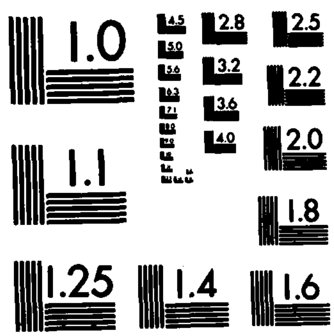
AD-A123 511   A FORTRAN PROGRAM FOR THE LEVEL PROBABILITIES OF ORDER RESTRICTED INFERENCE(U) IOWA UNIV IOWA CITY DEPT OF STATISTICS AND ACTURIAL SCIENCE.. C PILLERS ET AL.   1/1

UNCLASSIFIED   JUL 82 TR-87 N00014-80-C-0321   F/G 12/1   NL

END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

AD A123511

A FORTRAN PROGRAM FOR THE LEVEL PROBABILITIES OF ORDER RESTRICTED INFERENCE

Carolyn Pillers, Tim Robertson and F. T. Wright
University of Iowa and University of Missouri-Rolla

The University of Iowa
Department of Statistics and Actuarial Science

Technical Report No. 87

July, 1982

This work was supported by ONR Contract N00014-80-C-0321.

A FORTRAN PROGRAM FOR THE LEVEL PROBABILITIES OF ORDER RESTRICTED INFERENCE

Carolyn Pillers, Tim Robertson and F. T. Wright
University of Iowa and University of Missouri-Rolla

Key words: Order restricted inference, level probabilities, chi-bar-square distribution.

LANGUAGE

Fortran

DESCRIPTION AND PURPOSES

The level probabilities of order restricted inference are fundamental to that theory; their values are the probabilities that the order restricted, maximum likelihood estimates of normal means assume specified numbers of distinct values, called levels. Those probabilities are computed under the assumptions that the population means are equal and that the sampling from the various populations is done independently. The level probabilities depend upon a vector of weights (usually the various sample sizes) and the use of much of the theory of order restricted inference has been limited by the fact that those level probabilities can be virtually impossible to compute if the weights are not all equal.

Bohrer and Chow (1978; Algorithm AS 122) give a program for computing these level probabilities when the number, K, of populations is no more than 10. Their program uses an algorithm for computing orthant probabilities which is due to Milton (1972). The time needed to use Milton's algorithm increases exponentially in K and can require several minutes or more of computation when $K \geq 6$ (cf. Bohrer and Chow).

Cran (1981; Algorithm AS 158) gives a program for computing these level probabilities when the number K does not exceed 6. For $K = 5$ it uses an approximation due to Plackett (1954) and for $K = 6$ an approximation due to Childs (1967) is used.

Robertson and Wright (1982) develop an approximation which is based upon an idea of Chase (1974) and uses the pattern of large and small weights. We refer the interested reader to Robertson and Wright (1982) for an evaluation of the quality of this approximation. The Fortran program given below uses this approximation for the values of the level probabilities for K such that $K \geq 6$. For $K \leq 20$ and equal weights or for general weights and $K \leq 5$ the program is identical to Cran's (1981; Algorithm AS 158).

STRUCTURE

SUBROUTINE PROBS (K, W, P, IFAULT)

K	Integer	input:	the number of weights
W	Real Array (K)	input:	the original weights
P	Real Array (K)	output:	the computed probabilities
IFAULT	Integer	output:	a fault indicator, equal to 1 if at least one weight is not positive 2 if $K < 2$ or $K > 20$ 3 if an error occurred in function FACT 0 otherwise

Auxiliary Algorithms

FUNCTION PR1(I,J,W) computes explicitly the probabilities for $K \le 5$. (Algorithm AS 158.1)

FUNCTION F1(V1,V2,V3) computes the correlation $\rho = \left(\frac{V1 * V3}{(V1+V2)(V2+V3)}\right)^{-1/2}$. (Algorithm AS 158.2)

FUNCTION FACT(M,IFAULT) computes n factorial. (Algorithm AS 158.5)

SUBROUTINE CHASE(K,CH,P1) computes the equal weight probabilities and Chase's approximations for a given K.

SUBROUTINE PAPRX(K,W,PA,CH,P1) computes the approximate probabilities:

K	Integer	input:	the number of weights
W	Real Array(K)	input:	the original weights
PA	Real Array(K)	output:	the approximate probabilities
CH	Real Array(K,K)	input:	Chase's approximate probabilities
P1	Real Array(K,K)	input:	equal weight probabilities

Restrictions

The weight array can have no more than 20 elements, so $K \leq 20$. In addition, all weights must be positive.

REFERENCES

Barlow, R.E., Bartholomew, D.J., Bremner, J.M. and Brunk, H.D. (1972). Statistical Inference under Order Restrictions. London: Wiley.

Bohrer, Robert and Chow, Winston (1978). Algorithm AS 122. Weights for one-sided multivariate inference. Appl. Statist. 27, 100-104.

Childs, D.R. (1967). Reduction of the multivariate normal integral to characteristic form. Biometrika 54, 293-299.

Cran, G.W. (1981). Algorithm AS 158. Calculation of the probabilities $\{P(\ell,k)\}$ for the simply ordered alternative. Appl. Statist. 30, 85-91.

Milton, R.C. (1972). Computer evaluation of the multivariate normal integral. Technometrics 14, 881-890.

Plackett, R.L. (1954). A reduction formula for normal multivariate integrals. Biometrika 41, 351-360.

Robertson, Tim and Wright, F.T. (1982). On approximation of the level probabilities and associated distributions in order restricted inference. University of Iowa, Department of Statistics and Actuarial Science Technical Report No. 81.

```
      DIMENSION P(20), W(20)
```



```
      DIMENSION P(20), W(20)
C
C        DRIVER PROGRAM FOR SUBROUTINE PROBS.
C
      READ( 5, 10 ) K
10    FORMAT( I2 )
      READ( 5, 20 ) ( W(I), I = 1, K )
20    FORMAT( 10F8.4 )
C
C        ECHO CHECK THE INPUT VALUES.
C
      WRITE( 6, 30 )
30    FORMAT( 1H1 )
      WRITE( 6, 40 ) K
40    FORMAT( 4X, 10HTHERE ARE , I2, 9HWEIGHTS. )
      WRITE( 6, 50 )
50    FORMAT( 4X, 15HTHE WEIGHTS ARE )
      WRITE( 6, 60 ) ( W(I), I = 1, K )
60    FORMAT( 4X, 10F12.7 )
      CALL PROBS( K, W, P, IFAULT )
      IF ( IFAULT .EQ. 1 ) GOTO 100
      IF ( IFAULT .EQ. 2 ) GOTO 120
      IF ( IFAULT .EQ. 3 ) GOTO 140
C
C        OUTPUT THE COMPUTED PROBABILITIES.
C
      WRITE( 6, 70 )
70    FORMAT( //4X, 30HTHE COMPUTED PROBABILITIES ARE, / )
      DO 90 L = 1, K
      WRITE( 6, 80 ) L, K P(L)
80    FORMAT( 4X, 2HP(, I2, 1H, , I2, 4H) = , F10.7 )
90    CONTINUE
      STOP
100   WRITE( 6, 110 )
110   FORMAT( 4X, 35HAT LEAST ONE WEIGHT IS NOT POSITIVE )
      STOP
120   WRITE( 6, 130 )
130   FORMAT( 4X, 17HK IS OUT OF RANGE )
      STOP
140   WRITE( 6, 150 )
150   FORMAT( 4X, 34HAN ERROR OCCURRED IN FUNCTION FACT )
      STOP
      END
```

SUBROUTINE PROBS(K, W, P, IFAULT)


```
      SUBROUTINE PROBS( K, W, P, IFAULT )
C
C     CALCULATION OF THE PROBABILITIES P(L,K) FOR
C     THE CASE OF SIMPLE ORDER.
C
      DIMENSION W( 20 ), P( 20 ), Q( 20, 20 ), CH( 20, 20 )
      DIMENSION P1( 20, 20 ), PA( 20, 20 )
      DATA C1/ 1.0E-6 /
C
C        CHECK THAT WEIGHTS ARE POSITIVE.
C
      IFAULT = 0
      DO 1 I = 1, K
      IF ( W( I ) .LE. 0.0 ) GO TO 101
    1 CONTINUE
C
C        CHECK THAT K .GE. 3 AND .LE. 20
C
      IF ( K .LT. 3 .OR. K .GT. 20 ) GO TO 102
      WW = W( 1 )
      DO 2 I = 2, K
      IF ( ABS( WW - W( I ) ) .GT. C1 ) GO TO 7
    2 CONTINUE
C
C         EQUAL WEIGHTS
C
      Q( 1, 3 ) = 1.0 / 3.0
      Q( 2, 3 ) = 0.5
      Q( 3, 3 ) = 1.0 / 6.0
      IF ( K .EQ. 3 ) GO TO 5
      DO 4 J = 4, K
      AJ = J
      A1 = 1.0 / AJ
      A2 = ( AJ - 1.0 ) * A1
      Q( 1, J ) = A1
      J1 = J - 1
      DO 3 L = 2, J1
      L1 = L - 1
      Q( L, J ) = A1 * Q( L1, J1 ) + A2 * Q( L, J1 )
    3 CONTINUE
      Q( J, J ) = 1.0 / FACT( J, IFAULT )
      IF( IFAULT .NE. 0 ) RETURN
    4 CONTINUE
    5 CONTINUE
      DO 6 J = 1, K
      P( J ) = Q( J, K )
    6 CONTINUE
      RETURN
C
C        UNEQUAL WEIGHTS - CHECK THAT K .LE. 5
C
    7 IF ( K .GT. 5 ) GO TO 11
      K2 = K - 2
      GO TO ( 8, 9, 10 ), K2
    8 P( 1 ) = PR1( 1, 3, W )
      P( 2 ) = 0.5
      P( 3 ) = 0.5 - P( 1 )
      RETURN
    9 P( 1 ) = PR1( 1, 4, W )
      P( 4 ) = PR1( 4, 4, W )
```

```
      SUBROUTINE PROBS( K, W, P, IFAULT )
```



```
      P( 2 ) = 0.5 - P( 4 )
      P( 3 ) = 0.5 - P( 1 )
      RETURN
10    P( 5 ) = PR1( 3, 5, W )
      P( 4 ) = PR1( 4, 5, W )
      P( 2 ) = 0.5 - P( 4 )
      P( 1 ) = PR1( 1, 5, W )
      P( 3 ) = 0.5 - P( 1 ) - P( 5 )
      RETURN
11    CALL CHASE( K, CH, P1 )
      CALL PAPRX( K, W, PA, CH, P1 )
      DO 12 J = 1, K
      P( J ) = PA( J, K )
12    CONTINUE
      RETURN
101   IFAULT = 1
      RETURN
102   IFAULT = 2
      RETURN
      END
C
      FUNCTION PR1( I, J, W )
C
C        ALGORITHM AS 158.1, APPL STAT ( 1981 ), VOL 30, NO 1
C
C        EXPLICIT CALCULATION OF PROBABILITIES FOR K .LE. 5
C        ALSO CALLED BY FUNCTION F2
C
      DIMENSION W( 10 )
      DATA PII/ 0.318309886 /
      IF ( J .NE. 3 ) GO TO 40
      C = 0.5 * PII * F1( W( 1 ), W( 2 ), W( 3 ) )
      IF ( I .EQ. 3 ) GO TO 30
      PR1 = 0.25 - C
      RETURN
30    PR1 = 0.25 + C
      RETURN
40    W1 = W( 1 )
      W2 = W( 2 )
      W3 = W( 3 )
      W4 = W( 4 )
      W12 = W1 + W2
      W23 = W2 + W3
      W34 = W3 + W4
      S12 = F1( W1, W2, W3 )
      S23 = F1( W2, W3, W4 )
      IF ( J .EQ. 5 ) GO TO 50
      IF ( I .EQ. 4 ) GO TO 41
      C1 = 0.25 * PII *
     *  (F1(W1,W2,W34) + F1(W1,W23,W4 ) + F1(W12,W3,W4 ) )
      PR1 = 0.125 - C1
      RETURN
41    C2 = 0.25 * PII * ( S12 + S23 )
      PR1 = 0.125 + C2
      RETURN
50    W5 = W( 5 )
      W45 = W4 + W5
      W123 = W12 + W3
      W234 = W23 + W4
      W345 = W34 + W5
```

```
      SUBROUTINE PROBS( K, W, P, IFAULT )
```



```
      S34 = F1( W3, W4, W5 )
      IF ( I .EQ. 4 ) GO TO 52
      C5 = 0.0625 + 0.125 * PII * ( S12 + S23 + S34 ) +
     *  0.25 * PII * S12 * S34
      IF ( I .EQ. 1 ) GO TO 51
      PR1 = C5
      IF ( PR1 .LT. 0.0 ) PR1 = 0.0
      RETURN
51    S113 = F1( W1, W2, W345 )
      S131 = F1( W1, W234, W5 )
      S311 = F1( W123, W4, W5 )
      C3 = 0.375 + 0.125 * PII * (S113 + S131 + S311 + F1(W1, W23, W45)
     *   + F1(W12, W3, W45) + F1(W12, W34, W5) - S12 - S23 - S34 )
     *   - 0.25 * PII * PII * ( S12 * S311 + S23 * S131 + S34 * S113 )
      PR1 = 0.5 - C3 - C5
      RETURN
52    C2 = 0.125 * PII * (S12 + S34 + F1(W1, W2, W34) + F1(W12, W3, W4)
     *  + F1(W1, W23, W4) + F1(W2, W3, W45) + F1( W23, W4, W5) +
     *  F1(W2, W34, W5 ) )
      PR1 = 0.25 + C2
      RETURN
      END
C
      FUNCTION F1( V1, V2, V3 )
C
C        ALGORITHM AS 158.2 APPL STAT (1981) VOL 30, NO 1
C
      RHO = -SQRT( V1 * V3 / ( ( V1 + V2 ) * ( V2 + V3 ) ) )
      F1 = ASIN( RHO )
      RETURN
      END
C
      FUNCTION FACT( M, IFAULT )
C
C        ALGORITHM AS 159.5 APPL STAT ( 1981 ) VOL 30, NO 1
C
C        CALCULATION OF M FACTORIAL
C
      DATA MAXM/ 50 /
      IFAULT = 3
      IF ( M .LT. 0 .OR. M .GT. MAXM ) RETURN
      IFAULT = 0
      FACT = 1.0
      IF ( M .LE. 1 ) RETURN
      A1 = 1.0
      DO 1 I = 2, M
      AI = I
         A1 = A1 * AI
1     CONTINUE
      FACT = A1
      RETURN
      END
      FUNCTION ASIN( X )
      ASIN = ATAN( X / SQRT( 1.0 - X*X ) )
      RETURN
      END
      SUBROUTINE CHASE(K, CH, P )
C
C     THIS SUBROUTINE COMPUTES CHASE'S AND EQUAL WEIGHTS
C     PLK'S.
```

```
      SUBROUTINE PROBS( K, W, P, IFAULT )
C
      REAL CH(20,20),P(20,20)
C
C       INITIALIZE MATRICES TO ZERO.
C
      DO 20 J = 1, 20
      DO 10 I = 1, 20
      CH(I,J) = 0.0
      P(I,J) = 0.0
10    CONTINUE
20    CONTINUE
      CH(1,1) = 1.0
      CH(1,2) = 0.5
      CH(2,2) = 0.5
      P(1,1) = 1.0
      P(1,2) = 0.5
      P(2,2) = 0.5
      DO 40 J =2, K
      J1 = J + 1
      X = 2 * J - 1
      Y = 2 * J
      U = J
      V = J + 1
      CH(1,J1) = X / Y * CH(1,J)
      CH(J1,J1) = 1.0 / Y * CH(J,J)
      P(1,J1) = U / V * P(1,J)
      P(J1,J1) = 1.0 / V * P(J,J)
      DO 30 I = 2, J
      IA = I - 1
      CH(I,J1) = (1.0 / Y) * CH(IA,J) + (X / Y) * CH(I,J)
      P(I,J1) = (1.0 / V) * P(IA,J) + (U / V) * P(I,J)
30    CONTINUE
40    CONTINUE
      RETURN
      END
      SUBROUTINE PAPRX(K, W, PA, CH, P )
C
C     THIS SUBROUTINE COMPUTES THE APPROXIMATE PLK'S
C
      REAL W(20),PA(20,20),PB(20,20),PBP(20,20),CH(20,20), P(20,20)
      INTEGER INDEX(20),A,B
      ALPHA = 1.0 / 3.0
C
C     INITIALIZE PA,PB,PBP
C
      DO 20 J = 1, 20
      DO 10 I = 1, 20
      PA(I,J) = 0.0
      PB(I,J) = 0.0
      PBP(I,J) = 0.0
10    CONTINUE
20    CONTINUE
C        DETERMINE MAXIMUM AND MINIMUM OF WEIGHTS.
C
      WMAX = W( 1 )
      WMIN = W( 1 )
      DO 25 I = 2, K
      IF ( W( I ) .LT. WMIN ) WMIN = W( I )
      IF ( W( I ) .GT. WMAX ) WMAX = W( I )
25    CONTINUE
```

```
      SUBROUTINE PROBS( K, W, P, IFAULT )

      CUT = 0.65 * WMIN + 0.35 * WMAX
C
C     DETERMINE THE INDICES OF THE WEIGHTS
C
      DO 40 I = 1, K
      IF ( W(I) .LT. CUT ) GOTO 30
      INDEX(I) = 1
      GOTO 40
30    INDEX(I) = 0
40    CONTINUE
C
C     COMPUTE THE NUMBER OF LARGE WTS
C
      M = 0
      DO 50 I = 1, K
      M = M + INDEX( I )
50    CONTINUE
C
C    IF ALL WTS LARGE OR SMALL SET PA=P
C
      IF((M.NE.0).AND.(M.NE.K)) GO TO 70
      DO 60 L = 1, K
      PA(L,K) = P(L,K)
60    CONTINUE
      GO TO 270
C
C   IF A=0 AND B=0 SET PB=PLM
C
70    N = INDEX(1) + INDEX(K)
      IF (N .NE. 2) GO TO 90
      DO 80 L = 1, M
      PB(L,K) = P(L,M)
80    CONTINUE
      GO TO 240
C
C   DETERMINE A,B
C
90    A = 0
      B = 0
      DO 100 I = 1, K
      IF (INDEX(I) .NE. 0) GO TO 110
      A = A + 1
100   CONTINUE
110   DO 120 I = 1, K
      J = K - I + 1
      IF (INDEX(J) .NE. 0) GO TO 130
      B = B + 1
120   CONTINUE
130   N1 = A + 1
      N2 = B + 1
      IF (A .NE. 0) GO TO 160
      DO 150 L = 1, K
      SUM = 0.0
      DO 140 I = 1, L
      J = L - I + 1
      SUM = SUM + P(I,M) * CH(J,N2)
140   CONTINUE
      PB(L,K) = SUM
150   CONTINUE
      GO TO 240
```

```
      SUBROUTINE PROBS( K, W, P, IFAULT )

160   IF (R .NE. 0) GO TO 190
      DO 180 L = 1, K
      SUM = 0.0
      DO 170 I = 1, L
      J = L - I + 1
      SUM = SUM + P(I,M) * CH(J,N1)
170   CONTINUE
      PB(L,K) = SUM
180   CONTINUE
      GO TO 240
190   DO 210 L = 1, K
      SUM = 0.0
      DO 200 I = 1, L
      J = L - I + 1
      SUM = SUM + CH(I,N1) * CH(J,N2)
200   CONTINUE
      PBP(L,K) = SUM
210   CONTINUE
      DO 230 L = 1, K
      SUM = 0.0
      DO 220 I = 1, L
      J = L - I + 1
      SUM = SUM + PBP(I,K) * P(J,M)
220   CONTINUE
      PB(L,K) = SUM
230   CONTINUE
C
C     DETERMINE R
C
240   SUM1 = 0.0
      SUM2 = 0.0
      DO 250 I = 1, K
      X = INDEX(I)
      Y = 1 - INDEX(I)
      SUM1 = SUM1 + W(I) * X
      SUM2 = SUM2 + W(I) * Y
250   CONTINUE
      X = K - M
      Y = M
      R = (SUM1 * X)/(SUM2 * Y)
      DO 260 L = 1, K
      PA(L,K)=(1.-(1.0)/(R**ALPHA))*PB(L,K)+((1.0)/(R**ALPHA))*P(L,K)
260   CONTINUE
270   RETURN
      END
```

2-8

DT