NAG C Library Function Document

nag_binary_factor_service (g11sbc)

1 Purpose

nag_binary_factor_service (g11sbc) is a service routine which may be used prior to calling nag_binary_factor (g11sac) to calculate the frequency distribution of a set of dichotomous score patterns.

2 Specification

```c
void nag_binary_factor_service (Nag_OrderType order, Integer p, Integer n, Integer *ns, Boolean x[], Integer pdx, Integer irl[], NagError *fail)
```

3 Description

When each of \( n \) individuals responds to each of \( p \) dichotomous variables the data assumes the form of the matrix \( X \) defined below

\[
X = \begin{bmatrix}
  x_{11} & x_{12} & \cdots & x_{1p} \\
  x_{21} & x_{22} & \cdots & x_{2p} \\
  \vdots & \vdots & & \vdots \\
  x_{n1} & x_{n2} & \cdots & x_{np}
\end{bmatrix} = \begin{bmatrix}
  x'_1 \\
  x'_2 \\
  \vdots \\
  x'_n
\end{bmatrix},
\]

where the \( x \) take the value of 0 or 1 and \( x'_l = (x_{1l}, x_{2l}, \ldots, x_{pl})' \), for \( l = 1, 2, \ldots, n \) denotes the score pattern of the \( l \)th individual (denoting the transpose of a vector). nag_binary_factor_service (g11sbc) calculates the number of different score patterns, \( s \), and the frequency with which each occurs. This information can then be passed to nag_binary_factor (g11sac).

4 References

None.

5 Parameters

1: \( \text{order} \) – Nag_OrderType

\( \text{Input} \)

On entry: the \( \text{order} \) parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by \( \text{order} = \text{Nag_RowMajor} \). See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

Constraint: \( \text{order} = \text{Nag_RowMajor} \) or \( \text{Nag_ColMajor} \).

2: \( p \) – Integer

\( \text{Input} \)

On entry: the number of dichotomous variables, \( p \).

Constraint: \( p \geq 3 \).

3: \( n \) – Integer

\( \text{Input} \)

On entry: the number of individuals in the sample, \( n \).

Constraint: \( n \geq 7 \).

4: \( \text{ns} \) – Integer *

\( \text{Output} \)

On exit: the number of different score patterns, \( s \).
5: \( x[\text{dim}] \) – Boolean  
\textit{Input/Output} 

Note: the dimension, \( \text{dim} \), of the array \( x \) must be at least \( \max(1, \text{pdx} \times p) \) when \textit{order} = \texttt{Nag.ColMajor} and at least \( \max(1, \text{pdx} \times n) \) when \textit{order} = \texttt{Nag.RowMajor}.

Where \( X(i, j) \) appears in this document, it refers to the array element

\[
\text{if order = Nag.ColMajor, } x[(j - 1) \times \text{pdx} + i - 1]; \\
\text{if order = Nag.RowMajor, } x[(i - 1) \times \text{pdx} + j - 1].
\]

On entry: \( X(i, j) \) must be set equal to \texttt{TRUE} if \( x_{ij} = 1 \), and \texttt{FALSE} if \( x_{ij} = 0 \), for \( i = 1, 2, \ldots, n; j = 1, 2, \ldots, p \).

On exit: the first \( s \) rows of \( x \) contain the \( s \) different score patterns.

6: \( \text{pdx} \) – Integer  
\textit{Input} 

On entry: the stride separating matrix row or column elements (depending on the value of \textit{order}) in the array \( x \).

Constraints:

\[
\text{if order = Nag.ColMajor, } \text{pdx} \geq n; \\
\text{if order = Nag.RowMajor, } \text{pdx} \geq p.
\]

7: \( \text{irl}[n] \) – Integer  
\textit{Output} 

On exit: the frequency with which the \( l \)th row of \( x \) occurs, for \( l = 1, 2, \ldots, s \).

8: \( \text{fail} \) – NagError *  
\textit{Input/Output} 

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE_INT} 

On entry, \( n = \langle \text{value} \rangle \).  
Constraint: \( n \geq 7 \).

On entry, \( \text{pdx} = \langle \text{value} \rangle \).  
Constraint: \( \text{pdx} > 0 \).

On entry, \( p = \langle \text{value} \rangle \).  
Constraint: \( p \geq 3 \).

\textbf{NE_INT_2} 

On entry, \( \text{pdx} = \langle \text{value} \rangle \), \( n = \langle \text{value} \rangle \).  
Constraint: \( \text{pdx} \geq n \).

On entry, \( \text{pdx} = \langle \text{value} \rangle \), \( p = \langle \text{value} \rangle \).  
Constraint: \( \text{pdx} \geq p \).

\textbf{NE_ALLOC_FAIL} 

Memory allocation failed.

\textbf{NE_BAD_PARAM} 

On entry, parameter \( \langle \text{value} \rangle \) had an illegal value.

\textbf{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.
7 Accuracy

Exact.

8 Further Comments

The time taken by the routine is small and increases with \( n \).

9 Example

A program to count the frequencies of different score patterns in the following list:

Score Patterns
000
010
111
000
001
000
000
110
001
011

9.1 Program Text

/* nag_binary_factor_service (gllsbc) Example Program.  *
 * Copyright 2002 Numerical Algorithms Group.  *
 * Mark 7, 2002.  */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagll1.h>

int main(void)
{
    /* Scalars */
    Integer exit_status, i, p, ns, j, n, nrx, pdx;
    NagError fail;
    Nag_OrderType order;
    char flag;
    /* Arrays */
    Integer *irl = 0;
    Boolean *x = 0;

    #ifdef NAG_COLUMN_MAJOR
    #define X(I,J) x[(J-1)*pdx + I - 1]
    order = Nag_ColMajor;
    #else
    #define X(I,J) x[(I-1)*pdx + J - 1]
    order = Nag_RowMajor;
    #endif

    INIT_FAIL(fail);
    exit_status = 0;
    Vprintf("gllsbc Example Program Results\n");

    /* Skip heading in data file */
    Vscanf("%*\[\n");
    Vscanf("%ld%ld%*\[\n", &n, &p);

    ...
if (n > 0 && p > 0)
{
    /* Allocate arrays */
    nrx = n;
    if ( !(irl = NAG_ALLOC(n, Integer)) ||
        !(x = NAG_ALLOC(nrx * p, Boolean)) )
    {
        Vprintf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
    if (order == Nag_ColMajor)
        pdx = nrx;
    else
        pdx = p;
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= p; ++j)
        {
            Vscanf(" %c", &flag);
            X(i,j) = (flag == 'T');
        }
        Vscanf("%*[\n ]");
    }
    gllsbc(order, p, n, &ns, x, pdx, irl, &fail);
    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from gllsbc.\n", fail.message);
        exit_status = 1;
        goto END;
    }
    Vprintf("\n");
    Vprintf("Frequency Score pattern\n");
    Vprintf("\n");
    for (i = 1; i <= ns; ++i)
    {
        Vprintf("%5ld ", irl[i-1]);
        for (j = 1; j <= p; ++j)
        {
            if (X(i,j))
                flag = 'T';
            else
                flag = 'F';
            Vprintf(" %c", flag);
        }
        Vprintf("\n");
    }
}

END:
if (irl) NAG_FREE(irl);
if (x) NAG_FREE(x);
return exit_status;

9.2 Program Data

gllsbc Example Program Data
10 3
F F F
F T F
T T T
F F F
F T T
### 9.3 Program Results

**gllsbc Example Program Results**

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Score pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>F F F</td>
</tr>
<tr>
<td>1</td>
<td>F T F</td>
</tr>
<tr>
<td>2</td>
<td>T T T</td>
</tr>
<tr>
<td>1</td>
<td>F F T</td>
</tr>
<tr>
<td>1</td>
<td>T T F</td>
</tr>
<tr>
<td>1</td>
<td>F T T</td>
</tr>
</tbody>
</table>