NAG C Library Function Document

nag_tsa_diff (g13aac)

1 Purpose

nag_tsa_diff (g13aac) carries out non-seasonal and seasonal differencing on a time series. Information which allows the original series to be reconstituted from the differenced series is also produced. This information is required in time series forecasting.

2 Specification

```c
void nag_tsa_diff (const double x[], Integer nx, Integer d, Integer ds, Integer s,
double xd[], Integer *nxd, NagError *fail)
```

3 Description

Let \( \nabla d \nabla D s x_i \) be the \( i \)th value of a time series \( x_i \), for \( i = 1, 2, \ldots, n \) after non-seasonal differencing of order \( d \) and seasonal differencing of order \( D \) (with period or seasonality \( s \)). In general,

\[

\begin{align*}
\nabla d \nabla D s x_i &= \nabla^{d-1} \nabla D s x_{i+1} - \nabla^{d-1} \nabla D s x_i & d > 0 \\
\nabla d \nabla D s x_i &= \nabla^d \nabla^{D-1}s x_{i+s} - \nabla^d \nabla^{D-1}s x_i & D > 0
\end{align*}
\]

Non-seasonal differencing up to the required order \( d \) is obtained using

\[

\begin{align*}
\nabla 1 x_i &= x_{i+1} - x_i & \text{for } i = 1, 2, \ldots, (n-1) \\
\nabla 2 x_i &= \nabla 1 x_{i+1} - \nabla 1 x_i & \text{for } i = 1, 2, \ldots, (n-2) \\
\vdots \\
\nabla d x_i &= \nabla^{d-1} x_{i+1} - \nabla^{d-1} x_i & \text{for } i = 1, 2, \ldots, (n-d)
\end{align*}
\]

Seasonal differencing up to the required order \( D \) is then obtained using

\[

\begin{align*}
\nabla 1 x_i &= \nabla s x_{i+s} - \nabla s x_i & \text{for } i = 1, 2, \ldots, (n-d-s) \\
\nabla 2 x_i &= \nabla 1 x_{i+s} - \nabla 1 x_i & \text{for } i = 1, 2, \ldots, (n-d-2s) \\
\vdots \\
\nabla D x_i &= \nabla D+1 s x_{i+s} - \nabla D+1 s x_i & \text{for } i = 1, 2, \ldots, (n-d-D \times s)
\end{align*}
\]

Mathematically, the sequence in which the differencing operations are performed does not affect the final resulting series of \( m = n - d - D \times s \) values.

4 References

None.

5 Parameters

1. \( x[nx] \) – const double
   
   *Input*
   
   On entry: the undifferenced time series, \( x_i \), for \( i = 1, 2, \ldots, n \).

2. \( nx \) – Integer
   
   *Input*
   
   On entry: the number of values, \( n \), in the undifferenced time series.
   
   **Constraint:** \( nx > d + (ds \times s) \).
g13aac

3:  d – Integer
    On entry: the order of non-seasonal differencing, \( d \).
    Constraint: \( d \geq 0 \).

4:  ds – Integer
    On entry: the order of seasonal differencing, \( D \).
    Constraint: \( ds \geq 0 \).

5:  s – Integer
    On entry: the seasonality, \( s \).
    Constraints:
    if \( ds > 0 \), \( s > 0 \);
    if \( ds = 0 \), \( s \geq 0 \).

6:  xd[nx] – double
    On exit: the differenced values in elements 1 to \( nx \), and reconstitution data in the remainder of the array.

7:  nx – Integer *
    On exit: the number of differenced values in the array \( xd \).

8:  fail – NagError *
    The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

**NE_INT**
On entry, \( s = \text{(value)} \).
Constraint: \( s \geq 0 \).
On entry, \( ds = \text{(value)} \).
Constraint: \( ds \geq 0 \).
On entry, \( d = \text{(value)} \).
Constraint: \( d \geq 0 \).

**NE_INT_2**
On entry, \( s = 0 \) and \( ds > 0 \): \( ds = \text{(value)} \).

**NE_INT_4**
On entry, \( nx \leq d + (ds \times s) \): \( nx = \text{(value)} \), \( d = \text{(value)} \), \( ds = \text{(value)} \), \( s = \text{(value)} \).

**NE_BAD_PARAM**
On entry, parameter \( \text{(value)} \) 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 \((d + ds) \times nx\).

9 Example
The example program reads in a set of data consisting of 20 observations from a time series. Non-seasonal differencing of order 2 and seasonal differencing of order 1 (with seasonality of 4) are applied to the input data, giving an output array holding 14 differenced values and 6 values which can be used to reconstitute the output array.

9.1 Program Text
/* nag_tsa_diff (g13aac) 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, d, ds, s, nx, nxd;
  NagError fail;
  /* Arrays */
  double *x = 0, *xd = 0;

  INIT_FAIL(fail);
  exit_status = 0;
  Vprintf("g13aac Example Program Results\n");
  /* Skip heading in data file */
  Vscanf("%*[^\n] ", &nx, &d, &ds, &s);

  if (nx > 0)
  {
    /* Allocate memory */
    if ( !(x = NAG_ALLOC(nx, double)) || 
        !(xd = NAG_ALLOC(nx, double)) )
    {
      Vprintf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }
    for (i = 1; i <= nx; ++i)
      Vscanf("%lf", &x[i-1]);
    Vscanf("%*[^\n] ");
    Vprintf("Non-seasonal differencing of order %ld ", d);
    Vprintf("and seasonal differencing\nof order %ld ", ds);
    Vprintf("with seasonality %ld are applied\n", s);
  }
END:
g13aac

if (fail.code != NE_NOERROR)
{
    Vprintf("Error from g13aac.\n\n", fail.message);
    exit_status = 1;
    goto END;
}

Vprintf("\n");
Vprintf("The output array holds %2ld values, of which the "
"first %2ld are differenced values\n\n", nx, nxd);

for (i = 1; i <= nx; ++i)
{
    Vprintf("%10.1f", xd[i-1]);
    if (i % 5 == 0 || i == nx)
        Vprintf("\n");
}

END:
    if (x) NAG_FREE(x);
    if (xd) NAG_FREE(xd);
    return exit_status;

9.2 Program Data

g13aac Example Program Data
2 2 1 4
120.0 108.0 98.0 118.0 135.0
131.0 118.0 125.0 121.0 100.0
82.0 82.0 89.0 88.0 86.0
96.0 108.0 110.0 99.0 105.0

9.3 Program Results

g13aac Example Program Results

Non-seasonal differencing of order 2 and seasonal differencing
of order 1 with seasonality 4 are applied

The output array holds 20 values, of which the first 14 are differenced values

-11.0 -10.0 -8.0  4.0  12.0
-2.0  18.0  9.0 -4.0 -6.0
-5.0 -2.0 -12.0  5.0  2.0
-10.0 -13.0  17.0  6.0  105.0