

NAG Library Function Document

nag_order_data (g10zac)

1 Purpose

nag_order_data (g10zac) orders and weights data which is entered unsequentially, weighted or unweighted.

2 Specification

```
#include <nag.h>
#include <nagg10.h>

void nag_order_data (Integer n, const double x[], const double y[],
                    const double weights[], Integer *nord, double xord[], double yord[],
                    double wwt[], double *rss, NagError *fail)
```

3 Description

Given a set of observations (x_i, y_i) for $i = 1, 2, \dots, n$, with corresponding weights w_i , nag_order_data (g10zac) rearranges the observations so that the x_i are in ascending order.

For any equal x_i in the ordered set, say $x_j = x_{j+1} = \dots = x_{j+k}$, a single observation x_j is returned with a corresponding y' and w' , calculated as:

$$w' = \sum_{l=0}^k w_{i+l}$$

and

$$y' = \frac{\sum_{l=0}^k w_{i+l} y_{i+l}}{w'}$$

Observations with zero weight are ignored. If no weights are supplied by you, then unit weights are assumed; that is $w_i = 1$, for $i = 1, 2, \dots, n$.

In addition, the within group sum of squares is computed for the tied observations using West's algorithm (see West (1979)).

4 References

Draper N R and Smith H (1985) *Applied Regression Analysis* (2nd Edition) Wiley

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

5 Arguments

- 1: **n** – Integer *Input*
On entry: the number of observations, n .
Constraint: $n \geq 1$.
- 2: **x[n]** – const double *Input*
On entry: the values x_i , for $i = 1, 2, \dots, n$.

- 3: **y[n]** – const double *Input*
On entry: the values y_i , for $i = 1, 2, \dots, n$.
- 4: **weights[n]** – const double *Input*
On entry: **weights** must contain the n weights, if they are required. Otherwise, **weights** must be set to **NULL**.
Constraints:
 if **weights** are required, then $\mathbf{weights}[i - 1] \geq 0.0$, for $i = 1, 2, \dots, n$;
 at least one $\mathbf{weights}[i - 1] > 0.0$, for some i .
- 5: **nord** – Integer * *Output*
On exit: the number of distinct observations.
- 6: **xord[n]** – double *Output*
On exit: the first **nord** elements contain the ordered and distinct x_i .
- 7: **yord[n]** – double *Output*
On exit: the first **nord** elements contain the values y' corresponding to the values in **xord**.
- 8: **wwt[n]** – double *Output*
On exit: the first **nord** elements contain the values w' corresponding to the values of **xord** and **yord**.
- 9: **rss** – double * *Output*
On exit: the within group sum of squares for tied observations.
- 10: **fail** – NagError * *Input/Output*
 The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_ARRAY_CONS

The contents of array **weights** are not valid.
 Constraint: at least one element of **weights** must be > 0 .

NE_INT_ARG_LT

On entry, $\mathbf{n} = \langle \text{value} \rangle$.
 Constraint: $\mathbf{n} \geq 1$.

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_ARRAY_CONS

On entry, $\mathbf{weights}[\langle \text{value} \rangle] = \langle \text{value} \rangle$.
 Constraint: $\mathbf{weights}[i] \geq 0$, for $i = 0, 1, \dots, n - 1$.

7 Accuracy

For a discussion on the accuracy of the algorithm for computing mean and variance see West (1979).

8 Parallelism and Performance

Not applicable.

9 Further Comments

nag_order_data (g10zac) may be used to compute the pure error sum of squares in simple linear regression along with nag_regsn_mult_linear (g02dac), see Draper and Smith (1985).

10 Example

A set of unweighted observations are input and nag_order_data (g10zac) used to produce a set of strictly increasing weighted observations.

10.1 Program Text

```

/* nag_order_data (g10zac) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 6, 2000.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg10.h>

int main(void)
{
    Integer    exit_status = 0, i, *iwrk = 0, n, nord;
    NagError   fail;
    char       weight[2];
    double     rss, *weights = 0, *wtord = 0, *wtptr, *x = 0, *xord = 0, *y = 0;
    double     *yord = 0;

    INIT_FAIL(fail);

    printf("nag_order_data (g10zac) Example Program Results\n");

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

#ifdef _WIN32
    scanf_s("%"NAG_IFMT"", &n);
#else
    scanf("%"NAG_IFMT"", &n);
#endif

    if (!(x = NAG_ALLOC(n, double))
        || !(y = NAG_ALLOC(n, double))
        || !(weights = NAG_ALLOC(n, double))
        || !(xord = NAG_ALLOC(n, double))
        || !(yord = NAG_ALLOC(n, double))
        || !(wtord = NAG_ALLOC(n, double))
        || !(iwrk = NAG_ALLOC(n, Integer)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
    }
}

```

```

        goto END;
    }

#ifdef _WIN32
    scanf_s(" %1s ", weight, _countof(weight));
#else
    scanf(" %1s ", weight);
#endif
    for (i = 1; i <= n; ++i)
#ifdef _WIN32
        scanf_s("%lf %lf", &x[i - 1], &y[i - 1]);
#else
        scanf("%lf %lf", &x[i - 1], &y[i - 1]);
#endif
    if (*weight == 'W')
        wtptr = weights;
    else
        wtptr = 0;

    /* nag_order_data (g10zac).
     * Reorder data to give ordered distinct observations
     */
    nag_order_data(n, x, y, wtptr, &nord, xord, yord, wtord, &rss, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_order_data (g10zac).\n%s\n",
            fail.message);
        exit_status = 1;
        goto END;
    }

    /* Print results */
    printf("\n");
    printf("%s%6"NAG_IFMT"\n", "Number of distinct observations = ",
        nord);
    printf("%s%13.5f\n", "Residual sum of squares = ", rss);
    printf("\n");
    printf("          X          Y          WEIGHTS\n");
    for (i = 1; i <= nord; ++i)
        printf("          %13.5f          %13.5f          %13.5f\n", xord[i - 1],
            yord[i - 1], wtord[i - 1]);
END:
    NAG_FREE(x);
    NAG_FREE(y);
    NAG_FREE(weights);
    NAG_FREE(xord);
    NAG_FREE(yord);
    NAG_FREE(wtord);
    NAG_FREE(iwrk);
    return exit_status;
}

```

10.2 Program Data

nag_order_data (g10zac) Example Program Data

```

10
U
1.0 4.0
3.0 4.0
5.0 1.0
5.0 2.0
3.0 5.0
4.0 3.0
9.0 4.0
6.0 9.0
9.0 7.0
9.0 4.0

```

10.3 Program Results

nag_order_data (g10zac) Example Program Results

Number of distinct observations = 6
Residual sum of squares = 7.00000

X	Y	WEIGHTS
1.00000	4.00000	1.00000
3.00000	4.50000	2.00000
4.00000	3.00000	1.00000
5.00000	1.50000	2.00000
6.00000	9.00000	1.00000
9.00000	5.00000	3.00000
