NAG Library Function Document

nag_prob_durbin_watson (g01epc)

1 Purpose

nag_prob_durbin_watson (g01epc) calculates upper and lower bounds for the significance of a Durbin–Watson statistic.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
void nag_prob_durbin_watson (Integer n, Integer ip, double d, double *pdl,
     double *pdu, NagError *fail)
```

3 Description

Let $r = (r_1, r_2, \ldots, r_n)^{\mathrm{T}}$ be the residuals from a linear regression of y on p independent variables, including the mean, where the y values $y_1, y_2, \ldots, y_n$ can be considered as a time series. The Durbin–Watson test (see Durbin and Watson (1950), Durbin and Watson (1951) and Durbin and Watson (1971)) can be used to test for serial correlation in the error term in the regression.

The Durbin–Watson test statistic is:

$$d = \frac{\sum_{i=1}^{n-1}(r_{i+1} - r_i)^2}{\sum_{i=1}^{n} r_i^2},$$

which can be written as

$$d = \frac{r^{\mathrm{T}} A r}{r^{\mathrm{T}} r},$$

where the n by n matrix A is given by

$$A = \begin{bmatrix} 1 & -1 & 0 & \ldots & : \\ -1 & 2 & -1 & \ldots & : \\ 0 & -1 & 2 & \ldots & : \\ : & 0 & -1 & \ldots & : \\ : & : & : & \ldots & : \\ : & : & : & \ldots & -1 \\ 0 & 0 & 0 & \ldots & 1 \end{bmatrix}$$

with the nonzero eigenvalues of the matrix A being $\lambda_j = (1 - \cos(\pi j/n))$, for $j = 1, 2, \ldots, n-1$.

Durbin and Watson show that the exact distribution of d depends on the eigenvalues of a matrix HA, where H is the hat matrix of independent variables, i.e., the matrix such that the vector of fitted values, $\hat{y}$, can be written as $\hat{y} = Hy$. However, bounds on the distribution can be obtained, the lower bound being

$$d_l = \frac{\sum_{i=1}^{n-p} \lambda_i u_i^2}{\sum_{i=1}^{n-p} u_i^2}$$

and the upper bound being

$$d_u = \frac{\sum_{i=1}^{n-p} \lambda_{i-1+p} u_i^2}{\sum_{i=1}^{n-p} u_i^2},$$

where u_i are independent standard Normal variables.

Two algorithms are used to compute the lower tail (significance level) probabilities, p_l and p_u, associated with d_l and d_u. If $n \leq 60$ the procedure due to Pan (1964) is used, see Farebrother (1980), otherwise Imhof's method (see Imhof (1961)) is used.

The bounds are for the usual test of positive correlation; if a test of negative correlation is required the value of d should be replaced by $4 - d$.

4 References

Durbin J and Watson G S (1950) Testing for serial correlation in least squares regression. I *Biometrika* **37** 409–428

Durbin J and Watson G S (1951) Testing for serial correlation in least squares regression. II *Biometrika* **38** 159–178

Durbin J and Watson G S (1971) Testing for serial correlation in least squares regression. III *Biometrika* **58** 1–19

Farebrother R W (1980) Algorithm AS 153. Pan's procedure for the tail probabilities of the Durbin–Watson statistic *Appl. Statist.* **29** 224–227

Imhof J P (1961) Computing the distribution of quadratic forms in Normal variables *Biometrika* **48** 419–426

Newbold P (1988) *Statistics for Business and Economics* Prentice–Hall

Pan Jie–Jian (1964) Distributions of the noncircular serial correlation coefficients *Shuxue Jinzhan* **7** 328–337

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of observations used in calculating the Durbin–Watson statistic.

Constraint: **n** > **ip**.

2: **ip** – Integer *Input*

On entry: p, the number of independent variables in the regression model, including the mean.

Constraint: **ip** ≥ 1.

3: **d** – double *Input*

On entry: d, the Durbin–Watson statistic.

Constraint: **d** ≥ 0.0.

4: **pdl** – double * *Output*

On exit: lower bound for the significance of the Durbin–Watson statistic, p_l.

5: **pdu** – double * *Output*

On exit: upper bound for the significance of the Durbin–Watson statistic, p_u.

6: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, **ip** = ⟨*value*⟩.
Constraint: **ip** ≥ 1.

NE_INT_2

On entry, **n** = ⟨*value*⟩ and **ip** = ⟨*value*⟩.
Constraint: **n** $>$ **ip**.

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 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NE_REAL

On entry, **d** = ⟨*value*⟩.
Constraint: **d** ≥ 0.0.

7 Accuracy

On successful exit at least 4 decimal places of accuracy are achieved.

8 Parallelism and Performance

nag_prob_durbin_watson (g01epc) is not threaded in any implementation.

9 Further Comments

If the exact probabilities are required, then the first $n - p$ eigenvalues of HA can be computed and nag_prob_lin_chi_sq (g01jdc) used to compute the required probabilities with **c** set to 0.0 and **d** to the Durbin–Watson statistic.

10 Example

The values of n, p and the Durbin–Watson statistic d are input and the bounds for the significance level calculated and printed.

10.1 Program Text

```
/* nag_prob_durbin_watson (g01epc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg01.h>

int main(void)
{

  /* Scalars */
  double d, pdl, pdu;
  Integer exit_status, ip, n;
  NagError fail;

  INIT_FAIL(fail);

  exit_status = 0;
  printf("nag_prob_durbin_watson (g01epc) Example Program Results\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%" NAG_IFMT "%lf%*[^\n] ", &n, &ip, &d);
#else
  scanf("%" NAG_IFMT "%" NAG_IFMT "%lf%*[^\n] ", &n, &ip, &d);
#endif

  /* nag_prob_durbin_watson (g01epc).
   * Computes bounds for the significance of a Durbin-Watson
   * statistic
   */
  nag_prob_durbin_watson(n, ip, d, &pdl, &pdu, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_prob_durbin_watson (g01epc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  printf("\n");
  printf(" Durbin-Watson statistic %10.4f\n\n", d);
  printf(" Probability for the lower bound = %10.4f\n", pdl);
  printf(" Probability for the upper bound = %10.4f\n", pdu);
END:
  return exit_status;
}
```

10.2 Program Data

```
nag_prob_durbin_watson (g01epc) Example Program Data
 10 2 0.9238
```

10.3 Program Results

```
nag_prob_durbin_watson (g01epc) Example Program Results

 Durbin-Watson statistic     0.9238

 Probability for the lower bound =     0.0610
 Probability for the upper bound =     0.0060
```