

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

```
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

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

where the x take the value of 0 or 1 and $\underline{x}_l = (x_{l1}, x_{l2}, \dots, x_{lp})'$, for $l = 1, 2, \dots, n$ 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: **order** – Nag_OrderType *Input*
On entry: the **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 **order = Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.
Constraint: **order = Nag_RowMajor** or **Nag_ColMajor**.
- 2: **p** – Integer *Input*
On entry: the number of dichotomous variables, p .
Constraint: **p** \geq 3.
- 3: **n** – Integer *Input*
On entry: the number of individuals in the sample, n .
Constraint: **n** \geq 7.
- 4: **ns** – Integer * *Output*
On exit: the number of different score patterns, s .

5: **x**[*dim*] – Boolean *Input/Output*

Note: the dimension, *dim*, of the array **x** must be at least $\max(1, \mathbf{pdx} \times \mathbf{p})$ when **order** = **Nag_ColMajor** and at least $\max(1, \mathbf{pdx} \times \mathbf{n})$ when **order** = **Nag_RowMajor**.

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

if **order** = **Nag_ColMajor**, $\mathbf{x}[(j - 1) \times \mathbf{pdx} + i - 1]$;
 if **order** = **Nag_RowMajor**, $\mathbf{x}[(i - 1) \times \mathbf{pdx} + j - 1]$.

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

On exit: the first *s* rows of **x** contain the *s* different score patterns.

6: **pdx** – Integer *Input*

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

Constraints:

if **order** = **Nag_ColMajor**, $\mathbf{pdx} \geq \mathbf{n}$;
 if **order** = **Nag_RowMajor**, $\mathbf{pdx} \geq \mathbf{p}$.

7: **irl**[*n*] – Integer *Output*

On exit: the frequency with which the *l*th row of **x** occurs, for $l = 1, 2, \dots, s$.

8: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = *⟨value⟩*.
 Constraint: **n** ≥ 7.

On entry, **pdx** = *⟨value⟩*.
 Constraint: **pdx** > 0.

On entry, **p** = *⟨value⟩*.
 Constraint: **p** ≥ 3.

NE_INT_2

On entry, **pdx** = *⟨value⟩*, **n** = *⟨value⟩*.
 Constraint: **pdx** ≥ **n**.

On entry, **pdx** = *⟨value⟩*, **p** = *⟨value⟩*.
 Constraint: **pdx** ≥ **p**.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter *⟨value⟩* had an illegal value.

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 (g11sbc) Example Program.
 *
 * Copyright 2002 Numerical Algorithms Group.
 *
 * Mark 7, 2002.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg11.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("g11sbc 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] ");
    }

    g11sbc(order, p, n, &ns, x, pdx, irl, &fail);
    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from g11sbc.\n%s\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

```

g11sbc Example Program Data
10 3
F F F
F T F
T T T
F F F
F F T

```

F F F
F F F
T T F
F F T
F T T

9.3 Program Results

g11sbc Example Program Results

Frequency	Score pattern
4	F F F
1	F T F
1	T T T
2	F F T
1	T T F
1	F T T
