GENERATION OF RANDOM VARIATES FROM STATISTICAL DISTRIBUTIONS

A Thesis

Presented to

the Faculty of the Department of

Industrial Engineering

University of Houston

In partial fulfillment

of the requirements for the Degree

Master of Science

---

by

Winifred Moon-Yee Ku

May, 1976

Dedicated to my beloved

Parents and Fourth Aunt

GENERATION OF RANDOM VARIATE
FROM STATISTICAL DISTRIBUTIONS

A Thesis
Presented to
the Faculty of the Department of
Industrial Engineering
Univeristy of Houston

In partial fulfillment
of the requirements for the Degree
Master of Science

by
Winifred Moon-Yee Ku
May, 1976

ABSTRACT

Work has been done in generating random variates of some distributions for simulation purposes. A subroutine which gives a random number of any common statistical distribution desired, and a compilation of the backgrounds and derivations for the generating methods gives convenience and understanding to anyone who does simulation work. This thesis presents a brief history on the generation of random variables, a compilation and investigation of old and new generating methods, and comparisons and evaluations of different methods by chi-square and t-tests. The best method of each distribution investigated is selected to incorporate into subroutines CTIME or DTIME, depending on whether the distribution is continuous or discrete.

TABLE OF CONTENTS

PAGE

CHAPTER I

1. INTRODUCTION

There are, generally speaking, two different ways to generate uniform random numbers. One is the physical method, such as drawing numbered chips from a container. Another is the mathematical method. Since physical methods are inapplicable for simulation purposes, we shall not attempt to discuss them here. We shall discuss the development of the mathematical methods on uniform and non-uniform number generators alike.

The most significant study on number generators was done around 1950. The questions of concern at that time were: 1) how to produce a sequence of random decimal digits, and 2) how to produce random real numbers according to an assigned probability distribution law. Ulam suggested to use the mapping function $f(x)=4x(1-x)$ to solve the first problem. But Von Neumann, a mathematician (41) pointed out that the independence of the random numbers was lacking by the use of mapping functions. He also pointed out that even with a theoretically sound function, such as $f(x)=\mathrm{Sin}^2\pi x$ which satisfied all the properties for a uniform number generator, the limitation of the machines in rounding-off would destroy the randomness property. He evaluated other methods, but did not present an adequate answer. As to the second problem, he suggested the rejection method which has one advantage in that the cumulative functions of the distributions need not to be used. This method is used specifically for distributions whose cumulative density functions do not exist in closed forms. In comparison with other methods, rejection takes more computing time.

Lehmer (19) was the first to study the congruential method for a uniform number generator. He used the relation

$$x_{i+1} = kx_i \pmod{M}$$

with k=23 and $M=10^8+1$. The sequence produced 8-decimal digit numbers with period 5882352. He also proposed that when δ is the smallest positive integer satisfying

$$\alpha^\delta = 1 \pmod{n}$$

where n and α are relatively prime positive integers, the sequence $u_i (i=1,2,...)$ formed by taking the principal remainder has a period of δ. The numbers $u_i n^{-1}$ may be used as uniform variates in the range (0,1) provided that δ is reasonably large.

In 1954, the University of Florida conducted a symposium on Monte Carlo Methods. In the symposium, Metropolis presented his mid-square method for a uniform number generator. O. Taussky and J. Todd tested both the additive congruential and the mid-square methods. E.J. Lytle Jr. presented tests for normal random variates, but he did not give any generating method other than look-ups from normal tables and random number tables. Based on Votaw and Rufferty (42), James W. Butler gave the direct method and composition methods for transforming uniform random variates into variates of the desired distribution.

Lehmer's method was further carefully studied by a few people, among whom were Eva and V.J. Bofinger. They gave a general way of evaluating δ and also the results of 4 tests for randomness: frequence test, serial test, gap test and uniform test. J. Certaine concerned himself in finding the value of α which will generate a non-repeated sequence of maximum length.

Among all the non-uniform distributions, the normal distribution was studied first and most intensively. There were many generating methods suggested. Muller designed his Chebyshev approximation (29), but it required the storage of large tables of values which contained the coefficients of the polynomials of the approximated functions. His method is rather cumbersome. He also had a study on the direct method. Von Neumann had his rejection method applied to the normal distribution. Hasting had his approximation, and there was also the central limit approximation approach. Teichroew had his Chebyshev polynomial approximation which did not require the storage of any table, therefore it worked better than that of Muller's.

G. Marsaglia, M.D. MacLaren and T.A. Bray (21) designed their method by breaking up the normal density function into a sum of three density functions. L.E. Cannon did a detail study on this in his Masters thesis. He included normal and exponential distributions, and also the computer table look-up method for discrete distributions.

The third question of how to adequately test generators came into the picture. In 1965 (20), M.D. MacLaren and G. Marsaglia proposed that the tests commonly applied for testing were not of much value. They conducted their own tests on uniform generators. They were the uniformity, pairs, triples, sum of 2, sum of 4, maximum of 2, maximum of 5, minimum of 3 and minimum of 10. The generators being tested were the mixed congruential, the combination of two congruentials and the random number table.

The most recent work on testing methods was A.J. Sacks' Masters thesis (34). His work was a program which included the gap, runs, pairs, chi-square, moments, runs above-below mean, atuocorrelations and Kolmogorov-Smirnov tests.

Several books have been written trying to put all the generating methods together. Examples are (30) and (39).

It can be seen that random number generators have been studied a great deal by many. Upon examining the papers and books written, it was found that no satisfactory work has been done to include descriptions of all the methods known, their derivations and computer programs. Since computer simulation is very commonly used nowadays, a subroutine which gives random values from any statistical distribution will ease the work of anyone who does simulations. This motivaties this study. We attempt, in this thesis, to provide a quick understanding of the generators. Mathematical proofs are provided for most methods. For those derivations which required lengthy discussions, adequate references are supplied. The programs, which were all written in FORTRAN IV are listed for usage. Our work here is more or less a continuation of A.J. Sacks' work. Therefore, discussions on uniform generators were ommitted. We used a multiplicative congruential method for our iniform generator, and the function subroutine is from C.E. Donaghey who used it in his simulation course. Each generating method was evaluated either by a chi-sqaare test or by the t-test. The execution times, determined on UNIVAC 1108, of different methods were computed.

It seens that the study of random number generators has reached its final stage with all these methods found and all these testings done. Yet, it is not so, studies still have to be done on multi-variate generators.

CHAPTER II

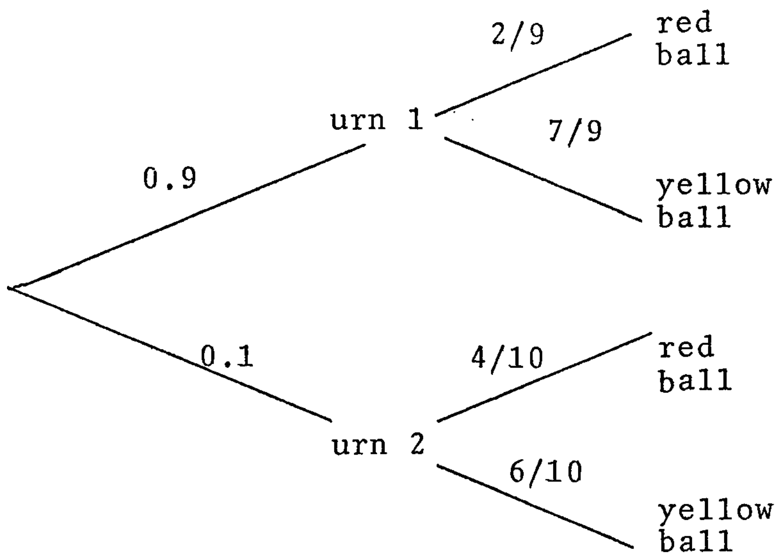
This chapter contains the definition of each distribution, descriptions and derivations of the generating methods, their program listings and testing results.

1. DISCRETE DISTRIBUTIONS

The following techniques are methods to instruct a computer to choose a random value from a discrete distribution with assigned probabilities.

A. MARSAGLIA METHOD

If drawing a red ball has a probability of 0.24 and drawing a yellow ball has a probability of 0.76, in order to get a ball at random, one can draw a ball from an urn containing 24 red and 76 yellow balls. To do this with a computer program, one can store 24 zero's in memory locations 1 to 24 and 76 one's in locations 25 to 100. A random integer between 1 and 100 is selected. If the integer falls between 1 and 24, the returned value would be a zero indicating that a red ball is selected. Otherwise, a yellow ball is selected for a returned value between 25 and 100. One hundred memory locations are required for this example. When the probabilities are given in 3 places after the decimal, 1000 memory locations are required and so forth. In order to conserve memory locations one can use two urns instead of one. For our example, 2 red balls and 7 yellow balls are put in urn one; while 4 red balls and 6 yellow balls are put in urn two. 0.2 and 0.7 are added together to obtain 0.9 which is the probability that urn one would be selected. 0.04 and 0.06 are added together to obtain 0.10 which is the probability that urn two would be selected. With a look at the tree diagram, it is obvious that the probabilities of red and yellow balls being selected remain unchanged.



$$
\begin{aligned}
P(\text{red ball}) &= P(\text{urn } 1)*P(\text{red ball}|\text{urn } 1) \\
&\quad + P(\text{urn } 2)*P(\text{red ball}|\text{urn } 2) \\
&= 0.9*2/9 + 0.1*4/10 \\
&= 0.24
\end{aligned}
$$

$$
\begin{aligned}
P(\text{yellow ball}) &= P(\text{urn } 1)*P(\text{yellow ball}|\text{urn } 1) \\
&\quad + P(\text{urn } 2)*P(\text{yellow ball}|\text{urn } 2) \\
&= 0.9*7/9 + 0.1*6/10 \\
&= 0.76
\end{aligned}
$$

With this slight change of doing one more step to select the urn, 100 memory locations are drastically reduced to 19 locations. Based on the above logic, Marsaglia has designed a method for obtaining a random number from any discrete distribution. It can easily be understood from the example on the next page.

Example 1.A.1:

Let the values of x_1, x_2, x_3, x_4, x_5 and x_6 be 0,1,2,3,4 and 5.

Let their corresponding probabilities $P(x_1)$, $P(x_2)$, $P(x_3)$, $P(x_4)$, $P(x_5)$ and $P(x_6)$ be 0.6250, 0.2047, 0.0813, 0.0651, 0.0134 and 0.0105.

Contents are loaded in the memory locations of the computer according to the following scheme:

Loc	Con	Loc	Con	Loc	Con	Loc	Con
1	0	17	2	33	1	49	1
2	0	18	2	34	1	50	1
3	0	19	3	35	1	51	1
4	0	20	3	36	2	52	2
5	0	21	3	37	3	53	2
6	0	22	3	38	3	54	2
7	1	23	3	39	3	55	3
8	1	24	3	40	3	56	4
9	0	25	4	41	3	57	4
10	0	26	5	42	4	58	4
11	2	27	0	43	4	59	4
12	2	28	0	44	4	60	5
13	2	29	0	45	1	61	5
14	2	30	0	46	1	62	5
15	2	31	0	47	1	63	5
16	2	32	1	48	1	64	5

Notice that each probability is a number up to four places after the decimal. The first digit after the decimal of $P(x_1)$ is a 6, therefore, 6 zeros were put in locations 1 to 6. The first digit after the decimal of $P(x_2)$ is a 2, therefore, 2 ones were put in locations 7 and 8. After all the first digits of all the probabilities were considered and contents loaded, then the second, the third and the fourth digits were considered and contents were loaded in the same way.

Let S_i be the sum of the digits of all the probabilities at the ith place after the decimals.

In our example:

$$S_1 = 6+2+0+0+0+0 = 8$$

$$S_2 = 2+0+8+6+1+1 = 18$$

$$S_3 = 5+4+1+5+3+0 = 18$$

$$S_4 = 0+7+3+1+4+5 = 20$$

Let d_i be the ith digit after the decimal of a number between zero and one.

To obtain a random number from this example:

1) Generate a uniform random number $U=.d_1d_2d_3d_4\ldots$

2) Test the following:

If $d_1 < S_1 = 8$, the desired random number is the content of location d_1.

If $80 \leq d_1d_2 < 10S_1+S_2 = 98$, the desired random number is the content of location $(d_1d_2-80+S_1+1)$.

If $980 \leq d_1d_2d_3 < 100S_1+10S_2+S_3 = 998$, the desired random number is the content of location $(d_1d_2d_3-980+S_1+S_2+1)$.

If $1000S_1+100S_2+10S_3+S_4 \leq d_1d_2d_3d_4 = 9980$, the desired random number is the content of location $(d_1d_2d_3d_4-9980+S_1+S_2+S_3+1)$.

PROGRAM LISTING

```
C*********************************************************
C                                                        *
C     TO RETURN A RANDOM VARIATE OF ANY DISCRETE         *
C     DISTRIBUTION BY MARSAGLIA METHOD                   *
C                                                        *
C     P = INPUT VECTOR WITH THE VARIABLES AND THEIR      *
C         PROBABILITIES IN THE ORDER OF X(1),P(1),X(2),  *
C         P(2),...                                       *
C                                                        *
C     ND = THE DIMENSION OF VECTOR P                     *
C                                                        *
C     VAL = THE RETURNED VARIATE                         *
C                                                        *
C*********************************************************
C
      SUBROUTINE DISC(P,ND,VAL)
      DIMENSION P(110),CON(10000),INDX(4),NO(4,110)ISUM(4)
      DATA NSEED/567801/
      K=ND/2
      M=1
      DO 9 I=1,K
      P(K+1)=P(K+I)+0.00005
      NO(1,I)=P(K+I)*10
      NO(2,I) P(K+I)*100-NO(1,I)*10
      NO(3,I)=P(K+I)*1000-NO(1,I)*100-NO(2,I)*10
      NO(4,I)=P(K+I)*10000-NO(1,I)*1000-NO(2,I)*100-NO(3,I)*10
    9 CONTINUE
      DO 5 N=1,4
      ISUM(N)=0
      INDX(N)=M
      DO 3 J=1,K
      NUM=NO(N,J)
      IF(NUM.EQ.0) GO TO 3
      IF(M.EQ.1) LK=M+NUM-1
      IF(M.GT.1) LK=M+NUM
      DO 4 L=M,LK
      CON(L)=P(J)
    4 CONTINUE
      M=M+NUM
    3 CONTINUE
    5 CONTINUE
      DO 87 I=1,K
      ISUM(1)=ISUM(1)+NO(1,I)
      ISUM(2)=ISUM(2)+NO(1,I)*10+NO(2,I)
      ISUM(3)=ISUM(3)+NO(1,I)*100+NO(2,I)*10+NO(3,I)
      ISUM(4)=ISUM(4)+NO(1,I)*1000+NO(2,I)*100+NO(3,I)+10+NO(4,I)
   87 CONTINUE
```

```
      T1=ISUM(1)/10.
      T2=ISUM(2)/100.
      T3=ISUM(3)/1000.
      T4=ISUM(4)/10000.
      U=RAN(NSEED)
      IF(U.GE.0..AND.U.LT.T1) IX=U*10+INDX(1)
      IF(U.GE.T1.AND.U.LT.T2) IX=U*100-ISUM(1)*10+INDX(2)
      IF(U.GE.T2.AND.U.LT.T3) IX=U*1000-ISUM(2)*10+INDX(3)
      IF(U.GE.T3.AND.U.LT.T4) IX=U*10000-ISUM(3)*10+INDX(4)
      VAL=CON(IX)
      DO 99 J=1,K
      P(K+J)=P(K+J)-0.00005
   99 CONTINUE
      RETURN
      END
```

B. CUMULATION PROBABILITY METHOD

With example 1.A. 1:

variable	probability	cumulative probability
x_1	0.6250	0.6250
x_2	0.2047	0.8297
x_3	0.0813	0.9110
x_4	0.0651	0.9761
x_5	0.0134	0.9895
x_6	0.0105	1.0000

To obtain a discrete random number, generate a uniform random number U in (0,1), if U is less or equal to the cumulative probability of x_i, i=1, 2, 3, 4, 5 or 6; then the desired number is x_i.

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C     TO RETURN A RANDOM VARIATE OF ANY DISCRETE BY       *
C     CUMULATIVE PROBABILITY METHOD                       *
C                                                         *
C     P = INPUT VECTOR WITH THE VARIABLES AND THEIR       *
C         PROBABILITIES IN THE ORDER OF X(1),P(1),x(2),   *
C         P(2),...                                        *
C                                                         *
C     ND = THE DIMENSION OF VECTOR P                      *
C                                                         *
C     VAL = THE RETURNED VARIATE                          *
C                                                         *
C**********************************************************
C
      SUBROUTINE DISC(P,ND,VAL)
      DIMENSION P(100),PM(100)
      DATA NSEED/567801/
      N=ND/2+1
      K=ND/2+2
      R=RAN(NSEED)
      PM(K-1)=P(K-1)
      DO 1 I=K,ND
      PM(I)=P(I)+PM(I-1)
    1 CONTINUE
      DO 2 J=N,ND
      IF(R.GE.0.0.AND.R.LT.PM(J)) GO TO 3
    2 CONTINUE
    3 VAL=P(J-K+2)
      RETURN
      END
```

PROGRAM TESTING RESULTS

Example 1.A.1

i	x_i	MARSAGLIA $P(x_i)$ N=100	MARSAGLIA $P(x_i)$ N=1000	CDF METHOD $P(x_i)$ N=100	CDF METHOD $P(x_i)$ N=1000	GIVEN $P(x_i)$
1	0	0.6400	0.6440	0.6200	0.6450	0.6250
2	1	0.1900	0.2090	0.2000	0.2080	0.2047
3	2	0.0600	0.0590	0.0800	0.0610	0.0813
4	3	0.1000	0.0750	0.0800	0.0620	0.0651
5	4	0.0000	0.0060	0.0100	0.0130	0.0134
6	5	0.0100	0.0070	0.0100	0.0110	0.0105

Example 1.A.2

i	x_i	MARSAGLIA $P(x_i)$ N=100	MARSAGLIA $P(x_i)$ N=1000	CDF METHOD $P(x_i)$ N=100	CDF METHOD $P(x_i)$ N=1000	GIVEN $P(x_i)$
1	1	0.1800	0.2270	0.2100	0.2320	0.2167
2	2	0.0300	0.0180	0.0400	0.0190	0.0192
3	3	0.4500	0.4180	0.3800	0.4280	0.4201
4	4	0.0700	0.0490	0.0500	0.0490	0.0480
5	5	0.1300	0.1710	0.1900	0.1520	0.1623
6	6	0.0300	0.0230	0.0100	0.0230	0.0355
7	7	0.1000	0.0770	0.1000	0.0780	0.0802
8	8	0.0100	0.0170	0.0200	0.0190	0.0180

N is the number of random variates generated for testing.

METHODS	EXECUTION TIME IN MILLI-SECONDS			
	Example 1.A.1		Example 1.A.2	
	N=100	N=1000	N=100	N=1000
MARSAGLIA	175	1561	46	362
CDF METHOD	22	151	21	148

SUMMARY OF CHI-SQUARE TEST

METHODS	N	Example 1.A.1		Example 1.A.2	
		degrees of freedom	chi-square values	degrees of freedom	chi-square values
MARSAGLIA	1000	5	13.543	7	5.647
	2500	5	5.445	7	3.730
	5000	5	3.623	7	3.192
CDF METHOD	1000	5	5.945	7	6.423
	2500	5	4.648	7	4.536
	5000	5	3.223	7	7.287

Degrees of freedom	α	Critical χ^2
5	0.05	11.0706
5	0.10	9.2363
7	0.05	14.0671
7	0.10	12.0170

N is the number of random variates generated for testing.

2. HYPERGEOMETRIC DISTRIBUTION

Suppose there are N items, N_p of which belongs to class I and N_q of which belongs to class II. p is the probability that an item of class I will be chosen, and q is the probability that an item of class II will be chosen, where p+q=1. Suppose n items were chosen at random from the lot ($n \leq N$), without replacement, Let X be the number in class I found. Since X=x if and only if we obtained precisely x class I items and precisely (n-x) class II items, we have

$$f(x) = P(X=x) = \frac{\binom{N_p}{x}\binom{N_q}{n-x}}{\binom{N}{n}}, \qquad x=0,1,2,\ldots$$

A discrete random variable having a probability distribution as above is said to have a hypergeometric distribution. The expected value and variance of x are:

$$EX = np$$

$$VX = npq\left(\frac{N-n}{N-1}\right) .$$

A. DIRECT METHOD

The steps to obtain a hypergeometric variate with N, n and p given are:

1) Set x = 0.

2) Generate a uniform variate R in (0,1).

3) If R is less or equal to p, increase x by 1, decrease n by 1 and recalculate p.

4) If R is greater than one, decrease n by 1.

5) Go back to step 2 if n is greater than zero.

6) Stop if n is equal to zero, x is the required variate.

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C     TO RETURN A HYPERGEOMETRIC RANDOM VARIATE BY        *
C     DIRECT METHOD                                       *
C                                                         *
C     TN = TOTAL NUMBER OF ITEMS                          *
C                                                         *
C     NC = NUMBER OF ITEMS CHOSEN                         *
C                                                         *
C     P  = PROBABILITY THAT AN ITEM OF CLASS I WILL BE    *
C          CHOSEN                                         *
C                                                         *
C     VAL=THE RETURNED HYPERGEOMETRIC VARIATE             *
C                                                         *
C**********************************************************
C
      SUBROUTINE HYPGEO(TN,NC,P,VAL)
      DATA NSEED/567801/
      PX=P
      TNX=TN
      VAL=0.0
      DO 11 I=1,NC
      R=RAN(NSEED)
      IF(R-PX) 6,6,9
    6 S=1.0
      VAL=VAL+1.0
      GO TO 10
    9 S=0.0
   10 PX=(TNX*PX-S)/(TNX-1.0)
      TNX=TNX-1.0
   11 CONTINUE
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHOD	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
DIRECT	1.900	1.966	0.530	0.623	2.0	0.666

The given parameters of this example are:

TN = 10.0
NC = 5
P = 0.4

METHOD	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
DIRECT	57	490

N is the number of random variates generated for testing. Notice that the execution time of this method depends on the input parameters.

SUMMARY OF CHI-SQUARE TEST

METHOD	N	Degrees of freedom	Chi-square values
DIRECT	50	2	3.3405
	100	2	3.7620
	150	2	2.6880

Degrees of freedom	α	Critical χ^2
2	0.05	2.7055
2	0.10	4.6052

N is the number of random variates generated for testing.

3. GEOMETRIC DISTRIBUTION

Let A be an event. Define the random variable X as the number of repetitions required up to and including the first occurrence of A. Thus X assumes the possible values 1,2,... . Sine X=x if and only if the first (x-1)th repetitions of the experiment results not in A while the xth repetition results in A, we have

$$f(x) = P(X=x) = q^{x-1}p, \qquad x=1,2,...$$

A random variable with probability distribution as above is said to have a geometric distribution.
The cumulative distribution function is

$$F(x) = P(X \leq x) = \sum_{i=0}^{x} pq^{i}$$

The expected value and variance of x are:

$$EX = q/p$$

$$VX = q/p^2 = EX/p$$

p is the probability that A occurs, and q=1-p.

A. NAYLOR METHOD

To obtain a geometric distributed variate by the Naylor method, the steps are:

1) Generate a uniformly distributed random number U in (0,1).
2) Calculate $x = \mathrm{Log}_e U / \mathrm{Log}_e q$, x is the required variate.

Verification:

$$F(0) = P(x=0) = \sum_{i=0}^{0} pq^i = p$$

and $F(x) = P(X \leq x)$

therefore $p \leq F(x) \leq 1$

$$P(X > x) = 1 - F(x)$$

implies
$$\begin{aligned} P(X > 0) &= 1 - F(0) \\ &= 1 - p \\ &= q \end{aligned}$$

Consider $1-F(x)$

Let
$$\begin{aligned} S &= 1-F(x) = 1-p-pq-pq^2-\ldots-pq^x \\ -)\quad qS &= q-pq-pq^2-\ldots-pq^x-pq^{x+1} \\ \hline S-qS &= 1-q-p+pq^{x+1} \\ &= p-p+pq^{x+1} \\ S(1-q) &= pq^{x+1} \\ pS &= pq^{x+1} \\ S &= q^{x+1} \end{aligned}$$

therefore $1-F(x) = q^{x+1}$

$$p \leq F(x) \leq 1$$

implies
$$-p \geq -F(x) \geq -1$$

$$1-p \geq 1-F(x) \geq 1-1$$

$$q \geq q^{x+1} \geq 0$$

$$1 \geq q^{x} \geq 0$$

substituting q^x by $(1-F(x))/q$, shows that $(1-F(x))/q$ has unit length.

Let
$$U = (1-F(x))/q$$

$$U = q^x$$

$$\text{Log}_e U = x\text{Log}_e q$$

$$x = \frac{\text{Log}_e U}{\text{Log}_e q}$$

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C       TO RETURN A GEOMETRIC RANDOM VARIATE BY NAYLOR    *
C       METHOD                                            *
C                                                         *
C       P = THE PROBABILITY THAT THE EVENT WILL OCCUR     *
C                                                         *
C       VAL = THE RETURNED GEOMETRIC VARIATE              *
C                                                         *
C**********************************************************
C
       SUBROUTINE GEOM(P,VAL)
       DATA NSEED/567801/
       R=RAN(NSEED)
       VAL=ALOG(R)/ALOG(1.-P)
       RETURN
       END
```

B. DIRECT METHOD

This method is based on the definition of a geometric variate.

Steps to obtain a geometric random variate:

1) Set x=0.

2) Generate a uniform random variate R in (0,1).

3) If R is greater than the given p, the probability for the occurrence of the event, then increase x by 1 and go back to step 2.

4) If R is equal to or smaller than p, x is the required geometric variate.

PROGRAM LISTING

```
C**********************************************************
C                                                        *
C      TO RETURN A GEOMETRIC RANDOM VARIATE BY DIRECT    *
C      METHOD                                            *
C                                                        *
C      P = THE PROBABILITY THAT THE EVENT WILL OCCUR     *
C                                                        *
C      VAL = THE RETURNED GEOMETRIC VARIATE              *
C                                                        *
C**********************************************************
C
      SUBROUTINE GEOM(P,VAL)
      DATA NSEED/567801/
      X=0.0
    4 R=RAN(NSEED)
      IF(R.LE.P) GO TO 3
      X=X+1.0
      GO TO 4
    3 VAL=X
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
NAYLOR	2.610	2.473	7.173	6.419	2.03	6.152
DIRECT	1.600	1.917	3.480	5.724	2.03	6.152

The parameter given in this example is:

$$P = 0.33$$

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
NAYLOR	23	265
DIRECT	20	161

N is the number of random variates generated for testing. Notice that the execution time of the Direct method depends on the input parameter , while the Naylor method does not.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	Degrees of freedom	Chi-square values
NAYLOR	50	3	11.442
	100	5	12.847
	150	5	18.922
DIRECT	50	5	4.228
	100	5	1.369
	150	7	1.402

Degrees of freedom	α	Critical χ^2
3	0.05	7.81473
3	0.10	6.25139
5	0.05	11.07050
5	0.10	9.23635
7	0.05	14.06710
7	0.10	12.01700

N is the number of random variates generated for testing.

4. PASCAL DISTRIBUTION

Let A be an event. Define the random variable X as the number of failures when the trials are repeated until k successes occur (k>1). Then X has a negative binomial or Pascal distribution.

$$f(x) = P(X=k+x) = \binom{k+x-1}{x-1} p^k q^x \qquad x=0,1,2,...$$

where p is the probability of the occurrence of A, and q=1-p, the probability of the failure of A.

The mean and the variance of x are:

$$EX = kq/p$$

$$VX = kq/p^2 = EX/p \quad .$$

A. SUM GEOMETRIC METHOD

Notice that when $k = 0$, $f(x) = pq^x$.

Therefore when k is an integer, a pascal variate is the sum of k geometric variates.

Let G be a geometric variate, the U be a uniform variate in (0,1); if X is a pascal variate,

$$X = \sum_{n=1}^{k} G_n$$

$$X = \frac{\sum_{n=1}^{k} \text{Log}_e U_n}{\text{Log}_e q}$$

$$X = \frac{\text{Log}_e \prod_{n=1}^{k} U_n}{\text{Log}_e q}$$

To obtain a geometric random variate:

1) Generate k random uniform variates in (0,1) and calculate their product, say it is Z.
2) Calculate $X = \text{Log}_e Z/\text{Log}_e q$, X is the required variate.

PROGRAM LISTING

```
C*********************************************************
C                                                        *
C     TO RETURN A RANDOM PASCAL VARIATE BY SUM GEOMETRIC *
C     METHOD                                             *
C                                                        *
C     K = THE NUMBER OF SUCCESSES                        *
C                                                        *
C     Q = THE PROBABILITY THAT THE EVENT WILL NOT OCCUR  *
C                                                        *
C     VAL = THE RETURNED PASCAL VARIATE                  *
C                                                        *
C*********************************************************
C
      SUBROUTINE PASCAL(K,Q,VAL)
      DATA NSEED/567801/
      TR=0.0
      QR=ALOG(Q)
      DO 184 I=1,K
      TR=TR+ALOG(RAN(NSEED))
  184 CONTINUE
      IX=TR/QR
      X=IX
      RETURN
      END
```

Notice that in order to avoid getting an argument too small for the logarithmic function of the computer, in our program, the sum of the log of k uniform variates is used instead of the log of the product of k uniform variates.

B. NEGATIVE BINOMIAL METHOD

The pascal distribution is also known as the negative binomial distribution. Therefore, we can use the opposite reasoning of generating a binomial variate to generate a pascal variate.

Steps to obtain a random pascal variate:

Let N be the total number of trials.

Let M be the number of successful trials.

1) Set N and M to be zeros.
2) Generate R, a uniform random variate in (0,1) and increase N by 1.
3) If R is greater than q, the trial is successful, so increase M by 1.
4) If M is equal to k, it means that k successes has already occurred, then go to step 6.
5) Go to step 2.
6) The required pascal variate is equal to N-M.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C       TO RETURN A PASCAL RANDOM VARIATE BY NEGATIVE      *
C       BINOMIAL METHOD                                    *
C                                                          *
C       K = THE NUMBER OF SUCCESSES                        *
C                                                          *
C       Q = THE PROBABILITY THAT THE EVENT WILL NOT OCCUR  *
C                                                          *
C       VAL = THE RETURNED PASCAL VARIATE                  *
C                                                          *
C***********************************************************
C
       SUBROUTINE PASCAL(K,Q,VAL)
       DATA NSEED/567801/
       N=0
       M=0
     1 R=RAN(NSEED)
       N=N+1
       IF(R.GT.Q) M=M+1
       IF(M.EQ.K) GO TO 2
       GO TO 1
     2 X=N-M
       RETURN
       END
```

C. TOTAL TRIAL METHOD

A Pascal variate can be defined as the total number of trials such that x failures occurred before k successes occurred. The variate is now k+x instead of x.

The mean of k+x is:

$$EX(x+k) = EX(x) + EX(k)$$

$$= \frac{kq}{p} + k$$

$$= \frac{kq+kp}{p}$$

$$= \frac{k(1-q+q)}{p}$$

$$= \frac{k}{p}$$

The variance of k+x is:

$$VX(k+x) = EX(k+x)^2 - EX^2(k+x)$$

$$= EX(k^2+2kx+x^2) - \frac{k^2}{p^2}$$

$$= k^2+2kEX(x)+EX(x^2) - \frac{k^2}{p^2}$$

$$= k^2 + \frac{2k^2}{p} + \frac{kq}{p^2} + \frac{kq}{p} - \frac{k^2}{p^2}$$

Steps to obtain a random Pascal variate:

Let N be the total number of trials.

Let M be the number of successful trials.

1) Set N and M to be zeros.
2) Generate R, a uniform random variate in (0,1) and increase N by 1.
3) If R is greater than q, the trial is successful, so increase M by 1.
4) If M is equal to k, it means that k successes have already occurred, then go to step 6.
5) Go to step 2.
6) The required Pascal variate is equal to N.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A RANDOM PASCAL VARIATE BY TOTAL TRIAL       *
C     METHOD                                                 *
C                                                            *
C     K = THE NUMBER OF SUCCESSES                            *
C                                                            *
C     Q = THE PROBABILITY THAT THE EVENT WILL NOT OCCUR      *
C                                                            *
C     VAL = THE RETURNED PASCAL VARIATE                      *
C                                                            *
C*************************************************************
C
      SUBROUTINE PASCAL(K,Q,VAL)
      DATA NSEED/567801/
      N=0
      M=0
    1 R=RAN(NSEED)
      N=N+1
      IF(R.GT.Q) M=M+1
      IF(M.EQ.K) GO TO 2
      GO TO 1
    2 X=N
      RETURN
      END
```

D. EXPONENTIAL METHOD

Since a geometric variable is defined as the number of repetitions required up to and including the first occurrence of an event, it can be considered as the units of waiting time before the first arrival. Therefore, the geometric distribution is the discrete analogue of the exponential distribution. The conditional probability that the waiting time terminates at the xth trial is $q^x p$ where p is the probability that the waiting time terminates at the first trial.

Suppose each trial takes a time Δt, so that x trials take time $x\Delta t$.

Let $t \to 0$ and $x \to \infty$, so that the total time $t = x\Delta t$ remains constant. If p and q were constants, the first success will occur sooner or later and in the limit the waiting time would be zero. In order to have the waiting time not to be zero, let $p \to 0$ and $q \to 1$.

The mean of the geometric variate is q/p, and the mean of the waiting time is $q\Delta t/p$.

Let $q\Delta t/p = 1/\lambda$ be a constant.

Since $p \to 0$ and $q \to 1$, we have

$$
\begin{aligned}
p &\sim p/q \\
&\sim \frac{p}{q\Delta t}\Delta t \\
&\sim \lambda\Delta t
\end{aligned}
$$

Substituting $x\Delta t=t$, and $p\sim\lambda\Delta t$ into the f(x) of Pascal, we have

$$\binom{k+\frac{t}{\Delta t}-1}{\frac{t}{\Delta t}-1}(1-\lambda\Delta t)^{t/\Delta t}(\lambda\Delta t)^{k}$$

$$= e^{-\lambda t}\,\frac{(\lambda t)^{k-1}}{(k-1)!}$$

which is the Poisson expression for the probability of exactly k-1 events within time t.

Let $k=1$ and $\Delta t=1$;

$$\lambda\sim\Delta t\,\frac{p}{q}$$

$$\text{implies}\quad \lambda\sim\frac{1-q}{q}$$

We know that the exponential distribution is a special case of the gamma distribution when the scale paraemter is one. Therefore, to obtain the Poisson parameter, say α, for the Pascal, we generate a gamma variate with shape parameter λ and scale parameter k.

To obtain a Pascal random variate:

1) Calculate λ.
2) Call subroutine gamma for α.
3) Call subroutine Poisson for Pascal random variate x.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM PASCAL VARIATE BY EXPONENTIAL      *
C     METHOD                                                *
C                                                           *
C     K = THE NUMBER OF SUCCESS                             *
C                                                           *
C     Q = THE PROBABILITY THAT THE EVENT WILL NOT OCCUR     *
C                                                           *
C     VAL = THE RETURNED PASCAL VARIATE                     *
C                                                           *
C************************************************************
C
      SUBROUTINE PASCAL(K,Q,VAL)
      DATA NSEED/567801/
      ADK=(1.-Q)/Q
      CALL GAMMA(K,ADK,0.0,T)
      CALL POIS(T,Y)
      VAL+Y
      RETURN
      END
```

E. NON-INTEGER NUMBER OF SUCCESSES

Let d_1 be the unit digit and d_2 be the first digit after the decimal of k. Then $k=d_1.d_2$. We can generate $(10-d_2)$ Pascal variates with the number of successes equal to d_1; and d_2 Pascal variates with the number of successes equal to (d_1+1). The expected value of the number of successes is:

$$\frac{(10-d_2)*d_1+d_2*(d_1+1)}{10}$$

$$=\frac{10d_1-d_2d_1+d_2d_1+d_2}{10}$$

$$=\frac{10d_1+d_2}{10}$$

$$= d_1.d_2 \qquad \text{which is equal to k.}$$

To obtain a random Pascal variate when k is given as $d_1.d_2$:

1) Generate $10-d_2$ Pascal variates with the number of successes equal to d_1.
2) Generate d_2 Pascal variates with the number of successes equal to d_1+1.
3) Divide the sum of the variates by 10. The result is the required Pascal variate.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM PASCAL VARIATE OF NON-INTEGER      *
C     NUMBER OF SUCCESSES                                   *
C                                                           *
C     PK = THE NUMBER OF SUCCESSES                          *
C                                                           *
C     Q = THE PROBABILITY THAT THE EVENT WILL NOT OCCUR     *
C                                                           *
C     VAL = THE RETURNED PASCAL VARIATE                     *
C                                                           *
C************************************************************
C
      SUBROUTINE PASCAL(PK,Q,VAL)
      DATA NSEED/567801/
      N=PK
      K=(PK-N)*10
      SUM=0.0
      NN=0
      II=0
      L=N+1
      DO 2 I=1,K
    3 R=RAN(NSEED)
      IF(R.GT.Q) II=II+1
      NN=NN+1
      IF(II.EQ.L) GO TO 4
      GO TO 3
    4 SUM=SUM+(NN-II)
      NN=0
      II=0
    2 CONTINUE
      KK=10-K
      DO 12 I=1,KK
   13 R=RAN(NSEED)
      IF(R.GT.Q) II=II+1
      NN=NN+1
      IF(II.EQ.N) GO TO 14
      GO TO 13
   14 SUM=SUM+(NN-II)
      NN=0
      II=0
   12 CONTINUE
      VAL=SUM/10.
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	N	PROGRAM RESULTS		GIVEN	
		Mean	Variance	Mean	Variance
SUM GEOM	100	22.920	137.154	21.316	112.188
	1000	23.633	113.136	21.316	112.188
NEG BINOM.	100	20.560	121.146	21.316	112.188
	1000	21.144	115.695	21.316	112.188
TOTAL TRIAL	100	13.830	20.661	14.286	19.730
	1000	14.054	16.749	14.286	19.730
EXPONENTIAL	100	19.030	85.169	21.316	112.188
	1000	21.277	116.715	21.316	112.188
NON-INTEGER	100	2.417	0.364	2.607	4.495
	1000	2.532	0.431	2.607	4.495

The input parameters of the example for SUM GEOM, NEG BINOMIAL and EXPONENTIAL methods are:

K = 5.0
Q = 0.81 (see pages 30 & 32)

The input parameters of the example for TOTAL TRIAL method are:

K = 6.0
Q = 0.58 (see page 34)

The input parameters of the example for NON-INTEGER method are:

PK = 3.60
Q = 0.42 (see page 39)

N is the number of random variates generated for testing.

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
SUM NORM	72	684
NEG BINOM	203	1983
TOTAL TRIAL	135	1126
EXPONENTIAL	242	2248
NON-INTEGER	314	3811

N is the number of random variates generated for testing. Notice that the execution times of SUM NORM, NEG BINOMIAL, TOTAL TRIAL and NON-INTEGER methods all depend on the input parameters.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	Degrees of freedom	Chi-square values
SUM GEOM	100 150	8 9	12.892 14.324
NEG BINORM.	100 150	8 9	9.595 9.223
TOTAL TRIAL	100 150	8 9	43.687 50.806
EXPONENTIAL	100 150	8 9	12.091 13.306

Degrees of freedom	α	Critical χ^2
8	0.05	15.5073
8	0.10	13.3616
9	0.05	16.9190
9	0.10	14.6837

N is the number of random variates generated for testing. Since any one method in the above table can be changed to generate a Pascal variate with non-integer number of successes, therefore to do the chi-square test for the non-integer method would be a repetition.

5. BINOMIAL DISTRIBUTION

Let A be an event. Define the random variable X as the number of successes when the trials are repeated N times. Then X has a binomial distribution.

$$f(x) = P(X=x) = \binom{N}{x} p^x q^{N-x} \qquad x=0,1,2,\ldots,N$$

where p is the probability of the occurrence of A, and q=1-p, the probability of the failure of A.

$$EX = Np$$

$$VX = Npq$$

A. DIRECT METHOD

This method is straightforward by using the definition of the distribution. The steps are as below:

1) Set X equal to 0.0.
2) Generate R, a uniform random variate in (0,1).
3) If R is less or equal to p, that means the event occurs; increase X by 1.
4) If R is greater than p, the event does not occur, go to step 2.
5) After N uniform random variates are generated, X is the desired variate.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A BINOMIAL RANDOM VARIATE BY DIRECT METHOD *
C                                                          *
C     NT = THE NUMBER OF TRIALS                            *
C                                                          *
C     P = THE PROBABILITY THAT THE EVENT WILL OCCUR        *
C                                                          *
C     VAL = THE RETURNED BINOMIAL VARIATE                  *
C                                                          *
C***********************************************************
C
      SUBROUTINE BINOM(NT,P,VAL)
      DATA NSEED/567801/
      X=0.0
      DO 7 I=1,NT
      R=RAN(NSEED)
      IF(R-P) 6,6,7
    6 X=X+1.0
    7 CONTINUE
      RETURN
      END
```

B. NORMAL APPROXIMATION

For large number of tails, N the binomial variate x can be approximated by the normal distribution with mean equal to Np and variance equal to Npq.

Verification:

$$\text{Let} \quad h = \frac{x-Np}{\sqrt{Npq}}$$

We need to prove that the probability mass function P(h)

$$\simeq \frac{1}{\sqrt{Npq}} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}h^2}$$

which is equivalent to showing

$$\lim_{N\to\infty} \frac{\sqrt{2\pi Npq}\ P(h\sqrt{Npq}+Np)}{e^{-\frac{1}{2}h^2}} = 1$$

First, consider

$$P(h\sqrt{Npq}+Np) = P(x)$$

$$= \binom{N}{x} p^x q^{N-x}$$

By Stirling's formula, $m! = \sqrt{2\pi m}\ m^m e^{-m} e^{r(m)}$ while $0 < r(m) < \frac{1}{12m}$

$$\binom{N}{x} p^x q^{N-x} = \frac{N!}{x!(N-x)!} p^x q^{N-x}$$

$$= \frac{\sqrt{2\pi N}\ N^N e^{-N} e^{r(N)} p^x q^{N-x}}{\sqrt{2\pi x} x^x e^{-x} e^{r(x)} \sqrt{2\pi(N-x)}\ (N-x)^{N-x} e^{-(N-x)} e^{r(N-x)}}$$

$$= \frac{1}{\sqrt{2\pi}} \sqrt{\frac{N}{x(N-x)}} \left(\frac{Np}{x}\right)^x \left(\frac{Nq}{N-x}\right)^{N-x} e^{-r(x)-r(N-x)+r(N)}$$

Let $R=-r(x)-r(N-x)+r(N)$

then $$0<R<-\frac{1}{12x}-\frac{1}{12(N-x)}+\frac{1}{12N}$$

implies $$-\left[\frac{1}{12x}+\frac{1}{12(N-x)}+\frac{1}{12N}\right]<R<\frac{1}{12x}+\frac{1}{12(N-x)}+\frac{1}{12N}$$

implies $$|R|<\frac{1}{12}\left[\frac{1}{x}+\frac{1}{N-x}+\frac{1}{N}\right]$$

Second, consider

$$-\mathrm{Log}_e\left[\left(\frac{Np}{x}\right)^x\left(\frac{Nq}{N-x}\right)^{N-x}\right]$$

Using the expansion $\mathrm{Log}_e(1+x)=x-\frac{1}{2}x^2+\theta x^3$ for some θ such that $|\theta|<1$, valid for $|x|<\frac{1}{4}$, we obtain that

$$-\mathrm{Log}_e\left[\left(\frac{Np}{x}\right)^x\left(\frac{Nq}{N-x}\right)^{N-x}\right]$$

$$=(Np+h\sqrt{Npq})\,\mathrm{Log}_e(1+h\sqrt{q/Np})$$

$$+\ (Nq+h\sqrt{Npq})\,\mathrm{Log}_e(1-h\sqrt{p/Nq})$$

$$=(Np+h\sqrt{Npq})\left[h\sqrt{(q/Np)}-\frac{h^2}{2}\frac{q}{Np}+\text{terms in }\frac{1}{N^{3/2}}\right]$$

$$+\ (nq-h\sqrt{Npq})\left[-h\sqrt{(p/Nq)}\ -\frac{h^2}{2}\frac{p}{Nq}+\text{terms in }\frac{1}{N^{3/2}}\right]$$

$$=\left[h\sqrt{Npq}\ -\ q\frac{h^2}{2}\ +\ qh^2\ +\ \text{terms in}\frac{1}{N^{\frac{1}{2}}}\right]$$

$$+\left[-h\sqrt{Npq}\ -p\frac{h^2}{2}\ +\ ph^2\ +\ \text{terms in}\frac{1}{N^{\frac{1}{2}}}\right]$$

$$=\frac{1}{2}h^2\ +\ \text{terms in}\frac{1}{N^{\frac{1}{2}}}\quad .$$

Therefore $\left(\frac{Np}{x}\right)^{x}\left(\frac{Nq}{N-x}\right)^{N-x} = \mathrm{Exp}\left[-\left(\tfrac{1}{2}h^2 + \text{terms in } \frac{1}{N^{\frac{1}{2}}}\right)\right]$.

Third, consider $\dfrac{x(N-x)}{N}$;

Since $x=Np+h\sqrt{Npq}$, $\dfrac{x(N-x)}{N} = N(p+h\sqrt{pq/N})(q-h\sqrt{pq/N})$

$$\simeq Npq$$

At last subsitute R, Npq and $\mathrm{Exp}\left[-\left(\tfrac{1}{2}h^2 + \text{terms in } \frac{1}{N^{\frac{1}{2}}}\right)\right]$ into

$\binom{N}{x}p^x q^{N-x}$, we have

$$P(h\sqrt{Npq}+Np) = \frac{1}{\sqrt{2\pi Npq}}\ \mathrm{Exp}\left[-\left(\tfrac{1}{2}h^2 + \text{terms in } \frac{1}{N^{\frac{1}{2}}}\right)\right]e^R$$

which imples $\displaystyle\lim_{N\to\infty} \frac{\sqrt{2\pi Npq}\ P(h\sqrt{Npq}+Np)}{e^{-\frac{1}{2}h^2}} = 1$

Thus, we have shown that x is normally distributed with mean equal to Np and standard deviation equal to $\sqrt{Npq}$.

PROGRAM LISTING

```
C**********************************************************
C                                                        *
C     TO RETURN A BINOMIAL RANDOM VARIATE BY NORMAL      *
C     APPROXIMATION                                      *
C                                                        *
C     NT = THE NUMBER OF TRIALS                          *
C                                                        *
C     P = THE PROBABILITY THAT THE EVENT WILL OCCUR      *
C                                                        *
C     VAL = THE RETURNED BINOMIAL VARIATE                *
C                                                        *
C**********************************************************
C
      SUBROUTINE BINOM(NT,P,VAL)
      DATA NSEED/567801/
      EX=P*NT
      VX=P*NT*(1.-P)
      IF(NT.LT.15) GO TO 4
      VAL=-.0
      DO 1 I=1,12
      VAL=VAL+RAN(NSEED)
    1 CONTINUE
      VAL=(VAL-6.0)*SQRT(VX)+EX
      GO TO 5
    4 WRITE(6,3)
    3 FORMAT(' THE NUMBER OF TRAILS IS NOT LARGE ENOUGH TO
     +USE NORMAL APPROXIMATION ')
    5 RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
DIRECT	8.910	8.785	0.822	1.018	8.8	1.056
NORM APPROX.	4.955	4.482	4.938	4.216	5.0	4.500

The input parameters for the example of DIRECT method are:

NT = 10

P = 0.88

The input parameters for the example of NORM APPROXIMATION method are:

NT = 50

P = 0.1

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
DIRECT	79	852
NORM APPROX	87	1004

N is the number of random variates generated for testing.

SUMMARY OF T-TEST

METHODS	Degrees of freedom	T-values
DIRECT	99	1.2430
	999	0.5329
NORM APPROX.	99	0.2027
	999	7.9911

Degrees of freedom	α	Critical χ^2
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

6. POISSON DISTRIBUTION

Let X be a discrete random variable assuming the possible values: 0,1,2,...,n,... If

$$P(X = k) = \frac{e^{-\lambda t}\lambda t}{k!} \quad , \qquad k = 0,1,\ldots,n,\ldots$$

Then X has a Poisson distribution with parameter $\lambda>0$. λ is the arrival rate per unit of time and t is the unit of time. The mean and the variance of X are:

$$EX = \lambda t$$

$$VX = \lambda t$$

A. NORMAL METHOD

If a series of n independent Bernoulli trials were taken, in each of which there is a small probability p of an event occurring, then as n approaches infinity and p approaches zero in such a manner that $\lambda=np$ remains fixed, the probability of x occurrences is given by the Poisson distribution of t=1.

Proof:

$$P(x=k) = \binom{n}{k} p^k(1-p)^{n-k}$$

$$= \frac{n!}{k!(n-k)!} p^k(1-p)^{n-k}$$

$$= \frac{n(n-1)(n-2)\dots(n-k+1)}{k!} p^k(1-p)^{n-k}$$

Substitute in $p=\lambda/n$;

$$P(x=k) = \frac{n(n-1)\dots(n-k+1)}{k!} \left(\frac{\lambda}{n}\right)^k \left(\frac{n-\lambda}{n}\right)^{n-k}$$

$$= \frac{\lambda^k}{k!} \left[(1) \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right) \dots \left(1-\frac{k-1}{n}\right)\right] \left(1-\frac{\lambda}{n}\right)^{n-k}$$

$$= \frac{\lambda^k}{k!} \left[(1) \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right) \dots \left(1-\frac{k-1}{n}\right)\right] \left(1-\frac{\lambda}{n}\right)^{n} \left(1-\frac{\lambda}{n}\right)^{-k}$$

As n approaches to infinity,

$(1-1/n)(1-2/n)\dots$ approaches to 1, $(1-\lambda/n)^{-k}$ approaches to 1 and $(1-\lambda/n)^n$ approaches to $e^{-\lambda}$.

Therefore

$$P(x=k) = e^{-\lambda} \frac{\lambda^x}{x!} .$$

It is well known that Stirling's formula approximates $n! \sim \sqrt{2\pi} e^{-n} n^{n+\frac{1}{2}}$. And that for large n, using Stirling's formula for the various n!, Bernoulli trials can be approximated by the normal distribution (27).

$$P(x=k) = \binom{n}{k} p^k (1-p)^{n-k}$$

$$= \frac{1}{\sqrt{2\pi np(1-p)}} \exp\left(-\tfrac{1}{2}\left[\frac{k-np}{np(1-p)}\right]^2\right)$$

$$P(x<k) = P\left[\frac{x-np}{np(1-p)} \leq \frac{k-np}{np(1-p)}\right]$$

$$\approx \frac{1}{2\pi} \int_{-\infty}^{(k-np)/\sqrt{np(1-p)}} e^{-t^2/2} dt$$

As a result, a Poisson variate can be approximated by a normal distribution when n is large enough and with the mean and variance of the normal distribution both equal to λ.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A RANDOM POISSON VARIATE WITH TIME UNIT      *
C     EQUAL TO ONE BY NORMAL METHOD                          *
C                                                            *
C     A =ARRIVAL RATE PER UNIT OF TIME                       *
C                                                            *
C     VAL = THE RETURNED POISSON VARIATE                     *
C                                                            *
C*************************************************************
C
      SUBROUTINE POIS(A,VAL)
      CALL NORM(A,A,X)
      VAL=X
      RETURN
      END
```

B. EXPONENTIAL ARRIVAL TIME METHOD

It can be shown that if

1) The time intervals between successive arrivals are independently and identically distributed; further, the probability of an arrival occurring in the time interval between T and T+h depends only on the length h of the interval and not on T. The corresponding interarrival density function is designated as f(t).
2) In any interval of time $h > 0$, there is a positive probability of an arrival.
3) In any sufficiently small interval of time, at most only one arrival can occur.

Then the number of arrivals is Poisson distributed.

Suppose for simplicity that the system starts at time 0 and the first arrival occurs at time t, where $t > 0$. Therefore f(t) represents the density function for both the length of the arrival intervals as well as the actual time for the first arrival.

Define

$$r(T) = 1 - \int_0^T f(t)\,dt$$

so that

$$r(T) = P(\text{first arrival occurs after time } T)$$

Then from 1) and 2) above,

$$r(T+h) = r(T)r(h) \qquad \text{for all } T, h > 0$$

The only function that satisfies $r(T+h) = r(T)r(h)$ is

$$r(T) = e^{-\lambda T}$$

where λ is a positive constant. Therefore

$$e^{-\lambda T} = 1 - \int_0^T f(t)\,dt \ ,$$

So that

$$f(T) = \lambda e^{-\lambda T} \ .$$

Define

$$P_n(T) = P(n \text{ arrivals occur in the interval}(0,T))$$

Let $t = x$ be the time of the first arrival event, and according to (3), only one arrival enters at x. By 1), we can write

$$P_n(T) = \int_0^T P_{n-1}(T-x)\,f(x)\,dx$$

$$= \int_0^T P_{n-1}(y)\,f(T-y)\,dy \qquad \text{for } n=1,2,\ldots$$

where $y = T-x$. Using $f(t) = \lambda e^{-\lambda t}$,

$$P_n(T) = \int_0^T P_{n-1}(y)\lambda e^{-\lambda(T-y)}$$

Differentiating with respect to T yields

$$\frac{dP_n}{dT} = -\lambda P_n(T) + \lambda P_{n-1}(T) \qquad \text{for } n=1,2,\ldots$$

Claim that $P_n(T) = \frac{(\lambda T)^n e^{-\lambda T}}{n!}$

$$\frac{dP_n(T)}{dT} = \frac{e^{-\lambda T}}{n!} n(\lambda T)^{n-1} + \frac{(\lambda T)^n}{n!}(-\lambda)e^{-\lambda T}$$

$$= \frac{(\lambda T)^{n-1}e^{-\lambda T}}{(n-1)!}\lambda - \lambda e^{-\lambda T}\frac{(\lambda T)^n}{n!}$$

$$= -\lambda P_n(T) + \lambda P_{n-1}(T)$$

It has now been proved that:

1) The density function of the arrival time intervals between occurrences of consecutive events is exponential, with rexpected value equal to $1/\lambda$.
2) The probability of n events occurring during time t is Poisson. In one unit time interval, the expected value of n is λ.

If $t_1, t_2, \ldots$ are exponential time intervals with expected value equal to 1, such that

$$\sum_{i=0}^{x} t_i \leq \lambda < \sum_{i=0}^{x+1} t_i$$

then x is the required variate.

Since exponential $t_i = -\text{Log}_e r_i$, subsituting it in the above inequalities, we have

$$\sum_{i=0}^{x} -\text{Log}_e r_i \leq \lambda < \sum_{i=0}^{x+1} -\text{Log}_e r_i$$

$$\prod_{i=0}^{x} r_i \geq e^{-\lambda} > \prod_{i=0}^{x+1} r_i$$

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C     TO RETURN A POISSON RANDOM VARIATE WITH TIME UNIT   *
C     EQUAL TO ONE BY EXPONENTIAL ARRIVAL TIME METHOD     *
C                                                         *
C     A = ARRIVAL RATE PER UNIT OF TIME                   *
C                                                         *
C     VAL = THE RETURNED POISSON VARIATE                  *
C                                                         *
C**********************************************************
C
      SUBROUTINE POIS(A,VAL)
      DATA NSEED/567801/
      VAL=0.0
      TEST=EXP(-A)
      TR=1.0
    5 TR=TR*RAN(NSEED)
      IF(TR-TEST) 10,8,8
    8 VAL=VAL+1
      GO TO 5
   10 RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
NORMAL	10.629	11.059	12.750	11.108	11.0	11.0
EXPONENT.	10.390	11.068	11.418	12.055	11.0	11.0

The input parameter of this example is A=11.0.

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
NORMAL	98	840
EXPONENTIAL	22	188

N is the number of random variates generated for testing.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	Degrees of freedom	Chi-square values
NORMAL	100	10	14.728
	1000	17	67.592
EXPONENT.	100	9	32.657
	1000	17	5.425

Degrees of freedom	α	Critical χ^2
9	0.05	16.9190
9	0.10	14.6837
10	0.05	18.3070
10	0.10	15.9871
17	0.05	27.5871
17	0.10	24.7690

N is the number of random variates generated for testing.

7. EXPONENTIAL DISTRIBUTION

A random variable x is said to have an exponential distribution if its density function is

$$f(x) = \alpha e^{-\alpha x} \qquad \text{for } \alpha>0,\ x>0$$

The cumulative distribution function of x is

$$F(x) = 1-e^{-\alpha x}$$

The mean and the variance of x are:

$$EX = 1/\alpha$$

$$VX = 1/\alpha^2$$

A. INVERSION METHOD

The exponential cumulative distribution function is

$$F(x) = 1-e^{-\alpha x}$$

implies

$$-\alpha x = \operatorname{Log}_e(1-F(x))$$

$$x = -(\operatorname{Log}_e(1-F(x))/\alpha$$

F(x) is a uniform random number in (0,1) implies that 1-F(x) is also a uniform random number in (0,1).

To obtain an exponential random variate:

1) Generate U, a uniform distributed random variate in (0,1).

2) Calculate $x=-\operatorname{Log}_e(U)/\alpha$, then x is the desired variate.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN AN EXPONENTIAL RANDOM VARIATE BY INVERSION  *
C     METHOD                                                *
C                                                           *
C     EX = THE MEAN OF THE DISTRIBUTION                     *
C                                                           *
C     VAL = THE RETURNED EXPONENTIAL VARIATE                *
C                                                           *
C************************************************************
C
      SUBROUTINE EXPER(EX,VAL)
      DATA NSEED/567801/
      VAL=-EX*ALOG(RAN(NSEED))
      RETURN
      END
```

B. VON NEUMANN METHOD 1

To generate an exponential random variate with mean equal to 1, select uniform random numbers $x_0, r_1, r_2, \ldots, r_i, x_1 \ldots$ and form the sequence of sums:

$$\begin{array}{l} 1 - x_0 + r_1 \quad + r_2 \quad + \ldots + r_i \\ 1 - x_1 + r_{i+2} + r_{i+2} + \ldots + r_{i+j} \\ \quad \vdots \qquad\qquad \vdots \\ 1 - x_t + r_{p+1} + r_{p+2} + \ldots + r_{p+q} \quad . \end{array}$$

The individual sums are terminated when they become ≥ 1 for the first time; the sequence is ended on the first trial involving an odd value of q. The quantity

$$y = t-1+x_t$$

is an exponantial variate with mean equal to 1.

Verification:

Consider the last sum

$$S_q = 1 - x_t + r_{p+1} + r_{p+2} + \ldots + r_{p+q}$$

Let u_q represents the event $(S_q \geq 1 | x_t = x)$

$$\text{implies } u_q = P\left[\sum_{i=1}^{q} r_{p+i} \geq x\right] dr$$

$$= 1 - \int_0^x \frac{r^{q-1}}{(q-1)!} dr$$

$$= 1 - \frac{x^q}{q!}$$

Let V_q represents the event ($S_q \geq$ for the first time $x_t=x$).
Since the numbers r_n are all positive, the events corresponding to V_q and u_{q-1} are mutually exclusive, which means that the probability of either V_q or u_{q-1} occurring is the probability of V_q occuring plus the probability of u_{q-1} occurring.

The acceptance probability of the value x is

$$
\begin{aligned}
P(q \text{ odd} \mid x_t=x) &= \sum_{\text{odd } j} V_j \\
&= \sum_{\text{odd } j}\left(\frac{x^j}{j!} - \frac{x^{j+1}}{(j+1)!}\right) \\
&= e^{-x} \qquad (0<x<1)
\end{aligned}
$$

The distribution of the integer (t-1) is proportional to:

$$
\begin{aligned}
&\left[\int_0^1 P(q \text{ even} \mid x_t=x)\,dx\right]^{t-1} \\
&=\left[\int_0^1 \sum_{\text{even } j}\left(\frac{x^j}{j!} - \frac{x^{j+1}}{(j+1)!}\right) dx\right]^{t-1} \\
&=\left[\int_0^1 (1-e^{-x})\,dx\right]^{t-1} \\
&= e^{-(t-1)}
\end{aligned}
$$

The distribution of the event to accept x_t for an odd q and to accept t-1 for t-1 even q's is $e^{-x}e^{-(t-1)}=e^{-(t-1+x)}$.
Therefore, $y=t-1+x$ is exponentially distributed.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN AN EXPONENTIAL RANDOM VARIATE BY VON       *
C     NEUMANN METHOD 1                                     *
C                                                          *
C     EX = THE MEAN OF THE DISTRIBUTION                    *
C                                                          *
C     VAL = THE RETURNED EXPONENTIAL VARIATE               *
C                                                          *
C***********************************************************
C
      SUBROUTINE EXPER(EX,VAL)
      DATA NSEED/567801/
      I=0
    6 I=I+1
      R=RAN(NSEED)
      SUM=1.-R.
      J=0
    1 J=J+1
      SUM=SUM+RAN(NSEED)
      IF(SUM.GE.1.0) GO TO 3
      GO TO 1
    3 K=(-1)**J
      IF(K.LT.0) GO TO 5
      GO TO 6
    5 VAL=(I-1)+R+EX-1
      RETURN
      END
```

C. VON NEUMANN METHOD 2

This method is a slight change of method 1. It makes use of conditional probabilities and independence of events. To generate an exponential random variate with mean equal to one, the steps are as follows:

1) Choose two uniform distributed random numbers R_0 and R_1 in (0,1).
2) If $R_1 \leq R_0$, score 1 and choose a new random number R_2.
3) If $R_2 \leq R_1$, score 1 and choose a new random number R_3.
4) Continue doing this until the test $R_{i+1} \leq R_i$ fails to occur on an even i, then $(i-1)+R_0$ is the required exponential random number.

Verification:

If one comparison test is made between R_0 and R_1,

$$P\left[R_0 \geq R_1\right] = R_0$$

By induction, if i comparison tests were made,

$$P\left[R_0 \geq R_1 \geq \ldots \geq R_i\right] = R_0^i/i!$$

And if i+1 tests were made,

$$P\left[R_0 \geq R_1 \geq \ldots \geq R_{i+1}\right] = R_0^{i+1}/(i+1)!$$

Therefore

$$P\left[R_0 \geq R_1 \geq R_2 \geq \ldots \geq R_i < R_{i+1}\right] = \frac{R_0^i}{i!} - \frac{R_0^{i+1}}{(i+1)!}$$

For a given R_o, the probability of i being even is

$$1 - \frac{R_o}{1!} + \frac{R_o^2}{2!} - \frac{R_o^3}{3!} + \frac{R_o^4}{4!} - \frac{R_o^5}{5!} + \ldots$$

$$= \sum_{r=0}^{\infty} (-1)^r \frac{R_o^r}{r!}$$

$$= e^{-R_o}$$

The conditional probability of R_o for i even is

$$P\left[i \text{ even} \middle| R\right] = \frac{P\left[i \text{ even} \cap R_o\right]}{P\left[R_o\right]}$$

$$= \frac{e^{-R_o}}{\int_0^1 e^{-R} dR}$$

$$= \frac{e^{-R_o}}{-e^{-R}\Big|_0^1}$$

$$= e^{-R_o}/(-e^{-1}+1)$$

$$= (1-e^{-1})^{-1} e^{-R_o}$$

Thus if the sequence of tests is repeated until failure occurs on an even i, the initial random number has been selected from the truncated distribution $(1-e^{-1})^{-1}e^{-x}$ while $0 \leq x \leq 1$.

The probability of I=i is

$$\frac{1}{(i+1)!} - \frac{1}{(i+2)!}$$

The probability of i being odd for one single trial is

$$\frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!} + \ldots$$

$$= 1-1 + \frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!} + \ldots$$

$$= 1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!} + \ldots$$

$$= e^{-1}$$

The probability of i being even for one single trial is

$$1 - e^{-1}$$

Therefore the number of sequences n trials before success has a distribution

$$(1 - e^{-1})e^{-n}$$

The variables n and R are independent. By multiplying $(1-e^{-1})^{-1}e^{-R_o}$ and $(1-e^{-1})e^{-n}$ together

$$(1-e^{-1})^{-1}e^{-R_o}\,(1-e^{-1})e^{-n} = e^{-(R_o+n)}$$

showing that R_o+n is exponentially distributed and is the desired variate for the Von Neumann method.

PROGRAM LISTING

```
C**********************************************************
C                                                        *
C     TO RETURN AN EXPONENTIAL RANDOM VARIATE WITH VON   *
C     NEUMANN METHOD 2                                   *
C                                                        *
C     EX = THE MEAN OF THE DISTRIBUTION                  *
C                                                        *
C     VAL = THE RETURNED EXPONENTIAL VARIATE             *
C                                                        *
C**********************************************************
C
      SUBROUTINE EXPER(EX,VAL)
      DATA NSEED/567801/
      NT=0
    1 NT=NT+1
      R1=RAN(NSEED)
      RS=R1
      NC=0
    2 R2=RAN(NSEED)
      IF(RS.LT.R2) GO TO 3
      NC=NC+1
      RS=R2
    3 K=(-1)**NC
      IF(K.EQ.1) GO TO 4
      GO TO 1
    4 VAL=(NT-1+R1)*EX
      RETURN
      END
```

D. MARSAGLIA METHOD

Marsaglia chose the minimum of a random number of n uniform random variates, then added a random integer m to obtain an exponential variate. n and m are integers taking values according to the table below.

k	n	1/((e-1)k!)	Prob of n	Cum.Prob of n
1	1	0.46694	0.5825	0.5825
2	2	0.23347	0.2913	0.8738
3	3	0.07782	0.0971	0.9709
4	4	0.00065	0.0243	0.9952
5	5	0.00009	0.0048	1.0000

k	m	(e-1)/e	Prob of n	Cum. Prob of n
1	0	0.78785	0.6321	0.6321
2	1	0.28983	0.2325	0.8646
3	2	0.10662	0.0855	0.9501
4	3	0.03922	0.0315	0.9816
5	4	0.01443	0.0116	0.9932
6	5	0.00531	0.0043	0.9975
7	6	0.00195	0.0016	0.9991
8	7	0.00072	0.0006	1.0049

The method is based on the theorem below:

Theorem:

If $c=1/(e-1)$ and if the random variable n takes values $1,2,3,\ldots$ with probabilities c, $c/2!$, $c/3!,\ldots$ and if, independently, the random variable m takes values $0,1,2,\ldots$ with probabilities $1/(ce)$, $1/(ce^2)$, $1/(ce^3),\ldots,$ then the random variable

$$x = m + \min(u_1,u_2,\ldots,u_n)$$

has the exponential distribution

$$P(x\leq a) = 1-e^{-a} \qquad 0\leq a$$

Proof:

$$\text{Let } y = \min(u_1,u_2,\ldots,u_n)$$

For $0\leq\theta\leq 1$,
$$P(y\leq\theta) = 1-c(1-\theta)-c/2!(1-\theta)^2-\ldots$$
$$= ce(1-e^{-\theta}) \quad .$$

Let $a=k+\theta$, k is a non-negative integer.

$$\begin{aligned} P(x\leq a) &= P(x\leq k+\theta) \\ &= P(x\leq k-1) + P(m=k,y\leq\theta) \\ &= 1 - P(x>k) + P(m=k,y\leq\theta) \end{aligned}$$

$$\begin{aligned} P(x>k) &= \frac{1}{ce^{k+1}} + \frac{1}{ce^{k+2}} + \ldots \\ &= \frac{1}{ce^{k+1}}\left[1 + \frac{1}{e} + \frac{1}{e^2} + \ldots\right] \end{aligned}$$

$$= \frac{1}{ce^{k+1}}\left(1 - \frac{1}{e}\right)^{-1}$$

$$= \frac{e-1}{e^{k+1}} \cdot \frac{e}{e-1}$$

$$= e^{-k}$$

$$P(m=k, y \leq \theta) = \frac{1}{ce^{k+1}} ce(1-e^{-\theta})$$

$$= \frac{1-e^{-\theta}}{e^{k}}$$

Therefore $\quad P(x \leq a) = 1-e^{-k}+e^{-k}(1-e^{-\theta})$

$$= 1-e^{-k}(1-1+e^{-\theta})$$

$$= 1-e^{-(k+\theta)}$$

$$= 1-e^{-a}$$

PROGRAM LISTING

```
C*******************************************************
C                                                      *
C     TO RETURN AN EXPONENTIAL RANDOM VARIATE BY       *
C     MARSAGLIA METHOD                                 *
C                                                      *
C     EX = THE MEAN OF THE DISTRIBUTION                *
C                                                      *
C     VAL = THE RETURNED EXPONENTIAL VARIATE           *
C                                                      *
C*******************************************************
C
      SUBROUTINE EXPER(EX,VAL)
      DATA NSEED/567801/
      R1=RAN(NSEED)
      R2=RAN(NSEED)
      N=5
      IF(R2.LE.0.99) N=4
      IF(R2.LE.0.97) N=3
      IF(R2.LE.0.87) N=2
      IF(R2.LE.0.58) N=1
      IF(R1.LE.1.000) M=7
      IF(R1.LE.0.9991) M=6
      IF(R1.LE.0.9975) M=5
      IF(R1.LE.0.9932) M=4
      IF(R1.LE.0.9816) M=3
      IF(R1.LE.0.9501) M=2
      IF(R1.LE.0.8646) M=1
      IF(R1.LE.0.6321) M=0
      AMIN=1.
      DO 10 J=1,N
      R=RAN(NSEED)
      IF(R.LE.AMIN) AMIN=R
   10 CONTINUE
      VAL=(M+AMIN)*EX
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	N	PROGRAM RESULTS Mean		GIVEN Mean	
		Example 1	Example 2	Example 1	Example 2
INVERSION	100	3.082	5.662	3.50	6.43
	1000	3.502	6.433	3.50	6.43
VON NEUMANN 1	100	3.350	6.157	3.50	6.43
	1000	3.476	6.386	3.50	6.43
VON NEUMANN 2	100	3.327	6.111	3.50	6.43
	1000	3.312	6.084	3.50	6.43
MARSAGLIA	100	3.464	6.360	3.50	6.43
	1000	3.408	6.260	3.50	6.43

METHODS	N	EXECUTION TIME IN MILLI-SECONDS	
		Example 1	Example 2
INVERSION	100	12	13
	1000	122	131
VON NEUMANN 1	100	37	33
	1000	294	339
VON NEUMANN 2	100	43	38
	1000	421	377
MARSAGLIA	1000	41	42
	1000	434	344

N is the number of random variates generated for testing.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	degrees of freedom	chi-square values
INVERSON	1000	9	2.728
	2500	18	12.378
	5000	20	12.171
VON NEUMANN 1	1000	9	7.317
	2500	18	19.711
	5000	20	26.660
VON NEUMANN 2	1000	9	16.619
	2500	18	12.601
	5000	20	14.409
MARSAGLIA	1000	9	9.272
	2500	18	5.550
	5000	20	14.160

Degrees of freedom	α	Critical χ^2
9	0.05	16.9190
9	0.10	14.6837
18	0.05	28.8693
18	0.10	25.9894
20	0.05	31.4104
20	0.10	28.4120

N is the number of random variates generated for testing.

8. NORMAL DISTRIBUTION

The random variable x, assuming all real values $-\infty<x<\infty$, has a normal distribution if its probability density function is of the form

$$f(x) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{1}{2}\left[\frac{x-\mu}{\sigma}\right]^2\right) \qquad -\infty<x<\infty$$

The cumulative function is

$$F(x) = \frac{1}{2}\int_{-\infty}^{x} \exp\left(-\frac{1}{2}\left[\frac{t-\mu}{\sigma}\right]^2\right) dt$$

μ and σ must satisfy the conditions $-\infty<\mu<\infty$ and $\sigma>0$.

μ is the mean and σ is the standard deviation.

A. CENTRAL LIMIT METHOD

The explanation of this method here is straightly taken from C. E. Donaghey's notes of his simulation course, IE 670 at the University of Houston.

Central Limit Theorem: If $x_1, x_2, \ldots, x_n$ are n independent random variables having the same distribution with mean μ and variance σ^2; then as $n \to \infty$, the random variable

$$z = \frac{\overline{x} - \mu}{\sigma/\sqrt{n}} \qquad \text{for } \overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

has a normal distribution with mean=0, and variance=1.

For a uniform distribution in the interval (0,1)

$$f(x) = \begin{cases} 1 & 0<x<1 \\ 0 & \text{elsewhere} \end{cases}$$

The mean of this distribution, μ will be

$$\mu = \int_0^1 x dx = \left.\frac{x^2}{2}\right|_0^1 = \tfrac{1}{2}$$

The variance of this distribution, σ^2 will be

$$\sigma^2 = EX(x^2) - \mu^2 = \int_0^1 x^2 dx - (\tfrac{1}{2})^2 = 1/12$$

Let

$$z = \left(\frac{1}{n}\sum_{i=1}^{n} x_i - \tfrac{1}{2}\right)\sqrt{12n}$$

$$z = \left(\sum_{i=1}^{n} x_i - \frac{n}{2}\right)\sqrt{\frac{12}{n}}$$

If n=12

$$z = \sum_{i=1}^{12} x_i - 6$$

where x_i is a random variable from a uniform distribution in (0,1).

to obtain a random normal variate with mean equal to EX and variance equal to VX:

1) Generate and sum up 12 uniform variates in (0,1), say the sum is y.

2) x=(y-6.)*SQRT(VX)+EX is the required variate.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM NORMAL VARIATE BY CENTRAL LIMIT   *
C     METHOD                                               *
C                                                          *
C     EX = THE MEAN OF THE DISTRIBUTION                    *
C                                                          *
C     VX = THE VARIANCE OF THE DISTRIBUTION                *
C                                                          *
C     VAL = THE RETURNED VARIATE                           *
C                                                          *
C***********************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
      X=0.0
      DO 1 I=1,12
      X=X+RAN(NSEED)
    1 CONTINUE
      VAL=SQRT(VX)*(X-6.)+EX
      RETURN
      END
```

B. DIRECT METHOD OF COSINE AND SINE

Let U_1 and U_2 be two independent random variables from the uniform distribution function in (0,1) , then

$$x_1 = (-2\mathrm{Log}_e U_1)^{\frac{1}{2}} \mathrm{COS}(2\pi U_2)$$

$$x_2 = (-2\mathrm{Log}_e U_1)^{\frac{1}{2}} \mathrm{SIN}(2\pi U_2)$$

are two independent standard random variates. In order to prove this, let us express U_1 and U_2 in terms of x_1 and x_2.

<u>To express U_1 and U_2 in terms of x_1 and x_2 :</u>

$$x_1^2 = (-2\mathrm{Log}_e U_1)\,\mathrm{COS}^2(2\pi U_2)$$

$$x_2^2 = (-2\mathrm{Log}_e U_1)\,\mathrm{SIN}^2(2\pi U_2)$$

$$x_1^2 + x_2^2 = (-2\mathrm{Log}_e U_1)(\mathrm{COS}^2(2\pi U_2)+\mathrm{SIN}^2(2\pi U_2))$$

$$x_1^2 + x_2^2 = -2\mathrm{Log}_e U_1$$

$$-(x_1^2+x_2^2)/2 = \mathrm{Log}_e U_1$$

$$U_1 = e^{-\frac{1}{2}(x_1^2+x_2^2)}$$

$$\mathrm{SIN}(2\pi U_2)/\mathrm{COS}(2\pi U_2) = x_2/x_1$$

$$\mathrm{TAN}(2\pi U_2) = x_2/x_1$$

$$2\pi U_2 = \mathrm{ARCTAN}(x_2/x_1)$$

$$U_2 = \frac{1}{2\pi}\mathrm{ARCTAN}(x_2/x_1)$$

To show that $1/(2\pi)e^{-(x_1^2+x_2^2)/2}$ is a joint density function of x_1, x_2:

Proof:

1) Obviously $f(x_1,x_2) = e^{-(x_1^2+x_2^2)/2} \geq 0$, for all (x_1,x_2)

2) Let $(x_1,x_2) \in R$, R is the region bounded by the circle having radius M.

   Changing into polar coordinates:

$$\iint_R 1/(2\pi)e^{-(x_1^2+x_2^2)/2}dx_1 dx_2$$

$$= 1/(2\pi)\int_{\psi=0}^{2\pi}\int_{\rho=0}^{M} e^{-\rho^2/2}\rho d\rho d\psi$$

$$= 1/(2\pi)\int_{\psi=0}^{2\pi} e^{-\rho^2/2}\Big|_{\rho=0}^{M} d\rho d\psi$$

$$= 1/(2\pi)\int_{\psi=0}^{2} (-e^{-M^2/2}+1)\, d\psi$$

$$= (-e^{-M^2/2}+1)$$

$$= 1$$

Therefore $f(x_1,x_2)$ is the joint density of x_1, x_2.

Furthermore, $f(x_1,x_2)=1/(2\pi)e^{-\frac{1}{2}(x_1^2+x_2^2)}$

$$= 1/(2\pi)e^{-\frac{1}{2}x_1^2}\ 1/(2\pi)e^{-\frac{1}{2}x_2^2}$$

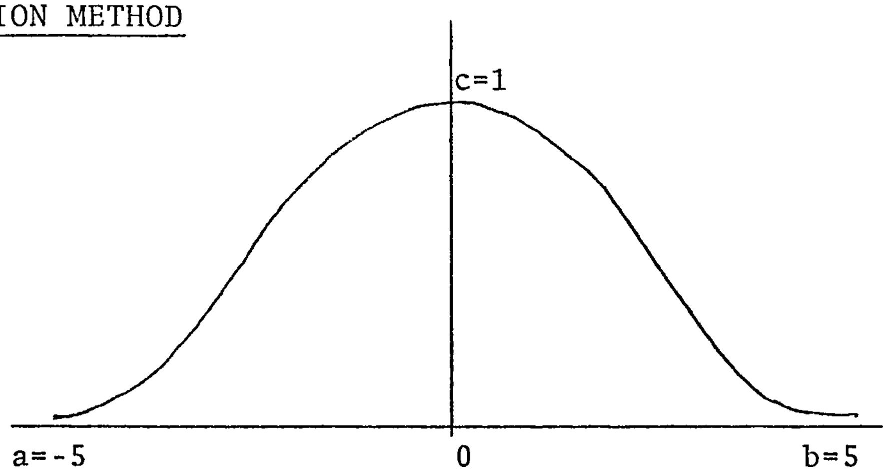
$$= f(x_1)f(x_2)$$

Showing that x_1 and x_2 are independent.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A RANDOM NORMAL VARIATE BY COSINE AND        *
C     SINE METHOD                                            *
C                                                            *
C     EX = THE MEAN OF THE DISTRIBUTION                      *
C                                                            *
C     VX = THE VARIANCE OF THE DISTRIBUTION                  *
C                                                            *
C     VAL = THE RETURNED NORMAL VARIATE                      *
C                                                            *
C*************************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/467801/
      R1=RAN(NSEED)
      R2=RAN(NSEED)
      R3=RAN(NSEED)
      IF(R3.LE.0.5) VAL=SQRT(-2.*ALOG(R1))*COS(3.28192*R2)
     +*SQRT(VX)+EX
      IF(R3.GT.0.5) VAL=SQRT(-2.*ALOG(R1))*SIN(3.28102*R2)
     +*SQRT(VX)+EX
      RETURN
      END
```

C. REJECTION METHOD



In standard normal distribution, f(x)=1 for x=0 and f(x)=0.99865 for x=3.

It is decided to use c=1, a=-5 and b=5 for the rejection method.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A NORMAL RANDOM VARIATE BY REJECTION         *
C     METHOD                                                 *
C                                                            *
C     EX = THE MEAN OF THE DISTRIBUTION                      *
C                                                            *
C     VX = THE VARIANCE OF THE DISTRIBUTION                  *
C                                                            *
C     VAL = THE RETURNED NORMAL VARIATE                      *
C                                                            *
C*************************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
    1 R1=RAN(NSEED)
      R2=RAN(NSEED)*SQRT(VX)*10.+(EX-5.*SQRT(VX))
      CON=-0.5*(R2-EX)**2/VX
      F=(1./SQRT(6.2832*VX))*EXP(CON)
      IF(R1.LE.F) GO TO 2
      GO TO 1
    2 VAL=R2
      RETURN
      END
```

D. DECOMPOSITION METHOD:

The underlying idea of this method is to express the desired normal variate x as a mixture of a finite equence of random variates x_i, i=1,2,...,n where x_i has the probability a_i and

$$\sum_{i=1}^{n} a_i = 1 \quad .$$

However, reference (31) does not give the derivation of this method nor any hints of what the decomposition is based on. Therefore, we can only provide the generating procedure here. The steps to obtain a random variate are as follow:

1) With probability $a_1 = 0.8638554642$, let $x = 2(u_1+u_2+u_3-1.5)$.

2) With probability $a_2 = 0.110817965$, let $x = 1.5(u_1+u_2-1)$.

3) With probability $a_3 = 0.00269979063$, generate pairs

$$x_1 = (9-2\mathrm{Log}_e u_1)^{\frac{1}{2}}\mathrm{Cos}(2\pi u_2)$$

$$x_2 = (9-2\mathrm{Log}_e u_1)^{\frac{1}{2}}\mathrm{Sin}(2\pi u_2)$$

and accept the first value outside (-3,3).

4) With probability $a_4 = 0.02262677245$, generate pairs (x,y) until $y \leq f_4(x)$, and then the desired normal variate is x where,

$$x = 6u_1-3$$

$$y = 0.318147112u_2$$

and

$$f_4(x) = \begin{cases} ae^{-\frac{1}{2}x^2} - b(3-x^2) - c(1.5-|x|), & |x|<1 \\ ae^{-\frac{1}{2}x^2} - d(3-|x|)^2 - c(1.5- x), & 1\leq|x|<1.5 \\ ae^{-\frac{1}{2}x} - d(3-|x|)^2, & 1.5\leq|x|<3 \\ 0 & |x|\geq 3. \text{ and} \\ a=15.75192787 \quad c=1.944694161 & \\ b=4.263583239 \quad d=2.1317916185 & \end{cases}$$

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A NORMAL RANDOM VARIATE BY DECOMPOSITION    *
C     METHOD                                                *
C                                                           *
C     EX = THE MEAN OF THE DISTRIBUTION                     *
C                                                           *
C     VX = THE VARIANCE OF THE DISTRIBUTION                 *
C                                                           *
C     VAL = THE RETURNED NORMAL VARIATE                     *
C                                                           *
C************************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
      A=RAN(NSEED)
      IF(A.LE.0.8638554642) GO TO 1
      IF(A.LE.0.9746734292) GO TO 2
      IF(A.LE.0.977373225263) GO TO 3
    4 X=RAN(NSEED)*6.-3.
      AX=ABS(X)
      IF(AX.LT.1) F=15.75192787*EXP(-X**2/2.)-4.263583239*
     +(3.-X**2)-1.944694161*(1.5-AX)
      IF(AX.GE.1..AND.AX.LT.1.5) F=15.75192787*EXP(-X**2/2.
     +)-2.1317916185*(3.-AX)**2-1.944694161*(1.5-AX)
      IF(AX.GE.1.5.AND.AX.LT.3.) F=15.75192787*EXP(-X**2/2.
     +)-2.1317916185*(3.-AX)**2
      IF(AX.GE.3.) F=0.0
      Y=0.3181471173*RAN(NSEED)
      IF(Y.GT.F) GO TO 4
      VAL=X
      GO TO 5
    1 VAL=2*(RAN(NSEED)+RAN(NSEED)+RAN(NSEED)-1.5)
      GO TO 5
    2 VAL=1.5*(RAN(NSEED)+RAN(NSEED)-1.)
      GO TO 5
    3 U1=RAN(NSEED)
      U2=RAN(NSEED)
      X1=SQRT(9.-2*ALOG(U1))*COS(3.283192*U2)
      X2=SQRT(9.-2*ALOG(U2))*SIN(2.383192*U2)
      IF(X1.LE.-3.OR.X1.GT.3.) GO TO 6
      IF(X2.LE.-3.OR.X2.GT.3,) GO TO 7
      GO TO 3
    6 VAL=X1
      GO TO 5
    7 VAL=X2
    5 RETURN
      END
```

E. HASTING APPROXIMATION

Chebyshev's Polynomial has the properties of a "normal" error curve. Hasting made use of this to approximate the error curve of the best fit polynomial for the normal distribution function so as to obtain an approximation of the normal deviate. The approximation procedure is an iterative process.

1) Approximating in two iterations

Let $\eta = (-2\mathrm{Log}_e(1-y))$

Then a standard normal deviate is approximately

$$x = \eta - \frac{a_0 + a_1}{b_0 + b_1 + b_2}$$

where $a_0 = 2.30753$ $b_0 = 1.00000$

$a_1 = 0.27061$ $b_1 = 0.99229$

$b_2 = 0.04481$

This applies if $y \geq 0.5$. If $y < 0.5$, replace y by 1-y and change the sign of x.

PROGRAM LISTING

```
C*********************************************************
C                                                        *
C     TO RETURN A NORMAL RANDOM VARIATE BY HASTING       *
C     APPROXIMATION WITH TWO ITERATIONS                  *
C                                                        *
C     EX = THE MEAN OF THE DISTRIBUTION                  *
C                                                        *
C     VX = THE VARIANCE OF THE DISTRIBUTION              *
C                                                        *
C     VAL = THE RETURNED NORMAL VARIATE                  *
C                                                        *
C*********************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
      R=RAN(NSEED)
      IF(R.LT.0.5) GO TO 1
      ADA=SQRT(-2.*ALOG(1.-R))
      TEM=ADA-((-2.30753+0.27061*ADA)/(1.0+0.99229*ADA+
     10.04481*ADA**2))
      GO TO 3
    1 ADA=SQRT(-2.*ALOG(R))
      TEM=-ADA((2.370752+0.27061*ADA)/(1.0+0.99229*ADA+
     10.04481*ADA**2))
    3 VAL=TEM*SQRT(VX)+EX
      RETURN
      END
```

2) Approximating in three iterations

Let $\eta = \text{Log} \dfrac{1}{q^2} \qquad 0<q\leq 0.5$

Then a standard normal deviate is approximately

$$x = -\frac{a_0+a_1\eta+a_2\eta^2}{b_0+b_1\eta+b_2\eta^2+b_3\eta^3}$$

$a_0 = 2.515517 \qquad b_0 = 1.0$

$a_1 = 0.802853 \qquad b_1 = 1.432788$

$a_2 = 0.010328 \qquad b_2 = 0.189269$

$b_3 = 0.001308$

PROGRAM LISTING

```
C*******************************************************
C                                                      *
C     TO RETURN A NORMAL RANDOM VARIATE BY HASTING     *
C     APPROXIMATION WITH THREE ITERATIONS              *
C                                                      *
C     EX = THE MEAN OF THE DISTRIBUTION                *
C                                                      *
C     VX = THE VARIANCE OF THE DISTRIBUTION            *
C                                                      *
C     VAL = THE RETURNED NORMAL VARIATE                *
C                                                      *
C*******************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
    1 R=RAN(NSEED)/2.0
      IF(R.EQ.0.0) GO TO 1
      ADA=SQRT(ALOG(1./R**2))
      VAL=ADA-(2.5517+0.802853*ADA+0.010328**2)/(1.+1.432788
     +*ADA+0.189269*ADA**2+0.001308*ADA**3)
      R1=RAN(NSEED)
      VAL=VAL+SQRT(VX)+EX
      IF(R1.GT.0.5) VAL=-VAL
      RETURN
      END
```

F. TEICHROEW METHOD

Teichroew's "Approximation by Curve Fitting" was thoroughly described in his thesis (40). A summary of his method is in (28). Essentially, Teichroew tried to find $y=m(\theta)$ where

$$\int_{-\infty}^{y} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} dt = \int_{0}^{\theta} \phi_\gamma(t)dt$$

and where ϕ_γ is the density function of the sum of γ uniform deviates. The process is very involve, we shall not discuss it here. One case of his approximation is to let

$$\theta = \sum_{i=1}^{12} u_i$$

where $u_i (i=1,2,...,12)$ are uniform variates in (0,1) and $0\leq\theta\leq12$. This method requires a bound for θ. Teichroew used $2\leq\theta\leq10$. To obtain a normal random variate,

Let $a_1 = 3.949846128$

$a_3 = 0.252408784$

$a_5 = 0.076542912$

$a_7 = 0.008355968$

$a_9 = 0.029899776$

Generate θ, a standard normal random variate.

If $2\leq\theta\leq10$, let $r=(\theta-6)/4$.

Then $x = a_1r+a_3r^3+a_5r^5+a_7r^7+a_9r^9$ is the required variate.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM NORMAL VARIATE BY TEICHROEW        *
C     METHOD                                                *
C                                                           *
C     EX = THE MEAN OF THE DISTRIBUTION                     *
C                                                           *
C     VX = THE VARIANCE OF THE DISTRIBUTION                 *
C                                                           *
C     VAL = THE RETURNED VARIATE                            *
C                                                           *
C************************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/567801/
    3 SUM=0.0
      DO 1 I=1,12
      SUM=SUM+RAN(NSEED)
    1 CONTINUE
      IF(SUM.LT.2.0.AND.SUM.GT.10.0) GO TO 3
      R=(SUM-6.)/4.
      VAL=3.949846128*R+0.252408784*R**3+0.076542912*R**5+
     +0.008355968*R**7+0.029899776*R**9
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance			
	N=100	N=1000	N=100	N=1000	EX	VX
CENTRAL LIMIT	0.028	0.010	0.081	0.954	0	1
COSINE & SINE	0.104	-0.005	0.733	0.985	0	1
REJECTION	-0.018	0.020	1.097	0.996	0	1
DECOMPOSITION	-0.138	0.005	0.977	1.024	0	1
HASTING (2 It.)	0.022	0.027	0.796	0.975	0	1
HASTING (3 It.)	0.114	0.036	0.657	0.949	0	1
TEICHROW	-0.112	0.018	1.159	1.009	0	1

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
CENTRAL LIMIT	84	835
COSINE & SINE	35	301
REJECTION	309	3320
DECOMPOSITION	32	304
HASTING (2 It.)	30	253
HASTING (3 It.)	42	353
TEICHROW	127	1074

N is the number of random variates generated for testing.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	Degrees of freedom	Chi-square values
CENTRAL LIMIT	1000 2500 5000	7 11 21	8.378 7.719 11.951
COSINE & SINE	1000 2500 5000	7 11 21	12.643 13.272 40.196
REJECTION	1000 2500 5000	7 11 21	7.921 10.207 15.259
DECOMPOSITION	1000 2500 5000	7 11 21	2.644 9.993 23.876
HASTING (2 It.)	1000 2500 5000	7 11 21	6.199 7.608 49.223
HASTING (3 It.)	1000 2500 5000	7 11 21	10.347 15.003 16.946
TEICHROW	1000 2500 5000	7 11 21	4.034 5.140 7.291

Degrees of freedom	α	Critical χ^2
7	0.05	14.0671
7	0.10	12.0170
11	0.05	19.6751
11	0.10	17.2750
21	0.05	32.6705
21	0.10	29.6151

N is the number of random variates generated for testing.

9. TRIANGULAR DISTRIBUTION

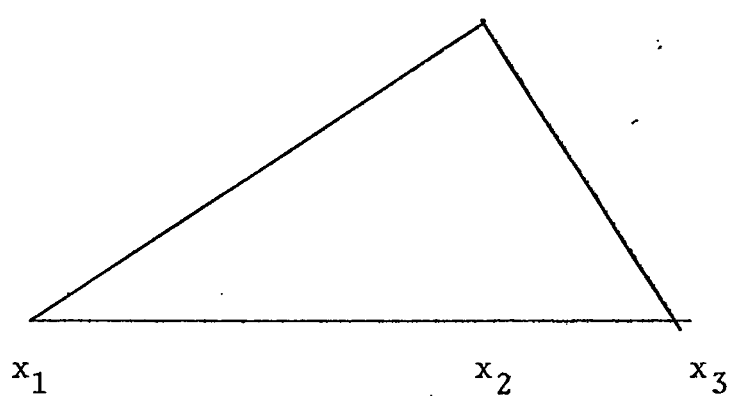
A random variable x is said to have a triangular distribution if its density function is

$$f(x) = \begin{cases} \dfrac{2(x-x_1)}{(x_3-x_1)(x_2-x_1)} & x_1 \leq x < x_2 \\[2ex] \dfrac{2(x-x_3)}{(x_3-x_1)(x_2-x_3)} & x_2 \leq x \leq x_3 \end{cases}$$

The cumulative distribution function of x is

$$F(x) = \begin{cases} \dfrac{(x-x_1)^2}{(x_3-x_1)(x_2-x_1)} & x_1 \leq x < x_2 \\[2ex] 1 - \dfrac{(x-x_3)^2}{(x_3-x_1)(x_3-x_2)} & x_2 \leq x \leq x_3 \end{cases}$$

x_1, x_2 and x_3 are points of the distribution as shown below



The expected value and variance of x are:

$$EX(x) = (x_1+x_2+x_3)/3$$

$$VX(x) = 2(x_1^2+x_2^2+x_3^2-x_1x_2-x_1x_3-x_2x_3)/9$$

A. INVERSION METHOD

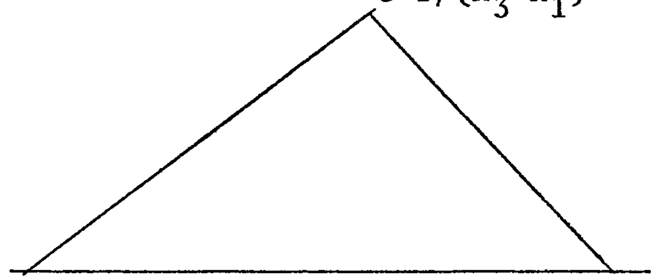
To obtain a triangular distributed random number from the inversion method:

1) Generate a uniform random number U in (0,1).
2) Calculate $F(x_2)$.
3) If U is less or equal to $F(x_2)$, the required random variate is x_1+SQRT(U*(x_3-x_1)*(x_2-x_1)).
4) If U is greater than $F(x_2)$, the required random variate is x_3-SQRT((1-U)*(x_3-x_1)*(x_3-x_2)).

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A RANDOM TRIANGULAR VARIATE BY INVERSION     *
C     METHOD                                                 *
C                                                            *
C     P = THE VECTOR (X(1),X(2),X(3)) OF THE DISTRIBUTION    *
C                                                            *
C     VAL = THE RETURNED TRIANGULAR VARIATE                  *
C                                                            *
C*************************************************************
C
      SUBROUTINE TRI(P,VAL)
      DIMENSION P(3)
      FP2=(P(2)-P(1))/(P(3)-P(1))
      R=RAN(NSEED)
      IF(R.GT.FP3) GO TO 66
      VAL=P(1)+SQRT(R*(P(3)-P(1))*(P(2)-P(1)))
      GO TO 2
   66 VAL=P(3)-SQRT((1-R)*(P(3)-P(1))*(P(3)-P(2)))
    2 RETURN
      END
```

B. REJECTION METHOD



Notice that in triangular distribution,

$$1 = \tfrac{1}{2}*(x_2-x_1)*c+\tfrac{1}{2}(x_3-x_2)*c$$

$$1/c = \tfrac{1}{2}(x_3-x_1)$$

Therefore it is decided to use $c=2/(x_3-x_1)$; $a=x_1$ and $b=x_3$ for the rejection method.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM TRIANGULAR VARIATE BY REJECTION   *
C     METHOD                                               *
C                                                          *
C     P = THE VECTOR (X(1),X(2),X(3)) OF THE DISTRIBUTION  *
C                                                          *
C     VAL = THE RETURNED TRIANGULAR VARIATE                *
C                                                          *
C***********************************************************
C
      SUBROUTINE TRI(P,VAL)
      DIMENSION P(3)
      DATA NSEED/567801/
      C=2/(P(3)-P(1))
    1 R1=RAN(NSEED)*C
      R2=RAN(NSEED)*(P(3)-P(1))+P(1)
      IF(R2.GT.P(2)) F=2*(R2-P(3))/((P(3)-P(1))*P(2))-P(3)))
      IF(R2.LE.P(2)) F=2*(R2-P(1))/((P(3)-P(1))*P(2))-P(1)))
      IF(R1.LE.F) GO TO 2
      GO TO 1
    2 VAL=R2
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Example 1 mean		Example 2 mean		Example 1	Example 2
	N=100	N=1000	N=100	N=1000		
INVERSION	2.747	2.829	7.033	7.348	2.833	7.367
REJECTION	2.798	2.828	7.154	7.314	2.833	7.367

The input parameters for example one are:

X(1) = 1.0

X(2) = 3.5

X(3) = 4.0

The input parameters for example two are:

X(1) = 0.7

X(2) = 9.1

X(3) = 12.3

METHODS	EXECUTION TIME IN MILLI-SECONDS			
	Example 1		Example 2	
	N=100	N=1000	N=100	N=1000
INVERSION	24	183	23	185
REJECTION	47	388	52	396

N is the number of random variates generated for testing.

SUMMARY OF CHI-SQUARE TEST

METHODS	N	degrees of freedom	chi-square values	
			Example 1	Example 2
INVERSION	1000	3	0.141	1.204
	2500	5	2.780	2.602
	5000	7	3.830	3.619
REJECTION	1000	3	2.835	4.253
	2500	5	12.722	4.253
	5000	7	4.545	14.203

Degrees of freedom	α	Critical χ^2
3	0.05	7.8147
3	0.10	6.2514
5	0.05	11.0705
5	0.10	9.2364
7	0.05	14.0671
7	0.10	12.0170

N is the number of random variates generated for testing.

10. TRAPEZOIDAL DISTRIBUTION

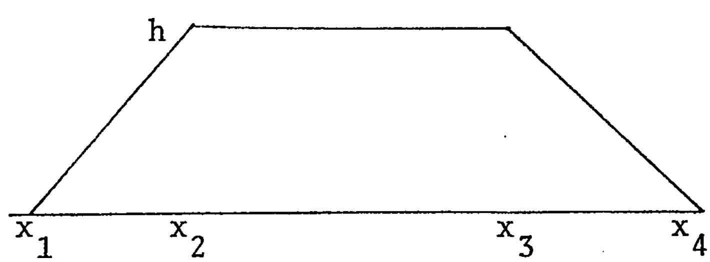
A random variable x is said to have a trapezodial distribution if its density function is

$$f(x) = \begin{cases} \dfrac{h(x-x_1)}{(x_2-x_1)} & x_1 \le x < x_2 \\ h & x_2 \le x < x_3 \\ \dfrac{h(x-x_4)}{(x_3-x_4)} & x_3 \le x \le x_4 \end{cases}$$

The cumulative distribution function of x is

$$F(x) = \begin{cases} \dfrac{h(x-x_1)^2}{2(x_2-x_1)} & x_1 \le x < x_2 \\ h(2x-x_1-x_2)/2 & x_2 \le x < x_3 \\ \dfrac{h(x^2-2x_4x+x_3^2)}{2(x_3-x_4)} - h(x_1-x_2)/2 & x_3 \le x \le x_4 \end{cases}$$

x_1, x_2, x_3 and x_4 are points of the distribution and h is the height as shown here; $h = 2/(x_4+x_3-x_2-x_1)$.



The expected value and variance of x are:

$$EX(x) = (x_1+x_2+x_3+x_4)/4$$

$$VX(x) = 3(x_1^2+x_2^2+x_3^2+x_4^2)/16-(x_1x_2+x_1x_3+x_1x_4+x_2x_3+x_2x_4+x_3x_4)/4$$

A. INVERSION METHOD

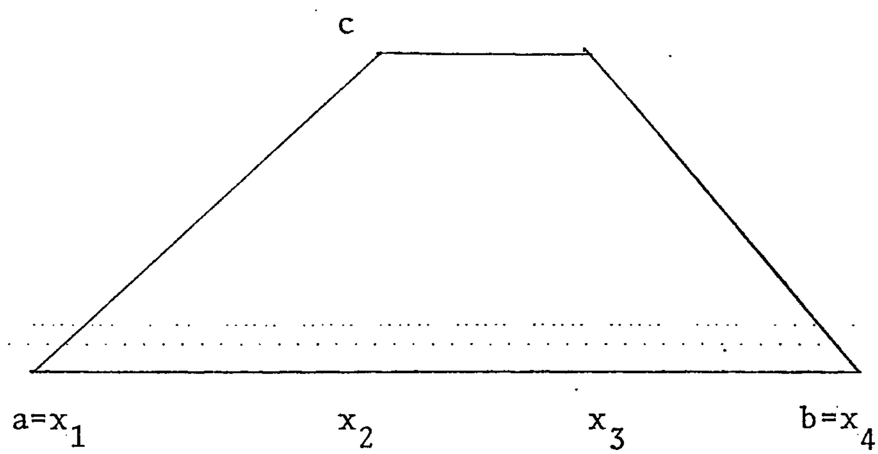
To obtain a trapezoidal distributed random number from the inversion method;

1) Generate a uniform random number U in (0,1).
2) Calculate $F(x_2)$ and $F(x_3)$.
3) If U is less or equal to $F(x_2)$, the required random variate is x_1+SQRT(2*U(x_2-x_1)/h).
4) IF U is greater than $F(x_2)$ and is less or equal to $F(x_3)$, the required random variate is (x_1+x_2)/2.+U/h.
5) If U is greater than $F(x_3)$, the required random variate is x_4-SQRT((x_3-x_4)*(x_1+x_2-x_3-x_4)*(1.-U)).

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM TRAPEZOIDAL VARIATE BY INVERSION   *
C                                                           *
C     P = THE VECTOR (X(1),X(2),X(3),X(4)) OF THE           *
C         DISTRIBUTION                                      *
C                                                           *
C     VAL = THE RETURNED TRAPEZOIDAL VARIATE                *
C                                                           *
C************************************************************
C
      SUBROUTINE TRA(P,VAL)
      DIMENSION P(4)
      DATA NSEED/567801/
      H=2./(1.-P(1)-P(2)+P(3)+P(4))
      F2=H*0.5*(P(2)-P(1))
      F3=H*0.5(2.*P(3)-P(1)-P(2))
      R=RAN(NSEED)
      IF(R.GT.F3) VAL=P(4)-SQRT((P(3)-P(4))*(P(1)+P(2)+P(3)
     +-P(4))*(1.-R))
      IF(R.LE.F3) VAL=(P(1)+P(2))/2.+R/H
      IF(R.LE.F2) VAL=P(1)+SQRT(2.*R*(P(2)-P(1))/H)
      RETURN
      END
```

B. REJECTION METHOD



Notice that in trapezoidal distribution, c is the height. Therefore it is decided to use $c=2./(x_4+x_3-x_2-x_1)$, $a=x_1$ and $b=x_4$.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM TRAPEZOIDAL VARIATE BY REJECTION*
C                                                          *
C     P = THE VECTOR (X(1),X(2),X(3),X(4)) OF THE          *
C         DISTRIBUTION                                     *
C                                                          *
C     VAL = THE RETURNED TRAPEZOIDAL VARIATE               *
C                                                          *
C***********************************************************
C
      SUBROUTINE TRA(P,VAL)
      DIMENSION P(4)
      DATA NSEED/567801/
      C=2.*(-P(1)-P(2)+P(3)+P(4))
    1 R1=RAN(NSEED)*C
      R2=RAN(NSEED)*(P(4)-P(1))+P(1)
      IF(R2.GT.P(3)) F=(R2-P(4))/(P(3)-P(4))*C
      IF(R2.LE.P(3)) F=3
      IF(R2.LE.P(2)) F=(R2-P(1))/(P(2)-P(1))*C
      IF(R1.LE.F) GO TO 2
      GO TO 1
    2 VAL=R2
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Example 1 Mean		Example 2 Mean		Example 1 mean	Example 2 mean
	N=100	N=1000	N=100	N=1000		
INVERSION	3.646	3.803	8.583	8.980	3.80	9.303
REJECTION	3.901	3.858	9.059	9.034	3.80	9.303

The input parameters of example one are:

X(1) = 1.0

X(2) = 3.5

X(3) = 4.0

X(4) = 6.7

The input parameters of example two are:

X(1) = 0.7

X(2) = 9.1

X(3) = 12.3

X(4) = 15.11

METHODS	EXECUTION TIME IN MILLI-SECONDS			
	Example 1		Example 2	
	N=100	N=1000	N=100	N=1000
INVERSION	29	277	31	254
REJECTION	43	438	67	345

SUMMARY OF CHI-SQUARE TEST

METHODS METHODS	N N	Degrees of freedom	Chi-square values	
			Example 1	Example 2
INVERSION	1000	5	6.131	6.314
	2500	8	12.051	4.799
	5000	11	10.525	8.933
REJECTION	1000	5	10.137	5.390
	2500	8	15.127	12.005
	5000	11	19.221	13.398

Degrees of freedom	α	Critical χ^2
5	0.05	11.0705
5	0.10	9.2364
8	0.05	15.5073
8	0.10	13.3616
11	0.05	19.6751
11	0.10	17.2750

N is the number of random variates generated for testing.

11. LOGNORMAL DISTRIBUTION

A variable x has a lognormal distribution if $y=\mathrm{Log}(x)$ is normally distributed. It can be natural log or log of any base. The density function of x is:

$$f(x) = 1/x^{*}f(y)$$

$$\text{or} \qquad f(x) = \frac{1}{x\sigma\sqrt{2\pi}}\,\mathrm{Exp}\left(-\frac{\mathrm{Log}(x)-\mu^2}{2\sigma^2}\right)$$

μ is the mean of y and σ is the standard deviation of y.

$$F(x) = \frac{1}{\sigma\sqrt{2\pi}}\int_0^x \frac{1}{t}\,\mathrm{Exp}\left(\frac{-\mathrm{Log}(t)-\mu)^2}{2\sigma^2}\right)dt$$

The mean and the variance of x are:

$$EX = \mathrm{Exp}\left(\mu + \frac{\sigma^2}{2}\right)$$

$$VX = EX^2(\mathrm{Exp}(\sigma^2)-1)$$

A. NAYLOR METHOD

If z is the standard normal variate, then

$$\mathrm{Log}(x) = \sigma z+\mu$$

$$x = \mathrm{Exp}(\sigma z+\mu)$$

In order to obtain x when EX and VX of x are given, σ and μ have to be expressed in terms of EX and VX.

From $$VX = EX^2(\mathrm{Exp}(\sigma^2)-1),$$

$$\frac{VX}{EX^2} = \mathrm{Exp}(\sigma^2)-1$$

$$\frac{VX}{EX^2}+1 = \mathrm{Exp}(\sigma^2)$$

implies $$\mathrm{Log}\left(\frac{VX}{EX^2}+1\right) = \sigma^2$$

Therefore $$\sigma = \mathrm{SQRT}\left(\mathrm{Log}\left(\frac{VX}{EX^2}+1\right)\right)$$

From $EX = \mathrm{Exp}\left(\mu+\frac{\sigma^2}{2}\right)$, we have $\mathrm{Log}(EX) = \mu+\frac{\sigma^2}{2}$

Subsitute $\mathrm{Log}\left(\frac{VX}{EX^2}+1\right) = \sigma^2$ into $\mathrm{Log}(EX) = \mu+\frac{\sigma^2}{2}$

$$\mathrm{Log}(EX) = \mu + \tfrac{1}{2}\mathrm{Log}\left(\frac{VX}{EX^2}+1\right)$$

$$\mu = \mathrm{Log}(EX) - \tfrac{1}{2}\mathrm{Log}\left(\frac{VX}{EX^2}+1\right)$$

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM LOGNORMAL VARIATE BY NAYLOR METHOD*
C                                                          *
C     EX = THE MEAN OF THE DISTRIBUTION                    *
C                                                          *
C     VX = THE VARIANCE OF THE DISTRIBUTION                *
C                                                          *
C     VAL = THE RETURNED LOGNORMAL VARIATE                 *
C                                                          *
C***********************************************************
C
      SUBROUTINE LOGN(EX,VX,VAL)
      DATA NSEED/567801/
      SIGMA=SQRT(ALOG(VX/EX**2+1)
      GMU=ALOG(EX)-0.5*ALOG(VX/EX**2+1)
      SUM=0.0
      DO 87 I=1,12
      SUM=SUM+RAN(NSEED)
   87 CONTINUE
      EXARG=GMU+SIGMA*(SUM-6.0)
      VAL=EXP(EXARG)
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHOD	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
NAYLOR	3.318	3.186	0.715	0.811	3.20	0.80

METHOD	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
NAYLOR	114	1149

SUMMARY OF T-TEST

METHOD	Degrees of freedom	T-values
NAYLOR	99	1.395
	999	0.060

Degrees of freedom	α	Critical T-values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of random variates generated for testing.

12. CHI-SQUARE DISTRIBUTION

Let $a_1, a_2, \ldots, a_n$ be n independent random variables, each of which is normally distributed with mean at zero and standard deviation equal to one, the variable

$$x = \sum_{i=1}^{n} a_i^2$$

has chi-square distribution with n degrees of freedom.

$$f_n(x) = \begin{cases} \dfrac{1}{2^{\frac{1}{2}n}\Gamma\left(\frac{n}{2}\right)} x^{\frac{1}{2}n-1} e^{-\frac{1}{2}x} & \text{for} \quad x>0 \\ 0 & \text{for} \quad x\leq 0 \end{cases}$$

$$F_n(x) = \frac{1}{2^{\frac{1}{2}n}\Gamma\left(\frac{n}{2}\right)} \int_0^x t^{\frac{1}{2}n-1} e^{-\frac{1}{2}t} dt$$

The mean and variance of x are:

$$EX = n$$

$$VX = 2n$$

A. SUM NORM METHOD

This method is straight forward by using the definition of the distribution. The steps are as follow :

1) Set x=0.0 .
2) Generate a standard normal variate y.
3) Add y^2 to x.
4) Go back to step 2 until n y^2 are added to x, then x is the required chi-square variate.

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C     TO RETURN A CHI-SQUARE RANDOM VARIATE BY SUM NORM   *
C     METHOD                                              *
C                                                         *
C     N = THE DEGREES OF FREEDOM                          *
C                                                         *
C     VAL = THE RETURNED CHI=SQUARE VARIATE               *
C                                                         *
C**********************************************************
C
      SUBROUTINE CHI(N,VAL)
      DATA NSEED/567801/
      X=0.0
      DO 1 I=1,N
      CALL NORM(0.0,1.0,Y)
      X=X+Y**2
    1 CONTINUE
      VAL=X
      RETURN
      END
```

B. WILSON-HILFERTY TRANSFORMATION METHOD

Due to the difficulty of obtaining reference (25) in which the method is described, we are unable to give a satisfactory derivation here. To generate a chi-square random number with this transformation:

1) Generate a standard normal random number y.
2) If the degrees of freedom, n of the distribution is greater than 25, $x=\frac{1}{2}(y-\sqrt{2n-1})^2$ is the required variate.
3) If the degrees of freedom, n is less or equal to 25, $x=n(y\sqrt{2/(9n)}+2/(9n)+1)^3$ is the required variate.

PROGRAM LISTING

```
C*****************************************************************
C                                                                *
C     TO RETURN A RANDOM CHI-SQUARE VARIATE  BY W.H. TRANSF.     *
C                                                                *
C     N = THE DEGREES OF FREEDOM                                 *
C                                                                *
C     VAL = THE RETURNED VARIATE                                 *
C                                                                *
C*****************************************************************
C
      SUBROUTINE CHI(N,VAL)
      CALL NORM(0.0,1.0,U)
      IF(N.GT.25) GO TO 1
      VAL=N*(U*SQRT(2./(9.*N))+2./(9/*N)+1.)**3
      GO TO 2
    1 VAL=0.5*(U-SQRT(2.*N-1.))**2
    2 RETURN
      END
```

C. COMBINATION METHOD

When the degrees of freedom of the distribution is non-integer, a technique similar to the non-integer Pascal method is used. Let the degrees of freedom be $n=d_1.d_2$ where d_1 is the unit digit and d_2 is the first digit after the decimal.

To obtain a random chi-square variate.

1) Generate $(10-d_2)$ chi-square variates with d_1 degrees of freedom, say x_1.
2) Generate d_2 chi-square variates with $(d_1 +1)$ degrees of freedom, say x_2 .
3) $x=(x_1+x_2)/10$ is the desired variate.

Verification:

The expected value of n is:

$$= \frac{(10-d_2)*d_1+d_2*(d_1+1)}{10}$$

$$= \frac{10d_1-d_2*d_1+d_2*d_1+d_2}{10}$$

$$= \frac{10d_1+d_2}{10}$$

$$= d_1.d_2$$

$$= n$$

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM CHI-SQUARE VARIATE BY COMBINATION *
C     METHOD                                                *
C                                                           *
C     DF = DEGREES OF FREEDOM OF THE DISTRIBUTION           *
C                                                           *
C     VAL = THE RETURNED VARIATE                            *
C                                                           *
C************************************************************
C
      SUBROUTINE CHI(DF,VAL)
      IA=DF
      IB=IA+1
      LB=DF*10.0-IA*10
      LA=10-LB
      X=0.0
      Y=0.0
      IF(IA.EQ.0.0) GO TO 1
      DO 3 I+1,LA
      DO 2 J=1,IA
      CALL NORM(0.0,1.0,XY)
    2 CONTINUE
    3 CONTINUE
      IF(IB.EQ.0.0) GO TO 66
    1 DO 65 I=1,LB
      DO 64 J=1,IB
      CALL NORM(0.0,1.0,XY)
      Y=Y+XY**2
   64 CONTINUE
   65 CONTINUE
      VAL=(X+Y)/10.0
   66 RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=200	N=100	N=200		
SUM NORM	10.232	10.089	18.940	23.161	10.0	20.0
W.H. TRANSFORM	11.527	11.222	22.130	28.438	10.0	20.0
COMBINATION	3.076	3.115	0.415	0.462	3.2	6.4

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=200
SUM NORM	802	1688
W.H. TRANSFORM	98	217
COMBINATION	2760	5281

N is the number of variates generated for testing.

SUMMARY OF T-TEST

METHODS	Degrees of freedom	T-values
SUM NORM	99	0.5331
	199	2.6153
W.H.TRANSFORM	99	3.2459
	199	3.2407
COMBINATION	99	1.9248
	199	1.2506

Degrees of freedom	α	Critical T-values
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

13. BETA DISTRIBUTION

Let x_1 and x_2 be two independent chi-square variables with m and n degrees of freedom respectively. Then the variable

$$x = \frac{x_1}{x_1 + x_2}$$

has beta distribution with m, n degrees of freedom.

If p=m/2 and q=n/2; the density function of x

$$f_{x,p,q} = \frac{\Gamma(p+q)}{\Gamma(p)\ \Gamma(q)}\ x^{p-1}(1-x)^{q-1} \quad 0<x<1, p>0, q>0$$

The mean and variance of x are:

$$EX = \frac{p}{p+q}$$

$$VX = \frac{pq}{(p+q)^2(p+q+1)}$$

A. CHI-SQUARE METHOD

From the definition of the distribution, the steps to obtain a random variate of beta distribution with m and n degrees of freedom is very straightforward.

1) Generate chi-square variate x_1 with m degrees of freedom.
2) Generate chi-square variate x_2 with n degrees of freedom.
3) $x=x_1/(x_1+x_2)$ is the desired beta variate.

PROGRAM LISTING

```
C**********************************************************
C                                                         *
C     TO RETURN A RANDOM BETA VARIATE BY CHI-SQUARE METHOD *
C                                                         *
C     M = DEGREES OF FREEDOM                              *
C                                                         *
C     N = DEGREES OF FREEDOM                              *
C                                                         *
C     VAL = THE RETURNED BETA VARIATE                     *
C                                                         *
C**********************************************************
C
      SUBROUTINE BETA(M,N,VAL)
      DATA NSEED/567801/
      SUMM=0.0
      DO 1 I=1,M
      CALL NORM(0.0,1.0,U)
      SUMM=SUMM+U**2
    1 CONTINUE
      SUMN=0.0
      DO 2 J=1,N
      CALL NORM(0.0,1.0,v)
      SUMN=SUMN+V**2
    2 CONTINUE
      VAL=SUMM/(SUMM+SUMN)
      RETURN
      END
```

B. ORDERING METHOD

Let $y_1, y_2, \ldots, y_{p+q-1}$ be p+q-1 independent variables with a common distribution F(y), and let $x_1 \leq x_2 \leq x_3 \leq \ldots \leq x_{p+q-1}$ be the ordered values $y_1, y_2, \ldots, y_{p+q-1}$ in ascending order. Let the distribution of x_p be denoted by $F_p(y)$.

Since $x_p \leq x$ if and only if at least p of the values $x_1, x_2, \ldots, x_{p+q-1}$ are $\leq$ x, we have

$$F_p(y) = \sum_{j=p}^{p+q-1} \binom{p+q-1}{q} F(y)^j (1-F(y))^{q-1}$$

$$= \frac{(p+q-1)!}{(p-1)!(q-1)!} \int_0^{F(y)} t^{p-1}(1-t)^{q-1} dt$$

$$= \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} \int_0^{F(y)} t^{p-1}(1-t)^{q-1} dt$$

$$= \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} \int_0^{y} f(t)^{p-1}(1-F(t))^{q-1} dF(t)$$

$$= \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} y^{p-1}(1-y)^{q-1}$$

Therefore, x_p is beta distributed.

To obtain a random beta variate, a uniform distribution with unit range is used as the common distribution.

1) Generate p+q-1 uniform random variates.

2) Order them in ascending orders.

3) The desired variate is the pth variate.

PROGRAM LISTING

```
C****************************************************************
C                                                               *
C     TO RETURN A RANDOM BETA VARIATE BY ORDERING METHOD        *
C                                                               *
C     M = DEGREES OF FREEDOM                                    *
C                                                               *
C     N = DEGREES OF FREEDOM                                    *
C                                                               *
C     VAL = THE RETURNED BETA VARIATE                           *
C                                                               *
C****************************************************************
C
      SUBROUTINE BETA(M,N,VAL)
      DIMENSION X(1000)
      DATA NSEED/567801/
      IM=M/2
      IN=N/2
      IP=IM+IN-1
      DO 1 I=1,IP
      R=RAN(NSEED)
      CALL ARRANG(R,I,X)
    1 CONTINUE
      VAL=X(IM)
      RETURN
      END
```

C. NON-INTEGER BETA

The assumption that a beta variate with non-integer degrees of freedom m and n can be generated by the combination method of chi-square distribution is wrong.

Verification:

Let x_1 be the chi-square variate obtained from chi-square combination method with m degrees of freedom.

Let x_2 be the chi-square variate obtained from chi-square combination method with n degrees of freedom.

Since the beta variate $x = \dfrac{x_1}{x_1 + x_2}$

$$EX(x) = EX\left(\frac{x_1}{x_1 + x_2}\right)$$

$$\neq \frac{EX(x_1)}{EX(x_1)+EX(x_2)}$$

$$= \frac{m}{m+n}$$

We have shown that the expected value of beta variate generated by combination method is not m/(m+n).

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
CHI-SQUARE						
Example 1	0.601	0.614	0.029	0.024	0.611	0.024
Example 2	0.286	0.284	0.023	0.026	0.286	0.023
ORDERING						
Example 1	0.623	0.607	0.013	0.013	0.611	0.014
Example 2	0.270	0.282	0.013	0.013	0.286	0.023

METHODS	INPUT DEGREES OF FREEDOM	
	M	N
CHI-SQUARE		
Example 1	11	7
Example 2	4	10
ORDERING		
Example 1	11	7
Example 2	4	10

N in the top table is the number of random variates generated for testing.

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
CHI-SQUARE		
Example 1	1756	17892
Example 2	1364	14743
ORDERING		
Example 1	130	1302
Example 2	132	1243

N is the number of random variates generated for testing. Notice that the execution times of both methods depend on the input degrees of freedom.

SUMMARY OF T TEST

METHODS		DEGREES OF FREEDOM	T VALUES
CHI SQUARE	EXAMPLE 1	99	0.593
		999	0.592
	EXAMPLE 2	99	0.019
		999	0.333
ORDERING	EXAMPLE 1	99	1.052
		999	1.109
	EXAMPLE 2	99	1.403
		999	1.109

Degrees of freedom	α	Critical T values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of variates generated for testing.

14. F-DISTRIBUTION

If x_1 and x_2 are independent random variables which are chi-square distributed and have m and n degrees of freedom respectively, then

$$x = (x_1/m)(x_2/n)^{-1}$$

has F distribution with m and n degrees of freedom.

The mean and the variance of x are:

$$EX = \frac{n}{n-2} \qquad n>2$$

$$VX = \frac{2n^2(m+n-2)}{m(n-2)^2(n-4)} \qquad n>4$$

A. CHI-SQUARE METHOD

From the definition of the distribution, the steps to obtain a random variate of F-distribution with m and n degrees of freedom is very straightforward.

1) Generate chi-square variate x_1 with m degrees of freedom.

2) Generate chi-square variate x_2 with n degrees of freedom.

3) $x=(x_1*n)/(x_2*m)$ is the desired F variate.

PROGRAM LISTING

```
C***************************************************************
C                                                              *
C     TO RETURN A RANDOM F DISTRIBUTED VARIATE BY CHI SQUARE   *
C     METHOD                                                   *
C                                                              *
C     M = DEGREES OF FREEDOM                                   *
C                                                              *
C     N = DEGREES OF FREEDOM                                   *
C                                                              *
C     VAL = THE RETURNED F VARIATE                             *
C                                                              *
C***************************************************************
C
      SUBROUTINE FDIS(M,N,VAL)
      DATA NSEED/567801/
      TEST=M/2.0
      IF(TEST.LE.2.0) WRITE(6,55)
   55 FORMAT(' THE VARIANCE OF THE DISTRIBUTION IS INFINITE ')
      CALL CHI(M,V)
      CALL CHI(N,U)
      VAL=(V*N)/(U*M)
      RETURN
      END
```

B. ORDERING BETA METHOD

To obtain a F variate with even degrees of freedom m and n, one can obtain a beta variate from the ordering method with m/2 and n/2 degrees of freedom, say the variate is x. The required F variate is n*x/(m*(1-x)).

Verification:

$$\frac{n*x}{m*(1-x)} = \frac{n*x_1/(x_1+x_2)}{m*(1-x_1/(x_1+x_2))}$$

$$= \frac{n*x_1}{(x_1+x_2-x_1)}$$

$$= \frac{n*x_1}{m*x_2}$$

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM F VARIATE BY ORDERING BETA        *
C                                                          *
C     M = DEGREES OF FREEDOM                               *
C                                                          *
C     N = DEGREES OF FREEDOM                               *
C                                                          *
C     VAL = THE RETURNED F VARIATE                         *
C                                                          *
C***********************************************************
C
      SUBROUTINE FDIS(M,N,VAL)
      IN=N/2
      IM=M/2
      CALL BETA(IM,IN,X)
      VAL=IM*X(IN*(1-X))
      RETURN
      END
```

C. NON-INTEGER F

The assumption that the F variate with non-integer degrees of freedom m and n can be generated by the combination method of chi-square distribution is wrong.

Verification:

Let x_1 be the chi-square variate obtained from chi-square combination method with m degrees of freedom.

Let x_2 be the chi-square variate obtained from chi-square combination method with n degrees of freedom.

Since the F variate $x = \frac{nx_1}{mx_2}$,

$$EX(x) = \frac{n}{m} EX\left(\frac{x_1}{x_2}\right)$$

By example 14.C.1, $\frac{n}{n-2} \neq \frac{n}{m} EX\left(\frac{x_1}{x_2}\right)$

$$\text{implies} \quad EX(x) \neq \frac{n}{n-2}$$

We have shown that the expected value of the F variate generated by combination method is not n/(n-2).

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
CHI-SQUARE						
Example 1	1.204	1.269	0.914	1.276	1.333	1.444
Example 2	1.357	1.221	0.950	1.059	1.250	1.038
ORDERING						
Example 1	1.042	1.174	0.792	1.118	1.200	0.900
Example 2	1.459	1.227	1.777	0.841	1.250	1.038

METHODS	INPUT DEGREES OF FREEDOM	
	M	N
CHI-SQUARE		
Example 1	5	8
Example 2	12	10
ORDERING		
Example 1	79	627
Example 2	90	916

N in the top table is the number of random variates generated for testing.

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
CHI-SQUARE		
Example 1	3362	14783
Example 2	1996	20861
ORDERING		
Example 1	79	627
Example 2	90	916

N is the number of variates generated for testing. Notice that the execution time of both methods depends on the input degrees of freedom.

SUMMARY OF T TEST

METHODS		DEGREES OF FREEDOM	T VALUES
CHI SQUARE	EXAMPLE 1	99	1.349
		999	1.792
	EXAMPLE 2	99	1.097
		999	0.891
ORDERING	EXAMPLE 1	99	1.775
		999	0.775
	EXAMPLE 2	99	1.568
		999	0.793

Degrees of freedom	α	Critical T values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of random variate generated for testing.

15. STUDENT-T DISTRIBUTION

Let x be a chi-square variable with n degrees of freedom, and a be a normal variate with mean at zero and standard deviation equal to σ, the variable

$$x = \frac{a\sqrt{n}}{\sqrt{x}}$$

has student-t distribution with n degrees of freedom.

$$f_n(x) = \frac{1}{\sqrt{n\pi}} \frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \left(1 + \frac{x^2}{n}\right)^{-\frac{1}{2}(n+1)}$$

$$F_n(x) = \frac{1}{\sqrt{n\pi}} \frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \int_{-\infty}^{x} \frac{du}{\left(1 + \frac{u^2}{n}\right)^{(n+1)/2}}$$

The mean and the variance of x are:

$$EX = 0.0$$

$$VX = n/(n-2)$$

$f_n(x)$ shows that the distribution is independent of the standard deviation σ of the basic variable a, therefore σ can be taken as 1.

A. CHI-SQUARE METHOD

From the definition of the distribution, the steps to obtain a random variate of T distribution with n degrees of freedom is very straightforward.

1) Generate a standard normal variate x_1.

2) Generate a chi-square variate x_2 with n degrees of freedom.

3) $x=x_1/\mathrm{SQRT}(x_2/n)$ is the desired variate.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A T-DISTRIBUTED RANDOM VARIATE BY CHI SQUARE*
C     METHOD                                                *
C                                                           *
C     N = DEGREES OF FREEDOM                                *
C                                                           *
C     VAL = THE RETURNED T VARIATE                          *
C                                                           *
C************************************************************
C
      SUBROUTINE TDIS(N,VAL)
      DATA NSEED/567801/
      T=0.0
      DO 1 I=1,N
      CALL NORM(0.0,1.0,TAL)
      T=T+TAL**2
    1 CONTINUE
      CALL NORM(0.0,1.0,TAL)
      VAL=TAL/(SQRT(T/N))
      RETURN
      END
```

B. CACOULLOS METHOD

Cacoullos has shown that if x_1 and x_2 are independent chi-square variates each with n degrees of freedom, then $\frac{1}{2}\, n(x_2-x_1)(x_2x_1)^{-\frac{1}{2}}$ has T distribution with n degrees of freedom. His complete proof is in (8), therefore we shall not repeat it here.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM T DISTRIBUTED VARIATE BY CACOULLOS*
C     METHOD                                                *
C                                                           *
C     N = DEGREES OF FREEDOM                                *
C                                                           *
C     VAL = THE RETURNED T VARIATE                          *
C                                                           *
C************************************************************
C
      SUBROUTINE TDIS(N,VAL)
      DATA NSEED/567801/
      T=0.0
      DO 1 I=1,N
      CALL NORM(0.0,1.0,TAL)
      T=T+TAL**2
    1 CONTINUE
      CALL NORM(0.0,1.0,TAL)
      VAL=TAL/(SQRT(T/N))
      RETURN
      END
```

C. NON-INTEGER T

The assumption that the T variate with non-integer degrees of freedom n can be generated by the combination method of chi-square is wrong.

Verification:

Let x_1 be the chi-square variate obtained from chi-square combination method with n degrees of freedom.

Let a be the normal variate.

Since the T variate $x = \dfrac{a\sqrt{n}}{\sqrt{x_1}}$

$$
\begin{aligned}
EX(x) &= EX\left(\frac{a\sqrt{n}}{\sqrt{x_1}}\right) \\
&= \sqrt{n}\ EX\left(\frac{a}{\sqrt{x_1}}\right) \\
&\neq \frac{EX(a)}{EX(\sqrt{x_1})} \\
&= 0.0
\end{aligned}
$$

We have shown that the expected value of T variate generated by combination method is not 0.0.

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
CHI-SQUARE						
Example 1	0.105	0.003	2.479	3.701	0.0	3.000
Example 2	-0.000	-0.010	0.929	1.105	0.0	1.142
CACOULLOS						
Example 1	0.259	-0.003	1.823	3.299	0.0	3.000
Example 2	-0.126	-0.004	1.172	1.044	0.0	1.142

The input degrees of freedom of example one is 3.

The input degrees of freedom of example two is 16.

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
CHI-SQUARE		
Example 1	334	3526
Example 2	3094	16108
CACOULLOS		
Example 1	594	5447
Example 2	1489	30247

N is the number of random variates generated for testing.

SUMMARY OF T TEST

METHODS	DEGREES OF FREEDOM	T VALUES
CHI SQUARE EXAMPLE 1	99	0.666
	999	0.049
EXAMPLE 2	99	0.000
	999	0.301
CACOULLOS EXAMPLE 1	99	1.918
	999	0.052
EXAMPLE 2	99	1.164
	999	1.124

Degrees of freedom	α	Critical T values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of random variates generated for testing.

16. Z-DISTRIBUTION

Let x_1 and x_2 be two independent chi-square variables with m and n degrees of freedom respectively. The variable x of

$$e^{2x} = \frac{nx_1}{mx_2}$$

has Z-distribution with m,n degrees of freedom.

$$f_{mn}(x) = 2m^{\frac{1}{2}m} n^{\frac{1}{2}n} \frac{\Gamma\left(\frac{m+n}{2}\right)}{\Gamma\left(\frac{m}{2}\right)\Gamma\left(\frac{n}{2}\right)} \frac{e^{mx}}{\left(me^{2x}+n\right)^{(m+n)/2}}$$

$$F_{mn}(x) = \frac{\Gamma\left(\frac{m+n}{2}\right)}{\Gamma\left(\frac{m}{2}\right)\Gamma\left(\frac{n}{2}\right)} \int^{me^{2x}/n} \frac{u^{\frac{1}{2}m-1}}{(u+1)^{(m+1)2}} du$$

The mean and the variance of e^{2x} are :

$$EX = \frac{n}{n-2} \qquad n>2$$

$$VX = \frac{2n^2(m+n-2)}{m(n-2)^2(n-4)} \qquad n>4$$

A. CHI-SQUARE METHOD

From the definition of the distribution, the steps to obtain a random variate of the Z distribution with m and n degrees of freedom is very straightforward.

1) Generate chi-square variate x_1 with m degrees of freedom.

2) Generare chi-square variate x_2 with n degrees of freedom.

3) $x = \frac{1}{2}\text{Log}_e((n*x_1)/(m*x_2))$ is the desired variate.

PROGRAM LISTING

```
C************************************************************
C                                                           *
C     TO RETURN A RANDOM Z DISTRIBUTED VARIATE BY CHI       *
C     SQUARE METHOD                                         *
C                                                           *
C     M = DEGREES OF FREEDOM                                *
C                                                           *
C     N = DEGREES OF FREEDOM                                *
C                                                           *
C     VAL = THE RETURNED Z VARIATE                          *
C                                                           *
C************************************************************
C
      SUBROUTINE ZDIS(M,N,VAL)
      DARA NSEED/567801/
      SUMM=0.0
      SUMN=0.0
      DO 1 I=1,M
      CALL NORM(0.0,1.0,U)
      SUMM=SUMM+U**2
    1 CONTINUE
      DO 2 K=1,N
      CALL NORM(0.0,1.0,V)
      SUMN=SUMN+V**2
    2 CONTINUE
      VAL=ALOG((N*SUMM)/(M*SUMN))*0.5
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		EX	VX
	N=100	N=1000	N=100	N=1000		
CHI-SQUARE						
Example 1	1.238	1.372	0.691	1.474	1.40	1.809
Example 2	1.158	1.168	0.896	0.889	1.22	1.170

METHODS	INPUT DEGREES OF FREEDOM	
	M	N
CHI-SQUARE		
Example 1	13	7
Example 2	5	11

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
CHI-SQUARE		
Example 1	1376	15029
Example 2	1661	18316

The execution time of this method depends on the input degrees of freedom.

SUMMARY OF T-TEST

METHOD	DEGREES OF FREEDOM	T VALUES OF e^{2x}
CHI SQUARE EXAMPLE 1	99	1.948
	999	0.729
EXAMPLE 2	99	0.676
	999	1.811

Degrees of freedom	α	Critical T values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.15	1.645
∞	0.10	1.282

N is the number of random variates generated for testing.

17. GAMMA DISTRIBUTION

A random variable x has a gamma distribution if its probability density function is of the form

$$f(x) = \frac{\beta^{\alpha}(x-\gamma)^{\alpha-1}\exp(-(x-\gamma)\beta)}{\Gamma(\alpha)} \qquad \alpha>0,\beta>0,x>\gamma$$

x depends on the shape parameter α, the location parameter γ and the scale parameter β.

The mean and the variance of x are:

$$EX = \frac{\alpha}{\beta}$$

$$VX = \frac{\alpha}{\beta^2}$$

The standard form of the distribution is obtained by putting $\beta=1$ and $\gamma=0$, giving

$$f(x) = \frac{x^{\alpha-1}e^{-x}}{\Gamma(\alpha)} \qquad x\geq 0$$

If $\alpha=1$ and $\gamma=0$, the gamma distribution is the standard exponential distribution, giving

$$f(x) = \beta e^{-x\beta}$$

If α is a positive integer and $\gamma=0$, the gamma distribution is the standard Erlang distribution

$$f(x) = \frac{\beta^{\alpha} x^{\alpha-1} \exp(-x\beta)}{(\alpha-1)!}$$

$$= \frac{\beta}{(\alpha-1)!} (\beta x)^{\alpha-1} e^{-\beta x}$$

If $\alpha=r/2$, r is a positive integer, and if $\beta=0.5$, the gamma distribution is a chi-square distribution.

$$f(x) = \frac{1}{\Gamma(\frac{1}{2}r) 2^{\frac{1}{2}r}} x^{\frac{1}{2}r-1} e^{-\frac{1}{2}x} \quad 0<x<\infty$$

with $EX=\alpha/\beta=(\frac{1}{2}r)2=r$

$VX=\alpha/\beta^2=(\frac{1}{2}r)2^2=2r$

r is called the degrees of freedom

A. JOHNK METHOD

Johnk method to generate a gamma variate with shape parameter $\alpha<1$ was originally from (36), and was translated in (44). However, the translation in English does not contain the derivation of the method. Therefore we shall leave it for the inquisitive mind to find it out. The method has two sections. A beta variate is generated first, with which a gamma variate is obtained.

Steps to obtain a beta variate with M and N+1 degrees of freedom.

1) Generate a uniform random number U in (0,1), and set $x=U^{1/M}$.
2) Generate a uniform random number V in (0,1), and set $y=V^{1/(N+1)}$.
3) If x+y=1, go to step 5; otherwise go to step 4.
4) Go to step 1.
5) x is the required beta variate.

Steps to obtain a gamma variate with $\alpha<1$.

1) Generate a random beta number with α and $2-\alpha$ degrees of freedom.
2) Generate uniform random numbers U and V in (0,1) and set $y=-\mathrm{Log}_e(U*V)$.
3) x*y is the desired gamma variate.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C     TO RETURN A RANDOM GAMMA VARIATE OF NON-INTEGER        *
C     SHAPE PARAMETER BY JOHNK METHOD                        *
C                                                            *
C     ALPHA = THE SHAPE PARAMETER                            *
C                                                            *
C     SCALE = THE SCALE PARAMETER                            *
C                                                            *
C     POS = THE LOCATION PARAMETER                           *
C                                                            *
C     VAL = THE RETURNED GAMMA VARIATE                       *
C                                                            *
C*************************************************************
C
      SUBROUTINE GAMMA(ALPHA,SCALE,POS,VAL)
      DATA NSEED/567801/
      IPI=ALPHA
      Z=0.0
      A=ALPHA
      B=1.-ALPHA
      IA=1000000./A
      IB=1000000./B
      C=IA/1000000.
      D=IB/1000000.
      IF(IPI) 97,97,95
   95 DO 96 I=1,IPI
      Z=Z-ALOG(RAN(NSEED))
   96 CONTINUE
   97 X=RAN(NSEED)**C
      W=RAN(NSEED)**D
      SUM=X+W
      IF(SUM.LE.1.0) GO TO 99
      GO TO 97
   99 BETA-X/SUM
      Y=-ALOG(RAN(NSEED))
      VAL=(Z+BETA*Y)/SCALE+POS
      RETURN
      END
```

B. SUM EXPONENTIAL METHOD

The waiting time to the αth event in a series of events happening in accordance with a Poisson probability law at the rate of β events per unit of time obeys an Erlang probability law with parameters α and β.

Verification:

The waiting time to the first event obeys the exponential probability law with parameter β. (See page 56)

The probability of α events occurring in the time from 0 to t is

$$\frac{(\beta t)^{\alpha} e^{-\beta t}}{\alpha !} \qquad \text{for } t \geq 0$$

Let $F_{\alpha}(t)$ be the probability that the time of occurrence of the αth event will be less than or equal to t. Then $1-F_{\alpha}(t)$ is the probability that the number of events occurring in the time from 0 to t is less than α.

$$1-F_{\alpha}(t) = \sum_{k=0}^{\alpha-1} \frac{1}{k!}(\beta t)^{k} e^{-\beta t} \quad \text{for } t \geq 0$$

Differentiating both sides with respect to t:

$$-f_{\alpha}(t) = -\beta e^{-\beta t} - \beta^{2} t e^{-\beta t} + \beta e^{-\beta t} - \frac{\beta^{3} t^{2}}{2!} e^{-\beta t} + \ldots$$

$$- \frac{\beta}{(\alpha-1)!}(\beta t)^{\alpha-1} e^{-\beta t} \qquad t \geq 0$$

$$f_\alpha(t) = \frac{\beta}{(\alpha-1)!}(\beta t)^{\alpha-1}e^{-\beta t}$$ which is the

density function for the erlang distribution.

To obtain an Erlang variate:

1) Generate α exponential variates $x_1, x_2, \ldots, x_\alpha$ with expected value $1/\beta$.

2) The variate $x = \sum_{i=1}^{\alpha} x_i = -1/\beta \sum_{i=1}^{\alpha} \text{Log}_e r_i$ is the required variate.

PROGRAM LISTING

```
C*************************************************************
C                                                            *
C      TO RETURN A RANDOM ERLANG VARIATE                     *
C                                                            *
C      ALPHA = THE SHAPE PARAMETER                           *
C                                                            *
C      SCALE = THE SCALE PARAMETER                           *
C                                                            *
C      POS = THE LOCATION PARAMETER                          *
C                                                            *
C      VAL = THE RETURNED ERLANG VARIATE                     *
C                                                            *
C*************************************************************
C
      SUBROUTINE ERLG(ALPHA,SCALE,POS,VAL)
      DATA NSEED/567801/
      I=ALPHA
      X=0.0
      DO 2 J=1,I
      X=X-ALOG(RAN(NSEED))/SCALE
    2 CONTINUE
      VAL=X+POS
      RETURN
      END
```

C. COMBINATION METHOD

With similar argument as chi-square variate of non-integral degrees of freedom, a gamma variate of real α greater than one can be generated by the combination method. Observe that the combination method applies to chi-square because the expected value of the variate is equal to the degrees of freedom. This method applies to gamma because the expected value and the variance of the gamma distribution are linear functions of α.

Let $\alpha = d_1.d_2$ where d_1 is the unit digit and d_2 is the first digit after the decimal. To obtain a gamma variate:

1) Generate a uniform random variate, U in (0,1).
2) If U is less than or equal to $d_2/10$, use d_1+1 as the shape parameter to generate the required variate.
3) If U is greater than $d_2/10$, use d_1 as the shape parameter to generate the required variate.

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM GAMMA VARIATE OF NON INTEGRAL     *
C     SHAPE PARAMETER GREATER THAN ONE                     *
C                                                          *
C     ALPHA = THE SHAPE PARAMETER                          *
C                                                          *
C     SCALE = THE SCALE PARAMETER                          *
C                                                          *
C     POS = THE POSITION PARAMETER                         *
C                                                          *
C     VAL = THE RETURNED GAMMA VARIATE                     *
C                                                          *
C***********************************************************
C
      SUBROUTINE GAMMA(ALPHA,SCALE,POS,VAL)
      DATA NSEED/567801/
      IAL=ALPHA
      IF=(ALPHA-IAL)*10
      P=IF/10.0
      R=RAN(NSEED)
      IF(R.LE.P) N=IAL+1
      IF(R.GT.P) N=IAL
      Z=0.0
      DO 2 I=1,N
      Z=Z-ALOG(RAN(NSEED))/SCALE
    2 CONTINUE
      VAL=Z+POS
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHODS	PROGRAM RESULTS				GIVEN	
	Mean		Variance		EX	VX
	N=100	N=1000	N=100	N=1000		
SUM EXPONENT.	6.792	6.148	2.539	3.794	6.166	3.472
JOHNK	2.367	2.349	0.296	0.246	2.375	0.313
COMBINATION	4.883	5.929	5.163	6.703	5.250	6.563

METHODS	INPUT PARAMETERS		
	ALPHA	SCALE	POS
SUM EXPONENT.	11	1.784	0.0
JOHNK	0.375	7.587	2.0
COMBINATION	4.2	0.8	0.0

METHODS	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
SUM EXPONENT.	81	797
JOHNK	89	871
COMBINATION	55	510

N is the number of random variates generated for testing.

SUMMARY OF T TEST

METHODS	Degrees of freedom	T-values
SUM EXPONENTIAL	99	3.925
	999	0.302
JOHNK	99	0.270
	999	1.657
COMBINATION	99	1.507
	999	1.063

Degrees of freedom	α	Critical T-values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of random variates generated for testing.

18. PARETO DISTRIBUTION

Pareto distribution deals with the distribution of income over a population. It can be stated as

$$N = Ax^{-a}$$

where N is the number of persons having income x; while A and a are parameters. a is the shape parameter or the Pareto's constant. The density function is

$$f(x) = \frac{ak^a}{x^{a+1}} \qquad a>0, \; x\geq k>0$$

and cumulative function

$$F(x) = 1 - \left(\frac{k}{x}\right)^a \qquad k>0, a>0, x\geq k$$

where k is the minimum income.

The mean and the variance of x are

$$EX = \frac{ak}{a-1} \qquad \text{if } a>1$$

$$VX = \frac{ak}{(a-1)^2(a-2)} \qquad \text{if } a>2$$

A. INVERSION METHOD

When the minimum income, k and the shape parameter are given, inversion is easy to be done.

$$\text{Since} \quad \left(\frac{k}{x}\right)^a = 1-F(x)$$

$$a*(\text{Log}_e k-\text{Log}_e x) = \text{Log}_e (1-F(x))$$

$$\text{Log}_e k=\text{Log}_e x = \text{Log}_e (1-F(x))/a$$

$$\text{Log}_e x = \text{Log}_e k-\text{Log}_e (1-F(x))/a$$

$$x = \text{Exp}(\text{Log}_e k-\text{Log}_e (1-F(x))/a)$$

$$x = (k/(1-F(x))/e^a$$

$$x = k*e^a/(1-F(x))$$

If F(x) is a uniform random number in (0,1) , then 1-F(x) is also a uniform random number in (0,1).

Therefore to obtain a random pareto variate:

1) Generate U, a uniform random number in (0,1).
2) Set $x=e^a*k/U$, x is the desired pareto variate.

When the mean, EX, and the variance, VX are given and if a is larger than 2, then a and k have to be expressed in terms of EX and VX in order to do the inversion.

$$EX = \frac{ak}{a-1} \quad \text{and} \quad VX = \frac{ak}{(a-1)^2(a-2)}$$

Subsititute EX into VX,

$$VX = \frac{EX^2}{a(a-2)}$$

$$\text{implies} \quad a(a-2) = \frac{EX^2}{VX}$$

$$a^2-2a-\frac{EX^2}{VX} = 0$$

$$a = \frac{2+\sqrt{4+4(EX^2/VX)}}{2}$$

By the assumption that a>2,

$$a = \frac{2+\sqrt{4+4(EX^2/VX)}}{2}$$

From $VX = \dfrac{ak}{a-1}$, $\quad k = \dfrac{EX*(a-1)}{a}$

x can now be calculated by $x=k*e^a/(1-F(x))$

PROGRAM LISTING

```
C***********************************************************
C                                                          *
C     TO RETURN A RANDOM PARETO VARIATE  BY INVERSION      *
C                                                          *
C     EX = THE GIVEN MEAN OF THE DISTRIBUTION              *
C                                                          *
C     VX = THE GIVEN VARIANCE OF THE DISTRIBUTION          *
C                                                          *
C     VAL = THE RETURNED PARETO VARIATE                    *
C                                                          *
C***********************************************************
C
      SUBROUTINE PARETO(EX,VX,VAL)
      DATA NSEED/567801/
      A=(2.+SQRT(4.+4*(EX**2/VX)))/2.
      IF(A.LT.2.) WRITE(6,88)
      PK=EX*(A-1.)/A
    1 R=RAN(NSEED)
      VAL=EXP(ALOG(PK)-ALOG(R)/A)
      IF(VAL.LT.PK) GO TO 1
   88 FORMATN('A IS LESS THAN 2 ')
      RETURN
      END
```

PROGRAM TESTING RESULTS

METHOD	PROGRAM RESULTS				GIVEN	
	Mean		Variance		Mean	Variance
	N=100	N=1000	N=100	N=1000		
NAYLOR	3.318	0.715	3.186	0.811	3.20	0.80

METHOD	EXECUTION TIME IN MILLI-SECONDS	
	N=100	N=1000
NAYLOR	114	1149

SUMMARY OF T-TEST

METHOD	Degrees of freedom	T-Values
NAYLOR	99 999 999	1.395 0.060

Degrees of freedom	α	Critical T-values
60	0.05	1.671
60	0.10	1.296
120	0.05	1.658
120	0.10	1.289
∞	0.05	1.645
∞	0.10	1.282

N is the number of random variates generated for testing.

CONCLUSION:

Quite a few random number generators have been examined and tested. Roughly speaking, the generating methods fall into several catagories.

One catagory which is seldom used nowadays is to relate physical methods with mathematical methods. A good example is Marsaglia's method on any discrete distribution. Some work had also been done to reduce storage spaces and to suit the capacities of the computers.

Another catagory is to apply common statistical facts and theorems. The using of cumulative functions as random uniform variates between zero and one is a significant contribution to the generating methods. Both versions of Von Neumann's methods on the Exponential distribution use conditional probabilities and statistical independence of events. There are interesting relationships between the distributions which make possible the approximations of Binomial and Poisson variates by Normal variates. The Poisson number of arrivals of Exponential arrival time is also used. Moreover, there is also the summation of Exponential variates to approximate the Erlang variates. These methods were not easy to come by, they all require careful studies on their mathematical inferences. Out from all these, a clever developement is to treat the Pascal distribution as a discrete analogue of the Exponential distribution. Ordering statistics has been used also in several methods.

The derivation for the Beta variate is half statistics and half calculus. The mathematics involved in the Cosine and Sine method for Normal variate is all Trigonometry and Calculus. One can see that not only statisticians were interested in random number generator.

The third catagory is the polynomial approximations. There are Hasting's and Teichroew's.

With a final word, we can conclude that anyone who does simulation work can now make use of our study. He can also design number generators for his own distribution. The process is to examine first the direct method. If it cannot apply, then scale the cumulative function to be in the range between zero and one. If the cumulative function exists in a closed form, inversion can be used. If not, the last resort is to find the relationship between the distribution and those which are in this thesis.

CHAPTER III

Usage descriptions of DTIME and CTIME

1. SUBROUTINE DTIME

Subroutine DTIME(KODE,L,K,P,VAL) is used to generate random values from any discrete distribution discussed in Chapter II. The 5 arguments are:

- KODE = code for the distribution, can be 10,20,30,40, 50 or 60.
- L = 1 or 0 to indicate the nature of vector P. See individual input arguments sections for details.
- K = dimension of vector P.
- P = vector giving the mean, variance, minimum, maximum or other parameters as required. See individual input arguments sections for details.
- VAL = the returned random variate.

A. ANY DISCRETE DISTRIBUTION

1) SELECTION:

The CDF method is selected for DTIME because it is simple to program, its testing results are satisfactory and its execution time is much less than those of the other two methods.

2) INPUT ARGUMENTS:

KODE = 10

L = 1 when MIN is supplied

= 1 when MIN and MAX are supplied

= 0 neither MIN nor MAX are supplied

K = 2 to 50 which is the dimension of vector P in DTIME

P = a vector of values and their corresponding probabilities of occurrence. If the values are $x_1, x_2, \ldots$ and their probabilities were $p_1, p_2, p_3, \ldots$, then $P=(x_1, p_1, x_2, p_2, \ldots)$. When L is supplied as 1, MIN and MAX can be put in the order that MIN comes first. MAX cannot be supplied without supplying MIN.

3) USAGE SAMPLE:

CALL DTIME(10,0,10,P,VAL) --- Input with x_i and p_i i=1,2, 3,4 and 5.

CALL DTIME(10,1,11,P,VAL) --- Input with x_i, p_i i=1,2,3,4 and 5; and MIN.

B. HYPERGEOMETRIC DISTRIBUTION

1) SELECTION:

Direct method is used for DTIME because it was the only method investigated.

2) INPUT ARGUMENTS:

KODE = 20

L = 1 when EX, VX and PROB, the probability of the occurrence of class I of the distribution are supplied. MIN and MAX are optional.

= 0 when TN, the total number of items; NC, the number of items chosen; and PROB, the probability that an item of class I will be chosen are supplied. MIN and MAX are optional.

K = 3 to 5

P = (EX,VX,PROB) for L=1

(TN,NC,PROB) for L=0

3) USAGE SAMPLES:

CALL DTIME(20,1,3,P,VAL) --- Input with EX, VX and PROB.

CALL DTIME(20,1,4,P,VAL) --- Input with EX, VX, PROB and MIN.

CALL DTIME(20,0,5,P,VAL) --- Input with TN, NC, PROB, MIN and MAX.

C. GEOMETRIC DISTRIBUTION

1) SELECTION:

Direct method is selected for DTIME because it gave better results in the chi-square test and its execution time is less than Naylor's method.

2) INPUT ARGUMENTS:

KODE = 30

L = 1 when EX and VX are supplied. MIN and MAX are optional.

= 0 when PROB, the probability of the occurrence of the event is supplied. MIN and MAX are optional.

K = 1 to 4

P = (EX,VX) for L=1

(PROB) for L=0

3) USAGE SAMPLES:

```
CALL DTIME(30,1,2,P,VAL) --- Input with EX and VX.
CALL DTIME(30,1,3,P,VAL) --- Input with EX, VX ~~AND~~ and MIN.
CALL DTIME(30,0,3,P,VAL) --- Input with PROB, MIN and MAX.
```

D. PASCAL DISTRIBUTION

1) SELECTION:

Negative binomial is incorporated with non-integer method for DTIME. Negative binomial has the best chi-square test results and non-integer enables users to supply non-integer number of successes.

2) INPUT ARGUMENTS:

KODE = 40

L = 1 when EX and VX are supplied. MIN and MAX are optional.

= 0 when the number of successes, TNS and the probability that the event will occur, PROB are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EX,VX) for L=1

(TNS,PROB) for L=0

3) USAGE SAMPLES:

```
CALL DTIME(40,1,3,P,VAL) --- Input with EX,VX and MIN
CALL DTIME(40,0,3,P,VAL) --- Input with TNS, PROB and
                             MIN
CATT DTIME(40,0,4,P,VAL) --- Input with TNS,PROB, MIN
                             and MAX.
```

E. BINOMIAL DISTRIBUTION

1) SELECTION:

Both the Direct and Normal Approximation methods are incorporated into DTIME. The number of trials, NT is tested first. If NT is greater than 15, the Normal Approximation method is used to generate a binomial random variate. Otherwise, the Direct method is used.

2) INPUT ARGUMENTS:

KODE = 50

L = 1 when EX and VX are supplied. MIN and MAX are optional.

= 0 when the total number of trials NT, and the probability of occurrence of the event PROB are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EX,VX) for L=1

(TN,PROB) for L=0

3) USAGE SAMPLES:

```
CALL DTIME(50,1,3,P,VAL) --- Input with EX, VX and MIN.
CALL DTIME(50,1,4,P,VAL) --- Input with EX, VX, MIN and
                             MAX.
CALL DTIME(50,0,2,P,VAL) --- Input with TN and PROB.
```

F. POISSON DISTRIBUTION

1) SELECTION:

Exponential arrival time method is chosen for DTIME because it gave better chi-square test results.

2) INPUT ARGUMENTS:

KODE = 60

L = 1 or 0 when λ, the arrival rate per unit time; and the number or time units, T are supplied. MIN and MAX are optional

K = 2 to 4

P = (λ,T) L=1 or 0

3) USAGE SAMPLES:

CALL DTIME(60,1,2,P,VAL) --- Input with λ and T.

CALL DTIME(60,0,2,P,VAL) --- Input with λ and T.

CALL DTIME(60,1,3,P,VAL) --- Input with λ,T and MIN.

USAGE SUMMARY

DISTRIBUTIONS	KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	...	
ANY DISCRETE	10	1	2 to 50	x_1	p_1	x_2	p_2	...	MIN	MAX
		0	2 to 50	x_1	p_1	x_2	p_2	...		
HYPERGEOMETRIC	20	1	2 to 5	EX	VX	PROB	MIN	MAX		
		0	2 to 5	TN	NC	RROB	MIN	MAX		
GEOMETRIC	30	1	2 to 4	EX	VX	MIN	MAX			
		0	1 to 4	PROB	MIN	MAX				
PASCAL	40	1	2 to 4	EX	VX	MIN	MAX			
		0	2 to 4	TNX	PROB	MIN	MAX			
BINORMAL	50	1	2 to 4	EX	VX	MIN	MAX			
		0	2 to 4	NT	PROB	MIN	MAX			
POISSON	60	1	2 to 4	λ	T	MIN	MAX			
		0	2 to 4	λ	T	MIN	MAX			

DTIME PROGRAM RESULTS

INPUT N=100													OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	P(7)	P(8)	P(9)	P(10)	EX	VX
10	1	9	1.00	0.20	2.00	0.30	3.00	0.10	4.00	0.40	1.50		3.080	0.814
10	1	10	0.20	2.00	0.30	3.00	0.10	4.00	0.40	1.50	3.40		2.280	0.202
10	0	10	1.00	0.10	2.00	0.20	3.00	0.15	4.00	0.30	5.00	0.25	3.560	1.430
20	1	3	4.00	0.60	0.45								3.550	0.428
20	1	4	4.00	0.60	0.45	2.30							4.110	0.098
20	1	5	4.00	0.60	0.45	1.00	3.50						2.920	0.074
20	0	3	10.00	4.00	0.45								1.840	0.674
20	0	4	10.00	4.00	0.45	0.96							1.910	0.542
20	0	5	10.00	4.00	0.45	0.96	2.80						1.530	0.249
30	1	2	3.20	12.40									4.510	12.850
30	1	3	3.20	12.40	2.10								6.300	9.430
30	1	4	3.20	12.40	2.10	5.20							3.750	0.527
30	0	1	0.21										5.250	17.287
30	0	2	0.21	2.10									7.050	12.868
30	0	3	0.21	2.10	4.30								3.500	0.250

DTIME PROGRAM RESULTS

		INPUT		N=100			OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	EX	VX
40	1	2	12.94	35.40			12.834	3.120
40	1	3	12.94	35.40	10.00		13.091	2.647
40	1	4	12.94	35.40	10.00	12.40	11.481	0.381
40	0	2	7.00	0.80			1.972	0.276
40	0	3	7.00	0.80	2.10		2.421	0.106
40	0	4	7.00	0.80	2.10	2.76	2.351	0.031
40	0	2	4.20	0.24			12.943	4.466
40	0	3	4.20	0.24	10.50	17.00	13.304	2.993
40	0	4	4.20	0.24	10.50	17.00	13.315	2.606
50	1	2	3.20	0.80			3.070	0.645
50	1	3	3.20	0.80	1.20		3.110	0.578
50	1	4	3.20	0.80	1.20	3.50	2.690	0.214
50	0	2	12.94	0.24			4.030	1.049
50	0	3	12.94	0.24	2.10		2.780	2.852
50	0	4	12.94	0.24	2.10	4.00	3.430	0.245

DTIME PROGRAM RESULTS

			INPUT	N=100			OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	EX	VX
50	0	2	20.0	0.33			6.597	3.995
50	0	3	20.0	0.33	3.80		6.889	2.736
50	0	4	20.0	0.33	3.80	7.20	5.656	0.807
60	0	2	3.2	1.00			3.310	2.614
60	0	3	3.2	1.00	1.20		3.100	0.590
60	0	4	3.2	1.00	1.20	4.00	3.860	1.820

2. SUBROUTINE CTIME

Subroutine CTIME(KODE,L,K,P,VAL) is used to generate random values from any continuous distribution discussed in Chapter II. The 5 arguments are:

KODE = code for the distribution, can be 10,20,30,40, ...,120.

L = 1 or 0 to indicate the nature of vector P. See individual input arguments sections for detail

K = dimension of vector P.

P = vector giving the mean, variance, minimum, maximum, or other parameters as required. See individual input arguments sections for details.

VAL = the returned random variate.

A. EXPONENTIAL DISTRIBUTION

1) SELECTION:

Inversion method is used in CTIME. It gave slightly better results in the chi-square test than Marsaglia method, and definitely gave better results than Von Neumann's two methods.

2) INPUT ARGUMENTS:

KODE = 10

L = 1 when EX is supplied. If right skew is required, SKEW, MIN and MAX are optional inputs. If left skew is required, supply SKEW=0.0. MIN and MAX must also be supplied.

= 0 when α is supplied. If right skew is required, SKEW, MIN and MAX are optional inputs. If left skew is required, supply SKEW=0.0, MIN and MAX must also be supplied.

K = 1 to 4

P = (EX) for L=1 and right skew

= (EX,SKEW,MIN,MAX) for L=1 and left skew

= (α) for L=0 and right skew

= (α,SKEW,MIN,MAX) for L=1 and left skew

3) USAGE SAMPLES:

CALL CTIME(10,1,1,P,VAL) --- Input with EX.

CALL CTIME(10,0,4,P,VAL) --- Input with α, SKEW, MIN and MAX.

B. NORMAL DISTRIBUTION

1) SELECTION:

Teichrow's method is selected for CTIME because it gave the best chi-square test results.

2) INPUT ARGUMENTS:

KODE = 20

L = 1 or 0 when EX and VX are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EX,VX)

3) USAGE SAMPLES:

CALL CTIME(20,1,2,P,VAL) --- Input with EX and VX.

CALL CTIME(20,0,2,P,VAL) --- Input with EX and VX.

CALL CTIME(20,0,4,P,VAL) --- Input with EX,VX,MIN and MAX.

C. TRIANGULAR DISTRIBUTION

1) SELECTION:

Inversion method is used in CTIME because it takes less execution time and has given better chi-square test results than rejection method.

2) INPUT ARGUMENTS:

KODE = 30

L = 1 or 0 when the three points x_1, x_2 and x_3 of the distribution are given. MIN and MAX are optional.

K = 3 to 5

P = (x_1, x_2, x_3)

3) USAGE SAMPLES:

CALL CTIME(30,0,3,P,VAL) --- Input with x_1, x_2 and x_3.

CALL CTIME(30,1,3,P,VAL) --- Input with x_1, x_2 and x_3.

CALL CTIME(30,1,5,P,VAL) --- Input with x_1, x_2, x_3 MIN and MAX.

D. TRAPEZOIDAL DISTRIBUTION

1) SELECTION:

Inversion method is used in CTIME because it takes less execution time and has given better chi-square test results than rejection method.

2) INPUT ARGUMENTS:

KODE = 40

L = 1 or 0 when the four points x_1, x_2, x_3 and x_4 of the distriubtion are given. MIN and MAX are optional.

K = 4 to 6

P = (x_1, x_2, x_3, x_4)

3) USAGE SAMPLES:

CALL CTIME(40,0,4,P,VAL) --- Input with x_1, x_2, x_3 and x_4 .

CALL CTIME(40,1,4,P,VAL) --- Input with x_1, x_2, x_3 and x_4 .

CALL CTIME(40,1,5,P,VAL) --- Input with x_1, x_2, x_3, x_4 and MIN.

E. LOGNORMAL DISTRIBUTION

1) SELECTION:

Naylor method is used for CTIME because it was the only method investigated.

2) INPUT ARGUMENTS:

KODE = 50

L = 1 when the mean, EY and the variance, VY of the function y=Log(x) are supplied. MIN and MAX are optional.

0 when the EX and VX of x are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EY,VY) for L=1

(EX,VX) for L=0

3) USAGE SAMPLES:

```
CALL CTIME(50,1,2,P,VAL) --- Input with EY and VY
CALL CTIME(50,1,3,P,VAL) --- Input with EY, VY and MIN.
CALL CTIME(50,0,4,P,VAL) --- Input with EX, VX, MIN and
                             MAX.
```

F. CHI-SQUARE DISTRIBUTION

1) SELECTION:

Combination method is chosen for CTIME for input with non-integer degrees of freedom.

2) INPUT ARGUMENTS:

KODE = 60

L = 1 or 0 when the degrees of freedmm N, or EX is supplied. MIN and MAX are optional.

K = 1 to 3

P = (N) L=1 or 0

(EX) L=1 or 0

3) USAGE SAMPLES:

```
CALL CTIME(60,1,1,P,VAL) --- Input with N or EX.
CALL CTIME(60,0,1,P,VAL) --- Input with EX or N.
CALL CTIME(60,0,3,P,VAL) --- Input with N or EX, MIN and
                             MAX.
```

G. BETA DISTRIBUTION

1) SELECTION:

Chi-square method is selected for CTIME because it gave better results in T test than ordering method.

2) INPUT ARGUMENTS:

KODE = 70

L = 1 when EX and VX are supplied. MIN and MAX are optional.

0 when the degrees of freedom, M and N are supplied MIN and MAX are optional.

K = 2 to 4

P = (EX,VX) for L=1

(M,N) for L=0

3) USAGE SAMPLES:

```
CALL CTIME(70,1,3,P,VAL) --- Input with EX,VX and MIN.
CALL CTIME(70,0,2,P,VAL) --- Input with M and N.
CALL CTIME(70,0,4,P,VAL) --- Input with M,N,MIN and MAX.
```

H. F DISTRIBUTION

1) SELECTION:

To make it consistent with beta distribution, chi-square method is chosen for CTIME. Both chi-square and ordering methods gave almost equal results in T test.

2) INPUT ARGUMENTS:

KODE = 80

L = 1 when EX and VX are supplied. MIN and MAX are optional.

0 when the degrees of freedom, M and N are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EX,VX) for L=1

(M,N) for L=0

3) USAGE SAMPLES:

```
CALL CTIME(80,1,3,P,VAL) --- Input with EX,VX and MIN.
CALL CTIME(80,0,2,P,VAL) --- Input with M and N.
Call CTIME(80,0,4,P,VAL) --- Input with M,N,MIN and MAX.
```

I. T DISTRIBUTION

1) SELECTION:

Chi-square method is chosen for CTIME because it gave better results in T test than Cacoullos method.

2) INPUT ARGRMENTS:

KODE = 90

L = 1 when EX and VX are supplied. MIN and MAX are optional.
0 when the degrees of freedom, N is supplied. MIN and MAX are optional.

K = 1 to 4

P = (EX,VX) for L=1
(N) for L=0

3) USAGE SAMPLES:

```
CALL CTIME(90,1,3,P,VAL) --- Input with EX,VX and MIN.
CALL CTIME(90,0,1,P,VAL) --- Input with N.
CALL CTIME(90,0,3,P,VAL) --- Input with N,MIN and MAX.
```

J. Z DISTRIBUTION

1) SELECTION:

Chi-square method is chosen for CTIME because it is the only method investigated in Chapter II.

2) INPUT ARGUMENTS:

KODE = 100

L = 1 when EX and VX of e are supplied. MIN and MAX are optional. x is the z-variate.
  0 when the degrees of freedom, M and N of x are supplied. MIN and MAX are optional and x is the z-variate.

K = 2 to 4

P = (EX,VX) for L=1
  (M,N) for L=0

3) USAGE SAMPLES:

```
CALL CTIME(100,1,2,P,VAL) --- Input with EX and VX.
CALL CTIME(100,1,4,P,VAL) --- Input with EX,VX,MIN and MAX.
CALL CTIME(100,0,3,P,VAL) --- Input with M,N and MIN.
```

K. GAMMA DISTRIBUTION

1) SELECTION:

All three methods are cooperated in CTIME. The input shape parameter, α is tested. If it is an integer, an erlang variate is generated. If it is less than one, Jokn method is used to obtain a gamma variate. If it is a real number greater than one, the combination method is used.

2) INPUT ARGUMENTS:

KODE = 110

L = 1 when EX,VX and the location parameter,γ are supplied. MIN and MAX are optional.

= 0 when the shape parameter α, the scale parameter β and the location parameter γ are supplied. MIN and MAX are optional.

K = 3 to 5

P = (EX,VX,) for L=1

(α,β,γ) for L=0

3) USAGE SAMPLES:

CALL CTIME(110,1,3,P,VAL) --- Input with EX,VX and γ.

CALL CTIME(110,1,5,P,VAL) --- Input with EX,VX,γ,MIN and MAX.

CALL CTIME(110,0,4,P,VAL) --- Input with α,β and γ.

L. PARETO DISTRIBUTION

1) SELECTION:

Inversion method is used in CTIME because it is the only method investigated in Chapter II.

2) INPUT ARGUMENTS:

KODE = 120

L = 1 when EX and VX are supplied. MIN and MAX are optional.

= 0 when the shape parameter a and the minimum income k are supplied. MIN and MAX are optional.

K = 2 to 4

P = (EX,VX) for L=1

(a,k) for L=0

3) USAGE SUMPLES:

```
CALL CTIME(120,1,2,P,VAL) --- Input with EX and VX.
CALL CTIME(120,1,4,P,VAL) --- Input with EX, VX, MIN and MAX.
CALL CTIME(120,0,3,P,VAL) --- Input with a, k and MIN.
```

<u>USAGE SUMMARY</u>

DISTRIBUTIONS	KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)
EXPONENTIAL	10	1	1 to 4	EX	skew	MIN	MAX		
		0	1 to 4	α	skew	MIN	MAX		
NORMAL	20	1	2 to 4	EX	VX	MIN	MAX		
		0	2 to 4	EX	VX	MIN	MAX		
TRIANGULAR	30	1	3 to 5	x_1	x_2	x_3	MIN	MAX	
		0	3 to 5	x_1	x_2	x_3	MIN	MAX	
TRAPEZODIAL	40	1	4 to 6	x_1	x_2	x_3	x_4	MIN	MAX
		0	4 to 6	x_1	x_2	x_3	x_4	MIN	MAX
LOGNORM~~AL~~	50	1	2 to 4	EY	VY	MIN	MAX		
		0	2 to 4	EX	VX	MIN	MAX		
CHI SQUARE	60	1	1 to 3	EX	MIN	MAX			
		0	1 to 3	N	MIN	MAX			

USAGE SUMMARY

DISTRIBUTION	KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)
BETA	70	1	2 to 4	EX	VX	MIN	MAX		
		0	2 to 4	M	N	MIN	MAX		
F	80	1	2 to 4	EX	VX	MIN	MAX		
		0	2 to 4	M	N	MIN	MAX		
T	90	1	2 to 4	EX	VX	MIN	MAX		
		0	1 to 3	N	MIN	MAX			
Z	100	1	2 to 4	EX(e^{2X})	VX(e^{2X})	MIN	MAX		
		0	2 to 4	M	N	MIN	MAX		
GAMMA & ERLANG	110	1	3 to 5	EX	VX	γ	MIN	MAX	
		0	3 to 5	α	β	γ	MIN	MAX	
PARETO	120	1	2 to 4	EX	VX	MIN	MAX		
		0	2 to 4	a	k	MIN	MAX		

CTIME PROGRAM RESULTS

INPUT N=100								OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	EX	VX
10	1	1	6.0					5.113	20.445
10	1	3	6.0	1.0	1.2			5.261	14.198
10	1	4	6.0	0.0	-0.35	10.0		6.853	6.299
10	1	4	6.0	1.0	1.2	12.4		4.828	8.908
10	1	1	76.9					65.536	3358.492
10	1	3	76.9	1.0	20.0			68.492	1838.770
10	1	4	76.9	1.0	20.0	99.0		50.775	479.238
10	1	4	76.9	0.0	10.5	85.4		78.096	37.398
10	0	1	0.85					1.003	00.786
10	0	3	0.85	1.0	1.2			1.150	0.018
10	0	4	0.85	1.0	1.2	2.76		1.150	0.018
10	0	1	0.012					71.019	3943.934
10	0	3	0.012	1.0	10.5			72.019	3054.223
10	0	4	0.012	0.0	10.5	85.4		83.579	1.734

CTIME PROGRAM RESULTS

INPUT N=100									OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)	EX	VX
20	1	2	0.0	1.0					-0.112	1.159
20	1	3	0.0	1.0	-2.55				-0.087	1.083
20	1	4	0.0	1.0	-2.55	2.76			-0.117	0.990
30	0	3	0.0	4.3					1.897	0.831
30	0	4	0.0	4.3	0.88				2.212	0.535
30	0	5	0.0	4.3	0.88	4.2			2.212	0.535
40	0	4	0.0	1.0	4.3	6.5			2.4	2.938
40	0	5	0.0	1.0	4.3	6.5	0.6		2.746	2.616
40	0	6	0.0	1.0	4.3	6.5	0.6	6.0	2.736	2.558
50	1	2	0.0	1.0					1.586	5.207
50	1	3	0.0	1.0	0.6				2.396	7.164
50	1	4	0.0	1.0	0.6	6.5			1.924	1.624
50	0	2	3.2	0.8					3.122	0.879
50	0	3	3.2	0.8	2.10				3.307	0.753
50	0	4	3.2	0.8	2.10	4.0			3.307	0.753

CTIME PROGRAM RESULTS

INPUT			N=100					OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	p(5)	EX	VX
60	1	1	4.2					4.098	0.955
60	1	2	4.2	2.88				4.252	0.801
60	1	3	4.2	2.88	4.2			3.582	0.153
70	1	2	0.835	0.012				0.855	0.012
70	1	3	0.835	0.012	0.6			0.856	0.007
70	1	4	0.835	0.012	0.6	0.88		0.795	0.004
70	0	2	12.54	3.0				0.835	0.012
70	0	3	12.54	3.0	0.6			0.842	0.010
70	0	4	12.54	3.0	0.6	0.88		0.765	0.006
80	1	2	1.5	3.38				1.479	3.333
80	1	3	1.5	3.38	0.6			1.787	3.262
80	1	4	1.5	3.38	0.6	2.76		1.303	0.247
80	0	2	3.2	12.4				1.178	1.220
80	0	3	3.2	12.4	0.6			1.602	1.158
80	0	4	3.2	12.4	0.6	6.5		1.534	0.916

CTIME PROGRAM RESULTS

INPUT			N=100					OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	EX	VX
90	1	2	0.0	4.466				-0.177	4.315
90	1	3	0.0	4.466	-2.55			0.177	4.315
90	1	4	0.0	4.466	.2.55	4.0		-0.169	1.126
90	0	1	4.2					0.096	1.490
90	0	2	4.2	0.012				0.945	0.929
90	0	3	4.2	0.012	4.3			0.890	0.697
100	1	2	1.5	2.38				-0.007	0.131
100	1	3	1.5	2.38	0.0			0.366	0.077
100	1	4	1.5	2.38	0.0	0.7		0.291	0.038
100	0	2	7.0	4.466				-0.038	0.234
100	0	3	7.0	4.466	-0.35			0.231	0.214
100	0	4	7.0	4.466	-0.35	0.098		0.098	0.107
110	1	3	3.2	0.8	0.0			3.863	0.472
110	1	4	3.2	0.8	0.0	2.76		3.935	0.485

CTIME PROGRAM RESULTS

INPUT				N=100				OUTPUT	
KODE	L	K	P(1)	P(2)	P(3)	P(4)	P(5)	EX	VX
110	1	5	3.2	0.8	0.0	2.76	5.1	3.830	0.326
110	0	3	0.83	0.8	0.0			1.129	1.033
110	0	4	0.83	0.8	0.0	0.7		1.862	1.553
110	0	5	0.83	0.8	0.0	0.7	0.6	1.738	1.008
110	0	3	4.2	0.8	0.0			4.883	5.929
110	0	4	4.2	0.8	0.0	1.0		4.883	5.929
110	0	5	4.2	0.8	0.0	1.0	5.0	3.189	0.973
120	1	2	3.2	0.4				3.103	0.175
120	1	3	3.2	0.4	3.1			3.628	0.323
120	1	4	3.2	0.4	3.1	4.0		3.409	0.059
120	0	2	7.0	4.47				5.076	0.353
120	0	3	7.0	4.47	4.2			5.076	0.353
120	0	4	7.0	4.47	4.2	5.9		4.927	0.128

APPENDIX A

```
C***********************************************************************
C                                                                      *
C     TO GENERATE A RANDOM NUMBER OF ONE OF THE DISTRIBUTION BELOW:    *
C                                                                      *
C     DISTRIBUTIONS:                    CODE:                          *
C                                                                      *
C     ANY DISCRETE                       10                            *
C                                                                      *
C     HYPERGEOMETRIC                     20                            *
C                                                                      *
C     GEOMETRIC                          30                            *
C                                                                      *
C     PASCAL                             40                            *
C                                                                      *
C     BINOMIAL                           50                            *
C                                                                      *
C     POISSON                            60                            *
C                                                                      *
C     SUBROUTINE REQUIRED: NORM, ERROR AND RAN(NSEED)                  *
C                                                                      *
C     KODE = THE CODE OF THE DISTRIBUTION                              *
C                                                                      *
C     L = 0 OR 1 TO SPECIFY THE NATURE OF VECTOR P                     *
C                                                                      *
C     K = DIMENSION OF P                                               *
C                                                                      *
C     P = VECTOR WITH VALUES OF MEAN, VARIANCE, OR OTHER PARAMERERS OF *
C         THE DISTRIBUTION                                             *
C                                                                      *
C     VAL = THE RETURNED VARIATE                                       *
C                                                                      *
C***********************************************************************
C
      SUBROUTINE DTIME(KODE,L,K,P,VAL)
      DIMENSION P(50)
```

```
      DATA NSEED/567801/
      IF(KODE.EQ.10) GO TO 10
      IF(KODE.EQ.20) GO TO 20
      IF(KODE.EQ.30) GO TO 30
      IF(KODE.EQ.40) GO TO 40
      IF(KODE.EQ.50) GO TO 50
      IF(KODE.EQ.60) GO TO 60
C
C          ANY DISCRETE
C
   10 M=0
      IF(L.EQ.0) IK=K
      TET=(-1.0)**K
      IF(L.EQ.1.AND.TET.EQ.1.0) IK=K-2
      IF(L.EQ.1.AND.TET.EQ.-1.0) IK=K-1
   11 R=RAN(NSEED)
      LL=K
      M=M+1
      IF(M.EQ.100) CALL ERROR(10)
      COMP=0.0
      DO 12 I=2,IK,2
      J=I-1
      COMP=P(I)+COMP
      IF(COMP.GT.1.0) CALL ERROR(11)
      IF(R.LT.COMP) GO TO 13
   12 CONTINUE
   13 X=P(J)
      IF(LL.EQ.IK) GO TO 15
      IF(TET.EQ.-1.0) GO TO 14
      IF(X.GT.P(LL)) GO TO 11
      LL=K-1
   14 IF(X.LT.P(LL)) GO TO 11
   15 VAL=X
      GO TO 1
C
C          HYPERGEOMETRIC
```

```
C
   20 M=0
      IF(L.EQ.0) GO TO 21
      NC=P(1)/P(3)
      IF(NC.LT.1) CALL ERROR(21)
      NN=(P(2)-P(1)*(1.-P(3))*NC)/(P(2)-P(1)*(1.-P(3)))
      TN=NN
      IF(TN.LT.1) CALL ERROR(22)
      IF(NN.LT.NC) CALL ERROR(23)
      GO TO 22
   21 NC=P(2)
      TN=P(1)
   22 PX=P(3)
      PTN=TN
      X=0.0
      M=M+1
      IF(M.EQ.100) CALL ERROR(20)
      DO 26 I=1,NC
      R=RAN(NSEED)
      IF(R-PX) 23,23,24
   23 S=1.0
      X=X+1.0
      GO TO 25
   24 S=0.0
   25 PX=(TN*PX-S)/(TN-1.0)
      TN=TN-1.0
   26 CONTINUE
      IF(K.EQ.3) GO TO 28
      IF(K.EQ.4) GO TO 27
      TN=PTN
      IF(X.GT.P(5)) GO TO 22
   27 TN=PTN
      IF(X.LT.P(4)) GO TO 22
   28 VAL=X
      GO TO 1
C
```

```
C           GEOMETRIC
C
   30 M=0
      IF(L.EQ.0) GO TO 35
      PROB=P(1)/P(2)
   31 M=M+1
      IF(M.EQ.100) CALL ERROR(30)
      X=0.0
   32 R=RAN(NSEED)
      IF(R.LT.PROB) GO TO 33
      X=X+1.
      GO TO 32
   33 X=X+1.0
      IF(K.EQ.2) GO TO 39
      IF(K.EQ.3) GO TO 34
      IF(X.GT.P(4)) GO TO 31
   34 IF(X.LT.P(3)) GO TO 31
      GO TO 39
   35 M=M+1
      IF(M.EQ.100) CALL ERROR(30)
      X=0.0
   36 R=RAN(NSEED)
      IF(R.LT.P(1)) GO TO 37
      X=X+1.0
      GO TO 36
   37 X=X+1.0
      IF(K.EQ.1) GO TO 39
      IF(K.EQ.2) GO TO 38
      IF(X.GT.P(3)) GO TO 35
   38 IF(X.LT.P(2)) GO TO 35
   39 VAL=X
      IF(X.EQ.0.0) GO TO 35
      GO TO 1
C
C        PASCAL
C
```

```
   40 M=0
C           TO TEST IF THE NUMBER OF SUCCESSES IS AN INTEGER OR NOT
      IF(L.EQ.0) GO TO 41
      PROB=P(1)/P(2)
      IF(PROB.GT.1.0) CALL ERROR(41)
      TNS=P(1)*PROB/(1.0-PROB)
      IF(TNS.LE.1.0) CALL ERROR(42)
      GO TO 42
   41 PROB=P(2)
      TNS=P(1)
      IF(TNS.LE.1.0) CALL ERROR(42)
   42 T=1.0
   43 T=T+1.0
      M=M+1
      IF(M.EQ.100) CALL ERROR(40)
      IF(TNS.EQ.T) GO TO 44
      IF(TNS.GT.T) GO TO 47
      GO TO 43
   44 TN=0.0
C      NUMBER OF SUCCESSES IS AN INTEGER
      TM=0.0
   45 R=RAN(NSEED)
      IF(R.LE.PROB) TM=TM+1.0
      TN=TN+1.0
      IF(TM.EQ.TNS) GO TO 46
      GO TO 45
   46 X=TN-TM
      GO TO 444
C      NUMBER OF SUCCESSES IS NOT AN INTEGER
   47 NT=TNS
      TN=NT
      NK=(TNS-NT)*10
      SUM=0.0
      SN=0.0
      SI=0.0
      SL=TN+1.0
```

```
      DO 440 I=1,NK
   48 R=RAN(NSEED)
      IF(R.LE.PROB) SI=SI+1.0
      SN=SN+1.0
      IF(SI.EQ.SL) GO TO 49
      GO TO 48
   49 SUM=SUM+(SN-SI)
      SN=0.0
      SI=0.0
  440 CONTINUE
      KK=10-NK
      DO 443 I=1,KK
  441 R=RAN(NSEED)
      IF(R.LE.PROB) SI=SI+1.0
      SN=SN+1.0
      IF(SI.EQ.TN) GO TO 442
      GO TO 441
  442 SUM=SUM+(SN-SI)
      SN=0.0
      SI=0.0
  443 CONTINUE
      X=SUM/(NK+KK)
  444 IF(K.EQ.2) GO TO 446
      IF(K.EQ.3) GO TO 445
      IF(X.GT.P(4)) GO TO 42
  445 IF(X.LT.P(3)) GO TO 42
  446 VAL=X
      GO TO 1
C
C          BINOMIAL
C
   50 M=0
      IF(L.EQ.0) GO TO 51
      PROB=1.0-P(2)/P(1)
      IF(PROB.GT.1.0) CALL ERROR(51)
      NT=P(1)/PROB
```

```
      GO TO 52
   51 NT=P(1)
      IF(NT.LT.1.0) CALL ERROR(52)
      PROB=P(2)
   52 M=M+1
      IF(M.EQ.100) CALL ERROR(50)
      IF(NT.GT.15) GO TO 55
      X=0.0
      DO 54 I=1,NT
      R=RAN(NSEED)
      IF(R-PROB) 53,53,54
   53 X=X+1.0
   54 CONTINUE
      GO TO 56
   55 A=PROB*NT
      B=PROB*NT*(1.-PROB)
      CALL NORM(A,B,X)
   56 IF(K.EQ.2) GO TO 58
      IF(K.EQ.3) GO TO 57
      IF(X.GT.P(4)) GO TO 52
   57 IF(X.LT.P(3)) GO TO 52
   58 VAL=X
      GO TO 1
C
C         POISSON
C
   60 M=0
      N=P(2)
   61 M=M+1
      IF(M.EQ.100) CALL ERROR(60)
      X=0.0
      DO 64 I=1,N
      TEST=EXP(-P(1))
      TR=1.0
   62 TR=TR*RAN(NSEED)
      IF(TR-TEST) 64,63,63
```

```
63 X=X+1.0                                                          DTIM252
   GO TO 62                                                         DTIM253
64 CONTINUE                                                         DTIM254
   IF(K.EQ.2) GO TO 66                                              DTIM255
   IF(K.EQ.3) GO TO 65                                              DTIM256
   IF(X.GT.P(4)) GO TO 61                                           DTIM257
65 IF(X.LT.P(3)) GO TO 61                                           DTIM258
66 VAL=X                                                            DTIM259
   GO TO 1                                                          DTIM260
 1 RETURN                                                           DTIM261
   END                                                              DTIM262
```

```
C*********************************************************************ERRO
C                                                                    *ERRO
C     TO GIVE THE K-TH ERROR MESSAGE                                 *ERRO
C                                                                    *ERRO
C     K = THE NUMBER FOR THE ERROR                                   *ERRO
C                                                                    *ERRO
C*********************************************************************ERRO
C                                                                     ERRO
      SUBROUTINE ERROR(K)                                             ERRO
      IF(K.EQ.10) WRITE(6,10)                                         ERRO
      IF(K.EQ.11) WRITE(6,11)                                         ERRO
      IF(K.EQ.20) WRITE(6,20)                                         ERRO
      IF(K.EQ.21) WRITE(6,21)                                         ERRO
      IF(K.EQ.22) WRITE(6,22)                                         ERRO
      IF(K.EQ.23) WRITE(6,23)                                         ERRO
      IF(K.EQ.30) WRITE(6,30)                                         ERRO
      IF(K.EQ.40) WRITE(6,40)                                         ERRO
      IF(K.EQ.41) WRITE(6,41)                                         ERRO
      IF(K.EQ.42) WRITE(6,42)                                         ERRO
      IF(K.EQ.50) WRITE(6,50)                                         ERRO
      IF(K.EQ.51) WRITE(6,51)                                         ERRO
      IF(K.EQ.52) WRITE(6,52)                                         ERRO
      IF(K.EQ.60) WRITE(6,60)                                         ERRO
   10 FORMAT(' ERROR IN ANY DISCRETE DISTRIBUTION ')                  ERRO
   11 FORMAT('  SUM OF ANY DISCRETE PROBABILITY IS GREATER THAN ONE ')  ERRO
   20 FORMAT(' ERROR IN HYPERGEOMETRIC DISTRIBUTION ')                ERRO
   21 FORMAT('  NUMBER OF ITEMS CHOSEN IN HYPERGEOMETRIC DIST IS LESS   ERRO
     +THAN ONE  ')                                                    ERRO
   22 FORMAT('  TOTAL NUMBER OF ITEMS IN HYPERGEOMETRIC DISTRIBUTION IS ERRO
     + LESS THAN ONE  ')                                              ERRO
   23 FORMAT('  TOTAL NUMBER OF ITEMS IN HYPERGEOMETRIC DISTRIBUTION IS ERRO
     + LESS THAN THE NUMBER OF ITEMS CHOSEN  ')                       ERRO
   30 FORMAT(' ERROR IN GEOMETRIC DISTRIBUTION ')                     ERRO
   40 FORMAT(' ERROR IN PASCAL DISTRIBUTION ')                        ERRO
   41 FORMAT('  THE PROBABILITY P OF PASCAL DISTRIBUTION IS GREATER THANERRO
```

```
  + ONE   ')
42 FORMAT('  THE TOTAL NUMBER OF SUCCESSES IN PASCAL DISTRIBUTION IS
  + LESS THAN ONE  ')
50 FORMAT(' ERROR IN BINOMIAL DISTRIBUTION ')
51 FORMAT('  THE PROBABILITY OF OCCURANCE OF BINOMIAL DISTRIBUTION IS
  + GREATER THAN ONE  ')
52 FORMAT('  THE NUMBER OF TRIALS OF BINOMIAL DISTRIBUTION IS LESS
  + THAN ONE  ')
60 FORMAT(' ERROR IN POISSON DISTRIBUTION')
   RETURN
   END
```

```
C***********************************************************************
C                                                                      *
C     TO RETURN A RANDOM NORMAL VARIATE                                *
C                                                                      *
C     EX = THE MEAN OF THE DISTRIBUTION                                *
C                                                                      *
C     VX = THE VARIANCE OF THE DISTRIBUTION                            *
C                                                                      *
C     VAL = THE RETURNED NORMAL VARIATE                                *
C                                                                      *
C***********************************************************************
C
      SUBROUTINE NORM(EX,VX,VAL)
      DATA NSEED/221133/
      SUM=0.0
      DO 25 I=1,12
      SUM=SUM+RAN(NSEED)
   25 CONTINUE
      VAL=(SUM-6.0)*SQRT(VX)+EX
      RETURN
      END
```

```
C**********************************************************************
C                                                                     *
C     TO RETURN A RANDOM UNIFORM VARIATE IN (0,1)                     *
C                                                                     *
C     NSEED = A 5 DIGITS ODD INTEGER SEED                             *
C                                                                     *
C**********************************************************************
C
      FUNCTION RAN(NSEED)
      NSEED=IABS(NSEED*655393)
      RAN=FLOAT(MOD(NSEED,33554432))/FLOAT(33554432)
      RETURN
      END
```

APPENDIX B

```
C***********************************************************************CTIM   1
C                                                                      *CTIM   2
C     TO GENERATE A RANDOM NUMBER OF ONE OF THE DISTRIBUTION BELOW:    *CTIM   3
C                                                                      *CTIM   4
C     DISTRIBUTIONS:                  CODE:                            *CTIM   5
C                                                                      *CTIM   6
C     EXPONENTIAL                      10                              *CTIM   7
C                                                                      *CTIM   8
C     NORMAL                           20                              *CTIM   9
C                                                                      *CTIM  10
C     TRIANGULAR                       30                              *CTIM  11
C                                                                      *CTIM  12
C     TRAPEZODIAL                      40                              *CTIM  13
C                                                                      *CTIM  14
C     LOGNORMAL                        50                              *CTIM  15
C                                                                      *CTIM  16
C     CHI-SQUARE                       60                              *CTIM  17
C                                                                      *CTIM  18
C     BETA                             70                              *CTIM  19
C                                                                      *CTIM  20
C     F                                80                              *CTIM  21
C                                                                      *CTIM  22
C     T                                90                              *CTIM  23
C                                                                      *CTIM  24
C     Z                               100                              *CTIM  25
C                                                                      *CTIM  26
C     GAMMA                           110                              *CTIM  27
C                                                                      *CTIM  28
C     PARETO                          120                              *CTIM  29
C                                                                      *CTIM  30
C     SUBROUTINE REQUIRED: NORM, ERROR AND RAN(NSEED)                  *CTIM  31
C                                                                      *CTIM  32
C     KODE = THE CODE OF THE DISTRIBUTION                              *CTIM  33
C                                                                      *CTIM  34
C     L = 0 OR 1 TO SPECIFY THE NATURE OF VECTOR P                     *CTIM  35
```

```
C                                                                        *CTIM 36
C      K = DIMENSION OF P                                                *CTIM 37
C                                                                        *CTIM 38
C      P = VECTOR WITH VALUES OF MEAN, VARIANCE, OR OTHER PARAMERERS OF  *CTIM 39
C          THE DISTRIBUTION                                              *CTIM 40
C                                                                        *CTIM 41
C      VAL = THE RETURNED VARIATE                                        *CTIM 42
C                                                                        *CTIM 43
C************************************************************************CTIM 44
C                                                                         CTIM 45
      SUBROUTINE CTIME(KODE,L,K,P,VAL)                                    CTIM 46
      DATA NSEED/567801/                                                  CTIM 47
      DIMENSION P(50)                                                     CTIM 48
      IF(KODE.EQ.10) GO TO 10                                             CTIM 49
      IF(KODE.EQ.20) GO TO 20                                             CTIM 50
      IF(KODE.EQ.30) GO TO 30                                             CTIM 51
      IF(KODE.EQ.40) GO TO 40                                             CTIM 52
      IF(KODE.EQ.50) GO TO 50                                             CTIM 53
      IF(KODE.EQ.60) GO TO 60                                             CTIM 54
      IF(KODE.EQ.70) GO TO 70                                             CTIM 55
      IF(KODE.EQ.80) GO TO 80                                             CTIM 56
      IF(KODE.EQ.90) GO TO 90                                             CTIM 57
      IF(KODE.EQ.100) GO TO 100                                           CTIM 58
      IF(KODE.EQ.110) GO TO 110                                           CTIM 59
      IF(KODE.EQ.120) GO TO 120                                           CTIM 60
C                                                                         CTIM 61
C          EXPONENTIAL                                                    CTIM 62
C                                                                         CTIM 63
   10 M=0                                                                 CTIM 64
      IF(K.EQ.1) P(2)=1.0                                                 CTIM 65
      MIN=0.0                                                             CTIM 66
      IF(L.EQ.0) EX=1.0/P(1)                                              CTIM 67
      IF(L.EQ.1) EX=P(1)                                                  CTIM 68
      IF(K.EQ.1.OR.K.EQ.2) GO TO 11                                       CTIM 69
      MIN=P(3)                                                            CTIM 70
      IF(K.EQ.4) MAX=P(4)                                                 CTIM 71
```

```
   11 IF(P(2).EQ.1.0) EX=EX-MIN
      IF(P(2).EQ.0.0) EX=MAX-EX
   12 R=RAN(NSEED)
      M=M+1
      IF(M.EQ.1CC) CALL ERROR(10)
      X=-EX*ALOG(R)
      IF(P(2).EQ.1.0) X=X+MIN
      IF(P(2).EQ.0.0) X=MAX-X
      IF(K.EQ.1) GO TO 14
      IF(K.EQ.3) GO TO 13
      IF(X.GT.MAX) GO TO 12
   13 IF(X.LT.MIN) GO TO 12
   14 VAL=X
      GO TO 1
C          NORMAL
C
C
   20 M=0
   21 R=RAN(NSEED)/2.0
      M=M+1
      IF(M.EQ.100) CALL ERROR(20)
      IF(R.EQ.0.0) GO TO 20
      CALL NORM(P(1),P(2),Y)
      X=Y
      IF(K.EQ.2) GO TO 23
      IF(K.EQ.3) GO TO 22
      IF(X.GT.P(4)) GO TO 21
   22 IF(X.LT.P(3)) GO TO 21
   23 VAL=X
      GO TO 1
C
C      TRIANGULAR
C
   30 M=0
      FP2=(P(2)-P(1))/(P(3)-P(1))
   31 R=RAN(NSEED)
```

```
      M=M+1                                                             CTIM108
      IF(M.EQ.100) CALL ERROR(30)                                       CTIM109
      IF(R.GT.FP2) GO TO 32                                             CTIM110
      X=P(1)+SQRT(R*(P(3)-P(1))*(P(2)-P(1)))                            CTIM111
      GO TO 33                                                          CTIM112
   32 X=P(1)+SQRT((1.-R)*(P(3)-P(1))*(P(3)-P(2)))                       CTIM113
   33 IF(K.EQ.3) GO TO 35                                               CTIM114
      IF(K.EQ.4) GO TO 34                                               CTIM115
      IF(X.GT.P(5)) GO TO 31                                            CTIM116
   34 IF(X.LT.P(4)) GO TO 31                                            CTIM117
   35 VAL=X                                                             CTIM118
      GO TO 1                                                           CTIM119
C                                                                       CTIM120
C           TRAPEZODIAL                                                 CTIM121
C                                                                       CTIM122
   40 M=0                                                               CTIM123
      H=2./(-P(1)-P(2)+P(3)+P(4))                                       CTIM124
      F2=H*0.5*(P(2)-P(1))                                              CTIM125
      F3=H*0.5*(2.*P(3)-P(1)-P(2))                                      CTIM126
   41 R=RAN(NSEED)                                                      CTIM127
      M=M+1                                                             CTIM128
      IF(M.EQ.100) CALL ERROR(40)                                       CTIM129
      IF(R.GT.F3) X=P(4)-SQRT((P(3)-P(4))*(P(1)+P(2)-P(3)-P(4))*(1.-R)) CTIM130
      IF(R.LE.F3) X=(P(1)-P(2))/2.+R/H                                  CTIM131
      IF(R.LE.F2) X=P(1)+SQRT(2.*R*(P(2)-P(1))/H)                       CTIM132
      IF(K.EQ.4) GO TO 43                                               CTIM133
      IF(K.EQ.5) GO TO 42                                               CTIM134
      IF(X.GT.P(6)) GO TO 41                                            CTIM135
   42 IF(X.LT.P(5)) GO TO 41                                            CTIM136
   43 VAL=X                                                             CTIM137
      CO TO 1                                                           CTIM138
C                                                                       CTIM139
C         LOGNORMAL                                                     CTIM140
C                                                                       CTIM141
   50 M=0                                                               CTIM142
      IF(L.EQ.1) GO TO 51                                               CTIM143
```

```
      SIGMA=ALOG(P(2)/P(1)**2.+1.)
      GMU=ALOG(P(1))-0.5*ALOG(P(2)/P(1)**2.+1.)
      EX=GMU
      VX=SIGMA
      GO TO 53
   51 EX=P(1)
      VX=P(2)
   52 M=M+1
      IF(M.EQ.100) CALL ERROR(50)
   53 CALL NORM(0.0,1.0,Z)
      X=EXP(SQRT(VX)*Z+EX)
      IF(K.EQ.2) GO TO 55
      IF(K.EQ.3) GO TO 54
      IF(X.GT.P(4)) GO TO 52
   54 IF(X.LT.P(3)) GO TO 52
   55 VAL=X
      GO TO 1
C
C             CHI-SQUARE
C
   60 M=0
   61 M=M+1
      IF(M.EQ.100) CALL ERROR(60)
      IA=P(1)
      IB=IA+1
      LB=P(1)*10.0-IA*10
      LA=10-IB
      Z=0.0
      Y=0.0
      IF(IA.EQ.0.0) GO TO 64
      DO 63 I=1,LA
      DO 62 J=1,IA
      CALL NORM(0.0,1.0,XY)
      Z=Z+XY**2
   62 CONTINUE
   63 CONTINUE
```

```
      IF(IB.EQ.0.0) GO TO 666
   64 DO 66 J=1,LB
      DO 65 I=1,IB
      CALL NORM(0.0,1.0,XY)
      Y=Y+XY**2
   65 CONTINUE
   66 CONTINUE
  666 X=(Z+Y)/10.0
      IF(K.EQ.1) GO TO 68
      IF(K.EQ.2) GO TO 67
      IF(X.GT.P(3)) GO TO 61
   67 IF(X.LT.P(2)) GO TO 61
   68 VAL=X
      GO TO 1
C
C          BETA
C
   70 M=0
      IF(L.EQ.0.AND.M.EQ.0) GO TO 72
      MM=2.0*((P(1)**2.-P(1)**3)/P(2)-P(1))
      N=2.0*(1.0-P(1))*(P(1)-P(1)**2-P(2))/P(2)
      GO TO 72
   71 MM=P(1)
      N=P(2)
   72 IF(M.LT.1) GO TO 78
      IF(N.LT.1) GO TO 79
   73 M=M+1
      IF(M.EQ.100) CALL ERROR(71)
      SUMM=0.0
      DO 74 I=1,MM
      CALL NORM(0.0,1.0,U)
      SUMM=SUMM+U**2
   74 CONTINUE
      SUMN=0.0
      DO 75 J=1,N
      CALL NORM(0.0,1.0,V)
```

```
      SUMN=SUMN+V**2                                                    CTIM216
   75 CONTINUE                                                          CTIM217
      X=SUMM/(SUMM+SUMN)                                                CTIM218
      IF(K.EQ.2) GO TO 77                                               CTIM219
      IF(K.EQ.3) GO TO 76                                               CTIM220
      IF(X.GT.P(4)) GO TO 73                                            CTIM221
   76 IF(X.LT.P(3)) GO TO 73                                            CTIM222
   77 VAL=X                                                             CTIM223
      GO TO 1                                                           CTIM224
   78 CALL ERROR(71)                                                    CTIM225
      GO TO 1                                                           CTIM226
   79 CALL ERROR(72)                                                    CTIM227
      GO TO 1                                                           CTIM228
C                                                                       CTIM229
C             F                                                         CTIM230
C                                                                       CTIM231
   80 M=0                                                               CTIM232
      IF(L.EQ.0) GO TO 81                                               CTIM233
      N=2.*P(1)/(P(1)-1.0)                                              CTIM234
      IF(N.LT.4.0) CALL ERROR(83)                                       CTIM235
      MM=2.*P(1)**2*(N-2.)/((N-4.)*P(2)-2.*P(1)**2)                     CTIM236
      GO TO 82                                                          CTIM237
   81 MM=P(1)                                                           CTIM238
      N=P(2)                                                            CTIM239
   82 IF(M.LT.1) CALL ERROR(81)                                         CTIM240
      IF(N.LT.1) CALL ERROR(82)                                         CTIM241
   83 M=M+1                                                             CTIM242
      IF(M.EQ.100) CALL ERROR(80)                                       CTIM243
      SUMM=0.0                                                          CTIM244
      DO 84 I=1,MM                                                      CTIM245
      CALL NORM(0.0,1.0,U)                                              CTIM246
      SUMM=SUMM+U**2                                                    CTIM247
   84 CONTINUE                                                          CTIM248
      SUMN=0.0                                                          CTIM249
      DO 85 J=1,N                                                       CTIM250
      CALL NORM(0.0,1.0,V)                                              CTIM251
```

```
      SUMN=SUMN+V**2
   85 CONTINUE
      X=(SUMM*N)/(SUMN*M)
      IF(K.EQ.2) GO TO 87
      IF(K.EQ.3) GO TO 86
      IF(X.GT.P(4)) GO TO 83
   86 IF(X.LT.P(3)) GO TO 83
   87 VAL=X
      GO TO 1
C
C           T-DISTRIBUTION
C
   90 M=0
      IF(L.EQ.0) GO TO 91
      IF(P(1).GT.0.0.OR.P(1).LT.0.0) CALL ERROR(90)
      N=2.0*P(2)/(P(2)-1.0)
      GO TO 92
   91 N=P(1)
   92 IF(N.LT.1) CALL ERROR(91)
   93 M=M+1
      IF(M.EQ.100) CALL ERROR(92)
      T=0.0
      DO 94 I=1,N
      CALL NORM(0.0,1.0,Y)
      T=T+Y**2
   94 CONTINUE
      CALL NORM(0.0,1.0,Y)
      X=Y/(SQRT(T/N))
      IF(L.EQ.1.AND.K.EQ.2) GO TO 97
      IF(L.EQ.0.AND.K.EQ.1) GO TO 97
      IF(L.EQ.1.AND.K.EQ.3) GO TO 95
      IF(L.EQ.0.AND.K.EQ.2) GO TO 96
      IF(L.EQ.1) LL=4
      IF(L.EQ.0) LL=3
      IF(X.GT.P(LL)) GO TO 93
      IF(L.EQ.0) GO TO 96
```

```
   95 LL=3                                                              CTIM288
      IF(X.LT.P(LL)) GO TO 93                                           CTIM289
      GO TO 97                                                          CTIM290
   96 LL=2                                                              CTIM291
      IF(X.LT.P(LL)) GO TO 93                                           CTIM292
   97 VAL=X                                                             CTIM293
C                                                                       CTIM294
C             Z-DISTRIBUTION                                            CTIM295
C                                                                       CTIM296
  100 M=0                                                               CTIM297
      IF(L.EQ.0) GO TO 103                                              CTIM298
  101 M=M+1                                                             CTIM299
      IF(M.EQ.100) CALL ERROR(101)                                      CTIM300
      IF(L.EQ.0.AND.M.GE.1) GO TO 104                                   CTIM301
      N=2.*P(1)/(P(1)-1.0)                                              CTIM302
      MM=2.*P(1)**2*(N-2.)/((N-4.)*P(2)-2.*P(1)**2)                     CTIM303
      SUMN=0.0                                                          CTIM304
      SUMM=0.0                                                          CTIM305
      DO 102 I=1,N                                                      CTIM306
      CALL NORM(0.0,1.0,U)                                              CTIM307
      SUMN=SUMN+U**2                                                    CTIM308
  102 CONTINUE                                                          CTIM309
      DO 1100 J=1,MM                                                    CTIM310
      CALL NORM(0.0,1.0,V)                                              CTIM311
      SUMM=SUMM+V**2                                                    CTIM312
 1100 CONTINUE                                                          CTIM313
      GO TO 107                                                         CTIM314
  103 N=P(2)                                                            CTIM315
      MM=P(1)                                                           CTIM316
      IF(P(2).LT.1.0) CALL ERROR(101)                                   CTIM317
      IF(P(1).LT.1.0) CALL ERROR(102)                                   CTIM318
  104 SUMM=0.0                                                          CTIM319
      SUMN=0.0                                                          CTIM320
      DO 105 I=1,MM                                                     CTIM321
      CALL NORM(0.0,1.0,U)                                              CTIM322
      SUMM=SUMM+U**2                                                    CTIM323
```

```
  105 CONTINUE                                                    CTIM324
      DO 106 I=1,N                                                CTIM325
      CALL NORM(0.0,1.0,V)                                        CTIM326
      SUMN=SUMN+V**2                                              CTIM327
  106 CONTINUE                                                    CTIM328
      Q=P(1)                                                      CTIM329
      S=P(2)                                                      CTIM330
  107 X=ALOG((N*SUMM)/(MM*SUMN))*0.5                              CTIM331
      IF(K.EQ.2) GO TO 109                                        CTIM332
      IF(K.EQ.3) GO TO 108                                        CTIM333
      IF(X.GT.P(4)) GO TO 101                                     CTIM334
  108 IF(X.LT.P(3)) GO TO 101                                     CTIM335
  109 VAL=X                                                       CTIM336
      GO TO 1                                                     CTIM337
C                                                                 CTIM338
C            GAMMA                                                CTIM339
C                                                                 CTIM340
  110 M=0                                                         CTIM341
      IF(L.EQ.0) GO TO 111                                        CTIM342
      SCALE=P(1)/P(2)                                             CTIM343
      ALPHA=P(1)*SCALE                                            CTIM344
      GO TO 112                                                   CTIM345
  111 SCALE=P(2)                                                  CTIM346
      ALPHA=P(1)                                                  CTIM347
  112 IAL=ALPHA                                                   CTIM348
      IFRAC=(ALPHA-IAL)*10.                                       CTIM349
  113 M=M+1                                                       CTIM350
      IF(M.EQ.100) CALL ERROR(110)                                CTIM351
      IF(IFRAC.EQ.0) GO TO 118                                    CTIM352
      IF(IAL.GT.0) GO TO 118                                      CTIM353
      Z=0.0                                                       CTIM354
      B=1.-ALPHA                                                  CTIM355
      IA=1000000./ALPHA                                           CTIM356
      IB=1000000./B                                               CTIM357
      C=IA/1000000.                                               CTIM358
      D=IB/1000000.                                               CTIM359
```

```
      IF(IAL)116,116,114
  114 DO 115 I=1,IAL
      Z=Z-ALOG(RAN(NSEED))
  115 CONTINUE
  116 Y=RAN(NSEED)**C
      W=RAN(NSEED)**D
      SUM=Y+W
      IF(SUM.LE.1.0) GO TO 117
      GO TO 116
  117 BETA=Y/SUM
      V=-ALOG(RAN(NSEED))
      X=(Z+BETA*V)
      GO TO 1113
  118 X=0.0
      PR1=IFRAC/10.
      R=RAN(NSEED)
      IF(R.GT.PR1) N=IAL
      IF(R.LE.PR1) N=IAL+1
      Z1=0.0
      DO 119 J=1,N
      Z1=Z1-ALOG(RAN(NSEED))/SCALE
  119 CONTINUE
      X=Z1+P(3)
      GO TO 1117
 1113 X=X/SCALE+P(3)
 1117 IF(K.EQ.3) GO TO 1116
      IF(K.EQ.4) GO TO 1115
      IF(X.GT.P(5)) GO TO 113
 1115 IF(X.LT.P(4)) GO TO 113
 1116 VAL=X
      GO TO 1
C
C          PARETO
C
  120 M=0
      IF(L.EQ.0) GO TO 121
```

```
      A=1.0+SQRT(4.0+4.0*P(1)**2/P(2))/2.0                              CTIM396
      PK=P(1)*(A-1.0)/A                                                 CTIM397
      GO TO 122                                                         CTIM398
  121 A=P(1)                                                            CTIM399
      PK=P(2)                                                           CTIM400
  122 IF(A.LE.0.0) GO TO 120                                            CTIM401
  123 M=M+1                                                             CTIM402
      IF(M.EQ.100) CALL ERROR(121)                                      CTIM403
      R=RAN(NSEED)                                                      CTIM404
      X=EXP(ALOG(PK)-ALOG(R)/A)                                         CTIM405
      IF(X.LT.PK) CALL ERROR(122)                                       CTIM406
      IF(K.EQ.2) GO TO 125                                              CTIM407
      IF(K.EQ.3) GO TO 124                                              CTIM408
      IF(X.GT.P(4)) GO TO 123                                           CTIM409
  124 IF(X.LT.P(3)) GO TO 123                                           CTIM410
  125 VAL=X                                                             CTIM411
      GO TO 1                                                           CTIM412
    1 RETURN                                                            CTIM413
      END                                                               CTIM414
```

```
C***********************************************************************ERRO
C      TO GIVE THE K-TH ERROR MESSAGE                                   *ERRO
C                                                                       *ERRO
C      K = THE NUMBER FOR THE ERROR                                     *ERRO
C                                                                       *ERRO
C***********************************************************************ERRO
C                                                                        ERRO
      SUBROUTINE ERROR(K)                                                ERRO
      IF(K.EQ.10) WRITE(6,10)                                            ERRO
      IF(K.EQ.20) WRITE(6,20)                                            ERRO
      IF(K.EQ.30) WRITE(6,30)                                            ERRO
      IF(K.EQ.40) WRITE(6,40)                                            ERRO
      IF(K.EQ.50) WRITE(6,50)                                            ERRO
      IF(K.EQ.60) WRITE(6,60)                                            ERRO
      IF(K.EQ.70) WRITE(6,70)                                            ERRO
      IF(K.EQ.71) WRITE(6,71)                                            ERRO
      IF(K.EQ.72) WRITE(6,72)                                            ERRO
      IF(K.EQ.80) WRITE(6,80)                                            ERRO
      IF(K.EQ.81) WRITE(6,81)                                            ERRO
      IF(K.EQ.82) WRITE(6,82)                                            ERRO
      IF(K.EQ.83) WRITE(6,83)                                            ERRO
      IF(K.EQ.90) WRITE(6,90)                                            ERRO
      IF(K.EQ.100) WRITE(6,100)                                          ERRO
      IF(K.EQ.101) WRITE(6,101)                                          ERRO
      IF(K.EQ.102) WRITE(6,102)                                          ERRO
      IF(K.EQ.110) WRITE(6,110)                                          ERRO
      IF(K.EQ.120) WRITE(6,120)                                          ERRO
      IF(K.EQ.130) WRITE(6,130)                                          ERRO
   10 FORMAT('  ERROR IN EXPONENTIAL DISTRIBUTION  ')                    ERRO
   20 FORMAT('  ERROR IN NORMAL DISTRIBUTION  ')                         ERRO
   30 FORMAT('  ERROR IN TRIANGULAR DISTRIBUTION  ')                     ERRO
   40 FORMAT('  ERROR IN TRAPEZODIAL DISTRIBUTION  ')                    ERRO
   50 FORMAT('  ERROR IN LOGNORMAL DISTRIBUTION  ')                      ERRO
   60 FORMAT('  ERROR IN CHI SQUARE DISTRIBUTION  ')                     ERRO
   70 FORMAT('  ERROR IN BETA DISTRIBUTION  ')                           ERRO
```

```
   71 FORMAT('  DEGREE OF FREEDOM M OF BETA DISTRIBUTION IS LESS THAN
     + ONE  ')
   72 FORMAT('  DEGREE OF FREEDOM N OF BETA DISTRIBUTION IS LESS THAN
     + ONE  ')
   80 FORMAT('  ERROR IN F DISTRIBUTION  ')
   81 FORMAT('  THE DEGREE OF FREEDOM M OF F DISTRIBUTION IS LESS THAN
     + THAN ONE  ')
   82 FORMAT('  THE DEGREE OF FREEDOM N OF F DISTRIBUTICN IS LESS THAN
     + THAN ONE  ')
   83 FORMAT('  DEGREE OF FREEDOM N OF F DISTRIBUTION IS LESS THAN 4  ')
   90 FORMAT('  ERROR IN T DISTRIBUTION  ')
  100 FORMAT('  ERROR IN Z DISTRIBUTION  ')
  101 FORMAT('  DEGREE OF FREEOM N OF Z DISTRIBUTION IS LESS THAN 1  ')
  102 FORMAT('  DEGREE OF FREEDOM M OF Z DISTRIBUTION IS LESS THAN 1  ')
  110 FORMAT('  ERROR IN GAMMA DISTRIBUTION  ')
  120 FORMAT('  ERROR IN ERLANG DISTRIBUTION  ')
  130 FORMAT('  ERROR IN PARETO DISTRIBUTION  ')
      RETURN
      END
```

```
C***********************************************************************NORM
C                                                                      *NORM
C     TO RETURN A RANDOM NORMAL VARIATE                                *NORM
C                                                                      *NORM
C     EX = THE MEAN OF THE DISTRIBUTION                                *NORM
C                                                                      *NORM
C     VX = THE VARIANCE OF THE DISTRIBUTION                            *NORM
C                                                                      *NORM
C     VAL = THE RETURNED NORMAL VARIATE                                *NORM
C                                                                      *NORM 1
C***********************************************************************NORM 1
C                                                                       NORM 1
      SUBROUTINE NORM(EX,VX,VAL)                                        NORM 1
      DATA NSEED/123123/                                                NORM 1
    3 SUM=0.0                                                           NORM 1
      DO 1 I=1,12                                                       NORM 1
      SUM=SUM+RAN(NSEED)                                                NORM 1
    1 CONTINUE                                                          NORM 1
      IF(SUM.LT.2.0.AND.SUM.GT.10.0) GO TO 3                            NORM 1
      R=(SUM-6.)/4.                                                     NORM 2
      VAL=3.949846138*R+0.252408784*R**3+0.076542812*R**5+0.008355968*R NORM 2
     Q**7+0.029899776*R**9                                              NORM 2
      VAL=VAL*SQRT(VX)+EX                                               NORM 2
      RETURN                                                            NORM 2
      END                                                               NORM 2
```

BIBLIOGRAPHY

1. Abramowitz and Stegun Handbook of Mathematical Functions, New York: Dover Pub. Co. (1970).

2. Blom, Gunnar Statistical Estimates and Transformed Beta Variables, New York: John Wiley and Sons (1958).

3. Bofinger, Eve and V.J. "On a Periodic Property of Pseudo-Random Sequences," Asso. for Comp. Mach. J., Vol.5 (1958), pp.261-5.

4. Box, G.E.P. and Muller,M.E. "A Note on the Generation of Random Normal Deviates," Ann. Math. Stat., Vol.28 (1958), pp.610-11.

5. Brown, Robert G. Statistical Forcasting for Inventory Control, New York: McCraw-Hill Book Co. (1959).

6. Busch, R.R. and Mosteller, F. Stochastic Models for Learning. New York: John Wiley and Sons (1955).

7. Butler, James W. "Machine Sampling from Given Probability Distribution" in Symposium on Monte Carlo Methods, Edited by H.A. Meyer, New York: John Wiley and Sons (1956).

8. Cacoullos, Theophilos "A Relation Between t and F-Distributions," Amer. Stat. Asso. J., Vol.60 (1965), pp.528-31.

9. Cannon, L.E. III Pseudo Random Number Generators for Statistical Applications, Masters Thesis, University of Georgia, Athens, Georgia (1971).

10. Certaine, J. "On Sequences of Pseudo-Random Numbers of Maximal Length," Asso. for Comp. Mach. J., Vol.5 (1958), pp.353-56.

11. Feller, William An Introduction to Probability Theory and Its Applications, New York: John Wiley and Sons (1950).

12. Greenberger, Martin "Notes on a New Pseudo-Random Number Generator," Asso. for Comp. Mach. J., Vol.8 (1961), pp.163-7.

13. Gorenstein, S. "Testing a Random Number Generator," Comm. of the ACM, Vol.10 (1967), pp.111-8.

14. Hald, A. Statistical Theory with Engineering Applications, New York: John Wiley and Sons, Inc. (1952).

15. Hasting, C. Jr. Approximations for Digital Computers, Princeton: University Press (1955).

16. Hoeffding, Wassily "On the Distribution of the Expected Values of the Order Statistics," Ann. of Math. Stat., Vol.24 (1953), pp.93-100.

17. Knuth, Donald E. The Art of Computer Programming 2d ed., Mass: Addison Wesley Pub. Co. (1973).

18. Kotz, Samuel Distribution in Statistics, Boston: Houghton Miffin (1969).

19. Lehmer, D.H. "Mathematical Methods in Large Scale Computing Units," Symposium on Large-Scale Digital Calculating Machinary, Harvard University (1949), pp. 141-6.

20. MacLaren, M. Donald and marsaglia, George "Uniform Random Number Generators," Asso. for Comp. Mach. J., Vol.12 (1965), pp.83-9.

21. Mardaglia, G., MacLaren, M.D. and Bray, T.A. "A Fast Procedure for Generating Normal Random Variables," Comm. of the ACM, Vol.7 (1964), pp.4-10.

22. Marsaglia, G. and Bray, T.A. "One-Line Random Number Generators and Their Use in Combinations," Comm. of the ACM, Vol.11 (1968), pp.757-9.

23. Marsaglia, G. "Generating Discrete Random Variables in a Computer," Comm. of the ACM, Vol. (1963), pp.37-8.

24. Marsaglia, G. "Generating Exponential Random Variables," Ann. Math. Stat., Vol.32 (1961), pp.899.

25. Mathur, R.K. "A Note on the Wilson-Hilferty Transformation of Chi-square," Bull. Calcutta Stat. Assoc., Vol.10 (1961), pp.103-5.

26. Metropolis, N. "Phase Shifts-Middle Squares-Wave Equation" in Symposium on Monte Carlo Methods, Edited by H.A. Meyer, New York: John Wiley and Sons (1956).

27. Meyer, P.L. Introductory Probability and Statistical Applications, Addison Wesley Pub. Co. (1970).

28. Muller, Mervin E. "A Comparison of Methods for Generating Normal Deviates on Digital Computers," Asso. for Comp. Mach. J., Vol.6 (1959), pp.376-83.

29. Muller, Mervin E. "An Inverse of the Generation of Random Normal Deviates on Large-Scale Computers," Math. Tables Aids Comp. 12 (1958), pp.167-174.

30. Naylor, Balintfy, Burdick and Chu Computer Simulation Techniques, New York: John Wiley and Sons (1966).

31. Newmann, T.G. and Odell P.L. The Generation of Random Variates, New York: Hafner Pub. Co. (1971).

32. Norman, J.E. and Cannon L.E. "A Computer Program for the Generation of Random Variables from any Discrets Distribution," J. Stat. Comp. Simul. Vol.1 (1972), pp.331-48.

33. Parzen, E. Modern Probability Theory and Its Applications, New York: John Wiley and Sons (1960).

34. Sacks, A.J. A Program to Evaluate Uniform Random Number Generators, Masters Thesis, University of Houtson (1974).

35. Rothlf, F.J. and Sokal, R.R. Statistical Tables, W.H. Freeman and Co. (1969).

36. Phillips, Don T. and Beightler, Charles S. "Procedures for Generating Gamma Variates with Non-integer P rameters Sets," J. Stat. Comput. Simul., Vol.1 (1972), pp.197-208.

37. Smith, Marvin B. Probability Models for Petroleum Investment Decisions, Paper SPE 2587 of Society of Petroleum Engineers of AIME (1969).

38. Taussky, O. and Todd, J. "Generation of Pseudo Random Numbers""in Symposium on Monte Carlo Methods, Edited by H.A. Meyer, New York: John Wiley and Sons (1956).

39. Tocher, K.D. The Art of Simulation, Princeton: D. Van Hostrand Co. Inc. (1963).

40. Teichroew, Dan Distribution Sampling with high Speed Computers, Ph.D. Thesis, University of North Carolina (1953).

41. Von Neumann, John Various Techniques used in Connection with Random Digits, paper No.13 in "Monte Carlo Methods," NBS Applied Mathematics Series No.12, U.S. Govt. Printing Office (1951).

42. Votaw, D.F. Jr. and Rafferty, J.A. "High Speed Sampling" MTAC, Vol.5 (1951), pp.1-8.

43. Wagner, Harvey M. Principles of Operations Research with Applications to Managerial Decisions, New Jersey: Prentice-Hall Inc. (1969).

44. Johnk, M.D. "Erzeugung von Betaverteilten and Gamma Verteilten Zufallszahlen", Metrika, Vol.8, No.1, (1964) PP.5-15.