NAG Library Routine Document

G13AGF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

G13AGF accepts a series of new observations of a time series, the model of which is already fully specified, and updates the 'state set' information for use in constructing further forecasts. The previous specifications of the time series model should have been obtained by using G13AEF or G13AFF to estimate the relevant parameters. The supplied state set will originally have been produced by G13AEF or G13AFF, but may since have been updated by earlier calls to G13AGF.

A set of residuals corresponding to the new observations is returned. These may be of use in checking that the new observations conform to the previously fitted model.

2 Specification

```
SUBROUTINE G13AGF (ST, NST, MR, PAR, NPAR, C, ANX, NUV, ANEXR, WA, NWA,    &
                   IFAIL)

INTEGER            NST, MR(7), NPAR, NUV, NWA, IFAIL
REAL (KIND=nag_wp) ST(NST), PAR(NPAR), C, ANX(NUV), ANEXR(NUV),            &
                   WA(NWA)
```

3 Description

The time series model is specified as outlined in Section 3 in G13AEF or G13AFF. This also describes how the state set, which contains the minimum amount of time series information needed to construct forecasts, is made up of

(i) the differenced series w_t (uncorrected for the constant c), for $(N - P \times s) < t \leq N$,

(ii) the d' values required to reconstitute the original series x_t from the differenced series w_t,

(iii) the intermediate series e_t, for $(N - \max(p, Q \times s)) < t \leq N$, and

(iv) the residual series a_t, for $(N - q) < t \leq N$.

If the number of original undifferenced observations was n, then $d' = d + (D \times s)$ and $N = n - d'$.

To update the state set, given a number of new undifferenced observations x_t, $t = n + 1, n + 2, \ldots, n + k$, the four series above are first reconstituted.

Differencing and residual calculation operations are then applied to the new observations and k new values of w_t, e_t and a_t are derived.

The first k values in these three series are then discarded and a new state set is obtained.

The residuals in the a_t series corresponding to the k new observations are preserved in an output array. The parameters of the time series model are not changed in this routine.

4 References

None.

5 Arguments

1: ST(NST) – REAL (KIND=nag_wp) array *Input/Output*

On entry: the state set derived from G13AEF or G13AFF, or as modified using earlier calls of G13AGF.

On exit: the updated values of the state set.

2: NST – INTEGER *Input*

On entry: the number of values in the state set array ST.

Constraint: $\text{NST} = P \times s + D \times s + d + q + \max(p, Q \times s)$. (As returned by G13AEF or G13AFF).

3: MR(7) – INTEGER array *Input*

On entry: the orders vector (p, d, q, P, D, Q, s) of the ARIMA model, in the usual notation.

Constraints:

$p, d, q, P, D, Q, s \geq 0$;
$p + q + P + Q > 0$;
$s \neq 1$;
if $s = 0$, $P + D + Q = 0$;
if $s > 1$, $P + D + Q > 0$.

4: PAR(NPAR) – REAL (KIND=nag_wp) array *Input*

On entry: the estimates of the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters and the Q values of the Θ parameters in the model – in that order, using the usual notation.

5: NPAR – INTEGER *Input*

On entry: the number of ϕ, θ, Φ and Θ parameters in the model.

Constraint: $\text{NPAR} = p + q + P + Q$.

6: C – REAL (KIND=nag_wp) *Input*

On entry: the constant to be subtracted from the differenced data.

7: ANX(NUV) – REAL (KIND=nag_wp) array *Input*

On entry: the new undifferenced observations which are to be used to update ST.

8: NUV – INTEGER *Input*

On entry: k, the number of new observations in ANX.

9: ANEXR(NUV) – REAL (KIND=nag_wp) array *Output*

On exit: the residuals corresponding to the new observations in ANX.

10: WA(NWA) – REAL (KIND=nag_wp) array *Workspace*
11: NWA – INTEGER *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13AGF is called.

Constraint: $\text{NWA} \geq (4 \times \text{NPAR} + 3 \times \text{NST})$.

12: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, NPAR $\neq p + q + P + Q$,
or the orders vector MR is invalid (check the constraints in Section 5).

IFAIL $= 2$

On entry, NST $\neq P \times s + D \times s + d + q + \max(Q \times s, p)$.

IFAIL $= 3$

On entry, NUV ≤ 0.

IFAIL $= 4$

On entry, NWA $< 4 \times$ NPAR $+ 3 \times$ NST.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computations are believed to be stable.

8 Parallelism and Performance

G13AGF is not threaded in any implementation.

9 Further Comments

The time taken by G13AGF is approximately proportional to NUV $\times$ NPAR.

10 Example

The following program is based on data derived from a study of monthly airline passenger totals (in thousands) to which a logarithmic transformation had been applied. The time series model was based on seasonal and non-seasonal differencing both of order 1, with seasonal period 12. The number of parameters estimated was two: a non-seasonal moving average parameter θ_1 with value 0.327 and a seasonal moving average parameter Θ_1 with value 0.6270. There was no constant correction. These, together with the state set array, were obtained using G13AEF.

Twelve new observations are supplied. The routine updates the state set and outputs a set of residuals corresponding to the new observations.

10.1 Program Text

```
    Program g13agfe

!     G13AGF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g13agf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: c
      Integer                          :: ifail, npar, nst, nuv, nwa
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: anexr(:), anx(:), par(:), st(:),     &
                                          wa(:)
      Integer                          :: mr(7)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max
!     .. Executable Statements ..
      Write (nout,*) 'G13AGF Example Program Results'
      Write (nout,*)

!     Skip heading in data file
      Read (nin,*)

!     Read in the problem size etc
      Read (nin,*) nuv, c

!     Read in the orders
      Read (nin,*) mr(1:7)

!     Calculate NPAR and various array lengths
      npar = mr(1) + mr(3) + mr(4) + mr(6)
      nst = mr(4)*mr(7) + mr(5)*mr(7) + mr(2) + mr(3) + max(mr(1),mr(6)*mr(7))
      nwa = 4*npar + 3*nst

      Allocate (st(nst),anx(nuv),anexr(nuv),wa(nwa),par(npar))

!     Read in parameter estimates
      Read (nin,*) par(1:npar)

!     Read in state set from G13AEF, G13AFF or previous call to G13AGF
      Read (nin,*) st(1:nst)

!     Read in new observations to update state set
      Read (nin,*) anx(1:nuv)
```

```
!     Update state set
      ifail = 0
      Call g13agf(st,nst,mr,par,npar,c,anx,nuv,anexr,wa,nwa,ifail)

!     Display results
      Write (nout,*) 'The updated state set array now holds the values'
      Write (nout,99999) st(1:nst)
      Write (nout,*)
      Write (nout,99998) 'The residuals corresponding to the', nuv
      Write (nout,*) 'values used to update the system are'
      Write (nout,99999) anexr(1:nuv)

99999 Format (1X,8F8.4)
99998 Format (1X,A,I3,A)
    End Program g13agfe
```

10.2 Program Data

```
G13AGF Example Program Data
  12  0.0                                                            :: NUV,C
   0   1   1   0   1   1   12                                        :: MR
  0.3270  0.6270                                                     :: PAR
  0.0118 -0.0669  0.1296 -0.0394  0.0422  0.1809  0.1211  0.0281
 -0.2231 -0.1181 -0.1468  0.0835  5.8201 -0.0157 -0.0361 -0.0266
 -0.0199  0.0298  0.0290  0.0147  0.0373 -0.0931  0.0223 -0.0172
 -0.0353 -0.0413                                                     :: End of ST
  5.8861  5.8348  6.0064  5.9814  6.0403  6.1570  6.3063  6.3261
  6.1377  6.0088  5.8916  6.0039                                     :: End of ANX
```

10.3 Program Results

```
 G13AGF Example Program Results

 The updated state set array now holds the values
   0.0660 -0.0513  0.1716 -0.0250  0.0589  0.1167  0.1493  0.0198
  -0.1884 -0.1289 -0.1172  0.1123  6.0039  0.0444 -0.0070  0.0253
   0.0019  0.0354 -0.0460  0.0374  0.0151 -0.0237  0.0032  0.0188
   0.0067  0.0126

 The residuals corresponding to the 12
 values used to update the system are
   0.0309  0.0031  0.0263  0.0105  0.0388 -0.0333  0.0265  0.0238
  -0.0159 -0.0020  0.0182  0.0126
```