Les Cahiers de l'Analyse des Données
Vol. XXII - 1997 - n°1 - pp.71-82.

SOURCES DE PROGRAMMES D'ANALYSE DE DONNÉES EN LANGAGE PASCAL: (I) : ANALYSE DES CORRESPONDANCES (ID) : ADJONCTION DE TABLEAUX EN SUPPLÉMENT

[SOURCES PASCAL (ID)]

J.-P. & F. BENZÉCRI

0 Introduction : structure et déroulement du programme 'qorlsup'

On sait qu'à l'analyse d'un tableau de base, I × J, peut être adjoint un tableau supplémentaire externe: tableau, Ia × J, de lignes supplémentaire; ou tableau, Jb × I, de colonnes supplémentaires.

Dans la suite, comme le suggère la notation adoptée ci-dessus, on supposera toujours qu'un tableau de colonnes supplémentaires est présenté sous forme transposée: ainsi, les éléments supplémentaires, jb, décrit chacun par son sigle et ses composantes sur I, peuvent être lus et traités consécutivement, les uns indépendamment des autres. Si le tableau est donné, d'abord, sous la forme I × Jb, on doit le transposer par un programme qui, tel 'zrang', offre cette option à l'utilisateur (cf. IID§5, IIIA§1.1.2).

De plus, le programme 'qorlsup' n'accepte un tableau externe que si le nom de celui-ci est formé en ajoutant, au nom du tableau de base, un sigle d'au plus 3 lettres; dont la dernière doit être 'a' ou 'b', selon qu'il s'agit de lignes ou de colonnes supplémentaires. Il va sans dire qu'au sigle peut être ajouté l'un des suffixes {'yy', '.z', '.w'} spécifiant le format d'écriture des nombres (cf. procédure 'litab': IA§2.1).

On a déjà traité, dans 'qori', des ensembles, Is ou Js, de lignes ou de colonnes supplémentaires: (cf. IB§§2.1.1, 2.2.3, 2.2.4). Dans 'qorlsup', comme dans 'qori', on applique la formule de transition. Partant, respectivement, des valeurs d'un facteur pour l'ensemble, Ip ou Jp, des éléments principaux du tableau de base, on calcule ce même facteur pour une colonne ou une ligne supplémentaire, jb ou ia, donnée par l'ensemble de ses composantes sur I ou J; ou, comme nous dirons en bref, sur l'autre ensemble.

De façon précise, les racines des valeurs propres interviennent comme coefficients avec le profil de l'élément à adjoindre, suivant les formules:

$$F_\alpha(ia) = (1/\sqrt{(\lambda_\alpha)}).\ \Sigma\{f_j^{ia}\ .\ G_\alpha(j)) \mid j \in Jp\} = \Sigma\{f_j^{ia}\ .\ \varphi_\alpha{}^j) \mid j \in Jp\} \quad ;$$

$$G_\alpha(jb) = (1/\sqrt{(\lambda_\alpha)}).\ \Sigma\{f_i^{jb}\ .\ F_\alpha(i)) \mid i \in Ip\} = \Sigma\{f_i^{jb}\ .\ \varphi_\alpha{}^i) \mid i \in Ip\} \quad .$$

Mais il faut prendre garde que ce profil est calculé sur les éléments principaux de l'autre ensemble. Lors de l'exécution du programme 'qori', la distinction est claire: car, par la procédure 'tabuler', cf. IB§3.2.3, le tableau de base, $I \times J$, est disposé suivant l'ordre $(Ip \cup Is) \times (Jp \cup Js)$, les éléments principaux et supplémentaires formant deux blocs consécutifs.

Pour 'qorlsup', la place des éléments supplémentaires est connue d'après les fichiers 'ipr', 'jpr', créés lors de l'analyse du tableau de base, (cf. IB§3.2.2); et, en bref, (cf. *infra*, §3.1, procédure 'princer'), des nombres lus pour un élément du tableau externe, ne sont gardés que ceux dont le rang est, pour le tableau de base, celui d'un élément principal de l'autre ensemble.

Finalement, pour mettre en supplément tout tableau externe, $Ia \times J$ ou $Jb \times I$, 'qorlsup' doit disposer des quatre fichiers 'iFac.w', 'jFac.w', 'ipr' et 'jpr', créés, par 'qori', lors de l'analyse du tableau de base (cf. IC§3).

```
begin benzecri;rpc:='O';
while not(rpc='N') do begin
 principer;
 if (rpc='O') then sigsuper;
 if (rpc='O') then begin entrer;unloadseg(@entrer);
   for ns:=1 to nsup do begin
       six:=sigler(sixn[ns]);spus:=spn[ns];tabsuper end;
   fermer end;
 if (rpc='O') then begin
   write('faut-il poursuivre O ou N ');readln(rpc);end;end;
readln(rpz);end.
```

Avant d'expliquer le programme 'qorlsup' en suivant le listage, lequel donne, après des déclarations, une longue suite hiérarchique de procédures, rangées en segments, il vaut la peine (comme on l'a fait pour 'qori': cf. IB§0) de considérer le programme principal, qui est très bref.

D'abord, par la procédure 'principer', l'utilisateur choisit le tableau de base auquel seront adjoints des tableaux supplémentaires externes; et le programme vérifie que les fichiers 'Fac' et 'pr' existent sur le disque.

À cette condition, le choix étant fixé par la réponse de confirmation rpc='O', l'utilisateur donne, dans 'sigsuper', les sigles des tableaux externes qu'il désire traiter.

Par la procédure 'entrer', le programme, met alors en mémoire centrale les résultats de l'analyse du tableau de base.

Dans la boucle: for ns:=1 to nsup do begin..., sont traités, successivement, par 'tabsuper', les tableaux externes choisis dans 'sigsuper'. Pour chacun de ceux-ci est créé, d'une part, un fichier binaire de facteurs,

'iaFac.w' ou 'jbFac.w'; et, d'autre part, un listage, 'iacorsutx' ou 'jbcorsutx'. On notera que l'entier spus=spn[ns] caractérise le format, de texte ou binaire, du tableau externe considéré: cf. *infra*, §2.

La procédure 'fermer' libère l'espace réservé en mémoire centrale pour garder les résultats de l'analyse du tableau de base. Et si l'utilisateur désire poursuivre, on rentre dans la boucle: while not(rpc='N') do..., pour adjoindre des tableaux externes à un nouveau tableau de base.

```
program qorlsup;
uses memtypes,quickdraw,osintf,toolintf,sane,uver;
var
  i,j,dir,a,amax,nsup,ns,spus,u,lu,c,ih,jh,eps,izl,iz2,ccc,karh,aaa,bbb,
  garti,cari,caris,cartj,carj,carjs,carf,care,carte,carel,erl,carh,sgn:integer;
  spn:array[1..20]of integer;tampi:array[1..titab]of integer;
  ffli:plng;ttri:stringptr;sgli:pzgl;plww,wwi:pww;
  ff,ffl,PDS,INR,QLT,CO2,CTR:longint;
  fin,firi:file of integer;ffs:single;
  ft,ftu:text;trace,res,ret,mtot,fre,d2,dr,ke:extended;
  usla,lama,Fa:array[1..aamax]of extended;
  Phai,Phaj,Phae:array[1..aamax]of pti;nouv:ptr;
  fi,fj,fe,fhe:ptdri;pri,prj,pre:ptni;sisi,sisj,sise:pzgi;
  rpc,rpf,rpz,carac,che,chh,chv,chc:char;sql:zigle;
  nomf,nomba,titre,six,sig,sur:string;
  sfx:string[9];sixn:array[1..20]of zigle;
  riri:ricla;ptrcl:ptrc;
```

1 Choix du tableau de base: la procédure 'principer'

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

```
{$S initium}
procedure vericher;begin
  verichaine(stringptr(@nomba),stringptr(@sfx),rpc);end;
procedure principer;begin
 rpc:='N';erl:=0;
 ttri:=stringptr(@tampi);sgli:=pzgl(@tampi);
 ffli:=plng(@tampi);wwi:=pww(@tampi);
 while not ((rpc='O') or (erl=6)) do begin rpc:='*';
   write('le nom de base du fichier principal est ');readln(nomba);
   sfx:='iFac.w';vericher;
   sfx:='jFac.w';vericher;
   sfx:='ipr';vericher;
   sfx:='jpr';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.

Dans la version publiée de 'qorlsup', est requise l'existence de quatre fichiers: en toute rigueur, il suffirait des informations relatives à un seul des deux ensembles, I ou J; mais alors, seuls pourraient être adjoints, repectivement, des tableaux de la forme: I × Jb, ou Ia × J.

'iaFac.w' ou 'jbFac.w'; et, d'autre part, un listage, 'iacorsutx' ou 'jbcorsutx'. On notera que l'entier spus=spn[ns] caractérise le format, de texte ou binaire, du tableau externe considéré: cf. *infra*, §2.

La procédure 'fermer' libère l'espace réservé en mémoire centrale pour garder les résultats de l'analyse du tableau de base. Et si l'utilisateur désire poursuivre, on rentre dans la boucle: while not(rpc='N') do..., pour adjoindre des tableaux externes à un nouveau tableau de base.

```
program qorlsup;
uses memtypes,quickdraw,osintf,toolintf,sane,uver;
var
  i,j,dir,a,amax,nsup,ns,spus,u,lu,c,ih,jh,eps,iz1,iz2,ccc,karh,aaa,bbb,
  garti,cari,caris,cartj,carj,carjs,carf,care,carte,carel,er1,carh,sgn:integer;
  spn:array[1..20]of integer;tampi:array[1..titab]of integer;
  ffli:plng;ttri:stringptr;sgli:pzgl;plww,wwi:pww;
  ff,ff1,PDS,INR,QLT,CO2,CTR:longint;
  fin,firi:file of integer;ffs:single;
  ft,ftu:text;trace,res,ret,mtot,fre,d2,dr,ke:extended;
  usla,lama,Fa:array[1..aamax]of extended;
  Phai,Phaj,Phae:array[1..aamax]of pti;nouv:ptr;
  fi,fj,fe,fhe:ptdri;pri,prj,pre:ptni;sisi,sisj,sise:pzgi;
  rpc,rpf,rpz,carac,che,chh,chv,chc:char;sql:zigle;
  nomf,nomba,titre,six,sig,sur:string;
  sfx:string[9];sixn:array[1..20]of zigle;
  riri:ricla;ptrcl:ptrc;
```

1 Choix du tableau de base: la procédure 'principer'

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

```
{$S initium}
procedure vericher;begin
  verichaine(stringptr(@nomba),stringptr(@sfx),rpc);end;
procedure principer;begin
 rpc:='N';er1:=0;
 ttri:=stringptr(@tampi);sgli:=pzgl(@tampi);
 ffli:=plng(@tampi);wwi:=pww(@tampi);
 while not ((rpc='O') or (er1=6)) do begin rpc:='*';
   write('le nom de base du fichier principal est ');readln(nomba);
   sfx:='iFac.w';vericher;
   sfx:='jFac.w';vericher;
   sfx:='ipr';vericher;
   sfx:='jpr';vericher;
   if (rpc='N') then er1:=er1+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.

Dans la version publiée de 'qorlsup', est requise l'existence de quatre fichiers: en toute rigueur, il suffirait des informations relatives à un seul des deux ensembles, I ou J; mais alors, seuls pourraient être adjoints, repectivement, des tableaux de la forme: I × Jb, ou Ia × J.

2 Choix des tableaux externes à adjoindre en supplément: la procédure 'sigsuper'

```
procedure sigsuper;begin rpc:='O';
 write('nombre(1 a 20) de tableaux à mettre en supplémentaire = ');
 readln(nsup);if (nsup<1) then nsup:=1;if (20<nsup) then nsup:=20;
 writeln('NB les sigles ont 1 à 3 lettres et se terminent par a ou b');
 for ns:=1 to nsup do if (rpc='O') then begin rpc:='N';er1:=0;
   while not((rpc='O') or (er1 =6)) do begin
     write('le suffixe du tableau',ns:3,' est ');readln(six);
     if (3<length(six))or not(six[length(six)]in['a','b']) then begin
       rpc:='N';writeln('ERREUR sur la forme du sigle') end
      else begin
       nomf:=concat(nomba,six);sixn[ns]:=zigler(six);
       iz1:=verif(stringptr(@nomf));iz2:=0;
       if (iz1=1) then begin nomf:=concat(nomba,six,'yy');
           iz2:=verif(stringptr(@nomf));
           if (iz2=2) then begin iz1:=2;iz2:=3 end end;
       if (iz1=1) then begin nomf:=concat(nomba,six,'.z');
           iz2:=verif(stringptr(@nomf)) end;
       if (iz2=1) then nomf:=concat(nomba,six,'.w');
       rpc:=dialof(stringptr(@nomf)) end;
     if (rpc='N') then er1:=er1+1 else begin
       writeln(nomf);spn[ns]:=(4*iz1)+iz2;
       write('ce choix est-il confirmé oui(O) ou non(N) ');
       readln(rpc) end end end;
 aaa:=0;bbb:=0;
 if (rpc='O') then for ns:=1 to nsup do begin
    six:=sigler(sixn[ns]);
    if (six[length(six)]='a') then aaa:=1 else bbb:=1 end;end;
```

Le nombre, 'nsup' des tableaux supplémentaires est, arbitrairement, limité à 20. Ce nombre étant fixé, on entre dans une boucle: for ns:=1 to nsup do if (rpc='O') then... La procédure 'sigsuper' sort de la boucle si, en six essais, l'utilisateur n'a pu proposer un sigle auquel corresponde un tableau de l'un ou l'autre des formats, de texte ou binaire, acceptés par 'ulire' (procédure qui, toutefois, ne sert pas ici, car le tableau externe est lu, non en un seul bloc, mais ligne par ligne; cf. *infra* §4.1.4).

De façon précise, est d'abord vérifié le format du sigle, déjà expliqué au §0. Soit: Disq:tab, le nom (privé de son suffixe éventuel) du tableau de base; et soit 'sga' le sigle. On cherche un tableau externe, 'Disq:tabsga' (sans suffixe: texte, avec nombres entiers écrits tels quels). Si ce tableau existe, on a: iz1=2, iz2=0; et il n'y a pas lieu de s'enquérir d'un autre format. Sinon, iz1=1 (la valeur zéro étant exclue, car elle correspond à l'absence du disque 'Disq', dont la présence a été vérifiée par 'principer'). L'on cherche alors un tableau externe avec le suffixe 'yy' (texte, avec nombres écrits en format quelconque). Si celui-ci existe, on s'y arrête, avec: iz1=2, iz2=3. En cas d'échec, on passe au suffixe '.z' (nombres entiers en format binaire), et si le tableau se trouve, on a: iz1=1, iz2=2. Sinon, il reste pour dernière éventualité le suffixe '.w' (réels en format binaire) avec: iz1=1, iz2=1. Finalement, c'est la réponse 'rpc' (à 'verif': cf. IA§1.3.2) qui exprime s'il y a eu succès: 'O', ou échec: 'N'. Et selon le cas, er1 (nombre des échecs) est accru de 1; ou le sigle 'sga' est soumis à la confirmation de l'utilisateur.

3 Lecture des informations afférentes à l'analyse du tableau de base: la procédure 'entrer'

3.1 Les procédures appelées par la procédure 'entrer'

```
procedure licarte;begin
 reset(fin,concat(nomba,che,'pr'));carte:=0;
 while not eof(fin) do begin carte:=carte+1;read(fin,i) end;
 close(fin);end;
procedure princer;begin
 reset(fin,concat(nomba,che,'pr'));i:=0;
 while not eof(fin) do begin i:=i+1;read(fin,pre^[i]) end;
 carte:=i;close(fin);writeln('princer',che:3);end;
procedure licarel;begin
 reset(fin,concat(nomba,che,'Fac.w'));
 for u:=1 to titab-1 do read(fin,c);read(fin,carel);close(fin);end;
procedure liPher;begin
 reset(fin,concat(nomba,che,'Fac.w'));
 with riri do begin
  for u:=1 to 61 do read(fin,c);
  for u:=1 to ddir do read(fin,ptrc1^[u]);
  for u:=62+ddir to titab-1 do read(fin,c);
  read(fin,carel);read(fin,c);amax:=c-2;dir:=7+(2*amax);
  for u:=1 to 3*(2+amax) do read(fin,c);
  for a:=1 to amax do begin lama[a]:=Fac[a];
    if (lama[a]=0) then usla[a]:=0 else usla[a]:=1/sqrt(lama[a]) end;
  trace:=Fac[amax+1];mtot:=Fac[amax+2];carf:=amax;
  for i:=1 to carel do begin
    for a:=1 to dir do read(fin,ptrc1^[a]);sise^[i]:=sis;
    fe^[i]:=Fac[amax+1];
    for a:=1 to amax do Phae[a]^[i]:=usla[a]*Fac[a] end;end;close(fin);
 writeln('liPher',che:4,' carel = ',carel:4,' mtot = ',mtot:6);end;
```

Il faut prendre garde que les procédures expliquées ici servent aussi bien pour I que pour J, selon que le caractère: che, est 'i' ou 'j'.

D'après le fichier 'Disq:tabepr', 'licarte' détermine le cardinal, carte, de l'ensemble E (carti, pour I; ou cartj, pour J), sans distinction entre éléments principaux, Ep, et supplémentaires, Es.

La procédure 'princer' lit les mêmes fichiers que 'licarte' ('ipr' ou 'jpr'). Mais 'princer', note, en mémoire centrale, une suite, en {0, 1}, caractérisant les éléments principaux: pre^[i]=1, et supplémentaires: pre^[i]=0. On ne peut appeler 'princer' directement, parce que la mémoire n'est réservée pour pre^ qu'après que 'licarte' en ait fixé la taille, 'carte'.

Par 'licarel' le nombre des éléments principaux est lu sur un fichier de facteurs: e.g. cari est lu sur 'iFac.w'. Sur ce même fichier, la procédure 'liPher' lit d'abord les valeurs propres; puis les valeurs des facteurs pour les éléments principaux du tableau de base. Comme dans IC§§1.2.3, 1.2.4.5, 2.7.1 (pour le tracé des graphiques plans), ces informations ont pu être comprises sous le format 'ricla'. Il faut prendre garde que, compte tenu de la formule de transition rappelée au §0, les valeurs propres λ_α servent à calculer, en fonction des F_α, les φ_α, qui sont mis en mémoire centrale dans l'espace vers lequel pointent les Phae[a].

3.2 Explication du corps de la procédure 'entrer'

```
procedure entrer;begin ptrc1:=ptrc(@riri);cari:=0;carj:=0;
if (bbb=1) then begin
 che:='i';licarel;cari:=carel;licarte;carti:=carte;
 nouv:=newptr(2*carte);pri:=ptni(nouv);pre:=pri;
 if (ibmax<carel) then new(sisi) else begin
    nouv:=newptr(6*carel);sisi:=pzgi(nouv) end;
 sise:=sisi;
 if (iamax<carel) then new(fi) else begin
    nouv:=newptr(xtn*carel);fi:=ptdri(nouv) end;
 fe:=fi;
 for a:=1 to aamax do begin
    nouv:=newptr(4*carel);Phai[a]:=pti(nouv);Phae[a]:=Phai[a] end;
 princer;liPher end;
if (aaa=1) then begin
 che:='j';licarel;carj:=carel;licarte;cartj:=carte;
 nouv:=newptr(2*carte);prj:=ptni(nouv);pre:=prj;
 if (ibmax<carel) then new(sisj) else begin
    nouv:=newptr(6*carel);sisj:=pzgi(nouv) end;
 sise:=sisj;
 if (iamax<carel) then new(fj) else begin
    nouv:=newptr(xtn*carel);fj:=ptdri(nouv) end;
 fe:=fj;
 for a:=1 to aamax do begin
    nouv:=newptr(4*carel);Phaj[a]:=pti(nouv);Phae[a]:=Phaj[a] end;
 princer;liPher end;
if (carf<9) then care:=carf else care:=8;
ccc:=cari;if (cari<carj) then ccc:=carj;ccc:=ccc+1;
if (iamax<ccc) then new(fhe) else begin
    nouv:=newptr(xtn*ccc);fhe:=ptdri(nouv) end;end;
```

À la fin de la procédure 'sigsuper', cf. §2, on a calculé les nombres 'aaa' et 'bbb', valant respectivement 1 ou 0, selon qu'existe ou non un tableau externe (ou plusieurs) dont le sigle se termine par 'a' (lignes supplémentaires); ou 'b' (colonnes supplémentaires). Il n'y a lieu de mettre en mémoire centrale les facteurs sur Ip (issus de l'analyse du tableau de base) que si: bbb=1; et, de même, les facteurs sur Jp ne servent que si: aaa=1.

Prenons pour exemple l'instruction composée: if (bbb=1) then begin... Les procédures considérées au §3.1 y sont appelées avec: che='i'. Corrélativement, des zones sont réservées, en mémoire centrale; vers lesquelles pointent: pri (caractérisation des éléments principaux et supplémentaires de I); sisi (sigles des éléments principaux); fi (fréquences pour Ip); Phai[a] (facteurs sur Ip). Ainsi, quand, par exemple, est appelée la procédure 'liPher', les pointeurs Phae[a] (qui indiquent les zones où inscrire les facteurs pris dans 'iFac.w') sont, en fait, les Phai[a], déjà initialisés.

Le nombre: care, des facteurs à écrire sur les listages afférents aux tableaux externes, ne peut dépasser le nombre, carf, des facteurs conservés dans l'analyse du tableau de base; et il est limité à 8, par la largeur des lignes; d'où la formule: care:= sup(carf, 8).

Enfin, avec le pointeur: fhe, on réserve une zone en mémoire centrale pour appliquer la formule de transition sous le format 'extended'.

4 Traitement des tableaux externes: la procédure 'tabsuper'

Ainsi qu'on l'a annoncé au §2, les tableaux externes sont traités ligne par ligne, au fur et à mesure de la lecture des données. Mais, à cette différence près, on retrouve, dans la lecture, les instructions de la procédure 'litab' (cf. IA§2); et le calcul, puis l'écriture des facteurs, se font comme dans 'qori' (cf. IB§2). Nos commentaires en seront allégés d'autant.

4.1 Les procédures appelées par la procédure 'tabsuper'

NB L'ordre de succession des procédures sur le listage n'importe qu'en ce que toute procédure doit être précédée de celles qu'elle appelle.

4.1.1 Lecture d'une donnée sur un tableau externe en format de texte

```
{$S actio}
procedure lecsig;   begin    sig := '';
   while (ord(carac) in [0..32]) and not eof(ftu) do read(ftu,carac);
   while not ((ord(carac) in [0..32]) or (length(sig)=4)) do begin
      sig := concat(sig,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(ft) 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;
```

Les procédures 'lecsig' et 'lecnombre' sont celles de 'ulire' (cf. IA§2.3); à ceci près qu'ici 'lecsig' utilise le format 'sigle' (chaine de caractères); et non 'zigle' (packed array of byte: cf. IA§1.3.4).

4.1.2 Écriture, sur le listage, de la ligne afférente à un élément

```
procedure lignage;begin
     PDS:=round(1000*fre);
     INR:=round(1000*d2*fre/trace);
     if (INR>999) then INR:=999;if (INR<-999) then INR:=-999;
     if (d2=0) then  QLT:=0 else begin dr:=0;
         for a:=1 to care do dr:=dr+sqr(Fa[a]); QLT:=round(1000*dr/d2) end;
     write(ft,'|',sigler(sql):4,'|',QLT:4,PDS:4,INR:4,'|');
     write(sigler(sql):4);
     for a:=1 to care do begin ff:=round(1000*Fa[a]);
         CTR:=round(1000*fre*sqr(Fa[a])/lama[a]);
      if (CTR>999) then CTR:=999;
      if (CTR<-999) then CTR:=-999;
      if (d2=0) then CO2:=0 else CO2:=round(1000*sqr(Fa[a])/d2) ;
      write(ft,ff:5,CO2:4,CTR:4,'|'); write(ff:7)  end;
     writeln(ft);writeln;
     end;
```

La procédure publiée ici ne diffère de celle de IB§2.1.2 qu'en ce que, la masse des éléments supplémentaires étant d'un ordre de grandeur quelconque vis-à-vis de celle des éléménts du tableau de base, on peut craindre que n'apparaissent des nombres de format insolite.

4.1.3 Écriture, sur un fichier binaire de facteurs

```
procedure ecriclun;begin
 rewrite(firi,concat(nomba,six,'Fac.w'));
 ttri^:=concat(six,':',titre);
 if (127<length(ttri^)) then ttri^:= copy(ttri^,1,127);
 for a:=1 to carf do begin
   plww:=pww(@tampi[63+(2*a)]); plww^:=lama[a] end;
 plww:=pww(@tampi[65+(2*carf)]); plww^:=trace;
 tampi[titab]:=carh;
 for u:=1 to titab do write(firi,tampi[u]);write(firi,2+carf);
 for a:=1 to carf do begin
  if (a<10) then sgli^:=zigler(concat('axe',chr(48+a))) else
     sgli^:=zigler(concat('ax',chr(48+(a div 10)),chr(48+(a mod 10))));
  for u:=1 to 3 do write(firi,tampi[u]) end;
 sgli^:=zigler('poid');for u:=1 to 3 do write(firi,tampi[u]);
 sgli^:=zigler('dis2');for u:=1 to 3 do write(firi,tampi[u]);end;
procedure recompter;begin
  open(firi,concat(nomba,six,'Fac.w'));
  seek(firi,titab-1);write(firi,karh);close(firi) end;
procedure riclage;begin
  with riri do begin sis:=sql;
     for a:=1 to carf do Fac[a]:=Fa[a];
      Fac[carf+1]:=fre;Fac[carf+2]:=d2;
      for u:=1 to dir do write(firi,ptrc1^[u]);end;end;
```

On retrouve ici des procédures, 'ecriclun' et 'riclage', déjà vues dans IB§§2.1.3, 2.1.4; respectivement: création d'un fichier binaire de facteurs et écriture de l'en-tête; et: écriture des facteurs pour un élément. Quant à la procédure 'recompter', il faut prendre garde que le nombre des lignes, carh, figurant en tête d'un tableau écrit en format de texte (ou la valeur 1000, prise par défaut; cf. IA§§1.1, 2.5.2) n'a valeur que de borne supérieure pour la lecture du tableau: il se peut que le nombre effectif des lignes lues, karh, soit strictement inférieur à carh. Dans ce cas, quand est achevée l'écriture des facteurs afférents au tableau externe traité, il convient de corriger le nombre des lignes, placé au rang 'titab', en tête du fichier 'Fac.w'. C'est également parce que le tableau 'Fac.w' est écrit ligne par ligne, avant d'être entièrement calculé, que la procédure 'ecriclun', donnée ici, n'écrit pas de valeur 'mtot', comme le fait celle de IB§2.1.3.

4.1.4 Lecture d'un élément sur un tableau externe

```
procedure ligner;begin carac:=' ';lecsig;jh:=1;
 if not (sig='') then for j:=1 to carte do begin
   if (iz2=0) then begin lecnombre;fhe^[jh]:=ffl end
              else begin read(ftu,ret);fhe^[jh]:=ret end;
   jh:=jh+pre^[j] end;end;
procedure lignez;begin
 for c:=1 to 3 do read(fin,tampi[c]);sig:=sigler(sgli^);jh:=1;
 for j:=1 to carte do begin read(fin,tampi[1]);read(fin,tampi[2]);
   if (iz2=1) then fhe^[jh]:=wwi^ else fhe^[jh]:=ffli^;
   jh:=jh+pre^[j] end;end;
```

Selon que le tableau est en format de texte, (iz1=2; nom sans suffixe, iz2=0; ou avec suffixe: 'yy', iz2=3; cf.§2) ou en format binaire, (iz1=1; avec suffixe: '.z', iz2=2; ou :'.w', iz2=1), on utilise la procédure 'ligner', ou la procédure 'lignez'.

4.1.5 Calcul des facteurs pour un élément: la procédure 'preligni'

```
procedure preligni;begin
     if (ke=0) then res:=1 else res:=1/ke;d2:=-1;
     for j:=1 to carel do begin
         fhe^[j]:=fhe^[j]*res;
         if not(fe^[j]=0) then d2:=d2 +(sqr(fhe^[j])/fe^[j]) end;
     if (d2<0) then d2:=0;
     for a:=1 to carf do begin Fa[a]:=0;
         for j:=1 to carel do Fa[a]:=Fa[a]+(fhe^[j]*Phae[a]^[j]) end;end;
```

Cette procédure ne diffère de celle, de même nom, donnée en IB§2.1.1 que dans le calcul du profil de la ligne, fondé ici sur la connaissance du total, déjà calculé et gardé en ke, avant l'appel de 'preligni'.

4.2 Explication du corps de la procédure 'tabsuper'

4.2.1 Affectation des pointeurs vers les résultats de l'analyse de base

```
procedure tabsuper;begin writeln('début tabsuper ');
 che:=six[length(six)];iz1:=spus div 4;iz2:=spus mod 4;
 if (che='a') then begin
    carel:=carj;carte:=cartj;fe:=fj;pre:=prj;chh:='i';chc:='j';sise:=sisj;
    for a:=1 to aamax do Phae[a]:=Phaj[a] end
  else begin
    carel:=cari;carte:=carti;fe:=fi;pre:=pri;chh:='j';chc:='i';sise:=sisi;
    for a:=1 to aamax do Phae[a]:=Phai[a] end;
 writeln('fin d''affectation des Phae ');
```

Les paramètres, iz1 iz2, donnant le type du tableau externe, sont pris dans le nombre unique spus=spn[ns] (§0, *in fine*; §2, listage de 'sigsuper', *in fine*). D'après la dernière lettre du sigle ('a' ou 'b'), on fixe les scalaires et les pointeurs afférents aux colonnes ou aux lignes du tableau de base.

4.2.2 Vérification de l'en-tête du tableau externe: concordance des sigles avec ceux du tableau de base

```
 chv:='O';
 if (iz1=2) then begin
   if (iz2=0) then sur:='' else sur:='yy';
   reset(ftu,concat(nomba,six,sur));
   readln(ftu,titre); carac:=' ';writeln(titre);
   lecnombre;carh:=ffl div 1000;
   if not(carte=ffl mod 1000) then begin chv:='N';
      writeln('ERREUR sur le nombre des ',che);
      write('crdpr =',carte:4,'   crdsp =',ffl mod 1000:4);readln(rpf);end;
   writeln(six,' : ',titre);jh:=0;
   for j:=1 to carte do if (chv='O') then begin
     lecsig;eps:=pre^[j];jh:=jh+eps;
     if (eps=1) and not(sig=sigler(sise^[jh])) then begin chv:='N';
       writeln('ERREUR sur les sigles des ',chc);
       write('sgpr =',sigler(sise^[jh]):5,'   sgsp =',sig:5);readln(rpf) end;
     end;
   if ((chv='O') and (carh=0)) then carh:=1000;
   close(ftu) end;
```

4.2.2.1 Cas d'un tableau externe en format de texte

Le nombre, carh, des lignes du tableau externe (cf. §4.1.3, procédure: 'recompter') peut être calculé par quotient ou majoré par 1000. Le programme affiche tout désaccord, avec le tableau de base, quant au nombre des colonnes du tableau externe et aux sigles (des éléments principaux).

4.2.2.2 Cas d'un tableau externe en format binaire

```
if (iz1=1) then begin
  if (iz2=2) then sur:='.z' else sur:='.w';
  reset(fin,concat(nomba,six,sur));
  writeln(concat(nomba,six,sur));
  for c:=1 to titab do read(fin,tampi[c]);titre:=ttri^;carh:=tampi[titab];
  read(fin,lu);
  if not(carte=lu) then begin chv:='N';
     writeln('ERREUR sur le nombre des ',che);
     write('crdpr =',carte:4,'   crdsp =',lu:4);readln(rpf);end;
  writeln(six,' : ',titre);jh:=0;
  for j:=1 to carte do if (chv='O') then begin eps:=pre^[j];jh:=jh+eps;
    for c:=1 to 3 do read(fin,tampi[c]);sig:=sigler(sgli^);
    if (eps=1) and not(sig=sigler(sise^[jh])) then begin chv:='N';
       writeln('ERREUR sur les sigles des ',chc);
   write('sgpr =',sigler(sise^[jh]):5,'   sgsp =',sig:5);readln(rpf) end;
    end;
  close(fin) end;writeln('fin de lecture');
```

Comme, dans le cas du format de texte, on ouvre le tableau externe, afin de lire le nombre des colonnes et d'en vérifier les sigles (du moins ceux des colonnes principales; seuls connus du programme, d'après le fichier, 'Fac.w', donnant les facteurs des éléments figurant en principal dans l'analyse du tableau de base). On ferme le tableau externe par: close(fin). Et, si aucune incompatibilité n'est apparue entre tableau externe et tableau de base, la lettre chv donne la réponse 'O' (oui); et l'on passe au calcul des facteurs.

4.2.3 Calcul des facteurs pour les lignes du tableau externe

4.2.3.1 Création de l'en-tête du fichier binaire et du listage; nouvelle lecture de l'en-tête du tableau externe

```
if (chv='O') then begin
  nomf:=concat(nomba,six,'corsutx');rewrite(ft,nomf);
  writeln(ft,nomf,' : tableau ajouté');writeln(nomf);
  ecriclun;writeln(ft,titre);
  for a:=1 to 19+(14*care) do write(ft,'_'); writeln(ft);
  write(ft,'|SIG',chr(ord(chh)-32),'| QLT PDS INR|');
  for a:=1 to care do write(ft,'  F',a:2,' CO2 CTR|'); writeln(ft);
  for a:=1 to 19+(14*care) do write(ft,'_'); writeln(ft);
  if (iz1=2) then begin
    reset(ftu,concat(nomba,six,sur));
    readln(ftu, titre); carac:=' ';lecnombre;
    for j:=1 to carte do lecsig end;
  if (iz1=1) then begin
    reset(fin,concat(nomba,six,sur));
    for c:=1 to titab+1+(3*carte) do read(fin,lu) end;
  karh:=0;
```

L'en-tête du fichier binaire, destiné à recevoir les facteurs calculés pour le tableau extene, est créé par 'ecriclun' (cf. *supra*, §4.1.3). L'en-tête du listage est créé avec, pour les facteurs, 'care' blocs de colonnes (cf. §3.2, *in fine*). Quel que soit le format du tableau de base, l'en-tête en est lu jusqu'à parvenir à l'enregistrement afférent au premier élément à adjoindre à l'analyse de base.

4.2.3.2 Boucle de traitement des éléments du tableau externe

```
for ih:=1 to carh do begin
  if (izl=2) then ligner else lignez;ke:=0;
  if not(sig='') then begin karh:=karh+1;
    for j:=1 to carel do ke:=ke+fhe^[j];fre:=ke/mtot;sql:=zigler(sig);
    preligni;lignage;riclage end end;
for a:=1 to 19+(14*care) do write(ft,'_');writeln(ft);writeln;
if (izl=2) then close(ftu) else close(fin);
close(ft);close(firi);if not(carh=karh) then recompter end;end;
```

Les éléments du tableau externe sont traités successivement; au fur et à mesure de la lecture (par 'ligner' dans le cas du format de texte; par 'lignez', pour le format binaire: cf. §4.1.4).

Nous rappelons ce qui a été annoncé dès le début du §0: qu'il s'agisse de lignes ou de colonnes supplémentaires, relativement au tableau de base, les éléments externes sont toujours lus comme des lignes.

Le calcul des facteurs se fait par 'preligni' (cf. §4.1.5); leur inscription sur le listage et le fichier binaire, respectivement par 'lignage' (§4.1.2) et 'riclage' (§4.1.3).

La procédure 'tabsuper' termine en fermant le fichier du tableau externe (ftu, si c'est un texte; fin, pour le format binaire) ainsi que les fichiers qu'elle a créés (listage: ft; et fichier binaire, écrit au format entier: firi). Le nombre, carh, des élément du tableau externe, est corrigé, s'il y a lieu, sur le fichier binaire des facteurs (rouvert à cet effet; puis refermé).

5 Libération des zones affectées, en mémoire centrale, aux résultats de l'analyse du tableau de base: la procédure 'fermer'

```
procedure fermer; begin
  if (aaa=1) then begin
     for a:=1 to aamax do dispose(Phaj[a]);
     dispose(fj);dispose(sisj);dispose(prj) end;
  if (bbb=1) then begin
     for a:=1 to aamax do dispose(Phai[a]);
     dispose(fi);dispose(sisi);dispose(pri) end;
  dispose(fhe);end;
```

Ainsi qu'on l'a dit au §0, *in fine*, la procédure 'fermer' libère l'espace réservé en mémoire centrale pour garder les résultats de l'analyse du tableau de base: fréquence marginale; sigles; caractérisation des éléments principaux; facteurs.

On notera que la procédure comprend deux instructions composées: if (aaa=1) then..., et: if (bbb=1) then...; jouant, respectivement, (cf. §3.2), selon qu'existe ou non un tableau externe (ou plusieurs) dont le sigle se termine par 'a' (lignes supplémentaires); ou 'b' (colonnes supplémentaires).

Avec **'fermer'** s'achève la liste des procédures appelées par le programme principal; dont le listage a déjà été commenté au §0.

Références bibliographiques pour les quatre chapitres [SOURCES PASCAL IA, B, C, D]

J.-P. & F. BENZÉCRI : "Programmes d'analyse des correspondances et de classification ascendante hiérarchique: notice d'utilisation"; [NOT. CORR. CAH]; in *CAD*, Vol.XIV, n°1; pp. 7-34; (1989);

J.-P. & F. BENZÉCRI : "Programmes de création de tableaux: notice d'utilisation"; [NOT. CORR. CAH]; in *CAD*, Vol.XIV, n°1; pp. 35-54; (1989);

K. Ben SALEM : "Associativité de la fusion et parallélisme dans les algorithmes de tri"; [ASS. FUS. TRI]; in *CAD*, Vol.XV, n°2; pp. 133-138; (1990).

J.-P. & F. BENZÉCRI : "Liste des programmes du logiciel MacSAIF, Système d'Analyse des InFormations, et description de leurs fonctions"; [MacSAIF]; in *CAD*, Vol.XV, n°3; pp. 359-366; (1990).

J.-P. & F. BENZÉCRI : "État des modifications apportées aux versions du logiciel MacSAIF"; [ÉTAT MacSAIF]; in *CAD*, Vol.XVIII, n°3; pp. 321-344; (1993).

N.B. On trouve divers formulaires et exposés théoriques généraux de l'analyse des correspondances dans le Tome II du traité sur "*L'Analyse des Données*"; et dans les volumes de la série: "*Pratique de l'Analyse des Données*".