NAG Library Function Document

nag_anova_row_col (g04bcc)

1 Purpose

nag_anova_row_col (g04bcc) computes the analysis of variance for a general row and column design together with the treatment means and standard errors.

2 Specification

```
#include <nag.h>
#include <nagg04.h>
```

```
void nag_anova_row_col (Integer nrep, Integer nrow, Integer ncol,
     const double y[], Integer nt, const Integer it[], double *gmean,
     double tmean[], double table[], double c[], Integer tdc, Integer irep[],
     double rpmean[], double rmean[], double cmean[], double r[],
     double ef[], double tol, Integer irdf, NagError *fail)
```

3 Description

In a row and column design the experimental material can be characterised by a two-way classification, nominally called rows and columns. Each experimental unit can be considered as being located in a particular row and column. It is assumed that all rows are of the same length and all columns are of the same length. Sets of equal numbers of rows and columns can be grouped together to form replicates, sometimes known as squares or rectangles, as appropriate.

If for a replicate, the number of rows, the number of columns and the number of treatments are equal and every treatment occurs once in each row and each column then the design is a Latin square. If this is not the case the treatments will be non-orthogonal to rows and columns. For example in the case of a lattice square each treatment occurs only once in each square.

For a row and column design, with t treatments in r rows and c columns and b replicates or squares with $n = brc$ observations, the linear model is:

$$y_{ijk(l)} = \mu + \beta_i + \rho_j + \gamma_k + \tau_l + e_{ijk}$$

$i = 1, 2, \ldots, b$; $j = 1, 2, \ldots, r$;$k = 1, 2, \ldots, c$; $l = 1, 2, \ldots, t$, where β_i is the effect of the ith replicate, ρ_j is the effect of the jth row, γ_k is the effect of the kth column and the $ijk(l)$ notation indicates that the lth treatment is applied to the unit in row j, column k of replicate i.

To compute the analysis of variance for a row and column design the mean is computed and subtracted from the observations to give, $y'_{ijk(l)} = y_{ijk(l)} - \hat{\mu}$. Since the replicates, rows and columns are orthogonal the estimated effects, ignoring treatment effects, $\hat{\beta}_i$, $\hat{\rho}_j$, $\hat{\gamma}_k$, can be computed using the appropriate means of the $y'_{ijk(l)}$, and the unadjusted sum of squares computed as the appropriate sum of squared totals for the $y'_{ijk(l)}$ divided by number of units per total. The observations adjusted for replicates, rows and columns can then be computed by subtracting the estimated effects from $y'_{ijk(l)}$ to give $y''_{ijk(l)}$.

In the case of a Latin square design the treatments are orthogonal to replicates, rows and columns and so the treatment effects, $\hat{\tau}_l$, can be estimated as the treatment means of the adjusted observations, $y''_{ijk(l)}$. The treatment sum of squares is computed as the sum of squared treatment totals of the $y''_{ij(l)}$ divided by the number of times each treatment is replicated. Finally the residuals, and hence the residual sum of squares, are given by, $r_{ij(l)} = y''_{ij(l)} - \hat{\tau}_l$.

For a design which is not orthogonal, for example a lattice square or an incomplete Latin square, the treatment effects adjusted for replicates, rows and columns need to be computed. The adjusted treatment effects are found as the solution to the equations:

$$A\hat{\tau} = \left(R - N_b N_b^{\mathrm{T}}/(rc) - N_r N_r^{\mathrm{T}}/(bc) - N_c N_c^{\mathrm{T}}/(br)\right)\hat{\tau} = q$$

where q is the vector of the treatment totals of the observations adjusted for replicates, rows and columns, $y''_{ijk(l)}$; R is a diagonal matrix with R_{ll} equal to the number of times the lth treatment is replicated, and N_b is the t by b incidence matrix, with $N_{l,i}$ equal to the number of times treatment l occurs in replicate i, with N_r and N_c being similarly defined for rows and columns. The solution to the equations can be written as:

$$\hat{\tau} = \Omega q$$

where, Ω is a generalized inverse of A. The solution is found from the eigenvalue decomposition of A. The residuals are first calculated by subtracting the estimated adjusted treatment effects from the adjusted observations to give $r'_{ij(l)} = y''_{ij(l)} - \hat{\tau}_l$. However, since only the unadjusted replicate, row and column effects have been removed and they are not orthogonal to treatments, the replicate, row and column means of the $r'_{ij(l)}$ have to be subtracted to give the correct residuals, $r_{ij(l)}$ and residual sum of squares.

Given the sums of squares, the mean squares are computed as the sums of squares divided by the degrees of freedom. The degrees of freedom for the unadjusted replicates, rows and columns are $b-1$, $r-1$ and $c-1$ respectively and for the Latin square designs the degrees of freedom for the treatments is $t-1$. In the general case the degrees of freedom for treatments is the rank of the matrix Ω. The F-statistic given by the ratio of the treatment mean square to the residual mean square tests the hypothesis:

$$H_0 : \tau_1 = \tau_2 = \cdots = \tau_t = 0.$$

The standard errors for the difference in treatment effects, or treatment means, for Latin square designs, are given by:

$$se\left(\hat{\tau}_j - \hat{\tau}_{j*}\right) = \sqrt{2s^2/(bt)}$$

where s^2 is the residual mean square. In the general case the variances of the treatment effects are given by:

$$\mathrm{Var}(\hat{\tau}) = \Omega s^2$$

from which the appropriate standard errors of the difference between treatment effects or the difference between adjusted means can be calculated.

The analysis of a row-column design can be considered as consisting of different strata: the replicate stratum, the rows within replicate and the columns within replicate strata and the units stratum. In the Latin square design all the information on the treatment effects is given at the units stratum. In other designs there may be a loss of information due to the non-orthogonality of treatments and replicates, rows and columns and information on treatments may be available in higher strata. The efficiency of the estimation at the units stratum is given by the (canonical) efficiency factors, these are the nonzero eigenvalues of the matrix, A, divided by the number of replicates in the case of equal replication, or by the mean of the number of replicates in the unequally replicated case, (see John (1987)). If more than one eigenvalue is zero then the design is said to be disconnected and information on some treatment comparisons can only be obtained from higher strata.

4 References

Cochran W G and Cox G M (1957) *Experimental Designs* Wiley

Davis O L (1978) *The Design and Analysis of Industrial Experiments* Longman

John J A (1987) *Cyclic Designs* Chapman and Hall

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

Searle S R (1971) *Linear Models* Wiley

5 Arguments

1: **nrep** – Integer *Input*

On entry: the number of replicates, b.

Constraint: **nrep** ≥ 1.

2: **nrow** – Integer *Input*

On entry: the number of rows per replicate, r.

Constraint: **nrow** ≥ 2.

3: **ncol** – Integer *Input*

On entry: the number of columns per replicate, c.

Constraint: **ncol** ≥ 2.

4: **y**[**nrep** × **nrow** × **ncol**] – const double *Input*

On entry: the $n = brc$ observations ordered by columns within rows within replicates. That is **y**$[rc(i-1) + r(j-1) + k - 1]$ contains the observation from the k column of the jth row of the ith replicate, $i = 1, 2, \ldots, b$; $j = 1, 2, \ldots, r$ and $k = 1, 2, \ldots, c$.

5: **nt** – Integer *Input*

On entry: the number of treatments. If only replicates, rows and columns are required in the analysis then set **nt** $= 1$.

Constraint: **nt** ≥ 1.

6: **it**[×] – const Integer *Input*

On entry: if **nt** > 1, **it**$[i-1]$ indicates which of the **nt** treatments unit i received, $i = 1, 2, \ldots, n$. If **nt** $= 1$, **it** is not referenced.

Constraint: if **nt** ≥ 2, $1 \leq$ **it**$[i-1] \leq$ **nt**, for $i = 1, 2, \ldots, n$.

7: **gmean** – double * *Output*

On exit: the grand mean, $\hat{\mu}$.

8: **tmean**[**nt**] – double *Output*

On exit: if **nt** ≥ 2, **tmean**$[l-1]$ contains the (adjusted) mean for the lth treatment, $\hat{\mu}^* + \hat{\tau}_l$, $l = 1, 2, \ldots, t$, where $\hat{\mu}^*$ is the mean of the treatment adjusted observations $y_{ijk(l)} - \hat{\tau}_l$. Otherwise **tmean** is not referenced.

9: **table**[**6** × **5**] – double *Output*

Note: the (i, j)th element of the matrix is stored in **table**$[(i-1) \times 5 + j - 1]$.

On exit: the analysis of variance table. Column 1 contains the degrees of freedom, column 2 the sum of squares, and where appropriate, column 3 the mean squares, column 4 the F-statistic and column 5 the significance level of the F-statistic. Row 1 is for replicates, row 2 for rows, row 3 for columns, row 4 for treatments (if **nt** > 1), row 5 for residual and row 6 for total. Mean squares are computed for all but the total row, F-statistics and significance are computed for treatments, replicates, rows and columns. Any unfilled cells are set to zero.

10: **c**[**nt** × **tdc**] – double *Output*

On exit: the upper triangular part of **c** contains the variance-covariance matrix of the treatment effects, the strictly lower triangular part contains the standard errors of the difference between two treatment effects (means), i.e., **c**$[(i-1) \times$ **tdc** $+ j - 1]$ contains the covariance of treatment i

and j if $j \geq i$ and the standard error of the difference between treatment i and j if $j < i$, $i = 1, 2, \ldots, t$ and $j = 1, 2, \ldots, t$.

11: **tdc** – Integer *Input*

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

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

12: **irep**[**nt**] – Integer *Output*

On exit: if **nt** > 1, **irep**[$l - 1$] contains the treatment replications, R_{ll}, $l = 1, 2, \ldots,$ **nt**. Otherwise **irep** is not referenced.

13: **rpmean**[**nrep**] – double *Output*

On exit: if **nrep** > 1, **rpmean**[$i - 1$] contains the mean for the ith replicate, $\hat{\mu} + \hat{\beta}_i$, $i = 1, 2, \ldots, b$. Otherwise **rpmean** is not referenced.

14: **rmean**[**nrep** $\times$ **nrow**] – double *Output*

On exit: **rmean**[$j - 1$] contains the mean for the jth row, $\hat{\mu} + \hat{\rho}_i$, $j = 1, 2, \ldots, r$.

15: **cmean**[**nrep** $\times$ **ncol**] – double *Output*

On exit: **cmean**[$k - 1$] contains the mean for the kth column, $\hat{\mu} + \hat{\gamma}_k$, $k = 1, 2, \ldots, c$.

16: **r**[**nrep** $\times$ **nrow** $\times$ **ncol**] – double *Output*

On exit: **r**[$i - 1$] contains the residuals, r_i, $i = 1, 2, \ldots, n$.

17: **ef**[**nt**] – double *Output*

On exit: if **nt** ≥ 2, the canonical efficiency factors. Otherwise **ef** is not referenced.

18: **tol** – double *Input*

On entry: the tolerance value used to check for zero eigenvalues of the matrix Ω. If **tol** $= 0.0$ a default value of 0.00001 is used.

Constraint: **tol** ≥ 0.0.

19: **irdf** – Integer *Input*

On entry: an adjustment to the degrees of freedom for the residual and total.

irdf ≥ 1
The degrees of freedom for the total is set to $n -$ **irdf** and the residual degrees of freedom adjusted accordingly.

irdf $= 0$
The total degrees of freedom for the total is set to $n - 1$, as usual.

Constraint: **irdf** ≥ 0.

20: **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 **nt** $= \langle$*value*$\rangle$. These arguments must satisfy **tdc** $\geq$ **nt**.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_ARRAY_CONS

The contents of array **it** are not valid.
Constraint: if **nt** ≥ 2, $1 \leq$ **it**$[i] \leq$ **nt**, for $i = 0, 1, 2, \dots,$ **nrep** $\times$ **nrow** $\times$ **ncol**.
The contents of array **it** are not valid.
Constraint: some value of **it** $= j$ for all $j = 1, 2, \dots,$**nt**.

NE_ARRAY_CONSTANT

On entry, the elements of the array **y** are constant.

NE_G04BC_DISCON

The design is disconnected, the standard errors may not be valid. The design may have a nested structure.

NE_G04BC_REPS

The treatments are totally confounded with replicates, rows and columns, so the treatment sum of squares and degrees of freedom are zero. The analysis of variance table is not computed, except for replicate, row, column, total sum of squares and degrees of freedom.

NE_G04BC_RESD

The residual degrees of freedom or the residual sum of squares are zero, columns 3, 4 and 5 of the analysis of variance table will not be computed and the matrix of standard errors and covariances, **c**, will not be scaled.

NE_G04BC_ST_ERR

A computed standard error is zero due to rounding errors, or the eigenvalue computation failed to converge. Both are unlikely errors.

NE_INT_ARG_LT

On entry, **irdf** $= \langle value \rangle$.
Constraint: **irdf** ≥ 0.

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

On entry, **nrep** $= \langle value \rangle$.
Constraint: **nrep** ≥ 1.

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

On entry, **nt** $= \langle value \rangle$.
Constraint: **nt** ≥ 1.

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 contact NAG for assistance.

NE_REAL_ARG_LT

On entry, **tol** must not be less than 0.0: **tol** $= \langle value \rangle$.

7 Accuracy

The algorithm used in nag_anova_row_col (g04bcc), described in Section 3, achieves greater accuracy than the traditional algorithms based on the subtraction of sums of squares.

8 Parallelism and Performance

nag_anova_row_col (g04bcc) is not threaded in any implementation.

9 Further Comments

To estimate missing values the Healy and Westmacott procedure or its derivatives may be used (see John and Quenouille (1977)). This is an iterative procedure in which estimates of the missing values are adjusted by subtracting the corresponding values of the residuals. The new estimates are then used in the analysis of variance. This process is repeated until convergence. A suitable initial value may be the grand mean. When using this procedure **irdf** should be set to the number of missing values plus one to obtain the correct degrees of freedom for the residual sum of squares.

For analysis of covariance the residuals are obtained from an analysis of variance of both the response variable and the covariates. The residuals from the response variable are then regressed on the residuals from the covariates using, say, nag_regress_confid_interval (g02cbc) or nag_regsn_mult_linear (g02dac). The results from those functions can be used to test for the significance of the covariates. To test the significance of the treatment effects after fitting the covariate, the residual sum of squares from the regression should be compared with the residual sum of squares obtained from the equivalent regression but using the residuals from fitting replicates, rows and columns only.

10 Example

The data for a 5 × 5 Latin square is input and the ANOVA and treatment means computed and printed. Since the design is orthogonal only one standard error need be printed

10.1 Program Text

```
/* nag_anova_row_col (g04bcc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg04.h>

int main(void)
{
  Integer c__0 = 0, exit_status = 0, i, *irep = 0, *it = 0, j, n, ncol,
         nrep, nrow, nt;
  NagError fail;
  const char *fmt_99999[] = { "%3.0f  ", "%10.4f  ", "%10.4f  ", "%10.4f  ",
    "%8.4f"
  };
  double *c = 0, c_b20 = 1e-5, *cmean = 0, *ef = 0, gmean, *r = 0;
  double *rmean = 0, *rpmean = 0, *table = 0, *tmean = 0, *y = 0;

#define TABLE(I, J) table[((I) -1)*5 + (J) -1]
#define C(I, J)     c[((I) -1)*nt + (J) -1]

  INIT_FAIL(fail);

  printf("nag_anova_row_col (g04bcc) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

#ifdef _WIN32
  scanf_s("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "", &nrep,
          &nrow, &ncol, &nt);
#else
  scanf("%" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "", &nrep,
        &nrow, &ncol, &nt);
#endif
  if (!(c = NAG_ALLOC(nt * nt, double))
      || !(cmean = NAG_ALLOC(nrep * ncol, double))
      || !(ef = NAG_ALLOC(nt, double))
      || !(r = NAG_ALLOC(nrep * nrow * ncol, double))
      || !(y = NAG_ALLOC(nrep * nrow * ncol, double))
      || !(rmean = NAG_ALLOC(nrep * nrow, double))
      || !(rpmean = NAG_ALLOC(nrep, double))
      || !(tmean = NAG_ALLOC(nt, double))
      || !(table = NAG_ALLOC(30, double))
      || !(irep = NAG_ALLOC(nt, Integer))
      || !(it = NAG_ALLOC(nrep * nrow * ncol, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  n = nrep * nrow * ncol;
  for (i = 1; i <= n; ++i)
#ifdef _WIN32
    scanf_s("%lf", &y[i - 1]);
#else
    scanf("%lf", &y[i - 1]);
#endif
  for (i = 1; i <= n; ++i)
#ifdef _WIN32
    scanf_s("%" NAG_IFMT "", &it[i - 1]);
#else
    scanf("%" NAG_IFMT "", &it[i - 1]);
#endif
  /* nag_anova_row_col (g04bcc).
   * Analysis of variance, general row and column design,
   * treatment means and standard errors
   */
  nag_anova_row_col(nrep, nrow, ncol, y, nt, it, &gmean, tmean, table,
                    c, nt, irep, rpmean, rmean, cmean, r, ef, c_b20, c__0,
                    &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_anova_row_col (g04bcc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }
  printf("\n ANOVA TABLE\n\n");
  if (nrep > 1) {
    printf("\n Reps        ");
    for (j = 1; j <= 5; ++j)
      printf(fmt_99999[j - 1], TABLE(1, j));
  }

  printf("\n Rows        ");
  for (j = 1; j <= 5; ++j)
    printf(fmt_99999[j - 1], TABLE(2, j));

  printf("\n Columns     ");
  for (j = 1; j <= 5; ++j)
    printf(fmt_99999[j - 1], TABLE(3, j));
  printf("\n\n Treatments  ");
  for (j = 1; j <= 5; ++j)
    printf(fmt_99999[j - 1], TABLE(4, j));

  printf("\n Residual    ");
  for (j = 1; j <= 3; ++j)
    printf(fmt_99999[j - 1], TABLE(5, j));

  printf("\n Total       ");
  for (j = 1; j <= 2; ++j)
    printf(fmt_99999[j - 1], TABLE(6, j));

  printf("\n Treatment means\n\n");
  for (i = 1; i <= nt; ++i)
    printf("%10.4f%s", tmean[i - 1], i % 6 ? "" : "\n");
  printf("\n\n S.E. of difference (orthogonal design) = %10.4f\n", C(2, 1));
END:
  NAG_FREE(c);
  NAG_FREE(cmean);
  NAG_FREE(ef);
  NAG_FREE(r);
  NAG_FREE(y);
  NAG_FREE(rmean);
  NAG_FREE(rpmean);
  NAG_FREE(tmean);
  NAG_FREE(table);
  NAG_FREE(irep);
  NAG_FREE(it);
  return exit_status;
}
```

10.2 Program Data

```
nag_anova_row_col (g04bcc) Example Program Data

1 5 5 5

6.67  7.15  8.29  8.95  9.62
5.40  4.77  5.40  7.54  6.93
7.32  8.53  8.50  9.99  9.68
4.92  5.00  7.29  7.85  7.08
4.88  6.16  7.83  5.38  8.51

5  4  1  3  2
2  5  4  1  3
3  2  5  4  1
1  3  2  5  4
4  1  3  2  5
```

10.3 Program Results

```
nag_anova_row_col (g04bcc) Example Program Results

 ANOVA TABLE


 Rows          4     29.4231      7.3558      9.0266    0.0013
 Columns       4     22.9950      5.7487      7.0545    0.0037

 Treatments    4      0.5423      0.1356      0.1664    0.9514
 Residual     12      9.7788      0.8149
 Total        24     62.7392
 Treatment means

    7.3180    7.2440    7.2060    6.9000    7.2600

 S.E. of difference (orthogonal design) =     0.5709
```