NAG C Library Function Document

nag_tsa_multi_inp_model_forecast (g13bjc)

1 Purpose

nag_tsa_multi_inp_model_forecast (g13bjc) produces forecasts of a time series (the output series) which may depend on one or more other (input) series via a previously estimated multi-input model. The future values of any input series must be supplied. Standard errors of the forecasts are produced. If future values of some of the input series have been obtained as forecasts using ARIMA models for those series, this may be allowed for in the calculation of the standard errors.

2 Specification

```c
#include <nag.h>
#include <nagl3.h>

void nag_tsa_multi_inp_model_forecast(Nag_ArimaOrder *arimav,
    Integer nseries, Nag_TransfOrder *transf, double para[],
    Integer npara, Integer nev, Integer nfv, double xxy[],
    Integer txdxy, double rmsxy[], Integer mxr, Integer tdmrx,
    double parx[], Integer 1dpdx, Integer tdpdx, double fva[],
    double fsd[], Nag_G13_Opt *options, NagError *fail)
```

3 Description

The function has two stages. The first stage is essentially the same as a call to the model estimation function nag_tsa_multi_inp_model_estim (g13bec), with zero iterations. In particular, all the parameters remain unchanged in the supplied input series transfer function models and output noise series ARIMA model except that a further iteration takes place for any \( \omega \) corresponding to a simple input. The internal nuisance parameters associated with the pre-observation period effects of the input series are estimated where requested, and so are any backforecasts of the output noise series. The output components \( z_t \) and \( n_t \), and residuals \( a_t \) are calculated exactly as described in the Section 3 of nag_tsa_multi_inp_model_estim (g13bec).

In the second stage, the forecasts of the output series \( y_t \) are calculated for \( t = n + 1, n + 2, \ldots, n + L \) where \( n \) is the latest time point of the observations and \( L \) is the maximum lead time of the forecasts.

First the new values, \( x_t \) for any input series are used to form the input components \( z_t \) for \( t = n + 1, n + 2, \ldots, n + L \) using the transfer function models:

- \( z_t = \delta_1 z_{t-1} + \delta_2 z_{t-2} + \ldots + \delta_p z_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - \ldots - \omega_q x_{t-b-q} \).

  The output noise component \( n_t \) for \( t = n + 1, n + 2, \ldots, n + L \) is then forecast by setting \( a_t = 0 \) for \( t = n + 1, n + 2, \ldots, n + L \) and using the ARIMA model equations:

- \( e_t = \phi_1 e_{t-1} + \phi_2 e_{t-2} + \ldots + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \ldots - \theta_q a_{t-q} \).
- \( w_t = \Phi_1 w_{t-s} + \Phi_2 w_{t-2s} + \ldots + \Phi_p w_{t-ps} + \psi_1^2 e_{t-s} - \Theta_1 e_{t-s} - \Theta_2 e_{t-2s} - \ldots - \Theta_q e_{t-q} \).
- \( n_t = (\nabla^d \nabla^p)^{-1}(w_t + c) \).

This last step of ‘integration’ reverses the process of differencing. Finally the output forecasts are calculated as

\( y_t = z_{1,t} + z_{2,t} + \ldots + z_{m,t} + n_t \).

The forecast error variance of \( y_{t+l} \) (i.e., at lead time \( l \)) is \( \sigma^2_l \), which is the sum of parts which arise from the various input series, and the output noise component. That part due to the output noise is

\( sn^2_l = V_n \times (\psi_0^2 + \psi_1^2 + \ldots + \psi_{l-1}^2) \)

\( V_n \) is the estimated residual variance of the output noise ARIMA model, and \( \psi_0, \psi_1, \ldots \), are the
'psi-weights' of this model as defined in Box and Jenkins (1976). They are calculated by applying the equations (b), (c) and (d) above for \( t = 0, 1, \ldots, L \), but with artificial values for the various series and with the constant \( c \) set to 0. Thus all values of \( a_t, e_t, w_t \) and \( n_t \) are taken as zero for \( t < 0 \); \( a_t \) is taken to be 1 for \( t = 0 \) and 0 for \( t > 0 \). The resulting values of \( n_t \) for \( t = 0, 1, \ldots, L \) are precisely \( \psi_0, \psi_1, \ldots, \psi_L \) as required.

Further contributions to \( S^2 \) come only from those input series, for which future values are forecasts which have been obtained by applying input series ARIMA models. For such a series the contribution is

\[
S^2 = V_x \times (\nu_0^2 + \nu_1^2 + \ldots + \nu_{l-1}^2)
\]

\( V_x \) is the estimated residual variance of the input series ARIMA model. The coefficients \( \nu_0, \nu_1, \ldots \) are calculated by applying the transfer function model equation (a) above for \( t = 0, 1, \ldots, L \), but again with artificial values of the series. Thus all values of \( z_t \) and \( x_t \) for \( t < 0 \) are taken to be zero, and \( x_0, x_1, \ldots \) are taken to be the psi-weight sequence \( \psi_0, \psi_1, \ldots \) for the input series ARIMA model. The resulting values of \( z_t \) for \( t = 0, 1, \ldots, L \) are precisely \( \nu_0, \nu_1, \ldots, \nu_L \) as required.

In adding such contributions \( S^2 \) to \( S^2 \) to make up the total forecast error variance \( S^2 \), it is assumed that the various input series with which these contributions are associated, are statistically independent of each other.

When using the routine in practice an ARIMA model is required for all the input series. In the case of those inputs for which no such ARIMA model is available (or its effects are to be excluded), the corresponding orders and parameters and the estimated residual variance should be set to zero.

4 Parameters

1: `arimav` = Nag_ArimaOrder *

Pointer to structure of type Nag_ArimaOrder with the following members:

- `p` – Integer
- `d` – Integer
- `q` – Integer
- `bigp` – Integer
- `bigd` – Integer
- `bigq` – Integer
- `s` – Integer

These seven members of `arimav` must specify the orders vector \( (p, d, q, P, D, Q, s) \), respectively, of the ARIMA model for the output noise component.

\( p, q, P \) and \( Q \) refer, respectively, to the number of autoregressive (\( \phi \)), moving average (\( \theta \)), seasonal autoregressive (\( \Phi \)) and seasonal moving average (\( \Theta \)) parameters.

\( d, D \) and \( s \) refer, respectively, to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

2: `nseries` = Integer

On entry: the number of input and output series. There may be any number of input series (including none), but only one output series.

Constraints: \( nseries > 1 \) if there are no parameters in the model (that is \( p = q = P = Q = 0 \) and `options.cfixed = TRUE`), \( nseries \geq 1 \) otherwise.

3: `transf` = Nag_TransfOrder *

Pointer to structure of type Nag_TransfOrder with the following members:
Before use these member pointers must be allocated memory by calling nag_tsa_transf_orders (g13byc) which allocates nseries − 1 elements to each pointer. The memory allocated to these pointers must be given the transfer function model orders b, q and p of each of the input series. The order parameters for input series r are held in the ith element of the allocated memory for each pointer. b[i − 1] holds the value bi, q[i − 1] holds the value qj, and p[i − 1] holds the value pij.

For a simple input, bi = qi = pij = 0.

r[i − 1] holds the value ri, where ri = 1 for a simple input, and ri = 2 or 3 for a transfer function input.

The choice ri = 3 leads to estimation of the pre-period input effects as nuisance parameters, and ri = 2 suppresses this estimation. This choice may affect the returned forecasts.

When ri = 1, any non-zero contents of the ith element of the memory of b, q and p are ignored.

Constraint: r[i − 1] = 1, 2 or 3, for i = 1, 2, . . . , nseries − 1. The memory allocated to the members of transf must be freed by a call to nag_tsa_trans_free (g13bzc).

4: para[np] – double

On entry: estimates of the multi-input model parameters. These are in order firstly the ARIMA model parameters: p values of φ parameters, q values of θ parameters, P values of Φ parameters, Q values of Θ parameters. These are followed by the transfer function model parameter values ω0, ω1, . . . , ωq, and δ1, δ2, . . . , δp, for the first of any input series and similarly for each subsequent input series. The final component of para is the value of the constant c.

On exit: the input estimates are unaltered except that any ω estimates corresponding to a simple input will be undated by a single iteration.

5: np – Integer

On entry: the exact number of φ, θ, Φ, Θ, ω, δ, c parameters, so that np = p + q + P + Q + nseries + Σ (pi + qi), the summation being over all the input series. (c must be included whether its value was previously estimated or was set fixed.)

6: nev – Integer

On entry: the number of original (undifferenced) values in each of the input and output time-series.

7: nf – Integer

On entry: the number of forecast values of the output series required.

Constraint: nf > 0.

8: xxy[nev+nf][tdxy] – double

On entry: the columns of xxy must contain in the first nev places, the past values of each of the input and output series, in that order. In the next nf places, the columns relating to the input series (i.e., columns 0 to nseries−2) contain the future values of the input series which are necessary for construction of the forecasts of the output series y.

9: tdxy – Integer

On entry: the last dimension of array xxy as declared in the function from which nag_tsa_multi_inpm_model_forecast is called.

Constraint: tdxy ≥ nseries.
rmsxy[nsseries] – double

On entry: elements of rmsxy[0] to rmsxy[nsseries−2] must contain the estimated residual variance of the input series ARIMA models. In the case of those inputs for which no ARIMA model is available or its effects are to be excluded in the calculation of forecast standard errors, the corresponding entry of rmsxy should be set to 0.

On exit: rmsxy[nsseries−1] contains the estimated residual variance of the output noise ARIMA model which is calculated from the supplied series. Otherwise rmsxy is unchanged.

mrx[7][tdmrx] – integer

On entry: the orders array for each of the input series ARIMA models. Thus, column i – 1 contains values of p, q, P, Q, s for input series i. In the case of those inputs for which no ARIMA model is available, the corresponding orders should be set to 0.

If there are no input series then the null pointer (Integer *)0 may be supplied in place of mrx.

dmrx – integer

On entry: the last dimension of array mrx as declared in the function from which nag_tsa_multi_inp_model_forecast is called.

Constraint: dmrx ≥ nsseries−1.

parx[ldparx][tdparx] – double

On entry: values of the parameters (φ, θ, Φ, and Θ) for each of the input series ARIMA models. Thus column i contains mrx[0][i] values of φ, mrx[2][i] values of θ, mrx[3][i] values of Φ and mrx[5][i] values of Θ – in that order.

Values in the columns relating to those input series for which no ARIMA model is available are ignored.

If there are no input series then the null pointer (double *)0 may be supplied in place of parx.

ldparx – integer

On entry: the maximum number of parameters in any of the input series ARIMA models. If there are no input series then ldparx is not referenced.

Constraint: ldparx ≥ nce = max(1,(mrx[0][i]+mrx[2][i]+mrx[3][i]+mrx[5][i])), for i = 0, 1, . . . , nsseries−1.

tdparx – integer

On entry: the last dimension of array parx as declared in the function from which nag_tsa_multi_inp_model_forecast is called.

Constraint: tdparx ≥ nsseries−1.

fva[nfv] – double

On exit: the required forecast values for the output series.

fsd[nfv] – double

On exit: the standard errors for each of the forecast values.

options – Nag_G13_Opt *

On entry/on exit: a pointer to a structure of type Nag_G13_Opt whose members are optional parameters for nag_tsa_multi_inp_model_forecast. If the optional parameters are not required, then the null pointer, G13_DEFAULT, can be used in the function call to nag_tsa_multi_inp_model_forecast. Details of the optional parameters and their types are given below in Section 9.
5 Error Indicators and Warnings

**NE_G13_OPTIONS_NOT_INIT**

On entry, the option structure, `options`, has not been initialised using `nag_tsa_options_init` (g13bxc).

**NE_G13_ORDERS_NOT_INIT**

On entry, the orders array structure `transfv` in function `nag_tsa_transf_orders` (g13byc) has not been initialised.

**NE_INT_ARRAY_2**

Value `<value>` given to `transfv.r[<value>]` not valid. Correct range for elements if `transfv.r` is 1 ≤ r[i] ≤ 3.

**NE_BAD_PARAM**

On entry, parameter `options.cf` had an illegal value.

**NE_INT_ARG_LT**

On entry, `nseries` must not be less than 1: `nseries = <value>`.

**NE_INT_ARG_LE**

On entry, `nfv` must not be less than or equal to 0: `nfv = <value>`.

**NE_2_INT_ARG_LT**

On entry, `tdxx` = `<value>` while `nseries` = `<value>`.
These parameters must satisfy `tdxx ≥ nseries`.

On entry, `tdmx` = `<value>` while `nseries−1 = <value>`.
These parameters must satisfy `tdmx ≥ nseries−1`.

On entry, `ldparx` = `<value>` while `nce` = `<value>`.
These parameters must satisfy `ldparx ≥ nce`. (See the expression for `nce` in Section 4 where `ldparx` is described).

On entry, `tdparx` = `<value>` while `nseries−1 = <value>`.
These parameters must satisfy `tdparx ≥ nseries−1`.

**NE_ALLOC_FAIL**

Memory allocation failed.

**NE_INVALID_NSER**

On entry, `nseries` = 1 and there are no parameters in the model, i.e., ( p = q = P = Q = 0 and `options.cf` = TRUE).

**NE_NSER_INCONSIST**

Value of `nseries` passed to `nag_tsa_transf_orders` (g13byc) was `<value>` which is not equal to the value `<value>` passed in this function.

**NE_NPARA_MR_MT_INCONSIST**

On entry, there is inconsistency between `npara` on the one hand and the elements in the orders structures, `arimav` and `transfv` on the other.
NE_DELTA_TEST_FAILED
On entry, or during execution, one or more sets of \( \delta \) parameters do not satisfy the stationarity or invertibility test conditions.

NE_SOLUTION_FAIL_CONV
Iterative refinement has failed to improve the solution of the equations giving the latest estimates of the parameters. This occurred because the matrix of the set of equations is too ill-conditioned.

NE_MAT_NOT_POS_DEF
Attempt to invert the second derivative matrix needed in the calculation of the covariance matrix of the parameter estimates has failed. The matrix is not positive-definite, possibly due to rounding errors.

NE_ARIMA_TEST_FAILED
On entry, or during execution, one or more sets of the ARIMA \((\phi, \theta, \Phi, \Theta)\) parameters do not satisfy the stationarity or invertibility test conditions.

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.

6 Further Comments
The time taken by the function is approximately proportional to the product of the length of each series and the square of the number of parameters in the multi-input model.

6.1 Accuracy
The computation used is believed to be stable.

6.2 References

7 See Also
nag_tsa_multi_inp_model_estim (g13bec)
nag_tsa_options_init (g13bxc)
nag_tsa_transf_orders (g13byc)
nag_tsa_trans_free (g13bzc)
nag_tsa_free (g13zc)

8 Example
This example illustrates the use of the default option G13_DEFAULT in a call to nag_tsa_multi_inp_model_forecast. An example showing the use of optional parameters is given in Example 2. There is one example program file, the main program of which calls both examples. The main program is given below.

8.1 Example 1
This example illustrates the use of the default option G13_DEFAULT in a call to nag_tsa_multi_inp_model_forecast.
The data in the example relate to 40 observations of an output time series and 5 input time series. This example differs from Example 1 in nag_tsa_multi_inp_model_estim (g13bec) in that there are now 4 simple input series. The output series has one autoregressive ($\phi$) parameter and one seasonal moving average ($\Theta$) parameter. The seasonal period is 4. The transfer function input (the fifth in the set) is defined by orders $b_5 = 1$, $q_5 = 0$, $p_5 = 1$, $r_5 = 3$, so that it allows for pre-observation period effects. The initial values of the specified model are:

$$\phi = 0.495, \ \Theta = 0.238, \omega_1 = -0.367, \ \omega_2 = -3.876, \ \omega_3 = 4.516$$
$$\omega_4 = 2.474, \ \omega_{5,1} = 8.629, \ \delta_1 = 0.688, \ c = -82.858.$$

A further 8 values of the input series are supplied, and it is assumed that the values for the fifth series have themselves been forecast from an ARIMA model with orders 2 0 2 0 1 1 4, in which $\phi_1 = 1.6743$, $\phi_2 = -0.9505$, $\theta_1 = 1.4605$, $\theta_2 = -0.4862$ and $\Theta_1 = 0.8993$, and for which the residual mean square is 0.1720.

The following are computed and printed out: the estimated residual variance for the output noise series, the 8 forecast values and their standard errors.

8.1.1 Program Text

/* nag_tsa_multi_inp_model_forecast(g13bjc) Example Program */
/*
 */

#include <nag.h>
#include <stdio.h>
#include <nag_string.h>
#include <nag_stdf.h>
#include <naggl3.h>

static void ex1(void);
static void ex2(void);

#define NSERMX 6
#define NPMAX 10
#define LDPARX 8
#define NFVMAX 10
#define NEVMAX 40
#define LDXXY NEVMAX + NFVMAX

#define TDMRX NSERMX
#define TDPARX NSERMX
#define TDXXY NSERMX

main()
{
/* Two examples are called, ex1() which uses the
 * default settings to solve the problem and
 * ex2() which solves the same problem with
 * some optional parameters set by the user.
 */

Vprintf("g13bjc Example Program Results\n");
Vscanf(" %*[\n"] ; /* Skip heading in data file */
ex1();
ex2();
exit(EXIT_SUCCESS);

[NP3491/6]
static void ex1()
{
  Integer i, j, n, nev, nfv, npara, nseries, inser;
  double fsd[NFVMAX], fva[NFVMAX], para[NFVMAX], parx[LDPARX][NSERMX],
    rmsxy[NSERMX], xxy[LDXXY][NSERMX];
  Integer mrx[7][NSERMX];
  Nag_ArimaOrder arimav;
  Nag_TransfOrder transfv;
  static NagError fail;

  Vprintf("\ngl13bjc example 1: no option setting.\n\n");
  
  /* Skip heading in data file */
  Vscanf("%*[\n]");

  Vscanf("%d%d%d", &nev, &nfv, &nseries);
  if (nseries>0 & nseries<=NSERMX & nev>0 & nev<=NEVMAX &
    nfv>0 & nfv<=NFVMAX)
  {
  
    /* Allocate memory to the arrays in structure transfv containing
     * the transfer function model orders of the input series.
     */
    g13byc(nseries, &transfv, NAGERR_DEFAULT);

    /*
     * Read the orders vector of the ARIMA model for the output noise
     * component into structure arimav.
     */
    Vscanf("%d%d%d%d%d%d", &arimav.p, &arimav.d, &arimav.q,
      &arimav.bigp, &arimav.bigd, &arimav.bigq, &arimav.s);

    /*
     * Read the transfer function model orders of the input series into
     * structure transfv.
     */
    inser = nseries - 1;

    for (j=0; j<inser; ++j)
      Vscanf("%ld", &transfv.b[j]);
    for (j=0; j<inser; ++j)
      Vscanf("%ld", &transfv.q[j]);
    for (j=0; j<inser; ++j)
      Vscanf("%ld", &transfv.p[j]);
    for (j=0; j<inser; ++j)
      Vscanf("%ld", &transfv.r[j]);

    npara = 0;
    for (i=0; i<inser; ++i)
      npara += transfv.q[i] + transfv.p[i];
    npara += nfv;
    for (i=0; i<nf; ++i)
for (j=0; j<nseries; ++j)
    Vscanf("%lf", &xxy[i][j]);
for (i=0; i<nseries; ++i)
    Vscanf("%lf", &rmsxy[i]);
for (i=0; i<7; ++i)
    for (j=0; j<inser; ++j)
        Vscanf("%ld", &mrx[i][j]);
for (i=0; i<5; ++i)
    for (j=0; j<inser; ++j)
        Vscanf("%lf", &parx[i][j]);

fail.print = TRUE;
gl3bjc(&arimav, nseries, &transfv, para, npara, nev, nfv,
    (double *)xxy, (Integer)TDXXY, rmsxy, (Integer *)mrx,
    (Integer)TDMRX, (double *)parx, (Integer)LDPARX,
    (Integer)LDPARX, fva, fsd, GL3_DEFAULT, &fail);

if (fail.code==NE_NOERROR || fail.code==NE_SOLUTION_FAIL_CONV ||
    fail.code==NE_MAT_NOT_POS_DEF)
{
    Vprintf("\nThe residual mean square for the output\n");
    Vprintf("\nseries is also derived and its value is %10.4f\n\n", rmsxy[nseries-1]);
    Vprintf("\nThe forecast values and their standard errors are\n");
    Vprintf("\n i  fva  fsd\n");
    for (i=0; i<nfv; ++i)
        Vprintf("%4ld%10.3f%10.4f\n", i+1, fva[i], fsd[i]);
}
else
{
    Vprintf(stderr, "npara is out of range: npara = %ld\n", npara);
    gl3bzc(&transfv);
    exit(EXIT_FAILURE);
}
else
{
    Vprintf(stderr, "One or more of nseries, nev and nfv are out of \nrange: nseries = %ld, nev = %ld while nfv = %ld\n", nseries, nev, nfv);
    exit(EXIT_FAILURE);
}
gl3bzc(&transfv);
if (fail.code!=NE_NOERROR)
    exit(EXIT_FAILURE);

8.1.2 Program Data

gl3bjc Example Program Data

Example 1 data

<table>
<thead>
<tr>
<th>40</th>
<th>8</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

[NP3491/6]
The image contains a numerical table with several columns and rows. The table appears to be a part of a mathematical or statistical analysis, possibly involving regression or another form of data analysis. The specific context or application of the data is not clear from the image alone.
8.1.3 Program Results

**gl3bjc Example Program Results**

*gl3bjc example 1: no option setting.*

**Parameters to gl3bjc**

```
nseries.....................  6
cfixed......................... FALSE
outfile..................... stdout
```

The residual mean square for the output series is also derived and its value is 20.7599

The forecast values and their standard errors are

<table>
<thead>
<tr>
<th>i</th>
<th>fva</th>
<th>fsd</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.398</td>
<td>4.5563</td>
</tr>
<tr>
<td>2</td>
<td>96.958</td>
<td>6.2172</td>
</tr>
<tr>
<td>3</td>
<td>86.046</td>
<td>7.0933</td>
</tr>
<tr>
<td>4</td>
<td>77.589</td>
<td>7.3489</td>
</tr>
<tr>
<td>5</td>
<td>82.139</td>
<td>7.3941</td>
</tr>
<tr>
<td>6</td>
<td>96.276</td>
<td>7.5823</td>
</tr>
<tr>
<td>7</td>
<td>98.345</td>
<td>8.1445</td>
</tr>
<tr>
<td>8</td>
<td>93.577</td>
<td>8.8536</td>
</tr>
</tbody>
</table>

9 Optional Parameters

A number of optional input and output parameters to nag_tsa_multi_inp_model_forecast are available through the structure argument **options** of type **Nag_G13_Opt**. A parameter may be selected by assigning an appropriate value to the relevant structure member and those parameters not selected will be assigned default values. If no use is to be made of any of the optional parameters the user should use the null pointer, **G13_DEFAULT**, in place of **options** when calling nag_tsa_multi_inp_model_forecast; the default settings will then be used for all parameters.

Before assigning values to **options** the structure must be initialised by a call to the function nag_tsa_options_init (g13bac). Values may then be assigned directly to the structure members in the normal C manner.

Options selected by direct assignment are checked within nag_tsa_multi_inp_model_forecast for being within the required range, if outside the range, an error message is generated.

When all calls to nag_tsa_multi_inp_model_forecast have been completed and the results contained in the options structure are no longer required; then nag_tsa_free (g13zac) should be called to free the NAG allocated memory from **options**.
9.1 Optional Parameters Checklist and Default Values

For easy reference, the following list shows the input and output members of options which are valid for nag_tsa_multi_inp_model_forecast together with their default values where relevant.

<table>
<thead>
<tr>
<th>Type</th>
<th>Member</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>list</td>
<td>TRUE</td>
<td>Input Default = TRUE. On entry: if options.list = TRUE then the parameter settings which are used in the call to nag_tsa_multi_inp_model_forecast will be printed.</td>
</tr>
<tr>
<td>Boolean</td>
<td>cfixed</td>
<td>FALSE</td>
<td>Input Default = FALSE. On entry: cfixed must be set to FALSE if the constant was estimated when the model was fitted, and TRUE if it was held at a fixed value. This only affects the degrees of freedom used in calculating the estimated residual variance.</td>
</tr>
<tr>
<td>double</td>
<td>zt</td>
<td></td>
<td>Output Default memory = (nev+nfv)\times (nseries−1) On exit: this pointer is allocated memory internally with (nev+nfv) \times (nseries−1) rows by nseries−1 columns. The columns of zt hold the values of the input component series zt.</td>
</tr>
<tr>
<td>double</td>
<td>noise</td>
<td></td>
<td>Output Default memory = nev+nfv On exit: this pointer is allocated memory internally with nev + nfv elements. It holds the output noise component nt.</td>
</tr>
</tbody>
</table>

9.2 Description of Optional Parameters

list – Boolean Input Default = TRUE

On entry: if options.list = TRUE then the parameter settings which are used in the call to nag_tsa_multi_inp_model_forecast will be printed.

cfixed – Boolean Input Default = FALSE

On entry: cfixed must be set to FALSE if the constant was estimated when the model was fitted, and TRUE if it was held at a fixed value. This only affects the degrees of freedom used in calculating the estimated residual variance.

zt – double * Output Default memory = (nev+nfv)\times (nseries−1) On exit: this pointer is allocated memory internally with (nev+nfv) \times (nseries−1) rows by nseries−1 columns. The columns of zt hold the values of the input component series zt.

noise – double * Output Default memory = nev+nfv On exit: this pointer is allocated memory internally with nev + nfv elements. It holds the output noise component nt.

10 Example 2

This example illustrates the use of the options parameter in a call to nag_tsa_multi_inp_model_forecast.

The data in the example relate to the same 40 observations of an output time series and 5 input time series as in Example 1. This example differs from Example 2 in nag_tsa_multi_inp_model_estim (g13bec) in that there are now 4 simple input series. The output series has one autoregressive (φ) parameter and one seasonal moving average (θ) parameter. The seasonal period is 4. The transfer function input (the fifth in the set) is defined by orders b5 = 1, q5 = 0, p5 = 1, r5 = 3, so that it allows for pre-observation period effects. The initial values of the specified model are:

φ = 0.495, θ = 0.238, ω1 = −0.367, ω2 = −3.876, ω3 = 4.516
ω4 = 2.474, ω5,1 = 8.629, δ5,1 = 0.688, c = −82.858.

A further 8 values of the input series are supplied, and it is assumed that the values for the fifth series have themselves been forecast from an ARIMA model with orders 2 0 2 0 1 1 4, in which φ1 = 1.6743, φ2 = −0.9505, θ1 = 1.4605, θ2 = −0.4862 and θ1 = 0.8993, and for which the residual mean square is 0.1720.

The following are computed and printed out: the estimated residual variance for the output noise series, the 8 forecast values and their standard errors, and the values of the components zt and the output noise component nt.
10.1 Program Text

```c
static void ex2()
{
    Integer i, j, n, nev, nfv, npara, nseries, inser;
    double fsd[NFVMAX], fva[NFVMAX], para[NFVMAX], parx[LDPARX][TDPARX],
    rmsxy[NSERMX], xxy[LDXXY][TDXXY];
    Integer mrx[7][NSERMX];
    Nag_ArimaOrder arimav;
    Nag_TransfOrder transfv;
    Nag_G13_Opt options;
    static NagError fail;

    #define ZT(I,J) options.zt[(J)+(I) * options.tdzt]

    Vprintf("\n\ngl3bjc example 2: using option setting.\n\n")

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

    /*
     * Initialise the option-setting function.
     */
    gl3bxc(&options);

    Vscanf("%ld%ld%ld", &nev, &nfv, &nseries);
    if (nseries>0 && nseries<NSERMX && nev>0 && nev<NEVMAX &&
        nfv>0 && nfv<NFVMAX)
    {
        /*
         * Set option variable to the desired value.
         */
        options.cfixed = TRUE;

        /*
         * Allocate memory to the arrays in structure transfv containing
         * the transfer function model orders of the input series.
         */
        gl3byc(nseries, &transfv, NAGERR_DEFAULT);

        /*
         * Read the orders vector of the ARIMA model for the output noise
         * component into structure arimav.
         */
        Vscanf("%ld%ld%ld%ld%ld", &arimav.p, &arimav.d, &arimav.q,
               &arimav.bigp, &arimav.bigd, &arimav.bigq, &arimav.s);

        /*
         * Read the transfer function model orders of the input series into
         * structure transfv.
         */
        inser = nseries - 1;
        for (j=0; j<inser; ++j)
            Vscanf("%ld", &transfv.b[j]);
        for (j=0; j<inser; ++j)
            Vscanf("%ld", &transfv.q[j]);
        for (j=0; j<inser; ++j)
            Vscanf("%ld", &transfv.p[j]);
        for (j=0; j<inser; ++j)
```
Vscanf("%ld", &transf.v.r[j]);
np = 0;
for (i=0; i<transf.q[i] + transf.p[i];)
np = npa + transf.q[i] + transf.p[i];
np = npa + arimav.p + arimav.q + arimav.bigp + arimav.bigq + nseries;
if (np<NPMAX)
{
    for (i=0; i<np; ++i)
        Vscanf("%lf", &para[i]);
    n = nev + nfv;
    for (i=0; i<n; ++i)
        for (j=0; j<nseries; ++j)
            Vscanf("%lf", &xy[i][j]);
    for (i=0; i<nseries; ++i)
        Vscanf("%lf", &rmsxy[i]);
    for (i=0; i<7; ++i)
        for (j=0; j<nseries; ++j)
            Vscanf("%ld", &mx[i][j]);
    for (i=0; i<5; ++i)
        for (j=0; j<nseries; ++j)
            Vscanf("%lf", &parx[i][j]);
    fail.print = TRUE;

gl3bjc(arimav, nseries, &transf, para, npa, nev, nfv,
    (double *)xxy, (Integer)TDXXY, rmsxy, (Integer *)mr,
    (Integer)TDMRX, (double *)parx, (Integer)LDPARX,
    (Integer)LDHARX, fva, fsd, &options, &fail);

if (fail.code==NE_NOERROR || fail.code==NE_SOLUTION_FAIL_CONV ||
    fail.code==NE_MAT_NOT_POS_DEF)
{
    Vprintf("%ld sets of observations were processed.\n",nev);
    Vprintf("\nThe residual mean square for the output ");
    Vprintf("series is %10.4f\n\n", rmsxy[nseries-1]);
    Vprintf("The forecast values and their standard errors are\n\n");
    Vprintf("\n  i   fva   fsd\n");
    for (i=0; i<nf; ++i)
        Vprintf("%4d%10.3f%10.4f\n", i+1, fva[i], fsd[i]);
    Vprintf("\nThe values of z(t) and noise(t) are\n\n");
    Vprintf(" i   z1   z2   z3   z4\nnoise\n\n");
    for (i=0; i<n; ++i)
    {
        Vprintf("%4d", i+1);
        for (j=0; j<nseries-1; ++j)
            Vprintf("%10.3f ", 2T(i,j));
        Vprintf("%10.3f\n", options.noise[i]);
    }
}
else
{
    Vfprintf(stderr, "npa is out of range: npa = %ld\n", npa);
    gl3xzc(&options);
    gl3bzc(&transf);
    exit(EXIT_FAILURE);


10.2 Program Data

Example 2 data

40 8 6
1 0 0 0 0 1 4
0 0 0 0 1
0 0 0 0 0
0 0 0 0 1
1 1 1 1 1

0.4950 0.2380 -0.3670 -3.8760 4.5160 2.4740 8.6290 0.6880
-82.8580

1.0 1.0 0.0 0.0 8.075 105.0
1.0 0.0 1.0 0.0 7.819 119.0
1.0 0.0 0.0 1.0 7.366 119.0
1.0 -1.0 -1.0 -1.0 8.113 109.0
2.0 1.0 0.0 0.0 7.380 117.0
2.0 0.0 1.0 0.0 7.134 135.0
2.0 0.0 0.0 1.0 7.222 126.0
2.0 -1.0 -1.0 -1.0 7.768 112.0
3.0 1.0 0.0 0.0 7.386 116.0
3.0 0.0 1.0 0.0 6.965 122.0
3.0 0.0 0.0 1.0 6.478 115.0
3.0 -1.0 -1.0 -1.0 8.105 115.0
4.0 1.0 0.0 0.0 8.060 122.0
4.0 0.0 1.0 0.0 7.684 138.0
4.0 0.0 0.0 1.0 7.580 135.0
4.0 -1.0 -1.0 -1.0 7.093 125.0
5.0 1.0 0.0 0.0 6.129 115.0
5.0 0.0 1.0 0.0 6.026 108.0
5.0 0.0 0.0 1.0 6.679 100.0
5.0 -1.0 -1.0 -1.0 7.414 96.0
6.0 1.0 0.0 0.0 7.112 107.0
6.0 0.0 1.0 0.0 7.762 115.0
6.0 0.0 0.0 1.0 7.645 123.0
6.0 -1.0 -1.0 -1.0 8.639 122.0
7.0 1.0 0.0 0.0 7.667 128.0
7.0 0.0 1.0 0.0 8.080 136.0
7.0 0.0 0.0 1.0 6.678 140.0
7.0 -1.0 -1.0 -1.0 6.739 122.0
8.0 1.0 0.0 0.0 5.569 102.0
8.0 0.0 1.0 0.0 5.049 103.0
8.0 0.0 0.0 1.0 5.642 89.0
8.0 -1.0 -1.0 -1.0 6.808 77.0
9.0 1.0 0.0 0.0 6.636 89.0
<table>
<thead>
<tr>
<th>i</th>
<th>fva</th>
<th>fsd</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.398</td>
<td>4.4822</td>
</tr>
<tr>
<td>2</td>
<td>96.958</td>
<td>6.1498</td>
</tr>
<tr>
<td>3</td>
<td>86.046</td>
<td>7.0315</td>
</tr>
<tr>
<td>4</td>
<td>77.589</td>
<td>7.2885</td>
</tr>
<tr>
<td>5</td>
<td>82.139</td>
<td>7.3327</td>
</tr>
<tr>
<td>6</td>
<td>96.276</td>
<td>7.5220</td>
</tr>
<tr>
<td>7</td>
<td>98.345</td>
<td>8.0883</td>
</tr>
<tr>
<td>8</td>
<td>93.577</td>
<td>8.8020</td>
</tr>
</tbody>
</table>
The values of $z(t)$ and noise(t) are

<table>
<thead>
<tr>
<th>i</th>
<th>$z_1$</th>
<th>$z_2$</th>
<th>$z_3$</th>
<th>$z_4$</th>
<th>$z_5$</th>
<th>noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.339</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>188.603</td>
<td>-79.375</td>
</tr>
<tr>
<td>2</td>
<td>-0.339</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>199.438</td>
<td>-84.613</td>
</tr>
<tr>
<td>3</td>
<td>-0.339</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>204.683</td>
<td>-87.823</td>
</tr>
<tr>
<td>4</td>
<td>-0.339</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>204.383</td>
<td>-91.940</td>
</tr>
<tr>
<td>5</td>
<td>-0.678</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>210.623</td>
<td>-89.056</td>
</tr>
<tr>
<td>6</td>
<td>-0.678</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>208.591</td>
<td>-77.426</td>
</tr>
<tr>
<td>7</td>
<td>-0.678</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>205.070</td>
<td>-80.870</td>
</tr>
<tr>
<td>8</td>
<td>-0.678</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>203.407</td>
<td>-87.624</td>
</tr>
<tr>
<td>9</td>
<td>-1.017</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>206.974</td>
<td>-86.068</td>
</tr>
<tr>
<td>10</td>
<td>-1.017</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>206.132</td>
<td>-87.628</td>
</tr>
<tr>
<td>11</td>
<td>-1.017</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>201.920</td>
<td>-88.381</td>
</tr>
<tr>
<td>12</td>
<td>-1.356</td>
<td>-3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>194.819</td>
<td>-75.698</td>
</tr>
<tr>
<td>13</td>
<td>-1.356</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>203.974</td>
<td>-76.729</td>
</tr>
<tr>
<td>14</td>
<td>-1.356</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>210.705</td>
<td>-86.828</td>
</tr>
<tr>
<td>16</td>
<td>-1.695</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>205.942</td>
<td>-85.358</td>
</tr>
<tr>
<td>17</td>
<td>-1.695</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>194.575</td>
<td>-89.394</td>
</tr>
<tr>
<td>18</td>
<td>-1.695</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>185.866</td>
<td>-86.650</td>
</tr>
<tr>
<td>19</td>
<td>-2.035</td>
<td>-3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>185.509</td>
<td>-84.709</td>
</tr>
<tr>
<td>20</td>
<td>-2.035</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>191.606</td>
<td>-78.682</td>
</tr>
<tr>
<td>21</td>
<td>-2.035</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>199.896</td>
<td>-80.673</td>
</tr>
<tr>
<td>22</td>
<td>-2.035</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>203.497</td>
<td>-76.358</td>
</tr>
<tr>
<td>23</td>
<td>-2.374</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>214.552</td>
<td>-80.290</td>
</tr>
<tr>
<td>24</td>
<td>-2.374</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>213.770</td>
<td>-79.910</td>
</tr>
<tr>
<td>25</td>
<td>-2.374</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>216.796</td>
<td>-76.901</td>
</tr>
<tr>
<td>27</td>
<td>-2.713</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>200.416</td>
<td>-91.814</td>
</tr>
<tr>
<td>28</td>
<td>-2.713</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>185.943</td>
<td>-84.742</td>
</tr>
<tr>
<td>29</td>
<td>-2.713</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>171.495</td>
<td>-82.261</td>
</tr>
<tr>
<td>30</td>
<td>-3.052</td>
<td>-3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>166.673</td>
<td>-83.857</td>
</tr>
<tr>
<td>31</td>
<td>-3.052</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>173.418</td>
<td>-77.477</td>
</tr>
<tr>
<td>32</td>
<td>-3.052</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>176.573</td>
<td>-84.035</td>
</tr>
<tr>
<td>33</td>
<td>-3.052</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>192.594</td>
<td>-88.021</td>
</tr>
<tr>
<td>34</td>
<td>-3.052</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>201.261</td>
<td>-87.105</td>
</tr>
<tr>
<td>35</td>
<td>-3.391</td>
<td>-3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>207.879</td>
<td>-81.599</td>
</tr>
<tr>
<td>36</td>
<td>-3.391</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>210.249</td>
<td>-85.372</td>
</tr>
<tr>
<td>37</td>
<td>-3.391</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>205.262</td>
<td>-85.350</td>
</tr>
<tr>
<td>38</td>
<td>-3.391</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>193.874</td>
<td>-84.379</td>
</tr>
<tr>
<td>39</td>
<td>-3.730</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>185.617</td>
<td>-84.600</td>
</tr>
<tr>
<td>40</td>
<td>-3.730</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>178.969</td>
<td>-82.795</td>
</tr>
<tr>
<td>41</td>
<td>-3.730</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>169.607</td>
<td>-82.309</td>
</tr>
<tr>
<td>42</td>
<td>-3.730</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>166.832</td>
<td>-82.409</td>
</tr>
<tr>
<td>43</td>
<td>-4.069</td>
<td>-3.889</td>
<td>0.000</td>
<td>0.000</td>
<td>172.733</td>
<td>-82.636</td>
</tr>
<tr>
<td>44</td>
<td>-4.069</td>
<td>0.000</td>
<td>4.514</td>
<td>0.000</td>
<td>178.579</td>
<td>-82.748</td>
</tr>
<tr>
<td>45</td>
<td>-4.069</td>
<td>0.000</td>
<td>0.000</td>
<td>2.479</td>
<td>182.739</td>
<td>-82.804</td>
</tr>
<tr>
<td>46</td>
<td>-4.069</td>
<td>3.889</td>
<td>-4.514</td>
<td>-2.479</td>
<td>183.582</td>
<td>-82.831</td>
</tr>
</tbody>
</table>