5-2006

Programs to Compute Distribution Functions and Critical Values for Extreme Value Ratios for Outlier Detection

George McBane
Grand Valley State University, mcbaneg@gvsu.edu

Journal of Statistical Software

May 2006, Volume 16, Issue 3. http://www.jstatsoft.org/

Programs to Compute Distribution Functions and Critical Values for Extreme Value Ratios for Outlier Detection

George C. McBane
Grand Valley State University

Abstract

A set of FORTRAN subprograms is presented to compute density and cumulative distribution functions and critical values for the range ratio statistics of Dixon (1951, *The Annals of Mathematical Statistics*) These statistics are useful for detection of outliers in small samples.

Keywords: Dixon's r, outlier detection, gross errors, Q test.

1. Introduction

Dixon (1951) described a set of statistics for the purpose of detecting outliers in simple data sets so that they could be excluded from subsequent analysis. For a set of n observations x_i, ordered such that $x_1 \leq x_2 \leq \ldots \leq x_n$, the statistics are defined by $r_{j,i-1} = (x_n - x_{n-j})/(x_n - x_i)$. The first subscript on the r symbol indicates the number of outliers that are suspected at the upper end of the data set, and the second subscript indicates the number of outliers suspected at the lower end. Dixon gave analytic formulas for the density and cumulative distribution functions for r_{10}, r_{11}, r_{12}, r_{20}, r_{21}, and r_{22} for a few small values of n, and presented numerically generated tables of critical values for each of those statistics for $n \leq 30$.

Dixon (1950) described the use and performance of these "range ratio" statistics for the purpose of rejecting outliers, and Dixon (1950); Beckman and Cook (1983); Barnett and Lewis (1984) and Hampel (1985) compared them to other approaches to the same problem. Dixon's ratios and many other approaches share the weakness that the user must guess the number of outliers that exist in the data set, and the tests often fail if the guess is wrong. In other ways Dixon's tests perform as well as or better than most other tests and they are very easy to use. Dixon's tests have become a standard for outlier rejection in analytical chemistry and

are used in other fields as well. Dixon's r_{10} is also called Q, and the outlier rejection test using it is called the Q test in the chemistry literature.

I can find no indication in the literature that anyone has recomputed the numerical values in Dixon's tables in the 55 years since he published them. In fact, the literature gives the impression that the art of generating the critical values for his statistics has been lost. Rorabacher (1991), in a paper intended to provide definitive reference values for the analytical chemistry community and to add critical values for new confidence levels, interpolated in Dixon's tables. The **outliers** package (Komsta 2005) for version 2 of the R statistical environment (R Development Core Team 2005) similarly interpolates between Dixon's published values. Efstathiou (1992) estimated critical values stochastically and compared them to Dixon's tables.

Dixon gives enough information in his paper to permit recomputation of his results at higher precision. This article describes one numerical approach to that task and gives FORTRAN functions for the probability density, cumulative distribution function, and critical values for each of the statistics presented by Dixon. The functions should be easy to translate into other languages for incorporation into larger software packages.

2. Formulas for density and cumulative distribution functions

2.1. Probability density for r

Each of Dixon's ratios is a function of three of the n data values: $r_{j,i-1} = (x_n - x_{n-j})/(x_n - x_i)$. The joint probability density for those three variables is obtained by multiplying the density functions for all of the data values, integrating over the possible values of all the variables except the three being used, and applying a combinatorial normalization factor. There are $i-1$ observations below x_i, $n-j-i-1$ observations between x_i and x_{n-j}, and $j-1$ observations between x_{n-j} and x_n. The joint density for normally distributed data is therefore

$$P(x_i, x_{n-j}, x_n) = \frac{n!}{(i-1)!(n-j-i-1)!(j-1)!} \left[\int_{-\infty}^{x_i} \phi(t)\,dt\right]^{i-1} \left[\int_{x_i}^{x_{n-j}} \phi(t)\,dt\right]^{n-j-i-1} \times \left[\int_{x_{n-j}}^{x_n} \phi(t)\,dt\right]^{j-1} \phi(x_i)\phi(x_{n-j})\phi(x_n), \quad (1)$$

where $\phi(x) = (2\pi)^{-1/2}\exp(-x^2/2)$ is the density function for the standard normal distribution. Equation 1 in Dixon (1951) is the special case of Equation 1 for r_{10}, that is, for $j = i = 1$. Following Dixon, I simply write r when there is no ambiguity.

Dixon then changes variables from $\{x_i, x_{n-j}, x_n\}$ to $\{x, v, r\}$: $x = x_n$, $v = x_n - x_i$, $r = (x_n - x_{n-j})/v$. The Jacobian for this transformation is v. To obtain the density for r alone we integrate x and v over their ranges $(-\infty < x < \infty, 0 \le v < \infty)$ to yield

$$P(r) = \frac{n!}{(i-1)!(n-j-i-1)!(j-1)!} \int_{-\infty}^{\infty}\int_{0}^{\infty} \left[\int_{-\infty}^{x-v} \phi(t)\,dt\right]^{i-1} \left[\int_{x-v}^{x-rv} \phi(t)\,dt\right]^{n-j-i-1} \times \left[\int_{x-rv}^{x} \phi(t)\,dt\right]^{j-1} \phi(x-v)\phi(x-rv)\phi(x)v\,dv\,dx. \quad (2)$$

Equation 2 in Dixon (1951) is Equation 2 for $j = i = 1$. The last equation in his paper should be exactly Equation 2, but it is missing the Jacobian factor v. Similarly, his equation 14 should be my Equation 2 for $j = 1, i = 2$, but it has several errors; it should read

$$P(r_{11}) = \frac{n!}{(n-4)!} \int_{-\infty}^{\infty} \int_{0}^{\infty} \left[\int_{-\infty}^{x-v} \phi(t)\,dt\right] \left[\int_{x-v}^{x-r_{11}v} \phi(t)\,dt\right]^{n-4} \times \phi(x-v)\phi(x-r_{11}v)\phi(x)v\,dv\,dx. \quad (3)$$

Despite these printing errors, all the tables in Dixon's paper were computed from correct formulas.

Rewriting Equation 2 in terms of the cumulative standard normal distribution $\Phi(x)$ gives

$$P(r) = \frac{n!}{(i-1)!(n-j-i-1)!(j-1)!} \int_{-\infty}^{\infty} \int_{0}^{\infty} [\Phi(x-v)]^{i-1} \times [\Phi(x-rv) - \Phi(x-v)]^{n-j-i-1} [\Phi(x) - \Phi(x-rv)]^{j-1} \phi(x-v)\phi(x-rv)\phi(x)v\,dv\,dx. \quad (4)$$

The main numerical problem solved in this paper is the evaluation of Equation 4 by numerical quadrature.

2.2. Cumulative distribution function and critical values for R

The cumulative distribution function $G(R)$ is

$$G(R) = \int_{0}^{R} P(r)\,dr. \quad (5)$$

The range of r is $0 \le r \le 1$, so $G(0) = 0$ and $G(1) = 1$. The critical values are the roots of $(1-\alpha) - G(R) = 0$ where α is a specified probability; $G(R)$ increases monotonically from 0 to 1 so there is exactly one critical value for each value of α.

3. Numerical approach

3.1. Probability density

The integrand in Equation 4 can change sharply because of the Gaussian form of the ϕ terms, and for large n also because of the high power in the second Φ term. I chose to regard the Gaussian dependences on x and v as weight functions and use Gaussian quadratures to accomodate them; the integral over x can be done with a classical Gauss-Hermite quadrature, but the integral over v requires a special quadrature because of the $[0, \infty]$ limits.

Writing out the three ϕ terms in Equation 4 and gathering terms in x^2 and v^2 yields

$$\begin{aligned} P(r) = {} & \frac{n!}{(i-1)!(n-j-i-1)!(j-1)!} (2\pi)^{-3/2} \int_{-\infty}^{\infty} e^{-3x^2/2} \int_{0}^{\infty} e^{-(1+r^2)v^2/2} \\ & \times [\Phi(x-v)]^{i-1} [\Phi(x-rv) - \Phi(x-v)]^{n-j-i-1} [\Phi(x) - \Phi(x-rv)]^{j-1} \\ & \times e^{xv(1+r)} v\,dv\,dx \quad (6) \\ = {} & N \int_{-\infty}^{\infty} e^{-3x^2/2} \int_{0}^{\infty} e^{-(1+r^2)v^2/2} J(x,v,r) e^{xv(1+r)} v\,dv\,dx, \quad (7) \end{aligned}$$

where N is the normalization factor (including the $(2\pi)^{-3/2}$ term) and $J(x, r, v)$ represents the terms containing Φ.

The change of variable $t^2 = (1+r^2)v^2/2$, $u^2 = 3x^2/2$ converts the integral into a form suitable for Gauss-Hermite quadratures:

$$P(r) \;=\; N\sqrt{\frac{2}{3}}\sqrt{\frac{2}{1+r^2}}\int_{-\infty}^{\infty} e^{-u^2}\int_{0}^{\infty} e^{-t^2} J(x(u), v(t,r), r)\exp\left(\frac{2ut(1+r)}{\sqrt{3(1+r^2)}}\right) dt\, du, \quad (8)$$

where $x(u) = u\sqrt{2/3}$ and $v(t,r) = t\sqrt{2/(1+r^2)}$.

Now the quadrature rules

$$\int_{0}^{\infty} e^{-t^2} f(t)\, dt \approx \sum_{l=1}^{n_{\mathrm{hh}}} w_l f(t_l), \quad (9)$$

$$\int_{-\infty}^{\infty} e^{-u^2} g(u)\, du \approx \sum_{k=1}^{n_{\mathrm{fh}}} w_k g(u_k), \quad (10)$$

can be introduced. Here the w_l are the weights and the t_l are the abscissas for an n_{hh}-point *half-range Hermite* quadrature. The abscissas and weights can be obtained with the **ORTHPOL** program package described in Gautschi (1994); the `test7` program in that package does the calculation that is needed. In the programs accompanying this paper, the weights and abscissas for $n_{\mathrm{hh}} = 17$ and $n_{\mathrm{hh}} = 15$ are included in tabular form. Similarly the w_k and u_k are the weights and abscissas for an ordinary n_{fh}-point Gauss-Hermite quadrature. Application of the quadrature rules yields

$$P(r) \approx N\sqrt{\frac{2}{3}}\sqrt{\frac{2}{1+r^2}}\sum_{k=1}^{n_{\mathrm{fh}}}\sum_{l=1}^{n_{\mathrm{hh}}} w_k w_l J(x(u_k), v(t_l, r), r)\exp\left(\frac{2u_k t_l(1+r)}{\sqrt{3(1+r^2)}}\right) \quad (11)$$

The function `rdens(r)` implements Equation 11.

3.2. Cumulative distribution function

$P(r)$ is a well-behaved function, and its integral over the domain $[0, R]$ presents no numerical difficulties. For purposes of computing critical values it is important that the numerical implementation of $G(R)$ should be monotonic and smooth, so it is best to avoid numerical procedures that incorporate convergence tests. I have chosen to use Gauss-Legendre quadrature with a fixed number n_{gl} of quadrature points, so that

$$G(R) \approx \frac{R}{2}\sum_{m=1}^{n_{\mathrm{gl}}} w_m P(Ry_m/2), \quad (12)$$

where w_m and y_m are the usual Gauss-Legendre weights and abscissas on the range $[-1, 1]$, and the variable transformation $y = 2r/R - 1$ was used to change the $[0, R]$ range of integration to $[-1, 1]$. The function `rcdf(R)` implements Equation 12.

3.3. Critical values of R

The critical values are the values of R such that $1 - \alpha - G(R) = 0$, and they always lie between 0 and 1. The function `rcrit(alpha)` uses Brent's method (Brent 1973) to converge on the critical values.

4. Implementation

4.1. User callable routines

The program package includes five user-callable routines. All floating-point arguments and function results are `DOUBLE PRECISION`.

1. The subroutine `rinit(n,i,j,prec)` initializes the programs. Its arguments are integer n, i, and j as used above, and the integer variable `prec`, whose value must be 1 or 2. If `prec` is 2, the programs use 17, 31, and 16 quadrature points in the integrals over v, x, and r respectively; if `prec` is 1, it uses 2 fewer quadrature points in all three integrations. Comparison of results obtained with the two values of `prec` gives an estimate of the numerical error in the results. `prec=2` is adequate to converge all the values in Dixon's tables to an absolute error of 5×10^{-4} or less.

   The user must call `rinit` once before calling any of the other programs, and again if the value of `prec` needs to be changed. It sets up the quadrature abscissas and weights, stores all the r-independent terms needed for evaluation of Equation 11, and calls `rreset` to compute the normalization factor. All the information is stored in a `COMMON` block in the file `dixonr.fi`.

2. The subroutine `rreset(n,i,j)` stores n, i, and j, and computes and stores the value of N in Equation 11 for use by the other functions. It must be called any time the values of n, i, or j need to be changed.

3. The function `rdens(r)` returns the value of the probability density function at r for $r_{j,i-1}$ for n data points, where n, j, and i were specified in the most recent call to `rreset` or `rinit`. It implements Equation 11 with a single loop over a "composite index" that runs from 1 to $n_{\mathrm{hh}} n_{\mathrm{fh}}$, using values of t, x, ut, $w_k w_l$, and $\Phi(x)$ that were computed at all the quadrature points by `rinit`.

4. The function `rcdf(R)` returns the value of the cumulative distribution function at R, for the current n, i, j. It implements Equation 12 by calling `rdens` to obtain the values of $P(r)$ at the necessary quadrature points.

5. The function `rcrit(alpha)` returns the critical value of R for the specified α, for the current n, i, j. It solves $1 - \alpha - G(R) = 0$ using Brent's method (Brent 1973) as implemented in the routine `zeroin`. It uses a subsidiary function `rcerr(R)`, called by `zeroin`, to evaluate $1 - \alpha - G(R)$. In the package the error tolerance for `zeroin` is set to 10^{-6}, which is adequate for the numbers of quadrature points provided; users who generate larger numbers of quadrature points for special purposes may want to decrease the error tolerance.

4.2. Utility routines

The package contains several support routines in `utility.f`.

1. `fhquad(nfh,xfh,wfh)` returns Gauss-Hermite abscissas and weights for 29 or 31 quadrature points; `hhquad(nhh,xhh,whh)` returns half-range Hermite absicssas and weights for

15 or 17 quadrature points; and `glquad(ngl,xgl,wgl)` returns Gauss-Legendre abscissas and weights for 14 or 16 quadrature points.

Many numerical libraries contain routines that can generate Gauss-Hermite or Gauss-Legendre abscissas and weights for arbitrary numbers of quadrature points, and `fhquad` and `glquad` can easily be replaced with such routines. The only source of the half-range Hermite abscissas and weights I know is the **ORTHPOL** package. More points may become necessary if i increases beyond 3 or n increases beyond 30.

A user who wishes to use different numbers of quadrature points must replace `fhquad`, `hhquad`, and/or `glquad` with other versions, and change the values of `maxfh`, `maxhh`, and `maxgl` in `dixonr.fi`. In addition he may want to change the error tolerance used in `rcrit`.

`Phi(x)` is a double precision routine to return the cumulative standard normal distribution at x. I have included the "little C function" of Marsaglia (2004), translated to FORTRAN.

`zeroin` is a function from Netlib (Browne, Dongarra, Grosse, and Rowan 1995, `http://www.netlib.org/`) that uses Brent's method to find the roots of a function of one variable. It calls `d1mach` to obtain a value of the machine precision, and I include the current `d1mach` routine from Netlib that adapts automatically to the host machine.

4.3. Example programs

Several example programs, with corresponding output files, are included. To create an executable for `test1`, for example, the user must compile and link `test1.f`, `rfuncs.f` and `utility.f`; the file `dixonr.fi` must be in the same directory with `rfuncs.f` at compile time.

1. `test1.f` generates a table of the density and cumulative distribution functions of the Dixon r_{11} statistic ($j = 1, i = 2$) for $n = 7$ by calls to `rdens` and `rcdf`.

```
!  generate table of density and cumulative distribution function
!  for Dixon's r_{11} statistic for n=7

      program test1
      implicit none
      double precision rdens, rcdf !functions
      integer npts, i, j, n, k, prec
      double precision r, rstep
      j = 1
      i = 2
      n = 7
      prec = 2  !  use higher available precision
      npts = 50
      rstep = 1.0d0/(npts-1)
      call rinit(n, i, j, prec)  !initialize
      write(*,'(3(a,I2,2x))') 'i = ', i,'j = ', j, 'n = ', n
      write(*,*) '   r      P(r)    G(R) '
```

```
      do k = 1, npts
         r = (k-1)*rstep
         write(*,'(3(2x,F6.3))') r, rdens(r), rcdf(r)
      end do
      end
```

2. `test2.f` generates a table of critical values for r_{11} at $\alpha = 0.1$, for $4 \leq n \leq 15$ by calls to `rcrit`. This table can be compared to the fifth column of Dixon's Table II.

```
!  generate table of critical values at alpha=0.1
!  for Dixon's r_{11} statistic for n=4 to 15

      program test2
      implicit none
      double precision rcrit !function
      integer npts, i, j, n, prec
      double precision alpha
      j = 1
      i = 2
      n = 4
      prec = 2
      alpha = 0.1d0
      call rinit(n, i, j, prec)  ! initialize
      write(*,'(2(a,I2,2x),a,F7.4)') 'j = ', j, 'i = ', i,
     1     'alpha = ', alpha
      write(*,'(a)') ' n   Rcrit'
      do n = 4, 15
         call rreset(n,i,j)  ! choose new n
         write(*,'(I2,2x,F6.3)') n, rcrit(alpha)
      end do
      end
```

3. `test3.f` generates a similar table of critical values, but it does the calculation for two different values of `prec`, and prints the absolute and relative differences between the results obtained at low and high precisions. Part of its output is

```
j =  1  i =  2  alpha =  0.1000
 n   low     high       diff     diff/high
 4   0.910   0.910   -0.10E-07   -0.11E-07
 6   0.610   0.610   -0.42E-09   -0.70E-09
 8   0.480   0.480   -0.34E-11   -0.71E-11
```

[...]

```
22   0.269   0.269   -0.22E-05   -0.81E-05
24   0.259   0.259   -0.15E-05   -0.58E-05
26   0.251   0.251    0.96E-06    0.38E-05
28   0.243   0.243    0.56E-05    0.23E-04
30   0.237   0.237    0.12E-04    0.53E-04
```

4. `test4.f` regenerates all the tables from Dixon's paper in a single run. It takes a few minutes to run on a circa-2001 computer.

   Dixon asserted that Tables I–III in his paper (for r_{10}, r_{11}, and r_{12}) were accurate to within 0.002, and Tables IV–VI were accurate to within 0.004. His table for r_{10} nearly meets that standard; two entries ($n = 4$ and 6 for $\alpha = 0.005$) are in error by 0.005 and 0.003 respectively. The other tables, however, contain more entries that disagree with the `test4` output, in addition to the typographical errors identified in Rorabacher (1991). Most of the discrepant entries are for $6 \leq n \leq 10$ and for $\alpha \leq 0.10$, except that in Dixon's table for r_{22} most of the entries for $n = 6$ and 7 are incorrect by 0.01–0.02.

5. Usage notes

The language used is fixed-form FORTRAN 90, but the few FORTRAN 90 features used are supported by most modern FORTRAN 77 compilers as well. The programs have been tested with Compaq Visual FORTRAN version 6.6 and with GNU FORTRAN 3.3.1.

The programs are reasonably efficient implementations of the numerical procedures described above, but they do the quadrature anew each time a new value of $P(r)$ is needed. For single calls or for generating tables of values this design is adequate. However, if many values of $P(r)$, $G(R)$, or the critical values are needed, the programs in this package should not be used directly. Instead, it will be faster to use these programs to generate a table of accurate values with moderate density over the range of r or R needed, then use interpolation in that table to generate individual values.

The notation used in this paper is consistent with that of Dixon, and the α values and corresponding critical values of R are those appropriate for one-tailed tests. In most uses of these tables for outlier rejection, a two-tailed test is needed, so that "90% confidence" corresponds to the $\alpha = 0.05$ columns in the output of `test4`. See Rorabacher (1991) for a discussion.

References

Barnett V, Lewis T (1984). *Outliers in Statistical Data*. Wiley, New York, 2nd edition.

Beckman RJ, Cook RD (1983). "Outlier.....s." *Technometrics*, **25**(25), 119–149.

Brent RP (1973). *Algorithms for Minimization Without Derivatives*. Prentice-Hall, Englewood Cliffs, N.J.

Browne S, Dongarra J, Grosse E, Rowan T (1995). "The Netlib Mathematical Software Repository." URL http://www.netlib.org/srwn/srwn21.html.

Dixon WJ (1950). "Analysis of Extreme Values." *The Annals of Mathematical Statistics*, **21**(4), 488–506.

Dixon WJ (1951). "Ratios Involving Extreme Values." *The Annals of Mathematical Statistics*, **22**(1), 68–78.

Efstathiou CE (1992). "Stochastic Calculation of Critical Q-Test Values for the Detection of Outliers in Measurements." *Journal of Chemical Education*, **69**(9), 733–736.

Gautschi W (1994). "Algorithm 726: **ORTHPOL** – A Package of Routines for Generating Orthogonal Polynomials and Gauss-type Quadrature Rules." *ACM Transactions on Mathematical Software*, **20**(1), 21–62.

Hampel FR (1985). "The Breakdown Points of the Mean Combined with Some Rejection Rules." *Technometrics*, **27**(2), 95–107.

Komsta L (2005). **outliers***: Tests for Outliers*. R package version 0.12, URL http://CRAN.R-project.org/.

Marsaglia G (2004). "Evaluating the Normal Distribution." *Journal of Statistical Software*, **11**(4), 1–7. URL http://www.jstatsoft.org/v11/i04/.

R Development Core Team (2005). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Rorabacher DB (1991). "Statistical Treatment for Rejection of Deviant Values: Critical Values of Dixon's "Q" Parameter and Related Subrange Ratios at the 95% Confidence Level." *Analytical Chemistry*, **63**(2), 139–146.

Affiliation:

George C. McBane
Department of Chemistry
Grand Valley State University
Allendale, MI 49401, United States of America
E-mail: mcbaneg@gvsu.edu
URL: http://faculty.gvsu.edu/mcbaneg/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 16, Issue 3 *Submitted:* 2006-01-09
May 2006 *Accepted:* 2006-05-04