ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XV, 1 (1976)

ALGORITHM 41

ANNA BARTKOWIAK (Wrocław)

INTERDEPENDENCE EXAMINATIONS BY ANALYSIS OF REGRESSION

1. Procedure declaration. Given the number of variables n and the optional number of representatives r, the procedure *idep* performs a systematic examination of all subsets to select that one which, while considering the regression of the omitted variables upon the selected subset, yields the minimax of residual variances.

Data:

n — number of variables;

r — optional number of representatives (the size of the subset to be chosen);

$c[1 : q\times(q+1)\div 2]$ — lower triangle of the correlation matrix (with diagonals), $q \geqslant n$ depends upon the Boolean variables *sq* and *prod* as explained below;

sq — Boolean variable assuming the value **true** if the regression with squares is considered; thus $q = 2n$;

prod — Boolean variable assuming the value **true** if the regression with product terms is considered; thus $q = 2n+n(n-1)/2$;

combi — procedure identifier of the procedure yielding subsequent combinations of r out of n objects, headed as follows:
procedure *combi* $(n, r, ind, first)$; **value** n, r;
integer n, r; **integer array** *ind*; **Boolean** *first*;

The procedure *COMB1* of Mifsud [2] can be applied here.

Results:

optind$[1 : r]$ — integer array enumerating the selected combination of variables (the representative subset of variables);

res$[0 : n]$ — residual sums of squares for the n variables under consideration; *res*$[0]$ stands for the maximum of all residuals; the residuals for the selected set of variables are assumed to be equal to zero.

```
procedure idep(n,r,c,sq,prod,combi,optind,res);
 value n,r,sq,prod;
 integer n,r;
 array c,res;
 integer array optind;
 Boolean sq,prod;
 procedure combi;
 begin
  integer dep,h,i,i1,i2,i3,j,k,p,pr,q,qr;
  real x,y,min,max;
  Boolean first;
  h:=r;
  p:=n;
  if sq
   then
   begin
    p:=p+n;
    h:=h+r
   end sq;
  if prod
   then
   begin
    p:=p+n×(n-1)÷2;
    h:=h+r×(r-1)÷2
   end prod;
  if r<n
   then
   begin
    integer array ind,ii[1:p];
    array a[1:h×(h+1)÷2],b[1:(n-r)×p],dt[1:p];
```

```
    max:=1.0;
    pr:=r;
    if sq
     then pr:=pr+r;
    if prod
     then pr:=pr+r×(r-1)+2;
    qr:=pr×(pr+1)+2;
    first:=true;
newcomb:
    combi(n,r,ind,first);
    if first
     then go to fin;
    for i:=1 step 1 until p do
     ii[i]:=0;
    for i:=1 step 1 until r do
     ii[ind[i]]:=1;
    if sq
     then
     for i:=1 step 1 until r do
      ii[n+ind[i]]:=1;
    if prod
     then
     for i:=1 step 1 until r-1 do
      begin
       j:=ind[i];
       i1:=n-j;
       i1:=p-i1×(i1+1)+2-j;
       for j:=i+1 step 1 until r do
        ii[i1+ind[j]]:=1
      end prod;
```

```
k:=i1:=0;
for i:=1 step 1 until p do
 begin
  if ii[i]=1
   then
   for j:=1 step 1 until i do
    if ii[j]=1
     then
     begin
      k:=k+1;
      a[k]:=c[i1+j]
     end ii[j]=1;
  i1:=i1+i
 end i;
min:=.0;
q:=0;
for dep:=1 step 1 until n do
 if ii[dep]=0
  then
  begin
   j:=dep×(dep-1)+2;
   for i:=1 step 1 until dep do
    dt[i]:=c[j+i];
   for j:=dep+1 step 1 until p do
    dt[j]:=c[j×(j-1)+2+dep];
   for i:=1 step 1 until p do
    if ii[i]=1
     then
     begin
      q:=q+1;
```

```
        b[q]:=dt[i]
       end i;
     q:=q+1;
     b[q]:=1.0
    end dep;
  k:=pr+1;
  i1:=0;
  for q:=1 step 1 until pr do
   begin
     i1:=i1+q;
     x:=a[i1];
     if x>.0
      then
      begin
       x:=-1.0/x;
       i2:=i1+q;
       for i:=q+1 step 1 until pr do
        begin
         y:=dt[i]:=a[i2];
         y:=y×x;
         for j:=q+1 step 1 until i do
          begin
           i2:=i2+1;
           a[i2]:=a[i2]+y×dt[j]
          end j;
         i2:=i2+q
        end i;
       i3:=0;
       for i:=1 step 1 until n do
        if ii[i]=0
```

```
          then
          begin
           i3:=i3+q;
           y:=dt[k]:=b[i3];
           y:=y×x;
           for j:=q+1 step 1 until k do
            begin
             i3:=i3+1;
             b[i3]:=b[i3]+y×dt[j]
            end j
          end i
      end x>.0
    end q;
   i3:=k;
   for i:=n-r step -1 until 1 do
    begin
     x:=b[i3];
     if x>min
      then min:=x;
     i3:=i3+k
    end i;
   if min<max
    then
    begin
     res[0]:=max:=min;
     for i:=1 step 1 until r do
      optind[i]:=ind[i];
     i3:=0;
     for i:=1 step 1 until n do
      if ii[i]=1
```

```
            then res[i]:=.0
            else
            begin
              i3:=i3+k;
              res[i]:=b[i3]
            end ii[i]≠1
         end min>max;
        if ¬first
         then go to newcomb:
  fin:
     end
    end idep
```

2. Method used. The essential features of the method used are described in the paper of Beale [1]. Suppose that the variables $x_1, x_2, \ldots, x_n$ are considered. The first subset of indices obtained by *combi* is $1, 2, \ldots, r$; thus the variables indexed as $r+1, \ldots, n$ are dropped. For each of the dropped variables we calculate the residual sum of squares assuming the regression relationship of the following forms $(l = r+1, \ldots, n)$:

(a) if $sq \equiv$ **false** and $prod \equiv$ **false**, then

$$x_l = b_0 + b_1 x_1 + \ldots + b_r x_r;$$

(b) if $sq \equiv$ **true** and $prod \equiv$ **false**, then

$$x_l = b_0 + b_1 x_1 + \ldots + b_r x_r + b_{r+1} x_1^2 + \ldots + b_{2r} x_r^2;$$

(c) if $sq \equiv$ **true** and $prod \equiv$ **true**, then

$$x_l = b_0 + b_1 x_1 + \ldots + b_{2r} x_r^2 + b_{2r+1} x_1 x_2 + b_{2r+2} x_1 x_3 + \ldots + b_{2r+r(r-1)/2} x_{r-1} x_r.$$

We calculate the residual sum of squares for $l = r+1, \ldots, n$ and mark the maximum value $maxres(1, \ldots, r)$. Next we continue examining further subsets $(i_1, \ldots, i_r)$ yielded by subsequent calls of *combi*. We choose as a representative set that one which gives the minimum of the *maxres* values $maxres(i_1, \ldots, i_r)$ calculated for each subset.

3. Certification.

Example 1. Calling *idep* with the values

$$n = 3, \quad r = 2,$$

$$c[1:6] = [1.0000, .4899, 1.0000, .4899, -.5000, 1.0000],$$

$$sq \equiv false, \quad prod \equiv false,$$

we get the following results:

$$optind[1:2] = [1, 2], \quad res[0:3] = [.0395, .0, .0, .0395].$$

Example 2. Calling *idep* with the values

$$n = 3, \quad r = 1,$$

$c[1:15]$ = [1.0000, .4899, 1.0000, .4899, −.5000, 1.0000, .9883, .4721, .4721, 1.0000, .5094, .9878, −.4939, .5065, 1.0000, .5094, −.4939, .9878, .5065, −.4797, 1.00000],

$$sq \equiv \mathbf{true}, \quad prod \equiv \mathbf{false},$$

we get the following results:

$$optind[1:1] = [2], \quad res[0:3] = [.7500, .7223, 0.0000, .7500].$$

Example 3. Calling *idep* with the values

$$n = 3, \quad r = 2,$$

$c[1:36]$ = [1.0000, .4899, 1.0000, .4899, −.5000, 1.0000, .9883, .4721, .4721, 1.0000, .5094, .9878, −.4939, .5056, 1.0000, .5094, −.4939, .9878, .5065, −.4797, 1.0000, .7340, .9350, −.2158, .7316, .9534, −.2056, 1.0000, .7340, −.2158, .9350, .7316, −.2056, .9534, .0862, 1.0000, .8581, .4671, .4671, .8120, .4250, .4250, .6439, .6439, 1.0000],

$$sq \equiv \mathbf{true}, \quad prod \equiv \mathbf{true},$$

we get following results:

$$optind[1:2] = [2, 3], \quad res[0:3] = [.0, .0, .0\ .0].$$

References

[1] E. M. L. Beale, M. G. Kendall and D. W. Mann, *The discarding of variables in multivariate analysis*, Biometrika 54 (1967), p. 357-366.

[2] Ch. J. M. Mifsud, *Algorithm 154, Combinations in lexicographical order, procedure COMB1*, Comm. ACM 6 (1963), p. 103.

INSTITUTE OF INFORMATICS
UNIVERSITY OF WROCŁAW
50-384 WROCŁAW

Received on 26. 5. 1974

ALGORYTM 41

ANNA BARTKOWIAK (Wrocław)

BADANIE WSPÓŁZALEŻNOŚCI CECH METODĄ ANALIZY REGRESJI

STRESZCZENIE

Procedura *idep* przeszukuje w sposób systematyczny wszystkie podzbiory r-elementowe ($r < n$) danego zespołu zmiennych $(x_1, \ldots, x_n)$, wybierając ten podzbiór $(x_{i_1}, \ldots, x_{i_r})$, na podstawie którego można wyznaczyć pozostałe zmienne z najmniejszą wariancją resztową. W zależności od zmiennych boolowskich *sq* i *prod* uwzględnia się również regresję z kwadratami i iloczynami zmiennych (por. wzory (a), (b) i (c)).