NAG Library Function Document

nag_anova_factorial (g04cac)

1 Purpose

nag_anova_factorial (g04cac) computes an analysis of variance table and treatment means for a complete factorial design.

2 Specification

```
#include <nag.h>
#include <nagg04.h>
void nag_anova_factorial (Integer n, const double y[], Integer nfac,
     const Integer lfac[], Integer nblock, Integer inter, Integer irdf,
     Integer *mterm, double **table, double **tmean, Integer *maxt,
     double **e, Integer **imean, double **semean, double **bmean,
     double r[], NagError *fail)
```

3 Description

An experiment consists of a collection of units, or plots, to which a number of treatments are applied. In a factorial experiment the effects of several different sets of conditions are compared, e.g., three different temperatures, T_1, T_2 and T_3, and two different pressures, P_1 and P_2. The conditions are known as factors and the different values the conditions take are known as levels. In a factorial experiment the experimental treatments are the combinations of all the different levels of all factors, e.g.,

$$T_1P_1T_2P_1T_3P_1,$$

$$T_1P_2T_2P_2T_3P_2$$

The effect of a factor averaged over all other factors is known as a main effect, and the effect of a combination of some of the factors averaged over all other factors is known as an interaction. This can be represented by a linear model. In the above example if the response was y_{ijk} for the kth replicate of the ith level of T and the jth level of P the linear model would be

$$y_{ijk} = \mu + t_i + p_j + \gamma_{ij} + e_{ijk}$$

where μ is the overall mean, t_i is the main effect of T, p_j is the main effect of P, γ_{ij} is the $T \times P$ interaction and e_{ijk} is the random error term. In order to find unique estimates constraints are placed on the parameter estimates. For the example here these are:

$$\sum_{i=1}^{3}\hat{t}_i = 0, \quad \sum_{j=1}^{2}\hat{p}_j = 0, \quad \sum_{i=1}^{3}\hat{\gamma}_{ij} = 0 \quad \text{for } j = 1,2 \quad \text{and} \quad \sum_{j=1}^{2}\hat{\gamma}_{ij} = 0 \quad \text{for } i = 1,2,3,$$

where $\hat{}$ denotes the estimate.

If there is variation in the experimental conditions, (e.g., in an experiment on the production of a material, different batches of raw material, may be used, or the experiment may be carried out on different days) then plots that are similar are grouped together into blocks. For a balanced complete factorial experiment all the treatment combinations occur the same number of times in each block.

nag_anova_factorial (g04cac) computes the analysis of variance (ANOVA) table by sequentially computing the totals and means for an effect from the residuals computed when previous effects have been removed. The effect sum of squares is the sum of squared totals divided by the number of observations per total. The means are then subtracted from the residuals to compute a new set of residuals. At the same time the means for the original data are computed. When all effects are removed

the residual sum of squares is computed from the residuals. Given the sums of squares an ANOVA table is then computed along with standard errors for the difference in treatment means.

The data for nag_anova_factorial (g04cac) has to be in standard order given by the order of the factors. Let there be k factors, $f_1, f_2, \ldots, f_k$ in that order with levels $l_1, l_2, \ldots, l_k$ respectively. Standard order requires the levels of factor f_1 are in order $1, 2, \ldots, l_1$ and within each level of f_1 the levels of f_2 are in order $1, 2, \ldots, l_2$ and so on.

For an experiment with blocks the data is for block 1 then for block 2, etc. Within each block the data must be arranged so that the levels of factor f_1 are in order $1, 2, \ldots, l_1$ and within each level of f_1 the levels of f_2 are in order $1, 2, \ldots, l_2$ and so on. Any within block replication of treatment combinations must occur within the levels of f_k.

The ANOVA table is given in the following order. For a complete factorial experiment the first row is for blocks, if present, then the main effects of the factors in their order, e.g., f_1 followed by f_2 etc. These are then followed by all the two factor interactions then all the three factor interactions etc. The last two rows being for the residual and total sums of squares. The interactions are arranged in lexical order for the given factor order. For example, for the three factor interactions for a five factor experiment the 10 interactions would be in the following order:

$$f_1 f_2 f_3,$$

$$f_1 f_2 f_4,$$

$$f_1 f_2 f_5,$$

$$f_1 f_3 f_4,$$

$$f_1 f_3 f_5,$$

$$f_1 f_4 f_5,$$

$$f_2 f_3 f_4,$$

$$f_2 f_3 f_5,$$

$$f_2 f_4 f_5,$$

$$f_3 f_4 f_5$$

4 References

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

5 Arguments

1: **n** – Integer *Input*

On entry: the number of observations.

Constraints:

n ≥ 4.

n must be a multiple of **nblock** if **nblock** > 1.

n must be a multiple of the number of treatment combinations, that is a multiple of $\prod_{i=1}^{k}$**lfac**$[i-1]$.

2: **y**[**n**] – const double *Input*

On entry: the number of observations in standard order, see Section 3.

3: **nfac** – Integer *Input*

On entry: the number of factors, k.

Constraint: **nfac** ≥ 1.

4: **lfac**[**nfac**] – const Integer *Input*

On entry: **lfac**$[i-1]$ must contain the number of levels for the ith factor, for $i = 1, 2, \ldots, k$.

Constraint: **lfac**$[i-1] \geq 2$, for $i = 1, 2, \ldots, k$.

5: **nblock** – Integer *Input*

On entry: the number of blocks. If there are no blocks, set **nblock** $= 0$ or 1.

Constraint: **nblock** ≥ 0.

If **nblock** ≥ 2, **n**/**nblock** must be a multiple of the number of treatment combinations, that is a multiple of $\prod_{i=1}^{k}$**lfac**$[i-1]$.

6: **inter** – Integer *Input*

On entry: the maximum number of factors in an interaction term. If no interaction terms are to be computed, set **inter** $= 0$ or 1.

Constraint: $0 \leq$ **inter** $\leq$ **nfac**.

7: **irdf** – Integer *Input*

On entry: the adjustment to the residual and total degrees of freedom. The total degrees of freedom are set to **n** $-$ **irdf** and the residual degrees of freedom adjusted accordingly. For examples of the use of **irdf** see Section 9.

Constraint: **irdf** ≥ 0.

8: **mterm** – Integer * *Output*

On exit: the number of terms in the analysis of variance table, see Section 9.

The number of treatment effects is **mterm** $- 3$.

9: **table** – double ** *Output*

On exit: a pointer which points to **mterm** $\times 5$ memory locations, allocated internally. Viewing this memory as a two-dimensional **mterm** $\times 5$ array, the first **mterm** rows of **table** contain the analysis of variance table. The first column contains the degrees of freedom, the second column contains the sum of squares, the third column (except for the row corresponding to the total sum of squares) contains the mean squares, i.e., the sums of squares divided by the degrees of freedom, and the fourth and fifth columns contain the F ratio and significance level, respectively (except for rows corresponding to the total sum of squares, and the residual sum of squares). All other cells of the table are set to zero.

The first row corresponds to the blocks and is set to zero if there are no blocks. The **mterm**th row corresponds to the total sum of squares for **y** and the $(\mathbf{mterm}-1)$th row corresponds to the residual sum of squares. The central rows of the table correspond to the main effects followed by the interaction if specified by **inter**. The main effects are in the order specified by **lfac** and the interactions are in lexical order, see Section 3.

10: **tmean** – double ** *Output*

On exit: a pointer pointing to **maxt** memory locations, allocated internally. It contains the treatment means. The position of the means for an effect is given by the index in **imean**. For a given effect the means are in standard order, see Section 3.

11: **maxt** – Integer * *Output*

On exit: the number of treatment means that have been computed, see Section 9.

12: **e** – double ** *Output*

On exit: a pointer pointing to **maxt** memory locations, allocated internally. It contains the estimated effects in the same order as for the means in **tmean**.

13: **imean** – Integer ** *Output*

On exit: a pointer pointing to **mterm** memory locations, allocated internally. It indicates the position of the effect means in **tmean**. The effect means corresponding to the first treatment effect in the ANOVA table are stored in **tmean**[0] up to **tmean**[**imean**[0] − 1]. Other effect means corresponding to the ith treatment effect, for $i = 2, 3, \ldots,$ **mterm** $- 3$, are stored in **tmean**[**imean**[$i-2$]] up to **tmean**[**imean**[$i-1$] − 1].

14: **semean** – double ** *Output*

On exit: a pointer pointing to **mterm** memory locations, allocated internally. It contains the standard error of the difference between means corresponding to the ith treatment effect in the ANOVA table.

15: **bmean** – double ** *Output*

On exit: a pointer pointing to **nblock** $+ 1$ memory locations, allocated internally. **bmean**[0] contains the grand mean, if **nblock** > 1, **bmean**[1] up to **bmean**[**nblock**] contain the block means.

16: **r**[**n**] – double *Output*

On exit: the residuals.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_GT

On entry, **inter** $= \langle value \rangle$ while **nfac** $= \langle value \rangle$. These arguments must satisfy **inter** $\leq$ **nfac**.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_ARRAY_CONSTANT

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

NE_G04CA_RES_DF

There are no degrees of freedom for the residual or the residual sum of squares is zero. In either case the standard errors and F-statistics cannot be computed.

NE_INT_2

On entry, **nblock** $= \langle value \rangle$, **n** $= \langle value \rangle$.
Constraint: **n** must be a multiple of **nblock**, when **nblock** > 1.

NE_INT_ARG_LT

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

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

On entry, **n** $= \langle value \rangle$.
Constraint: **n** ≥ 4.

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

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

NE_INTARR

On entry, **lfac**$[\langle value \rangle] = \langle value \rangle$.
Constraint: **lfac**$[i-1] \geq 2$, for $i = 1, 2, \ldots,$ **nfac**.

NE_PLOT_TREAT

The number of plots per block is not a multiple of the number of treatment combinations.

7 Accuracy

The block and treatment sums of squares are computed from the block and treatment residual totals. The residuals are updated as each effect is computed and the residual sum of squares computed directly from the residuals. This avoids any loss of accuracy in subtracting sums of squares.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The number of rows in the ANOVA table and the number of treatment means are given by the following formulae.

Let there be k factors with levels l_i, for $i = 1, 2, \ldots, k$, and let t be the maximum number of terms in an interaction, then the number of rows in the ANOVA table is

$$\sum_{i=1}^{t} \binom{k}{i} + 3.$$

The number of treatment means is

$$\sum_{i=1}^{t} \prod_{j \in S_i} l_j,$$

where S_i is the set of all combinations of the k factors i at a time.

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 coefficients obtained from the regression can be examined for significance and used to produce an adjusted dependent variable using the original response variable and covariate. An approximate adjusted analysis of variance table can then be produced by using the adjusted dependent variable. In this case **irdf** should be set to one plus the number of fitted covariates.

For designs such as Latin squares one more of the blocking factors has to be removed in a preliminary analysis before the final analysis. This preliminary analysis can be performed using nag_anova_random (g04bbc) or a prior call to nag_anova_factorial (g04cac) if the data is reordered between calls. The residuals from the preliminary analysis are then input to nag_anova_factorial (g04cac). In these cases **irdf** should be set to the difference between **n** and the residual degrees of freedom from preliminary analysis. Care should be taken when using this approach as there is no check on the orthogonality of the two analyses.

If nag_anova_factorial (g04cac) is to be called repeatedly then the memory allocated to **table**, **tmean**, **e**, **imean**, **semean**, and **bmean** must be freed between calls. You are advised to call nag_anova_factorial_free (g04czc) to achieve this.

10 Example

The data, given by John and Quenouille (1977), is for the yield of turnips for a factorial experiment with two factors, the amount of phosphate with 6 levels and the amount of liming with 3 levels. The design was replicated in 3 blocks. The data is input and the analysis of variance computed. The analysis of variance table and tables of means with their standard errors are printed.

10.1 Program Text

```
/* nag_anova_factorial (g04cac) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 5, 1998.
 *
 * Mark 6 revised, 2000.
 * Mark 7b revised, 2004.
 * Mark 8 revised, 2004.
 */

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

#define MTERM 6

#define LFAC(I)     lfac[(I) -1]
#define IWK(I)      iwk[(I) -1]
#define IMEAN(I)    imean[(I) -1]
#define Y(I)        y[(I) -1]
#define TMEAN(I)    tmean[(I) -1]
#define SEMEAN(I)   semean[(I) -1]
#define R(I)        r[(I) -1]
#define E(I)        e[(I) -1]
#define BMEAN(I)    bmean[(I) -1]
#define TABLE(I, J) table[((I) -1) * (5) + ((J) -1)]
int main(void)
```

```
{

  Integer  c__27 = 27, exit_status = 0, i, *imean = 0, inter, irdf, j, k, l;
  Integer  *lfac = 0;
  Integer  mterm = MTERM, n, nblock, nfac, ntreat, num;
  NagError fail;
  double   *bmean = 0, *e = 0, *r = 0, *semean = 0, *table = 0, *tmean = 0;
  double   *y = 0;
  INIT_FAIL(fail);

  printf("nag_anova_factorial (g04cac) Example Program Results\n\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"%*[^\n]", &n, &nblock,
          &nfac, &inter);
#else
  scanf("%"NAG_IFMT"%"NAG_IFMT"%"NAG_IFMT"%"NAG_IFMT"%*[^\n]", &n, &nblock,
          &nfac, &inter);
#endif
  if (n >= 4 && nfac >= 1 && nblock >= 1 && !(n%nblock))
    {
      if (!(r = NAG_ALLOC(n, double)) ||
          !(y = NAG_ALLOC(n, double)) ||
          !(lfac = NAG_ALLOC(nfac, Integer)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
    }
  else
    {
      printf("Invalid n or nfac or nblock.\n");
      exit_status = 1;
      return exit_status;
    }
  for (j = 0; j < nfac; ++j)
#ifdef _WIN32
    scanf_s("%"NAG_IFMT"", &lfac[j]);
#else
    scanf("%"NAG_IFMT"", &lfac[j]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  for (i = 0; i < n; ++i)
#ifdef _WIN32
    scanf_s("%lf", &y[i]);
#else
    scanf("%lf", &y[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  irdf = 0;
  /* nag_anova_factorial (g04cac).
   * Complete factorial design
   */
```

```
  nag_anova_factorial(n, y, nfac, lfac, nblock, inter, irdf, &mterm, &table,
                      &tmean, &c__27, &e, &imean, &semean, &bmean, r,
                      &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_anova_factorial (g04cac).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  printf("\n  ANOVA table\n\n");
  printf("   Source      df         SS          MS          F"
         "          Prob\n\n");
  k = 0;
  if (nblock > 1)
    {
      ++k;
      printf("%s   ", " Blocks  ");
      printf("%4"NAG_IFMT"  ", (Integer) TABLE(1, 1));
      for (j = 2; j <= 5; ++j)
        printf("%12.2f", TABLE(1, j));
      printf("\n");
    }
  ntreat = mterm - 2 - k;
  for (i = 1; i <= ntreat; ++i)
    {
      printf("%s%2"NAG_IFMT" ", " Effect  ", i);
      printf("%4"NAG_IFMT"  ", (Integer) TABLE(k+i, 1));
      for (j = 2; j <= 5; ++j)
        printf("%12.2f", TABLE(k+i, j));
      printf("\n");
    }
  printf("%s   ", " Residual");
  printf("%4"NAG_IFMT"  ", (Integer) TABLE(mterm-1, 1));
  for (j = 2; j <= 3; ++j)
    printf("%12.2f", TABLE(mterm-1, j));
  printf("\n");

  printf("%s   ", " Total   ");
  printf("%4"NAG_IFMT"  ", (Integer) TABLE(mterm, 1));
  printf("%12.2f\n\n", TABLE(mterm, 2));

  printf("  Treatment Means and Standard Errors \n\n");
  k = 0;
  for (i = 0; i < ntreat; ++i)
    {
      l = imean[i];
      printf("%s%2"NAG_IFMT"\n\n", " Effect ", i+1);

      num = 1;
      for (j = k; j < l; ++j)
        {
          printf("%10.2f%s", tmean[j], num%8?"":"\n");
          num++;
        }

      printf("\n\n%s%10.2f\n\n", " SE of difference in means  = ",
              semean[i]);
      k = l;
    }
  /* nag_anova_factorial_free (g04czc).
   * Memory freeing function for nag_anova_factorial (g04cac)
   */
  nag_anova_factorial_free(&table, &tmean, &e, &imean, &semean, &bmean);
 END:
  NAG_FREE(r);
  NAG_FREE(y);
  NAG_FREE(lfac);
  return exit_status;
}
```

10.2 Program Data

```
nag_anova_factorial (g04cac) Example Program Data
54 3 2 2  : n nblock nfac inter
6 3       : lfac

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
```

10.3 Program Results

```
nag_anova_factorial (g04cac) Example Program Results


  ANOVA table

   Source      df         SS          MS           F          Prob

 Blocks         2      30118.78    15059.39        7.68        0.00
 Effect   1     5      73008.17    14601.63        7.45        0.00
 Effect   2     2      21596.33    10798.17        5.51        0.01
 Effect   3    10      31191.67     3119.17        1.59        0.15
 Residual      34      66627.89     1959.64
 Total         53     222542.83

  Treatment Means and Standard Errors

 Effect  1

    254.78    339.00    333.33    367.78    330.78    360.67

 SE of difference in means  =      20.87

 Effect  2

    334.28    353.78    305.11

 SE of difference in means  =      14.76

 Effect  3

    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

 SE of difference in means  =      36.14
```