1 Purpose

nag_tsa_mean_range (g13auc) calculates the range (or standard deviation) and the mean for groups of successive time series values. It is intended for use in the construction of range-mean plots.

2 Specification

```cpp
#include <nag.h>
#include <nagg13.h>

void nag_tsa_mean_range (Integer n, const double z[], Integer m,
                          Nag_RangeStat rs, double y[], double mean[], NagError *fail)
```

3 Description

Let $Z_1, Z_2, \ldots, Z_n$ denote $n$ successive observations in a time series. The series may be divided into groups of $m$ successive values and for each group the range or standard deviation (depending on a user-supplied option) and the mean are calculated. If $n$ is not a multiple of $m$ then groups of equal size $m$ are found starting from the end of the series of observations provided, and any remaining observations at the start of the series are ignored. The number of groups used, $k$, is the integer part of $n/m$. If you wish to ensure that no observations are ignored then the number of observations, $n$, should be chosen so that $n$ is divisible by $m$.

The mean, $M_i$, the range, $R_i$, and the standard deviation, $S_i$, for the $i$th group are defined as

$$M_i = \frac{1}{m} \sum_{j=1}^{m} Z_{l+m(i-1)+j}$$

$$R_i = \max_{1 \leq j \leq m} \{Z_{l+m(i-1)+j}\} - \min_{1 \leq j \leq m} \{Z_{l+m(i-1)+j}\}$$

and

$$S_i = \sqrt{\frac{1}{m-1} \sum_{j=1}^{m} (Z_{l+m(i-1)+j} - M_i)^2}$$

where $l = n - km$, the number of observations ignored.

For seasonal data it is recommended that $m$ should be equal to the seasonal period. For non-seasonal data the recommended group size is 8.

A plot of range against mean or of standard deviation against mean is useful for finding a transformation of the series which makes the variance constant. If the plot appears random or the range (or standard deviation) seems to be constant irrespective of the mean level then this suggests that no transformation of the time series is called for. On the other hand an approximate linear relationship between range (or standard deviation) and mean would indicate that a log transformation is appropriate. Further details may be found in either Jenkins (1979) or McLeod (1982).

You have the choice of whether to use the range or the standard deviation as a measure of variability. If the group size is small they are both equally good but if the group size is fairly large (e.g., $m = 12$ for monthly data) then the range may not be as good an estimate of variability as the standard deviation.
4 References

Jenkins G M (1979) *Practical Experiences with Modelling and Forecasting Time Series* GJP Publications, Lancaster


5 Arguments

1: \( n \) – Integer  
   *Input*  
   *On entry:* \( n \), the number of observations in the time series.  
   *Constraint:* \( n \geq m \).

2: \( z[n] \) – const double  
   *Input*  
   *On entry:* \( z[t-1] \) must contain the \( t \)th observation \( Z_t \), for \( t = 1, 2, \ldots, n \).

3: \( m \) – Integer  
   *Input*  
   *On entry:* \( m \), the group size.  
   *Constraint:* \( m \geq 2 \).

4: \( rs \) – Nag_RangeStat  
   *Input*  
   *On entry:* indicates whether ranges or standard deviations are to be calculated.  
   \( rs = \) Nag_UseRange  
   Ranges are calculated.  
   \( rs = \) Nag_UseSD  
   Standard deviations are calculated.  
   *Constraint:* \( rs = \) Nag_UseRange or Nag_UseSD.

5: \( y\lfloor\text{int}(n/m)\rfloor \) – double  
   *Output*  
   *On exit:* \( y[i-1] \) contains the range or standard deviation, as determined by \( rs \), of the \( i \)th group of observations, for \( i = 1, 2, \ldots, k \).

6: \( \text{mean}[\text{int}(n/m)] \) – double  
   *Output*  
   *On exit:* \( \text{mean}[i-1] \) contains the mean of the \( i \)th group of observations, for \( i = 1, 2, \ldots, k \).

7: \( \text{fail} \) – 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.  
See Section 3.2.1.2 in the Essential Introduction for further information.

**NE_BAD_PARAM**  
On entry, argument \( \langle value \rangle \) had an illegal value.
NE_INT
On entry, \( m = \langle\text{value}\rangle \).
Constraint: \( m \geq 2 \).

NE_INT_2
On entry, \( n = \langle\text{value}\rangle \) and \( m = \langle\text{value}\rangle \).
Constraint: \( n \geq m \).

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.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

NE_NO_LICENCE
Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

7 Accuracy
The computations are believed to be stable.

8 Parallelism and Performance
Not applicable.

9 Further Comments
The time taken by nag_tsa_mean_range (g13auc) is approximately proportional to \( n \).

10 Example
The following program produces the statistics for a range-mean plot for a series of 100 observations
divided into groups of 8.

10.1 Program Text
/* nag_tsa_mean_range (g13auc) Example Program. *
 * Copyright 2014 Numerical Algorithms Group.
 * * Mark 7, 2002.
 */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <naggl3.h>

int main(void)
{
    /* Scalars */
    Integer exit_status, i, ngrps, m, n;

    /* Arrays */
    double *mean = 0, *range = 0, *z = 0;
    NagError fail;
    INIT_FAIL(fail);
exit_status = 0;

printf("nag_tsa_mean_range (g13auc) Example Program Results\n");

#ifdef _WIN32
    scanf_s("%[^\n]");
#else
    scanf("%[^\n]");
#endif

#ifdef _WIN32
    scanf_s("%"NAG_IFMT"%"NAG_IFMT"%[^\n]", &n, &m);
#else
    scanf("%"NAG_IFMT"%"NAG_IFMT"%[^\n]", &n, &m);
#endif

if (n >= m && m >= 1)
{
   ngrps = n / m;

    /* Allocate arrays */
    if (!(mean = NAG_ALLOC(ngrps, double)) ||
        !(range = NAG_ALLOC(ngrps, double)) ||
        !(z = NAG_ALLOC(n, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

    for (i = 1; i <= n; ++i)
    {
        #ifdef _WIN32
            scanf_s("%lf", &z[i-1]);
        #else
            scanf("%lf", &z[i-1]);
        #endif
        #ifdef _WIN32
            scanf_s("%[^\n]");
        #else
            scanf("%[^\n]");
        #endif
    }

    printf("\n");

    /* nag_tsa_mean_range (g13auc).
     * Computes quantities needed for range-mean or standard deviation-mean plot.
     */
    nag_tsa_mean_range(n, z, m, Nag_UseRange, range, mean, &fail);

    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_tsa_mean_range (g13auc).\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }

    printf(" Range Mean\n");
    for (i = 1; i <= ngrps; i++)
        printf("%8.3f %8.3f\n", range[i-1], mean[i-1]);

END:
NAG_FREE(mean);
NAG_FREE(range);
NAG_FREE(z);

return exit_status;
}
10.2 Program Data

nag_tsa_mean_range (g13auc) Example Program Data

<table>
<thead>
<tr>
<th>Range</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>148.000</td>
<td>72.375</td>
</tr>
<tr>
<td>123.000</td>
<td>70.000</td>
</tr>
<tr>
<td>84.000</td>
<td>43.500</td>
</tr>
<tr>
<td>45.000</td>
<td>29.750</td>
</tr>
<tr>
<td>28.000</td>
<td>7.625</td>
</tr>
<tr>
<td>40.000</td>
<td>26.750</td>
</tr>
<tr>
<td>65.000</td>
<td>30.250</td>
</tr>
<tr>
<td>131.000</td>
<td>61.000</td>
</tr>
<tr>
<td>92.000</td>
<td>47.625</td>
</tr>
<tr>
<td>85.000</td>
<td>75.250</td>
</tr>
<tr>
<td>92.000</td>
<td>46.875</td>
</tr>
<tr>
<td>67.000</td>
<td>39.250</td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_tsa_mean_range (g13auc) Example Program Results

Example Program

Plot of Range vs Mean (Y vs Mean)