UNITED STATES
DEPARTMENT OF THE INTERIOR
GEOLOGICAL SURVEY

COMPUTER ROUTINES FOR PROBABILITY DISTRIBUTIONS, RANDOM NUMBERS, AND RELATED FUNCTIONS

By

W. Kirby

Open-File Report 80-448

Reston, Virginia

1980

CONTENTS

COMPUTER ROUTINES FOR PROBABILITY DISTRIBUTIONS, RANDOM NUMBERS, AND RELATED FUNCTIONS

By

W. Kirby

ABSTRACT

Use of previously coded and tested subroutines simplifies and speeds up program development and testing. This report presents routines that can be used to calculate various probability distributions and other functions of importance in statistical hydrology.

INTRODUCTION

This report describes computer subprograms that compute several probability distributions of importance in statistics and hydrology. Several other routines performing closely related functions such as random number generation, sorting, and plotting also are described. Various versions of these routines have been compiled from various sources and have been used in a variety of programs over a period of several years. Use of these previously coded and tested routines helps to simplify and speed up program development and testing. For this reason the routines have been collected here in a common format to improve their usability and accessability to others.

The routines are written in Fortran. Versions are operational on the U.S. Geological Survey IBM/370[1/] computers (Fortran IV) and on the Harris S-125 computer (Fortran 77) in Reston, Va. There are only trivial differences between the IBM and Harris versions, so it is expected that these routines could be transported easily to other machines as well. A listing of the source code for the Harris computer is included in this report. Instructions for obtaining machine-readable copies of these routines are given in the next two sections of this report, and instructions for calling the routines from your own Fortran program are given in the section after that.

General information about the definition and applications of various probability distributions is given by Benjamin and Cornell (1970) and by Mood, Graybill, and Boes (1974). Computational formulas and approximations are summarized by Zelen and Severo (1964). The mathematical and computational details of all commonly used distributions are exhaustively treated by Johnson and Kotz (1970).

It is believed that the routines presented in this report perform with reasonable efficiency substantially as described. No guarantees can be given, however, regarding accuracy, reliability, or speed. Users performing critical computations, particularly at extreme values of the function parameters and arguments, are encouraged to obtain a copy of the latest version of the routine and perform their own tests. The author would be grateful to receive reports of any such tests as well as reports of errors or deficiencies in any of the routines.

[1/] The use of brand names in this report is for identification only and does not imply endorsement by the U.S. Geological Survey.

SOURCE CODE AVAILABILITY

Up-to-date information on the operational status and Fortran coding of these subprograms may be obtained by running the following job on either of the USGS computers Re1 or Re2:

```
/*RELAY  PUNCH REn        (n=1 or 2)
//•••        JOB (•••)    (job card)
/*PROCLIB    VG48DST.PROCLIB
//           EXEC  SUBLIST,VERSION=xxx
//
$$$
```

The VERSION parameter should be set equal to either HARRIS or IBM. This job produces a short print-out of general information on how to obtain copies of the source code and how to link machine-language (already-compiled) copies of these routines to your own Fortran program. It also produces a short print-out showing the beginning and ending line numbers of each subprogram in the indicated version of the package; this information can be used to get copies of selected routines from the package. The line numbers are subject to occasional change, so it is recommended that this job be rerun as necessary to ensure that up-to-date information is used.

A complete listing of the Fortran source code (about 3000 lines) may be obtained by adding the parameter TYPE=LL to the EXEC card, as follows:

```
// EXEC SUBLIST,VERSION=xxx,TYPE=LL
```

Selected lines may be printed by using TYPE=DE and supplying one or more control cards to delete the unwanted lines. The format of the control card is:

```
DE m,n
```

where DE is punched in columns 1 and 2 and where m and n are the beginning and ending line numbers of the group of lines to be omitted; they are punched starting in any convenient column and are separated by a comma with no intervening blanks. The pertinent line numbers are those appearing in columns 73-80 of the FORTRAN card images; any decimal points should be omitted from the DE control card. For example, to print the Harris version of PRPLOT, the following cards would be used

```
// EXEC SUBLIST,VERSION=HARRIS,TYPE=DE
DE 1,1628
DE 1838,2826
```

The entire Fortran source code (about 3000 cards) or selected portions of it may be punched into cards by adding the parameter PUNCH=SYSOUT to the EXEC card. The appropriate TYPE value, LL or DE, must be used as well. The punched output can be directed to magnetic tape at a terminal with a tape drive by adding the parameter SYSOUT='(K,,idno)' to the EXEC card. In this parameter, K designates output to a terminal tape drive and idno is an arbitrary 4-character identifier for the tape to be used at the terminal. The cards can be copied to a data set by supplying an appropriate DD statement named EXECLIST; in this case the PUNCH and SYSOUT parameters may be omitted. The printed listing of the punched cards may be suppressed, if desired, by the parameter PRINT=NOSOURCE on the EXEC card.

LINKAGE INFORMATION

These routines have been compiled and stored in subroutine libraries on the Harris computer in Reston and on the Rel-Re2 IBM system. The user has only to call the routines, as explained in the next section, and to make the appropriate subroutine library available at linkage-editing time, and the necessary routines automatically will be included in the program. It is not necessary to retrieve and recompile the source code each time it is to be used.

On the Harris computer the routines were compiled with the Fortran-77 compiler SAUF77 using the implicit-double-precision option (P). They are stored in a link-ready library named *XLIB77. They may be made available to your Fortran program by the following commands:

```
SAUF77   your*fortprog    $ Fortran-77 compile
VU       your*xeprog      $ Vulcanize
LI      *XLIB77  *SAUL77  *LIBERY   $ Subroutine libraries
BE                        $ Begin Vulcanizing
```

On the IBM Rel-Re2 system, the routines were compiled with the Fortran H-Extended compiler. They are stored in cataloged load-module library named VG48DST.PUBLIB. They may be made available to your Fortran program by means of the ULIB parameter provided with the Fortran procedures described in chapter 7 of the USGS Computer Users' Manual (unpublished documentation available from the USGS Computer Center Division, User Services). Typical compile-load-and-go usage is as follows:

```
// EXEC FORTXCG,ULIB='VG48DST.PUBLIB'
     your Fortran program
//GO.SYSIN DD *
     your data
```

Any other parameters needed on the EXEC card may be placed before or after the ULIB parameter. These routines were designed for use with the currently supported FORTX... and FTG1... procedures described in the USGS Computer Users' Manual; they may not operate correctly under the obsolescent FORTG... and FORTH... procedures.

CALLING INSTRUCTIONS

The following paragraphs contain instructions for calling the routines from Fortran programs. Each paragraph contains the name of the routine, a typical argument list, and a brief description of the function performed. Familiarity with the general mathematical character of the functions is assumed, as is familiarity with the Fortran language. In particular, variable names beginning with the letters I-N refer to integer data and names beginning with A-H or O-Z refer to real-valued data. Detailed discussion of computational methods, limitations on arguments, handling of error conditions, and performance generally is not attempted. The references and source listings may be consulted for this information if necessary.

BESFIO(X) - Function returns the value of the modified Bessel function I_0 ("I-zero") of order zero at argument x. Source - IBM (1970) SSP routine IO.

BETAP(X,A,B) - Function returns the value of the incomplete beta function

$$I_X(a,b) = \frac{1}{B(a,b)} \int_0^X u^{a-1}(1-u)^{b-1}\, du$$

This value also is the probability that the beta-distributed random variable with parameters a and b will be less than or equal to X. Restriction -- The parameters a and b must be strictly positive; otherwise, BETAP will be set to a large negative number. Source - Stanford University Computation Center/ (unpublished library program C060, 1969).

CHISQP(X,N) - Function returns the probability that the chi-square random variable with n degrees of freedom will be less than or equal to x, $P\{\chi_n^2 \leq x\}$. Reference - Hill and Pike (1967).

CHISQX(P,N) - Function returns the p-th quantile of the chi-square random variable with n degrees of freedom. That is, it returns the solution for x of the equation $P\{\chi_n^2 \leq x\} = p$ for the given value of p. Reference - Goldstein (1973).

DGAMMA(X) - Double precision gamma function of double precision argument X. DGAMMA and X must be declared double precision in the calling program. See GAMMA.

DLGAMA(X) - Double precision function returns the natural logarithm of the gamma function $\Gamma(X) = \int_0^\infty t^{x-1}e^{-t}\,dt$. Restrictions - X must be greater than 10^{-9} and less than 10^{35}; otherwise DLGAMA is set to 10^{38}. X and DLGAMA must be declared double precision in the calling program. Source - IBM (1970) SSP routine DLGAM.

ERF(X) - Function returns the value of the error function $(2/\sqrt{\pi})\int_0^x e^{-u^2}\,du$. Source - IBM (1970) SSP routine NDTR.

EXPIF(X) - Function returns the value of the exponential integral $\int_x^\infty (e^{-t}/t)dt$. variously denoted as Ei(x), -Ei(-x), and $E_1(x)$. It is also known as the well function, W(u), for the Theis-Jacob-Wenzel non-equilibrium artesian well. The value of x may be positive or negative; EXPIF is set to a large positive number when x=0. Source - IBM (1970) SSP routine EXPI.

GAMMA(X) - Function returns the value of the gamma function, defined for $x>0$ by

$$\Gamma(X) = \int_0^\infty t^{x-1}e^{-t}\,dt$$

and extended to other x by the recursion formula $\Gamma(x+1) = x\,\Gamma(x)$. Restriction - x must be less than 34. and not within 10^{-6} of a negative integer or 0.0; otherwise GAMMA is set to 10^{38}. The IBM FORTRAN IV library version of GAMMA requires positive values of x. Source - IBM (1970) SSP routine GMMMA.

GAMMAP(A,X) - Function returns the value of the incomplete gamma function ratio

$$\gamma(a,x) = \int_0^x t^{a-1} e^{-t}\, dt/\Gamma(a), \qquad (a>0)$$

which also is the probability that the gamma-distributed random variable with shape factor a is less than or equal to x. Note - The gamma is a member of the Pearson Type III family of distributions. It has mean a, standard deviation $\sqrt{a}$, skewness $2/\sqrt{a}$, and lower bound 0. See also HARTIV and WILFRT. Restriction - The shape factor a must be greater than 0; otherwise the distribution is concentrated at x=0. Source - IBM (1970) SSP routine CDTR.

GAUSAB(P) - Function returns the p-th quantile (or Gaussian abscissa) of the Gaussian or standardized normal distribution. This value also is known as the standardized normal deviate with non-exceedance probability p. It is the solution for x of the equation

$$P\{Z \le x\} = (1/\sqrt{2\pi})\int_{-\infty}^{x} e^{-z^2/2}\, dz = p$$

for the given value of p, where Z is the standardized normal random variate. Restriction - If p is less than or equal to 0, GAUSAB is set to -10; if p is greater than or equal to 1, GAUSAB is set to +10. Reference - Zelen and Severo (1964).

GAUSCF(X) - Function returns the value of the Gaussian or standard normal cumulative probability function at x, as follows:

$$P\{Z \le x\} = (1/\sqrt{2\pi}) \int_{-\infty}^{x} e^{-z^2/2} dz$$

where Z is the standardized normal random variate. Reference - Zelen and Severo (1964).

GAUSDY(X) - Function returns the value of the Gaussian or standard normal density, $(1/\sqrt{2\pi})\ e^{-x^2/2}$, at x.

GAUSSF(PEX) - Function returns the Gaussian or standard normal deviate with exceedance probability (tail probability) PEX. It is the solution for x of the equation

$$P\{Z>x\} = (1/\sqrt{2\pi}) \int_x^\infty e^{-z^2/2}\,dz = P_{ex}$$

for the given value of P_{ex}, where Z is the standarized normal random variate. Restriction - If P_{ex} is greater than or equal to 1.0, GAUSEX is set to -10. If P_{ex} is less than or equal to 0.0, GAUSEX is set to +10. Reference - Zelen and Severo (1964).

GAUSSF(IRAN) - Function returns a quasi-random number drawn from the standard normal (Gaussian) distribution. IRAN is the "seed" of the random number generator; it must be initialized by the user but thereafter is updated automatically by the generator. The initial IRAN may be any value between 1 and 2147483646. On the Harris computer IRAN must be declared INTEGER*6. References - Box and Muller (1958), Lewis and others (1969).

GAUSSV(IRAN, X, N) - Subroutine places a quasi-random sample of size N in vector X. The sample is drawn from the standard normal (Gaussian) distribution. IRAN (which must be declared INTEGER*6 on the Harris computer) is the "seed" of the random number generator; it must be initialized by the user but thereafter is updated automatically by the subroutine. The initial IRAN may be anywhere between 1 and 2147483646. References - Box and Muller (1958), Lewis and others (1969).

GUMBEL(P) - Function returns the value of the standardized deviate with non-exceedance probability P from the Gumbel Type I (double-exponential) extreme-value distribution. For a general Gumbel variate with mean m and standard deviation s, the p-th quantile is

$$X_p = m + s*GUMBEL(p).$$

Reference - Benjamin and Cornell (1970).

GUMBEP(X) - Function returns the value of the cumulative probability distribution function of the standardized Gumbel Type I (double-exponential) extreme value random variate. If Y is a general Gumbel variate with mean m and standard deviation s, then

$$P\{Y \leq x\} = GUMBEP((x-m)/s).$$

Reference - Benjamin and Cornell (1970).

HARTIV(SKEW,RIPS) - Subroutine looks up percentage points of the standardized Pearson Type III distribution with skew coefficient SKEW in Harter's (1969, 1971) tables. Interpolation is done on the skew coefficient at all 31 tabular probabilities considered in Harter's two papers. The resultant interpolated percentage points are placed in the 31-word vector RIPS in increasing order. Restriction - The skew must not exceed 9.0 in absolute value; otherwise RIPS is filled with zeros. References - Harter (1969, 1971), U.S. Water Resources Council (1977).

HARTK(P,RIPS) - Function looks up the p-th quantile (standardized deviate with non-exceedance probability p) of the standardized Pearson Type III distribution tabulated in the vector RIPS. The vector RIPS must have been filled by a prior call to HARTIV. Restriction - The value of p must be between 0.0001 and 0.9999; otherwise HARTK will be set to a large number (positive if $p>0.9999$, negative if $p<0.0001$). Reference - see HARTIV.

HARTKK(Z,RIPS) - Function looks up that quantile of the standardized Pearson Type III distribution tabulated in the vector RIPS that has the same non-exceedance probability as the given standard normal deviate z. That is, HARTKK solves for x in the equation

$$P\{R \le x\} = P\{\underline{Z} \le z\}$$

where $\underline{Z}$ is the standard normal random variable and R is the standard Pearson Type III random variable tabulated in RIPS. The vector RIPS must have been filled by a call to HARTIV. Restriction - The value of z must not exceed 3.719 in absolute value; otherwise HARTKK will be set to a large value of the same sign as z. Reference - see HARTIV.

HARTP(Z,RIPS) - Function looks up the cumulative (non-exceedance) proability of the given value Z in the standardized Pearson Type III distribution tabulated in the vector RIPS. The vector RIPS must have been filled by a prior call to HARTIV. If z is outside the range of the table, HARTP is set to 0 (if $z<0$) or 1 (if $z>0$). Reference - see HARTIV.

HARTTP(IORDER, PROBV) - Subroutine copies the 31 standard tabular probabilities from Harter's (1969, 1971) tables into the user's 31-word Vector PROBV. If IORDER is negative, the probabilities are stored in decreasing order as exceedance probabilities; otherwise in increasing order. Reference - see HARTIV.

OUTKGB(SIG,N) - Function returns the value of the Grubbs-Beck (1972) one-sided single-outlier criterion for normal samples of size N at signifiance level SIG. That is, OUTKGB is the solution for x of the equation

$$P\{(X_{max} - \bar{x})/s > x\} = SIG$$

where X_{max} is the maximum of a sample of N normal random variates, $\bar{x}$ and s are the sample mean and standard deviation ($S=\{\Sigma(x-\bar{x})^2/(N-1)\}^{\frac{1}{2}}$), and SIG is the given significance level, expressed as either a percentage or a decimal fraction. This is the high-outlier criterion; the low-outlier criterion is just the negative of this: that is, -OUTKGB. Restrictions - The sample size N must be at least 3. The available significance levels are 1., 2.5, 5., and 10. percent (or .01, .025, .05, and .10). Otherwise OUTKGB will be set to a large negative number. Note - Piecewise linear approximation is used for N above 100 and OUTKGB is constant for N above 180. Reference - Grubbs and Beck (1972).

PRPLOT - Subroutine produces "printer-plotted" graphs on the line printer. Graphs are produced in three phases by calling three entry points, as follows:

PLOT2(AREA, XMAX, XMIN, YMAX, YMIN) - blanks out the array AREA and then fills it with a rectangular coordinate grid. The standard grid has a vertical (Y) axis 51 print lines high and a horizontal (X) axis 101 print columns wide. Five horizontal grid lines and 10 vertical ones are drawn at 10-space intervals in each direction. The axis annotations at the grid lines are printed with three decimal places. Nonstandard grids can be defined by PLOT1: see the reference for details. Subroutine PRPSCL, described below, can be used to make the axis annotations come out as "even" numbers. The array AREA must be of type CHARACTER*1 (Harris) or LOGICAL*1 (IBM) and must be dimensioned large enough to hold the entire plot area (5151 characters for the standard grid). (The IBM version will function with AREA arrays of any type, as long as they provide enough storage space. Such usage is discouraged, however.)

PLOT3(SYMBOL, X, Y, N) - Plots the character SYMBOL at the N locations defined by the vectors X and Y. The SYMBOLS are plotted in the array AREA initialized by PLOT2. SYMBOL may be either a literal constant or a CHARACTER*1(Harris) or LOGICAL*1(IBM) variable or array. If SYMBOL contains more than one character, only the leftmost one is plotted. X-Y coordinates outside the limits of the grid are ignored. Several curves can be drawn on one graph by repeated calls to PLOT3. When several points occupy the same grid position, the first-plotted points are obliterated; only the last-plotted point shows.

PLOT4(N, LABEL) - Prints the plot array AREA and scale annotations for the X and Y axes. The N-character string LABEL is printed as a label for the vertical (Y) axis. The standard grid uses 53 print lines, including the X-axis annotation. The user is reponsible for ejecting the page and printing page headings, captions, and the X-axis label. PLOT4 does not destroy the plot area, so additional curves may be plotted on the same graph, if desired, by additional calls to PLOT3 and then printed by PLOT4. Calling PLOT2 reinitializes the plotting grid.

Probability plots (normal probability grid) may be made by calling PLOT2P and PLOT4P instead of PLOT2 and PLOT4. The horizontal (X) axis is used for exceedance probabilities running from 99.9 percent at the left to 0.1 percent at the right. Probabilities outside this range are ignored. The user must convert the exceedance probabilities to the corresponding standard normal deviates; X = GAUSAB(PEX) may be used for this purpose. Then PLOT3 is used in the usual way to plot the transformed exceedance probabilities/deviates on the X scale and the data observations on the Y scale. The necessary subroutine calls are as follows:

PLOT2(AREA, YMAX, YMIN) - Sets up the standard normal probability grid. The plot array AREA must be dimensioned to hold 51 x 121 = 6171 characters and must be of type CHARACTER*1 (Harris) or LOGICAL*1 (IBM). (The IBM version will function with AREA arrays of any type as long as they provide enough storage space. Such usage is discouraged, however.)

PLOT4P(N, LABEL) - Prints the plot area and scale annotations. The N-character string LABEL is printed on the vertical axis and an exceedance probability label is printed under the X axis. 54 lines are printed. The user is responsible for page ejects, titles, etc.

The scale annotations may be forced to come out as "even" values by the following subroutine call:

PRPSCL(U1, U2, NGRID, P1, P2) - Subroutine computes "pretty" endpoints P1, P2 for a scale with NGRID gid marks for plotting data between the "ugly" values U1 and U2.

Reference - unpublished documentation for USGS program number B524 (PRPLOT), available from USGS Computer Center Division, User Services. PLOT1 - PLOT4 are part of program B524 (PRPLOT); PLOT2P and PLOT4P make use of PRPLOT but are not part of program B524; PRPSCL is an independent subroutine.

RANDUB(IRAN) - Function returns a quasi-random number drawn from the uniform distribution on the unit interval. IRAN (which must be declared INTEGER*6 on the Harris computer) is the "seed" of the random number generator; it must be initialized by the user but thereafter is updated automatically by the generator. The initial value of IRAN may be between 1 and 2147483646. Note - This generator is the Lewis-Goodman-Miller (1969) generator designed for the IBM/360 computer.

RANDUV(IRAN,X,N) - Subroutine places a quasi-random sample of size N from the unit uniform (0,1) distribution into the vector X. IRAN (which must be declared INTEGER*6 on the Harris computer) is the "seed" of the random number generator; it must be initialized by the user but thereafter is updated automatically by the generator. The initial value of IRAN may be between 1 and 2147483646. Note - this generator is the Lewis-Goodman-Miller (1969) generator designed for the IBM/360 computer.

SMIRP(X) - Function returns the large-sample limiting probability distribution of the Kolmogorov and Smirnov statistics, as follows:

$$P\ \{\sqrt{n}\ D_n \leq x\} \qquad \text{(Kolmogorov 1 - sample)}$$

$$P\ \{\sqrt{(n+m)/nm}\ \ D_{n,m} \leq x\} \qquad \text{(Smirnov 2 - sample)}$$

where

$$D_n = \max_x |F_n(x) \quad F(x)|$$

$$D_{n,m} = \max_x |F_n(x) - F_m(x)|$$

and F_n, F_m are empirical (sample) distribution functions of sizes n and m and F is the corresponding hypothetical distribution. Source - IBM (1970) Subroutine SMIRN.

SNCTPA(X,N,D) - Function returns an approximate value of the Student noncentral t probability $P\{t_{N,D} \leq X\}$ with N degrees of freedom and noncentrality parameter D. When D=0 the noncentral t becomes the ordinary Student t. The approximation is intended for N greater than say, 20, but retains some usefulness down to N equal 5 or 10. Reference - Zelen and Severo (1964, eqn. 26.7.10).

SNCTXA(P,N,D) - Function returns an approximate value of the Student noncentral t quantile with N degrees of freedom, noncentrality parameter D, and nonexceedance probability P. That is, it is the solution for x of the equation $P\{t_{N,D} \leq x\} = P$. The approximation is intended for large N but retains some value for N as small as 5 or 10. Reference - Zelen and Severo (1964, eqn. 26.7.10).

SNEFPB(X,N1,N2) - Function returns the probability that the Snedecor F (or variance-ratio) random variable is less than or equal to the given value of x. That is,

$$SNEFPB = P\{F_{N1,N2} \leq x\}$$

where $F_{N1,N2} = S_1^2 / S_2^2$ is the F or variance-ratio statistic with N_1 degrees of freedom in the numerator and N_2 in the denominator. SNEFPB is an entry point of subprogram BETAP (which see).

SORT(X,N) - Subroutine rearranges the N real numbers in the vector X into increasing order. If the value of N is negative (SORT(X,-N)), the numbers are sorted in decreasing order. This routine uses the N-log-N Quicksort algorithm. It is about 3 times as fast as a simple bubble sort for N=25 and about 15 times as fast for N=250. Source--Stanford University Computation Center (unpublished library program C063, 1971).

STAT3(X,N,A,S,G) - Subroutine computes the sample mean, standard deviation, and skew coefficient (A,S, and G) of the N real numbers in the vector X. The sample statistics are defined as follows:

$$A = \Sigma X/N \qquad S = \{\Sigma(X-A)^2/(N-1)\}^{\frac{1}{2}}$$

$$G = \{N\Sigma(X-A)^3\}/\{(N-1)(N-2)S^3\}$$

The sample size N must be at least 3 to define the skew, 2 to define the standard deviation, and 1 to define the mean. If the sample is too small to define any of these statistics (or if N is negative) the undefined statistics are set equal to zero; no error messages or warnings are issued.

STUTP(X,N) - Function returns the probability that Student's t with n degrees of freedom is less than or equal to x, $P\{t_n \leq x.\}$ Note - Tail probabilities for two-sided t tests can be computed as follows:

$$P\{|t_n| > x\} = 2.*STUTP(-x,n)$$

for $x \geq 0$. Reference - Hill (1970a).

STUTPB(X,N) - Function returns the probability that Student's t with n degrees of freedom is less than or equal to x. STUTPB performs the same function as STUTP, but is provided automatically as part of the BETAP routine for the beta distribution (which see).

STUTX(P,N) - Function returns the p-th quantile of Student's t with n degrees of freedom. That is, it solves for x the equation $P\{t_n \leq x\} = p$ for the given value of p. Note - Two-sided t-criteria can be computed as follows: $P\{|t_n| > x\} = a$ has the solution x = STUTX(1.-a/2.,n). Reference - Hill (1970b).

WCFGSM(FLAT,FLON) - Function returns the value of the generalized skew coefficient shown on Plate 1 of the U.S. Water Resources Council's Guidelines for Determining Flood Flow Frequency (Hydrology Committee Bulletin 17, 1976, or 17-A, 1977) at latitude FLAT and longitude FLON. FLAT and FLON must be expressed in decimal degrees so that, for example, 45°30' would have to be expressed as 45.5°. For points outside the conterminous United States, Alaska, and Hawaii, the function returns a large negative number.

WEIBUL(SKU,P) - Function returns the p-th quantile of a standardized Weibull variate with skew SKU. Thus it returns the value $(W_p-EW)/s_w$, in which EW and s_W are the mean and standard deviation of the Weibull variate W and W_p is the solution for x of the equation

$$P\{W \leq x\} = 1 - \exp(-x^c) = p$$

for any probability p. In this equation, c is the Weibull shape factor, which depends on the skew; the program automatically calculates c, EW, and s_w each time the skew coefficient is changed. Restrictions - The skew coefficient must be less than 100 in absolute value; otherwise 100 is used. The skew decreases to zero as the shape factor c increases to about 3.6. Negatively skewed distributions are obtained by reflecting the positively skewed distribution, not by increasing c beyond 3.6. Reference - Benjamin and Cornell (1970).

WILFRT(SKU,ZETA) - Function returns an (approximately) Pearson Type III standardized variate with skew SKU and probability corresponding to that of the given standard normal deviate ZETA. That is, the value returned is an approximate solution for x of the equation

$$P\{W \leq x\} = P\{Z \leq ZETA\}$$

where Z and W are respectively the standard normal and Pearson Type III variates. Specifically, WILFRT(SKU,GAUSAB(P)) is an approximation to the p-th quantile (x such that $P\{W \leq x\} = p$) of the Pearson Type III distribution with skew SKU. Restriction - The skew must not exceed 9.75 in absolute value; otherwise $\pm$ 9.75 will be used. Note - the approximation is an improved Wilson-Hilferty (cube-root-normal) transformation which matches the mean, standard deviation, skew, and lower bound of the Pearson type III distribution. The approximation is excellent for skews below 2 in absolute value and is useful throughout the defined range of skews. Reference - Kirby (1972).

WILFRV(SKU, ZETAV, N) - Subroutine transforms the N-element vector ZETAV from standard normal deviates to (approximate) Pearson Type III standardized deviates with skew SKU. The vector ZETAV must be filled with standard normal deviates before calling WILFRV. GAUSAB or GAUSSV may be used for this purpose. WILFRV is a vectorial version of WILFRT (which see).

REFERENCES

Benjamin, J. R., and Cornell, C. A., 1970, Probability, statistics, and decision for civil engineers: New York, McGraw-Hill, 684 p.

Box, G. E. P., and Muller, M. E., 1958, A note on the generation of random normal deviates: Annals of Mathematical Statistics, v. 29, no. 2, p. 610-611.

Goldstein, R. B., 1973, Algorithm 451, Chi-square quantiles: Communications of the ACM, v.16, no.8, p.483-485.

Grubbs, F. E., and Beck, G. 1972, Extension of sample sizes and percentage points of outlying observations: Technometrics, v. 14, no. 4, p. 847-854.

Harter, H. L., 1969, A new table of percentage points of the Pearson Type III distribution: Technometrics, v. 11, no. 1, p. 177-187.

--- 1971, More percentage points of the Pearson Type III distribution: Technometrics, v. 13, no. 1, p. 203-204.

Hill, G. W., 1970a, Algorithm 395, Student's t-distribution: Communications of the ACM, v. 13, no. 10, p. 617-619.

--- 1970b, Algorithm 396, Student's t-quantiles: Communications of the ACM, v. 13, no. 10, p. 619-620.

Hill, I.D., and Pike, M.C., 1967, Algorithm 299, Chi-squared integral: Communications of the ACM, v. 10, no. 4, p. 243-244.

IBM, 1970, System/360 scientific subroutine package version III programmer's manual: IBM Order No. GH20-0205.

Johnson, N. L., and S. Kotz, 1970, Continuous univariate distributions, 2 vols: New York, John Wiley, 300 p., 306 p.

Kirby, W., 1972, Computer-oriented Wilson-Hilferty transformation that preserves the first three moments and the lower bound of the Pearson Type 3 distribution:. Water Resources Research, v. 8, no. 5, p. 1251-1254.

Lewis, P. A. W., Goodman, A. S., and Miller, J. M., 1969, A pseudorandom number generator for the System/360: IBM Systems Journal v. 8, no. 2, p. 136-146.

Mood, A. M., Graybill, F. A., and Boes, D. C., 1974, Introduction to the theory of statistics, 3d ed: New York, McGraw-Hill, 564 p.

U.S. Water Resources Council, Hydrology Committee, 1977, Guidelines for determining flood flow frequency, Bulletin 17A: Washington, D.C.

Zelen, N., and Severo, N.C., 1964, Probability functions, Chap. 26 in M. Abramowitz and I. A. Stegun, eds., Handbook of Mathematical Functions: U.S. National Bureau of Standards, Applied Mathematics Series No. 55, 1045 p.

PROGRAM LISTING

```
C=================================================  HARRIS 05/08/80  ===
C
C  COMPUTER ROUTINES FOR PROBABILITY DISTRIBUTIONS,
C  RANDOM NUMBERS, AND RELATED FUNCTIONS.
C
C  W.KIRBY    OPEN FILE REPORT 80-448    JUNE 1980
C
C
C   ********   FORTRAN 77  VERSION  FOR  HARRIS  S-125/6   ********
C
C==================================================
C==================================================  BESFIO       ======
      FUNCTION BESFIO(X)
C
C        PURPOSE
C           COMPUTE THE MODIFIED BESSEL FUNCTION I OF ORDER ZERO
C
C        USAGE
C          FUNCTION = BESFIO(X)
C
C        DESCRIPTION OF PARAMETERS
C           X    -GIVEN ARGUMENT OF THE BESSEL FUNCTION I OF ORDER 0
C
C        REMARKS
C           LARGE VALUES OF THE ARGUMENT MAY CAUSE OVERFLOW IN THE
C           BUILTIN EXP-FUNCTION
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C          NONE
C
C        METHOD
C          POLYNOMIAL APPROXIMATIONS GIVEN BY E.E. ALLEN ARE USED FOR
C          CALCULATION.
C          FOR REFERENCE SEE
C          M. ABRAMOWITZ AND I.A. STEGUN,'HANDBOOK OF MATHEMATICAL
C          FUNCTIONS', U.S. DEPARTMENT OF COMMERCE, NATIONAL BUREAU OF
C          STANDARDS APPLIED MATHEMATICS SERIES, 1966, P.378.
C        REVISIONS -- COPIED WITH MINOR CHANGES FROM IBM-SSP-III. WK /77
C
C     ...............................................................
C
      RIO=ABS(X)
      IF(RIO-3.75)1,1,2
    1 Z=X*X*7.111111E-2
      RIO=((((( 4.5813E-3*Z+3.60768E-2)*Z+2.659732E-1)*Z+1.206749E0)*Z
     1+3.089942E0)*Z+3.515623E0)*Z+1.
      BESFIO=RIO
      RETURN
    2 Z=3.75/RIO
      RIO= EXP(RIO)/SQRT(RIO)*((((((((3.92377E-3*Z-1.647633E-2)*Z
     1+2.635537E-2)*Z-2.057706E-2)*Z+9.16281E-3)*Z-1.57565E-3)*Z
     2+2.25319E-3)*Z+1.328592E-2)*Z+3.989423E-1)
      BESFIO=RIO
      RETURN
      END
C==================================================
C==================================================  BETAP-S3     ====
      FUNCTION BETAP (X, A, B)
C
C
C  INCOMPLETE BETA FUNCTION - BETA PROBABILITY DISTRIBUTION
```

```
C
C  BETAP(X,A,B)  = I SUB X  (A, B)
C               = INTEGRAL  (FROM 0 TO X)  OF  ((T**(A-1))*((1-T)**(B-1))
C                      *DT)   / BETA(A,B)
C               = PROB(BETA R.V. WITH PARAMS A,B .LE. X)
C  PRAMS A, B MUST BE .GT. 0.  ELSE BETAP = -1E10.
C
C  METHOD - SERIES EXPANSION IN HANDBK MATH FUN NBS.
C           ERROR LESS THAN 10**(-IDIGIT)
C  SUBPGMS USED - DLGAMA, DLOG, DEXP (IBM LIBRARY)   , SQRT
C
C  SOURCE - STANFORD U COMPUTATION CTR/ CAMPUS FAC. (R. W. CARR, 1969)
C  REVISIONS - 2/79 WK.
C
      IMPLICIT DOUBLE PRECISION (D)
      COMMON / IERSWT/ IERR
      DATA IDIGIT / 7 /
C
  100 JCON=0
      DSUM=0.0
C  CHECK IF X IS IN RANGE (0.0,1.0); IF SO GO TO 130
      IF(X.GT.0. .AND. X.LT.1.) GO TO 130
      IF  (X.GE.1.) DSUM=1.
      GO TO 290
C  CHECK TO SEE THAT A AND B ARE POSITIVE; IF SO GO TO 150
  130 IF(A.GT.0. .AND. B.GT.0.) GO TO 150
      IERR = 3
      DSUM = -1D10
      GO TO 290
  150 DA=A
      DB=B
      DX=X
C  IF B IS LARGE COMPARED TO A THEN WE MUST CONVERT  (50 IS ARBITRARY)
  200 IF(DB/DA .LT. (1.DO-DX)/DX) GO TO 210
      DN = DA
      DA=DB
      DB = DN
      DX=1.DO-DX
      JCON=1-JCON
C   CALCULATE THE COEFFICIENT OF THE SERIES EXPANSION
  210 DLX=DLOG(DX)
      DLX1=DLOG(1.DO-DX)
      DLGAB=DLGAMA(DA+DB)
      DLA=DLOG(DA)
      DLGA=DLGAMA(DA)
      DLGB=DLGAMA(DB)
      DLCO=DLX*DA + DLX1*DB + DLGAB -(DLA+DLGA+DLGB)
      DSUM=0.DO
C   IF THE COEFFICIENT IS TOO SMALL LET IT BE ZERO AND QUIT
      IF  (DLCO.LT.-150.DO) GO TO 255
      DCO=DEXP(DLCO)
C   CALCULATE CONVERGENCE CONDITION FOR REQUESTED ACCURRACY
      DEPS=(.5DO*.1DO**IDIGIT)
C  CALCULATE TERM(0)
      DZ1=((DA+DB)/(DA+1.DO))*DX
C  ADD FIRST TERM TO SUM
      DSUM=1.DO+ DZ1
      DN=0.DO
      DAB1=DA+DB+1.DO
      DA2=DA+2.DO
C  START CONVERGENCE LOOP
```

```
  240 DY=(DAB1+DN)/(DA2+DN)*DX
      DZ1=DZ1*DY
      DSUM=DSUM+DZ1
      DN=DN+1.D0
C  TEST FOR CONVERGENCE
      IF(DZ1.GT.DEPS*DSUM) GO TO 240
C  MULTIPLY SUM BY COEFFICIENT
      DSUM=DSUM*DCO
  255 IF(JCON.EQ.1) DSUM=1.D0-DSUM
      IF(DSUM.LT.0.D0) DSUM=0.D0
      IF(DSUM.GT.1.D0) DSUM=1.D0
  290 BETAP = DSUM
      STUTPB = DSUM
      SNEFPB = DSUM
      RETURN
      END
      FUNCTION SNEFPB(F, N1, N2)
C     SNEFPB(F,N1,N2)= PR(Y<=F)WHERE Y IS A RANDOM VARIABLE WITH
C                      AN F-DISTRIBUTION WITH N1,N2 DEGREES OF FREEDOM.
      X = 1. - N2/(N2+N1*F)
      A = .5 * N1
      B = .5 * N2
      SNEFPB = BETAP(X, A, B)
      RETURN
      END
      FUNCTION STUTPB(T, N)
C     STUTPB(T,N)=PR(Y <=T) WHERE Y IS A RANDOM VARIABLE WITH
C                 A T-DISTRIBUTION WITH N DEGREES OF FREEDOM.
      A = .5 * N
      B = A
      X = .5*(1. + T/SQRT(N+T**2))
      STUTPB = BETAP(X, A, B)
      RETURN
      END
C=====================================================
C=====================================================  CHISQP      ======
      FUNCTION CHISQP(X,N)
C
C  CHI-SQUARED PROBABILITIES FOR N DEG OF FR.
C  CHISQP(X,N) = PROB(CHISQUARE WITH N D.F.  .LE.  X)
C
C  REF - HILL AND PIKE, CACM 10(4)243 (APRIL 67), ACM ALG 299.
C        NBS HANDBK MATH FUN 26.4.(8 AND 14).
C  USGS -  WKIRBY 2/79.
C
      DOUBLE PRECISION SUM
      DATA GAMHAF / 1.7724538 /    ,  NWH / 75 /
C
      CHISQP = 0.
      IF(X.LT.1E-7)GO TO 900
      CHISQP = 1.
      IF(N.LE.0) GO TO 900
      IF(N.GE.NWH)GO TO 500
      IF(X.GT.200.) GO TO 900
C
      IF(MOD(N,2).EQ.0) GO TO 10
      CHISQP = 2.*GAUSCF(SQRT(X)) - 1.
      IF(N.LE.1)GO TO 900
      TERM = 1./(SQRT(.5*X)*GAMHAF)
      N2 = N/2
      C = 0.5
```

```
      GO TO 20
   10 CHISQP = 1. - EXP(-.5*X)
      IF(N.LE.2) GO TO 900
      TERM = 1.
      N2 = N/2 - 1
      C = 0.
   20 SUM = ODO
      X2 =0.5*X
      DO 50 I = 1, N2
      TERM = TERM*X2/(I-C)
   50 SUM = SUM + TERM
      CHISQP = CHISQP - EXP(-.5*X)*SUM
      GO TO 900
C
  500 CHISQP = GAUSCF(SQRT(4.5*N)*((X/N)**(.3333333) - (1.-1./(4.5*N))))
C
  900 RETURN
      END
C=================================================
C=================================================  CHISQX        ======
      FUNCTION CHISQX(PCUM, N)
C  CHI-SQUARE QUANTILES FOR  N  DEGREES OF FREEDOM AND CUMU-
C    LATIVE (NONEXCEEDANCE) PROB  PCUM.  (PCUM = 1. - TAIL PROB)
C
C  REF - R.B.GOLDSTEIN, CACM 16(8)483 (8/73), ACM ALG 451.  (CERT. 2/75)
C   USGS - WKIRBY 2/79.
C
      DIMENSION C(21), A(19)
      DATA C(1)/1.565326E-3/, C(2)/1.060438E-3/,
     * C(3)/-6.950356E-3/, C(4)/-1.323293E-2/,
     * C(5)/2.277679E-2/, C(6)/-8.986007E-3/,
     * C(7)/-1.513904E-2/, C(8)/2.530010E-3/,
     * C(9)/-1.430117E-3/, C(10)/5.169654E-3/,
     * C(11)/-1.153761E-2/, C(12)/1.128186E-2/,
     * C(13)/2.607083E-2/, C(14)/-0.2237368/,
     * C(15)/9.783499E-5/, C(16)/-8.426812E-4/,
     * C(17)/3.125580E-3/, C(18)/-8.550369E-3/,
     * C(19)/1.348028E-4/, C(20)/0.4713941/, C(21)/1.0000886/
      DATA A(1)/1.264616E-2/, A(2)/-1.425286E-2/,
     * A(3)/1.400483E-2/, A(4)/-5.886090E-3/,
     * A(5)/-1.091214E-2/, A(6)/-2.304527E-2/,
     * A(7)/3.135411E-3/, A(8)/-2.728484E-4/,
     *A(9)/-9.699681E-3/, A(10)/1.316872E-2/,
     * A(11)/2.618914E-2/, A(12)/-0.2222222/,
     * A(13)/5.406674E-5/, A(14)/3.483789E-5/,
     * A(15)/-7.274761E-4/, A(16)/3.292181E-3/,
     * A(17)/-8.729713E-3/, A(18)/0.4714015/, A(19)/1./
      P = 1DO - DBLE(PCUM)
      CHISQX = 0.
      IF(N.LT.1) RETURN
      IF (N-2) 10, 20, 30
   10 CHISQX = GAUSAB(.5*P)**2
      RETURN
   20 CHISQX = -2.*ALOG(P)
      RETURN
   30 F = N
      F1 = 1./F
      T = GAUSAB(1.-P)
      F2 = SQRT(F1)*T
      IF (N.GE.(2+INT(4.*ABS(T)))) GO TO 40
      CHISQX=(((((((C(1)*F2+C(2))*F2+C(3))*F2+C(4))*F2
```

```
     * +C(5))*F2+C(6))*F2+C(7))*F1+((((((C(8)+C(9)*F2)*F2
     * +C(10))*F2+C(11))*F2+C(12))*F2+C(13))*F2+C(14)))*F1 +
     *  (((((C(15)*F2+C(16))*F2+C(17))*F2+C(18))*F2
     * +C(19))*F2+C(20))*F2+C(21)
      GO TO 50
   40 CHISQX=(((A(1)+A(2)*F2)*F1+(((A(3)+A(4)*F2)*F2
     * +A(5))*F2+A(6)))*F1+(((((A(7)+A(8)*F2)*F2+A(9))*F2
     * +A(10))*F2+A(11))*F2+A(12)))*F1 +  (((((A(13)*F2
     * +A(14))*F2+A(15))*F2+A(16))*F2+A(17))*F2*F2
     * +A(18))*F2+A(19)
   50 CHISQX = F*(CHISQX**3)
      RETURN
      END
C=================================================
C=================================================  DGAMMA       ======
      FUNCTION DGAMMA(XX)
C  DOUBLEPRECISION GAMMA FUNCTION.
C  SAME AS GAMMA EXCEPT ALL VARIABLES ARE
      DOUBLE PRECISION   XX, GX, X, DGAMMA, Y, GY
      IF(XX-34.)6,6,4
    4 IER=2
      DGAMMA = 1D38
      RETURN
    6 X=XX
      ERR=1.0E-6
      IER=0
      GX=1.0
      IF(X-2.0)50,50,15
   10 IF(X-2.0)110,110,15
   15 X=X-1.0
      GX=GX*X
      GO TO 10
   50 IF(X-1.0)60,120,110
C
C        SEE IF X IS NEAR NEGATIVE INTEGER OR ZERO
C
   60 IF(X-ERR)62,62,80
   62 Y=DINT(X)-X
      IF(DABS(Y)-ERR)130,130,64
   64 IF(1.0-Y-ERR)130,130,70
C
C        X NOT NEAR A NEGATIVE INTEGER OR ZERO
C
   70 IF(X-1.0)80,80,110
   80 GX=GX/X
      X=X+1.0
      GO TO 70
  110 Y=X-1.0
      GY=1.0+Y*(-0.5771017+Y*(+0.9858540+Y*(-0.8764218+Y*(+0.8328212+
     1Y*(-0.5684729+Y*(+0.2548205+Y*(-0.05149930))))))
      GX=GX*GY
  120 DGAMMA = GX
      RETURN
  130 IER=1
      DGAMMA = 1D38
      RETURN
      END
C=================================================
C=================================================  DLGAMA       ======
      FUNCTION DLGAMA(XX)
C
```

```
C         PURPOSE
C            COMPUTES THE DOUBLE PRECISION NATURAL LOGARITHM OF THE
C            GAMMA FUNCTION OF A POSITIVE DOUBLE PRECISE ARGUMENT.
C
C         USAGE
C            FUNCTION = DLGAMA(XX)
C            DECLARE DLGAMA AND XX DOUBLE PRECISION IN CALLING PROGRAM.
C
C         REMARKS
C            FOR INVALID ARGUMENT .LT. 10**-9 OR .GT. 10**35,
C            DLGAMA = 1E38 IS RETURNED WITHOUT MESSAGE.
C
C         SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C            NONE
C
C         METHOD -- COPIED FROM IBM-SSP
C            THE EULER-MCLAURIN EXPANSION TO THE SEVENTH DERIVATIVE TERM
C            IS USED, AS GIVEN BY M. ABRAMOWITZ AND I.A. STEGUN,
C            'HANDBOOK OF MATHEMATICAL FUNCTIONS', U. S. DEPARTMENT OF
C            COMMERCE, NATIONAL BUREAU OF STANDARDS APPLIED MATHEMATICS
C            SERIES, 1966, EQUATION 6.1.41.
C
C     ..................................................................
C
      DOUBLE PRECISION XX,ZZ,TERM,RZ2,DLGAMA
      IER=0
      ZZ=XX
      IF(XX-1D10) 2,2,1
    1 IF(XX-1D35) 8,9,9
C
C         SEE IF XX IS NEAR ZERO OR NEGATIVE
C
    2 IF(XX-1.D-9) 3,3,4
    3 IER=-1
      DLGAMA = +1D38
      GO TO 10
C
C         XX GREATER THAN ZERO AND LESS THAN OR EQUAL TO 1.D+10
C
    4 TERM=1.DO
    5 IF(ZZ-18.DO) 6,6,7
    6 TERM=TERM*ZZ
      ZZ=ZZ+1.DO
      GO TO 5
    7 RZ2=1.DO/ZZ**2
      DLGAMA =(ZZ-0.5DO)*DLOG(ZZ)-ZZ +0.9189385332046727 -DLOG(TERM)+
     1(1.DO/ZZ)*(.8333333333333333D-1 -(RZ2*(.2777777777777777D-2 +(RZ2*
     2(.7936507936507936D-3 -(RZ2*(.5952380952380952D-3)))))))
      GO TO 10
C
C         XX GREATER THAN 1.D+10 AND LESS THAN 1.D+70
C
    8 DLGAMA = ZZ*(DLOG(ZZ) - 1DO)
      GO TO 10
C
C         XX GREATER THAN OR EQUAL TO 1.D+70
C
    9 IER=+1
      DLGAMA = 1D38
   10 RETURN
C
```

```
      END
C==================================================
C==================================================  ERF        ========
      FUNCTION ERF(XX)
C
C          COMPUTES THE ERROR FUNCTION --
C
C          2/SQRT(PI)*INTEGRAL(0 TO X)  EXP(-U**2)DU.
C             = 2.*GAUSSIANPROB  (SQRT(2)*X)  - 1.
C
C     RATIONAL APPROX FROM NBS HAND MATH FUN 26.2.17 VIA IBM SSP NDTR.
C
      X=1.4142136*XX
      AX=ABS(X)
      T=1.0/(1.0+.2316419*AX)
      D=0.3989423*EXP(-X*X/2.0)
      P = 1.0 - D*T*((((1.330274*T - 1.821256)*T + 1.781478)*T -
     1  0.3565638)*T + 0.3193815)
      IF(X.LT.0.)  P = 1.0 - P
      ERF=2.*P-1.
      RETURN
      END
C==================================================
C==================================================  EXPIF        =======
      FUNCTION EXPIF(X)
C
C          COMPUTES THE EXPONENTIAL INTEGRAL -EI(-X)
C            = INTEGRAL(EXP(-T)/T, SUMMED OVER T FROM X TO INFINITY).
C
C       USAGE
C          FUNCTION = EXPIF(X)
C
C       DESCRIPTION OF PARAMETERS
C          X      - ARGUMENT OF EXPONENTIAL INTEGRAL
C
C       REMARKS
C          X GT 170 (X LT -174) MAY CAUSE UNDERFLOW (OVERFLOW)
C          WITH THE EXPONENTIAL FUNCTION
C          FOR X = 0 THE RESULT VALUE IS SET TO -1.E75
C
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C          NONE
C
C       METHOD
C          DEFINITION
C          RES=INTEGRAL(EXP(-T)/T, SUMMED OVER T FROM X TO INFINITY).
C          EVALUATION
C          THREE DIFFERENT RATIONAL APPROXIMATIONS ARE USED IN THE
C          RANGES 1 LE X, X LE -9 AND -9 LT X LE -3 RESPECTIVELY,
C          A POLYNOMIAL APPROXIMATION IS USED IN -3 LT X LT 1.
C       REVISIONS -- COPIED WITH MINOR CHANGES FROM IBM-SSP-III. WK /77
C
C     ...............................................................
C
      IF(X-1.)2,1,1
    1 Y=1./X
      AUX=1.-Y*(((Y+3.377358E0)*Y+2.052156E0)*Y+2.709479E-1)/((((Y*
     11.072553E0+5.716943E0)*Y+6.945239E0)*Y+2.593888E0)*Y+2.709496E-1)
      RES=AUX*Y*EXP(-X)
      EXPIF=RES
      RETURN
```

```
    2 IF(X+3.)6,6,3
    3 AUX=(((((((7.122452E-7*X-1.766345E-6)*X+2.928433E-5)*X-2.335379E-4
     1)*X+1.664156E-3)*X-1.041576E-2)*X+5.555682E-2)*X-2.500001E-1)*X
     2+9.999999E-1
      EXPIF = +1E38
      IF(X)4,5,4
    4 RES=X*AUX-ALOG(ABS(X))-5.772157E-1
      EXPIF=RES
    5 RETURN
    6 IF(X+9.)8,8,7
    7 AUX=1.-((((5.176245E-2*X+3.061037E0)*X+3.243665E1)*X+2.244234E2)*X
     1+2.486697E2)/((((X+3.995161E0)*X+3.893944E1)*X+2.263818E1)*X
     2+1.807837E2)
      GOTO 9
    8 Y=9./X
      AUX=1.-Y*(((Y+7.659824E-1)*Y-7.271015E-1)*Y-1.080693E0)/((((Y
     1*2.518750E0+1.122927E1)*Y+5.921405E0)*Y-8.666702E0)*Y-9.724216E0)
    9 RES=AUX*EXP(-X)/X
      EXPIF=RES
      RETURN
      END
C==================================================
C==================================================  GAMMA      =======
      FUNCTION GAMMA(XX)
C
C            COMPUTES THE GAMMA FUNCTION FOR A GIVEN ARGUMENT
C
C        USAGE
C            FUNCTION = GAMMA(XX)
C
C        DESCRIPTION OF PARAMETERS
C            XX -THE ARGUMENT FOR THE GAMMA FUNCTION
C
C        REMARKS
C            ERROR CODE IER = 1 IF XX IS NEAR A NONPOSITIVE INTEGER.
C            IER = 2 IF XX IS GREATER THAN 34.  IN EITHER CASE GAMMA
C            IS SET TO 1E38 WITHOUT MESSAGE.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C            NONE
C
C        METHOD
C            THE RECURSION RELATION AND POLYNOMIAL APPROXIMATION
C            BY C.HASTINGS,JR., 'APPROXIMATIONS FOR DIGITAL COMPUTERS',
C            PRINCETON UNIVERSITY PRESS, 1955
C        REVISIONS -- COPIED WITH MINOR CHANGES FROM IBM-SSP-III. WK /77
C
C     ..............................................................
C
      IF(XX-34.)6,6,4
    4 IER=2
      GAMMA = 1E38
      RETURN
    6 X=XX
      ERR=1.0E-6
      IER=0
      GX=1.0
      IF(X-2.0)50,50,15
   10 IF(X-2.0)110,110,15
   15 X=X-1.0
      GX=GX*X
```

```
      GO TO 10
   50 IF(X-1.0)60,120,110
C
C        SEE IF X IS NEAR NEGATIVE INTEGER OR ZERO
C
   60 IF(X-ERR)62,62,80
   62 Y=FLOAT(INT(X))-X
      IF(ABS(Y)-ERR)130,130,64
   64 IF(1.0-Y-ERR)130,130,70
C
C        X NOT NEAR A NEGATIVE INTEGER OR ZERO
C
   70 IF(X-1.0)80,80,110
   80 GX=GX/X
      X=X+1.0
      GO TO 70
  110 Y=X-1.0
      GY=1.0+Y*(-0.5771017+Y*(+0.9858540+Y*(-0.8764218+Y*(+0.8328212+
     1Y*(-0.5684729+Y*(+0.2548205+Y*(-0.05149930)))))))
      GX=GX*GY
  120 GAMMA = GX
      RETURN
  130 IER=1
      GAMMA = 1E38
      RETURN
      END
C================================================
C================================================  GAMMAP      ======
      FUNCTION GAMMAP(AAA, XXX)
C
C  GAMMA PROBABILITY DISTRIBUTION, INCOMPLETE GAMMA FUNCTION
C   = PROB GAMMA RV WITH SHAPE FACTOR AAA IS .LE. XXX
C   = INTEGRAL (0 TO XXX) OF (U**(AAA-1)*EXP(-U)*DU) / GAMMA(AAA)
C
C        REMARKS
C           AAA MUST BE .GT. 0. ELSE GAMMAP = UNIT STEP FCN OF X.
C           AAA MUST BE .LT. 1E5 ELSE 1E5 IS USED.
C           GAMMAP=+1E75(IER=+1) IF CDTR ALGORITHM FAILS.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           DLGAMA, ERF    (IBM FTN IV LIB)
C
C        METHOD
C           USES IBM-SSP CHI-SQUARE CODE CDTR
C              GAMMA(A,X) = CDTR(2*X, 2*A)    (A .GE. 0.25)
C           FOR  A .LT. 0.25 --  (NBS MATH FUN 6.5.21) --
C              GAMMA(A,X) = GAMMA(A+1,X) + DENSITY(A+1,X)
C           REFER TO R.E. BARGMANN AND S.P. GHOSH, STATISTICAL
C           DISTRIBUTION PROGRAMS FOR A COMPUTER LANGUAGE,
C           IBM RESEARCH REPORT RC-1094, 1963.
C
C  REVISIONS - REPLACED IBM-SSP FCN REFS BY IBM FORT LIB REFS. WK12/76.
C          CONVERTED CDTR  TO GAMMAP. WK2/79.
C
C     ...................................................................
C
      DOUBLE PRECISION XX,DLXX,X2,DLX2,GG,G2,DLT3,THETA,THP1,
     1GLG2,DD,T11,SER,CC,XI,FAC,TLOG,TERM,GTH,A2,A,B,C,DT2,DT3,THPI
      DOUBLEPRECISION DLGAMA
C
      COMMON / IERSWT / IERRR
```

```
C
      IF (AAA .GT. 0.) GO TO 5
      GAMMAP = 1.
      IF(XXX.LE.0.) GAMMAP = 0.
      RETURN
   5  CONTINUE
C  CONVERT GAMMA(XXX,AAA) TO CDTR(X,G) AND NOTE RECURSION
      X = 2.*XXX
      IF(X.LT.0.) X = 1E-10
      DENFAC = 2.
      RECURS = 0.
      IF(AAA.LT.0.25)RECURS = 1.
      G = 2.*(AAA + RECURS)
      IF(G.GT.2E5) G = 2E5
C
C        TEST FOR VALID INPUT DATA
C
      IF(G-(.5-1.E-5)) 590,10,10
   10 IF(G-2.E+5) 20,20,590
   20 IF(X) 590,30,30
C
C        TEST FOR X NEAR 0.0
C
   30 IF(X-1.E-8) 40,40,80
   40 P=0.0
      IF(G-2.) 50,60,70
   50 D=1.E38
      GO TO 610
   60 D=0.5
      GO TO 610
   70 D=0.0
      GO TO 610
C
C        TEST FOR X GREATER THAN 1.E+6
C
   80 IF(X-1.E+6) 100,100,90
   90 D=0.0
      P=1.0
      GO TO 610
C
C        SET PROGRAM PARAMETERS
C
  100 XX=DBLE(X)
      DLXX=DLOG(XX)
      X2=XX/2.D0
      DLX2=DLOG(X2)
      GG=DBLE(G)
      G2=GG/2.D0
C
C        COMPUTE ORDINATE
C
      GLG2=DLGAMA(G2)
      DD=(G2-1.D0)*DLXX-X2-G2*.6931471805599453 -GLG2
      IF(DD-1.68D02) 110,110,120
  110 IF(DD+1.68D02) 130,130,140
  120 D=1.E38
      GO TO 150
  130 D=0.0
      GO TO 150
  140 DD=DEXP(DD)
      DD = DD * DENFAC
```

```
      D=SNGL(DD)
C
C        TEST FOR G GREATER THAN 1000.0
C        TEST FOR X GREATER THAN 2000.0
C
  150 IF(G-1000.)  160,160,180
  160 IF(X-2000.)  190,190,170
  170 P=1.0
      GO TO 610
  180 A=DLOG(XX/GG)/3.DO
      A=DEXP(A)
      B=2.DO/(9.DO*GG)
      C=(A-1.DO+B)/DSQRT(B)
      SC=SNGL(C)
      P=.5*(1.+ERF(SC*.7071068))
      GO TO 490
C
C        COMPUTE THETA
C
  190 K= IDINT(G2)
      THETA=G2-DFLOAT(K)
      IF(THETA-1.D-8)  200,200,210
  200 THETA=0.DO
  210 THP1=THETA+1.DO
C
C        SELECT METHOD OF COMPUTING T1
C
      IF(THETA)  230,230,220
  220 IF(XX-10.DO)  260,260,320
C
C        COMPUTE T1 FOR THETA EQUALS 0.0
C
  230 IF(X2-1.68D02)  250,240,240
  240 T1=1.0
      GO TO 400
  250 T11=1.DO-DEXP(-X2)
      T1=SNGL(T11)
      GO TO 400
C
C        COMPUTE T1 FOR THETA GREATER THAN 0.0 AND
C        X LESS THAN OR EQUAL TO 10.0
C
  260 SER=X2*(1.DO/THP1 -X2/(THP1+1.DO))
      J=+1
      CC=DFLOAT(J)
      DO 270  IT1=3,30
      XI=DFLOAT(IT1)
      FAC=DLGAMA(XI)
      TLOG= XI*DLX2-FAC-DLOG(XI+THETA)
      TERM=DEXP(TLOG)
      TERM=DSIGN(TERM,CC)
      SER=SER+TERM
      CC=-CC
      IF(DABS(TERM)-1.D-9)  280,270,270
  270 CONTINUE
      GO TO 600
  280 IF(SER)  600,600,290
  290 GTH=DLGAMA(THP1)
      TLOG=THETA*DLX2+DLOG(SER)-GTH
      IF(TLOG+1.68D02)  300,300,310
  300 T1=0.0
```

```
      GO TO 400
  310 T11=DEXP(TLOG)
      T1=SNGL(T11)
      GO TO 400
C
C        COMPUTE T1 FOR THETA GREATER THAN 0.0 AND
C        X GREATER THAN 10.0 AND LESS THAN 2000.0
C
  320 A2=0.D0
      DO 340 I=1,25
      XI=DFLOAT(I)
      GTH=DLGAMA(THP1)
      T11=-(13.D0*XX)/XI +THP1*DLOG(13.D0*XX/XI) -GTH-DLOG(XI)
      IF(T11+1.68D02) 340,340,330
  330 T11=DEXP(T11)
      A2=A2+T11
  340 CONTINUE
      A=1.01282051+THETA/156.D0-XX/312.D0
      B=DABS(A)
      C= -X2+THP1*DLX2+DLOG(B)-GTH-3.951243718581427
      IF(C+1.68D02) 370,370,350
  350 IF (A) 360,370,380
  360 C=-DEXP(C)
      GO TO 390
  370 C=0.D0
      GO TO 390
  380 C=DEXP(C)
  390 C=A2+C
      T11=1.D0-C
      T1=SNGL(T11)
C
C        SELECT PROPER EXPRESSION FOR P
C
  400 IF(G-2.) 420,410,410
  410 IF(G-4.) 450,460,460
C
C        COMPUTE P FOR G GREATER THAN ZERO AND LESS THAN 2.0
C
  420 GTH=DLGAMA(THP1)
      DT2=THETA*DLXX-X2-THP1*.6931471805599453 -GTH
      P = T1
      IF(DT2 .GT. -168D0) P = P + 2.*DEXP(DT2)
      GO TO 490
C
C        COMPUTE P FOR G GREATER THAN OR EQUAL TO 2.0
C        AND LESS THAN 4.0
C                    ( AAA .LT. 0.25  --  POSSIBLE RECURSION )
C
  450 CONTINUE
      P = T11 + RECURS*DD
      GO TO 490
C
C        COMPUTE P FOR G GREATER THAN OR EQUAL TO 4.0
C        AND LESS THAN OR EQUAL TO 1000.0
C
  460 DT3=0.D0
      DO 480 I3=2,K
      THPI=DFLOAT(I3)+THETA
      GTH=DLGAMA(THPI)
      DLT3=THPI*DLX2-DLXX-X2-GTH
      IF(DLT3+1.68D02) 480,480,470
```

```
  470 DT3=DT3+DEXP(DLT3)
  480 CONTINUE
      T3=SNGL(DT3)
      P=T1-T3-T3
C
C        SET ERROR INDICATOR
C
  490 IF(P) 500,520,520
  500 IF(ABS(P)-1.E-7) 510,510,600
  510 P=0.0
      GO TO 610
  520 IF(1.-P) 530,550,550
  530 IF(ABS(1.-P)-1.E-7) 540,540,600
  540 P=1.0
      GO TO 610
  550 IF(P-1.E-8) 560,560,570
  560 P=0.0
      GO TO 610
  570 IF((1.0-P)-1.E-8) 580,580,610
  580 P=1.0
      GO TO 610
  590 IER=-1
      D=-1.E38
      P=-1.E38
      GO TO 620
  600 IER=+1
      P= 1.E38
      GO TO 620
  610 IER=0
  620 CONTINUE
      IF(IER.NE.0)IERRR=IER
      GAMMAP = P
      RETURN
      END
C===============================================
C===============================================  GAUSEX      ======
      FUNCTION GAUSEX(EXPROB)
C
C  GAUSSIAN PROBABILITY FUNCTIONS   W.KIRBY  JUNE 71
C     GAUSEX=VALUE EXCEEDED WITH PROB EXPROB
C     GAUSAB=VALUE (NOT EXCEEDED) WITH PROBCUMPROB
C     GAUSCF=CUMULATIVE PROBABILITY FUNCTION
C     GAUSDY=DENSITY FUNCTION
C  GAUSCF MODIFIED 740906 WK -- REPLACED ERF FCN REF BY RATIONAL APPRX N
C    ALSO REMOVED DOUBLE PRECISION FROM GAUSEX AND GAUSAB.
C  76-05-04 WK -- TRAP UNDERFLOWS IN EXP IN GUASCF AND DY.
C
      DATA  XLIM  /  18.3 /
      DATAC0,C1,C2/2.51551700, .8028530000, .0103280000/
      DATAD1,D2,D3/1.432788000, .1892690000, .0013080000/
C
      P=EXPROB
    2 IF(P.LT.1.0)GOTO10
      GAUSEX=-10.
      RETURN
   10 IF(P.GT.0.)GOTO20
      GAUSEX=+10.
      RETURN
   20 PR=P
      IF(P.GT..5)PR=1.00-PR
      T= SQRT(-2.00*ALOG(PR))
```

```
      GAUSEX=T-(CO+T*(C1+T*C2))/(1.DO+T*(D1+T*(D2+T*D3)))
      IF(P.GT..5)GAUSEX=-GAUSEX
      RETURN
C
      ENTRYGAUSAB(CUMPRB)
      GAUSAB = 0.
      P=1.-CUMPRB
      GOTO2
C
      ENTRY GAUSCF(XX)
      AX=ABS(XX)
      GAUSCF=1.
      IF(AX.GT.XLIM)GOTO101
      T=1.0/(1.0+.2316419*AX)
      D=0.3989423*EXP(-XX*XX*.5)
      GAUSCF=1.-D*T*((((1.330274*T - 1.821256)*T + 1.781478)*T -
     1  0.3565638)*T + 0.3193815)
  101 IF(XX.LT.0)GAUSCF=1.-GAUSCF
      RETURN
C
      ENTRY GAUSDY  (XX)
      GAUSDY=0.
      IF(ABS(XX).GT.XLIM)  RETURN
      GAUSDY=.3989423*EXP(-.500*XX*XX)
      RETURN
      END
C=================================================
C=================================================  GAUSSF       ======
      FUNCTION GAUSSF(IRAN)
C  GAUSSIAN RANDOM NUMBER FCN -  RANDUB/BOX-MULLER
C  REVISION AND EXTENSION OF JIM SLACK'S GAUSSV.       WK 8/75. 4/78.
C  USES SINGLE PRECISION AND CALLS ONLY RANDUV, NOT RANDUB.
C  REVISED FOR HARRIS.  9/79 WK.
C     ..............................................................
      LOGICAL NONE
      DIMENSION W(2)
      INTEGER*6 IRAN
      DATA TWOPI/6.2831853/, FNEG2/-2./, ANGL/1E38/, NONE/.TRUE./
C
      IF(NONE)GOTO10
      GAUSSF=DIST*SIN(ANGL)
      NONE=.TRUE.
      RETURN
   10 CALLRANDUV(IRAN,W,2)
      DIST=SQRT(FNEG2*ALOG(W(2)))
      ANGL=TWOPI*W(1)
      GAUSSF=DIST*COS(ANGL)
      NONE=.FALSE.
      RETURN
      END
C=================================================
C=================================================  GAUSSV       ======
      SUBROUTINE GAUSSV(IRAN,V,N)
C  HARRIS COMPUTER VERSION 9/79 WK.
C  GAUSSIAN RANDOM NBR VECTOR  --  RANDUV / BOX-MULLER
C  NOTE - THIS DISCARDS THE (N+1) GAUSS NUMBER IF N IS ODD.
C  REVISION AND EXTENSION OF JIM SLACK'S GAUSSV.       WK 8/75. 4/78.
C  USES SINGLE PRECISION AND CALLS ONLY RANDUV, NOT RANDUB.
C     ..............................................................
      INTEGER*6 IRAN
      DIMENSION W(2), V(1)
```

```
      DATA TWOPI/6.2831853/, FNEG2/-2./, ANGL/1E38/
      CALLRANDUV(IRAN,V,N)
      L=N/2
      IF(L.LE.0)GOTO110
      DO100I=1,L
      K=2*I
      ANGL=TWOPI*V(K-1)
      DIST=SQRT(FNEG2*ALOG(V(K)))
C NOTE - SIN TAKES 80+ USEC ON 360/65 WHEREAS SQRT TAKES ONLY 60.
C        THUS WE COULD SAVE A FEW USEC BY  USINGSQRT(1.-COS**2)
      V(K-1)=DIST*COS(ANGL)
      V(K  )=DIST*SIN(ANGL)
  100 CONTINUE
      IF(2*L.EQ.N) RETURN
  110 CALL RANDUV(IRAN,W(2),1)
      V(N)=SQRT(FNEG2*ALOG(W(2)))*COS(TWOPI*V(N))
      RETURN
      END
C==================================================
C================================================== GUMBEL    ==========
      FUNCTION GUMBEL(P)
C
C  GUMBEL EXTREME-VALUE TYPE 1
C
      ALP = 1E37
      IF  (P.GT.0.)  ALP = -ALOG(P)
      IF  (ALP.LE.0.)  ALP = 1E-37
      GUMBEL  =  (-ALOG( ALP )-0.577216)/1.282550
      RETURN
C
      ENTRY GUMBEP(X)
      GUMBEP  =  EXP(-EXP(-(1.282550*X+0.577216)))
      RETURN
      END
C==================================================
C=================================================== HARTIV      ======
      SUBROUTINE  HARTIV  (SKU,PD)
C
C  VERSION FOR HARRIS 9/79 HAS LOCAL COPY OF TABLES NOT COMMON.
C  G387 - LOOK UP K OR P IN HARTERS TABLES.  WKIRBY 3/75. 9/76.G387.
C   HARTIV -- HARTER INTERPOLATE VECTOR PD BY 3-PT LAGRANGE INT W/R SKEW
C  REV 9/76 WK - NO LONGER USES SKEW TABLES IN HARTAB
C  ERROR IN IORDER MADE 9/76, CORRECTED 7/77. FOR SKU IN (-.05,-0.)
C
C   --  NOTE- ABS SKEW IS TRUNCATED AT 9. W/OUT COMMENT.
C
C
C  G387 --  HARTERS TABLES OF PEARSON TYPE III DISTRIBUTION.
C         TECHNOMETRICS FEB 69 AND FEB 71.  THESE TABLES
C         HAVE BEEN CHECKED AGAINST ANOTHER SET OF TABLES LOANED BY
C         W. H. SAMMONS OF SOIL CONS. SERVICE.
C         W.KIRBY  USGS-WRD 3/73.
C
C    SKEWS = -0.1, 0.0(0.1)4.8,  5.0(0.2)9.0
C
C  TABULAR PROBABILITIES /    0.00010,
C       0.00050,  0.00100,  0.00200,  0.00500,  0.01000,  0.02000,      0
C       0.02500,  0.04000,  0.05000,  0.10000,  0.20000,  0.30000,      0
C       0.40000,  0.429624, 0.50000,  0.570376, 0.60000,  0.70000,      0
C       0.80000,  0.90000,  0.95000,  0.96000,  0.97500,  0.98000,      0
C       0.99000,  0.99500,  0.99800,  0.99900,  0.99950,  0.99990 /     0
```

```
C
C
C
      DIMENSION PD(31)
C
      DIMENSION PCT(31, 71)
C
      DIMENSION
     $  VM1(31),  V 0(31),  V 1(31),  V 2(31),  V 3(31),  V 4(31),
     $  V 5(31),  V 6(31),  V 7(31),  V 8(31),  V 9(31),  V10(31),

           . . .  REPETITIVE LINES OMITTED  . . .

     $  V74(31),  V76(31),  V78(31),  V80(31),  V82(31),  V84(31),
     $  V86(31),  V88(31),  V90(31)
C
      EQUIVALENCE
     $   (VM1(1),PCT(1, 1)),   (V 0(1),PCT(1, 2)),   (V 1(1),PCT(1, 3)),
     $   (V 2(1),PCT(1, 4)),   (V 3(1),PCT(1, 5)),   (V 4(1),PCT(1, 6)),

                  . . .  REPETITIVE  LINES  OMITTED  . . .

     $   (V80(1),PCT(1,66)),   (V82(1),PCT(1,67)),   (V84(1),PCT(1,68)),
     $   (V86(1),PCT(1,69)),   (V88(1),PCT(1,70)),   (V90(1),PCT(1,71))
C
C
C
      DATA  VM1  /
     #    -3.93453,    -3.45513,    -3.23322,    -2.99978,
     #    -2.66965,    -2.39961,    -2.10697,    -2.00688,

           . . .   TABULAR  DATA  OMITTED   . . .

     #  -0.19277,  0.13431,  0.99519,  1.38213,  2.34242,  2.85782,
     #   4.65277,  6.67443,  9.59243, 11.93509, 14.36528, 20.25402 /
      DATA V90 /   -0.22222,
     #  -0.22222, -0.22222, -0.22222, -0.22222, -0.22222, -0.22222,
     #  -0.22222, -0.22222, -0.22222, -0.22222, -0.22222, -0.22222,
     #  -0.22222, -0.22222, -0.22222, -0.22219, -0.22214, -0.22030,
     #  -0.19338,  0.11146,  0.95435,  1.33922,  2.30138,  2.82035,
     #   4.63541,  6.68763,  9.65701, 12.04437, 14.52288, 20.53356 /
C
C
C
      IORDER=1
      IF(SKU.LT. 0.0)IORDER=-1
      S=ABS(SKU)
      IF(S.GT.9.01)GOTO500
      IF(S.GT.4.75 .AND. S.LT.4.90)GOTO300
      H=0.1
      IF(S.GT.4.75)H=0.2
      KKTV=50+(S-4.8)/H + .5
      IF(KKTV.GT.70)KKTV=70
      SKTV = 4.8 + H*(KKTV-50)
      P = (S - SKTV)/H
      C0=P*(P-1.)*.5 *FLOAT(IORDER)
      C1=(1.-P**2)    *FLOAT(IORDER)
      C2=P*(P+1.)*.5 *FLOAT(IORDER)
      GOTO400
  300 KKTV=50
      P=S-4.8
```

```
      Q=S-5.0
      R=S-4.7
      C2=P*R*16.66667*FLOAT(IORDER)
      C1=Q*R*(-50.)   *FLOAT(IORDER)
      C =P*Q*33.33333*FLOAT(IORDER)
  400 L=16*(1-IORDER)
      DO410I=1,31
      L=L+IORDER
  410 PD(L)=CO*PCT(I,KKTV-1)  + C1*PCT(I,KKTV) + C2*PCT(I,KKTV+1)
      RETURN
  500 DO520I=1,31
  520 PD(I)=0.
      RETURN
      END
C====================================================
C====================================================  HARTK       =======
      FUNCTION HARTK (P,V)
C  G387 - LOOK UP K OR P IN HARTERS TABLES.  WKIRBY 9/76.
C  HARTK  - LOOKUP STDIZED HARTER K AT CUMULATIVE PROB P
C  HARTP  - LOOKUP CUM (NONEXCEED) PROB P AT STDIZED HARTER K
C  V       - VECTOR OF SKEW-INTERPOLATED HARTER K VALUES (FROM HARTIV)
C  6/78 WK -- USING LOCAL NOT HARTAB COPY OF HARTER TAB PROBS/GAUSS DEV.
      DIMENSION V(1)
C  HARTER TABULAR PROBABILITIES OR  GAUSSIAN DEVIATES -------------
      REAL PROB(31)
      DATA PROB      /   0.00010,
     #    0.00050,  0.00100,  0.00200,  0.00500,  0.01000,  0.02000,
     #    0.02500,  0.04000,  0.05000,  0.10000,  0.20000,  0.30000,
     #    0.40000,  0.429624, 0.50000,  0.570376, 0.60000,  0.70000,
     #    0.80000,  0.90000,  0.95000,  0.96000,  0.97500,  0.98000,
     #    0.99000,  0.99500,  0.99800,  0.99900,  0.99950,  0.99990 /
C  --------------------------------------------------------------
      DATA HUGE / 1E38/
C  ENTRY HARTK -
      IF(P.GT.PROB(31)  .OR. P.LT.PROB(1))GOTO90
      IB=2
      IF(P.GE.0.5)IB=17
      IE=IB+13
      DO10I=IB,IE
      IF(P.LT.PROB(I))GOTO20
   10 CONTINUE
      I=IB+14
   20 HARTK=V(I-1)+(P-PROB(I-1))*(V(I)-V(I-1))/(PROB(I)-PROB(I-1))
      RETURN
C  RETURN + OR - 1E75 IF OUT OF RANGE OF PROB
   90 HARTK=HUGE
      IF(P.LT.PROB(1))HARTK=-HUGE
      RETURN
C
C  ENTRY HARTP -
      ENTRY HARTP(SHK, V)
      IF(SHK.GT.V(31).OR.SHK.LT.V(1))GOTO190
      IB=2
      IF(SHK.GE.V(16))IB=17
      IE=IB+13
      DO110I=IB,IE
      IF(SHK.LT.V(I))GOTO120
  110 CONTINUE
      I=IB+14
  120 DV=V(I)-V(I-1)
      IF(DV.EQ.0.)GOTO180
```

```
      HARTP=PROB(I-1)+(SHK-V(I-1))*(PROB(I)-PROB(I-1))/DV
      RETURN
  180 HARTP=PROB(I)
      IF(V(1).NE.V(31))       RETURN
  190 HARTP=1.
      IF(SHK.LT.V(1))HARTP=0.
      RETURN
      END
C===================================================
C===================================================  HARTKK        ======
      FUNCTION HARTKK(P,V)
C  LOOK UP K IN HARTER TABLES FOR NON-0 SKEW USING 0-SKEW K AS ARGUMENT.
C  MODIFICATION OF HARTK G387 WK 7/77.
C  P AND PROB REFER TO STD NORMAL DEVIATES (K FOR 0 SKEW)
C  HARTKK - LOOKUP STDIZED HARTER K AT STD NORMAL DEVIATE P.
C  V      - VECTOR OF SKEW-INTERPOLATED HARTER K VALUES (FROM HARTIV)
C  6/78 WK -- USING LOCAL NOT HARTAB COPY OF HARTER TAB PROBS/GAUSS DEV.
      DIMENSION V(1)
C  HARTER TABULAR PROBABILITIES OR  GAUSSIAN DEVIATES -------------
      REAL PROB(31)
      DATA PROB     /  -3.71902,
     #  -3.29053, -3.09023, -2.87816, -2.57583, -2.32635, -2.05375,
     #  -1.95996, -1.75069, -1.64485, -1.28155, -0.84162, -0.52440,
     #  -0.25335, -0.17733,  0.0    ,  0.17733,  0.25335,  0.52440,
     #   0.84162,  1.28155,  1.64485,  1.75069,  1.95996,  2.05375,
     #   2.32635,  2.57583,  2.87816,  3.09023,  3.29053,  3.71902 /
C  ------------------------------------------------------------------
      DATA HUGE / 1E38/
      IF(P.GT.PROB(31)  .OR.  P.LT.PROB(1))GOTO90
      IB=2
      IF(P.GE.0.5)IB=17
      IE=IB+13
      DO10I=IB,IE
      IF(P.LT.PROB(I))GOTO20
   10 CONTINUE
      I=IB+14
   20 HARTKK=V(I-1)+(P-PROB(I-1))*(V(I)-V(I-1))/(PROB(I)-PROB(I-1))
      RETURN
C  RETURN + OR - 1E75 IF OUT OF RANGE OF PROB
   90 HARTKK=HUGE
      IF(P.LT.PROB(1))HARTKK=-HUGE
      RETURN
      END
C===================================================
C===================================================  HARTTP        ======
      SUBROUTINE HARTTP(IR, PD)
C
C  VERSION FOR HARRIS 9/79 HAS LOCAL COPY OF TABLES NOT COMMON.
C  G387 - LOOK UP K OR P IN HARTERS TABLES.  WKIRBY 3/75. 9/76.G387.
C
C  HARTTP -- COPY HARTER TABULAR PROBS, IN REVERSE ORDER IF IR.LT.0.
C
      DIMENSION PROB1(31), PD(31)
      DATA PROB 1  /    0.00010,
     #   0.00050,  0.00100,  0.00200,  0.00500,  0.01000,  0.02000,
     #   0.02500,  0.04000,  0.05000,  0.10000,  0.20000,  0.30000,
     #   0.40000,  0.429624, 0.50000,  0.570376, 0.60000,  0.70000,
     #   0.80000,  0.90000,  0.95000,  0.96000,  0.97500,  0.98000,
     #   0.99000,  0.99500,  0.99800,  0.99900,  0.99950,  0.99990 /
C
      IF(IR.LT.0)GOTO850
```

```
      DO810I=1,31
  810 PD(I)=PROB1(I)
      RETURN
  850 DO860I=1,31
  860 PD(I)=PROB1(32-I)
      RETURN
      END
C===================================================
C===================================================  OUTKGB       ======
      FUNCTION OUTKGB (SIG, N)
C* HIGH-OUTLIER CRITERIA FOR NORMAL DTN.  - OUTKGB IS LOW-OUT CRIT.
C* OUTLIER TEST CRITERIA FROM GRUBBS-BECK 11/72 TECHNOMETRICS.
C  SAMPLE SIZES (3,1,100)  SIG LEVELS 10, 5, 2.5, 1  PCT (ONE-SIDED TEST
C  LINEAR INTERPOLATION-EXTRAP FOR N = 101-120-147-180.
C  SIG MAY BE GIVEN AS FRACTION OR PERCENT -- EG, 2.5 OR 0.025.
C  ORIGINAL TABLES FROM CRONSHEY, 7/77.  REVISED AND ENLARGED WK 5,6/78.
C
      DIMENSION OUTG10(100),OUTG5(100),OUTG25(100),OUTG(100,4),
     $  OUTG1(100),  TAIL(2,4)
      EQUIVALENCE(OUTG10(1),OUTG(1,1)),(OUTG5(1),OUTG(1,2)),
     $          (OUTG25(1),OUTG(1,3)),  (OUTG1(1),OUTG(1,4))
      DATA OUTG10/        2*-1E37,
     $1.148,1.425,1.602,1.729,1.828,1.909,1.977,          2.036,2.088,

          . . .  TABULAR  DATA  OMITTED   . . .

     $3.521,3.525,3.529,3.534,3.539,3.543,3.547,3.551,3.555,3.559,3.563,
     $3.567,3.570,3.575,3.579,3.582,3.586,3.589,3.593,3.597,3.600  /
C
      DATA TAIL /3.078,3.144,3.267,3.334,3.444,3.509,3.662,3.727/
C
      SIGG = SIG
      IF(SIGG.GT..99) SIGG = 0.01*SIGG
      ISIG = 0
      IF( ABS(SIGG -  .10  ) .LE. 0.001)  ISIG =  1
      IF( ABS(SIGG -  .05  ) .LE. 0.001)  ISIG =  2
      IF( ABS(SIGG -  .025 ) .LE. 0.001)  ISIG =  3
      IF(ABS(SIGG- .010) .LE. 0.001) ISIG = 4
      OUTKGB = -1E37
      IF(ISIG.EQ.0) RETURN
      IF(N.GT.120) GO TO 10
      NN = MIN0(100, MAX0(1, N))
      OUTKGB = OUTG(NN,ISIG)
      IF(N.GT.100)OUTKGB=OUTKGB + .05*(N-100)*(TAIL(1,ISIG)-OUTKGB)
      RETURN
   10 NN = MIN0(N,180)
      OUTKGB=TAIL(1,ISIG)+(TAIL(2,ISIG)-TAIL(1,ISIG))*(NN-120)/27.
      RETURN
      END
C===================================================
C===================================================  PRPLOT       ======
      SUBROUTINE PRPLOT
C
C  USGS PROG NO B524 - CCD USER SERVICES
C   PETE SMIDINGER   SUMMER 1966  MATH & COMP BR  GSFC  NASA
C  REVISION 730604 WKIRBY USGS-WRD ACTIVATE PLTAPE ENTRY AND
C         REPLACE DSRN = 6 (CONSTANT)  BY VARIABLE IJTAPE
C  REVISION 8/24/78 WK - PUT IJTAPE VARIABLE IN COMMON PRPCOM FOR USE
C            BY OTHER PRPLOT ROUTINES.
C
C  REV 9/79 11/79 WK FOR *** H A R R I S ***   *** H A R R I S ***
```

```
C            NOTICE USAGE OF CHARACTER*1 CHARACTER DATA
C
C
C    ADDED PLOT3Z ENTRY TO PLOT SYMBOL VECTOR.  WK 1/80.
C
      IMPLICIT  CHARACTER*1(W), LOGICAL(K)
C
      COMMON / PRPCOM /  IJTAPE
      DIMENSION  NSCALE(5),ABNOS(26),X(1),Y(1)
      CHARACTER*1 NOS(10), VC, HC,NC,BL,HF,HF1,HF2,FOR1(21),
     $         FOR2(17),FOR3(20)
      CHARACTER*1 IMAGE(1), CH(1), LABEL(1)
      DATA NOS   /'0','1','2','3','4','5','6','7','8','9'/
      DATA         VC/'!'/     ,HC/'-'/,NC/'+'/,BL/' '/
     $           , HF/'F'/,HF1/'.'/,HF2/','/
      DATA FOR1/ '(1X,A1,F8. ,1X,121A1)'/
      DATA       FOR2  /'(1X,A1, 9X,121A1)'/
      DATA      FOR3 / '(1H0,F  . ,  F   . )'/
      DATA IITAPE / 6 /
      DATA KPLOT1 /. FALSE./, KPLOT2/.FALSE./   , KTAPEI/.FALSE./
      DATA  KABSC,KORD,KBOTGL /3*.FALSE./
C
      ENTRY PLOT1(NSCALE,NHL,NSBH,NVL,NSBV)
      KPLOT1=.TRUE.
      KPLOT2=.FALSE.
      NH=IABS(NHL)
      NSH=IABS(NSBH)
      NV=IABS(NVL)
      NSV=IABS(NSBV)
      NSCL=NSCALE(1)
  125 CONTINUE
      IF(.NOT.KTAPEI) IJTAPE=IITAPE
      KTAPEI=.TRUE.
      IF(NH*NSH*NV*NSV.NE.0) GO TO 128
      WRITE(IJTAPE,       14 )
14    FORMAT(T5,'SOME PLOT1 ARG. ILLEGALLY 0')
      KPLOT=.FALSE.
      RETURN
128   KPLOT=.TRUE.
      IF(NV.LE.25) GO TO 126
      WRITE(IJTAPE,          12 )
      KPLOT=.FALSE.
12    FORMAT(T5,'NO. OF VERTICAL LINES >25')
      RETURN
126   CONTINUE
      NVM=NV-1
      NVP=NV+1
      NDH=NH*NSH
      NDHP=NDH+1
      NDV=NV*NSV
      NDVP=NDV+1
      NIMG=(NDHP*NDVP)
      IF(NDV.LE.120) GO TO 130
      KPLOT=.FALSE.
      WRITE(IJTAPE,         11 )
11    FORMAT(T5,'WIDTH OF GRAPH >121')
      RETURN
130   CONTINUE
      IF(NSCL.EQ.0) GO TO 70
      FSY=10.**NSCALE(2)
      FSX=10.**NSCALE(4)
```

```
      IY=MINO(IABS(NSCALE(3)),7)+1
      IX=MINO(IABS(NSCALE(5)),9)+1
      GO TO 75
70    FSY=1.
      FSX=1.
      IY=4
      IX=4
75    FOR1(11)=NOS(IY)
      NA=MINO(IX,NSV)-1
      NS=NA-MINO(NA,120-NDV)
      NB=11-NS+NA
      I1=NB/10
      I2=NB-I1*10
      FOR3(7)=NOS(I1+1)
      FOR3(8)=NOS(I2+1)
      FOR3(10)=NOS(NA+1)
      IF(NV.GT.0) GO TO 90
      DO 80 J=11,18
80    FOR3(J)=BL
      GO TO 100
90    I1=NV/10
      I2=NV-I1*10
      FOR3(11)=HF2
      FOR3(12)=NOS(I1+1)
      FOR3(13)=NOS(I2+1)
      FOR3(14)=HF
      I1=NSV/100
      I3=NSV-I1*100
      I2=I3/10
      I3=I3-I2*10
      FOR3(15)=NOS(I1+1)
      FOR3(16)=NOS(I2+1)
      FOR3(17)=NOS(I3+1)
      FOR3(18)=HF1
      FOR3(19)=FOR3(10)
100   IF(KPLOT1) RETURN
      KPLOT1=.TRUE.
C
      ENTRY PLOT2(IMAGE,XMAX,XMIN,YMAX,YMIN)
      KPLOT2=.TRUE.
      IF(KPLOT1) GO TO 210
      NSCL=0
      NH=5
      NSH=10
      NV=10
      NSV=10
       GO TO 125
210   CONTINUE
      IF(.NOT.KPLOT)RETURN
      YMX=YMAX
      DH=(YMAX-YMIN)/FLOAT (NDH)
      DV=(XMAX-XMIN)/FLOAT(NDV)
      DO 220 I=1,NVP
220   ABNOS(I)=(XMIN+FLOAT((I-1)*NSV)*DV)*FSX
      DO 225 I=1,NIMG
225   IMAGE(I)=BL
      DO 240 I=1,NDHP
      I2=I*NDVP
      I1=I2-NDV
      KNHOR=MOD(I-1,NSH).NE.0
      IF(KNHOR) GO TO 230
```

```
      DO 228 J=I1,I2
228   IMAGE(J)=HC
230   CONTINUE
      DO 240 J=I1,I2,NSV
      IF(KNHOR) GO TO 235
      IMAGE(J)=NC
      GO TO 240
235   IMAGE(J)=VC
240   CONTINUE
      XMIN1=XMIN-DV/2.
      YMIN1=YMIN-DH/2.
      RETURN
C
      ENTRY PLOT3(CH,X,Y,N3)
      ICHINC=0
      GO TO 300
      ENTRY  PLOT3Z(CH, X, Y, N3)
      ICHINC=1
300   IF(KPLOT2) GO TO 312
      IF(.NOT.KTAPEI) IJTAPE=IITAPE
      KTAPEI=.TRUE.
  301 WRITE(IJTAPE,       13 )
13    FORMAT(T5,'PLOT2 MUST BE CALLED')
312   CONTINUE
      IF(.NOT.KPLOT) RETURN
      IF(N3.GT.0) GO TO 314
      KPLOT=.FALSE.
      WRITE(IJTAPE,       15 )
15    FORMAT(T5,'PLOT3 ARG2 < 0')
      RETURN
 314  CONTINUE
      ICH = 1 - ICHINC
      DO 320 I=1,N3
      ICH = ICH + ICHINC
      DUM1=(X(I)-XMIN1)/DV
      DUM2=(Y(I)-YMIN1)/DH
      IF(DUM1.LT.0..OR.DUM2.LT.0.) GO TO 320
      IF(DUM1.GE.NDVP.OR.DUM2.GE.NDHP) GO TO 320
      NX=1+INT(DUM1)
      NY=1+INT(DUM2)
315   J=(NDHP-NY)*NDVP+NX
      IMAGE(J)=CH(ICH)
320   CONTINUE
      RETURN
C
      ENTRY PLOT4(NL,LABEL)
      ENTRY FPLOT4(NL,LABEL)
      IF(.NOT.KPLOT) RETURN
      IF(.NOT.KPLOT2) GO TO 301
      DO 420 I=1,NDHP
      IF(I.EQ.NDHP.AND.KBOTGL) GO TO 420
      WL=BL
      IF(I.LE.NL) WL = LABEL(I)
      I2=I*NDVP
      I1=I2-NDV
      IF(MOD(I-1,NSH).EQ.0.AND..NOT.KORD) GO TO 410
      WRITE (IJTAPE,      FOR2) WL,(IMAGE(J),J=I1,I2)
      GO TO 420
410   CONTINUE
      ORDNO=(YMX-FLOAT(I-1)*DH)*FSY
      WRITE (IJTAPE,      FOR1) WL,ORDNO,(IMAGE(J),J=I1,I2)
```

```
420   CONTINUE
      IF(KABSC) GO TO 430
      WRITE (IJTAPE,        FOR3)  (ABNOS(J),J=1,NVP)
430   RETURN
C
      ENTRY OMIT(LSW)
      KABSC=MOD(LSW,2).EQ.1
      KORD=MOD(LSW,4).GE.2
      KBOTGL=LSW.GE.4
      RETURN
C
      ENTRY PLTAPE(ITAPE)
      IJTAPE = ITAPE
      KTAPEI = .TRUE.
      RETURN
      END
C=====================================================
C=====================================================  PLOT2P       ======
      SUBROUTINE PLOT2P(IMAGE,DATAMX,DATAMN)
C
C**  PLOT2P - SETS UP PROBABILITY-PLOT GRID FOR USE WITH PRPLOT.
C**           PROBS 99.9 TO 0.1 PCT.  DATA GRID 5 X 10 SPACES.
C             IMAGE AREA NEEDS 6171 CHAR, 1543 WORDS.
C**  PLOT4P - PRINTS PROBABILITY PLOT AND LABELS FOR BOTH AXES.
C**           PRINTS TOTAL OF 54 LINES, STARTING WITH TOP GRID LINE OF
C**           THE PLOT AND ENDING WITH PROB AXIS LABEL.  USER IS RESPON-
C**           SIBLE FOR TITLES AND PAGE EJECTS.
C                                                 WKIRBY 7/7/78.
C  REV 8/24/78 WK - GET LOGICAL UNIT FOR PRINT FROM PRPCOM.
C  REV 4/80 WK - CONTROL DEC PTS ON GRAPH  ORDINATE W/ NSCALE/LSCALE
C
C  *** HARRIS ***  (IBM USES LOGICAL NOT CHAR)
C
      CHARACTER*1 IMAGE(121,51)
      CHARACTER*1 DLABEL(1)
      LOGICAL LSCALE
      INTEGER NSCALE(5)
      INTEGER NVGRID, IVGRID(17)
      COMMON / PRPCOM/ IPLOT
      DATA NSCALE / 1, 0, 3, 0, 3 /
      DATA NVGRID/17/, IVGRID  /5, 11, 16, 21, 29, 36, 45, 51, 61,
     $        71, 77, 86, 93, 101, 106, 111, 117 /
C
      LSCALE(X) = ABS(DATAMX).GE.X .OR. ABS(DATAMN).GE.X
C
      NSCALE(3) = 3
      IF(LSCALE(   1000.)) NSCALE(3) = 2
      IF(LSCALE(  10000.)) NSCALE(3) = 1
      IF(LSCALE(100000.)) NSCALE(3) = 0
      CALL PLOT1(NSCALE,5,10,1,120)
      CALL PLOT2(IMAGE, 3.091, -3.091,  DATAMX, DATAMN)
      DO 10 IV=1,NVGRID
      DO 10 J = 1,51
   10 IMAGE(IVGRID(IV),J) = IMAGE(1, J)
      RETURN
C
      ENTRY PLOT4P(N,DLABEL)
      CALL OMIT(1)
      CALL PLOT4(N, DLABEL)
      CALL OMIT(0)
      WRITE(IPLOT,90)
```

```
 90 FORMAT(
   $      '0       99.9 99.8  99.5 99.0 98.0    95.0   90.0      80.0',
   $   '  70.0      50.0        30.0  20.0       10.0     5.0       2.0  ',
   $   '1.0  0.5   0.2 0.1'/
   $   35X,'EXCEEDANCE PROBABILITY, PERCENT   (NORMAL PROBABILITY ',
   $       'SCALE)'  )
      RETURN
      END
C===================================================
C===================================================  PRPSCL       ======
      SUBROUTINE PRPSCL (XTOP,XBOT,NGRID,GTOP,GBOT)
C  PRPSCL -- PRINTER-PLOTTER  SCALE SELECTION.
      LOGICAL NOSWIT
      DATA TOL /.005/
      DATAMSG/6/
C****************************************
C  STATEMENT FUNCTIONS
      STEP(X)  = .5 + SIGN(.5,X)
      ZINT(X)  = AINT(X)  - 1. + STEP(X)
      FLOOR(X)  = ZINT(X+TOL)
      CEIL(X)  = ZINT(X+1.-TOL)
C****************************************
      XMX=AMAX1(XTOP,XBOT)
      XMN=AMIN1(XTOP,XBOT)
      NOSWIT=XTOP.GT.XBOT
      FNGRID=NGRID
      UGLY=(XMX-XMN)/FNGRID
      U=0.
      M=0
      IF(UGLY.EQ.0.)GO TO 11
      U = ALOG10(ABS(UGLY))
      M=INT(U)
      IF(U.LT.0.)  M = M - 1
      U = 10**(U-M)
   11 CONTINUE
      IF(U.LT.1.+TOL)U=10.*U
      TENTOM=UGLY/U
      U=U/(1.+TOL)
      SCALE=5.
      IF(U.LT.2.)SCALE=2.
      IF(U.GT.5.)SCALE=10.
      SCALE=SCALE*TENTOM
   20 CONTINUE
      GMX=CEIL(XMX/SCALE)
      GMN=FLOOR(XMN/SCALE)
      GRANGE=GMX-GMN
      IF(GRANGE.LE.FNGRID)GOTO100
      IF(ABS((SCALE/TENTOM)-5.)  .LT.  1.)GOTO80
      GMX=CEIL(2.*XMX/SCALE)/2.
      GMN=FLOOR(2.*XMN/SCALE)/2.
      GRANGE=GMX-GMN
      IF(GRANGE.LE.FNGRID)GOTO100
   80 CONTINUE
      SCALE=2.*SCALE
      IF(SCALE.LT.(5.*TENTOM))SCALE=(5.*TENTOM)
      GMX=CEIL(XMX/SCALE)
      GMN=FLOOR(XMN/SCALE)
      GRANGE=GMX-GMN
      IF(GRANGE.LE.FNGRID)GOTO100
      WRITE(MSG,98)XTOP,XBOT,GMX,GMN,SCALE
   98 FORMAT('0***PRPSCL098 LOGIC ERROR',1P5E16.6/)
```

```
      GBOT=XBOT
      GTOP=XTOP
      RETURN
  100 CONTINUE
      EXCESS=FNGRID-GRANGE
      SMALL=AINT(EXCESS/2.)
      BIG=EXCESS-SMALL
      IF((GMX*SCALE-XMX) .GT. (XMN-GMN*SCALE))GOTO110
      GMX=GMX+BIG
      GMN=GMN-SMALL
      GOTO120
  110 GMX=GMX+SMALL
      GMN=GMN-BIG
  120 CONTINUE
      GTOP=GMX*SCALE
      GBOT=GMN*SCALE
      IF(NOSWIT)GOTO130
      TEMP=GBOT
      GBOT=GTOP
      GTOP=TEMP
  130 CONTINUE
      RETURN
      END
C==================================================
C==================================================  RANDUB       ======
      FUNCTION RANDUB(IRAN)
C
C  * HARRIS *  * HARRIS *  * HARRIS *
C
      INTEGER*6 IRAN, MODU, MULT
      DATA MODU, MULT / 2147483647, 16807/
      IRAN = IRAN*MULT
      IRAN = IRAN - (IRAN/MODU)*MODU
      RANDUB =  FLOAT(IRAN)/FLOAT(MODU)
      RETURN
      END
C==================================================
C==================================================  RANDUV       ======
      SUBROUTINE RANDUV(IRAN,X, N)
C
C  * HARRIS *  * HARRIS *  * HARRIS *
C
      INTEGER*6 IRAN, MODU, MULT
      DIMENSION X(1)
      DATA MODU, MULT / 2147483647, 16807/
      SCALE=1./FLOAT(MODU)
      DO 10 I = 1, N
      IRAN = IRAN*MULT
      IRAN = IRAN - (IRAN/MODU)*MODU
      X(I) = SCALE*FLOAT(IRAN)
   10 CONTINUE
      RETURN
      END
C==================================================
C==================================================  SMIRP       =======
      FUNCTION SMIRP(X)
C
C           COMPUTES VALUES OF THE LIMITING DISTRIBUTION FUNCTION FOR
C           THE KOLMOGOROV-SMIRNOV STATISTIC.
C
C           SMIRP(X) = PROB(KSSTAT .LE. X), WHERE
```

```
C            KSSTAT = SQRT(N)*(D SUB N)  (KOLMO)
C                   = SQRT(N*M/(N+M))*(D SUB M,N)   (SMIRNOV)
C            D  =  MAX OVER X OF ABS( FA(X) - FB(X) )
C            FA, FB = SAMPLE AND THEO PROB DTNS FOR KOLMO
C                   = INDEP SAMPLE PROB DTNS FOR SMIRNOV
C         REMARKS
C            THE METHOD IS DESCRIBED BY W. FELLER-ON THE KOLMOGOROV-
C            SMIRNOV LIMIT THEOREMS FOR EMPIRICAL DISTRIBUTIONS- ANNALS
C            OF MATH. STAT., 19, 1948, 177-189, BY N. SMIRNOV--TABLE
C            FOR ESTIMATING THE GOODNESS OF FIT OF EMPIRICAL
C            DISTRIBUTIONS- ANNALS OF MATH. STAT., 19, 1948, 279-281,
C            AND GIVEN IN LINDGREN, STATISTICAL THEORY, THE MACMILLAN
C            COMPANY, N. Y., 1962.
C
C            SMIRP IS SET TO 0.0 OR 1.0 FOR X OUTSIDE (0.27,3.10).
C            MAX ERRORS = 0.000029 AND 0.000019 AT X = 0.62 AND 1.87.
C            LINDGREN  REFERENCE NOTES  THAT FOR HIGH SIGNIFICANCE LEVELS
C            (SAY, .01 AND .05) ASYMPTOTIC FORMULAS GIVE VALUES WHICH ARE
C            TOO HIGH ( BY 1.5 PER CENT WHEN N = 80), CAUSING THE
C            NO-DIFFERRENCE HYPOTHESIS TO BE REJECTED TOO SELDOM.
C
C         SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED  - NONE
C
C   REVISIONS - FROM IBM SCI SUB PKG 2/79 WK.
C
C       ...............................................................
C
      IF(X-.27)1,1,2
    1 Y=0.0
      GO TO 9
    2 IF(X-1.0)3,6,6
    3 Q1=EXP(-1.233701/X**2)
      Q2=Q1*Q1
      Q4=Q2*Q2
      Q8=Q4*Q4
      IF(Q8-1.0E-25)4,5,5
    4 Q8=0.0
    5 Y=(2.506628/X)*Q1*(1.0+Q8*(1.0+Q8*Q8))
      GO TO 9
    6 IF(X-3.1)8,7,7
    7 Y=1.0
      GO TO 9
    8 Q1=EXP(-2.0*X*X)
      Q2=Q1*Q1
      Q4=Q2*Q2
      Q8=Q4*Q4
      Y=1.0-2.0*(Q1-Q4+Q8*(Q1-Q8))
    9 SMIRP = Y
      RETURN
      END
C==================================================
C==================================================  SNCTPA       ======
      FUNCTION SNCTPA( X, NDF, DELT)
C   STUDENT NONCENTRAL T PROBABILITY APROXIMATION.
C   NORMAL APPROXIMATION FOR LARGE NDF.  NBS HANDBK MATH FUN 26.7.
      DF = NDF
      SNCTPA = GAUSCF((X*(1.-0.25/DF) - DELT)/SQRT(1.+0.5*(X**2)/DF))
      RETURN
      END
C==================================================
C==================================================  SNCTXA       ======
```

```
      FUNCTION SNCTXA( P, NDF, DELT)
C  STUDENT NONCENTRAL T  QUANTILE   APROXIMATION.
C  NORMAL APPROXIMATION FOR LARGE NDF.  NBS HANDBK MATH FUN 26.7.
C  SOLUTION FOR X SUCH THAT STNCPA(NDF,DELT, X) .EQ. P.
      DF = NDF
      Z = GAUSAB(P)
      W = 1. - 0.5*(Z**2)/DF
      SNCTXA =  (DELT + Z*SQRT(W+0.5*(DELT**2)/DF))/W
      RETURN
      END
C=====================================================
C=====================================================  SORT      ========
      SUBROUTINE  SORT(A, N)
C      TO MODIFY SEE CDS 35 AND 40.     USE -N TO SORT DOWN.  WK 5/78.
C
C       THIS IS A VERY FAST SUBROUTINE FOR SORTING A REAL ARRAY IN
C       ASCENDING ORDER. THIS PROGRAM IS A TRANSLATION INTO
C       FORTRAN IV OF ALGORITHM 271, QUICKERSORT, BY R.S. SCOWEN IN
C       COLLECTED ALGORITHMS OF THE ACM. CONSULT THIS FOR FURTHER
C       REFERENCES.      REVISED 5/78 TO SORT DOWN IF N .LT. 0.
C
C       PARAMETERS:
C            A            REAL*4 ARRAY           ON INPUT THIS IS THE
C                                                ARRAY TO BE SORTED.
C                                                ON OUTPUT THIS IS THE
C                                                SORTED ARRAY.
C
C            N            INTEGER*4              THE  DIMENSION OF THE
C                                                ARRAY A.
C                                                CALL WITH -N TO SORT DOWN
C
C       THE METHOD IS USED IS THAT OF CONTINUALLY SPLITTING THE ARRAY
C       INTO PARTS SUCH THAT ALL ELEMENTS OF ONE PART ARE LESS THAN
C       ALL ELEMENTS OF THE OTHER, WITH A THIRD PART IN THE MIDDLE
C       CONSISTING OF ONE ELEMENT.
C           AN ELEMENT WITH VALUE T IS CHOSEN ARBITRARILY (HERE WE
C       CHOOSE THE  MIDDLE ELEMENT). I AND J GIVE THE LOWER AND
C       UPPER LIMITS OF THE SEGMENT BEING SPLIT. AFTER THE SPLIT A
C       VALUE Q WILL HAVE BEEN FOUND SUCH THAT A(Q)=T AND
C       A(L)<=T<=A(M)  FOR ALL I<=L<Q<M<=J. THE PROGRAM THEN PERFORMS
C       OPERATIONS ON THE TWO SEGMENTS (I,Q-1) AND (Q+1,J) AS FOLLOWS.
C       THE SMALLER SEGMENT IS SPLIT AND THE POSITION OF THE LARGER
C       SEGMENT IS STORED IN THE LT AND UT ARRAYS. IF THE SEGMENT TO
C       BE SPLIT CONTAINS TWO OR FEWER ELEMENTS, IT IS SORTED AND
C       ANOTHER SEGMENT IS OBTAINED FROM THE LT AND UT ARRAYS. WHEN
C       NO MORE SEGMENTS REMAIN, THE ARRAY IS COMPLETELY SORTED.
C
C       TO SORT THE ARRAY IN DESCENDING ORDER CHANGE 'LE' TO 'GE'
C       IN LINES 66 AND 109, AND 'LT' TO 'GT' IN LINE 71
C
C
C
      REAL A(1), T, X
      INTEGER LT(17),UT(17),I,J,K,M,N,P,Q
C       THE DIMENSIONS FOR LT AND UT HAVE TO BE AT LEAST LOG  N.
C                                                           2
C       17 WAS CHOSEN TO HANDLE N<131,073.
C
      M=1
      I=1
      J=IABS(N)
```

```
      IF(J.NE.N)GO TO 200
C
C       IF THIS SEGMENT HAS MORE THAN TWO ELEMENTS  WE SPLIT IT
 10   IF (J-I-1) 100,90,15
C
C       P IS THE POSITION OF AN ARBITRARY ELEMENT IN THE SEGMENT
C       WE CHOOSE THE MIDDLE ELEMENT. UNDER CERTAIN CIRCUMSTANCES
C       IT MAY BE ADVANTAGEOUS TO CHOOSE P AT RANDOM.
 15   P=(J+I)/2
      T=A(P)
      A(P)=A(I)
C
C       STARTING AT THE BEGINNING OF THE SEGMENT, SEARCH FOR K
C       SUCH THAT A(K)>T
      Q=J
      K=I
 20   K=K+1
      IF (K.GT.Q) GO TO 60
       IF (A(K).LE.T) GO TO 20
C
C         SUCH AN ELEMENT HAS NOW BEEN FOUND
C         NOW SEARCH FOR A Q SUCH THAT A(Q)<T STARTING AT THE END OF THE
C         SEGMENT.
 30    IF (A(Q).LT.T) GO TO 40
       Q=Q-1
       IF (Q.GT.K) GO TO 30
       GO TO 50
C
C         A(Q) HAS NOW BEEN FOUND. WE INTERCHANGE A(Q) AND A(K)
 40    X=A(K)
       A(K)=A(Q)
       A(Q)=X
C
C         UPDATE Q AND SEARCH FOR ANOTHER PAIR TO INTERCHANGE
       Q=Q-1
       GO TO 20
 50    Q=K-1
 60   CONTINUE
C
C         THE UPWARDS SEARCH HAS NOW MET THE DOWNWARDS SEARCH
      A(I)=A(Q)
      A(Q)=T
C
C       THE SEGMENT IS NOW DIVIDED IN THREE PARTS: (I,Q-1),(Q),(Q+1,J)
C       STORE THE POSITION OF THE LARGEST SEGMENT IN LT AND UT
      IF (2*Q.LE.I+J) GO TO 70
        LT(M)=I
        UT(M)=Q-1
        I=Q+1
      GO TO 80
 70     LT(M)=Q+1
        UT(M)=J
        J=Q-1
C
C       UPDATE M AND SPLIT THE NEW SMALLER SEGMENT
 80   M=M+1
      GO TO 10
C
C       WE ARRIVE HERE IF THE SEGMENT HAS  TWO ELEMENTS
C       WE TEST TO SEE IF THE SEGMENT IS PROPERLY ORDERED
C       IF NOT, WE PERFORM AN INTERCHANGE
```

```
 90    IF (A(I).LE.A(J)) GO TO 100
       X=A(I)
       A(I)=A(J)
       A(J)=X
C
C        IF LT AND UT CONTAIN MORE SEGMENTS TO BE SORTED REPEAT PROCESS
 100   M=M-1
       IF (M.LE.O) RETURN
       I=LT(M)
       J=UT(M)
       GO TO 10
C
C
  200 CONTINUE
C
C  SORT IN DECREASING ORDER WHEN N IS .LT. O
C  DUPLICATE COPY OF CODE WITH CHANGES MADE AS NOTED IN CARD 35.
 210    IF (J-I-1)2100,290,215
 215    P=(J+I)/2
        T=A(P)
        A(P)=A(I)
        Q=J
        K=I
 220    K=K+1
        IF (K.GT.Q) GO TO 260
         IF (A(K).GE.T) GO TO 220
 230     IF (A(Q).GT.T) GO TO 240
         Q=Q-1
         IF (Q.GT.K) GO TO 230
         GO TO 250
 240     X=A(K)
         A(K)=A(Q)
         A(Q)=X
         Q=Q-1
         GO TO 220
 250     Q=K-1
 260    CONTINUE
        A(I)=A(Q)
        A(Q)=T
        IF (2*Q.LE.I+J) GO TO 270
          LT(M)=I
          UT(M)=Q-1
          I=Q+1
        GO TO 280
 270      LT(M)=Q+1
          UT(M)=J
          J=Q-1
 280    M=M+1
        GO TO 210
 290    IF (A(I).GE.A(J)) GO TO 2100
        X=A(I)
        A(I)=A(J)
        A(J)=X
 2100   M=M-1
        IF (M.LE.O) RETURN
        I=LT(M)
        J=UT(M)
        GO TO 210
        END
C=====================================================
C=====================================================  STAT3    ========
```

```
      SUBROUTINE STAT3(X,N,A,S,G)
C  COMPUTES MEAN, STD DEVIATION, AND SKEW COEFF OF DATA X.
C  UNBIASED ESTIMATES ((N-1) AND N/(N-1)(N-2)) ARE USED FOR CENT MOMENTS
C  STATISTICS ARE UNDEFINED IF N IS TOO SMALL -- USER MUST CHECK THIS.
C  NO WARNING IS GIVEN.  UNDEFINED STATS  SIMPLY ARE SET TO ZERO.
C  FOR EXAMPLE, IF N=2, A AND S ARE COMPUTED AND G = 0.0.
C
      DIMENSION X(1)
      DOUBLE PRECISION DA,DS, DG, T
C
      A = 0.
      S = 0.
      G = 0.
      IF(N.LE.0)RETURN
      DA=0D0
      DS=0D0
      DG=0D0
      DO 10 I=1,N
      T=X(I)
      DA=DA+T
      DS=DS+T**2
   10 DG=DG+T**3
      T=N
      FN1=N-1
      FN2=N-2
      DA=DA/T
      A=DA
      IF(N.LE.1)RETURN
      S=SQRT(SNGL(DS-T*DA**2)/FN1)
      IF(N.LE.2 .OR. S.LE.0.)RETURN
      G = (DG-3D0*DA*DS+2D0*T*DA**3)*T/((S**3)*FN1*FN2)
      RETURN
      END
C==================================================
C==================================================  STUTP      =======
      FUNCTION STUTP(X,N)
C
C  STUDENT T PROBABILITY
C  STUTP = PROB( STUDENT T WITH N DEG FR  .LT.  X )
C
C  NOTE  -  PROB(ABS(T).GT.X) = 2.*STUTP(-X,N)  (FOR X .GT. 0.)
C
C  SUBPGM USED - GAUSCF
C
C  REF - G.W. HILL, ACM ALGOR 395, OCTOBER 1970.
C
C    USGS - WK 12/79.
C
C
      DATA RHPI / 0.63661977 /
C
      STUTP  =  .5
      IF(N.LT.1) RETURN
C
      NN = N
      Z = 1.
      T = X**2
      Y = T/NN
      B = 1.0 + Y
C
      IF(NN.GE.20 .AND. T.LT.NN .OR. NN.GT.200) GO TO 200
```

```
C               ( OR IF NN NON-INTEGER)
C
      IF(NN.LT.20 .AND. T.LT.4.) GO TO 100
C
C  --  TAIL SERIES FOR LARGE T
      A = SQRT(B)
      Y = A*NN
      J = 0
   30 J = J + 2
      IF(A.EQ.Z) GO TO 40
      Z = A
      Y = Y*(J-1)/(B*J)
      A = A + Y/(NN+J)
      GO TO 30
   40 CONTINUE
      NN = NN + 2
      Z = 0.
      Y = 0.
      A = -A
      GO TO 110
C
C  --  NESTED SUMMATION OF COSINE SERIES
  100  Y = SQRT(Y)
      A = Y
      IF(NN.EQ. 1)  A = 0.
  110 NN = NN - 2
      IF(NN.LE.1) GO TO 120
      A = (NN-1)/(B*NN)*A + Y
      GO TO 110
  120 IF(NN.EQ.0)  A = A/SQRT(B)
      IF(NN.NE.0)  A = (ATAN(Y)+A/B)*RHPI
      STUTP  = 0.5*(Z-A)
      IF(X.GT.0.) STUTP = 1.-STUTP
      RETURN
C
C  --  ASYMPTOTIC SERIES FOR LARGE OR NONINTEGER N
  200 IF(Y.GT.1E-6) Y = ALOG(B)
      A = NN - 0.5
      B = 48.*A**2
      Y = A*Y
      Y = (((((-0.4*Y-3.3)*Y-24.)*Y-85.5)/
     $       (0.8*Y**2+100.+B)+Y+3.)/B+1.)*SQRT(Y)
      STUTP = GAUSCF(-Y)
      IF(X.GT.0.) STUTP = 1.-STUTP
      RETURN
C
      END
C================================================
C================================================  STUTX       =======
      FUNCTION STUTX(P,N)
C
C  STUDENT T QUANTILES --
C   STUTX(P,N) = X SUCH THAT PROB(STUDENT T WITH N D.F. .LE. X) = P.
C
C  NOTE - ABS(T) HAS PROB Q OF EXCEEDING STUTX( 1.-Q/2., N ).
C
C   NOTE -      IER - ERROR FLAG --  1 = F.LT.1.,
C                                    2 = P NOT  IN (0,1),  3 = 1+2
C   SUBPGMS USED -- GAUSAB     (GAUSSIAN ABSCISSA)
C
C  REF - G. W. HILL (1970) ACM ALGO 396.  COMM ACM 13(10)619-20.
```

```
C             REV BY WKIRBY 10/76. 2/79.  10/79.
C
      DATA HPI / 1.5707963268/
C
      SIGN=1.
      IF(P.LT.0.5)SIGN=-1.
      Q=2.*P
      IF(Q.GT.1.)Q=2.*(1.-P)
      IF(Q.LT.1.)GO TO 1010
      STUTX = 0.
      RETURN
C
1010  CONTINUE
      FN = N
      IF(N.GE.1 .AND. Q.GT.0. .AND. Q.LT.1.) GO TO 10
      IER = 3
      IF(N.GE.1) IER = 2
      STUTX = SIGN*1E38
      RETURN
C
   10 IF(N.NE.1) GO TO 20
C  -- 1 DEG FR - EXACT
      STUTX=SIGN/TAN(HPI*Q)
      RETURN
C
   20 IF(N.NE.2) GO TO 30
C  -- 2 DEG FR - EXACT
      STUTX = SQRT(2.0/(Q*(2.0-Q))-2.0) * SIGN
      RETURN
C
C  -- EXPANSION FOR N .GT. 2
   30 A = 1.0/(FN-0.5)
      B= 48.0/(A*A)
      C=((20700.*A/B-98.)*A-16.)*A+96.36
      D =  ((94.5/(B+C)-3.)/B+1.)*SQRT(A*HPI)*FN
      X = D*Q
      Y = X**(2.0/FN)
      IF (Y .GT. A+.05) GO TO 40
      Y = ((1.0/(((FN+6.)/(FN*Y)-0.089*D-0.822)*(FN+2.)*3.)+
     $   0.5/(FN+4.))*Y-1.)*(FN+1.)/(FN+2.)+1.0/Y
      STUTX = SQRT(FN*Y) * SIGN
      RETURN
C
C   -- ASYMPTOTIC INVERSE EXPANSION ABOUT NORMAL
   40 X = GAUSAB(0.5*Q)
      Y =  X*X
      IF (FN .LT. 5.) C = C+0.3*(FN-4.5)*(X+0.6)
      C = (((.05*D*X-5.)*X-7.)*X-2.)*X+B+C
      Y = (((((0.4*Y+6.3)* Y+36.)*Y+94.5)/C-Y-3.)/B+1.)*X
      X = A*Y**2
      Y = X + 0.5*X**2
      IF (X .GT. .002) Y = EXP(X) - 1.0
      STUTX = SQRT(FN*Y) * SIGN
      RETURN
      END
C=================================================
C=================================================  WCFGSM      ======
      FUNCTION WCFGSM ( FLAT, FLONG)
C
C  WRC BULL. 17-A GENERALIZED SKEW MAP.C
C  FOR LATITUDE AND LONGITUDE IN DEGREES AND FRACTIONS OF A DEGREE.
```

```
C  (E.G., 45 DEG, 30 MIN  =  45.5 DEG.)
C  POINTS OUTSIDE U.S. ARE ASSIGNED LARGE NEGATIVE VALUE.
C
C  DATA STATEMENT USES 2 CARDS PER MERIDIAN OF LONGITUDE, STARTING AT
C  67 DEG AND ENDING AT 125 DEG WEST LONGITUDE.  SKEW VALUES ARE  READ
C  FROM SOUTH (25 DEG) TO NORTH (50 DEG)  ALONG EACH MERIDIAN.
C  SKEW VALUES TABULATED ARE IN UNITS OF 0.01 SKEW UNIT.  POINTS
C  OUTSIDE CONTINENTAL U.S. ARE SET TO -222.
C  ALASKA AND HAWAII ARE NOT TABULATED BY LAT-LONG BUT ARE HANDLED
C  CORRECTLY BY PROGRAMMED TESTS.
C
C  REV 8/6/79 WK TO IMPROVE THE REPRESENTATION OF THE GEN SKEW RIDGE
C          AT GEN SKEW VALUE OF 0.2 RUNNING THROUGH EAST CALIFORNIA
C          AND WEST OREGON AND WASHINGTON.
C
C  REV 8/9/79 WK REVISED VARIOUS MAP SKEWS.
C
      LOGICAL LLCELL
      INTEGER  ISK(26,59)
      INTEGER  ISK1(26,9),ISK2(26,9),ISK3(26,9),ISK4(26,9),
     $           ISK5(26,9),ISK6(26,9),ISK7(26,5)
      EQUIVALENCE (ISK1(1,1),ISK(1, 1)), (ISK2(1,1),ISK(1,10)),
     $            (ISK3(1,1),ISK(1,19)), (ISK4(1,1),ISK(1,28)),
     $            (ISK5(1,1),ISK(1,37)), (ISK6(1,1),ISK(1,46)),
     $            (ISK7(1,1),ISK(1,55))
      DATA ISK1 /
     $ -222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,
     $ -222,-222,-222,-222,-222,-222,  55,  38,  12,  -8, -15,-222,-222,
     $ -222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,

       . . .   TABULAR  DATA  OMITTED   . . .

     $ -222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,
     $  -30, -30, -30, -30, -30, -30, -18,  -5,  15,  20,  20,   5,   0,
     $ -222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,-222,
     $ -222,-222, -30, -30, -30, -30, -30, -30, -30, -20,   0,   0,  20/
C
      LLCELL(FLATMN,FLATMX,FLONMN,FLONMX) = FLONG.GE.FLONMN
     $  .AND. FLONG.LT.FLONMX .AND. FLAT.GE.FLATMN .AND. FLAT.LT.FLATMX
C
      WCFGSM = -1E37
      IF(LLCELL(25., 50.01,  67.,  125.01 ) ) GO TO 100
C
C  CHECK ALASKA, HAWAII
C
      IF(LLCELL(54., 61.,129., 140.) ) WCFGSM = 0.33
      IF(LLCELL(58., 72.,140., 170.) ) WCFGSM = 0.70
      IF(LLCELL(50., 58.,150., 360.) ) WCFGSM = 0.70
      IF(LLCELL(18., 30.,154., 162.) ) WCFGSM = -0.05
      RETURN
C
C  CONTINENTAL U.S. SKEW MAP
  100 CONTINUE
      LAT = FLAT
      LONG = FLONG
      WY = FLAT - LAT
      WX = FLONG - LONG
      GSK = 0.
      DO 150 I =1, 2
      WY = 1. - WY
      DO 150 J = 1, 2
```

```
      WX = 1. - WX
      S = ISK( LAT-24+I-1, LONG-66+J-1 )
      IF(S.GT.-220.) GO TO 150
      IF(WX.LT.0.05) WX = 0.
      IF( WY.LT.0.05) WY = 0.
      S = -1E37
  150 GSK = GSK + 0.01*S*WX*WY
      IF(GSK.GT.-1E30) WCFGSM = GSK
      RETURN
      END
C==================================================
C==================================================  WEIBUL      ======
      FUNCTION WEIBUL (GAM,PROB)
C
C  STANDARDIZED WEIBULL QUANTILES -- POS OR NEG SKEW OR 0.0 SKEW (2/78)
C   PROB = CDF ORDINATE = CUMULATIVE PROB OF NONEXCEEDANCE
C      GAM = DESIRED SKEW   IN RANGE 0.01-100.    ** 0.0-100. (2/78)
C   MAIN FEATURE OF THIS IS SOLUTION FOR WEIBULL PARM C IN TERMS
C         OF SPECIFIED SKEW COEFF
C            W KIRBY  30 MARCH 72      REVISED 2/78 WK. CORRECTED 2/79.
C
      COMMON / IERSWT/ IERRR
      DOUBLE PRECISION  EW(3), CC
      LOGICAL  LEXPO
      DATA OLDSKU,TOL/-1E34,0.003/
C
      PRB1 = PROB
      IF(GAM.LT.0.) PRB1 = 1. - PRB1
      IF(PRB1*(1.-PRB1).LE.0.) IERRR = 1
      IF(ABS(GAM-OLDSKU) .GT.TOL)GOTO200
C
  100 IF(LEXPO)GOTO110
      WEIBUL =   ((-ALOG(1.-PRB1))**RC - AV) / SIG
      RETURN
  110 IF(PRB1.LE.0.)PRB1=2.
      WEIBUL = (-ALOG(1.-PRB1) - 1.) / SIG
      RETURN
C
  200 OLDSKU=GAM
      AG=ABS(GAM)
      LEXPO = ABS(AG - 2.) .LT. TOL
      IF(LEXPO) SIG = SIGN(1.,GAM)
      IF(LEXPO)GOTO110
      IF(AG.GT.100.)IERRR = 2
      IF(AG.GT.100.)AG=100.
      CUP=3.8
      CLO=0.2
  220 CC=(.5 *  (CLO+CUP))
      DO10K=1,3
   10 EW(K)=DGAMMA(1.+K/CC)
      AV=EW(1)
      SIG=DSQRT(EW(2)-EW(1)**2)
      SKU=(EW(3)-3.*EW(1)*EW(2)+2.*EW(1)**3)/SIG**3
      IF(SKU.LE.AG+TOL)GOTO260
      CLO=CC
      GOTO220
  260 IF(SKU.GE.AG-TOL)GOTO280
      CUP=CC
      GOTO220
  280 IF(GAM.LT.0.) SIG=- SIG
      RC= 1./CC
```

```
      GOTO100
      END
C==================================================
C==================================================  WILFRT+V       ====
C          WILFRT -- WILSON-HILFERTY REVISED TRANSFORM
C
C  PURPOSE -- APPROXIMATE TRANSFORMATION OF GAUSSIAN PERCENTAGE POINTS
C        INTO STANDARDIZED PEARSON TYPE III.   THIS VERSION REPRODUCES
C        CORRECT MEAN, VARIANCE, SKEW AND LOWER BOUND OF STANDARDIZED
C        PEARSON-III AT SKEWS UP TO 9.0 AT LEAST.  DIFFERENCES BETWEEN
C        WILFRT PERCENTAGE POINTS AND HARTERS TABLES ARE OF THE ORDER OF
C        A FEW HUNDREDTHS OF A STD. DEVIATION, EXCEPT IN EXTREME POSITIV
C        TAIL (95% OR SO) WHERE ERROR IS OF ORDER OF TENTHS IN MAGNI-
C        TUDE BUT ABOUT 3% IN RELATIVE MAG.
C
C  USAGE --      X=WILFRT(SKEW,ZETA)*STDDEV+AMEAN
C        SKEW IS INPUT SKEW, MAY BE ZERO OR NEGATIVE OR POSITIVE.
C              IF ABS(SKEW) IS GREATER THAN 9.75, 9.75 IS USED.
C        ZETA IS STANDARD GAUSSIAN VARIATE.   FOR EXAMPLE, GAUSSB(IRAN)
C              YIELDS RANDOM NOS WHILE GAUSAB(PROB) YIELDS THE
C              PROB-TH QUANTILE.
C        STDDEV AND AMEAN ARE DESIRED VALUES OF STD DEVIATION AND
C              MEAN, IF DIFFERENT FROM ONE AND ZERO.
C
C  NOTE -- EACH INPUT SKEW VALUE IS COMPARED WITH PREVIOUS INPUT
C        VALUE. IF DIFFERENT BY MORE THAN 0.0003, TABLE LOOKUP OF NEW
C        PARAMETERS TAKES PLACE.  THEREFORE, CHAGE THE INPUT SKEW
C        AS SELDOM AS POSSIBLE.
C
C      WKIRBY  72-02-25
C    REVISED 73-02-09   TO ACCEPT ZERO SKEW.
C
C  REF -- W.KIRBY, COMPUTER-ORIENTED WILSON-HILFERTY TRANSFORMATION...,
C         WATER RESOUR RESCH 8(5)1251-4, OCT 72.
C
      FUNCTIONWILFRT(SKU,ZETA)
      REALSKUTOL,      OLDSKU,        FMU,SIG,H,A,B
      DATA SKUTOL, OLDSKU / .0003, -1E38 /
      IF(ABS(OLDSKU-SKU).LT.SKUTOL)GOTOLABEL,(10,30)
      OLDSKU=SKU
      ASK=ABS(SKU)
      IF(ASK.LT.SKUTOL)GOTO20
C
C  NONZERO SKEW
      ASSIGN 10 TO LABEL
      CALLE443M4(ASK,G,H,A,B)
      SIG=G*.1666667
      FMU=1.-SIG*SIG
      IF(ASK.EQ.SKU)GOTO10
      SIG=-SIG
      A=-A
   10 Z=FMU+SIG*ZETA
      IF(Z.LT.H)Z=H
      WILFRT=A*(Z*Z*Z-B)
      RETURN
C
C  ZERO SKEW
   20 ASSIGN30TOLABEL
   30 WILFRT=ZETA
      RETURN
      END
```

```
      SUBROUTINEE443M4(SK,G,H,A,B)
C
C  COMPUTES PARAMETERS USED BY WILFRT TRANSFORMATIN
C   USES APPROX FORMULA AND CORRECTION TERMS PREPARED FROM
C   ROUTINE WHMPP (E443-5).  WKIRBY  FEB72
C    PARAMETERS RETURNED TO WILFRT ARE INTENDED TO MAKE
C     WILFRT A STANDARDIZED R.V.  (MEAN=0,STDEV=1) WITH
C           SPECIFIED SKEW AND CORRECT LOWER BOUND
C  REVISED CALC OF CORRECTION TABLE  72-03-03 WK
C
      DIMENSIONROW(4),SA(40),HALF1(80),HALF2(80),TABLE(40,4)
      EQUIVALENCE(ROW,S),(TABLE,SA,HALF1),(TABLE(1,3),HALF2)
      DATA NROZ / 40/
      DATAHALF1/
     # 0.0      , 0.250000, 0.500000, 0.750000, 1.000000, 1.250000,
     #  1.500000, 1.750000, 2.000000, 2.250000, 2.500000, 2.750000,

        . . .   TABULAR  DATA  OMITTED   . . .

     # 0.100831, 0.111114, 0.121283, 0.131245, 0.140853, 0.150120,
     # 0.158901, 0.167085, 0.174721, 0.181994/
      S=SK
      K=1
      DO70I=2,NROZ
      IF(SA(I).GE.S)GOTO180
   70 K=I
   71 FORMAT('0*WILFRT071 EXCESSIVE SKEW TRUNCATED',2E12.3/)
      WRITE(6,71)S,SA(NROZ)
      DO80I=1,4
   80 ROW(I)=TABLE(NROZ,I)
      GOTO500
  180 P=(S-SA(K))/(SA(K+1)-SA(K))
      Q=1.-P
      DO190I=2,4
  190 ROW(I)=Q*TABLE(K,I)+P*TABLE(K+1,I)
  500 G=S+ROW(2)
      IF(S.GT.1.)G=G-.063*(S-1.)**1.85
      TOG=2./S
      Q=TOG
      IF(Q.LT..4)Q=.4
      A=Q+ROW(3)
      Q=.12*(S-2.25)
      IF(Q.LT.0.)Q=0.
      B=1.+Q*Q+ROW(4)
      H=(B-TOG/A)**.3333333
      RETURN
      END
      SUBROUTINEWILFRV(SKU,ZETA,NZ)
C    VECTORIAL VERSION OF WILFRT          WKIRBY JUN74
C     ZETA HAS GAUSSIAN ON INPUT AND STDIZED WILFRT ON OUTPUT
      DIMENSIONZETA(NZ)
      REALSKUTOL,       OLDSKU,       FMU,SIG,H,A,B
      DATA SKUTOL, OLDSKU / .0003, -1E38 /
      IF(ABS(OLDSKU-SKU).LT.SKUTOL)GOTOLABEL,(10,30)
      OLDSKU=SKU
      ASK=ABS(SKU)
      IF(ASK.LT.SKUTOL)GOTO20
C
C  NONZERO SKEW
      ASSIGN 10 TO LABEL
      CALLE443M4(ASK,G,H,A,B)
```

```
      SIG=G*.1666667
      FMU=1.-SIG*SIG
      IF(ASK.EQ.SKU)GOTO10
      SIG=-SIG
      A=-A
   10 CONTINUE
      DO15I=1,NZ
      ZETA(I)=A*(AMAX1(H,FMU+SIG*ZETA(I))**3-B)
   15 CONTINUE
      RETURN
C
C  ZERO SKEW
   20 ASSIGN30TOLABEL
30    RETURN
      END
C==================   (LAST CARD)   ==============
```