Les Cahiers de l'Analyse des Données
Vol. XXII - 1997 - n°2 - pp.167-190.

SOURCES DE PROGRAMMES D'ANALYSE DE DONNÉES EN LANGAGE PASCAL: (II) : ÉLABORATION DES FICHIERS DE FACTEURS (IID) : VACOR: AIDE À L'INTERPRÉTATION EN CLASSIFICATION ASCENDANTE HIÉRARCHIQUE D'APRÈS LE TABLEAU DES DONNÉES

[SOURCES PASCAL (IID)]

J.-P. & F. BENZÉCRI

0 Introduction : informations traitées pour l'interprétation d'une partition par VACOR, en terme de variables

Ainsi qu'on l'a dit en IIB§0, avec VACOR, on remonte aux variables de base du tableau des données, pour considérer le système de coordonnées orthogonales que constituent les composantes elles-mêmes de la ligne ou de la colonne associée à tout élément, e, de E; voire les cumuls par classes de ces composantes.

Prenons, afin de fixer les notations, un ensemble E dont le sigle est 'sa'. Soit: 'Dur:tab', le nom du tableau de base: alors, 'Dur:tabsa' est un tableau externe de lignes supplémentaires. Convenons, également, de noter (, sauf si une confusion est possible): 'suf', un tableau dont le nom est: 'Dur:tabsuf'.

Pour soumettre à FACOR une partition de l'ensemble E, il faut (cf. §IIC1.2) avoir, avec le fichier 'saFac.w', donnée indispensable de la CAH, le fichier 'sacnabd', (créé par 'cah') décrivant la CAH; et les fichiers {'qcqabc', 'qcmabc', 'cqki'}, (créés par 'arbt') décrivant la partition de E considérée.

Les fichiers décrivant la CAH de E sont également requis pour VACOR; le fichier des facteurs n'est pas directement indispensable; mais il faut avoir le tableau 'dur:tabsa' lui même, afin de calculer les profils des classes sur l'ensemble J des colonnes. Plus précisément, puisque la CAH, est fondée sur les facteurs, calculés d'après les profils des éléments sur l'ensemble Jp des variables principales, celles-ci doivent seules compter dans les distances entre profils; or Jp est défini (comme partie de J) par le fichier 'jsup' créé lors de l'analyse factorielle du tableau de base.

De plus, pour faire VACOR en terme de cumuls des variables, il faut avoir la CAH de celles-ci.

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

Nous avons d'abord fait des calculs d'aide à l'interprétation d'après les variables pour les deux ensembles principaux, Ip et Jp, fixés dans l'analyse d'un tableau I × J. Le programme a été ensuite étendu pour traiter des tableaux externes de lignes ou de colonnes (cf. ID): auquel cas on parle, en bref, d'ensembles Ea (tel 'sa', ci-dessus) ou Eb.

Le calcul de VACOR ne prend tout son sens que lorsque les distances, inerties et contributions (ou cosinus) sont calculées relativement à la métrique même d'après laquelle a été faite la CAH. Tel est bien le cas pour les ensembles Ip et Jp; ainsi que pour un ensemble Ea ou Eb, dont la marge, respectivement sur Jp et Ip, est la même que celle du tableau principal; notamment pour le tableau des individus, codé sous forme disjonctive complète, adjoint en supplément à l'analyse du tableau de BURT; et aussi dans la généralisation au codage barycentrique (cf. III).

Mais en général, si on applique, à un tableau externe quelconque la métrique du chi^2 afférente au tableau principal (e.g., à Ea, la métrique du chi^2 pour les profils sur Jp), il se peut qu'une variable ait un poids nul, ou négligeable, dans la marge du tableau principal; d'où, dans la métrique du chi^2, un facteur $(1/f_j)$ très fort, voire infini; mais sans effet pour l'ensemble Ip; tandis que, pour Ea, le poids de la même variable est considérable; avec, dans les distances, des termes très forts afférents à j.

Dans le programme de CAH, la métrique de base est calculée comme somme des carrés des différences d'un certain nombre de facteurs; ce calcul peut éliminer, comme un filtre, l'effet des variables de masse quasi nulle; mais le calcul direct ne le fait pas. C'est pourquoi, sans que cela s'impose, on a préféré prendre toujours, pour VACOR, la métrique du chi^2 afférente à la marge du tableau décrivant l'ensemble E considéré, lui-même, quel que soit celui-ci.

1 Structure générale du programme 'VAX'

1.1 Versions du programme de calcul pour VACOR

Créé par extensions successives, notre programme gardait pour notion de base celle de tableau principal; et proposait, dans une seule boucle, le traitement de Ip, de Jp, puis d'éventuels ensembles Ea ou Eb. L'expérience de plusieurs années nous assure que le programme ainsi conçu sert bien à l'interprétation des CAH. Mais la complexité des appels et des renvois, régis par des paramètres en forme de lettre ou de sigle, rend les sources impénétrables au lecteur qui ne les a pas lui-même écrites.

Aussi le programme 'VAX' présenté ici a-t-il été récrit pour le présent article. Il n'y a pas de boucle prétendant atteindre tous les ensembles possibles associés à un tableau de base donné; mais seulement un boucle générale, où,

à chaque itération, indépendamment des passages précédents, on choisit, à la fois, le tableau de base et l'ensemble à traiter. Sans prétendre tout essayer, nous avons, sur divers cas, vérifié que le nouveau venu a même effet que son prédécesseur. Le seul inconvénient de cette enchaînement des calculs, est que, si l'on demande, indépendamment, de faire VACOR pour i sur jq et pour j sur iq, on fait deux fois le cumul des données du tableau de base, par classes de lignes et de colonnes.

Comme dans IIB et IIC, nous considérons d'abord le programme principal, puis les procédures que celui-ci appelle successivement. Dans le même but de faciliter la lecture des sources, le programme 'VAX' est écrit d'une seule pièce, avec des déclarations générales détaillées (cf. §1,3); puis une suite de procédures, dont certaines servent d'outil aux six procédures qui sont seules appelées dans le programme principal. Il n'y a comme procédures externes, outre celles décrites dans IA§1, qu'une procédure 'critab' dont, en bref, l'effet est inverse de celui de la procédure 'litab' (IA§2), celle-là écrivant des tableaux que celle-ci est capable de lire.

1.2 Le programme principal

```
begin;benzecri;rep:='O';
while not((rep='N') or (rep='Z')) do begin erl:=0;
  choisir;unloadseg(@choisir);
  if (rep='O') then begin
    taber;unloadseg(@taber);
    lirtabler;unloadseg(@lirtabler);
    critaber;unloadseg(@critaber);chx:='Z';
    if ((carv <vcmax) and (rVv='O')) then begin chx:=' ';vacorer end;
    if ((carvq<vcmax) and (rVq='O')) then begin chx:='q';vacorer end;
    if ((chx=' ') or (chx='q')) then unloadseg(@vacorer);
    memvider;unloadseg(@memvider);
    write('faut-il faire un autre VACOR oui(O) ou non(N) ');
    readln(rep) end end;
end.
```

Le programme principal consiste simplement en une boucle 'while....

Par la procédure 'choisir', l'utilisateur désigne, d'abord, un fichier de base et un ensemble. Prenons, afin de fixer les notations, un tableau de base dont le nom est: 'Dur:tab'. Si l'ensemble E a pour sigle: 'sa', 'Dur:tabsa' est un tableau externe de lignes supplémentaires. Mais il peut s'agir d'un tableau externe de colonnes: avec pour sigle: 'eb'; ou de l'un des deux ensembles principaux, de lignes et colonnes, Ip et Jp, désignés respectivement par les sigles: 'i', 'j'.

On sait que le programme VACOR ne peut jouer pour un ensemble que si existe pour celui-ci une CAH; et, plus précisément, qu'on en a extrait une partition. D'autre part, le listage d'aide à l'interprétation n'est utile que si y apparaissent des contributions majeures; ce qu'on ne peut attendre, *a priori*, que si l'on considère un ensemble de variables, ou un ensemble de classes de variables, dont le cardinal est assez restreint. Sans prétendre trancher de façon certaine, on a pris pour maximum: vcmax=28; nombre qu'il serait facile de

modifier, mais qui nous semble dépasser ce qui a pu servir dans la pratique. Avant d'entrer dans le détail de la procédure 'choisir' (cf. §2), on comprend donc que celle-ci a pour fonctions: d'une part de vérifier l'existence de fichiers; et d'autre part, de comparer, au seuil 'vcmax', le nombre des variables principales et celui des classes d'une éventuelle partition de celles-ci.

Si VACOR peut se faire, la réponse générale est: rep='O'; et, plus précisément, on a, respectivement: rVv='O' et: rVq='O', si VACOR est demandé en terme de variables ou en terme de classes, (les deux pouvant être faits simultanément).

En ce cas, par la procédure 'taber' (cf. §3), le programme affecte, à divers tableaux, des zones de mémoire centrale. Puis, 'taber' lit, d'une part, la description de la CAH de E (avec, s'il y a lieu, la decription de la CAH des variables); et, d'autre part, ce qui concerne la distinction entre éléments principaux et supplémentaires.

Ceci fait, la procédure 'lirtabler' (cf. §4), à partir du tableau de données afférent à l'ensemble E, crée un ou deux tableaux de cumul croisant la partition de E avec l'ensemble des variables ou l'ensemble des classes de variables. La procédure 'critaber' (cf. §5) a pour rôle d'écrire sur disque les tableaux de cumul; où l'utilisateur peut trouver une aide élémentaire à l'interprétation des CAH.

La procédure 'vacorer' (cf. §6) crée le listage VACOR proprement dit. On verra qu'à cette fin, le tableau de cumul doit d'abord être transformé en tableau de profils. Puis l'on calcule des distances au sens du chi^2; et, d'après les coordonnées masses et distances, des inerties et des contributions... Selon que le caractère: chx=' ' (espace blanc), ou que: chx='q', VACOR est fait en terme de variables ou de classes de variables.

Enfin, la procédure 'memvider' (cf. §7) libère les zones de mémoire affectées par 'taber'. Et, après un VACOR mené à bien, l'occasion est offerte à l'utilisateur d'en demander un autre: ce qui conduit à rentrer dans la procédure initiale 'choisir'.

1.3 Les déclarations de constantes et de variables

Sans entrer dans les détails, on montrera ce qui est hérité des programmes de CAH; et ce qui annonce les présents calculs.

La constante: qmax=100, nombre maximum de classes pour une partition extraite d'une CAH par 'arbt', a déjà été déclarée en tête d' 'arbt' (IIB§3); puis de 'Facx' (IIC§2.1). Le double de qmax est noté: qdmax=200. Un tableau à qdmax lignes suffit pour ranger les profils des nœuds et des classes de la partition retenue pour E; c'est pourquoi qdmax donne le cardinal des colonnes des tableaux: {k1ve, k2ve}, où sont calculés les cumuls, puis les profils, sur l'ensemble des variables ou des clsses de variables. Et l'on retrouve qmax et qdmax comme dimensions des tableaux {cdq, qAm, qBm, cAq, cBq}, qui

```
program VAX;uses memtypes,quickdraw,osintf,toolintf,uver5;
const  qdmax=200;qmax=100;qvmax=56;vcmax=28;
var nombas,nomf,nomf2,chat,chax,titre:str255;
fin:file of integer;ftu,ftq:text;
rrc,rep,rec,rea,ref,rVv,rVq,rpv,che,cht,chv,chx,carac:char;
i,ii,j,iz1,iz2,er1,resu,e1,ql,e,a,c,cq,ca,cb,vx,q,n,m,dir,pp,ppp,
  amax,vrf,v0,v1,v2,CLAS,AINE,BNJM,QLT,PDS,INR,FCT,CO2,CTR,ili,jco,
  care,caret,careq,carv,carvt,carvq,carvv,
  carco,carli,carcot,carlit,sgn,eb,vb:integer;
ffs,trace:single;res,ret,fv1,dv1,ppg:extended;
ffl:longint;
cdq:array[1..qdmax] of integer;
qAm,qBm,cAq,cBq:array[1..qmax] of integer;
tampi:array[1..titab]of integer;
dv:array[1..vcmax] of single;
poig,dg,dd:array[1..qdmax] of single;
nouv:ptr;psi,ttri:stringptr;vv,ee,wv:pint;
pre,prv,qre,qrv,prt,prco,prli:ptni;
sig:zigle;sgli:pzgl;ffli:plng;wwi:pww;riri:ricla;ptrc1:ptrc;
sigle,sig1v,sig1vq,sig1vv:pzgi;klve,k2ve,kxve:kji;t1k,t2k:tk;
```

suivant un format introduit avec 'arbt' et repris dans 'Facx', décrivent les partitions; ainsi que dans les tableaux de poids et distances propres à VACOR: {poig, dg, dd}.

On a déjà dit que, pour VACOR, on limite à: vcmax=28, le nombre des variables axplicatives (ou des cumuls de variables par classe). Le nombre double: qvmax=56, est pris comme limite du nombre des colonnes des tableaux de cumuls qui peuvent être gardés sur disque, même sans faire les calculs de VACOR.

Des nombres entiers, certains ne servent que d'indices dans des boucles de calcul. D'autres, tels {iz1, iz2}, reçoivent, une fois pour toute, des valeurs qui sont conservées pour la suite. Dans {CLAS, AINE, BNJM, QLT, PDS, INR, FCT, CO2, CTR} est préparé le calcul des lignes successives d'un tableau du listage VACOR. De nombreux entiers sont déclarés avec des noms commençant par 'car': il s'agit des cardinaux de divers ensembles: d'éléments (care) ou de variables (carv), principaux ou quelconques (caret: avec 't' pour: total), ou de classes (careq: classes d'éléments; carvq: classes de variables.

Les calculs de VACOR ont cette particularité que, dans le tableau sur lequel on effectue les cumuls, les variables ne figurent pas toujours comme colonnes: tel est le cas pour les ensembles externes, Ea ou Eb; ainsi que pour Ip, mais non pour Jp (ensemble relativement auquel Ip tient lieu d'ensemble des variables). Afin d'effectuer toujours les calculs par les mêmes procédures, on a multiplié les pointeurs. Par exemple, les pointeurs pre et prv donnent respectivement accès à l'indice des éléments ou variables principaux; mais, relativement aux lignes et colonnes du tableau dont le contenu fait l'objet de cumuls, on a les pointeurs prli et prco. De même, le pointeur sig1vv, donne, selon les cas, accès aux sigles des variables individuelles (sig1vv:=sig1v) ou des sigles de classes de variables (sig1vv:=sig1vq).

2 Dialogue du choix des données: la procédure 'choisir'

2.1 Procédures appelées par 'choisir'

```
{$S segchoi}
procedure verficher(sfx:str255);begin
if (resu=1) then begin
  nomf:=concat(nombas,chax,sfx);resu:=verif(@nomf) div 2;
  writeln(' resu =',resu:2,' pour ', nomf);end;end;
procedure verarber;begin resu:=1;
verficher('cnabd');
verficher('qcqabc');verficher('qcmabc');verficher('cqki');end;
```

Par la procédure 'verarber', qui appelle 'verficher', le programme vérifie la présence des fichiers décrivant une CAH. On a procédé de même dans 'FAX' (cf. IIC§1.2); à la seule différence près que le fichier de facteurs: 'Fac.w' , n'est pas indispensable pour 'VAX'. Dans la mesure où l'on s'est assuré, préalablement, de la présence du disque, la fonction 'verif' ne peut valoir zéro; mais seulemsnt 2 ou 1, selon que le fichier considéré est, ou non, présent; c'est pourquoi le résultat: resu, vaut 1 ou zéro, après division par 2.

2.2 Début de la procédure 'choisir': choix du tableau de base et l'ensemble considérés

```
procedure choisir;begin rep:='N';
while not((rep='O') or (er1>5)) do begin
 rep:='O';
 write('le fichier de base est ');
 readln(nombas);
 if not(setvol(@nombas,10)=noerr) then begin rep:='Z';er1:=er1+1;
    writeln('ERREUR le disque manque') end;
 if (rep='O') then begin
    writeln('le sigle peut être: i ou j,');
    writeln('ou un mot de 1, 2 ou 3 lettres terminé par a ou b');
    write('le sigle de l''ensemble à traiter par VACOR est ');
    readln(chat);ii:=length(chat);che:=chat[ii];rep:='N';
    if ((ii<4) and ((chat='i') or (chat='j') or (che='a') or (che='b')))
      then rep:='O' end;
 if (rep='N') then begin er1:=er1+1;rep:='Z';
      writeln('ERREUR sur la forme du sigle') end;
```

La procédure 'choisir' demande à l'utilisateur de fixer le fichier de base, puis l'ensemble à traiter par VACOR. Ensuite, on s'enquiert de la présence des fichiers de données; du nombre des variables et, éventuellement, des classes de variables. Chaque vérification peut aboutir à un échec; en ce cas, la réponse: rep, est d'abord mise à: 'N'; sous cette condition, s'affiche à l'écran un commentaire d'erreur; le nombre: er1, est augmenté de 1; et rep est mise à 'Z', ce qui arrête la suite des essais et renvoie au début de la boucle: 'while not...'.

Dans le bloc ci-dessus, le nom: nombas=disq:tab, selon l'exemple du §0, offre matière à vérifier l'existence du disque. Puis le sigle de l'ensemble E éatnt lu dans la chaîne de caractères:chat, on s'assure d'abbord que la forme en est correcte.

2.3 Recherche du format du tableau numérique afférent à l'ensemble E choisi

```
if (rep='O') then begin
   nomf2:=concat(nombas,chat);
   if ((che='i') or (che='j')) then nomf2:=nombas;nomf:=nomf2;
   iz1:=verif(stringptr(@nomf));iz2:=0;
   if (iz1=1) then begin nomf:=concat(nomf2,'yy');
     iz2:=verif(stringptr(@nomf));
      if (iz2=2) then begin iz1:=2;iz2:=3 end end;
   if (iz1=1) then begin nomf:=concat(nomf2,'.z');
     iz2:=verif(stringptr(@nomf)) end;
   if (iz2=1) then nomf:=concat(nomf2,'.w');
   rep:=dialof(stringptr(@nomf)) end;
if (rep='N') then begin er1:=er1+1;rep:='Z';
   writeln('il n''y a pas de tableau pour VACOR') end;
```

Dès IA§2, on a vu que la procédure 'litab' admet quatre format de tableaux de données, distingués par des suffixes. D'une part, deux formats de texte: tableau d'entier (sans suffixe) et tableaux de réels écrits en format quelconque (suffixe:'yy'). D'autre part deux formats binaires; distingués selon que les nombres, écrit chacun comme deux entiers succesifs ('integer'), doivent être lus comme un entier long (format: 'longint'; suffixe: '.z'); ou comme un réel (format: 'single'; suffixe: '.w').

Il faut, d'autre part, prendre garde que le nom: nomf2 (sans suffixe), du tableau numérique est: nombas, si E=Ip ou Jp (chat='i' ou 'j'); mais qu'autrement, il s'agit d'un tableau externe: nomf2:=concat(nombas, chat).

Finalement, l'entier: iz2, donne le premier format trouvé: 0 ou 3 ('yy') s'il s'agit d'un texte; 2 ('.z') ou 1 ('.w') pour le cas binaire. De plus l'entier: iz1 distingue entre texte (iz1=2); et binaire (iz1=1)..

2.4 Existence d'une partition pour E et fichiers d'éléments principaux

Pour les vérifications, l'ensemble des variables ainsi que le type de E sont notés par un caractère unique, che ou chv. Dans la cas particulier où E est Ip ou Jp (chat=che: 'i' ou 'j'), on doit vérifier l'existence d'un fichier 'pr' non seulement pour l'ensemble des variables mais pour E.

```
if (rep='O') then begin
   writeln(nomf);
   chax:=chat;verarber;if (resu=0) then rep:='N' end;
if (rep='N') then begin er1:=er1+1;rep:='Z';
   writeln('ERREUR il n''y a pas de CAH pour l''ensemble à traiter') end;
if (rep='O') then begin
   if ((che='i') or (che='a')) then chv:='j' else chv:='i';
   nomf:=concat(nombas,chv,'pr');resu:=verif(stringptr(@nomf));
   if not(resu=2) then rep:='N' end;
if (rep='N') then begin er1:=er1+1;rep:='Z';
   writeln('ERREUR on ne connaît pas les var principales') end;
if (rep='O') and ((che='i') or (che='j')) then begin
   nomf:=concat(nombas,che,'pr');resu:=verif(stringptr(@nomf));
   if not(resu=2) then rep:='N' end;
if (rep='N') then begin er1:=er1+1;rep:='Z';
   writeln('ERREUR on ne connaît pas les élém principaux de ',chat) end;
```

2.5 Dénombrement des variables et des classes de variables

```
if (rep='O') then begin carv:=0;carvt:=0;
   reset(fin,concat(nombas,chv,'pr'));
    while not eof(fin) do begin read(fin,i);
      carv:=carv+i;carvt:=carvt+1 end;
   close(fin);chax:=chv;verarber;carvq:=1000;
   if (resu=1) then begin nomf:=concat(nombas,chv,'cqki');
     reset(fin,nomf);read(fin,carvq);close(fin) end;
   if ((vcmax<carv) and (qvmax<carvq)) then begin rep:='Z';er1:=er1+1;
     writeln('ERREUR impossible de faire VACOR');
     writeln('car',chv,'p =',carv:4,' > ',vcmax:3);
     if (carvq=1000) then
       writeln('et il n''y a pas de CAH des chv');
     if (carvq<1000) then
       writeln('et car',chv,'q =',carvq:3,' > ',qvmax:3) end end;
```

En lisant le fichier 'pr' approprié, on calcule le nombre: carvt, des variables, parmi lesquelles on distingue: carv, nombre des variables principales.

On s'enquiert ensuite de l'existence d'une partition de l'ensemble des variables; et s'il y a lieu, on note dans carvq le nombre des classes de cette partition (sinon, carvq est, arbitrairement, mis à la valeur 1000).

Finalement, compte tenu des bornes fixées, on ne peut faire les calculs de VACOR sur les variables que si carv n'excède pas: vcmax=28. De même, VACOR sur les classes, que si carvq n'excède pas ce même nombre. Plus précisément, si carvq≤qvmax=56, on considère l'éventualité de créer un simple tableau de cumul. Si aucune voie ne reste ouverte, le choix aboutit à un échec, noté comme: rep:='Z'.

2.6 Choix des traitements à faire parmi ceux possibles

Ainsi qu'on l'a annocé au §1.2, en commentant le programme principal, la création d'un ou de deux listages VACOR, sur les variables ou les classes de variables est arrêté dans les réponses: rVv et rVq; réponses dont le choix n'est laissé à l'utilisateur que si les nombres: carv et carvq le permettent. Plus précisément, il peut s'agir non d'un listage VACOR, mais seulement d'un tableau de cumul.

```
if (rep='O') then begin rVv:='N';rVq:='N';
   if (carv<vcmax+1) then begin
     write('faut-il faire VACOR sur ',chv,' pour les ',chat,' O ou N ');
     readln(rVv);if not(rVv='N') then rVv:='O' end;
   if (carvq<vcmax+1) then begin
     write('faut-il faire VACOR sur ',chv,'q pour les ',chat,' O ou N ');
     readln(rVq);if not(rVq='N') then rVq:='O' end;
   if (vcmax<carvq) and (carvq<qvmax+1) then begin
     writeln('on ne peut pas faire VACOR, proprement dit');
     write('faut-il faire le tableau de cumul ',chat,'q x ',chv,'q O ou N ')
     readln(rVq);if not(rVq='N') then rVq:='O' end;
   write('tous ces choix sont-ils confirmés oui(O) ou non(N) ');
   readln(rep);if ((rVv='N') and (rVq='N')) then rep:='N'; end end
end;{choisir}
```

3 Affectation de zones en mémoire centrale; lecture des partitions; caractérisation des éléments principaux: la procédure 'taber'

3.1 Procédures appelées par 'taber'

3.1.1 Création d'un sigle pour un nombre entier: la procédure 'codi'

```
{$S segtab}
procedure codi(i0:integer);begin sig[0]:=1;sig[1]:=ord(cht);
 if (999<i0) then sig[1]:=48+((i0 div 1000) mod 10);
 if (99<i0)  then begin sig[0]:=2;
      sig[2]:=48+((i0 div 100) mod 10);end;
 if (9<i0)   then begin sig[0]:=sig[0]+1;
      sig[sig[0]]:=48+((i0 div 10) mod 10);end;
 sig[0]:=sig[0]+1;sig[sig[0]]:=48+(i0 mod 10);
end;
```

Le sigle est créé sous le format 'zigle' (suite d'octets: cf. IA§1.2). Destiné à une classe, il commence par le caractère cht propre à l'ensemble considéré; ce caractère n'est supprimé que si le nombre dépasse 1000.

3.1.2 Affectation de zones en mémoire centrale: la procédure 'initaber'

```
procedure initaber;begin
 new(pre);new(prv);new(qre);new(qrv);new(prt);
 nouv:=newptr(6*qdmax);sigle :=pzgi(nouv);
 nouv:=newptr(6*qvmax);siglv :=pzgi(nouv);
 nouv:=newptr(6*qvmax);siglvq:=pzgi(nouv);
 for j:=0 to qvmax do begin
   nouv:=newptr(4*qdmax);k1ve[j]:=pti(nouv);t1k[j]:=nouv; end;
 for j:=0 to qvmax do for i:=1 to qdmax do k1ve[j]^[i]:=0;
 for j:=0 to qvmax do begin
   nouv:=newptr(4*qdmax);k2ve[j]:=pti(nouv);t2k[j]:=nouv; end;
 for j:=0 to qvmax do for i:=1 to qdmax do k2ve[j]^[i]:=0;
 writeln('fin de l''initialisation des tableaux ');
end;
```

Cinq tableaux: {pre^, prv^, qre^, qrv^, prt^}, sont créés avec la dimension maxima: ipmax=8000, fixée pour les tableaux d'entiers (cf. IA§1.2). En effet, pre et prv donnent accès à la caractérisation des éléments principaux dans E et dans l'ensemble des variables qui le décrivent: or l'un ou l'autre de ces ensembles peut avoir le cardinal maximum; par exemple si E=Ip; ou, si E=Jp, pour l'ensemble des variables qui est alors J. De même, cf. *infra*, qre et qrv indiquent la répartition des éléments ou variables par classes. Enfin, prt, donne accès à une zone auxiliaire.

En revanche, les ensembles de sigles, qui ne servent ici que pour les tableaux cumulés, ont leur cardinal limitée par qdmax (nœuds et classes de la partition de E), ou qvmax (variables ou classes de variables servant pour VACOR).

De même, pour les données numériques cumulées, les tableaux: k1ve et k2ve, ont au plus qvmax colonnes utiles, dont chacune renferme, au plus, qdmax nombres réels. Ces tableaux, sitôt créés, sont mis à zéro.

3.2 Éléments principaux et partitions

3.2.1 L'ensemble V des variables explicatives

```
procedure taber;begin
 initaber;
 for j:=1 to ipmax do prt^[j]:=0;
 for j:=1 to ipmax do qrv^[j]:=0;
 nomf:=concat(nombas,chv,'pr');
 reset(fin,nomf);j:=0;
 while not eof(fin) do begin read(fin,i);j:=j+1;prv^[j]:=i;end;
 close(fin);{carvt:=j;}
 if (rVq='O') then begin
   nomf:=concat(nombas,chv,'cqki');
   reset(fin,nomf);read(fin,carvq);{carv:=0;}
   for q1:=1 to carvq do begin
      read(fin,c);read(fin,cq);cq:=cq-10000;{carv:=carv+cq;}
      for e1:=1 to cq do begin read(fin,c);prt^[c]:=q1 end;end;
   close(fin);vx:=0;
   for j:=1 to carvt do if (prv^[j]=1) then begin
      vx:=vx+1;qrv^[j]:=prt^[vx] end;end;
```

Initialement, sont mis à zéro le tableau auxiliaire: prt^, et le tableau: qrv^, destiné à définir le contenu d'éventuelles classes de variables. L'inventaire des variables principales est écrit dans: prv^. On notera que le fichier copié ici a été ouvert pour 'choisir' (cf. §2.5); en sorte que le nombre total: carvt, est déjà connu; ainsi que celui: carv, des variables principales. Si on doit utiliser une partition des variables principales (rVq='O'), celle-ci, lue dans 'cqki', est d'abord mise dans le tableau auxiliaire: prt^, comme la suite des numéros de classe afférents aux variables. Mais il reste à conjuguer toutes les informations lues pour inscrire, dans qrv^, la partition de l'ensemble de toutes les variables (celles en supplément gardant: zéro, pour numéro de clase).

3.2.2 La partition, à expliquer, des éléments de l'ensemble E

```
 for j:=1 to ipmax do prt^[j]:=0;
 for j:=1 to ipmax do qre^[j]:=0;
 for j:=1 to ipmax do pre^[j]:=1;
 nomf:=concat(nombas,chat,'cqki');
 reset(fin,nomf);read(fin,careq);care:=0;
 for q1:=1 to careq do begin
    read(fin,c);read(fin,cq);cq:=cq-10000;care:=care+cq;
    for e1:=1 to cq do begin read(fin,c);prt^[c]:=q1 end;end;
 close(fin);caret:=care;
 if ((che='i') or (che='j')) then begin
   nomf:=concat(nombas,che,'pr');
   reset(fin,nomf);j:=0;
   while not eof(fin) do begin read(fin,i);j:=j+1;pre^[j]:=i;end;
   close(fin);caret:=j;end;
 vx:=0;
 for j:=1 to caret do if (pre^[j]=1) then begin
      vx:=vx+1;qre^[j]:=prt^[vx] end;
```

Comme pour l'ensemble V, sont, d'abord, mis à zéro le tableau auxiliaire: prt^, et le tableau: qre^, destiné à définir le contenu des classes d'éléments de l'ensemble E.

De plus, est mis à 1 l'inventaire: prv^, des éléments principaux de E. En effet, la distinction entre principal et supplémentaire n'existe pour E que si E est l'un des ensembles croisés dans le tableau de base: E=Ip, ou: E=Jp (autrement dit: chat=che='i', ou: chat=che='j').

Dans tous les cas, la partition de E, lue dans 'cqki', est d'abord mise dans le tableau auxiliaire: prt^, comme la suite des numéros de classe afférents aux éléments. Ainsi est calculé le nombre: care des éléments (principaux) de E.

Alors, le tableau: pre^, est, éventuellement, corrigé d'après le fichier: 'ipr', ou: 'jpr'; des zéros étant introduits à la place des éléments supplémentaires.

Dans tous les cas, le traitement de E s'achève, comme celui de V, en conjuguant toutes les informations lues pour inscrire, dans qre^, la partition de l'ensemble E (les éléments en supplément gardant: 0, pour numéro de clase).

3.2.3 Structure hiérarchique des partitions et numérotage des classes

Le fichier 'qcmabc' (cf. IIB§3.3.3) décrit l'arborescence restreinte, afférente à la partition de E retenue. Après la valeur 10000+careq-1, on a une suite de triplets {qA, qB, cdq}, pour les careq-1 nœuds (rangés de haut en bas): cdq, numéro du nœud dans l'ensemble Eé (étendu: de 1 à (2.care)-1); qA et qB, numéros de ses deux descendants, considérés comme éléments de la hiérarchie restreinte (numéros de 1 à (2.careq)-1)

Le fichier 'qcqabc' (cf. IIB§3.3.3) contient, après la valeur careq+10000, une suite de carq triplets {cA, cB, cdq}, afférents aux classes la partition: cdq, est le numéro de la classe dans Eé; et, s'il s'agit d'un nœud, cA et cB sont, dans Eé, les numéros de son aîné et de son benjamin; sinon: cA=cB=0.

Par la procédure 'codi' (cf. §3.1.1), un numéro de classe est converti en sigle (considéré comme zigle: sig), avec la lettre initiale: cht.

S'il y a lieu (i.e. si: rVq='O'), on attribue des sigles aux classes de la partition de l'ensemble V de variables (mais non aux nœuds, que le listage ne mentionne pas); en procédant comme pour E.

```
nomf:=concat(nombas,chat,'qcmabc');
reset(fin,nomf);read(fin,m);{careq:=q-9999;}
for m:=careq-1 downto 1 do read(fin,qAm[m],qBm[m],cdq[m+careq]);
close(fin);{care:=(cdq[(2*careq)-1]+1) div 2;}
nomf:=concat(nombas,chat,'qcqabc');
reset(fin,nomf);read(fin,q);{careq:=q-10000}
for q:=1 to careq do read(fin,cAq[q],cBq[q],cdq[q]);
close(fin);
cht:=che;
for q:=1 to (2*careq)-1 do begin codi(cdq[q]);sigle^[q]:=sig end;
if (rVq='O') then begin
   cht:=chv;
   nomf:=concat(nombas,chv,'qcqabc');
   reset(fin,nomf);read(fin,q);{carvq:=q-10000;}
   for q:=1 to carvq do begin read(fin,ca,cb,c);codi(c);siglvq^[q]:=sig en
   close(fin);end;
end;{taber}
```

4 Calcul par cumul des tableaux de croisement: la procédure 'lirtabler'

La procédure 'lirtabler' (cf. §4), à partir du tableau de données afférent à l'ensemble E, crée, par cumul, un ou deux tableaux croisant la partition de E avec l'ensemble des variables ou l'ensemble des classes de variables.

4.1 Procédures appelées par 'litaber'

4.1.1 Accès aux données par lignes et colonnes: la procédure 'inilir'

```
{$S seglir}
{'lirtabler'}
procedure inilir;begin
ptrc1:=ptrc(@riri);
ttri:=stringptr(@tampi);sgli:=pzgl(@tampi);
ffli:=plng(@tampi);wwi:=pww(@tampi);psi:=stringptr(@sig);
prli:=pre;prco:=prv;
ee:=pint(@i);vv:=pint(@j);wv:=pint(@jco);
carco:=carv;carli:=care;carcot:=carvt;carlit:=caret;
if (chat='j') then begin
  prco:=pre;prli:=prv;
  ee:=pint(@j);vv:=pint(@i);wv:=pint(@ili);
  carco:=care;carli:=carv;carcot:=caret;carlit:=carvt end;
end;
```

On a dit (au §2.3) que le tableau croisant les ensembles E et V, peut être en format de texte ou en format binaire. Dans ce dernier cas, les données, lues comme des entiers dans un tableau: tampi, sont interprétées en passant par des pointeurs: {ttri, sgli, ffli, wwi}, qui ont accès à: tampi. De plus, les lignes, lues globalemnt comme des suites d'entiers sous le pointeur: ptrc1, sont comprises comme une structure: ricla, comportant un sigle, suivi de données numériques.

Il importe de noter qu'il n'y a pas d'équivalence constante entre, d'une part, les lignes et colonnes du tableau: tab, croisant E et V; et d'autres part, les notions mêmes d'éléments de variables (; lesquelles sont toujours, respectivement lignes et colonnes pour les tableaux de cumul créés en mémoire centrale). De façon précise, on a en général, l'équivalence: {ee, vv} pour le tableau de cumul; {ligne:i, colonne: j} pour 'tab'. Mais si E=Jp (chat='j'), l'équivalence est renversée. Participent à l'équivalence, non seulement les indices, mais encore les nombres cardinaux et les pointeurs:pr, qui caractérisent les ensembles principaux (cf. §3.2).

4.1.2 Lecture des sigles et nombres sur un fichier en format de texte: les procédures 'lecsig' et 'lecnombre'

Ces procédures sont apparues, d'abord, dans 'litab' (cf. IA§2.3); puis dans le programme 'qorlsup', avec la procédure 'tabsuper' (cf. ID§4.1.1).

Dans 'litaber', comme dans 'tabsuper', il faut que le tableau soit lu et traité ligne par ligne (et non lu globalement par 'litab' pour être traité ensuite). Car, avec un ensemble Eb de colonnes supplémentaires, il se peut que carv

```
procedure lecsig;begin sig[0]:=0;
   while (ord(carac) in [0..32]) and not eof(ftu) do read(ftu,carac);
   while not ((ord(carac) in [0..32]) or (sig[0]=4)) do begin
     sig[0]:=sig[0]+1;sig[sig[0]]:=ord(carac);
      if eof(ftu) then carac:=' ' else read(ftu,carac) end;
   while not ((ord(carac) in [0..32]) or eof(ftu)) do read(ftu,carac);
   if eof(ftu) then carac:=' ';end;
procedure lecnombre;begin ffl:=0;sgn:=1;
      while not((ord(carac) in [48..57]) or eof(ftu)) do begin
          read(ftu,carac);
          if not(ord(carac) in [48..57]) then sgn:=1;
          if (carac='-') then sgn:=-1;end;
      while (ord(carac) in [48..57]) do begin
          ffl:=(10*ffl)+(ord(carac)-48);
          if eof(ftu) then carac:=' ' else read(ftu,carac) end;
      ffl:=sgn*ffl; end;
```

dépasse le maximum: jmax=300, (cf. IA§1.1), assigné au nombre des colonnes d'un tableau écrit, tel quel, en mémoire centrale, sous des pointeurs: kji[0..jmax]. On verra (dans IV) que le programme 'zrang' peut créer, par transposition, des tableaux, en format de texte ou binaire, dont le nombre de colonnes excède jmax. Cest tableaux sont destinés à 'qorlsup', à 'VAX'...

4.1.3 Procédures générales de lecture accédant à des tableaux de tout format

Ainsi qu'on l'a dit au §2.3, le format du tableau croisant E et V, est noté sous la forme de deux entiers: {iz1, iz2}, dont le premier, redondant avec le second, donne seulement le type du fichier.

La procédure 'lirnombre' distingue quatre éventualités: on notera que, sur un texte, les nombres de format quelconque (iz2=3; suffixe 'yy') sont lus par la procédure 'read' de TML2 (cf. IA§2.3).

Les procédures 'lirsig' et 'lirtitr' n'en distingue que deux, selon le type du fichier. Il en est de même pour 'bailler' qui (en glissant sur un espace indéterminé de commentaires, s'il s'agit d'un texte) saisit le nombre total, carcot, des colonnes (nombre déjà connu d'ailleurs: cf. §4.1.1).

```
procedure lirnombre;begin
if (iz2=0) then begin lecnombre;ret:=ffl end;
if (iz2=3) then read(ftu,ret);
if (iz2=2) then begin read(fin,tampi[1]);read(fin,tampi[2]);
   ret:=ffli^ end;
if (iz2=1) then begin read(fin,tampi[1]);read(fin,tampi[2]);
   ret:=wwi^ end;end;
procedure lirsig;begin
if (iz1=2) then begin carac:=' ';lecsig end;
if (iz1=1) then begin read(fin,tampi[1],tampi[2],tampi[3]);
   sig:=sgli^;end;end;
procedure lirtitr;begin
if (iz1=2) then readln(ftu,titre);
if (iz1=1) then begin for c:=1 to titab do
   read(fin,tampi[c]);titre:=ttri^;end;end;
procedure bailler;begin {peut être mis dans lirtitr}
if (iz1=2) then begin lecnombre;carcot:=ffl mod 1000 end;
if (iz1=1) then read(fin,carcot);end;
```

4.2 Lecture et cumul par la procédure 'litaber'

4.2.1 Ouverture du fichier du tableau 'tab', croisant E et V

```
procedure lirtabler;
begin
if (iz2=0) then nomf:=nomf2;
if (iz2=3) then nomf:=concat(nomf2,'yy');
if (iz2=2) then nomf:=concat(nomf2,'.z');
if (iz2=1) then nomf:=concat(nomf2,'.w');
if (iz1=2) then reset(ftu,nomf);
if (iz1=1) then reset(fin,nomf);writeln(nomf);
```

On sait que le nom du fichier dépend du format, donné par iz2.

4.2.2 Lecture de l'en-tête du tableau 'tab'; sigles des colonnes

```
inilir;lirtitr;bailler;writeln(titre);
chx:='Z';
if (not(qvmax<carv) and not(chat='j')) then chx:='J';
if (not(qvmax<carv) and     (chat='j')) then chx:='I';
jco:=0;
for j:=1 to carcot do begin lirsig;
 pp:=prco^[j];jco:=jco+pp;
 write(psi^:5);if (j mod 14=0) then writeln;
 if ((chx='J') and (pp=1)) then begin
  sig1v^[jco]:=sig; end end;
```

Si le nombre des variables principales est inférieur à: qvmax=56 (cf. §1.3), les sigles des variables peuvent être gardés dans le tableau: sig1v^. Mais, en ce cas, il peut s'agir, pour le tableau 'tab' des sigles des colonnes ou de ceux des lignes: selon le cas, le caractère: chx, prend pour valeur: 'J', ou: 'I'. Sinon: chx:='Z'. Ultérieurement, chx sert à spécifier si la procédure 'vacorer' est appelée pour faire VACOR en terme de variables ou en terme de classes de variables (cf. §6). On notera que l'indice: j, suit la numération initiale des carcot colonnes de 'tab'; tandis que: jco, renvoie à la suite des colonnes principales.

4.2.3 Lecture du tableau 'tab', ligne par ligne

```
ili:=0;i:=0;
while not(i=carlit) do begin i:=i+1;
 ppp:=prli^[i];ili:=ili+ppp;
 lirsig;
 write(psi^:5);if (i mod 12=0) then writeln;
 if ((chx='I') and (ppp=1)) then sig1v^[ili]:=sig;
 jco:=0;
 for j:=1 to carcot do begin lirnombre;
   pp:=prco^[j];jco:=jco+pp;
   if ((ppp=1) and (pp=1)) then begin
     if ((rVq)='O') then begin eb:=qre^[ee^];vb:=qrv^[vv^];
       k1ve[vb]^[eb]:=k1ve[vb]^[eb]+ret;end;
     if ((rVv)='O') then begin eb:=qre^[ee^];
       k2ve[wv^]^[eb]:=k2ve[wv^]^[eb]+ret;end;end;end;
end;
```

Comme dans la lecture des sigles des colonnes, il faut distinguer entre les indices: {i, j}, qui suivent la disposition initiale de 'tab'; et les indices {ili, jco} qui renvoient à la suite des lignes et colonnes principales.

Mais, de plus, (cf. §4.1.1) il faut, pour chaque nombre: ret, lu dans tab,

non seulement déterminer s'il y a lieu de le prendre en compte (ce qui est le cas pour tout élément principal: ppp=pp=1), mais encore dans quelle case il doit être cumulé aux données déjà lues.

De façon précise, il faut distinguer deux cas. Si: rVq='O', le nombre: ret, doit être ajouté dans une case: k1ve[vb]^[eb], du tableau croisant les deux partitions de E et de V. Essentiellement, les indices: {vb, eb}, sont donnés par les tableaux: qre^ et qrv^, décrivant les partitions de E et de V (cf.§§3.2.1 et 3.2.2); mais, des indices: i et: j, du tableau 'tab', on passe aux numéros des éléments dans E et V par des pointeurs: ee, et: vv, dont la destination a été fixée comme l'explique le §4.1.1.

Si: rVv='O', pour le croisement de la partition de E avec l'ensemble Vp des variables principales, ret est ajouté dans une case: k2ve[wv^]^[eb], pour laquelle l'indice de la variable est atteint par un pointeur wv; lequel, selon les cas, accède à: jco, ou à: ili, cf. §4.1.1.

```
if (izl=2) then close(ftu);
if (izl=1) then close(fin);
end;{lirtabler}
{fin de seglir}
```

La procédure 'lirtabler' s'achève en fermant le fichier : ftu, ou: fin, où a été lu le tableau 'tab'.

4.3 Note: conflits entre versions successives d'analyses et de CAH

Sans prétendre protéger l'exécution du programme contre la lecture de fichiers n'ayant pas le format présumé, il faut considérer ici l'éventualité où CAH et partitions auraient été faites d'après une autre analyse factorielle que celle dont on consulte les fichiers 'ipr' et 'jpr'.

Divers nombres cardinaux d'ensembles, peuvent être calculés à plusieurs reprises, par des voies différentes; ce qui offrirait l'occasion de vérifications. La présente version de 'VAX' signale seulement ces reprises de calcul comme un commentaire, entre accolades. Mais nous croyons avoir protégé le programme contre tout ordre d'écriture en dehors des zones réservées de la mémoire centrale.

Pour les lignes des tableaux {k1ve, k2ve}, l'indice est un numéro de classe ou de nœud, qui ne peut dépasser qdmax, nombre déjà prévu lors de l'initialisation. Pour l'indice de colonne, il s'agit d'une variable principale, de Vp, ou d'une classe de variables. La caractérisation de Vp, passe toujours par le même tableau prv^, lu sur le fichier: 'jpr', ou: 'ipr', isu de l'analyse factorielle. Quant à la partition de Vp, elle est décrite par des fichiers 'cqki' et 'qcqabc' créés ensembles, par 'arbt', donc concordants; avec un numérotage des classes ne dépassant pas la dimension qvmqx des tableaux. Il se peut qu'en utilisant un fichier 'prv' postérieur à cette partition, on ait un numérotage: qrv^, incorrect. Mais, en dehors des numéros des véritables classes, il ne peut y avoir que des zéros, mis initialement dans qrv^, et les tableaux sont créés avec une colonne de rang zéro.

5 Écriture, en format binaire, de tableau de cumul: la procédure 'critaber'

```
{$S segcrire}  {$U ucrire5}
procedure critab(pi,pj,pk,pn,pt:ptr;ic,jc,tc:integer);external;
procedure critaber;begin
  if (rVq='O') then begin
    nomf:=concat(nombas,chat,'q',chv,'q.w');
    critab(@sigle^,@siglvq^,@t1k,@nomf,@titre,careq,carvq,0) end;
  if (rVv='O') then begin
    nomf:=concat(nombas,chat,'q',chv,'.w');
    critab(@sigle^,@siglv^,@t2k,@nomf,@titre,careq,carv,0) end;end;
```

La procédure 'critaber' écrit, éventuellement, sur disque, comme fichier binaire, un ou deux tableaux créés par cumul en mémoire centrale. À cette fin, 'critaber' appelle 'critab'; procédure qui, étant appelée par divers programmes (notamment: 'zrang', 'zBurt', cf. IIIA, IIIB), forme une unité séparée.

En bref, l'effet de 'critab' est inverse de celui de la procédure 'litab' (IA§2), celle-là écrivant des tableaux que celle-ci est capable de lire.

Par les pointeurs: {pi, pj, pk, pn, pt}, 'critab' a accès, respectivement, aux tableaux des sigles des lignes et colonnes; au tableau numérique lui-même, au

```
UNIT ucrire;
INTERFACE uses memtypes,quickdraw,osintf,toolintf,sane,uver;
procedure critab(pi,pj,pk,pn,pt:ptr;ic,jc,tc:integer);
implementation
procedure critab;
var fin:file of integer;
  sig:zigle;carac,rpf:char;nomf:string;ffl:longint;
  uc,c,j,i,i1,j1,iz1,iz2:integer;
  sgli:pzgl;ttri,sptit,sptr0:stringptr;ffli:plng;wwi:pww;ii,jj:pint;
  tampi:array[1..titab]of integer;
  sigli,siglj:pzgi;klji:kji;ptlk:ptk;
begin
sptr0:=stringptr(pn);sptit:=stringptr(pt);ptlk:=ptk(pk);
for j:=1 to jc do klji[j]:=pti(ptlk^[j]);
ffli:=plng(@tampi);wwi:=pww(@tampi);
sgli:=pzgl(@tampi);ttri:=stringptr(@tampi);iz2:=0;
if (copy(sptr0^,length(sptr0^)-1,2)='.w') then iz2:=1;
if (tc=1) then begin siglj:=pzgi(pi);sigli:=pzgi(pj);
                 ii:=pint(@j1);jj:=pint(@i1);uc:=jc;jc:=ic;ic:=uc end
          else begin sigli:=pzgi(pi);siglj:=pzgi(pj);
                 ii:=pint(@i1);jj:=pint(@j1) end;
rewrite(fin,sptr0^);ttri^:=sptit^;tampi[titab]:=ic;
for c:=1 to titab do write(fin,tampi[c]);write(fin,jc);
for j:=1 to jc do begin sgli^:=siglj^[j];writeln(sigler(sgli^):6);
     for c:=1 to 3 do write(fin,tampi[c]) end;
for i:=1 to ic do begin sgli^:=sigli^[i];writeln(sigler(sgli^):4);
     for c:=1 to 3 do write(fin,tampi[c]);il:=i;
     for j:=1 to jc do begin jl:=j;
          if (iz2=1) then wwi^:=klji[jj^]^[ii^]
                     else ffli^:=round(0+klji[jj^]^[ii^]);
          write(fin,tampi[1]);write(fin,tampi[2]) end end;
close(fin);
end;end.
```

nom du tableau à créer et au tritre de celui-ci. Les entiers: {ic, jc}, sont les cardinaux des ensembles des lignes et colonnes. Enfin, l'entier: tc, sert seulement à spécifier que le tableau soit créé tel quel: tc=0, ou transposé: tc=1. Mais on pourrait y mettre d'autres indications de format.

Dans la version publiée ici, 'critab' ne crée de fichier de tableau qu'en format binaire, comme tableau d'entiers longs ('longint') ou de rééls ('single'), le choix étant fait d'après le suffixe: '.w', ou: '.z', par lequel se termine le nom du fichier à créer.. On pourrait introduire les formats de texte: entiers longs et réels (suffixe: 'yy'). Présentement, nous avons seulement, dans 'zrang' (cf. IIIA§7), la possibilité de copier ou de transposer un tableau sous l'un ou l'autre de ces formats.

6 Création de listage VACOR d'aide à l'interprétation en terme de variables ou de cumuls de celles-ci: la procédure 'vacorer'

```
{$S segvacor}
{'vacorer'}
procedure vacorer;begin
if (chx=' ') then begin
   carvv:=carv ;for vx:=1 to carvv do kxve[vx]:=k2ve[vx];sig1vv:=sig1v  end;
if (chx='q') then begin
   carvv:=carvq;for vx:=1 to carvv do kxve[vx]:=k1ve[vx];sig1vv:=sig1vq end;
```

Comme on l'a dit (cf. *supra*, §§1.2 et 4.2.2), selon la valeur du caractère; chx, 'vacorer' est appelée pour créer un listage VACOR en terme de variables ou de classes de variables. Selon le cas, le tableau numérique: kxve, et le tableau de sigles: sig1vv, renvoient repectivement à: {k2ve, sig1v} ou à: {k1ve, sig1vq}.

6.1 Calculs de profils, marges, distances et inerties

```
for m:=1 to careq-1 do for vx:=1 to carvv do
   kxve[vx]^[m+careq]:=kxve[vx]^[qAm[m]]+kxve[vx]^[qBm[m]];
```

Le tableau de cumul, empli, au §4, avec careq lignes affectées aux classes de la partition retenue pour V, est maintenant complété par (careq-1) lignes affectées aux nœuds de la hiérarchie afférente à cette partition. On lit sur les tableaux: qAm, et: qBm, des aînés et benjamins, les lignes, déjà disponibles, qu'il faut cumuler pour créer une nouvelle ligne.

```
for e:=1 to (2*careq)-1 do begin ffs:=0;
   for vx:=1 to carvv do ffs:=ffs+kxve[vx]^[e];
   poig[e]:=ffs;if not(ffs=0) then ffs:=1/ffs;
   for vx:=1 to carvv do kxve[vx]^[e]:=kxve[vx]^[e]*ffs end;
ffs:=0;
for e:=1 to careq do ffs:=ffs+poig[e];
if not(ffs=0) then ffs:=1/ffs;
for e:=1 to (2*careq)-1 do poig[e]:=poig[e]*ffs;
```

Les lignes du tableau de cumul sont converties en profils. Les poids: poig[e], afférents aux careq classes, sont eux-mêmes converti en un profil en divisant par le total général des cases principales du tableau croisant E et V.

```
dg[(2*careq)-1]:=0;
for e:=1 to (2*careq)-2 do begin dg[e]:=-1;
    for vx:=1 to carvv do
        dg[e]:=dg[e]+(kxve[vx]^[e]*kxve[vx]^[e]/kxve[vx]^[(2*careq)-1]) end;
for e:=1 to (2*careq)-2 do if (dg[e]<0) then dg[e]:=0;
for m:=1 to careq-1 do begin res:=0;
    for vx:=1 to carvv do res:=res+
        (sqr(kxve[vx]^[qAm[m]]-kxve[vx]^[qBm[m]])/kxve[vx]^[(2*careq)-1]);
    dd[m]:=res end;
```

Dans: dg, sont calculés les distances, élevées au carré, des centres de classes de Ep, au centre de gravité qui n'est autre que le nœud supérieur numéroté: (2*careq)-1. Ainsi qu'on l'a annoncé à la fin du §0, il s'agit ici de la métrique du chi^2 afférente à la loi marginale, sur Vp, du tableau décrivant E; loi qui peut différer de celle du tableau de base sur le même ensemble (que celui-ci s'identifie à Ip ou à Jp). Le calcul de dg, avec instruction initiale: dg[e]:=-1, repose sur une formule classique, déjà rappelée en IB§2.1.1, pour la distance du chi^2 associée à la marge d'un tableau:

$$d^2(f_J^i, f_J) = \sum\{(f_j^i - f_j)^2\ (1/f_j) \mid j \in Jp\} = (-1) + \sum\{(f_j^i)^2\ (1/f_j) \mid j \in Jp\};$$

On calcule de même, pour chaque dipôle, le carré de la distance, dd, entre les centres de ses deux descendants immédiats.

Plus précisément, si VACOR est fait relativement aux cumuls de variables (cas: chx='q'), dd et dg sont calculées en projection sur l'espace engendré par les axes afférents à ces cumuls.

Il faut encore calculer, pour chaque coordonnée: v1, (variable ou classe de variable) une inertie totale de référence: dv[v1]. On doit ici distinguer plusieurs éventualités, notées dans le programme par le nombre: vrf.

Si l'ensemble E, dont on considère la partition, est Ip ou Jp (chat='i', ou: 'j'), et que l'on fait VACOR en terme de variables (chx=' '), le terme de référence naturel n'est autre que l'inertie sur cet axe du nuage initial Ep lui-même. Par exemple, si E=Jp, on a, pour expression de l'inertie sur l'axe i, une somme où l'on reconnaît, à un coefficient de masse près, le carré de la distance du chi^2, pour i:

$$\text{InrE}\ (Jp, i) = \sum\{f_j\ .\ (f_i^j - f_i)^2\ (1/f_i) \mid j \in Jp\} = f_i\ .\ d^2(f_J^i, f_J)\ ;$$

Le carré de distance: dv, se lit dans la colonne: 'dis2' du tableau: 'iFac' (cf. IB§§2.1.3 et 2.1.4), tableau généralement disponible sous l'hypothèse que: E=Jp (cas: verf=2). Et de même, *mutatis mutandis*, si E=Ip.

En tout autre cas (vrf<2), l'inertie de référence sur l'axe associé à une coordonnée: v1, variable ou cumul de variables, est calculée seulement pour le nuage des centres des classes; soit:

$$\text{InrE}\ (Eq, v1) = \sum\{f_q\ .\ (f_{v1}^q - f_{v1})^2\ (1/f_{v1}) \mid q \in Eq\}\ ;$$

et pour conserver l'analogie avec le cas où E est décrit par le tableau de base (vrf=2), on garde dans: dv[v1] le quotient: InrE (Eq, v1). $(1/f_{v1})$. Ce quotient

```
if not((chat='i') or (chat='j')) then vrf:=0 else vrf:=1;
if (((chat='i') or (chat='j')) and (chx=' ')) then vrf:=2;
if (vrf=2) then begin
  nomf:=concat(nombas,chv,'Fac.w');
  vrf:=verif(stringptr(@nomf));
  if (vrf=2) then begin
     reset(fin,nomf);
     for a:=1 to 61 do read(fin,c);
     for a:=1 to ddir do read(fin,ptrc1^[a]);
     for a:=62+ddir to titab do read(fin,c);
     read(fin,c);amax:=c-2;dir:=7+(2*amax);
     trace:=riri.Fac[amax+1];
     for a:=1 to 3*(2+amax) do read(fin,c);
     for v1:=1 to carvv do begin
       for a:=1 to dir do read(fin,ptrc1^[a]);
       dv[v1]:=riri.Fac[amax+2] end;
     close(fin) end end;
if (vrf<2) then for v1:=1 to carvv do begin
  dv1:=0;
  ffs:=kxve[v1]^[(2*careq)-1];fv1:=ffs;fv1:=sqr(fv1);
  if not(fv1=0) then begin
    for q:=1 to careq do begin
       res:=(kxve[v1]^[q]-ffs);res:=sqr(res)*poig[q];
       dv1:=dv1+res end;
    dv1:=dv1/fv1 end;
  dv[v1]:=dv1 end;
if (vrf=1) then begin
  nomf:=concat(nombas,chat,'Fac.w');
  vrf:=verif(stringptr(@nomf));
  if (vrf<2) then vrf:=0;
  if (vrf=2) then begin vrf:=1;
     reset(fin,nomf);
     for a:=1 to 61 do read(fin,c);
     for a:=1 to ddir do read(fin,ptrc1^[a]);
     for a:=62+ddir to titab do read(fin,c);
     read(fin,c);
     amax:=c-2;trace:=riri.Fac[amax+1];
     close(fin) end;end;
if (vrf=0) then begin trace:=0;
  for q:= 1 to careq do trace:=trace+(poig[q]*dg[q]) end;
res:=trace;writeln('trace = ',res:8);
```

est d'ailleurs bien un carré de distance, mais calculé dans l'espace des profils sur la partition, Eq, de E:

$$\text{InrE (Eq, v1)}.\ (1/f_{v1}) = \Sigma\{(f_q^{v1} - f_q)^2\ (1/f_q) \mid q \in \text{Eq}\}\ .$$

Dans la suite, les contributions, CTR, des classes aux axes (définis par variables, ou des classes de variables) sont calculées en fonction des dv[v1] par la même formule; quelle que soit l'origine des dv[v1].

Quant à l'inertie totale, qui sert de dénominateur pour calculer la colonne INR du listage VACOR, on peut, dans le cas: E=Ip, ou: E=Jp, la prendre égale à la trace de l'analyse de base, trace lue en tête d'un listage de facteurs. Sinon (cas: vrf=0), en guise de trace, on calcule, dans l'espace rapporté (selon que: chx=' ', ou: chx='q') aux variables ou à leurs cumuls suivant crvq classes, l'inertie totale du nuage des centres des carqe classes de la partition de E.

6.2 Écriture du listage VACOR

```
nomf:=concat(nombas,chat,'Vacor',chv);
if (chx='q') then nomf:=concat(nomf,'q');
rewrite(ftq,nomf);
writeln(ftq,titre);writeln(ftq,nomf);
if (vrf<2) then begin
  write(ftq,'NB les CTR ');
  if (vrf=0) then write(ftq,'et INR ');
  writeln(ftq,'se calculent relativement');
  writeln(ftq,'au nuage des centres de classes') end;
if (chx='q') then begin
  writeln(ftq,'les CO2 se calculent en projection sur l''espace');
  writeln(ftq,'engendré par les axes afférents aux cumuls par classe');
  writeln(ftq,'des variables primaires, ',chv);end;
```

6.2.1 Structure d'ensemble du listage et écriture de l'en-tête

Comme dans le listage FACOR (cf. IIC§2.3.1), on distingue, dans le listage VACOR, un en-tête et des bandes de résultats numériques. Chaque bande, peut être divisée, d'une part, horizontalement en 3 tranches; d'autre part, verticalement, en blocs de colonnes. Le listage donné comme exemple, reprend le cas choisi pour FACOR. Il n'a qu'une seule bande.

Dans chaque bande, la première et la deuxième tranche concernent respectivement, l'ensemble des nœuds et l'ensemble des classes de la partition retenue; avec les informations usuelles d'un listage d'analyse factorielle (complétées par la description de la hiérarchie). La troisième tranche, afférente aux dipoles, considère non les vecteurs joignant, dans l'espace ambiant, un point (classe ou nœud) à l'origine; mais les vecteurs joignant les deux descendants d'un même nœud.

Si le nombre: carvv, des coordonnées considérées sur le listage est ≤8, il n'y a qu'une seule bande; avec deux premiers blocs de colonnes afférents, repectivement, à la structure hiérarchique et à la qualité globale de la représentation; et, ensuite des blocs comprenant chacun trois colonnes. Si carvv>8, il y a plusieurs bandes: la première afférente aux coordonnées de rang 1 à 8 (variables, ou cumuls de variables par classe); la seconde, aux suivantes, jusqu'à concurrence de 8; etc. On reconnaît, sur le listage, les sigles usités en analyse des correspondances; avec, dans la troisième tranche, la lettre: D, qui rappelle qu'il s'agit de dipôles.

```
v0:=0;
while (v0<carvv) do begin
    v2:=v0+8;v0:=v0+1;if (carvv<v2) then v2:=carvv;
    for a:=1 to 38+(14*(v2-v0)) do write(ftq,'_');writeln(ftq);
    write(ftq,'CLAS AINE BNJM| PDS INR|');
    for v1:=v0 to v2 do write(ftq,sigler(sig1vv^[v1]):5,' CO2 CTR|');
    writeln(ftq);
```

La boucle: while (v0<carvv) do BEGIN, est parcourue autant de fois qu'il y a de bandes; avec pour régir l'écriture de la bande, le rang initial, v0 et le rang final, v2. Pour la 1-ère bande, v1=1 et v2=inf(carvv, 8). L'entrée dans une nouvelle itération dépend du calcul de nouvelles valeurs pour {v0, v2};

```
NB les CTR et INR se calculent relativement
au nuage des centres de classes
les CO2 se calculent en projection sur l'espace
engendré par les axes afférents aux cumuls par classe
des variables primaires,j
```

CLAS	AINE	BNJM	PDS	INR	j69	CO2	CTR	j71	CO2	CTR	j74	CO2	CTR	j73	CO2	CTR
repr des		4 noeuds sur l'ensemble jq														
25	24	23	1000	0	143	0	0	257	0	0	414	0	0	186	0	0
24	13	21	528	400	179	440	391	283	131	421	388	77	325	150	352	424
23	22	16	472	447	103	440	438	227	131	471	442	77	363	227	352	474
22	14	20	353	318	100	539	382	234	84	214	444	91	305	222	286	274
repr des		5 clases sur l'ensemble jq														
13			99	289	224	578	373	299	88	205	340	166	506	137	168	146
21	18	4	429	208	169	347	161	279	151	252	399	39	84	153	463	290
14	1	3	67	35	121	253	20	227	249	70	467	497	182	185	0	0
20	19	6	286	320	95	538	383	235	58	149	438	49	166	231	355	342
16	2	11	119	148	113	195	64	208	272	324	437	40	62	242	493	221
CDIP	AINE	BNJM	PDS	IND	j69	COD	CTD	j71	COD	CTD	j74	COD	CTD	j73	COD	CTD
repr des		4 dipôles sur l'ensemble jq														
25	24	23	1000	847	76	440	829	56	131	891	-54	77	688	-78	352	899
24	13	21	528	98	55	653	142	20	47	36	-59	258	266	-16	42	12
23	22	16	472	19	-13	195	8	26	436	67	7	19	4	-20	350	20
22	14	20	353	37	26	253	21	-9	16	5	29	111	43	-46	620	68

calcul effectué en tête de boucle, compte tenu de l'instruction: v0:=v2, qui répond, en fin de boucle, à l'instruction: v0:=0, précédant celle-ci.

6.2.2 Écriture de la première tranche d'une bande: les nœuds

On notera que, dans {CLAS, AINE, BNJM}, les numéros sont ceux de la CAH générale, lus dans le tableau cdq (cf. *supra*, §3.2.3). L'inertie: INR, est le produit du poids: poig[m+careq], par la distance au carré: dg[m+careq]; le tout étant divisé par une inertie totale, notée: trace, (cf. §6.1 *in fine*).

Dans CO2, on a, divisée par la distance au carré, dg, le terme en v1 de ce carré: $(f_{v1}{}^{q} - f_{v1})^2 \, (1/f_{v1})$.

Dans CTR, on doit avoir le rapport du terme: $(f_q \, . \, (f_{v1}{}^{q} - f_{v1})^2 \, (1/f_{v1}))$, à une inertie totale afférente à la coordonnée v1; inertie calculée de diverses manières, selon le cas, mais toujours exprimée comme: f_{v1} . dv[v1] .

```
writeln(ftq,'repr des',careq-1:4,' noeuds sur l''ensemble ',chv,chx);
for m:=careq-1 downto 1 do begin
  CLAS:=cdq[m+careq];AINE:=cdq[qAm[m]];BNJM:=cdq[qBm[m]];
  PDS:=round(1000*poig[m+careq]);
  INR:=round(1000*poig[m+careq]*dg[m+careq]/trace);
  write(ftq,CLAS:4,AINE:5,BNJM:5,'|',PDS:4,INR:4,'|');
  for v1:=v0 to v2 do begin res:=kxve[v1]^[m+careq];
     FCT:=round(1000*res);
     CO2:=0;CTR:=0;fv1:=kxve[v1]^[(2*careq)-1];
     res:=sqr(res-fv1);dv1:=fv1*dg[m+careq];
     if not (dv1=0) then CO2:=round(1000*res/dv1);
     dv1:=sqr(fv1)*dv[v1];
     if not (dv1=0) then CTR:=round(1000*poig[m+careq]*res/dv1);
     write(ftq,FCT:5,CO2:4,CTR:4,'|') end;
 writeln(ftq) end;
```

6.2.3 Écriture de la deuxième tranche d'une bande: les classes

```
writeln(ftq,'repr des',careq:4,' classes sur l''ensemble ',chv,chx);
for q:=1 to careq do begin c:=cdq[q];n:=c-care;write(ftq,c:4);
  if (0<n) then write(ftq,cAq[q]:5,cBq[q]:5) else write(ftq,'          ');
  INR:=round(1000*poig[q]*dg[q]/trace);PDS:=round(1000*poig[q]);
  write(ftq,'|',PDS:4,INR:4,'|');
  for v1:=v0 to v2 do begin
     res:=kxve[v1]^[q];FCT:=round(1000*res);
     CO2:=0;CTR:=0;
     fv1:=kxve[v1]^[(2*careq)-1];res:=sqr(res-fv1);
     dv1:=fv1*dg[q];
     if not (dv1=0) then CO2:=round(1000*res/dv1);
     dv1:=sqr(fv1)*dv[v1];
     if not (dv1=0) then CTR:=round(1000*poig[q]*res/dv1);
     write(ftq,FCT:5,CO2:4,CTR:4,'|') end;
  writeln(ftq) end;
```

Le calcul est tout analogue à celui fait pour les nœuds; à ceci près qu'une classe réduite à un élément n'a ni aîné ni benjamin (condition: c<care; classe 13 sur le listage donné comme exemple).

On rappelle que, sur l'ensemble des classes, le poids: PDS, a pour total 1000 millièmes. De plus, le PDS d'un nœud est la somme de ceux de son aîné et de son benjamin. Mais, pour INR ou CTR, le total des termes afférents à l'aîné et au benjamin dépasse, en général, le terme propre au nœud: la différnce, se retrouve dans la tranche des dipôles (cf. *infra*).

6.2.4 Écriture de la troisième tranche d'une bande: les dipôles

```
write(ftq,'CDIP AINE BNJM| PDS IND|');
for v1:=v0 to v2 do write(ftq,sigler(siglvv^[v1]):5,' COD CTD|');
writeln(ftq);
writeln(ftq,'repr des',careq-1:4,' dipôles sur l''ensemble ',chv,chx);
for m:=careq-1 downto 1 do begin
  CLAS:=cdq[m+careq];AINE:=cdq[qAm[m]];BNJM:=cdq[qBm[m]];
  PDS:=round(1000*poig[m+careq]);
  ppg:=poig[qAm[m]]+poig[qBm[m]];
  if not (ppg=0) then ppg:=(poig[qAm[m]]*poig[qBm[m]])/ppg;
  INR:=round(1000*ppg*dd[m]/trace);res:=0;
  write(ftq,CLAS:4,AINE:5,BNJM:5,'|',PDS:4,INR:4,'|');
  for v1:=v0 to v2 do begin
     res:=kxve[v1]^[qAm[m]]-kxve[v1]^[qBm[m]];
     FCT:=round(1000*res);
     CO2:=0;CTR:=0;
      fv1:=kxve[v1]^[(2*careq)-1];res:=sqr(res);dv1:=fv1*dd[m];
     if not (dv1=0) then CO2:=round(1000*res/dv1);
     dv1:=sqr(fv1)*dv[v1];
     if not (dv1=0) then CTR:=round(1000*ppg*res/dv1);
     write(ftq,FCT:5,CO2:4,CTR:4,'|') end;
  writeln(ftq) end;
 for a:=1 to 38+(14*(v2-v0)) do write(ftq,'_');writeln(ftq);
 v0:=v2 end;
close(ftq);writeln('fin d''écriture de ',nomf);
end;
```

Ainsi qu'on l'a annoncé au §6.2.1, les sigles usités pour la tranche des dipôles diffèrent quelque peu de ceux communs aux nœuds et aux classes.

Cette distinction est d'autant plus utile que les dipôles ont mêmes numéros que les nœuds; ceux-là ne se distinguant de ceux-ci que quant aux calculs effectués. Dans le calcul d'IND et de CTD, on a, pour bras de levier, la différence entre les deux descendants immédiats; et le facteur de masse, ppg, est donné par la formule:

(poig[qAm[m]]*poig[qBm[m]]) / (poig[qAm[m]]+poig[qBm[m]]) ;

quotient du produit des masses des deux descendants immédiats par leur somme.

Prenons des exemples dans le listage. L'inertie, INR, afférente au nœud 24, est: 400. Pour l'aîné et le benjamin, {13, 21}, les INR sont: {289, 208}. La différence: ((289+208)-400)=97, est (à une erreur d'arrondi près) la valeur IND du dipôle 24. De même, dans le bloc de colonnes afférent à la coordonnée: j71, on a des CTR pour {24, 13, 21} qui sont {421, 205, 252}; le CTD, pour 24, est: 36=((205+252)-421). En effet, appelons, en bref: point q, le centre gravité d'une classe q, compté avec sa masse. L'inertie (INR ou CTR) du système des deux points {13, 21} relativement à l'origine peut se décomposer en: inertie (IND ou CTD) de ce système relativement à son centre 24, et inertie (INR ou CTR) du point 24 relativement à l'origine.

Une fois sortie de la boucle d'écriture des bandes, la procédure 'vacorer' s'achève en fermant le fichier de texte: ftq, où a été écrit le listage VACOR.

7 Libération des zones de la mémoire centrale assignées à des pointeurs: la procédure 'memvider'

```
{$S segvider}
{'memvider'}
procedure memvider;begin
 dispose(pre);dispose(prv);dispose(qre);dispose(qrv);dispose(prt);
 dispose(sigle);dispose(siglv);dispose(siglvq);
 for j:=0 to qvmax do begin dispose(k1ve[j]);dispose(k2ve[j]) end;
end;
```

Ainsi qu'on l'a annoncé au §1.2, la procédure 'memvider' libère les zones de la mémoire centrale affectées par 'taber'. Et, après un VACOR mené à bien, l'occasion est offerte à l'utilisateur d'en demander un autre: ce qui conduit à rentrer dans la procédure initiale 'choisir'.

8 Appendice: étiquetage des arbres par VACOR

Le listage VACOR donne, pour chacune des classes de la partition retenue, les composantes de son profil: soit sur l'ensemble même des variables; soit sur une partitition de celui-ci; avec l'importance de chacune de ces composantes pour caractériser la classe, par excès ou par défaut, relativement au profil moyen.

Dans la mesure du possible, on s'applique à reporter, sur l'arbre de la CAH, les informations les plus saillantes notées sur le listage VACOR: c'est ce qu'on appelle: étiqueter l'arbre.

Parfois, l'étiquetage est si précis qu'il offre, sous la forme ordonnée de l'arbre, l'essentiel des données numériques que renferme, sous forme compacte, le tableau de base lui-même. Afin de s'assurer que les pourcentages, ou composantes du profil, afférents à une classe valent, quasi exactement, pour tout individu de la classe, il convient, en toute rigueur, de déterminer, pour la classe, non seulement son profil, mais l'intervalle dans lequel varie, sur l'ensemble des éléments de la classe, chacune des composantes du profil: tel est l'objet du programme INFSUP, dû à Y.L. CHEUNG (cf. [INTRPRÉT. CAH]).

On trouve dans le volume Prat5Éco, de nombreux exemples d'analyses de flux du commerce international où l'on s'est ingénié à reporter, sur un arbre, tout le contenu d'un dossier.

Références bibliographiques

J.-P. & F. BENZÉCRI, Y.L. CHEUNG, S. MAÏZA: "Aides à l'interprétation et étiquetage des arbres en classification ascendante hiérarchique: listages FACOR, VACOR et INFSUP"; [INTRPRÉT. CAH]; in *CAD*, Vol.X, n°3; pp. 311-338; (1985);

M. JAMBU : "Programme de calcul des contributions mutuelles entre classes d'une hiérarchie et facteurs d'une correspondance"; in *CAD*, Vol.I, n°1; pp. 77-92; (1976);

M.-O. LEBAUX : premières versions des programmes d'aide à l'interprétation; et notices de ces programmes;

Pratique de l'Analyse des Données en Économie, Prat5Éco, J.–P. & F. BENZÉCRI et coll.; Dunod, Paris; (1986).