NAG Toolbox for MATLAB

g13dj

1 Purpose

g13dj computes forecasts of a multivariate time series. It is assumed that a vector ARMA model has already been fitted to the appropriately differenced/transformed time series using g13dc. The standard deviations of the forecast errors are also returned. A reference vector is set up so that, should future series values become available, the forecasts and their standard errors may be updated by calling g13dk.

2 Syntax

```
[qq, predz, sefz, ref, ifail] = g13dj(z, tr, id, delta, ip, iq, mean,
par, qq, v, lmax, lref, 'k', k, 'n', n, 'lpar', lpar)
```

3 Description

Let the vector $Z_t = (z_{1t}, z_{2t}, \ldots, z_{kt})^{\mathrm{T}}$, for $t = 1, 2, \ldots, n$, denote a k-dimensional time series for which forecasts of $Z_{n+1}, Z_{n+2}, \ldots, Z_{n+l_{\max}}$ are required. Let $W_t = (w_{1t}, w_{2t}, \ldots, w_{kt})^{\mathrm{T}}$ be defined as follows:

$$w_{it} = \delta_i(B) z_{it}^*, \qquad i = 1, 2, \ldots, k,$$

where $\delta_i(B)$ is the differencing operator applied to the ith series and where z_{it}^* is equal to either z_{it}, $\sqrt{z_{it}}$ or $\log_e z_{it}$ depending on whether or not a transformation was required to stabilize the variance before fitting the model.

If the order of differencing required for the ith series is d_i, then the differencing operator for the ith series is defined by $\delta_i(B) = 1 - \delta_{i1}B - \delta_{i2}B^2 - \cdots - \delta_{id_i}B^{d_i}$ where B is the backward shift operator; that is, $BZ_t = Z_{t-1}$. The differencing parameters δ_{ij}, for $i = 1, 2, \ldots, k$; $j = 1, 2, \ldots, d_i$, must be supplied by you. If the ith series does not require differencing, then $d_i = 0$.

W_t is assumed to follow a multivariate ARMA model of the form:

$$W_t - \mu = \phi_1(W_{t-1} - \mu) + \phi_2(W_{t-2} - \mu) + \cdots + \phi_p\big(W_{t-p} - \mu\big) + \epsilon_t - \theta_1\epsilon_{t-1} - \cdots - \theta_q\epsilon_{t-q}, \qquad (1)$$

where $\epsilon_t = (\epsilon_{1t}, \epsilon_{2t}, \ldots, \epsilon_{kt})^{\mathrm{T}}$, for $t = 1, 2, \ldots, n$, is a vector of k residual series assumed to be Normally distributed with zero mean and positive-definite covariance matrix Σ. The components of ϵ_t are assumed to be uncorrelated at non-simultaneous lags. The ϕ_i and θ_j are k by k matrices of parameters. The matrices ϕ_i, for $i = 1, 2, \ldots, p$, are the autoregressive (AR) parameter matrices, and the matrices θ_i, for $i = 1, 2, \ldots, q$, the moving average (MA) parameter matrices. The parameters in the model are thus the p (k by k) ϕ-matrices, the q (k by k) θ-matrices, the mean vector μ and the residual error covariance matrix Σ. The ARMA model (1) must be both stationary and invertible; see g13dx for a method of checking these conditions.

The ARMA model (1) may be rewritten as

$$\phi(B)(\delta(B)Z_t^* - \mu) = \theta(B)\epsilon_t,$$

where $\phi(B)$ and $\theta(B)$ are the autoregressive and moving average polynomials and $\delta(B)$ denotes the k by k diagonal matrix whose ith diagonal elements is $\delta_i(B)$ and $Z_t^* = (z_{1t}^*, z_{2t}^* \ldots z_{kt}^*)^{\mathrm{T}}$.

This may be rewritten as

$$\phi(B)\delta(B)Z_t^* = \phi(B)\mu + \theta(B)\epsilon_t$$

or

$$Z_t^* = \tau + \psi(B)\epsilon_t = \tau + \epsilon_t + \psi_1\epsilon_{t-1} + \psi_2\epsilon_{t-2} + \cdots$$

where $\psi(B) = \delta^{-1}(B)\phi^{-1}(B)\theta(B)$ and $\tau = \delta^{-1}(B)\mu$ is a vector of length k.

Forecasts are computed using a multivariate version of the procedure described in Box and Jenkins 1976. If $\hat{Z}_n^*(l)$ denotes the forecast of Z_{n+l}^*, then $\hat{Z}_n^*(l)$ is taken to be that linear function of $Z_n^*, Z_{n-1}^*, \ldots$ which minimizes the elements of $E\{e_n(l)e_n'(l)\}$ where $e_n(l) = Z_{n+l}^* - \hat{Z}_n^*(l)$ is the forecast error. $\hat{Z}_n^*(l)$ is referred to as the linear minimum mean square error forecast of Z_{n+l}^*.

The linear predictor which minimizes the mean square error may be expressed as

$$\hat{Z}_n^*(l) = \tau + \psi_l \epsilon_n + \psi_{l+1}\epsilon_{n-1} + \psi_{l+2}\epsilon_{n-2} + \cdots.$$

The forecast error at t for lead l is then

$$e_n(l) = Z_{n+l}^* - \hat{Z}_n^*(l) = \epsilon_{n+l} + \psi_1\epsilon_{n+l-1} + \psi_2\epsilon_{n+l-2} + \cdots + \psi_{l-1}\epsilon_{n+1}.$$

Let $d = \max(d_i)$, for $i = 1, 2, \ldots, k$. Unless $q = 0$ the function requires estimates of ϵ_t, for $t = d+1, d+2, \ldots, n$, which are obtainable from g13dc. The terms ϵ_t are assumed to be zero, for $t = n+1, n+2, \ldots, n+l_{\max}$. You may use g13dk to update these $l_{\max}$ forecasts should further observations, $Z_{n+1}, Z_{n+2}, \ldots$, become available. Note that when $l_{\max}$ or more further observations are available then g13dj must be used to produce new forecasts for $Z_{n+l_{\max}+1}, Z_{n+l_{\max}+2}, \ldots$, should they be required.

When a transformation has been used the forecasts and their standard errors are suitably modified to give results in terms of the original series, Z_t; see Granger and Newbold 1976.

4 References

Box G E P and Jenkins G M 1976 *Time Series Analysis: Forecasting and Control* (Revised Edition) Holden–Day

Granger C W J and Newbold P 1976 Forecasting transformed series *J. Roy. Statist. Soc. Ser. B* **38** 189–203

Wei W W S 1990 *Time Series Analysis: Univariate and Multivariate Methods* Addison–Wesley

5 Parameters

The output quantities **k**, **n**, **kmax**, **ip**, **iq**, **par**, **npar**, **qq** and **v** from g13dc are suitable for input to g13dj.

5.1 Compulsory Input Parameters

1: **z(kmax,n) – double array**

kmax, the first dimension of the array, must be at least **k**.

$\mathbf{z}(i,t)$ must contain, z_{it}, the ith component of Z_t, for $i = 1, 2, \ldots, k$ and $t = 1, 2, \ldots, n$.

Constraints:

if $\mathbf{tr}(i) =$ 'L', $\mathbf{z}(i,t) > 0.0$;
if $\mathbf{tr}(i) =$ 'S', $\mathbf{z}(i,t) \geq 0.0$, for $i = 1, 2, \ldots, k$ and $t = 1, 2, \ldots, n$.

2: **tr(k) – string array**

$\mathbf{tr}(i)$ indicates whether the ith time series is to be transformed, for $i = 1, 2, \ldots, k$.

$\mathbf{tr}(i) =$ 'N'

No transformation is used.

$\mathbf{tr}(i) =$ 'L'

A log transformation is used.

$\mathbf{tr}(i) =$ 'S'

A square root transformation is used.

Constraint: $\mathbf{tr}(i) =$ 'N', 'L' or 'S', for $i = 1, 2, \ldots, k$.

3: **id(k) – int32 array**

id(i) must specify, d_i, the order of differencing required for the ith series.

Constraint: $0 \leq$ **id**$(i) <$ **n** $-$ max(**ip**, **iq**), for $i = 1, 2, \ldots, k$.

4: **delta(kmax,∗) – double array**

The first dimension of the array **delta** must be at least **k**

The second dimension of the array must be at least $\max(1, d)$

If **id**$(i) > 0$, then **delta**(i,j) must be set equal to δ_{ij}, for $j = 1, 2, \ldots, d_i$ and $i = 1, 2, \ldots, k$.

If $d = 0$, **delta** is not referenced.

5: **ip – int32 scalar**

p, the number of AR parameter matrices.

Constraint: **ip** ≥ 0.

6: **iq – int32 scalar**

q, the number of MA parameter matrices.

Constraint: **iq** ≥ 0.

7: **mean – string**

mean = 'M', if components of μ have been estimated and **mean** = 'Z', if all elements of μ are to be taken as zero.

Constraint: **mean** = 'M' or 'Z'.

8: **par(lpar) – double array**

Must contain the parameter estimates read in row by row in the order $\phi_1, \phi_2, \ldots, \phi_p$, $\theta_1, \theta_2, \ldots, \theta_q$, μ.

Thus,

if **ip** > 0, **par**$((l-1) \times k \times k + (i-1) \times k + j)$ must be set equal to an estimate of the (i,j)th element of ϕ_l, for $l = 1, 2, \ldots, p$, $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, k$;
if **iq** > 0, **par**$(p \times k \times k + (l-1) \times k \times k + (i-1) \times k + j)$ must be set equal to an estimate of the (i,j)th element of θ_l, for $l = 1, 2, \ldots, q$, $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, k$;
if **mean** = 'M', **par**$((p+q) \times k \times k + i)$ must be set equal to an estimate of the ith component of μ, for $i = 1, 2, \ldots, k$.

Constraint: the first **ip** $\times$ **k** $\times$ **k** elements of **par** must satisfy the stationarity condition and the next **iq** $\times$ **k** $\times$ **k** elements of **par** must satisfy the invertibility condition.

9: **qq(kmax,k) – double array**

kmax, the first dimension of the array, must be at least **k**.

qq(i,j) must contain an estimate of the (i,j)th element of Σ. The lower triangle only is needed.

Constraint: **qq** must be positive-definite.

10: **v(kmax,∗) – double array**

The first dimension of the array **v** must be at least **k**

The second dimension of the array must be at least $\max(1, \mathbf{n} - d)$

v(i,t) must contain an estimate of the ith component of ϵ_{t+d}, for $i = 1, 2, \ldots, k$ and $t = 1, 2, \ldots, n-d$, where d is the maximum value of the elements of the array **id**. If $q = 0$, **v** is not used.

11: **lmax – int32 scalar**

the number, $l_{\max}$, of forecasts required.

Constraint: **lmax** ≥ 1.

12: **lref – int32 scalar**

Constraint: **lref** $\geq (\mathbf{lmax}-1)\times\mathbf{k}\times\mathbf{k}+2\times\mathbf{k}\times\mathbf{lmax}+\mathbf{k}$.

5.2 Optional Input Parameters

1: **k – int32 scalar**

k, the dimension of the multivariate time series.

Constraint: **k** ≥ 1.

2: **n – int32 scalar**

Default: The dimension of the array **z**.

n, the number of observations in the series, Z_t, prior to differencing.

Constraint: **n** ≥ 3.

The total number of observations must exceed the total number of parameters in the model; that is

if **mean** = 'Z', $\mathbf{n}\times\mathbf{k} > (\mathbf{ip}+\mathbf{iq})\times\mathbf{k}\times\mathbf{k}+\mathbf{k}\times(\mathbf{k}+1)/2$;
if **mean** = 'M', $\mathbf{n}\times\mathbf{k} > (\mathbf{ip}+\mathbf{iq})\times\mathbf{k}\times\mathbf{k}+\mathbf{k}+\mathbf{k}\times(\mathbf{k}+1)/2$,

(see the parameters **ip**, **iq** and **mean**).

3: **lpar – int32 scalar**

Default: The dimension of the array **par**.

Constraints:

if **mean** = 'Z', **lpar** $\geq \max(1,(\mathbf{ip}+\mathbf{iq})\times\mathbf{k}\times\mathbf{k})$;
if **mean** = 'M', **lpar** $\geq (\mathbf{ip}+\mathbf{iq})\times\mathbf{k}\times\mathbf{k}+\mathbf{k}$.

5.3 Input Parameters Omitted from the MATLAB Interface

kmax, work, lwork, iwork, liwork

5.4 Output Parameters

1: **qq(kmax,k) – double array**

If **ifail** $\neq$ g13djIFneq1, then the upper triangle is set equal to the lower triangle.

2: **predz(kmax,lmax) – double array**

predz(i,l) contains the forecast of $z_{i,n+l}$, for $i=1,2,\ldots,k$ and $l=1,2,\ldots,l_{\max}$.

3: **sefz(kmax,lmax) – double array**

sefz(i,l) contains an estimate of the standard error of the forecast of $z_{i,n+l}$, for $i=1,2,\ldots,k$ and $l=1,2,\ldots,l_{\max}$.

4: **ref(lref) – double array**

The reference vector which may be used to update forecasts using g13dk. The first $(\mathbf{lmax}-1)\times\mathbf{k}\times\mathbf{k}$ elements contain the ψ weight matrices, $\psi_1,\psi_2,\ldots,\psi_{l_{\max}-1}$. The next $\mathbf{k}\times\mathbf{lmax}$ elements contain the forecasts of the transformed series $\hat{Z}^*_{n+1},\hat{Z}^*_{n+2},\ldots,\hat{Z}^*_{n+l_{\max}}$ and the

next $\mathbf{k} \times \mathbf{lmax}$ contain the variances of the forecasts of the transformed variables. The last **k** elements are used to store the transformations for the series.

5: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, $\mathbf{k} < 1$,
or $\mathbf{n} < 3$,
or $\mathbf{kmax} < \mathbf{k}$,
or $\mathbf{id}(i) < 0$ for some $i = 1, 2, \ldots, k$,
or $\mathbf{id}(i) \geq \mathbf{n} - \max(\mathbf{ip}, \mathbf{iq})$ for some $i = 1, 2, \ldots, k$,
or $\mathbf{ip} < 0$,
or $\mathbf{iq} < 0$,
or **mean** $\neq$ 'M' or 'Z',
or $\mathbf{lpar} < (\mathbf{ip} + \mathbf{iq}) \times \mathbf{k} \times \mathbf{k} + \mathbf{k}$, and **mean** $=$ 'M',
or $\mathbf{lpar} < (\mathbf{ip} + \mathbf{iq}) \times \mathbf{k} \times \mathbf{k}$ and **mean** $=$ 'Z',
or $\mathbf{n} \times \mathbf{k} \leq (\mathbf{ip} + \mathbf{iq}) \times \mathbf{k} \times \mathbf{k} + \mathbf{k} + \mathbf{k}(\mathbf{k} + 1)/2$, and **mean** $=$ 'M',
or $\mathbf{n} \times \mathbf{k} \leq (\mathbf{ip} + \mathbf{iq}) \times \mathbf{k} \times \mathbf{k} + \mathbf{k}(\mathbf{k} + 1)/2$ and **mean** $=$ 'Z',
or $\mathbf{lmax} < 1$,
or $\mathbf{lref} < (\mathbf{lmax} - 1) \times \mathbf{k} \times \mathbf{k} + 2 \times \mathbf{k} \times \mathbf{lmax} + \mathbf{k}$,
or **lwork** is too small,
or **liwork** is too small.

ifail $= 2$

On entry, at least one of the first k elements of **tr** is not equal to 'N', 'L' or 'S'.

ifail $= 3$

On entry, one or more of the transformations requested cannot be computed; that is, you may be trying to log or square-root a series, some of whose values are negative.

ifail $= 4$

On entry, either **qq** is not positive-definite or the autoregressive parameter matrices are extremely close to or outside the stationarity region, or the moving average parameter matrices are extremely close to or outside the invertibility region. To proceed, you must supply different parameter estimates in the arrays **par** and **qq**.

ifail $= 5$

This is an unlikely exit brought about by an excessive number of iterations being needed to evaluate the eigenvalues of the matrices required to check for stationarity and invertibility; see g13dx. All output parameters are undefined.

ifail $= 6$

This is an unlikely exit which could occur if **qq** is nearly non positive-definite. In this case the standard deviations of the forecast errors may be nonpositive. To proceed, you must supply different parameter estimates in the array **qq**.

ifail $= 7$

This is an unlikely exit. For one of the series, overflow will occur if the forecasts are computed. You should check whether the transformations requested in the array **tr** are sensible. All output parameters are undefined.

7 Accuracy

The matrix computations are believed to be stable.

8 Further Comments

The same differencing operator does not have to be applied to all the series. For example, suppose we have $k = 2$, and wish to apply the second order differencing operator ∇^2 to the first series and the first-order differencing operator ∇ to the second series:

$$w_{1t} = \nabla^2 z_{1t} = (1-B)^2 z_{1t} = \left(1 - 2B + B^2\right) Z_{1t}, \qquad \text{and}$$
$$w_{2t} = \nabla z_{2t} = (1-B) z_{2t}.$$

Then $d_1 = 2, d_2 = 1,\ d = \max(d_1, d_2) = 2,$ and

$$\textbf{delta} = \begin{bmatrix} \delta_{11} & \delta_{12} \\ \delta_{21} & \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ 1 & \end{bmatrix}.$$

Note: although differencing may already have been applied prior to the model fitting stage, the differencing parameters supplied in **delta** are part of the model definition and are still required by this function to produce the forecasts.

g13dj should not be used when the moving average parameters lie close to the boundary of the invertibility region. The function does test for both invertibility and stationarity but if in doubt, you may use g13dx, before calling this function, to check that the VARMA model being used is invertible.

On a successful exit, the output quantities **k**, **lmax**, **kmax**, **ref** and **lref** will be suitable for input to g13dk.

9 Example

```
z = [-1.49, -1.62, 5.2, 6.23, 6.21, 5.86, 4.09, 3.18, 2.62, 1.49, 1.17,
...
    0.85, -0.35, 0.24, 2.44, 2.58, 2.04, 0.4, 2.26, 3.34, 5.09, 5, 4.78,
...
     4.11, 3.45, 1.65, 1.29, 4.09, 6.32, 7.5, 3.89, 1.58, 5.21, 5.25,
4.93, ...
    7.38, 5.87, 5.81, 9.68, 9.07, 7.29, 7.84, 7.55, 7.32, 7.97, 7.76, 7,
8.35;
     7.34, 6.35, 6.96, 8.539999999999999, 6.62, 4.97, 4.55, 4.81, 4.75,
...
    4.76, 10.88, 10.01, 11.62, 10.36, 6.4, 6.24, 7.93, 4.04, 3.73, 5.6,
...
    5.35, 6.81, 8.27, 7.68, 6.65, 6.08, 10.25, 9.140000000000001, 17.75,
13.3, ...
    9.630000000000001, 6.8, 4.08, 5.06, 4.94, 6.65, 7.94, 10.76, 11.89,
...
      5.85, 9.01, 7.5, 10.02, 10.38, 8.15, 8.369999999999999, 10.73,
12.14];
tr = {'N'; 'N'};
id = [int32(0);
     int32(0)];
delta = [0;
     0];
ip = int32(1);
iq = int32(0);
mean = 'M';
par = [0.8016071892386086;
```

```
     0.0648134906597352;
     0;
     0.575015951133362;
     4.271122828253269;
     7.825342792089621];
qq = [2.964154253391392, 0.6372583252520638;
     0.6372583252520638, 5.379903126133676];
v    =    [-3.326133715726029,    -1.241508590516912,    5.746865699837245,
1.270368439927496, ...
         0.3223077197693467,  0.112781765620811,  -1.269713458557113,  -
0.7336470675276869, ...
        -0.5810360328920845,  -1.25824719747888,  -0.673079208545849,  -
1.133223470827074, ...
         -2.02032143339675,  -0.572742526272592,  1.235974230307894,  -
0.1309001630044928, ...
          -0.7727550109923405,  -2.089421928018445,  1.337338340999243,
0.9464411511199494, ...
       1.709504159208547,  0.2328949507059163,  -0.009588098625822217,  -
0.5978622133565419, ...
     -0.6825454370774304, -1.886726796800422, -0.7668901664948766, ...
          2.051416165579926,  2.108859010344129,  0.9432308237617111,  -
3.324245626104025, ...
           -2.50257816223142,   3.162556623476816,   0.4690152211351468,
0.05343371271906172, ...
          2.767725632154585,  -0.8170430505081534,  0.2497744022910866,
3.985096789984951, ...
          0.1996377231860347,  -0.6999084077936142,  1.072141758566346,
0.4391261753813116, ...
         0.2782622637979761,  1.08929906068535,  0.5027884718514648,  -
0.103132986353569, 1.703128639510798;
         -0.1864812965451277,  -1.196262944870486,  -0.01699715324845752,
1.212243116560191, ...
         -1.61628208623052,  -2.162251460054466,  -1.633475140684418,  -
1.131968441208406, ...
           -1.34147258850308,   -1.296971631435079,   4.817278209053589,
0.4281805881174119, ...
         2.538444465603437,  0.3526687842787246,  -2.882811117293238,  -
0.7657479508051256, ...
           1.016254601376212,   -3.84552235603917,   -1.918710306130392,
0.1295446387209505, ...
         -1.195735189898436,  0.4080187978849044,  1.028495509230196,  -
0.4010277794245127, ...
          -1.091768368255828,  -1.069501938588466,  3.42825715355755,  -
0.07955936266856867, ...
        9.168708343089461,  -0.2321789961687832,  -1.343358013625323,  -
2.063049472965885, ...
          -3.15575433125847,   -0.6117109441757265,   -1.29522657628642,
0.4837753378495833, ...
         0.7904980614115339,  2.868727484449496,  2.377182502253417,  -
4.312585522527284, ...
           2.320510822318223,   -1.0065395832632,   2.381734502948175,
1.292694306092105, ...
          -1.144311436315907,   0.3579741347114898,   2.591470625462152,
2.644432980787418];
lmax = int32(5);
lref = int32(150);
[qqOut, predz, sefz, ref, ifail] = ...
    g13dj(z, tr, id, delta, ip, iq, mean, par, qq, v, lmax, lref)
```

```
qqOut =
    2.9642    0.6373
    0.6373    5.3799
predz =
    7.8204    7.2771    6.7732    6.3300    5.9521
   10.3063    9.2520    8.6457    8.2970    8.0966
sefz =
    1.7217    2.2266    2.5095    2.6817    2.7898
    2.3195    2.6756    2.7833    2.8180    2.8294
ref =
     array elided
```

```
ifail =
           0
```