,72} - d_{0,28})$$

où d_β est l'estimation, obtenue à partir des résidus classés en ordre non décroissant pour la $k^{\text{ième}}$ étape, du percentile $\beta (0 < \beta < 1)$ de la distribution des erreurs. Cet estimateur présente un biais asymptotique inférieur à 0,004 ainsi qu'une distribution asymptotique normale.

Cet algorithme présente, à première vue, une certaine ressemblance avec la régression "Winsorisée"* (3). En effet dans cette dernière méthode, on ordonne les résidus $d_i (d_1 \leq d_2 \leq \dots \leq d_n)$ calculés à partir d'une première estimation de la droite de régression, puis on applique la méthode des moindres carrés à l'échantillon modifié (x_i, y'_i) où y'_i est défini par :

$$y'_i = y_i + d'_i \quad \text{avec}$$

$$d'_i = \begin{cases} d_{g+1} & \text{pour } i = 1, 2, \dots, g \\ d_i & \text{pour } i = g+1, \dots, n-g \\ d_{n-g} & \text{pour } i = n-g+1, \dots, n. \end{cases}$$

* Winsoriser une série statistique $x_i (i = 1, 2, \dots, n)$ c'est ordonner les x_i , puis redéfinir les valeurs extrêmes (soit les valeurs aberrantes potentielles) par les valeurs qui leur sont adjacentes.

g représente le nombre de points Winsorisés à chaque extrémité de l'échantillon. A partir de ce schéma général on trouve dans (3) la procédure itérative suivante : on calcule les résidus pour l'échantillon original (x_i, y_i) à partir de la droite des moindres carrés et on effectue la Winsorisation, puis on recommence le processus en calculant les d_i'' à partir de l'échantillon (x_i, y_i') pour la seconde itération, puis à partir de (x_i, y_i'') pour la troisième itération ($y_i'' = y_i' + d_i''$) et ainsi de suite jusqu'à ce que l'on ait atteint un nombre d'itérations (l) fixé à l'avance. La différence essentielle entre ces deux algorithmes réside dans le fait que, pour la Winsorisation, on fixe au préalable le nombre de points g à traiter à chaque extrémité de l'échantillon ainsi que le nombre l d'itérations alors que, pour l'algorithme étudié ici, ces deux nombres sont des variables aléatoires (puisque'ils dépendent étroitement du seuil $h.S_k$ qui est lui-même fonction de l'échantillon). D'autre part le nombre de points traités ne prend pas obligatoirement la même valeur à chaque extrémité de l'échantillon. Il est plus facile de comprendre cette différence à l'aide d'un exemple à une dimension. Soit l'échantillon x_i suivant :

1 ; 7 ; 7,1 ; 7,2 ; 7,5 ; 7,6 ; 8 ; 8,5 ; 22 ; 24 pour lequel on trouve :

$$\bar{x} = (1/10) \sum_{i=1}^{10} x_i \approx 10$$

$$S_1 = (1/1,166) (x_{0,72} - x_{0,28}) \approx 1,2.$$

Si on Winsorise une fois cet échantillon avec $g = 2$ on trouve :

7,1 ; 7,1 ; 7,1 ; 7,2 ; 7,5 ; 7,6 ; 8 ; 8,5 ; 8,5 ; 8,5.

L'application de notre algorithme avec $h = 2$ et $k = 1$ conduit à :

7,6 ; 7,6 ; 7,6, 7,6 ; 7,6, 7,6 ; 8 ; 8,5 ; 12,4 ; 12,4.

On constate que cette dernière méthode change les valeurs de cinq points à l'extrémité inférieure et de 2 points seulement à l'extrémité supérieure de l'échantillon alors que la Winsorisation modifie le même nombre de points aux deux extrémités. Si on Winsorise une nouvelle fois ($l = 2$) l'échantillon obtenu avec $g = 2$ on ne change plus rien au résultat. Par contre si on applique une nouvelle fois notre algorithme ($k = 2$) on obtient :

7,2 ; 7,2 ; 7,2 ; 7,2 ; 7,2 ; 7,2 ; 8 ; 8,5 ; 10,2 ; 10,2.

D'une façon générale les deux algorithmes ne conduiront donc pas au même résultat et leur ressemblance n'est qu'apparente.

Tukey (4) a proposé un algorithme, voisin de celui de Huber, relatif à l'estimation robuste de la position d'une distribution ("*skipped estimates*") : on élimine de l'échantillon toutes les observations x_i situées à l'extérieur de l'intervalle C_1, C_2 où

$$C_1 = v_1 - \beta(v_2 - v_1)$$

$$C_2 = v_2 + \beta(v_2 - v_1)$$

avec v_2 et v_1 = quartiles supérieur et inférieur des x_i .

β étant un coefficient, fixé à l'avance, pouvant prendre par exemple les valeurs 1 ; 1,5 ou 2. Ce processus peut être répété jusqu'à ce qu'on ne trouve plus de points à l'extérieur de l'intervalle C_1, C_2 .

On prend finalement, comme estimation de position, la moyenne des valeurs x_i restantes.

On distingue deux différences par rapport à l'algorithme étudié dans cette note :

- D'une part la définition de l'intervalle central n'est pas la même : notre intervalle fait intervenir la moyenne et les percentiles 0,28 et 0,72 alors que celui de Tukey ne prend en compte que les quartiles.

- D'autre part, les valeurs situées à l'extérieur de cet intervalle central ne sont pas traitées de la même façon : Tukey les élimine alors que nous les conservons après les avoir modifiées.

Signalons enfin que cet algorithme peut se généraliser au cas de la régression linéaire multiple (cette question sera traitée dans une note ultérieure).

b) cas du modèle polynomial

L'algorithme est identique dans son principe :

1. $k = 1$

2. Calculer les estimations $\hat{\delta}_{kj}$ ($j = 0, 1, \dots, p$) des moindres carrés pour l'échantillon (x_i, y_i) et pour la $k^{\text{ième}}$ itération.

3. Calculer une estimation robuste S_k de l'écart-type des résidus.

4. Tester chaque résidu $d_i = y_i - \sum_{j=0}^p \hat{\delta}_{kj} x_i^j$ par rapport à la quantité hS_k :

+ Si $d_i \geq 0$ et si $d_i > hS_k$ poser $y'_i = (\sum_{j=0}^p \hat{\delta}_{kj} x_i^j) + hS_k$.

+ Si $d_i < 0$ et si $|d_i| > hS_k$ poser $y'_i = (\sum_{j=0}^p \hat{\delta}_{kj} x_i^j) - hS_k$.

5. Si tous les d_i appartiennent à l'intervalle

$$\left[\left(\sum_{j=0}^p \hat{\delta}_{kj} x_i^j \right) - hS_k, \left(\sum_{j=0}^p \hat{\delta}_{kj} x_i^j \right) + hS_k \right] \text{ poser } \delta_j^* = \hat{\delta}_{kj}$$

pour $j = 0, 1, 2, \dots, p$ et aller en 6.

Si non :

+ Si $k = 1$ aller en 2.

+ Si $k > 1$, tester les variations relatives des coefficients $\hat{\delta}_{kj}$:

+ Si $\forall j, |\hat{\delta}_{kj} - \hat{\delta}_{k-1,j}| / |\hat{\delta}_{k-1,j}| \leq \alpha$

poser $\delta_j^* = \hat{\delta}_{kj}$ pour $j = 0, 1, \dots, p$ et aller en 6.

+ Si non, faire $k = k + 1$ et aller en 2.

6. Arrêter la procédure.

Comme, dans le cas du modèle linéaire, les estimations obtenues par cet algorithme sont notées δ_j^* ($j = 0, 1, \dots, p$). Pour la simulation, on a conservé les valeurs $h = 2$, $\alpha = 0,001$ et la même définition pour S_k .

Il faut remarquer qu'à chaque itération on aura à résoudre dans l'étape n° 2 un système linéaire de $p + 1$ équations. En conséquence il sera nécessaire d'utiliser un sous-programme pour résoudre ce système d'équations. Dans le sous-programme FORTRAN PULROB on a utilisé le sous-programme RESOL de la bibliothèque CII basé sur la méthode de Gauss.

3 Résultats

Les qualités des algorithmes sont testées à l'aide d'échantillons artificiels (méthode de Monte-Carlo). Les nombres au hasard sont obtenus par une méthode congruentielle et les échantillons normaux à partir de la transformation de BOX-MULLER. Les calculs ont été réalisés en double précision sur un ordinateur CII 10070. On a sélectionné cinq distributions symétriques différentes pour les erreurs E_j :

a) La distribution normale réduite : $N(0,1)$.

b) Une distribution normale réduite contaminée par une distribution normale $N(0,4)$:

$$0,8N(0,1) + 0,2N(0,4).$$

c) La distribution double-exponentielle de densité :

$$f(x) = (1/2) \exp(-|x|) \text{ où } -\infty < x < +\infty.$$

d) La distribution de Cauchy de densité :

$$f(x) = 1/\pi(1+x^2) \text{ où } -\infty < x < +\infty.$$

e) La distribution uniforme de densité :

$$f(x) = \begin{cases} 1 & \text{si } 0 < x < 1 \\ 0 & \text{partout ailleurs.} \end{cases}$$

Pour chacune de ces cinq lois on a utilisé $n' = 1000$ échantillons artificiels de taille $n = 10, 20, 50$ ou 51 . Le modèle linéaire a été testé pour $a = 0$ et $b = 1$ (le choix de valeurs différentes n'affecterait pas les résultats obtenus). Les x_i étant échantillonnés suivant une loi $N(0,1)$.

Pour le modèle polynômial on a choisi $p = 2$, $\delta_0 = -4,267$, $\delta_1 = 3,173$, $\delta_2 = -0,142$ (ces valeurs résultent d'une étude relative au coût d'entretien d'une flotte de camions).

Les x_i étant distribuées uniformément sur l'intervalle $[5, 15]$.

L'efficacité de l'ajustement au modèle théorique fourni par les deux algorithmes est mesurée grâce à l'erreur quadratique moyenne relative :

$$R_Y = \frac{\sum_{j=1}^{n'} (\hat{\gamma}_j - \gamma)^2}{\sum_{j=1}^{n'} (\gamma_j^* - \gamma)^2} \text{ où ,}$$

γ = coefficient de régression pour la population,

$\hat{\gamma}$ = coefficient de régression donné par la méthode des moindres carrés pour le $j^{\text{ème}}$ échantillon,

γ_j^* = coefficient de régression fourni par l'algorithme étudié pour le $j^{\text{ème}}$ échantillon.

Une valeur de R supérieure à un indiquera que l'algorithme proposé fournit une estimation meilleure que la méthode des moindres carrés

puisque dans cette hypothèse, les valeurs de γ_j^* seront statistiquement plus proches de γ que les valeurs de $\hat{\gamma}_j$.

Les résultats obtenus sont consignés ci-dessous :

$$\bar{k} = \text{nombre moyen d'itérations} : \bar{k} = (1/n') \sum_{j=1}^{n'} k_j,$$

$$\sigma_k = \text{écart-type de la distribution expérimentale du nombre d'itérations} : \sigma_k = (1/n') \sum_{j=1}^{n'} (k_j - \bar{k})^2,$$

k_j = nombre d'itérations nécessaires pour le $j^{\text{ème}}$ échantillon.

1) Cas du modèle linéaire :

	n = 10	n = 20	n = 50
R_a	0,98	0,99	0,99
R_b	0,99	1,00	0,98
\bar{k}	2,29	2,38	2,69
σ_k	2,86	1,70	1,24

Tableau 1 : Erreurs distribuées suivant $N(0,1)$

	n = 10	n = 20	n = 50
R_a	1,32	1,61	1,77
R_b	1,19	1,43	1,73
\bar{k}	3,79	4,30	4,18
σ_k	3,80	2,42	1,37

Tableau 2 : Erreurs distribuées suivant $0,8N(0,1) + 0,2N(0,4)$

Erreurs distribuées suivant $(1 - \epsilon)N(0,1) + \epsilon N(0,4)$ et $n = 51$:

$$\text{Pour } \epsilon = 0 : R_a = 0,98 ; R_b = 0,99 = \bar{k} ; 2,69 ; \sigma_k = 1,16$$

$$\text{Pour } \epsilon = 0,10 : R_a = 1,65 ; R_b = 1,63 ; \bar{k} = 3,92 ; \sigma_k = 1,24.$$

Erreurs distribuées suivant une loi double exponentielle et $n = 50$:

$$R_a = 1,65 ; R_b = 1,57 ; \bar{k} = 3,95 ; \sigma_k = 1,21$$

Erreurs distribuées suivant une loi de Cauchy et $n = 50$:

$$R_a = 12,38 ; R_b = 475,96 ; \bar{k} = 6,08 ; \sigma_k = 3,07.$$

Erreurs distribuées uniformément et $n = 50$:

$$R_a = 1,00 ; R_b = 0,99 ; \bar{k} = 1,04 ; \sigma_k = 0,23$$

Pour avoir une idée de la précision des chiffres indiqués on a appliqué l'algorithme à 50 groupes différents de 1000 échantillons pour $n = 10$. Avec

$\epsilon = 0$ on trouve pour R_a une moyenne de 0,99 et un écart-type de 0,008 et pour R_b une moyenne de 0,99 et un écart-type de 0,007. Pour $\epsilon = 0,2$ ces valeurs deviennent respectivement 1,34 et 0,028, 1,19 et 0,032.

L'examen des résultats montre, en premier lieu, la bonne robustesse de l'algorithme. En effet, son efficacité n'est jamais inférieure à 0,98 pour l'ensemble des situations testées. On observe, en outre, que pour toutes les distributions plus étalées que la loi normale (normale contaminée, double exponentielle, Cauchy), l'efficacité est toujours supérieure à un ce qui implique une amélioration par rapport à la méthode des moindres carrés. Cette amélioration est d'autant plus sensible que les erreurs sont plus grandes : Pour $n = 50$ R_b croît de 1,57 pour la loi double exponentielle à 475,96 pour la loi de Cauchy en passant par 1,73 pour la loi normale contaminée. R_a obéit à un schéma identique.

On remarque aussi, dans le cas des erreurs normales contaminées, que l'efficacité augmente avec la taille n de l'échantillon.

Le cas des erreurs uniformes indique que la méthode est pratiquement équivalente à la méthode des moindres carrés lorsque la distribution des erreurs possède des queues plus courtes que celles de la loi normale.

Bien que cela ne soit pas le but recherché dans cette note, il est intéressant de comparer ces résultats avec ceux publiés pour d'autres algorithmes. Il est clair que ces comparaisons n'auront qu'une valeur indicative puisque les conditions expérimentales dans lesquels ont été testés les différents algorithmes ne sont pas rigoureusement identiques (valeurs différentes de n' , manque d'information sur la façon dont sont créés les échantillons, plus précisément sur les suites de nombres au hasard utilisés pour générer les échantillons artificiels, etc.) On se limitera aux deux algorithmes suivants pour lesquels les conditions de simulations sont voisines des nôtres :

a) L'algorithme L_p de minimisation de la puissance p ($1 \leq p \leq 2$) des valeurs absolues des résidus où on minimise par rapport à a et b , la quantité :

$$F = \sum_{i=1}^n |y_i - a - bx_i|^p.$$

FORSYTHE (*Tecnometrics*, vol. 14, n°1, 1972, pp 159-166) recommande de choisir $p = 1,5$. EKBLOM (*Nordisk Tidskr-Information behandling* 14, 1974, pp 22-32) qui a testé ce même algorithme, pour une autre classe de distribution des erreurs incluant la loi de Cauchy et à l'aide d'un critère d'ajustement différent de celui de FORSYTHE, recommande $p = 1,25$.

Pour $p = 1,5$, $n' = 400$, $n = 51$, les erreurs étant distribuées suivant $(1 - \epsilon)N(0,1) + \epsilon N(0,4)$, on a les résultats suivants (cf FORSYTHE, *Tecnometrics*, 14, n° 1, 1972, p 159-166) :

Pour $\epsilon = 0$ (pas de contamination : $R_a = 0,95$; $R_b = 0,95$;

Pour $\epsilon = 0,10$: $R_a = 1,50$; $R_b = 1,60$

Avec $p = 1,25$, les autres conditions étant les mêmes on a :

Pour $\epsilon = 0$: $R_a = 0,85$; $R_b = 0,90$

Pour $\epsilon = 0,10$: $R_a = 1,55$; $R_b = 1,70$

b) L'algorithme itératif de Winsorisation (RES-ITER) recommandé pour sa stabilité dans (3) et qui a été décrit dans le second paragraphe de cet article.

Pour $n' = 200$, les erreurs étant distribuées suivant $(1-\epsilon)N(0,1) + \epsilon N(0,4)$, les résultats publiés dans (3) apparaissent les meilleurs avec $g = 1$ et $l = 5$ (un point Winsorisé à chaque extrémité de l'échantillon et cinq itérations successives) :

Pour $n = 10$ et $\epsilon = 0$: $R_a = 0,98$; $R_b = 0,93$
 $\epsilon = 0,20$: $R_a = 1,26$; $R_b = 1,29$

Pour $n = 20$ et $\epsilon = 0$: $R_a = 0,99$; $R_b = 0,95$
 $\epsilon = 0,20$: $R_a = 1,19$; $R_b = 1,26$

Pour $n = 50$ et $\epsilon = 0$: $R_a = 1,00$; $R_b = 0,98$
 $\epsilon = 0,20$: $R_a = 1,10$; $R_b = 1,32$

En fait, on trouve dans (3) plusieurs algorithmes basés sur la Winsorisation. Certains d'entre eux donnent ponctuellement de meilleurs résultats que l'algorithme de RES-ITER que nous avons retenu pour la comparaison. Malheureusement ces résultats ne sont pas stables. Ainsi un de ces algorithmes (désigné par PRES-IT/LV dans (3)) fournit une efficacité maximum de 1,91 alors que son efficacité minimum chute à 0,69.

Si on se restreint aux erreurs normales et normales contaminées on constate que $L_{1,5}$ est un peu moins robuste que notre algorithme puisque son R minimum vaut 0,95 contre 0,98 et son R maximum 1,60 contre 1,65. Quant à $L_{1,25}$ il n'est pas suffisamment efficace, dans le cas où $\epsilon = 0$, pour être compétitif.

RES-ITER donne d'assez bons résultats pour $n \geq 20$ et $\epsilon = 0$, mais son efficacité est nettement plus faible que celle de notre algorithme en présence de contamination. En effet, pour $n = 50$ et $\epsilon = 0,2$ on trouve $R_b = 1,32$ pour RES-ITER alors que nous trouvons dans le tableau 2, $R_b = 1,73$.

Si on tient compte, en outre, de la simplicité de mise en oeuvre, il est clair que la comparaison est en faveur de notre méthode.

Pour terminer donnons quelques informations sur le temps de calcul du sous-programme Fortran REGROB.

Ce temps dépend essentiellement de trois facteurs :

- 1) la taille n de l'échantillon.
- 2) la précision imposée α .
- 3) la longueur des queues de distribution des erreurs.

La croissance de n entraîne l'augmentation du nombre d'itérations comme on peut le voir dans les tableaux 1 et 2. On constate aussi que, pour n et α donnés, k est d'autant plus grand que la distribution des erreurs est plus étalée. Pour $n = 50$, par exemple, on passe de $\bar{k} = 2,69(N(0,1))$ à $\bar{k} = 6,08$ (Cauchy). On conçoit en effet, qu'à la première itération, la droite obtenue sera d'autant moins bonne que les erreurs sont plus grandes.

La valeur de α influe sur le temps de calcul : ainsi pour $n = 20$, $\varepsilon = 0,2$ et 1000 échantillons lorsque l'on passe de $\alpha = 0,001$ à $\alpha = 0,1$ celui-ci diminue de 20% environ.

2) Cas du modèle polynômial :

	n = 10	n = 20	n = 50
R_0	0,98	1,00	0,99
R_1	0,98	1,00	0,99
R_2	0,98	1,00	0,99
\bar{k}	2,43	2,23	2,37
σ_k	2,90	1,90	1,06

(On note R_i
au lieu de
 R_{δ_i}).

Tableau 3 : Erreurs distribuées suivant $N(0,1)$

	n = 10	n = 20	n = 50
R_0	1,13	1,40	1,68
R_1	1,14	1,40	1,69
R_2	1,14	1,40	1,69
\bar{k}	4,11	4,56	4,25
σ_k	5,13	3,13	1,56

Tableau 4 : Erreurs distribuées suivant $0,8N(0,1)+0,2N(0,4)$

Erreurs distribuées suivant une loi de Cauchy et $n = 10$:

$$R_0 = 11,21 ; R_1 = 11,12 ; R_2 = 10,13 ; \bar{k} = 6,58 ; \sigma_k = 7,71.$$

Erreurs distribuées suivant une loi de Cauchy et $n = 50$:

$$R_0 = 42,10 ; R_1 = 51,91 ; R_2 = 65,47 ; \bar{k} = 6,15 ; \sigma_k = 2,77.$$

Ces résultats confirment les conclusions générales formulées pour le modèle linéaire simple. La robustesse de l'algorithme est bonne puisque le minimum d'efficacité est 0,98. En plus des remarques déjà faites à propos du modèle linéaire on observe qu'ici les valeurs de R sont très stables et qu'elles sont pratiquement indépendantes du paramètre δ_j estimé pour un n et une distribution des erreurs donnés.

4 Conclusions

L'étude précédente montre l'excellente robustesse de l'estimateur de HUBER aussi bien pour le modèle linéaire simple que pour le modèle polynômial. En outre, cet algorithme présente l'avantage d'être assez simple à programmer.

Il convient toutefois de souligner que la garantie apportée par cet estimateur est une garantie statistique. Autrement dit, sur un ensemble assez grand d'échantillons on sera à peu près certain d'avoir, en moyenne, de bons résultats mais cela n'entraîne pas que l'on obtiendra systématiquement un résultat convenable pour chacun des échantillons étudiés. Pour

illustrer cette remarque considérons l'échantillon suivant :

$$x_i : 0, 1, 2, 3, 4, 12.$$

$$y_i : 0, 1, 2, 3, 4, 2.$$

Si on ajuste un modèle linéaire simple sur cet échantillon, la droite des moindres carrés passe par les points ($x = 20, y = 4$) et ($x = 0 ; y = 1,5$) ce qui constitue un très mauvais résultat.

Malheureusement, dans ce cas, l'estimateur de HUBER reste impuissant puisque la droite obtenue est confondue avec la droite des moindres carrés. Cette contre-performance s'explique par le fait que l'algorithme de HUBER est étroitement dépendant de la droite obtenue lors de la première itération. Cette première droite étant ici très éloignée de l'orientation générale du nuage, il n'est pas possible, par des itérations successives, de modifier sa pente. Le choix de la droite des moindres carrés comme point de départ des itérations n'apparaît donc pas toujours comme le meilleur. Mais il faut noter que le choix d'un point de départ plus robuste entraîne souvent une perte d'efficacité de l'algorithme lorsqu'il n'y a pas de points aberrants. C'est par exemple ce qui se passe si on adopte comme point de départ, une droite de régression par les médianes (5). Cette droite passerait dans l'exemple cité plus haut par les points ($x = 1, y = 1$) et ($x = 4, y = 3$) ce qui constituerait un bon point de départ. Cette question du choix d'un bon point de départ pour les itérations reste donc jusqu'à maintenant sans solution pleinement satisfaisante.

BIBLIOGRAPHIE

- (1) HUBER J.P. : *Robust regression : asymptotics, conjectures and Monte-Carlo* - *The Annals of Statistics*, Vol. 1 , n° 5, 1973, pp 799-821.
- (2) BRENOT J., CAZES P., LACOURLY N. :
Pratique de la régression : qualité et protection - Cahiers du B.U.R.O., n° 23, 1975, pp 3-84.
- (3) YALE C. et FORSYTHE A.B. :
Winsorized regression - *Technometrics*, Vol. 18, n°3, 1976, pp 291-300;
- (4) TUKEY J.W. : *Exploratory data analysis* - Addison-Wesley, Reading Massachusetts, 1970.
- (5) ANDREWS D.F. : *A robust method for multiple linear regression* - *Technometrics*, 16, n° 4, 1974, pp 523-531.