1 Purpose

nag_prob_dickey_fuller_unit (g01ewc) returns the probability associated with the lower tail of the distribution for the Dickey–Fuller unit root test statistic.

2 Specification

```c
#include <nag.h>
#include <nagg01.h>
double nag_prob_dickey_fuller_unit (Nag_TS_URProbMethod method,
               Nag_TS_URTestType type, Integer n, double ts, Integer nsamp,
               Integer state[], NagError *fail)
```

3 Description

If the root of the characteristic equation for a time series is one then that series is said to have a unit root. Such series are nonstationary. nag_prob_dickey_fuller_unit (g01ewc) is designed to be called after nag_tsa_dickey_fuller_unit (g13awc) and returns the probability associated with one of three types of (augmented) Dickey–Fuller test statistic: \( \tau \), \( \tau_\mu \) or \( \tau_r \), used to test for a unit root, a unit root with drift or a unit root with drift and a deterministic time trend, respectively. The three types of test statistic are constructed as follows:

1. To test whether a time series, \( y_t \), for \( t = 1, 2, \ldots, n \), has a unit root the regression model

   \[
   \nabla y_t = \beta_1 y_{t-1} + \sum_{i=1}^{p-1} \delta_i \nabla y_{t-i} + \epsilon_t
   \]

   is fit and the test statistic \( \tau \) constructed as

   \[
   \tau = \frac{\hat{\beta}_1}{\sigma_{11}}
   \]

   where \( \nabla \) is the difference operator, with \( \nabla y_t = y_t - y_{t-1} \), and where \( \hat{\beta}_1 \) and \( \sigma_{11} \) are the least squares estimate and associated standard error for \( \beta_1 \) respectively.

2. To test for a unit root with drift the regression model

   \[
   \nabla y_t = \beta_1 y_{t-1} + \sum_{i=1}^{p-1} \delta_i \nabla y_{t-i} + \alpha + \epsilon_t
   \]

   is fit and the test statistic \( \tau_\mu \) constructed as

   \[
   \tau_\mu = \frac{\hat{\beta}_1}{\sigma_{11}}
   \]

3. To test for a unit root with drift and deterministic time trend the regression model

   \[
   \nabla y_t = \beta_1 y_{t-1} + \sum_{i=1}^{p-1} \delta_i \nabla y_{t-i} + \alpha + \beta_2 t + \epsilon_t
   \]

   is fit and the test statistic \( \tau_r \) constructed as

   \[
   \tau_r = \frac{\hat{\beta}_1}{\sigma_{11}}
   \]
All three test statistics: $\tau$, $\tau_d$, and $\tau_c$ can be calculated using `nag_tsa_dickey_fuller_unit (g13awc)`. The probability distributions of these statistics are nonstandard and are a function of the length of the series of interest, $n$. The probability associated with a given test statistic, for a given $n$, can therefore only be calculated by simulation as described in Dickey and Fuller (1979). However, such simulations require a significant number of iterations and are therefore prohibitively expensive in terms of the time taken. As such `nag_prob_dickey_fuller_unit (g01ewc)` also allows the probability to be interpolated from a look-up table. Two such tables are provided, one from Dickey (1976) and one constructed as described in Section 9. The three different methods of obtaining an estimate of the probability can be chosen via the `method` argument. Unless there is a specific reason for choosing otherwise, `method = Nag_ViaLookUp` should be used.

4 References

Dickey A D (1976) Estimation and hypothesis testing in nonstationary time series PhD Thesis Iowa State University, Ames, Iowa


5 Arguments

1: `method` -- Nag_TS_URProbMethod

   On entry: the method used to calculate the probability.

   `method = Nag_ViaLookUp`
   The probability is interpolated from a look-up table, whose values were obtained via simulation.

   `method = Nag_ViaLookUpOriginal`
   The probability is interpolated from a look-up table, whose values were obtained from Dickey (1976).

   `method = Nag_ViaSimulation`
   The probability is obtained via simulation.

   The probability calculated from the look-up table should give sufficient accuracy for most applications.

   Suggested value: `method = Nag_ViaLookUp`.

   Constraint: `method = Nag_ViaLookUp, Nag_ViaLookUpOriginal or Nag_ViaSimulation`.

2: `type` -- Nag_TS_URTestType

   On entry: the type of test statistic, supplied in `ts`.

   Constraint: `type = Nag_UnitRoot, Nag_UnitRootWithDrift or Nag_UnitRootWithDriftAndTrend`.

3: `n` -- Integer

   On entry: $n$, the length of the time series used to calculate the test statistic.

   Constraints:

   if `method ≠ Nag_ViaSimulation`, $n > 0$;
   if `method = Nag_ViaSimulation` and `type = Nag_UnitRoot`, $n > 2$;
   if `method = Nag_ViaSimulation` and `type = Nag_UnitRootWithDrift`, $n > 3$;
   if `method = Nag_ViaSimulation` and `type = Nag_UnitRootWithDriftAndTrend`, $n > 4$. 

4: ts – double

\textit{On entry:} the Dickey–Fuller test statistic for which the probability is required. If
\begin{align*}
type &= \text{Nag\_UnitRoot} \\
\text{ts must contain } \tau.
\end{align*}
\begin{align*}
type &= \text{Nag\_UnitRootWithDrift} \\
\text{ts must contain } \tau_\mu.
\end{align*}
\begin{align*}
type &= \text{Nag\_UnitRootWithDriftAndTrend} \\
\text{ts must contain } \tau_\tau.
\end{align*}
If the test statistic was calculated using \textit{nag\_tsa\_dickey\_fuller\_unit} (g13awc) the value of \textit{type} and \textit{n} must not change between calls to \textit{nag\_prob\_dickey\_fuller\_unit} (g01ewc) and \textit{nag\_tsa\_dickey\_fuller\_unit} (g13awc).

5: nsamp – Integer

\textit{On entry:} if \textit{method} = \text{Nag\_ViaSimulation}, the number of samples used in the simulation; otherwise \textit{nsamp} is not referenced and need not be set.

\textit{Constraint:} if \textit{method} = \text{Nag\_ViaSimulation}, \textit{nsamp} \geq 0.

6: state[dim] – Integer

\textit{Communication Array}

\textit{Note:} the dimension, \textit{dim}, of this array is dictated by the requirements of associated functions that must have been previously called. This array MUST be the same array passed as argument \textit{state} in the previous call to \textit{nag\_rand\_init\_repeatable} (g05kfc) or \textit{nag\_rand\_init\_nonrepeatable} (g05kge).

\textit{On entry:} if \textit{method} = \text{Nag\_ViaSimulation}, \textit{state} must contain information on the selected base generator and its current state; otherwise \textit{state} is not referenced and may be \textit{NULL}.

\textit{On exit:} if \textit{method} = \text{Nag\_ViaSimulation}, \textit{state} contains updated information on the state of the generator otherwise a zero length vector is returned.

7: fail – NagError *

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE\_ALLOC\_FAIL}

Dynamic memory allocation failed.

See Section 3.2.1.2 in the Essential Introduction for further information.

\textbf{NE\_BAD\_PARAM}

On entry, argument \langle value \rangle had an illegal value.

\textbf{NE\_INT}

On entry, \textit{n} = \langle value \rangle.

\textit{Constraint:} if \textit{method} \neq \text{Nag\_ViaSimulation}, \textit{n} > 0.

On entry, \textit{n} = \langle value \rangle.

\textit{Constraint:} if \textit{method} = \text{Nag\_ViaSimulation} and \textit{type} = \text{Nag\_UnitRoot}, \textit{n} > 2.

On entry, \textit{n} = \langle value \rangle.

\textit{Constraint:} if \textit{method} = \text{Nag\_ViaSimulation} and \textit{type} = \text{Nag\_UnitRootWithDriftAndTrend}, \textit{n} > 4.

On entry, \textit{n} = \langle value \rangle.

\textit{Constraint:} if \textit{method} = \text{Nag\_ViaSimulation} and \textit{type} = \text{Nag\_UnitRootWithDrift}, \textit{n} > 3.
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 contact NAG for assistance.
An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

NE_INVALID_STATE
On entry, method = Nag_ViaSimulation and the state vector has been corrupted or not initialized.

NE_NO_LICENCE
Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

NE_SAMPLE
On entry, nsamp = (value).
Constraint: if method = Nag_ViaSimulation, nsamp > 0.

NW_EXTRAPOLATION
The supplied input values were outside the range of at least one look-up table, therefore
extrapolation was used.

7 Accuracy
When method = Nag_ViaLookUp, the probability returned by this function is unlikely to be accurate to
more than 4 or 5 decimal places, for method = Nag_ViaLookUpOriginal this accuracy is likely to drop
to 2 or 3 decimal places (see Section 9 for details on how these probabilities are constructed). In both
cases the accuracy of the probability is likely to be lower when extrapolation is used, particularly for
small values of n (less than around 15). When method = Nag_ViaSimulation the accuracy of the
returned probability is controlled by the number of simulations performed (i.e., the value of nsamp
used).

8 Parallelism and Performance
Not applicable.

9 Further Comments
When method = Nag_ViaLookUp or Nag_ViaLookUpOriginal the probability returned is constructed by interpolating from a series of look-up tables. In the case of method = Nag_ViaLookUpOriginal the look-up tables are taken directly from Dickey (1976) and the interpolation is carried out using
nag_2d_triang_interp (e01sjc) and nag_2d_triang_eval (e01skc). For method = Nag_ViaLookUp the
look-up tables were constructed as follows:
(i) A sample size, n was chosen.
(ii) $2^{28}$ simulations were run.
(iii) At each simulation, a time series was constructed as described in chapter five of Dickey (1976). The relevant test statistic was then calculated for each of these time series.
(iv) A series of quantiles were calculated from the sample of $2^{28}$ test statistics. The quantiles were calculated at intervals of 0.0005 between 0.0005 and 0.9995.
(v) A spline was fit to the quantiles using nag_1d_spline_fit (e02bec).
This process was repeated for $n = 25, 50, 75, 100, 150, 200, 250, 300, 350, 400, 450, 500, 600, 700, 800,
900, 1000, 1500, 2000, 2500, 5000, 10000$, resulting in 22 splines.
Given the 22 splines, and a user-supplied sample size, \( n \) and test statistic, \( \tau \), an estimated \( p \)-value is calculated as follows:

(i) Evaluate each of the 22 splines, at \( \tau \), using \texttt{nag_1d_spline_fit (e02bec)}. If, for a particular spline, the supplied value of \( \tau \) lies outside of the range of the simulated data, then a third-order Taylor expansion is used to extrapolate, with the derivatives being calculated using \texttt{nag_1d_spline_deriv (e02bcc)}.

(ii) Fit a spline through these 22 points using \texttt{nag_monotonic_interpolant (e01bec)}.

(iii) Estimate the \( p \)-value using \texttt{nag_monotonic_evaluate (e01bfc)}.

10 Example

See Section 10 in \texttt{nag_tsa_dickey_fuller_unit (g13awc)}.