NAG C Library Function Document

nag_estimate_agarchI (g13fac)

1 Purpose

nag_estimate_agarchI (g13fac) estimates the parameters of a standard univariate regression-GARCH(p, q) or a univariate regression-type I AGARCH(p, q) process (see Engle and Ng (1993)).

2 Specification

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

void nag_estimate_agarchI (const double yt[], const double x[], Integer tdx,
    Integer num, Integer p, Integer q, Integer nreg, Integer mn,
    Integer isym, double theta[], double se[], double sc[],
    double covar[], Integer tdc, double *hp, double et[], double ht[],
    double *lgf, Nag_Garch_Stationary_Type stat_opt,
    Nag_Garch_Stationary_Type est_opt, Integer max_iter,
    double tol, NagError *fail)

3 Description

When isym = 0, nag_estimate_agarchI models a standard (γ = 0) univariate regression-GARCH(p, q) process, with p coefficients αi, i = 1,...,p, q coefficients, βk, i = 1,...,q, mean b0, and k linear regression coefficients bi, i = 1,...,k, which can be represented by:

\[ y_t = b_0 + x_t^T b + \epsilon_t \]  
\[ \epsilon_t | \psi_{t-1} \sim N(0, \psi_t) \]  
\[ h_t = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i h_{t-i}, \quad t = 1,...,T. \]

When isym = 1, nag_estimate_agarchI models an asymmetric GARCH(p, q) process where the conditional variance \( h_t \) is given by:

\[ h_t = \alpha_0 + \sum_{i=1}^{q} \alpha_i (\epsilon_{t-i} + \gamma)^2 + \sum_{i=1}^{p} \beta_i h_{t-i}, \quad t = 1,...,T. \]

Here T is the number of terms in the sequence, yt denotes the endogenous variables, xt the exogenous variables, b0 the mean, b the regression coefficients, \( \epsilon_t \) the residuals, \( \gamma \) is the asymmetry parameter, \( h_t \) is the conditional variance, and \( \psi_t \) the information set of all information up to time t.

When isym = 1, nag_estimate_agarchI provides an estimate for \( \hat{\theta} \), the \((p + q + k + 3) \times 1 \) parameter vector \( \theta = (b_0, b^T, \omega^T) \) where \( \omega^T = (\alpha_0, \alpha_1, \ldots, \alpha_q, \beta_1, \ldots, \beta_p, \gamma) \) and \( b^T = (b_1, \ldots, b_k) \).

isym, mn, nreg (see Section 4) can be used to simplify the GARCH(p, q) expression in (1) as follows:

No Regression or Mean

\[ y_t = \epsilon_t, \]
\[ \text{isym} = 0, \]
\[ \text{mn} = 0, \]
\[ \text{nreg} = 0, \text{and} \]
\[ \theta \text{ is a } (p + q + 1) \times 1 \text{ vector}. \]
No Regression
\[ y_t = b_0 + \epsilon_t, \]
\[ \text{isym} = 0, \]
\[ \text{mn} = 1, \]
\[ \text{nreg} = 0, \]
\[ \theta \text{ is a } (p + q + 2) \times 1 \text{ vector.} \]

Note: if the \( y_t = \mu + \epsilon_t, \) where \( \mu \) is known (not to be estimated by nag_estimate_agarchl) then (1) can be written as \( y_t^\mu = \epsilon_t, \) where \( y_t^\mu = y_t - \mu. \) This corresponds to the case No Regression or Mean, with \( y_t \) replaced by \( y_t - \mu. \)

No Mean
\[ y_t = x_t^T b + \epsilon_t, \]
\[ \text{isym} = 0, \]
\[ \text{mn} = 0, \]
\[ \text{nreg} = k \text{ and} \]
\[ \theta \text{ is a } (p + q + k + 1) \times 1 \text{ vector.} \]

4 Parameters

Note: for convenience npar will be used here to denote the expression \( 1+q+p+\text{isym}+\text{mn}+\text{nreg} \)
representing the number of model parameters.

1: \[ \text{yt[num]} \text{ – const double} \]
   \text{On entry:} the sequence of observations, \( y_t, t = 1, \ldots, T. \]
   \text{Input}

2: \[ \text{x[num][tdx]} \text{ – const double} \]
   \text{On entry:} row \( t \) of \( x \) contains the time dependent exogenous vector \( x_t, \) where \( x_t^T = (x_t^1, \ldots, x_t^k), \)
   for \( t = 1, \ldots, T. \]
   \text{Input}

3: \[ \text{tdx} \text{ – Integer} \]
   \text{On entry:} the second dimension of the array \( x \) as declared in the function from which
   nag_estimate_agarchl is called.
   \text{Constraint:} \( \text{tdx} \geq \text{nreg}. \)

4: \[ \text{num} \text{ – Integer} \]
   \text{On entry:} the number of terms in the sequence, \( T. \)
   \text{Input}
   \text{Constraint:} \( \text{num} \geq \text{npar}. \)

5: \[ \text{p} \text{ – Integer} \]
   \text{On entry:} the GARCH(\( p, q \)) parameter \( p. \)
   \text{Constraint:} \( p \geq 0. \)

6: \[ \text{q} \text{ – Integer} \]
   \text{On entry:} the GARCH(\( p, q \)) parameter \( q. \)
   \text{Constraint:} \( q \geq 1. \)
7. nreg – Integer
   
   *Input*

   *On entry:* the number of regression coefficients, k.

   *Constraint:* nreg ≥ 0.

8. mn – Integer
   
   *Input*

   *On entry:* if mn = 1 then the mean term b0 will be included in the model.

   *Constraint:* mn = 0 or mn = 1.

9. isym – Integer
   
   *Input*

   *On entry:* if isym = 1 then the asymmetry term γ will be included in the model.

   *Constraint:* isym = 0 or isym = 1.

10. theta[npar] – double
    
    *Input/Output*

    *On entry:* the initial parameter estimates for the vector θ. The first element contains the coefficient α0, the next q elements contain the coefficients αi, i = 1, .., q. The next p elements are the coefficients βj, j = 1, .., p. If isym = 1 then the next element contains the asymmetry parameter γ. If est_opt = Nag_Garch_Est_Initial.False then (when mn = 1) the next term contains an initial estimate of the mean term b0 and the remaining nreg elements are taken as initial estimates of the linear regression coefficients bi, i = 1, .., k.

    *On exit:* the estimated values of θ for the vector θ. The first element contains the coefficient α0, the next q elements contain the coefficients αi, i = 1, .., q. The next p elements are the coefficients βj, j = 1, .., p. If isym = 1 then the next element contains the estimate for the asymmetry parameter γ. If mn = 1 then the next element contains an estimate for the mean term b0. The final nreg elements are the estimated linear regression coefficients bi, i = 1, .., k.

11. se[npar] – double
    
    *Output*

    *On exit:* the standard errors for θ. The first element contains the standard error for α0, the next q elements contain the standard errors for αi, i = 1, .., q, the next p elements are the standard errors for βj, j = 1, .., p. If isym = 1 then the next element contains the standard error for γ. If mn = 1 then the next element contains the standard error for b0. The final nreg elements are the standard errors for bj, j = 1, .., k.

12. sc[npar] – double
    
    *Output*

    *On exit:* the scores for θ. The first element contains the score for α0, the next q elements contain the score for αi, i = 1, .., q, the next p elements are the scores for βj, j = 1, .., p. If isym = 1 then the next element contains the score for γ. If mn = 1 then the next element contains the score for b0. The final nreg elements are the scores for bj, j = 1, .., k.

13. covar[npar][tde] – double
    
    *Output*

    *On exit:* the covariance matrix of the parameter estimates θ, that is the inverse of the Fisher Information Matrix.

14. tdc – Integer
    
    *Input*

    *On entry:* the second dimension of the array covar as declared in the function from which nag_estimate_agarch1 is called.

    *Constraint:* tdc ≥ npar.

15. hp – double *
    
    *Input/Output*

    *On entry:* If est_opt = Nag_Garch_Est_Initial.False then hp is the value to be used for the pre-observed conditional variance. If est_opt = Nag_Garch_Est_Initial.True then hp is not referenced.

[NP3491/6]
On exit: If \texttt{est\_opt} = \texttt{Nag\_Garch\_Est\_Initial\_True} then \texttt{hp} is the estimated value of the pre-observed conditional variance.

16. \texttt{et[num]} – double \hspace{1cm} \textit{Output}

\textit{On exit:} the estimated residuals, \(\epsilon_t, t = 1, \ldots, T\).

17. \texttt{ht[num]} – double \hspace{1cm} \textit{Output}

\textit{On exit:} the estimated conditional variances, \(h_t, t = 1, \ldots, T\).

18. \texttt{lgf} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the value of the log likelihood function at \(\hat{\theta}\).

19. \texttt{stat\_opt} – \texttt{Nag\_Garch\_Stationary\_Type} \hspace{1cm} \textit{Input}

\textit{On entry:} If \texttt{stat\_opt} = \texttt{Nag\_Garch\_Stationary\_True} then Stationary conditions are enforced. If \texttt{stat\_opt} = \texttt{Nag\_Garch\_Stationary\_False} then Stationary conditions are not enforced.

20. \texttt{est\_opt} – \texttt{Nag\_Garch\_Est\_Initial\_Type} \hspace{1cm} \textit{Input}

\textit{On entry:} If \texttt{est\_opt} = \texttt{Nag\_Garch\_Est\_Initial\_True} then the routine provides initial parameter estimates of the regression terms \((b_0, \beta)^T\). If \texttt{est\_opt} = \texttt{Nag\_Garch\_Est\_Initial\_False} then the initial estimates of the regression parameters \((b_0, \beta)^T\) must be supplied by the user.

21. \texttt{max\_iter} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the maximum number of iterations to be used by the optimisation routine when estimating the GARCH \((p, q)\) parameters. If \texttt{max\_iter} is set to 0 then the standard errors, score vector and variance-covariance are calculated for the input value of \(\theta\) in \texttt{theta}; however the value of \(\theta\) is not updated.

\textit{Constraint:} \texttt{max\_iter} \(\geq 0\).

22. \texttt{tol} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the tolerance to be used by the optimisation routine when estimating the GARCH \((p, q)\) parameters.

23. \texttt{fail} – \texttt{NagError} * \hspace{1cm} \textit{Input/Output}

The NAG error parameter (see the Essential Introduction).

5 \hspace{1cm} \textbf{Error Indicators and Warnings}

\textbf{NE\_BAD\_PARAM}

On entry, parameter \texttt{stat\_opt} had an illegal value.
On entry, parameter \texttt{est\_opt} had an illegal value.

\textbf{NE\_INT\_ARG\_LT}

On entry, \texttt{nreg} must not be less than 0: \texttt{nreg} = \texttt{<value>}. 
On entry, \texttt{q} must not be less than 1: \texttt{q} = \texttt{<value>}. 
On entry, \texttt{p} must not be less than 0: \texttt{p} = \texttt{<value>}. 
On entry, \texttt{max\_iter} must not be less than 0: \texttt{max\_iter} = \texttt{<value>}. 

\textbf{NE\_2\_INT\_ARG\_LT}

On entry, \texttt{tdx} = \texttt{<value>} while \texttt{nreg} = \texttt{<value>}. 
These parameters must satisfy \texttt{tdx} \(\geq\) \texttt{nreg}.
On entry, \( tdc < value > \) while \( 1 + q + p + isym + mn + nreg = value > \).
These parameters must satisfy \( tdc \geq 1 + q + p + isym + mn + nreg \).

On entry, \( num < value > \) while \( 1 + q + p + isym + mn + nreg = value > \).
These parameters must satisfy \( num \geq 1 + q + p + isym + mn + nreg \).

**NE_INVALID_INT_RANGE_2**

Value \( < value > \) given to \( mn \) is not valid. Correct range is 0 to 1.
Value \( < value > \) given to \( isym \) is not valid. Correct range is 0 to 1.

**NE_MAT_NOT_FULL_RANK**

Matrix \( X \) does not give a model of full rank.

**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_ALLOC_FAIL**

Memory allocation failed.

**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

6.1 Accuracy

Not applicable.

6.2 References


7 See Also

None.

8 Example

This example program illustrates the use of nag_estimate_agarch1 to model a GARCH(1,1) sequence generated by nag_generate_agarch1 (g05hkc), a three step forecast is then calculated using nag_forecast_agarch1 (g13fbc).
8.1 Program Text

/* nag_estimate_agarchI (gl3fac) Example Program. */
/* Copyright 2000 Numerical Algorithms Group. */
/* Mark 6, 2000. */
*/

#include <nag.h>
#include <nag_stdio.h>
#include <stdio.h>
#include <ctype.h>
#include <math.h>
#include <nag05.h>
#include <naggl3.h>

int main(void)
{

double *bx=0, *covar=0, *et=0, facl, gamma, hp;
double *ht=0, lgf, mean, *param=0;
double *rvec=0, *sc=0, *se=0, *theta=0, tol;
double *x=0, xterm, *yt=0, *cvar=0;
Integer i, ip, iq, isym, j, k, nt;
Integer exit_status=0;
Integer tdc, tdx;
Integer maxit, mn, num, num_startup, npar, nreg, seed;
Nag_Garch_Est_Initial_Type est_opt;
Nag_Garch_Stationary_Type stat_opt;
Nag_Garch_Fcall_Type fcall;
NagError fail;

INIT_FAIL(fail);
isym = 1;
gamma = -0.3;
nreg = 2;
ip = 1;
iq = 1;
mn = 1;

npar = iq + ip + 1;
um = 1000;
nt = 3;

tdc = npar+mn+isym+nreg;
tdx = nreg;
#define YT(I) yt[(I)-1]
#define THETA(I) theta[(I)-1]
#define SE(I) se[(I)-1]
#define SC(I) sc[(I)-1]
#define RVEC(I) rvec[(I)-1]
#define PARAM(I) param[(I)-1]
#define HT(I) ht[(I)-1]
#define ET(I) et[(I)-1]
#define BX(I) bx[(I)-1]
#define CVAR(I) cvar[(I)-1]
#define X(I,J) x[((I)-1) * tdx + ((J)-1)]
#define COVAR(I,J) covar[((I)-1) * tdc + ((J)-1)]

if (!bx = NAG_ALLOC (nreg, double))
  || (!covar = NAG_ALLOC ((npar+mn+isym+nreg) * (npar+mn+isym+nreg), double))
  || (!et = NAG_ALLOC (num, double))
  || (!ht = NAG_ALLOC (num, double))
  || (!param = NAG_ALLOC (npar+mn+isym+nreg, double))
  || (!rvec = NAG_ALLOC (40, double))
  || (!sc = NAG_ALLOC (npar+mn+isym+nreg, double))
  || (!se = NAG_ALLOC (npar+mn+isym+nreg, double))
  || (!theta = NAG_ALLOC (npar+mn+isym+nreg, double))
  || (!x = NAG_ALLOC (num * nreg, double))
  || (!cvar = NAG_ALLOC (nt, double))
  || (!yt = NAG_ALLOC (num, double))
{
  Vprintf("Allocation failure\n");
  exit_status = -1;
  goto END;
}

Vprintf ("gl13fac Example Program Results \n\n");
seed = 11;
mean = 3.0;

if (nreg > 0)
{
  for (i = 1; i <= num; ++i)
  {
    fac1 = (double) i * 0.01;
    X (i, 1) = sin (fac1) * 0.7 + 0.01;
    X (i, 2) = fac1 * 0.1 + 0.5;
  }
  BX (1) = 1.5;
  BX (2) = 2.5;
}

PARAM (1) = 0.15;
PARAM (2) = 0.1;
PARAM (3) = 0.8;
PARAM (4) = 0.1;

fcall = Nag_Garch_Fcall_True;
g05cbc(seed);
num_startup = num;
g05hkc (num_startup, ip, iq, &PARAM (1), gamma, &HT (1), &YT (1), fcall, &RVEC (1), &fail);
if (fail.code != NE_NOERROR)
{
  Vprintf("Error from g05hkc.\n\n");
  exit_status = 1;
  goto END;
}

fcall = Nag_Garch_Fcall_False;
g05hkc (num, ip, iq, &PARAM (1), gamma, &HT (1), &YT (1), fcall, &RVEC (1), &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from g05hkc.\n\n", fail.message);
    exit_status = 1;
    goto END;
}

for (i = 1; i <= num; ++i)
{
    xterm = 0.0;
    for (k = 1; k <= nreg; ++k)
        xterm += X (i, k) * BX (k);

    if (mn == 1)
        YT (i) = mean + xterm + YT (i);
    else
        YT (i) = xterm + YT (i);
}

for (i = 1; i <= npar; ++i)
    THETA (i) = PARAM (i) * 0.5;

if (isym == 1)
    THETA (npar + isym) = gamma * 0.5;

if (mn == 1)
    THETA (npar + isym + 1) = mean * 0.5;

for (i = 1; i <= nreg; ++i)
    THETA (npar + isym + mn + i) = BX (i) * 0.5;

maxit = 50;
tol = 1e-12;
stat_opt = Nag_Garch_Stationary_True;
est_opt = Nag_Garch_Est_Initial_True;

fraim (&YT (1), &X (1, 1), tdx, num, ip, iq, nreg, mn, isym, &THETA (1), &SE (1), &SC (1), &COVAR (1, 1), tdc, &hp, &ET (1), &HT (1), &gf, stat_opt, est_opt, maxit, tol, &fail);

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

Vprintf ("Parameter estimates Standard errors Correct values\n");
for (j = 1; j <= npar; ++j)
    Vprintf ("%20.4f (%6.4f %20.4f\n", THETA (j), SE (j), PARAM(j));

    if (isym)
        Vprintf ("%20.4f (%6.4f %20.4f\n", THETA (npar+isym), SE (npar+-
isym), gamma);
    if (mn)
8.2 Program Data
None.

8.3 Program Results

<table>
<thead>
<tr>
<th>Parameter estimates</th>
<th>Standard errors</th>
<th>Correct values</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0902</td>
<td>(0.0361)</td>
<td>0.1500</td>
</tr>
<tr>
<td>0.1030</td>
<td>(0.0253)</td>
<td>0.1000</td>
</tr>
<tr>
<td>0.8433</td>
<td>(0.0390)</td>
<td>0.8000</td>
</tr>
<tr>
<td>-0.1509</td>
<td>(0.1836)</td>
<td>-0.3000</td>
</tr>
<tr>
<td>3.0840</td>
<td>(0.1395)</td>
<td>3.0000</td>
</tr>
<tr>
<td>1.4989</td>
<td>(0.0790)</td>
<td>1.5000</td>
</tr>
<tr>
<td>2.4402</td>
<td>(0.1354)</td>
<td>2.5000</td>
</tr>
</tbody>
</table>

3 step forecast = 1.5355