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
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 the user wishes 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

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

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 nonseasonal 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).

The user has 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 Parameters

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

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

3: \( m \) – Integer  \hspace{1cm} Input
   
   On entry: the group size, \( m \).
   
   Constraint: \( m \geq 2 \).

4: \( rs \) – Nag_RangeStat  \hspace{1cm} Input
   
   On entry: indicates whether ranges or standard deviations are to be calculated.
   
   If \( rs = \text{Nag} \_\text{UseRange} \), then ranges are calculated.
   
   If \( rs = \text{Nag} \_\text{UseSD} \), then standard deviations are calculated.
   
   Constraint: \( rs = \text{Nag} \_\text{UseRange} \) or \( \text{Nag} \_\text{UseSD} \).

5: \( y[dim] \) – double  \hspace{1cm} Output
   
   Note: the dimension, \( dim \), of the array \( y \) must be at least \( (n/m) \).
   
   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: \( \text{mean[dim]} \) – double  \hspace{1cm} Output
   
   Note: the dimension, \( dim \), of the array \( \text{mean} \) must be at least \( (n/m) \).
   
   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 *  \hspace{1cm} Input/Output
   
   The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

**NE_INT**

On entry, \( m = \langle \text{value} \rangle \).

Constraint: \( m \geq 2 \).

**NE_INT_2**

On entry, \( n = \langle \text{value} \rangle \), \( m = \langle \text{value} \rangle \).

Constraint: \( n \geq m \).

**NE_BAD_PARAM**

On entry, parameter \( \langle \text{value} \rangle \) 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

The computations are believed to be stable.

8 Further Comments

The time taken by the routine is approximately proportional to $n$.

9 Example

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

9.1 Program Text

```c
/* nag_tsa_mean_range (g13auc) Example Program. */
/* Copyright 2002 Numerical Algorithms Group. */
/* Mark 7, 2002. */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg13.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;

    Vprintf("g13auc Example Program Results\n");

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

    Vscanf("%ld%ld%*[\n] ", &n, &m);
    if (n >= m && m >= 1)
    {
        ngrps = n / m;

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

        for (i = 1; i <= n; ++i)
            Vscanf("%lf", &z[i-1]);

        Vprintf("\n");

        g13auc(n, z, m, Nag_UseRange, range, mean, &fail);
    }
}
```

[NP3645/7]


```
{ 
  Vprintf("Error from gl3auc.\n%s\n", fail.message);
  exit_status = 1;
  goto END;
}

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

END:
if (mean) NAG_FREE(mean);
if (range) NAG_FREE(range);
if (z) NAG_FREE(z);
return exit_status;
}

9.2 Program Data

```
g13auc Example Program Data
```
100 8 : n, no. of obs in time series, m, no. of obs in each group
101 82 66 35 31 6 20 90 154 125
85 68 38 23 10 24 83 133 131 118
90 67 60 47 41 21 16 6 4 7
14 34 45 43 49 42 28 10 5 2
0 1 3 12 14 35 47 41 30 24
16 7 4 2 8 13 36 50 62 67
72 48 29 8 13 57 122 139 103 86
63 37 26 11 15 40 62 98 124 96
65 64 54 39 21 7 4 23 53 94
96 77 59 44 47 30 16 7 37 74 : End of time series
```

9.3 Program Results

```
g13auc Example Program Results
```

<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>