STATISTIQUE ET ANALYSE DES DONNÉES

CHRISTINE JACOB

Algorithme inverse de Moore-Penrose

Statistique et analyse des données, tome 4, nº 1 (1979), p. 65-72
<http://www.numdam.org/item?id=SAD_1979__4_1_65_0>

Statistiques et Analyse des Données
1 - 1979 pp. 65, 72.

ALGORITHME INVERSE DE MOORE-PENROSE

Christine JACOB
INRA - CNRZ, Laboratoire de Biométrie, 78350 Jouy-en-Josas

MOTS-CLES.

Inverse de Moore-Penrose ; inverse généralisée ; estimation du modèle de Gauss-Markov ; regression linéaire ; moindres carrés pondérés.

LANGAGE.

ISO FORTRAN

DESCRIPTION ET BUT.

Etant données, les matrices réelles $A(MxN)$ et $b(Mx1)$, le système linéaire

(1) $$Ax = b$$

n'est pas toujours consistant. Lorsqu'il ne l'est pas, le vecteur résiduel $r = Ax - b$ est différent de 0, quel que soit le vecteur x $(N \times 1)$. Dans ce cas, on cherchera une solution approchée de (1) qui minimisera le vecteur résiduel r dans un certain sens (i.e. selon une norme donnée). Une solution approchée souvent utilisée, en particulier pour l'estimation des paramètres en analyse de dispersion (modèle de Gauss-Markov) ou en analyse de régression est la solution des moindres carrés de (1), définie comme un vecteur $x(N \times 1)$ qui minimise la norme eudidienne du vecteur résiduel

$$(Ax - b)' (Ax - b)$$ - (l'exposant ' note la transposition)

x est alors solution de l'équation "normale"

$$A' Ax = A'b$$

Si $A'A$ est singulière, il y a une infinité de solution x. Mais si x est contraint à être de norme minimale (ce qui correspond à une solution de variance minimum en analyse de dispersion), alors x est unique et vaut :

$$x = A^{+}b$$

où A^+ est l'inverse de Moore-Penrose de A (étudiée par E.H. Moore (3), (4) , puis par R. Penrose (5)). C'est l'unique matrice réelle $G(N \times M)$ satisfaisant les cinq groupes d'équations équivalents suivants :

1) $A'AG = A'$; $G' = DA$
2) $GAA' = A'$; $G' = AD$
3) $GAA' = A'$; $AGG' = G'$
4) $A'AG = A'$; $G'GA = G$
5) $AGA = A$; $GAG = G$
   $(AG)' = AG$; $(GA)' = GA$

Ces équations impliquent en particulier que si A^+ est l'inverse de Moore-Penrose de A, alors $A^{+\prime}$ est l'inverse de Moore-Penrose de A'.

Remarque : Si une solution des moindres carrés pondérés de (1) est demandée, alors on minimisera l'équation

$$(A\,x - b)'\,W(A\,x - b) \tag{2}$$

où $W(M \times M)$ est une matrice définie positive donnée. L'équation (2) pouvant s'écrire de manière équivalente sous la forme suivante,

$$[V\,(Ax - b)]'\,[V(Ax - b)]$$

où V est une matrice, régulière d'ordre M telle que $W = V'V$. On est ramené au problème précédent, à savoir, trouver une solution des moindres carrés de l'équation

$$(V\,A)\,x = (V\,b)$$

Méthode.

Il existe un certain nombre de méthodes de calcul de A^+ qui sont :

- des méthodes indirectes à partir d'inverses généralisées moins spécifiques que celle de MOORE-PENROSE. Elles sont peu interessantes du point de vue de la programmation car elles nécessitent un programme de calcul de telles inverses généralisées,
- ou bien des méthodes directes nécessitant une décomposition de A. La décomposition en valeur singulière, la triangularisation de Householder, la triangularisation modifiée de Gram-Schmidt, la formule de Noble et la factorisation de plein rang de Mac Duffee, par exemple, sont de telles méthodes mais elles nécessitent beaucoup de calculs ((1), (2)),
- ou encore des méthodes itératives comme celle de Greville qui est finie et calcule à l'itération k, l'inverse de Moore-Penrose de la matrice constituée des k colonnes de A. La méthode donnée ici est une méthode itérative convergente ; elle est décrite par **A. Ben Israël** et N.E. Greville (1). Elle nécessite peu de calculs et est très précise : A^+ ayant la forme DA', on choisit une suite $(X_k(N \times M))_{k \in \mathbb{N}}$, telle que

$$X_{k+1} = D_{k+1}\,X_k \qquad \text{avec} \qquad X_o = D_o\,A'$$

A. Ben Israël et N. E. Greville (1) ont montré que la suite

$$X_{k+1} = (I + T_k + T^2{}_k + \ldots + T_k^{p-1})\,X_k \tag{A}$$

encore égale à la suite,

$$X_{k+1} = (I + R_k + R_k{}^2 + \ldots + R_k^{p-1})\,X_k$$

où $T_k = I - X_k A$ et $R_k = A^+ A - X_k A$
avec pour approximation initiale la matrice $X_o = A' BA'$
où $B(N \times M)$ est tel que $\rho(R_o)$, le rayon spectral de R_o est plus petit que 1, converge vers A^+ lorsque $k \to \infty$.
De plus, c'est une méthode itérative d'ordre car la suite des résidus R_k satisfait,

$$\| R_{k+1} \| \leq \| R_k \|^p$$

pour toute norme de matrice, multiplicative. En choisissant,

$$X_o = \beta A'$$

où est un scalaire réel satisfaisant

$$0 < \beta < 2/\lambda_1(A'A)$$

($\lambda_1(A'A) \geq \ldots \geq \lambda_r(A'A) > 0$ sont les valeurs propres non nulles de la matrice $A'A$), $\rho(R_o) < 1$ et la suite précédente converge. La norme spectrale de R_k est minimum lorsque,

$$\beta = \beta_o$$
$$= 2/[\lambda_1(A'A) + \lambda_r(A'A)]$$

Si $p = 2^q$ où q est un entier positif, la suite précédente peut s'écrire,

$$X_{k+1} = (I + T_k)(I + T^2_k) \ldots (I + T^{2^{q-1}}_k) X_k.$$

Le sous programme PENROS donné ci-après calcule cette suite avec $X_o = \beta_o A'$. A l'étape k, on calcule X_k et le produit $AX_k A$. Si ERR, la valeur absolue de l'erreur relative de chaque terme de cette matrice (par rapport à A),

$$ERR_{ij} = \left| \frac{(AX_k A - A)_{ij}}{A_{ij}} \right| \qquad (i = 1 \ldots M,\ j = 1 \ldots N),$$

ést plus petit que le paramètre d'entrée PRECIS, l'algorithme s'arrête là. S'il existe au moins un terme pour lequel $ERR_{ij} \geq$ PRECIS, le terme suivant de la suite X_{k+1} est calculé. L'algorithme s'arrête avant que ERR_{ij} n'augmente d'une itération à la suivante, à cause des erreurs d'arrondi.

STRUCTURE.

Subroutine PENROS(A,XK,M,N,MN,N2,TK,XX,DK,EK,AT,ERROR,IFAULT,ETA,PRECIS).
Paramètres formels (cf. cartes commentaires du sous programme), toutes les matrices sont stockées unidimensionnellement.

- A est la matrice M x N (en entrée) dont on veut calculer l'inverse généralisée A^+.
- XK est la matrice $A^+(N \times M)$ (en sortie)
- TK, XX, DK, EK, AT sont des matrices de travail
- ERROR est l'erreur relative de précision du résultat i.e., ERROR est de l'ordre de grandeur des erreurs ERR_{ij} relatives à XK. (cf. § précédent).
- IFAULT est un indicateur d'erreur.
- ETA et PRECIS sont des constantes dépendant de la matrice. ETA est le plus petit hombre strictement positif représentable en machine et PRECIS est le plus petit nombre positif représentable en machine tel que 1+PRECIS ≠ 1.

Sous programmes auxiliaires.: TRANSP, PRDCT, TDIAG2, LRVT2 (Pour leur structure, cf. le listing des sous programmes en fin d'article).
Le sous programme TRANSP calcule la transposée d'une matrice.
Le sous programme PRDCT calcule le produit de deux matrices. Les sous programmes TDIAG2 et LRVT2 sont respectivement les sous programmes TDIAG AS 60.1 et LRVT AS 60.2 publiés dans Applied Statistics (5), dans lesquels les tableaux bidimensionnels (A et Z dans TDIAG, et Z dans LRVT) sont stockés unidimensionnellement, colonne par colonne. TDIAG2 réduit une matrice symétrique réelle selon la forme tridiagonale (réduction de Householder), et LRVT2 calcule les valeurs et vecteurs propres d'une matrice symétrique tridiagonale.

OPTIMISATION, PRECISION, TEMPS DE CALCUL.

Etant donné l'encombrement mémoire des vecteurs de travail du sous programme PENROS (TK(N x N), XX(N x N), DK(N x N), EK(N x N), AT(M x N)) il est recommandé de n'utiliser le sous-programme que dans le cas $N \leq M$. Dans le cas inverse, on calculera l'inverse de Moore-Penrose de A'. Celle de A se déduit alors directement par la relation

$$(A')^{+} = (A^{+})' \quad \text{(cf. § DESCRIPTION ET BUT)}$$

Supposons donc $N \leq M$; on définit (quel que soit l'entier $k > 0$) l'ordre optimum p comme l'ordre de la méthode itérative (A) qui minimise l'effort de calcul nécessaire pour obtenir au moins l'approximation X_k obtenue en k itérations, par la méthode du premier ordre et avec la même approximation initiale X_o. En supposant que l'effort de calcul requis pour ajouter ou soustraire une matrice identité, est négligeable par rapport à celui demandé par la multiplication, et si on définit l'unité d'effort de calcul comme l'effort nécessaire pour multiplier deux matrices N x N, alors l'itération $k_{(p)}$ de la méthode d'ordre p nécessite $(p+a)$ ou $(2q + a)$ unités d'efforts $(a = -2+2(M/N))$ selon que $p \neq 2^q$ ou $p = 2^q$ et $k_{(p)}$ itérations nécessiteront respectivement $k_{(p)}\ (p+a)$ et $k_{(p)}\ (2q + a)$ unités, soit,pour obtenir une au moins aussi bonne précision qu'avec la méthode d'ordre 1 à l'itération $k_{(1)}$

$<\dfrac{\mathrm{Log}(k_{(1)}+1)}{\mathrm{Log}\ p}>\ (p+a)$ et $<\dfrac{\mathrm{Log}(k_{(1)}+1)}{q\ \mathrm{Log}\ 2}>\ (2q+a)$ unités ($<x>$ note le plus petit entier $\geq x$)).

L'ordre optimum approché p sera donc un entier $p \geq 2$ qui minimisera la fonction

$$p \to f(p) = \begin{cases} \dfrac{p+a}{\mathrm{Log}p} & \text{pour } p \neq 2q \\ \dfrac{2q+a}{q\ \mathrm{Log}2} & \text{pour } p = 2^q\ ;\ q = 1,2\ldots \end{cases}$$

p dépendra donc de M/N.

Soit $p_o = [x_o]$ (entier le plus proche de x_o), où x_o est le nombre réel qui minimise la fonction réelle $x \to g(x) = \dfrac{x+a}{\mathrm{Log}x}$ $(a \geq 0)$. La fonction $q \to \dfrac{2q+a}{q\mathrm{log}2}$ étant décroissante, on choisira p de la forme 2^q avec $q = q_o$ où q_o vérifie,

(B) $$\frac{2q_o + a}{q_o\ \mathrm{Log}2} \leq \frac{p_o+a}{\mathrm{Log}\ p_o} \qquad (a \geq 0)$$

on note $$h(q) = \frac{2q + a}{q\ \mathrm{Log}2}$$

On trouve par exemple,

$p_o = 3$ pour $a = 0.33 \Longrightarrow g(p_o) = 3.03$ et $h(4) = 3.$

$p_o = 4$ pour $a = 1.6 \Longrightarrow g(p_o) = 4.04$ et $h(4) = 3.46$

$p_o = 5$ pour $a = 3.076 \Longrightarrow g(p_o) = 5.019$ et $h(4) = 3.65$ $h(3) = 4.36$

$p_o = 9$ pour $a = 11. \Longrightarrow g(p_o) = 9.1$; $h(4) = 6.84$; $h(3) = 8.16$; $h(2) = 10.8$

$p_o = 15$ pour $a = 20 \Longrightarrow g(p_o) = 13$; $h(4) = 10.08$ etc...

Les valeurs $q_o \geq 4$ vérifient l'inégalité (B) pour des valeurs de a appartenant au moins à l'intervalle $[0,100]$ (i.e. pour un rapport M/N appartenant au moins à l'intervalle $[1, 50]$). On choisit $q_o = 4$ car plus q est grand et plus les erreurs d'arrondi risquent de diminuer (dès la première itération) la précision théorique espérée. Pour a beaucoup plus grand que 100, l'utilisateur aura intérêt à choisir $q_o > 4$.

<u>Temps d'exécusion sur IRIS 80</u>. (dont le cycle de base est d'environ 700 ns).

Pour une matrice A de dimension 20 x 10, et dont les éléments ont 5 chiffres significatifs, le temps d'exécution est inférieur à 57 s et l'erreur relative de précision obtenue est de l'ordre de 1.6×10^{-4}, en simple précision. Pour une matrice A (60 x 10) dont les éléments ont également 5 chiffres significatifs, le temps d'éxécution est inférieur à 55 s et l'erreur relative de précision (en simple précision) est de l'ordre de 2.10^{-5}.

BIBLIOGRAPHIE .

[1] A. Ben Israël, T.N.E. Greville (1974) *Generalized inverses theory and applications.* Wiley.

[2] A.A. Dubrulle An extension of the domain of the APL Domino function to rank deficient linear least-squares systems in A.P.L. 75 Pise, p. 105-114.

[3] E.H. Moore On the reciprocal of the general algebraïc matrix. *Bull. Amer. Math. Soc. 26, (1920).*

[4] E.H. Moore (1935) General analysis *Memoirs Amer. Philos. Soc. 1.*

[5] R. Penrose A generalized inverse for matrices. *Proc. Cambridge Philos. Soc. 51 (1955).*

[6] D. N. Sparks, A.D. Todd Algorithm AS 60. Latent Roots and Vectors of a Symmetric Matrix. *Appl. Statist. Vol. 22 n° 2 (1973)*

ALGORITHMS

```
      SUBROUTINE PENROS(A,XK,M,N,MN,N2,TK,XX,DK,EK,AT,ERROR,IFAULT,ETA,
     1PRECIS)
C
C     BUT
C
C     CALCUL DE L INVERSE DE MOORE PENROSE D UNE MATRICE REELLE
C
C     PARAMETRES FORMELS
C
C     A  VECTEUR REEL ENTREE-LA MATRICE DES DONNEES (M*N)
C     XK VECTEUR REEL SORTIE-L INVERSE DE MOORE PENROSE DE A.(N*M)
C     M  ENTIER       ENTREE-NOMBRE DE LIGNES DE A ET DE COLONNES DE XK
C     N  ENTIER       ENTREE-NOMBRE DE COLONNES DE A ET DE LIGNES DE XK
C     MN ENTIER       ENTREE-LE PRODUIT M*N
C     N2 ENTIER       ENTREE-LE PRODUIT N*N
C     TK VECTEUR REEL       -MATRICE DE TRAVAIL (N*N)
C     XX VECTEUR REEL       -MATRICE DE TRAVAIL (N*N)
C     DK VECTEUR REEL       -MATRICE DE TRAVAIL (N*N)
C     EK VECTEUR REEL       -MATRICE DE TRAVAIL (N*N)
C     AT VECTEUR REEL       -MATRICE DE TRAVAIL (M*N)
C     ERROR REEL      SORTIE-L ERREUR RELATIVE DE PRECISION APPROCHEE DE XK
C     IFAULT ENTIER   SORTIE-IFAULT=1 DES QUE N2 DIFFERE DE N*N
C                            IFAULT=2 SI MN#M*N
C                            IFAULT=3 SI M<N
C                            IFAULT=4 SI PLUS DE MITS=30 ITERATIONS SONT
C                            NECESSAIRES DANS LRVT2
C                            IFAULT=5 SI PLUS DE KMAX ITERATIONS SONT
C                            NECESSAIRES POUR OBTENIR UNE MEILLEURE PRECISION
C                            DE XK.LA VALEUR DE KMAX EST DECLAREE PAR DATA
C                            IFAULT=0 SINON
C                            SI IFAULT=1 OU 2,XK N EST PAS CALCULE
C     ETA REEL        ENTREE-LE PLUS PETIT NOMBRE POSITIF
C     PRECIS REEL     ENTREE-LE PLUS PETIT NOMBRE POSITIF TEL QUE
C                            1+PRECIS SOIT DIFFERENT DE 1
C
C     REMARK
C
C     XK NE PEUT ETRE DANS LE MEME EMPLACEMENT MEMOIRE QUE A
C
C     SOUS PROGRAMMES AUXILLIAIRES=TRANSP,PRODCT,TDIAG2,LRVT2
C
C     POUR UNE VERSION EN DOUBLE PRECISION,ENLEVER LE C EN COLONNE 1 DES
C     INSTRUCTIONS SUIVANTES
C       DOUBLE PRECISION ERROR,TOL,ETA,ETAN,PRECIS,AXA,EPS
C        DOUBLE PRECISION PRECIO,PRECI1,X,PRECIZ
C       DOUBLE PRECISION A,AT,XK,EK,TK,DK,XX,BZ,BZZ,ERR,ERR1,ERRO
C     DOIVENT ETRE EGALEMENT EN DOUBLE PRECISION,LES CONSTANTES ET LES FONCTIONS
C     DANS L INSTRUCY
C     LES SOUS PROGRAMMES AUXILLIAIRES APPELES ICI DOIVENT ETRE EGALEMENT
C     EN DOUBLE PRECISION
C
      DIMENSION A(MN),AT(MN),XK(MN),EK(N2),TK(N2),DK(N2),XX(N2)
      DATA NQ,KMAX/4,25/
C     2**NQ EST L ORDRE DE LA METHODE ITERATIVE
C     KMAX EST LE NB.MAXIMUM D ITERATIONS
      IFAULT=0
      IF(M.LT.N) IFAULT=3
      NN=N*N
      IF(NN.NE.N2)IFAULT=1
      IF(IFAULT.EQ.1) RETURN
      NM=N*M
      IF(NM.NE.MN) IFAULT=2
      IF(IFAULT.EQ.2) RETURN
```

```
C
C        CALCUL DES VALEURS PROPRES DU PRODUIT (A TRANSP.)*A
C
      CALL TRANSP(A,AT,M,N,MN)
      CALL PRDCT(AT,A,DK,N,M,N,MN,MN,N2)
      TOL=ETA/PRECIS
      CALL TDIAG2(N,TOL,DK,TK,XK,EK,N2,IFAULT)
      CALL LRVT2(N,PRECIS,TK,XK,EK,IFAULT,N2)
C
C        RECHERCHE DE LA PLUS PETITE VALEUR PROPRE NON NULLE DE (A TRANSP.)*A
C
      PRECIO=PRECIS*100
 1    PRECIZ=PRECIO
      PRECIO=PRECIZ/10
      X=TK(N)+TK(N)*PRECIO
      IF(X.NE.TK(N)) GO TO 1
      PRECIO=PRECIZ
      PRECI1=PRECIZ
 2    PRECIZ=PRECI1
      PRECI1=PRECIZ-PRECIO/10
      X=TK(N)+TK(N)*PRECI1
      IF(X.NE.TK(N)) GO TO 2
      EPS=TK(N)*PRECIZ
      DO 20 L=1,N
      IF(TK(L)-EPS)19,19,10
 10   LO=L
      GO TO 40
 19   IF(L.EQ.N) LO=N
 20   CONTINUE
C
C        CALCUL DE XO=BZZ*(A TRANSP.)
C
 40   BZ=TK(N)+TK(LO)
      BZZ=2.0/BZ
      DO 50 LL=1,MN
 50   XK(LL)=BZZ*AT(LL)
C
C        ALGORITHME
C
      DO 180 K=1,KMAX
C
C        CALCUL DE TK=I-XK*A
C
      CALL PRDCT(XK,A,TK,N,M,N,MN,MN,N2)
      DO 60 L1=1,N2
      TK(L1)=-TK(L1)
      DK(L1)=TK(L1)
 60   CONTINUE
      LL=-N
      DO 70 L2=1,N
      LL=LL+N+1
      TK(LL)=1.0+TK(LL)
      DK(LL)=1.0+TK(LL)
 70   CONTINUE
      DO 80 L1=1,N2
 80   EK(L1)=TK(L1)
```

```
C
C        CALCUL DE DK=(I+TK)(I+TK**2)...(I+TK**(2**LMAX))
C
      LMAX=NG-1
      DO 120 L=1,LMAX
      CALL PRDCT(EK,EK,XX,N,N,N,N2,N2,N2)
      DO 90 L1=1,N2
   90 EK(L1)=XX(L1)
      LL=-N
      DO 100 L2=1,N
      LL=LL+N+1
      XX(LL)=1.0+XX(LL)
  100 CONTINUE
      CALL PRDCT(DK,XX,TK,N,N,N,N2,N2,N2)
      DO 110 L1=1,N2
  110 DK(L1)=TK(L1)
  120 CONTINUE
C
C        CALCUL DE X(K+1)=DK*XK
C
      CALL PRDCT(DK,XK,AT,N,N,M,N2,MN,MN)
C
C        CALCUL D ERROR,L ERREUR RELATIVE DE PRECISION
C
      CALL PRDCT(AT,A,XX,N,M,N,MN,MN,N2)
      IJ=0
      KK=-N
      DO 130 J=1,N
      KK=KK+N
      DO 130 I=1,M
      IJ=IJ+1
      IK=I-M
      KJ=KK
      AXA=0.0
      DO 129 KL=1,N
      IK=IK+M
      KJ=KJ+1
      AXA=AXA+A(IK)*XX(KJ)
  129 CONTINUE
      ERR1=AXA-A(IJ)
      ETAN=-ETA
      IF(A(IJ).LT.ETAN.OR.A(IJ).GT.ETA) ERR1=ERR1/A(IJ)
C     ERR=DABS(ERR1)
      ERR=ABS(ERR1)
      IF(ERR.GT.PRECIS) GO TO 150


  130 CONTINUE
      ERROR=ERR
      DO 140 L=1,MN
  140 XK(L)=AT(L)
      GO TO 200
  150 IF(K.EQ.1) GO TO 160
      IF(ERR.GE.ERRO) GO TO 190
  160 ERRO=ERR
      DO 170 L=1,MN
  170 XK(L)=AT(L)
  180 CONTINUE
      IFAULT=5
      ERROR=ERR
      GO TO 200
  190 ERROR=ERRO
  200 RETURN
      END
```