NAG C Library Function Document

nag_tabulate_margin (g11bcc)

1 Purpose

nag_tabulate_margin (g11bcc) computes a marginal table from a table computed by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc) using a selected statistic.

2 Specification

```
void nag_tabulate_margin (Nag_TableStats stat, const double table[], Integer ncells,
      Integer ndim, const Integer idim[], const Integer isdim[], double sub_table[],
      Integer maxst, Integer *mcells, Integer *mdim, Integer mlevel[],
      double comm_ar[], NagError *fail)
```

3 Description

For a data set containing classification variables (known as factors) the routines nag_tabulate_stats (g11bac) and nag_tabulate_percentile (g11bbc) compute a table using selected statistics, for example the mean or the median. The table is indexed by the levels of the selected factors, for example if there were three factors A, B and C with 3, 2 and 4 levels respectively and the mean was to be tabulated the resulting table would be $3 \times 2 \times 4$ with each cell being the mean of all observations with the appropriate combination of levels of the three factors. In further analysis the table of means averaged over C for A and B may be required; this can be computed from the full table by taking the mean over the third dimension of the table, C.

In general, given a table computed by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc), nag_tabulate_margin (g11bcc) computes a sub-table defined by a subset of the factors used to define the table such that each cell of the sub-table is the selected statistic computed over the remaining factors. The statistics that can be used are the total, the mean, the median, the variance, the smallest and the largest value.

4 References

John J A and Quenouille M H (1977) *Experiments: Design and Analysis* Griffin

Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* (3rd Edition) Griffin

West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–555

5 Parameters

1: **stat** – Nag_TableStats *Input*

On entry: indicates which statistic is to be used to compute the marginal table.

If **stat** = **Nag_TableStatsNObs** the total.

If **stat** = **Nag_TableStatsAv** the average or mean.

If **stat** = **Nag_TableStatsMedian** the median.

If **stat** = **Nag_TableStatsVar** the variance.

If **stat** = **Nag_TableStatsLarge** the largest value.

If **stat** = **Nag_TableStatsSmall** the smallest value.

Constraint: **stat** = **Nag_TableStatsNObs**, **Nag_TableStatsAv**, **Nag_TableStatsMedian**, **Nag_TableStatsVar**, **Nag_TableStatsLarge** or **Nag_TableStatsSmall**.

2: **table**[**ncells**] – const double *Input*

On entry: the table as computed by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc).

3: **ncells** – Integer *Input*

On entry: the number of cells in **table** as returned by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc).

4: **ndim** – Integer *Input*

On entry: the number of dimensions for **table** as returned by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc).

Constraint: **ndim** ≥ 2.

5: **idim**[**ndim**] – const Integer *Input*

On entry: the number of levels for each dimension of **table** as returned by nag_tabulate_stats (g11bac) or nag_tabulate_percentile (g11bbc).

Constraint: **idim**$[i] \geq 2$ for $i = 0, 1, \ldots,$ **ndim** $- 1$.

6: **isdim**[**ndim**] – const Integer *Input*

On entry: indicates which dimensions of **table** are to be included in the sub-table. If **isdim**$[i-1] > 0$ the dimension or factor indicated by **idim**$[i-1]$ is to be included in the sub-table, otherwise it is excluded.

7: **sub_table**[**maxst**] – double *Output*

On exit: the first **mcells** elements contain the sub-table computed using the statistic indicated by **stat**. The table is stored in a similar way to **table** with the **mcells** cells stored so that for any two dimensions the index relating to the dimension given later in **idim** changes faster. For further details see Section 8.

8: **maxst** – Integer *Input*

On entry: the maximum size of sub-table to be computed.

Constraint: **maxst** $\geq$ the product of the levels of the dimensions of **table** included in the sub-table, **sub_table**.

9: **mcells** – Integer * *Output*

On exit: the number of cells in the sub-table in **sub_table**.

10: **mdim** – Integer * *Output*

On exit: the number of dimensions to the sub-table in **sub_table**.

11: **mlevel**[**ndim**] – Integer *Output*

On exit: the first **mdim** elements contain the number of levels for the dimensions of the sub-table in **sub_table**. The remaining elements are not referenced.

12: **comm_ar**[dim] – double *Output*

Note: the dimension, dim, of the array **comm_ar** must be at least **maxst** when **stat** = **Nag_TableStatsVar** and at least 1 otherwise.

On exit: if **stat** = **Nag_TableStatsVarcomm_ar** contains the sub-table of means corresponding to the sub-table of variances in **sub_table**. Otherwise **comm_ar** is not referenced.

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

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, element $\langle value \rangle$ of **idim** ≤ 1.

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

NE_INT_2

On entry, **ncells** is incompatible with **idim**.

On entry, **maxst** $(= \langle value \rangle)$ is too small, min value = $\langle value \rangle$.

NE_INT_ARRAY_ELEM_CONS

On entry, all elements of **isdim** > 0.

On entry, no elements of **isdim** > 0.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter $\langle value \rangle$ 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

Only applicable when **stat** = **Nag_TableStatsVar**. In this case a one pass algorithm is used as describe in West (1979).

8 Further Comments

The sub-tables created by nag_tabulate_margin (g11bcc) and stored in **sub_table** and, depending on **stat**, also in **comm_ar** are stored in the following way. Let there be m dimensions defining the table with dimension k having l_k levels, then the cell defined by the levels $i_1, i_2, \ldots, i_m$ of the factors is stored in sth cell given by

$$s = 1 + \sum_{k=1}^{m} [(i_k - 1)c_k],$$

where

$$c_j = \prod_{k=j+1}^{m} l_k \quad \text{for } j = 1, 2, \ldots, n-1 \quad \text{and} \quad c_m = 1.$$

9 Example

The data, given by John and Quenouille (1977), is for 3 blocks of a 3×6 factorial experiment. The data can be considered as a $3 \times 6 \times 3$ table (i.e., blocks $\times$ treatment with 6 levels $\times$ treatment with 3 levels). This table is input and the 6×3 table of treatment means for over blocks is computed and printed.

9.1 Program Text

```
/* nag_tabulate_margin (g11bcc) 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, j, k, maxst, mcells, mdim,
          ncells, ncol, ndim, nrow;
  NagError fail;
  Nag_TableStats stat_enum;
  char stat;

  /* Arrays */
  double *auxt = 0, *stable = 0, *table = 0;
  Integer *idim = 0, *isdim = 0, *mlevel = 0;

  INIT_FAIL(fail);
  exit_status = 0;
  Vprintf("g11bcc Example Program Results\n");

  /* Skip heading in data file */
  Vscanf("%*[^\n] ");

  Vscanf(" '%c'%ld%ld%*[^\n] ", &stat, &ncells, &ndim);
  maxst = 54;

  /* Allocate arrays */
  if ( !(auxt = NAG_ALLOC(maxst, double)) ||
       !(stable = NAG_ALLOC(maxst, double)) ||
       !(table = NAG_ALLOC(ncells, double)) ||
       !(idim = NAG_ALLOC(ndim, Integer)) ||
       !(isdim = NAG_ALLOC(ndim, Integer)) ||
       !(mlevel = NAG_ALLOC(ndim, Integer)) )
    {
      Vprintf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  for (i = 1; i <= ncells; ++i)
    Vscanf("%lf", &table[i-1]);
  Vscanf("%*[^\n] ");

  for (j = 1; j <= ndim; ++j)
    Vscanf("%ld", &idim[j-1]);
  Vscanf("%*[^\n] ");

  for (j = 1; j <= ndim; ++j)
    Vscanf("%ld", &isdim[j-1]);
  Vscanf("%*[^\n] ");

  switch (stat)
    {
    case 'T':
      stat_enum = Nag_TableStatsNObs;
      break;
    case 'A':
      stat_enum = Nag_TableStatsAv;
      break;
    case 'M':
      stat_enum = Nag_TableStatsMedian;
      break;
    case 'V':
      stat_enum = Nag_TableStatsVar;
      break;
    case 'L':
      stat_enum = Nag_TableStatsLarge;
      break;
    case 'S':
      stat_enum = Nag_TableStatsSmall;
      break;
    default:
      stat_enum = Nag_TableStatsNObs;
    }

  g11bcc(stat_enum, table, ncells, ndim, idim, isdim, stable, maxst,
         &mcells, &mdim, mlevel, auxt, &fail);
  if (fail.code != NE_NOERROR)
    {
      Vprintf("Error from g11bcc.\n%s\n",   fail.message);
      exit_status = 1;
      goto END;
    }

  Vprintf("\n");
  Vprintf(" Marginal Table\n");
  Vprintf("\n");

  ncol = mlevel[mdim-1];
  nrow = mcells / ncol;
  k = 1;
  for (i = 1; i <= nrow; ++i)
    {
      for (j = k; j <= k + ncol - 1; ++j)
        Vprintf("%7.2f ", stable[j-1]);
      Vprintf("\n");
      k += ncol;
    }

 END:
  if (auxt) NAG_FREE(auxt);
  if (stable) NAG_FREE(stable);
  if (table) NAG_FREE(table);
  if (idim) NAG_FREE(idim);
  if (isdim) NAG_FREE(isdim);
  if (mlevel) NAG_FREE(mlevel);

  return exit_status;
}
```

9.2 Program Data

```
g11bcc Example Program Data

'A' 54 3

274 361 253 325 317 339 326 402 336 379 345 361 352 334 318 339 393 358
350 340 203 397 356 298 382 376 355 418 387 379 432 339 293 322 417 342
 82 297 133 306 352 361 220 333 270 388 379 274 336 307 266 389 333 353

3 6 3
0 1 1
```

9.3 Program Results

```
g11bcc Example Program Results

 Marginal Table

 235.33  332.67  196.33
 342.67  341.67  332.67
 309.33  370.33  320.33
 395.00  370.33  338.00
 373.33  326.67  292.33
 350.00  381.00  351.00
```