NAG Library Function Document

nag_ken_spe_corr_coeff (g02brc)

1 Purpose

nag_ken_spe_corr_coeff (g02brc) calculates Kendall and Spearman rank correlation coefficients.

2 Specification

```
#include <nag.h>
#include <nagg02.h>
```

```
void nag_ken_spe_corr_coeff (Integer n, Integer m, const double x[],
     Integer tdx, const Integer svar[], const Integer sobs[], double corr[],
     Integer tdc, NagError *fail)
```

3 Description

nag_ken_spe_corr_coeff (g02brc) calculates both the Kendall rank correlation coefficients and the Spearman rank correlation coefficients.

The data consists of n observations for each of m variables:

$$x_{ij}, \quad i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, m \; (m, n \geq 2)$$

where x_{ij} is the ith observation on the jth variable. The function eliminates any variable x_{ij} , for $i = 1, 2, \ldots, n$, where the argument **svar**$[j - 1] = 0$, and any observation x_{ij}, for $j = 1, 2, \ldots, m$, where the argument **sobs**$[i - 1] = 0$.

The observations are first ranked as follows:

For a given variable, j say, each of the observations x_{ij} for which **sobs**$[i - 1] > 0$, for $i = 1, 2, \ldots, n$, has associated with it an additional number, the rank of the observation, which indicates the magnitude of that observation relative to the magnitudes of the other observations on that same variable for which **sobs**$[i - 1] > 0$.

The smallest of these valid observations for variable j is assigned the rank 1, the second smallest observation for variable j the rank 2, and so on until the largest such observation is given the rank n_s, where n_s is the number of observations for which **sobs**$[i - 1] > 0$.

If a number of cases all have the same value for a given variable, j, then they are each given an ‘average’ rank — e.g., if in attempting to assign the rank $h + 1$, k observations for which **sobs**$[i - 1] > 0$ were found to have the same value, then instead of giving them the ranks $h + 1, h + 2, \ldots, h + k$ all k observations would be assigned the rank $\frac{2h+k+1}{2}$ and the next value in ascending order would be assigned the rank $h + k + 1$. The process is repeated for each of the m variables for which **svar**$[j - 1] > 0$.

Let y_{ij} be the rank assigned to the observation x_{ij} when the jth variable is being ranked. For those observations, i, for which **sobs**$[i - 1] = 0$, $y_{ij} = 0$, for $j = 1, 2, \ldots, m$.

For variables j, k the following are computed:

(a) Kendall's tau correlation coefficients:

$$R_{jk} = \frac{\sum_{h=1}^{n} \sum_{i=1}^{n} \operatorname{sign}(y_{hj} - y_{ij}) \operatorname{sign}(y_{hk} - y_{ik})}{\sqrt{[n_s(n_s - 1) - T_j][n_s(n_s - 1) - T_k]}} \quad j, k = 1, 2, \ldots, m;$$

where n_s is the number of observations for which **sobs**$[i-1] > 0$,
and sign $u = 1$ if $u > 0$,
sign $u = 0$ if $u = 0$,
sign $u = -1$ if $u < 0$,

and $T_j = \sum t_j(t_j - 1)$ where t_j is the number of ties of a particular value of variable j, and the summation is over all tied values of variable j.

(b) Spearman's rank correlation coefficients:

$$R_{jk} = \frac{n_s(n_s^2-1) - 6\sum_{i=1}^{n}(y_{ij}-y_{ik})^2 - \frac{1}{2}(T_j+T_k)}{\sqrt{[n_s(n_s^2-1)-T_j][n_s(n_s^2-1)-T_k]}} \quad j,k = 1,2,\ldots,m;$$

where n_s is the number of observations for which **sobs**$[i-1] > 0$, and $T_j = \sum t_j\left(t_j^2 - 1\right)$ where t_j is the number of ties of a particular value of variable j, and the summation is over all tied values of variable j.

4 References

Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw–Hill

5 Arguments

1: **n** – Integer *Input*

On entry: the number of observations in the dataset.

Constraint: **n** ≥ 2.

2: **m** – Integer *Input*

On entry: the number of variables.

Constraint: **m** ≥ 2.

3: **x**[**n** $\times$ **tdx**] – const double *Input*

On entry: **x**$[i-1 \times$ **tdx** $+ j - 1]$ must contain the ith observation on the jth variable, for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,m$.

4: **tdx** – Integer *Input*

On entry: the stride separating matrix column elements in the array **x**.

Constraint: **tdx** $\geq$ **m**.

5: **svar**[**m**] – const Integer *Input*

On entry: **svar**$[j-1]$ indicates which variables are to be included, for the jth variable to be included, **svar**$[j-1] > 0$. If all variables are to be included then a **NULL** pointer `(Integer *)0` may be supplied.

Constraint: **svar**$[j-1] \geq 0$, and there is at least one positive element, for $j = 1,2,\ldots,m$.

6: **sobs**[**n**] – const Integer *Input*

On entry: **sobs**$[i-1]$ indicates which observations are to be included, for the ith observation to be included, **sobs**$[i-1] > 0$. If all observations are to be included then a **NULL** pointer `(Integer *)0` may be supplied.

Constraint: **sobs**$[i-1] \geq 0$, and there are at least two positive elements, for $i = 1,2,\ldots,n$.

7: **corr**[**m** $\times$ **tdc**] – double *Output*

On exit: the upper n_s by n_s part of **corr** contains the correlation coefficients, the upper triangle contains the Spearman coefficients and the lower triangle, the Kendall coefficients. That is, for the jth and kth variables, where j is less than k, **corr**$[j - 1 \times$ **tdc** $+ k - 1]$ contains the Spearman rank correlation coefficient, and **corr**$[k - 1 \times$ **tdc** $+ j - 1]$ contains Kendall's tau, for $j, k = 1, 2, \ldots, n_s$. The diagonal will be set to 1.

8: **tdc** – Integer *Input*

On entry: the stride separating matrix column elements in the array **corr**.

Constraint: **tdc** $\geq$ **m**.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, **tdc** $= \langle$*value*$\rangle$ while **m** $= \langle$*value*$\rangle$. These arguments must satisfy **tdc** $\geq$ **m**.

On entry, **tdx** $= \langle$*value*$\rangle$ while **m** $= \langle$*value*$\rangle$. These arguments must satisfy **tdx** $\geq$ **m**.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_INT_ARG_LT

On entry, **m** $= \langle$*value*$\rangle$.
Constraint: **m** ≥ 2.

On entry, **n** $= \langle$*value*$\rangle$.
Constraint: **n** ≥ 2.

NE_INT_ARRAY_1

Value $\langle$*value*$\rangle$ given to **sobs**[$\langle$*value*$\rangle$] not valid. Correct range for elements of **sobs** is **sobs**$[i] \geq 0$.

Value $\langle$*value*$\rangle$ given to **svar**[$\langle$*value*$\rangle$] not valid. Correct range for elements of **svar** is **svar**$[i] \geq 0$.

NE_INTERNAL_ERROR

An initial error has occurred in this function. Check the function call and any array sizes.

NE_SOBS_LOW

On entry, **sobs** must contain at least 2 positive elements.

Too few observations have been selected.

NE_SVAR_LOW

No variables have been selected.

On entry, **svar** must contain at least 1 positive element.

7 Accuracy

The computations are believed to be stable.

8 Parallelism and Performance

nag_ken_spe_corr_coeff (g02brc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

A program to calculate the Kendall and Spearman rank correlation coefficients from a set of data.

10.1 Program Text

```
/* nag_ken_spe_corr_coeff (g02brc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg02.h>

#define X(I, J)    x[(I) *tdx + J]
#define CORR(I, J) corr[(I) *tdcorr + J]

int main(void)
{
  Integer exit_status = 0, i, j, m, n, *sobs = 0, *sobsptr, *svar = 0;
  Integer *svarptr;
  Integer tdcorr, tdx;
  NagError fail;
  char s, w;
  double *corr = 0, *x = 0;

  INIT_FAIL(fail);

  printf("nag_ken_spe_corr_coeff (g02brc) Example Program Results\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s(" %*[^\n]");
#else
  scanf(" %*[^\n]");
#endif

  /* Read data */
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%" NAG_IFMT "\n", &m, &n);
#else
  scanf("%" NAG_IFMT "%" NAG_IFMT "\n", &m, &n);
#endif
  if (m >= 2 && n >= 2) {
    if (!(x = NAG_ALLOC(n * m, double)) ||
        !(corr = NAG_ALLOC(m * m, double)) ||
        !(svar = NAG_ALLOC(m, Integer)) || !(sobs = NAG_ALLOC(n, Integer)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }
    tdx = m;
    tdcorr = m;
  }
  else {
    printf("Invalid m or n.\n");
    exit_status = 1;
    return exit_status;
  }
  for (i = 0; i < n; i++)
    for (j = 0; j < m; j++)
#ifdef _WIN32
      scanf_s("%lf", &X(i, j));
#else
      scanf("%lf", &X(i, j));
#endif

  /* Read flag specifying if svar is to be supplied */
#ifdef _WIN32
  scanf_s(" %c", &s, 1);
#else
  scanf(" %c", &s);
#endif
  if (s == 'S' || s == 's') {
    /* Assign pointer to svar and read in values for svar */
    svarptr = svar;
    for (i = 0; i < m; i++)
#ifdef _WIN32
      scanf_s("%" NAG_IFMT "", &svar[i]);
#else
      scanf("%" NAG_IFMT "", &svar[i]);
#endif
  }
  else {
    /* Assign pointer to NULL and discard rest of line */
    svarptr = (Integer *) 0;
    /* skip rest of line */
#ifdef _WIN32
    scanf_s(" %*[^\n]");
#else
    scanf(" %*[^\n]");
#endif
  }

  /* Read flag specifying if sobs is to be supllied */
#ifdef _WIN32
  scanf_s(" %c", &w, 1);
#else
  scanf(" %c", &w);
#endif
  if (w == 'W' || w == 'w') {
    /* Assign pointer to sobs and read in values for sobs */
    sobsptr = sobs;
    for (i = 0; i < n; i++)
#ifdef _WIN32
      scanf_s("%" NAG_IFMT "", &sobs[i]);
#else
      scanf("%" NAG_IFMT "", &sobs[i]);
#endif
  }
  else {
    /* Assign pointer to NULL and discard rest of line */
    sobsptr = (Integer *) 0;
    /* skip rest of line */
#ifdef _WIN32
    scanf_s(" %*[^\n]");
#else
    scanf(" %*[^\n]");
#endif
  }

  /* Calculate the Kendall and Spearman coefficients */
  /* nag_ken_spe_corr_coeff (g02brc).
```

```
   * Kendall and/or Spearman non-parametric rank correlation
   * coefficients, allows variables and observations to be
   * selectively disregarded
   */
  nag_ken_spe_corr_coeff(n, m, x, tdx, svarptr, sobsptr, corr, tdcorr, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_ken_spe_corr_coeff (g02brc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  printf("\nCorrelation coefficients:\n\n");
  for (i = 0; i < m; i++) {
    for (j = 0; j < m; j++)
      printf("%8.5f ", CORR(i, j));
    printf("\n");
  }
END:
  NAG_FREE(x);
  NAG_FREE(corr);
  NAG_FREE(svar);
  NAG_FREE(sobs);
  return exit_status;
}
```

10.2 Program Data

```
nag_ken_spe_corr_coeff (g02brc) Example Program Data
3 7
1.0 2.0 4.0
7.0 7.0 3.0
2.0 3.0 4.0
4.0 4.0 5.0
5.0 6.0 7.0
3.0 1.0 3.0
6.0 5.0 5.0
s 1 1 1
w 1 1 1 1 1 1 1
```

10.3 Program Results

```
nag_ken_spe_corr_coeff (g02brc) Example Program Results

Correlation coefficients:

 1.00000  0.85714  0.12849
 0.71429  1.00000  0.33040
 0.10287  0.41148  1.00000
```