Algorithm AS 181

The *W* Test for Normality

By J. P. Royston

MRC Clinical Research Centre, Harrow HA1 3UJ, Middx, UK

[Received February 1981. Revised December 1981]

Keywords: OMNIBUS TEST FOR NORMALITY; NORMALIZING TRANSFORMATION; SHAPIRO AND WILK'S TEST; ANALYSIS OF VARIANCE TEST FOR NORMALITY

Language

Fortran 66

Description and Purpose

Shapiro and Wilk's (1965) *W* statistic has been shown to provide a superior omnibus test of normality (Pearson *et al.*, 1977). It has recently been extended to cope with samples of size up to 2000 (Royston, 1982a). The purpose of the present algorithm is to enable the calculation of *W* and its significance level for any sample size between 3 and 2000.

The full description of the theory behind this algorithm is given by Royston (1982a). Using Monte Carlo simulation, Royston (1982a) showed that the transformation

$$y = (1-W)^{\lambda} \tag{1}$$

yielded a variable y with approximately normal distribution. The transformation (1) was adequate for sample sizes $n = 7-2000$. The parameter λ was estimated for 50 selected sample sizes and then smoothed with polynomials in $\log_e(n)-d$, *where* $d = 3$ for $7 \leqslant n \leqslant 20$ and $d = 5$ for $21 \leqslant n \leqslant 2000$.

The mean μ_y and s.d. σ_y of the transforms y were calculated using the smoothed λ's, and their logarithms were themselves smoothed with polynomials in $\log(n)-d$. Given a value of W, therefore, its significance level is calculated by referring the quantity

$$z = [(1-W)^{\lambda} - \mu_y]/\sigma_y$$

to the upper tail of the standard normal distribution, since large values of z indicate non-normality of the original sample.

The significance level of W for $n = 3$ is exact, and for $4 \leqslant n \leqslant 6$ is calculated by adapting Table 1 of Wilk and Shapiro (1968). Full details of all procedures are given by Royston (1982a).

Structure

SUBROUTINE WEXT (*X*, *N*, *SSQ*, *A*, *N*2, *EPS*, *W*, *PW*, *IFAULT*)

Formal parameters

X	Real array (*N*)	input: ordered sample values
N	Integer	input: sample size
SSQ	Real	input: sum of squares of data about mean
A	Real array (*N*2)	input: coefficients for calculation of *W*, set by *WCOEF*
*N*2	Integer	input: $[N/2]$, i.e. $\frac{1}{2}N$ if N is even, $\frac{1}{2}(N-1)$ if N is odd
EPS	Real	input: minimum possible value of *W*, set by *WCOEF*
W	Real	output: *W* statistic
PW	Real	output: significance level of *W*
IFAULT	Integer	output: fault indicator, equal to 3 if $N2 \neq [N/2]$ 2 if $N > 2000$ 1 if $N \leqslant 2$ 0 otherwise

 0035–9254/82/31176 $2.00

SUBROUTINE WCOEF (*A*, *N*, *N*2, *EPS*, *IFAULT*)

Formal parameters

A	Real array (*N*2)	output: coefficients for calculation of *W*
N	Integer	input: sample size
*N*2	Integer	input: $[N/2]$, i.e. $\frac{1}{2}N$ if N is even, $\frac{1}{2}(N-1)$ if N is odd
EPS	Real	output: minimum possible value of *W*
IFAULT	Integer	output: fault indicator, equal to 3 if $N2 \neq [N/2]$ 2 if $N > 2000$ 1 if $N \leqslant 2$ 0 otherwise

WCOEF must be called once for a given sample size before *WEXT* is called.

Failure indications

No calculations are performed by either *WCOEF* or *WEXT* unless $IFAULT = 0$. The observations $X(N)$ should be placed into either ascending or descending order before *WEXT* is used, but there is no check that this has been done.

Auxiliary algorithms

The following auxiliary routines are required:
FUNCTION POLY (*C*, *NORD*, *X*)—supplied below.
FUNCTION ALNORM (*X*, *UPPER*)—Algorithm AS 66 (Hill, 1973).
*SUBROUTINE NSCOR*2 (*A*, *N*, *N*2, *IFAULT*)—Algorithm AS 177 (Royston, 1982b).

Restrictions

W cannot be evaluated for sample sizes outside the range $3 \leqslant N \leqslant 2000$. For samples of size 4 to 6, a significance level of *W* below 0·0002 or above 0·9998 is set to 0 or 1 respectively.

Precision

Fortran single precision should be adequate on all machines with 32-bit arithmetic. The user should ensure that the corrected sum of squares *SSQ* has been calculated sufficiently accurately (e.g. to six significant figures).

Additional Comments

The time required to calculate *W* for a large sample will mainly depend on the speed of the routine used to sort the sample values, which is not, of course, part of the present algorithm. For heavy use of this algorithm, therefore, an efficient sorting routine may be a practical necessity.

It is recommended that *WEXT* be used in conjunction with a routine to give a normal plot of the data.

References

Hill, I. D. (1973). Algorithm AS 66. The normal integral. *Appl. Statist.*, **22**, 424–427.

Pearson, E. S., D'Agostino, R. B. and Bowman, K. O. (1977). Tests for departure from normality: comparison of powers. *Biometrika*, **64**, 231–246.

Royston, J. P. (1982a). An extension of Shapiro and Wilk's *W* test for normality to large samples. *Appl. Statist.*, **31**, 115–124.

——(1982b). Algorithm AS 177. Expected normal order statistics (exact and approximate). *Appl. Statist.*, **31**, 161–165.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality. *Biometrika*, **52**, 591–611.

Wilk, M. B. and Shapiro, S. S. (1968). The joint assessment of normality of several independent samples. *Technometrics*, **10**, 825–839.

```
      SUBROUTINE WEXT(X, N, SSQ, A, N2, EPS, W, PW, IFAULT)
C
C        ALGORITHM AS 181  APPL. STATIST. (1982) VOL.31, NO.2
C
C        CALCULATES SHAPIRO AND WILK W STATISTIC AND ITS SIG. LEVEL
C
      REAL X(N), A(N2), LAMDA, WA(3), WB(4), WC(4), WD(6), WE(6), WF(7),
     *  C1(5, 3), C2(5, 3), C(5), UNL(3), UNH(3)
      INTEGER NC1(3), NC2(3)
      LOGICAL UPPER
      DATA WA(1), WA(2), WA(3)
     *  /0.118898, 0.133414, 0.327907/,
     *     WB(1), WB(2), WB(3), WB(4)
     *  /-0.37542, -0.492145, -1.124332, -0.199422/,
     *     WC(1), WC(2), WC(3), WC(4)
     *  /-3.15805, 0.729399, 3.01855, 1.558776/,
     *     WD(1), WD(2), WD(3), WD(4), WD(5), WD(6)
     *  /0.480385, 0.318828, 0.0, -0.0241665, 0.00879701, 0.002989646/,
     *     WE(1), WE(2), WE(3), WE(4), WE(5), WE(6)
     *  /-1.91487, -1.37888, -0.04183209, 0.1066339, -0.03513666,
     *   -0.01504614/,
     *     WF(1), WF(2), WF(3), WF(4), WF(5), WF(6), WF(7)
     *  /-3.73538, -1.015807, -0.331885, 0.1773538, -0.01638782,
     *   -0.03215018, 0.003852646/
      DATA C1(1, 1), C1(2, 1), C1(3, 1), C1(4, 1), C1(5, 1),
     *     C1(1, 2), C1(2, 2), C1(3, 2), C1(4, 2), C1(5, 2),
     *     C1(1, 3), C1(2, 3), C1(3, 3), C1(4, 3), C1(5, 3) /
     *     -1.26233, 1.87969, 0.0649583, -0.0475604, -0.0139682,
     *     -2.28135, 2.26186, 0.0, 0.0, -0.00865763,
     *     -3.30623, 2.76287, -0.83484, 1.20857, -0.507590/
      DATA C2(1, 1), C2(2, 1), C2(3, 1), C2(4, 1), C2(5, 1),
     *     C2(1, 2), C2(2, 2), C2(3, 2), C2(4, 2), C2(5, 2),
     *     C2(1, 3), C2(2, 3), C2(3, 3), C2(4, 3), C2(5, 3) /
     *     -0.287696, 1.78953, -0.180114, 0.0, 0.0,
     *     -1.63638, 5.60924, -3.63738, 1.08439, 0.0,
     *     -5.991908, 21.04575, -24.58061, 13.78661, -2.835295/
      DATA UNL(1), UNL(2), UNL(3) /-3.8, -3.0, -1.0/,
     *     UNH(1), UNH(2), UNH(3) / 8.6,  5.8,  5.4/
      DATA NC1(1), NC1(2), NC1(3) /5, 5, 5/,
     *     NC2(1), NC2(2), NC2(3) /3, 4, 5/
      DATA PI6 /1.90985932/, STQR /1.04719755/, UPPER /.TRUE./,
     *  ZERO /0.0/, TQR /0.75/, ONE /1.0/, ONEPT4 /1.4/, THREE /3.0/,
     *  FIVE /5.0/
      IFAULT = 1
      PW = ONE
      W = ONE
      IF (N .LE. 2) RETURN
      IFAULT = 3
      IF (N / 2 .NE. N2) RETURN
      IFAULT = 2
      IF (N .GT. 2000) RETURN
C
C        CALCULATE W
C
      IFAULT = 0
      W = ZERO
      AN = N
      I = N
      DO 10 J = 1, N2
      W = W + A(J) * (X(I) - X(J))
      I = I - 1
   10 CONTINUE
      W = W * W / SSQ
      IF (W .LT. ONE) GOTO 20
      W = ONE
      RETURN
C
C        GET SIGNIFICANCE LEVEL OF W
C
   20 IF (N .LE. 6) GOTO 100
C
C        N BETWEEN 7 AND 2000 ... TRANSFORM W TO Y, GET MEAN AND SD,
C        STANDARDIZE AND GET SIGNIFICANCE LEVEL
```

```
      IF (N .GT. 20) GOTO 30
      AL = ALOG(AN) - THREE
      LAMDA = POLY(WA, 3, AL)
      YBAR = EXP(POLY(WB, 4, AL))
      SDY = EXP(POLY(WC, 4, AL))
      GOTO 40
   30 AL = ALOG(AN) - FIVE
      LAMDA = POLY(WD, 6, AL)
      YBAR = EXP(POLY(WE, 7, AL))
      SDY = EXP(POLY(WF, 7, AL))
   40 Y = (ONE - W) ** LAMDA
      Z = (Y - YBAR) / SDY
      PW = ALNORM(Z, UPPER)
      RETURN
C
C        DEAL WITH N LESS THAN 7 (EXACT SIGNIFICANCE LEVEL FOR N=3).
C
  100 IF (W .LE. EPS) GOTO 160
      WW = W
      IF (N .EQ. 3) GOTO 150
      UN = ALOG((W - EPS) / (ONE - W))
      N3 = N - 3
      IF (UN .LT. UNL(N3)) GOTO 160
      IF (UN .GE. ONEPT4) GOTO 120
      NC = NC1(N3)
      DO 110 I = 1, NC
  110 C(I) = C1(I, N3)
      EU3 = EXP(POLY(C, NC, UN))
      GOTO 140
  120 IF (UN .GT. UNH(N3)) RETURN
      NC = NC2(N3)
      DO 130 I = 1, NC
  130 C(I) = C2(I, N3)
      UN = ALOG(UN)
      EU3 = EXP(EXP(POLY(C, NC, UN)))
  140 WW = (EU3 + TQR) / (ONE + EU3)
  150 PW = PI6 * (ATAN(SQRT(WW / (1.0 - WW))) - STQR)
      RETURN
  160 PW = ZERO
      RETURN
      END
C
      SUBROUTINE WCOEF(A, N, N2, EPS, IFAULT)
C
C        ALGORITHM AS 181.1  APPL. STATIST. (1982) VOL.31, NO.2
C
C        OBTAIN ARRAY A OF WEIGHTS FOR CALCULATING W
C
      REAL A(N2), C4(2), C5(2), C6(3)
      DATA C4(1), C4(2) /0.6869, 0.1678/, C5(1), C5(2) /0.6647, 0.2412/,
     *  C6(1), C6(2), C6(3) /0.6431, 0.2806, 0.0875/
      DATA RSQRT2 /0.70710678/, ZERO /0.0/, HALF /0.5/, ONE /1.0/,
     *  TWO /2.0/, SIX /6.0/, SEVEN /7.0/, EIGHT /8.0/, THIRT /13.0/
      IFAULT = 1
      IF (N .LE. 2) RETURN
      IFAULT = 3
      IF (N / 2 .NE. N2) RETURN
      IFAULT = 2
      IF (N .GT. 2000) RETURN
      IFAULT = 0
      IF (N .LE. 6) GOTO 30
C
C        N .GT. 6 CALCULATE RANKITS USING APPROXIMATE ROUTINE NSCOR2
C        (AS177)
C
      CALL NSCOR2(A, N, N2, IFAULT)
      SASTAR = ZERO
      DO 10 J = 2, N2
   10 SASTAR = SASTAR + A(J) * A(J)
      SASTAR = SASTAR * EIGHT
```

```
      NN = N
      IF (N .LE. 20) NN = NN - 1
      AN = NN
      A1SQ = EXP(ALOG(SIX * AN + SEVEN) - ALOG(SIX * AN + THIRT)
     *  + HALF * (ONE + (AN - TWO) * ALOG(AN + ONE) - (AN - ONE)
     *  * ALOG(AN + TWO)))
      A1STAR = SASTAR / (ONE / A1SQ - TWO)
      SASTAR = SQRT(SASTAR + TWO * A1STAR)
      A(1) = SQRT(A1STAR) / SASTAR
      DO 20 J = 2, N2
   20 A(J) = TWO * A(J) / SASTAR
      GOTO 70
C
C        N .LE. 6  USE EXACT VALUES FOR WEIGHTS
C
   30 A(1) = RSQRT2
      IF (N .EQ. 3) GOTO 70
      N3 = N - 3
      GOTO (40, 50, 60), N3
   40 DO 45 J = 1, 2
   45 A(J) = C4(J)
      GOTO 70
   50 DO 55 J = 1, 2
   55 A(J) = C5(J)
      GOTO 70
   60 DO 65 J = 1, 3
   65 A(J) = C6(J)
C
C        CALCULATE THE MINIMUM POSSIBLE VALUE OF W
C
   70 EPS = A(1) * A(1) / (ONE - ONE / FLOAT(N))
      RETURN
      END
C
      FUNCTION POLY(C, NORD, X)
C
C        ALGORITHM AS 181.2  APPL. STATIST. (1982) VOL.31, NO.2
C
C        CALCULATES THE ALGEBRAIC POLYNOMIAL OF ORDER NORD-1 WITH
C        ARRAY OF COEFFICIENTS C.  ZERO ORDER COEFFICIENT IS C(1).
C
      REAL C(NORD)
      POLY = C(1)
      IF (NORD .EQ. 1) RETURN
      P = X * C(NORD)
      IF (NORD .EQ. 2) GOTO 20
      N2 = NORD - 2
      J = N2 + 1
      DO 10 I = 1, N2
      P = (P + C(J)) * X
      J = J - 1
   10 CONTINUE
   20 POLY = POLY + P
      RETURN
      END
```

Algorithm AS 182

Finite Sample Prediction from ARIMA Processes

By A. C. HARVEY and C. R. MCKENZIE

London School of Economics, London WC2A 2AE, UK

[Received May 1981. Revised February 1982]

Keywords: AUTOREGRESSIVE INTEGRATED MOVING AVERAGE MODEL; KALMAN FILTER; FORECASTING; FINITE SAMPLE PREDICTION

 0035-9254/82/31180 $2.00