Les Cahiers de l'Analyse des Données
Vol. XXII - 1997 - n°2 - pp.125-132.

SOURCES DE PROGRAMMES D'ANALYSE DE DONNÉES EN LANGAGE PASCAL: (II) : ÉLABORATION DES FICHIERS DE FACTEURS (IIA) : ANALYSE DISCRIMINANTE

[SOURCES PASCAL (IIA)]

J.-P. & F. BENZÉCRI

0 Introduction : fonction et déroulement du programme 'discri'

Une méthode générale de l'analyse multidimensionnelle consiste à convertir, d'abord, les ensembles d'éléments à traiter en des ensembles de points, munis de masse, situés dans un espace euclidien; puis à résoudre, dans cet espace, en termes géométriques, les problèmes usuels de la statistique: classification, affectation de cas individuels à des types...; problèmes auxquels se réduisent ceux du diagnostic automatique, de la reconnaissance des formes, etc...

Pour nous, l'espace euclidien considéré, n'est autre que celui engendré par un ensemble d'axes issus de l'analyse d'un tableau de correspondance approprié; le choix du nombre des axes retenus permettant de varier l'application de la méthode.

La pratique confirme ce que l'on pouvait attendre *a priori*: tout dépend de la fidélité de la structure euclidienne construite à la structure réelle qu'elle est censée repésenter; c'est-à-dire, du choix du tableau de correspondance; lequel, dans de nombreux cas, n'est pas directement formé de données primaires; mais résulte de celles-ci par un codage.

Même si des codages divers peuvent convenir également à l'élaboration d'un même ensemble de données, (cf [CODAGE DISCRI.], in CAD, Vol. XXI, pp.75 sqq; [CARACT. MARCHE], *ibid*, pp.83 sqq. et 311 sqq.), il est indispensable qu'un logiciel d'analyse de données effectue tout codage usuel.

Cependant, après avoir, dans [SOURCES PASCAL IA, B, C, D], considéré, exclusivement, l'analyse de tableaux de correspondance (principal ou supplémentaires); avant de proposer des algorithmes pour construire de tels tableaux; nous traitons, dans [SOURCES PASCAL II] de l'élaboration des fichiers de facteurs, par l'analyse discriminante et la classification. Et le programme de codage 'zrang' reste à traiter ensuite; ainsi que divers programmes d'observation des données et de construction de tableaux.

Les cahiers de l'analyse des données 0339-3097/97/02/$ 7.00/© Gauthier-Villars

Le présent article est consacré au programme 'discri' d'analyse discriminante.

En bref, prenant pour cadre l'analyse d'un tableau de correspondance de base, 'Disq:tab', on considère les fichiers de facteurs, 'Fac.w', afférents à deux ensembles que nous noterons: Ic, ensemble des centres; et Ii, ensemble des individus. Dans l'espace euclidien engendré par les 'carf' premiers axes factoriels, on cherche, pour tout individu: ii ∈ Ii, le centre: ic ∈ Ic, dont il est le plus proche; ce qu'on écrit comme une formule d'affectation: ii -> ic.

Le nombre carf est laissé au choix de l'utilisateur. Pour Ic ou Ii, on peut prendre tout ensemble pour lequel existe un fichier de facteurs: ensembles principaux et supplémentaires de l'anayse de base: {Ip, Is, Jp, Js}; ensembles externes adjoints en lignes ou colonnes supplémentaires: {Ixa, Ixb...}; ensembles des centres de gravité des classes d'une CAH effectuée sur I, sur J, ou sur des ensembles externes (le calcul des facteurs pour les centres étant effectué par le programme FACOR, d'aide à l'interprétation: cf. IIB%%).

```
begin benzecri;rpc:='O';
while not(rpc='N') do begin
 rpc:='N';principer;
 if (rpc='O') then carfer;
 if (rpc='O') then begin
   sie:=sic;liFer;carc:=carel;sisc:=sise;
   for a:=1 to carf do Fcac[a]:=Fcae[a];
   sie:=sii;liFer;cari:=carel;sisi:=sise;
   for a:=1 to carf do Fcai[a]:=Fcae[a];
   unloadseg(@principer);
   affecter;texter;fermer;
   write('faut-il poursuivre O ou N ');readln(rpc);end;end;end.
```

Comme on l'a fait pour 'qori': cf. IB§0, et pour 'qorlsup': cf. ID§0, on considérera d'abord le programme principal, 'discri', qui est très bref.

Par la procédure 'principer', l'utilisateur choisit le tableau de base dont dépendent les ensembles Ic et Ii ; et le programme vérifie que les fichiers 'Fac' existent, sur le disque, pour ces deux ensembles. Par 'carfer', l'utilisateur choisit le nombre, 'carf', des facteurs à utiliser, dans la limite du nombre disponible; et le programme vérifie que ce dernier nombre est le même pour les deux ensembles; ce qui confirme qu'il 'agit de la même analyse de base (et non de deux analyses distinctes d'un même tableau de base, considéré avec des ensembles supplémentaires différents; etc.).

Après ces choix et vérifications, la procédure 'liFer' est appelée deux fois afin de mettre en mémoire centrale le fichier des carf premiers facteurs, pour les ensembles Ic et Ii.

On procède alors, par 'affecter' à l'analyse discriminante proprement dite: i.e. à l'affectation des individus aux centres. Le listage des résultats, sous un format que nous préciserons ensuite, est créé par 'texter' (cf. §2.2). Les zones de mémoire centrale affectées aux pointeurs sont libérées par 'fermer'. On peut poursuivre avec d'autres données.

```
program discri;
uses memtypes,quickdraw,osintf,toolintf,sane,uver;
var
  i,c,ci,dir,a,amax,aimax,ncol,nli,long,u,
  cari,carc,carf,carel,erl:integer;
  fin:file of integer;
  ft:text;res,d2,dmin:extended;dmil:longint;
  Fcai,Fcac,Fcae:array[1..aamax]of pti;
  cdi,dmni,cadc,idr,prdc,rpdc:ptni;
  sisi,sisc,sise:pzgi;
  rpc,rpz,carac,che,chh,chv,chc:char;
  sig,sii,sic,sie:zigle;
  nomf,nomba,titre,chaine:string;
  sfx:string;
  riri:ricla;ptrcl:ptrc;
```

1 Choix et acquisition des données de l'analyse discriminante

Nous ne commenterons pas la liste des variables déclarées en tête de 'discri': le lecteur attentif y reconnaîtra des notations déjà vues dans 'qori' et 'qorlsup'.

1.1 Choix du tableau de base, de l'ensemble des centres et de celui individus: la procédure 'principer'

```
{$S initium}
procedure vericher;begin
  verichaine(stringptr(@nomba),stringptr(@sfx),rpc);end;
procedure principer;begin
 writeln('ce programme affecte un ensemble d''individus');
 writeln('à un ensemble de centres');
 rpc:='N';erl:=0;
 while not ((rpc='O') or (erl=6)) do begin rpc:='*';
   write('le nom de base des fichiers est ');readln(nomba);
   write('le sigle de l''ensemble des centres est ');readln(chaine);
   if (4<length(chaine)) then chaine:=copy(chaine,1,4);sic:=zigler(chaine);
   write('le sigle de l''ensemble des individus est ');readln(chaine);
   if (4<length(chaine)) then chaine:=copy(chaine,1,4);sii:=zigler(chaine);
   sfx:=concat(sigler(sic),'Fac.w');vericher;
   sfx:=concat(sigler(sii),'Fac.w');vericher;
   if (rpc='N') then erl:=erl+1 else begin writeln(' BIEN');
      write('ce choix est-il confirmé  O ou N ');
      readln(rpc) end end;end;
```

Pour vérifier l'existence d'un fichier, 'principer' appelle, avec des arguments transmis par 'vericher', la procédure 'verichaine' décrite dans IA§1.3.2.

Ainsi, on peut vérifier commodément l'existence d'une suite de fichiers (ici, il n'y en a que deux) dont les noms sont construits en ajoutant divers suffixes à un même nom de base. Si un fichier manque, la suite des vérifications s'interrompt par un commentaire d'erreur; sinon, avant de passer à un nouveau suffixe, la procédure 'verichaine' écrit 'BIEN' à la fin de la ligne de la précédente demande.

On notera que 'principer' accepte que le même sigle soit donné pour l'ensemble des centres et pour celui des individus; dans ce cas, cf. *infra* §2.1, tout individu est affecté à l'individu, autre que lui, dont il est le plus proche; et, en tête du listage des résultats, l'utilisateur est averti de cette particularité.

1.2 Choix du nombre de facteurs: la procédure 'carfer'

```
procedure carfer;begin  ptrc1:=ptrc(@riri);
 reset(fin,concat(nomba,sigler(sic),'Fac.w'));
 for u:=1 to titab+1 do read(fin,c);amax:=c-2;
 close(fin);
 reset(fin,concat(nomba,sigler(sii),'Fac.w'));
 for u:=1 to titab+1 do read(fin,c);aimax:=c-2;
 close(fin);
 if not(aimax=amax) then begin rpc:='N';
    writeln('ERREUR: le nombre des facteurs sur le fichier');
    writeln('n''est pas le même pour les deux ensembles') end;
 if (amax=aimax) then begin dir:=7+2*amax;
    writeln('nombre de facteurs disponibles =',amax:4);
    write('nombre de facteurs à utiliser = ');readln(carf);
    if (carf<1) then carf:=1;if (amax<carf) then carf:=amax;
    rpz:='O';if (sigler(sii)=sigler(sic)) then rpz:='N';
    write('largeur, en caractères, du tableau des résultats = ');
    readln(long);
    if (long<12) then long:=12;if (136<long) then long:=136 end;end;
```

Le même espace est affecté à la structure 'riri', de type 'ricla', déclarée en tête du programme, et à un tableau unidimensionnel (ou séquence) d'entiers, vers lequel pointe 'ptrc1'. De la sorte, les informations, lues comme des entiers sur un fichier 'Fac.w', pourront être comprises, sous le format 'ricla' comme étant des sigles et des coordonnées réelles.

Ainsi qu' on l'a annoncé au §0, 'carfer' refuse les ensembles proposés si le nombre des facteurs disponibles n'est pas le même pour Ic et Ii; ce qui, compte tenu de la structure des programmes (tels que 'qorlsup') qui créent des fichiers 'Fac.w', implique qu'à l'origine de Ic et Ii, il y a deux analyses distinctes du même tableau de base.

Après le nombre, carf, des facteurs à utiliser dans le calcul de distance, l'utilisateur fixe la largeur du tableau des résultats. Avec une largeur de 12, il y a, par ligne, un seul bloc d'affectation: ii -> ic (cf. *supra*, §0; et *infra*, §2.2); ce qui permet d'effectuer facilement divers tris.

1.3 Lecture des facteurs pour l'ensemble des centres, Ic ; ou des individus, Ii : la procédure 'liFer'

```
procedure liFer;begin
 reset(fin,concat(nomba,sigler(sie),'Fac.w'));
 for u:=1 to titab do read(fin,c);carel:=c;read(fin,a);
 for u:=1 to 3*a do read(fin,c);
 with riri do begin
  for a:=1 to carf do Fcae[a]:=pti(newptr(4*carel));
  if (ibmax<carel) then new(sise) else sise:=pzgi(newptr(6*carel));
  for i:=1 to carel do begin
    for a:=1 to dir do read(fin,ptrc1^[a]);sise^[i]:=sis;
    for a:=1 to carf do Fcae[a]^[i]:=Fac[a] end;end;close(fin);
 writeln('liFer',sigler(sie):6,' carel = ',carel:4); end;
```

La lecture des facteurs par 'liFer' est plus simple, ici, qu'elle ne l'est dans 'qorlsup', par 'liPher', (cf. ID§3.1): car 'discri' utilise les facteurs F_α, tels quels; tandis que, pour 'qorlsup', 'liPher', (ainsi que le rappelle son sigle), calcule les facteurs φ_α, de variance 1.

2 Calcul et présentation des résultats

2.1 Affectation des individus aux centres et affichage à l'écran: la procédure 'afficher'

```
{$S actio}
procedure affecter;begin
 cdi :=ptni(newptr(2*cari));idr :=ptni(newptr(2*cari));
 dmni:=ptni(newptr(2*cari));
 cadc:=ptni(newptr(2*carc));
 prdc:=ptni(newptr(2*carc));rpdc:=ptni(newptr(2*carc));
 for c:=1 to carc do cadc^[c]:=0;
 for i:=1 to cari do begin dmin:=1000000000;ci:=1;
   for c:=1 to carc do begin d2:=0;a:=0;
     while (a<carf) and (d2<dmin) do begin a:=a+1;
        d2:=d2+sqr(Fcai[a]^[i]-Fcac[a]^[c]) end;
     if (rpz='N') and (i=c) then d2:=dmin;
     if (d2<dmin) then begin dmin:=d2;ci:=c end end;
   cdi^[i]:=ci;cadc^[ci]:=cadc^[ci]+1;
   dmil:=trunc(100*dmin);
   if (1000<dmil) then dmil:=1000;if (dmil<0) then dmil:=0;dmni^[i]:=dmil;
   writeln(' (',sigler(sisi^[i]):4,'->',sigler(sisc^[cdi^[i]]):4,') 100*dmin2 =',dmil:7) end;
 i:=0;
 for c:=1 to carc do begin rpdc^[c]:=i;prdc^[c]:=i+1;i:=i+cadc^[c] end;
 for i:=1 to cari do begin c:=cdi^[i];
    rpdc^[c]:=rpdc^[c]+1;idr^[rpdc^[c]]:=i end;end;
```

La procédure crée d'abord six tableaux d'entiers (type 'ptni', déclaré dans 'uver.p': cf. IA§1.2), dont la taille correspond à l'ensemble Ii des individus (2.cari octets, pour loger cari entiers) et à l'ensemble Ic des centres (2.carc octets). Le tableau cadc, dont le sigle doit être compris: cardinal du sous-ensemble des individus affectés au centre c, est mis à zéro; avant d'être empli au cours de la boucle d'affectation de Ii: for i:=1 to cari do...

Pour chaque individu, on parcourt l'ensemble des centres, en cherchant celui réalisant la distance carrée minima: d2 = dmin. Plus précisément le calcul de d2, pour un centre c particulier, est interrompu si la somme partielle dépasse le minimum provisoire trouvé en parcourant Ic. Dans le cas particulier où Ii=Ic, la procédure 'carfer' a donné au caractère rpz la valeur 'N' (Non); et le centre c=i est exclu de la recherche du minimum.

À l'issue de la boucle: for i:=1 to cari do... , le tableau cdi donne en cdi^[i], le numéro du centre, c, auquel est affecté l'individu i. La distance (carrée) d'affectation (évaluée en centièmes, et bornée à 1000) est dans dmni^[i]. Le nombre des individus affectés à c est dans cadc^[c]. Au fur et à mesure de l'affectation, les résultats afférents à l'individu i, qui vient d'être traité, sont affichés, à l'écran, sur une ligne.

La procédure 'affecter' se termine par deux boucles: for:=1 to carc do... et: for i:=1 to cari do..., dont l'effet se comprend relativement à un nouveau rangement de l'ensemble Ii des individus, déterminé par l'affectation à Ic. De façon précise, sont pris d'abord les individus de Ii affectés au centre c=1, dans

l'ordre où ils se trouvent initialement dans Ii. Puis viennent, les idividus i affectés à c=2 (i.e. tels que cdi^[i]=2), pris, de même, en conservant pour eux l'ordre initial. Et semblablement, ensuite, pour les i tels que cdi^[i]=3; etc..., jusqu'aux i affectés au dernier centre numéroté carc. On a ainsi carc blocs consécutifs; à ceci près que, si cadc^[c]=0, le bloc de rang c est vide.

La procédure 'affecter' fait en sorte qu'on trouve, dans idr^[r], le numéro initial, i, de l'individu qui, après constitution des blocs, a pour rang r..

D'abord, la boucle: for c:=1 to carc do begin rpdc^[c]:=i..., met dans rpdc^[c] le rang r qui précède immédiatement le premier élément du bloc c (à supposer que celui-ci ne soit pas vide; autrement dit: rpdc^[c] est le rang du dernier élément de la réunion des blocs c'<c). Si c=1, ce rang est 0. Simultanément, on met dans prdc^[c] le rang r qui peut être celui du premier élémént du bloc c (à supposer que celui-ci ne soit pas vide). En bref, on a, de 1 à cari, mis les bornes qui délimitent les blocs, dont il reste à parcourir le contenu.

Tel est le rôle de la boucle: for i:=1 to cari do begin c:=cdi^[i].... On imaginera que les éléments i, se présentant dans leur ordre naturel initial, viennent remplir carc blocs initialement vides; la place à prendre, r = 1+prdc^[cdi^[i]], étant attribuée à i, on en déduit que, réciproquement, idr^[r]=i. Quand s'achève la boucle, rpdc^[c] finit par être le dernier rang du bloc c; et idr^[r] dit quel individu i doit occuper rang r, dans le numérotage du contenu des blocs consécutifs.

Dans la suite, pour créer le listage des résultats, le tableau rpdc ne sera plus utilisé.

2.2 Création du listage des affectations des individus aux centres: la procédure 'texter'

Après un titre, le listage d'analyse discriminante, comprend trois tableaux successifs.

Dans le premier tableau, construit par la boucle: for i:=1 to cari do..., on a toutes les affectations dans l'ordre des éléments de Ii, e.g.: (µ01–>CH14)(µ02–>Or54)(µ03–>Or02)(µ04–> µ21)... Le nombre d'affectations portées sur une ligne dépend de la largeur spécifiée par l'utilisateur (cf. *supra* §1.2).

Dans le second tableau, construit par la boucle: for c:=1 to carc do..., les affectations sont données dans l'ordre de l'ensemble, Ic, des centres. Pour chacun des centres c faisant effectivement l'objet d'affectations (cadc^[c]>0), on a la suite des sigles des i qui y sont affectés. Cette suite n'est autre qu'un des blocs considérés au §2.1. Pour le bloc c, le rang r varie de prdc^[c] à prdc^[c]+cadc^[c]-1 (on a donc un intervalle vide si cadc^[c]=0). Le tableau

```
procedure texter;begin
 rewrite(ft,concat(nomba,'aff',sigler(sii),'->',sigler(sic)));
 writeln(ft,nomba,' ; nombre de facteurs utilisés =',carf:3);
 writeln(ft,'affectation des ',sigler(sii),' aux ',sigler(sic));
 if (rpz='N') then
  writeln(ft,'ATTENTION: un individu ne peut être affecté à lui-même');
 ncol:=long div 12;
 for i:=1 to cari do begin
    write(ft,'(',sigler(sisi^[i]):4,'->',sigler(sisc^[cdi^[i]]):4,')');
    if (i mod ncol=0) then writeln(ft) end;
 if not(cari mod ncol=0) then writeln(ft);ncol:=long div 5;
 for c:=1 to carc do if not (cadc^[c]=0) then begin writeln(ft);
    writeln(ft,c:3,'  liste des ',sigler(sii),' affectés à ',sigler(sisc^[c]));
    for i:=1 to cadc^[c] do begin
        write(ft,sigler(sisi^[idr^[prdc^[c]+i-1]]):5);
     if (i mod ncol=0) then writeln(ft) end;
    if not(cadc^[c] mod ncol=0) then writeln(ft) end;
 writeln(ft,'tableau à deux colonnes donnant le numéro du');
 writeln(ft,'centre c auquel on affecte l''individu i');
 writeln(ft,'et le carré, x cent, de la distance d''affectation');
 writeln(ft,cari:6,'002',sigler(sic):5,' daff');
 for i:=1 to cari do
    writeln(ft,sigler(sisi^[i]):4,'   ->',cdi^[i]:5,' daff =',dmni^[i]:6);
 close(ft);end;
```

idr donne le rang initial dans Ii: idr^[r]; et le sigle correspondant n'est autre que: sigler(sisi^[idr^[r]].

Le troisième tableau est présenté de telle sorte qu'il puisse être lu par la procédure 'litab' (cf. IA§2); ce qui permet de l'introduire facilement dans des traitements ultérieurs. L'ensemble des individus n'est autre que Ii; il y a deux variables (carj=2) qui sont le numéro du centre et la distance d'affectation (au carré).

Après trois lignes, (ne contenant pas de chiffres), qui expliquent le contenu du tableau, est donné le nombre (1000×cari) + carj (ce qui, selon IA§2.5.2, donne à 'litab' le format de lecture); suivi des noms des colonnes: respectivement le sigle de Ic, et 'daff' (qu'on lira distance d'affectation). Sur la ligne afférente à chaque individu, on a pu, sans compromettre la lecture par 'litab', interposer une flèche entre le sigle de l'individu et le numéro du centre; puis, après le centre, avant la valeur de 'daff', la mention:' daff =' .

Le §4 offre un exemple de listage d'affectation.

3 Libération des zones affectées, en mémoire centrale, aux données et aux résultats de l'analyse discriminante: la procédure 'fermer'

```
procedure fermer;begin dispose(sisi);dispose(sisc);
  dispose(cdi);dispose(dmni);dispose(cadc);
  dispose(idr);dispose(prdc);dispose(rpdc);
  for a:=1 to carf do begin dispose(Fcai[a]);dispose(Fcac[a]) end;end;
```

Ainsi qu'on l'a dit au §0, *in fine*, la procédure 'fermer' libère l'espace réservé en mémoire centrale pour garder les sigles des éléments des ensembles Ic et Ii, les tableaux de facteurs, les résultats de l'analyse discriminante.

4 Appendice: exemple de listage produit par le programme 'discri'

```
Disq:Dos:Pht ; nombre de facteurs utilisés = 10
affectation des πVb aux Vb
(  V1->i560)(  V2->i364)(  V3->i364)(  V4->i364)(  V5->i194)

194  liste des πVb affectés à i194
   V5

368  liste des πVb affectés à i368
   V2   V3   V4

487  liste des πVb affectés à i487
   V1
tableau à deux colonnes donnant le numéro du
centre c auquel on affecte l'individu i
et le carré, x cent, de la distance d'affectation
     5002   Vb daff
  V1   ->  487 daff =    76
  V2   ->  368 daff =    57
  V3   ->  368 daff =    43
  V4   ->  368 daff =    61
  V5   ->  194 daff =    54
```

Les ensemble de colonnes: πVb et Vb, sont adjoints en supplément à l'analyse du tableau: Disq:Dos:Pht. L'ensemble: Vb, compte plusieurs centaines d'individus, dont chacun a pour sigle la lettre i, suivie de son numéro de dossier. Les cinq éléments de: πVb, sont notés: {V1, V2, V3, V4, V5}.

Le listage d'affectation a pour nom: Disq:Dos:PhtaffVb->πVb.

Les 5 formules d'affectation: (Vi->ixyz) y tiennent sur une seule ligne.

Le bilan des affectations par centre (ici, les centres sont les dossiers individuels), ne comporte que trois listes (et non plusieurs centaines); parce que seuls: {i194, i369, i487}, font l'objet d'une affectation (ou de plusieurs).

Le tableau (dont le listage commence au mot 'tableau') sera lu, dans nos programmes, par la procédure 'litab' (cf.IA§2), comme s'il n'était autre que:

```
tableau à deux colonnes donnant le numéro du
5002   Vb daff
  V1  487   76
  V2  368   57
  V3  368   43
  V4  368   61
  V5  194   54
```

les flèches et la mention: 'daff =', sont éliminées, ainsi que le serait tout commentaire ne comportant pas de chiffre.