NAG Toolbox

nag_tsa_multi_kalman_sqrt_var (g13ea)

1 Purpose

nag_tsa_multi_kalman_sqrt_var (g13ea) performs a combined measurement and time update of one iteration of the time-varying Kalman filter using a square root covariance filter.

2 Syntax

```
[s, k, h, ifail] = nag_tsa_multi_kalman_sqrt_var(a, b, stq, q, c, r, s, 'n', n, 'm', m, 'l', l, 'tol', tol)
[s, k, h, ifail] = g13ea(a, b, stq, q, c, r, s, 'n', n, 'm', m, 'l', l, 'tol', tol)
```

3 Description

The Kalman filter arises from the state space model given by:

$$X_{i+1} = A_i X_i + B_i W_i, \quad \mathrm{Var}(W_i) = Q_i$$

$$Y_i = C_i X_i + V_i, \qquad \mathrm{Var}(V_i) = R_i$$

where X_i is the state vector of length n at time i, Y_i is the observation vector of length m at time i, and W_i of length l and V_i of length m are the independent state noise and measurement noise respectively.

The estimate of X_i given observations Y_1 to Y_{i-1} is denoted by $\hat{X}_{i|i-1}$ with state covariance matrix $\mathrm{Var}\big(\hat{X}_{i|i-1}\big) = P_{i|i-1} = S_i S_i^{\mathrm{T}}$, while the estimate of X_i given observations Y_1 to Y_i is denoted by $\hat{X}_{i|i}$ with covariance matrix $\mathrm{Var}\big(\hat{X}_{i|i}\big) = P_{i|i}$. The update of the estimate, $\hat{X}_{i|i-1}$, from time i to time $(i+1)$, is computed in two stages. First, the measurement-update is given by

$$\hat{X}_{i|i} = \hat{X}_{i|i-1} + K_i\big[Y_i - C_i\hat{X}_{i|i-1}\big] \tag{1}$$

and

$$P_{i|i} = [I - K_i C_i]P_{i|i-1} \tag{2}$$

where $K_i = P_{i|i-1}C_i^{\mathrm{T}}\big[C_i P_{i|i-1} C_i^{\mathrm{T}} + R_i\big]^{-1}$ is the Kalman gain matrix. The second stage is the time-update for X which is given by

$$\hat{X}_{i+1|i} = A_i\hat{X}_{i|i} + D_i U_i \tag{3}$$

and

$$P_{i+1|i} = A_i P_{i|i} A_i^{\mathrm{T}} + B_i Q_i B_i^{\mathrm{T}} \tag{4}$$

where $D_i U_i$ represents any deterministic control used.

The square root covariance filter algorithm provides a stable method for computing the Kalman gain matrix and the state covariance matrix. The algorithm can be summarised as

$$\begin{pmatrix} R_i^{1/2} & C_i S_i & 0 \\ 0 & A_i S_i & B_i Q_i^{1/2} \end{pmatrix} U = \begin{pmatrix} H_i^{1/2} & 0 & 0 \\ G_i & S_{i+1} & 0 \end{pmatrix} \tag{5}$$

where U is an orthogonal transformation triangularizing the left-hand pre-array to produce the right-hand post-array. The relationship between the Kalman gain matrix, K_i, and G_i is given by

$$A_iK_i = G_i\left(H_i^{1/2}\right)^{-1}.$$

nag_tsa_multi_kalman_sqrt_var (g13ea) requires the input of the lower triangular Cholesky factors of the noise covariance matrices $R_i^{1/2}$ and, optionally, $Q_i^{1/2}$ and the lower triangular Cholesky factor of the current state covariance matrix, S_i, and returns the product of the matrices A_i and K_i, A_iK_i, the Cholesky factor of the updated state covariance matrix S_{i+1} and the matrix $H_i^{1/2}$ used in the computation of the likelihood for the model.

4 References

Vanbegin M, van Dooren P and Verhaegen M H G (1989) Algorithm 675: FORTRAN subroutines for computing the square root covariance filter and square root information filter in dense or Hessenberg forms *ACM Trans. Math. Software* **15** 243–256

Verhaegen M H G and van Dooren P (1986) Numerical aspects of different Kalman filter implementations *IEEE Trans. Auto. Contr.* **AC-31** 907–917

5 Parameters

5.1 Compulsory Input Parameters

1: **a**$(lds, \mathbf{n})$ – REAL (KIND=nag_wp) array

lds, the first dimension of the array, must satisfy the constraint $lds \geq \mathbf{n}$.

The state transition matrix, A_i.

2: **b**$(lds, \mathbf{l})$ – REAL (KIND=nag_wp) array

lds, the first dimension of the array, must satisfy the constraint $lds \geq \mathbf{n}$.

The noise coefficient matrix B_i.

3: **stq** – LOGICAL

If **stq** $= true$, the state noise covariance matrix Q_i is assumed to be the identity matrix. Otherwise the lower triangular Cholesky factor, $Q_i^{1/2}$, must be provided in **q**.

4: **q**$(ldq, :)$ – REAL (KIND=nag_wp) array

The first dimension, ldq, of the array **q** must satisfy

if **stq** $= false$, $ldq \geq \mathbf{l}$;
otherwise $ldq \geq 1$.

The second dimension of the array **q** must be at least **l** if **stq** $= false$ and at least 1 if **stq** $= true$.

If **stq** $= false$, **q** must contain the lower triangular Cholesky factor of the state noise covariance matrix, $Q_i^{1/2}$. Otherwise **q** is not referenced.

5: **c**$(ldm, \mathbf{n})$ – REAL (KIND=nag_wp) array

ldm, the first dimension of the array, must satisfy the constraint $ldm \geq \mathbf{m}$.

The measurement coefficient matrix, C_i.

6: **r**$(ldm, \mathbf{m})$ – REAL (KIND=nag_wp) array

ldm, the first dimension of the array, must satisfy the constraint $ldm \geq \mathbf{m}$.

The lower triangular Cholesky factor of the measurement noise covariance matrix $R_i^{1/2}$.

7: **s**$(lds, \mathbf{n})$ – REAL (KIND=nag_wp) array

lds, the first dimension of the array, must satisfy the constraint $lds \geq \mathbf{n}$.

The lower triangular Cholesky factor of the state covariance matrix, S_i.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the arrays **a**, **b**, **s** and the second dimension of the arrays **a**, **c**, **s**. (An error is raised if these dimensions are not equal.)

n, the size of the state vector.

Constraint: $\mathbf{n} \geq 1$.

2: **m** – INTEGER

Default: the first dimension of the arrays **c**, **r** and the second dimension of the array **r**. (An error is raised if these dimensions are not equal.)

m, the size of the observation vector.

Constraint: $\mathbf{m} \geq 1$.

3: **l** – INTEGER

Default: the second dimension of the array **b**.

l, the dimension of the state noise.

Constraint: $\mathbf{l} \geq 1$.

4: **tol** – REAL (KIND=nag_wp)

Suggested value: $\mathbf{tol} = 0.0$.

Default: 0.0

The tolerance used to test for the singularity of $H_i^{1/2}$. If $0.0 \leq \mathbf{tol} < m^2 \times$ ***machine precision***, then $m^2 \times$ ***machine precision*** is used instead. The inverse of the condition number of $H^{1/2}$ is estimated by a call to nag_lapack_dtrcon (f07tg). If this estimate is less than **tol** then $H^{1/2}$ is assumed to be singular.

Constraint: $\mathbf{tol} \geq 0.0$.

5.3 Output Parameters

1: **s**$(lds, \mathbf{n})$ – REAL (KIND=nag_wp) array

The lower triangular Cholesky factor of the state covariance matrix, S_{i+1}.

2: **k**$(lds, \mathbf{m})$ – REAL (KIND=nag_wp) array

The Kalman gain matrix, K_i, premultiplied by the state transition matrix, A_i, A_iK_i.

3: **h**$(ldm, \mathbf{m})$ – REAL (KIND=nag_wp) array

The lower triangular matrix $H_i^{1/2}$.

4: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **n** < 1,
or **m** < 1,
or **l** < 1,
or $lds <$ **n**,
or $ldm <$ **m**,
or **stq** $= true$ and $ldq < 1$,
or **stq** $= false$ and $ldq <$ **l**,
or **tol** < 0.0.

ifail $= 2$

The matrix $H_i^{1/2}$ is singular.

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The use of the square root algorithm improves the stability of the computations as compared with the direct coding of the Kalman filter. The accuracy will depend on the model.

8 Further Comments

For models with time-invariant A, B and C, nag_tsa_multi_kalman_sqrt_invar (g13eb) can be used.

If W_i and V_i are independent multivariate Normal variates then the log-likelihood for observations $i = 1, 2, \ldots, t$ is given by

$$l(\theta) = \kappa - \tfrac{1}{2}\sum_{i=1}^{t} ln(\det(H_i)) - \tfrac{1}{2}\sum_{i=1}^{t}\left(Y_i - C_i X_{i|i-1}\right)^{\mathrm{T}} H_i^{-1}\left(Y_i - C_i X_{i|i-1}\right)$$

where κ is a constant.

The Cholesky factors of the covariance matrices can be computed using nag_lapack_dpotrf (f07fd).

Note that the model

$$X_{i+1} = A_i X_i + W_i, \quad \mathrm{Var}(W_i) = Q_i$$

$$Y_i = C_i X_i + V_i, \qquad \mathrm{Var}(V_i) = R_i$$

can be specified either with **b** set to the identity matrix and **stq** $= false$ and the matrix $Q^{1/2}$ input in **q** or with **stq** $= true$ and **b** set to $Q^{1/2}$.

The algorithm requires $\frac{7}{6}n^3 + n^2\left(\frac{5}{2}m + l\right) + n\left(\frac{1}{2}l^2 + m^2\right)$ operations and is backward stable (see Verhaegen and van Dooren (1986)).

9 Example

This example first inputs the number of updates to be computed and the problem sizes. The initial state vector and state covariance matrix are input followed by the model matrices A_i, B_i, C_i, R_i and optionally Q_i. The Cholesky factors of the covariance matrices can be computed if required. The model matrices can be input at each update or only once at the first step. At each update the observed values are input and the residuals are computed and printed and the estimate of the state vector, $\hat{X}_{i|i-1}$, and the deviance are updated. The deviance is $-2 \times$ log-likelihood ignoring the constant. After the final update the state covariance matrix is computed from **s** and printed along with final estimate of the state vector and the value of the deviance.

The data is for a two-dimensional time series to which a VARMA$(1,1)$ has been fitted. For the specification of a VARMA model as a state space model see the G13 Chapter Introduction. The initial value of P, P_0, is the solution to

$$P_0 = A_1 P_0 A_1^{\mathrm{T}} + B_1 Q_1 B_1^{\mathrm{T}}.$$

For convenience, the mean of each series is input before the first update and subtracted from the observations before the measurement update is computed.

9.1 Program Text

```
     function g13ea_example

fprintf('g13ea example results\n\n');

% Constant matrices
s = [ 8.2068  2.0599  1.4807  0.3627;
      2.0599  7.9645  0.9703  0.2136;
      1.4807  0.9703  0.9253  0.2236;
      0.3627  0.2136  0.2236  0.0542];
a = [ 0.607, -0.033,  1,      0;
      0,      0.543,  0,      1;
      0,      0,      0,      0;
      0,      0,      0,      0];
b = [ 1,      0;
      0,      1;
      0.543,  0.125;
      0.134,  0.026];
stq = false;
q = [ 2.598,  0.56;
      0.560,  5.33];
c = [ 1,      0,      0,      0;
      0,      1,      0,      0];
r = [ 0,      0;
      0,      0];
% Need to factorize S and Q
s = chol(s,'lower');
q = chol(q,'lower');

n = size(s,1);
m = 2;

% Data
x = zeros(n,1);
ym = [-1.49  7.34; -1.62  6.35;  5.20  6.96;  6.23  8.54;  6.21  6.62;
       5.86  4.97;  4.09  4.55;  3.18  4.81;  2.62  4.75;  1.49  4.76;
       1.17 10.88;  0.85 10.01; -0.35 11.62;  0.24 10.36;  2.44  6.40;
       2.58  6.24;  2.04  7.93;  0.40  4.04;  2.26  3.73;  3.34  5.60;
       5.09  5.35;  5.00  6.81;  4.78  8.27;  4.11  7.68;  3.45  6.65;
       1.65  6.08;  1.29 10.25;  4.09  9.14;  6.32 17.75;  7.50 13.30;
       3.89  9.63;  1.58  6.80;  5.21  4.08;  5.25  5.06;  4.93  4.94;
       7.38  6.65;  5.87  7.94;  5.81 10.76;  9.68 11.89;  9.07  5.85;
       7.29  9.01;  7.84  7.50;  7.55 10.02;  7.32 10.38;  7.97  8.15;
       7.76  8.37;  7.00 10.73;  8.35 12.14];
ymean = [4.404; 7.991];
ny = size(ym,1);
```

```
% Loop over data
fprintf('      Residuals\n\n');
dev = 0;
for j = 1:ny
  y = ym(j,:)';
  [s, K, H, ifail] = g13ea( ...
                            a, b, stq, q, c, r, s);

  % subtract the mean
  y = y - ymean;
  % Time and measurement update
  y = y-c*x;
  x = a*x + K*y;

  fprintf('%10.4f%10.4f\n',y);

  % Update Loglikelihood
  y = H\y;
  dev = dev + dot(y,y);
  for i = 1:m
    dev = dev + 2*log(H(i,i));
  end

end

% P from S
p = s*s';

% Final results

fprintf('\nFinal x\n\n')
disp (x');

[ifail] = x04ca(...
                'Lower','N', p, 'Final Value of P');

fprintf('\nDeviance = %13.4e\n', dev);
```

9.2 Program Results

```
     g13ea example results

      Residuals

   -5.8940   -0.6510
   -1.4710   -1.0407
    5.1658    0.0447
   -1.3280    0.4580
    1.3652   -1.5066
   -0.2337   -2.4192
   -0.8685   -1.7065
   -0.4624   -1.1519
   -0.7510   -1.4218
   -1.3526   -1.3335
   -0.6707    4.8593
   -1.7389    0.4138
   -1.6376    2.7549
   -0.6137    0.5463
    0.9067   -2.8093
   -0.8255   -0.9355
   -0.7494    1.0247
   -2.2922   -3.8441
    1.8812   -1.7085
   -0.7112   -0.2849
    1.6747   -1.2400
   -0.6619    0.0609
    0.3271    1.0074
   -0.8165   -0.5325
   -0.2759   -1.0489
   -1.9383   -1.1186
```

```
   -0.3131    3.5855
    1.3726   -0.1289
    1.4153    8.9545
    0.3672   -0.4126
   -2.3659   -1.2823
   -1.0130   -1.7306
    3.2472   -3.0836
   -1.1501   -1.1623
    0.6855   -1.2751
    2.3432    0.2570
   -1.6892    0.3565
    1.3871    3.0138
    3.3840    2.1312
   -0.5118   -4.7670
    0.8569    2.3741
    0.9558   -1.2209
    0.6778    2.1993
    0.4304    1.1393
    1.4987   -1.2255
    0.5361    0.1237
    0.2649    2.4582
    2.0095    2.5623

Final x

    3.6698    2.5888         0         0

 Final Value of P
             1          2          3          4
 1      2.5980
 2      0.5600     5.3300
 3      1.4807     0.9703     0.9253
 4      0.3627     0.2136     0.2236     0.0542

Deviance =    2.2287e+02
```