

NAG Toolbox

nag_rand_times_garch_asym2 (g05pe)

1 Purpose

nag_rand_times_garch_asym2 (g05pe) generates a given number of terms of a type II AGARCH(p, q) process (see Engle and Ng (1993)).

2 Syntax

```
[ht, et, r, state, ifail] = nag_rand_times_garch_asym2(dist, num, ip, iq, theta,
gamma, df, fcall, r, state, 'lr', lr)
```

```
[ht, et, r, state, ifail] = g05pe(dist, num, ip, iq, theta, gamma, df, fcall, r,
state, 'lr', lr)
```

3 Description

A type II AGARCH(p, q) process can be represented by:

$$h_t = \alpha_0 + \sum_{i=1}^q \alpha_i (|\epsilon_{t-i}| + \gamma \epsilon_{t-i})^2 + \sum_{i=1}^p \beta_i h_{t-i}, \quad t = 1, 2, \dots, T;$$

where $\epsilon_t | \psi_{t-1} = N(0, h_t)$ or $\epsilon_t | \psi_{t-1} = S_t(df, h_t)$. Here S_t is a standardized Student's t -distribution with df degrees of freedom and variance h_t , T is the number of observations in the sequence, ϵ_t is the observed value of the GARCH(p, q) process at time t , h_t is the conditional variance at time t , and ψ_t the set of all information up to time t . Symmetric GARCH sequences are generated when γ is zero, otherwise asymmetric GARCH sequences are generated with γ specifying the amount by which positive (or negative) shocks are to be enhanced.

One of the initialization functions nag_rand_init_repeat (g05kf) (for a repeatable sequence if computed sequentially) or nag_rand_init_nonrepeat (g05kg) (for a non-repeatable sequence) must be called prior to the first call to nag_rand_times_garch_asym2 (g05pe).

4 References

Bollerslev T (1986) Generalised autoregressive conditional heteroskedasticity *Journal of Econometrics* **31** 307–327

Engle R (1982) Autoregressive conditional heteroskedasticity with estimates of the variance of United Kingdom inflation *Econometrica* **50** 987–1008

Engle R and Ng V (1993) Measuring and testing the impact of news on volatility *Journal of Finance* **48** 1749–1777

Hamilton J (1994) *Time Series Analysis* Princeton University Press

5 Parameters

5.1 Compulsory Input Parameters

1: **dist** – CHARACTER(1)

The type of distribution to use for ϵ_t .

dist = 'N'

A Normal distribution is used.

dist = 'T'

A Student's t -distribution is used.

Constraint: **dist** = 'N' or 'T'.

2: **num** – INTEGER

T , the number of terms in the sequence.

Constraint: **num** ≥ 0 .

3: **ip** – INTEGER

The number of coefficients, β_i , for $i = 1, 2, \dots, p$.

Constraint: **ip** ≥ 0 .

4: **iq** – INTEGER

The number of coefficients, α_i , for $i = 1, 2, \dots, q$.

Constraint: **iq** ≥ 1 .

5: **theta**(**iq** + **ip** + 1) – REAL (KIND=nag_wp) array

The first element must contain the coefficient α_o , the next **iq** elements must contain the coefficients α_i , for $i = 1, 2, \dots, q$. The remaining **ip** elements must contain the coefficients β_j , for $j = 1, 2, \dots, p$.

Constraints:

$$\sum_{i=2}^{\text{iq}+\text{ip}+1} \text{theta}(i) < 1.0;$$

$$\text{theta}(i) \geq 0.0, \text{ for } i = 2, 3, \dots, \text{ip} + \text{iq} + 1.$$

6: **gamma** – REAL (KIND=nag_wp)

The asymmetry parameter γ for the GARCH(p, q) sequence.

7: **df** – INTEGER

The number of degrees of freedom for the Student's t -distribution.

If **dist** = 'N', **df** is not referenced.

Constraint: if **dist** = 'T', **df** > 2 .

8: **fcall** – LOGICAL

If **fcall** = *true*, a new sequence is to be generated, otherwise a given sequence is to be continued using the information in **r**.

9: **r**(**lr**) – REAL (KIND=nag_wp) array

The array contains information required to continue a sequence if **fcall** = *false*.

10: **state**(:) – INTEGER array

Note: the actual argument supplied **must** be the array **state** supplied to the initialization routines `nag_rand_init_repeat` (g05kf) or `nag_rand_init_nonrepeat` (g05kg).

Contains information on the selected base generator and its current state.

5.2 Optional Input Parameters

1: **lr** – INTEGER

Default: the dimension of the array **r**.

The dimension of the array **r**.

Constraint: $lr \geq 2 \times (ip + iq + 2)$.

5.3 Output Parameters

1: **ht(num)** – REAL (KIND=nag_wp) array

The conditional variances h_t , for $t = 1, 2, \dots, T$, for the GARCH(p, q) sequence.

2: **et(num)** – REAL (KIND=nag_wp) array

The observations ϵ_t , for $t = 1, 2, \dots, T$, for the GARCH(p, q) sequence.

3: **r(lr)** – REAL (KIND=nag_wp) array

Contains information that can be used in a subsequent call of nag_rand_times_garch_asym2 (g05pe), with **fcall** = *false*.

4: **state(:)** – INTEGER array

Contains updated information on the state of the generator.

5: **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, **dist** is not valid.

ifail = 2

Constraint: **num** ≥ 0 .

ifail = 3

Constraint: **ip** ≥ 0 .

ifail = 4

Constraint: **iq** ≥ 1 .

ifail = 5

Constraint: **theta**(i) ≥ 0.0 .

ifail = 7

Constraint: **df** ≥ 3 .

ifail = 11

ip or **iq** is not the same as when **r** was set up in a previous call.

ifail = 12

On entry, **lr** is not large enough, **lr** = $\langle value \rangle$: minimum length required .

ifail = 13

On entry, **state** vector has been corrupted or not initialized.

ifail = 51

On entry, sum of **theta**(i), for $i = 2, 3, \dots, ip + iq + 1$ is ≥ 1.0 .

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

Not applicable.

8 Further Comments

None.

9 Example

This example first calls `nag_rand_init_repeat` (g05kf) to initialize a base generator then calls `nag_rand_times_garch_asym2` (g05pe) to generate two realizations, each consisting of ten observations, from an asymmetric GARCH(1,1) model.

9.1 Program Text

```
function g05pe_example

fprintf('g05pe example results\n\n');

% Initialize the generator to a repeatable sequence
seed = [nag_int(1762543)];
genid = nag_int(1);
subid = nag_int(1);
[state, ifail] = g05kf( ...
                    genid, subid, seed);

% Input parameters
dist = 'N';
num = nag_int(10);
ip = nag_int(1);
iq = nag_int(1);
theta = [0.08; 0.2; 0.7];
gamma = -0.4;
df = nag_int(0);
fcall = true;
r = zeros(2*(ip+iq+2),1);

% Generate the first realisation
[ht, et, r, state, ifail] = ...
    g05pe( ...
```

```

        dist, num, ip, iq, theta, gamma, df, fcall, r, state);

% Display the results
fprintf('\n Realisation Number 1\n');
fprintf('   i           ht(i)           et(i)\n');
fprintf('-----\n');
for i=1:num
    fprintf('  %2d  %16.4f %16.4f\n', i, ht(i), et(i));
end

% Generate a second realisation
fcall = false;
[ht, et, r, state, ifail] = ...
    g05pe( ...
        dist, num, ip, iq, theta, gamma, df, fcall, r, state);

% Display the results
fprintf('\n Realisation Number 2\n');
fprintf('   i           ht(i)           et(i)\n');
fprintf('-----\n');
for i=1:num
    fprintf('  %2d  %16.4f %16.4f\n', i, ht(i), et(i));
end

```

9.2 Program Results

g05pe example results

Realisation Number 1		
i	ht(i)	et(i)

1	0.6400	0.2790
2	0.5336	-0.9098
3	0.7780	0.5840
4	0.6491	0.6731
5	0.5670	-0.9456
6	0.8275	-0.0172
7	0.6593	-0.2390
8	0.5639	0.5980
9	0.5005	-0.0032
10	0.4303	0.2917

Realisation Number 2		
i	ht(i)	et(i)

1	0.3874	-1.0205
2	0.7594	-0.5659
3	0.7371	0.2709
4	0.6013	-1.2499
5	1.1133	0.2505
6	0.8638	-0.5457
7	0.8014	-0.6395
8	0.8013	2.2341
9	1.0003	1.2908
10	0.9002	0.0727
