nag_mv_hierar_cluster_analysis (g03ecc)

1. Purpose

nag_mv_hierar_cluster_analysis (g03ecc) performs hierarchical cluster analysis.

2. Specification

```c
#include <nag.h>
#include <nag03.h>

void nag_mv_hierar_cluster_analysis(Nag_ClusterMethod method, Integer n, double d[],
        Integer ilc[], Integer iuc[], double cd[], Integer iord[],
        double dord[], NagError *fail)
```

3. Description

Given a distance or dissimilarity matrix for \( n \) objects (see nag_mv_distance_mat (g03eac)), cluster analysis aims to group the \( n \) objects into a number of more or less homogeneous groups or clusters. With agglomerative clustering methods, a hierarchical tree is produced by starting with \( n \) clusters, each with a single object and then at each of \( n-1 \) stages, merging two clusters to form a larger cluster, until all objects are in a single cluster. This process may be represented by a dendrogram (see nag_mv_dendrogram (g03ecc)).

At each stage, the clusters that are nearest are merged, methods differ as to how the distance between the new cluster and other clusters are computed. For three clusters \( i, j \) and \( k \) let \( n_i, n_j \) and \( n_k \) be the number of objects in each cluster and let \( d_{ij}, d_{ik} \) and \( d_{jk} \) be the distances between the clusters. Let clusters \( j \) and \( k \) be merged to give cluster \( jk \), then the distance from cluster \( i \) to cluster \( jk \), \( d_{i,jk} \) can be computed in the following ways.

1. Single link or nearest neighbour : \( d_{i,jk} = \min(d_{ij}, d_{ik}) \).
2. Complete link or furthest neighbour : \( d_{i,jk} = \max(d_{ij}, d_{ik}) \).
3. Group average : \( d_{i,jk} = \frac{n_i}{n_i + n_j + n_k} d_{ij} + \frac{n_j}{n_i + n_j + n_k} d_{ik} \).
4. Centroid : \( d_{i,jk} = \frac{n_i}{n_i + n_j + n_k} d_{ij} + \frac{n_j}{n_i + n_j + n_k} d_{ik} - \frac{n_i n_j}{(n_i + n_j + n_k)} d_{jk} \).
5. Median : \( d_{i,jk} = \frac{1}{2} d_{ij} + \frac{1}{2} d_{ik} - \frac{1}{4} d_{jk} \).
6. Minimum variance : \( d_{i,jk} = \{(n_i + n_j) d_{ij} + (n_i + n_k) d_{ik} - n_i d_{jk}\}/(n_i + n_j + n_k) \).

For further details see Everitt (1974) or Krzanowski (1990).

If the clusters are numbered \( 1, 2, \ldots, n \) then, for convenience, if clusters \( j \) and \( k \), \( j < k \), merge then the new cluster will be referred to as cluster \( j \). Information on the clustering history is given by the values of \( j, k \) and \( d_{jk} \) for each of the \( n-1 \) clustering steps. In order to produce a dendrogram, the ordering of the objects such that the clusters that merge are adjacent is required. This ordering is computed so that the first element is 1. The associated distances with this ordering are also computed.

4. Parameters

**method**

Input: indicates which clustering.

- If `method` = Nag_SingleLink, single link.
- If `method` = Nag_CompleteLink, complete link.
- If `method` = Nag_GroupAverage, group average.
- If `method` = Nag_Centroid, centroid.
- If `method` = Nag_Median, median.
- If `method` = Nag_MinVariance, minimum variance.

Constraint: `method` = Nag_SingleLink, Nag_CompleteLink, Nag_GroupAverage, Nag_Centroid, Nag_Median or Nag_MinVariance.
nag_mv_hierar_cluster_analysis

n
Input: the number of objects, n.
Constraint: n ≥ 2.

d[n*(n−1)/2]
Input: the strictly lower triangle of the distance matrix. D must be stored packed by rows, i.e., d[(i−1)(i−2)/2 + j−1], i > j must contain d_{ij}.
Constraint: d[i−1] ≥ 0.0, for i = 1, 2, ..., n(n−1)/2.

ilc[n−1]
Output: ilc[l−1] contains the number, j, of the cluster merged with cluster k (see iuc), j < k, at step l for l = 1, 2, ..., n−1.

iuc[n−1]
Output: iuc[l−1] contains the number, k, of the cluster merged with cluster j, j < k, at step l for l = 1, 2, ..., n−1.

cd[n−1]
Output: cd[l−1] contains the distance d_{jk}, between clusters j and k, j < k, merged at step l for l = 1, 2, ..., n−1.

iord[n]
Output: the objects in dendrogram order.

dord[n]
Output: the clustering distances corresponding to the order in iord. dord[l−1] contains the distance at which cluster iord[l−1] and iord[l] merge, for l = 1, 2, ..., n−1. dord[n−1] contains the maximum distance.

fail
The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_BAD_PARAM
On entry, parameter method had an illegal value.

NE_INT_ARG_LT
On entry, n must not be less than 2: n = ⟨value⟩.

NE_REALARR
On entry, d[⟨value⟩] = ⟨value⟩.
Constraint: d[i−1] ≥ 0.0, i = 1, 2, ..., n * (n−1)/2.

NE_DENDROGRAM
A true dendrogram cannot be formed because the distances at which clusters have merged are not increasing for all steps, i.e., cd[i−1] < cd[i−2] for some i = 2, 3, ..., n−1.
This can occur for the Nag_Centroid and Nag_Median methods.

NE_ALLOC_FAIL
Memory allocation failed.

NE_INTERNAL_ERROR
An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

6. Further Comments

The dendrogram may be formed using nag_mv_dendrogram (g03ehc). Groupings based on the clusters formed at a given distance can be computed using nag_mv_cluster_indicator (g03ejc).
6.1. Accuracy

For methods other than Nag_SingleLink or Nag_CompleteLink, slight rounding errors may occur in the calculations of the updated distances. These would not normally significantly affect the results, however there may be an effect if distances are (almost) equal.

If at a stage, two distances \( d_{ij} \) and \( d_{kl} \), \( i < k \) or \( i = k \) and \( j < l \), are equal then clusters \( k \) and \( l \) will be merged rather than clusters \( i \) and \( j \). For single link clustering this choice will only affect the order of the objects in the dendrogram. However, for other methods the choice of \( kl \) rather than \( ij \) may affect the shape of the dendrogram. If either of the distances \( d_{ij} \) or \( d_{kl} \) are affected by rounding errors then their equality, and hence the dendrogram, may be affected.

6.2. References


7. See Also

nag_mv_distance_mat (g03eac)
nag_mv_dendrogram (g03ehc)
nag_mv_cluster_indicator (g03ejc)

8. Example

Data consisting of three variables on five objects are read in. Euclidean squared distances based on two variables are computed using nag_mv_distance_mat (g03eac), the objects are clustered using nag_mv_hierar_cluster_analysis and the dendrogram computed using nag_mv_dendrogram (g03ehc). The dendrogram is then printed.

8.1. Program Text

/* nag_mv_hierar_cluster_analysis (g03ecc) Example Program. */
* Mark 5, 1998.*
*/
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg03.h>
#define NMAX 10
#define MMAX 10
#define LENC 20

main()
{
    double cd[NMAX-1], d[NMAX*(NMAX-1)/2], dord[NMAX],
        s[MMAX], x[NMAX][MMAX];
    double dmin_, dstep, ydist;
    Integer ilc[NMAX-1], iord[NMAX], isx[MMAX], iuc[NMAX-1];
    Integer nsym;
    Integer i, j;
    Integer m, n;
    Integer int_method;
    Integer tdx = MMAX;

    char name[NMAX][2];
    char char_dist[2];
    char **c=0;
    char char_scale[2];
    char char_update[2];
Nag_ClusterMethod method;
Nag_MatUpdate update;
Nag_DistanceType dist;
Nag_VarScaleType scale;

Vprintf("g03ecc Example Program Results\n\n");

/* Skip heading in data file */
Vscanf("%*[\n]");
Vscanf("%ld",&n);
Vscanf("%ld",&m);
if (n <= NMAX && m <= MMAX)
{
    Vscanf("%ld",&int_method);
    if (int_method == 1)
        method = Nag_SingleLink;
    else if (int_method == 2)
        method = Nag_CompleteLink;
    else if (int_method == 3)
        method = Nag_GroupAverage;
    else if (int_method == 4)
        method = Nag_Centroid;
    else if (int_method == 5)
        method = Nag_Median;
    else
        method = Nag_MinVariance;

    Vscanf("%s",char_update);
    if (*char_update == 'U')
        update = Nag_MatUp;
    else
        update = Nag_NoMatUp;

    Vscanf("%s",char_dist);
    if (*char_dist == 'A')
        dist = Nag_DistAbs;
    else if (*char_dist == 'E')
        dist = Nag_DistEuclid;
    else
        dist = Nag_DistSquared;

    Vscanf("%s",char_scale);
    if (*char_scale == 'S')
        scale = Nag_VarScaleStd;
    else if (*char_scale == 'R')
        scale = Nag_VarScaleRange;
    else if (*char_scale == 'G')
        scale = Nag_VarScaleUser;
    else
        scale = Nag_NoVarScale;

    for (j=0 ;j<n ; ++j)
    {
        for (i=0 ;i<m ; ++i)
            Vscanf("%lf",&x[j][i]);
        Vscanf("%s",name[j]);
    }

    for (i=0 ;i<m ; ++i)
        Vscanf("%ld",&isx[i]);
    for (i=0 ;i<m ; ++i)
        Vscanf("%lf",&s[i]);

    /* Compute the distance matrix */
g03eac(update, dist, scale, n, m, (double *)x, tdx, isx, s, d, NAGERR_DEFAULT);

    /* Perform clustering */
g03ecc(method, n, d, ilc, iuc, cd, iord, dord, NAGERR_DEFAULT);
    Vprintf("\n Distance Clusters Joined\n\n");
    for (i = 1; i <= n-1; ++i)
```c
Vprintf("\%10.3f %3s%3s\n", cd[i-1], name[ilc[i-1]-1], name[iuc[i-1]-1]);
/* Produce dendrogram */
nsym = 20;
dmin_ = 0.0;
dstep = cd[n - 2] / (double) nsym;
g03ehc(Nag_DendSouth, n, dord, dmin_, dstep, nsym, &c, NAGERR_DEFAULT);
Vprintf("\n");
Vprintf("Dendrogram ");
Vprintf("\n");
ydist = cd[n - 2];
for (i = 0; i < nsym; ++i)
{
    if ((i+1) % 3 == 1)
    {
        Vprintf("\%10.3f%6s",ydist," ");
        Vprintf("%s",c[i]);
        Vprintf("\n");
    }
    else
    {
        Vprintf("%6s","");
        Vprintf("\n");
    }
    ydist -= dstep;
}
Vprintf("\n");
Vprintf("%14s","");
for (i = 0; i < n; ++i)
{
    Vprintf("%3s",name[iord[i]-1]);
}
Vprintf("\n");
g03xxzc(&c);
exit(EXIT_SUCCESS);
}
else
{
    Vprintf("Incorrect input value of n or m.\n");
    exit(EXIT_FAILURE);
}
}

8.2. Program Data

g03ecc Example Program Data
5 3
5
1 5.0 2.0 A
2 1.0 1.0 B
3 4.0 3.0 C
4 1.0 2.0 D
5 5.0 0.0 E
0 1 1
1.0 1.0 1.0
```
8.3. Program Results

g03ecc Example Program Results

<table>
<thead>
<tr>
<th>Distance</th>
<th>Clusters Joined</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>B D</td>
</tr>
<tr>
<td>2.000</td>
<td>A C</td>
</tr>
<tr>
<td>6.500</td>
<td>A E</td>
</tr>
<tr>
<td>14.125</td>
<td>A B</td>
</tr>
</tbody>
</table>

Dendrogram

<table>
<thead>
<tr>
<th>Distance</th>
<th>Clusters Joined</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.125</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>12.006</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>9.887</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>7.769</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>5.650</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>3.531</td>
<td>I I I I I I I I</td>
</tr>
<tr>
<td>1.412</td>
<td>I I I I I I I I</td>
</tr>
</tbody>
</table>

A C E B D