NPS-72Ro75041

NAVAL POSTGRADUATE SCHOOL
Monterey, California

GENERATING GAMMA AND CAUCHY RANDOM VARIABLES:
AN EXTENSION TO THE NAVAL POSTGRADUATE SCHOOL
RANDOM NUMBER PACKAGE

D. W. Robinson
and
P. A. W. Lewis

April 1975

Approved for public release; distribution unlimited

NAVAL POSTGRADUATE SCHOOL

Monterey, California

Rear Admiral Isham Linder — Superintendent

Jack R. Borsting — Provost

The work reported herein was supported in part by the National Science Foundation under grant AG 476 and in part by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

This report was prepared by:

GENERATING GAMMA AND CAUCHY RANDOM VARIABLES: AN EXTENSION TO THE NAVAL POSTGRADUATE SCHOOL RANDOM NUMBER PACKAGE

by

D. W. Robinson

and

P. A. W. Lewis *

* Work partially supported by the National Science Foundation under grant AG 476.

TABLE OF CONTENTS

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS-72Ro75041

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): NAVAL POSTGRADUATE SCHOOL
GENERATING GAMMA AND CAUCHY RANDOM VARIABLES: AN EXTENSION TO THE NAVAL POSTGRADUATE SCHOOL RANDOM NUMBER PACKAGE

5. TYPE OF REPORT & PERIOD COVERED: Technical Report

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s):
D. W. ROBINSON
P.A.W. LEWIS

8. CONTRACT OR GRANT NUMBER(s): NSF AG-476

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS:
Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE: April 1975

13. NUMBER OF PAGES: 58

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
Chief of Naval Research
Arlington, Virginia 22217

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report):
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Random number generator
Pseudo-random numbers
Gamma distribution
Cauchy distribution

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
Two very efficient algorithms for generating pseudorandom numbers from the gamma distribution have been developed by Ahrens and Dieter; in the present work these are combined with a third method to produce a combination generator capable of excellent performance for any order of gamma variate. The algorithms are briefly described and an IBM 360 Assembler implementation of them is described and tested. A second computer program for the generation of pseudorandom Cauchy deviates is presented; this program uses a new

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (continued)

algorithm which is also described. Both computer programs are intended to be used with the Naval Postgraduate School random number package LLRANDOM.

I. Introduction

The use of uniformly or non-uniformly distributed pseudorandom numbers in systems simulation, statistical sampling experiments and analytical Monte Carlo work is by now well established. Numerous algorithms exist for producing such numbers from various distributions; for summaries of common techniques, see Knuth [5], Gaver and Thompson [2] or Ahrens and Dieter [1].

The user of pseudorandom numbers is usually not concerned with the details of the algorithm employed but rather with the results; a good algorithm, then, is one which is fast, uses minimum computer memory and produces numbers with satisfactory statistical properties. The search for statistically competent algorithms for pseudorandom numbers has resulted in the specification of many so-called "exact" generators, that is those whose deviation from the true distribution concerned is the result of computer rounding errors rather than any defect in the method itself. Such methods for nonuniform random numbers are often based on the assumption that "good" uniform numbers are available from an independent generator.

Exact generators for nonuniform pseudorandom numbers are often quite complex and so assembly-level coding is often resorted to when implementing them in order to meet the computer time and memory constraints on a good algorithm. An example is the LLRANDOM package developed at the Naval Postgraduate School by G.P. Learmonth and P.A.W. Lewis and described in [7]; it produces pseudorandom numbers from uniform, normal and exponential distributions. This report describes an extension to the LLRANDOM package for Cauchy and gamma distributed numbers.

The Cauchy distribution has density function

$$(1) \qquad f(x) = \frac{1}{\pi} \frac{1}{1 + x^2}, \qquad -\infty < x < \infty ,$$

and distribution function

$$F(x) = \frac{1}{2} + \frac{1}{\pi} \tan^{-1} x.$$

While the shape of the Cauchy density resembles the normal density, the tails are much heavier; in fact, Cauchy random variables have no expectation and an infinite variance. The density has mode at zero and often in applications the variates are often shifted by a location parameter T or scaled by multiplying by a scale parameter S. Because of the heavy tails, Cauchy variates might find application as a "pathological" case in a systems simulation study as well as in statistical sampling experiments for robust estimation techniques. See Chapter 16 of Johnson and Kotz [4] for further details on the Cauchy distribution.

The gamma distribution with shape parameter A and scale parameter s has the density function

$$(2) \qquad f(x) = \frac{s^A x^{A-1} e^{-sx}}{\Gamma(A)} ,$$

where $\Gamma(A)$ is Euler's gamma function

$$(3) \qquad \Gamma(A) = \int_0^\infty x^{A-1} e^{-x} dx .$$

Note that $\Gamma(n) = (n-1)!$ when n is a non-negative integer. If the random variable X has density (2) then

$$E[X] = A / s ,$$

$$V[X] = A / s^2 .$$

When $A = 1$, X has the exponential distribution while X, suitably scaled, has an asymptotically normal distribution as $A \to \infty$.

We note that if X has a $\Gamma(A,1)$ distribution then X/s has a $\Gamma(A,s)$ distribution, so we may set $s = 1$ in (2) as far as the generating algorithm is concerned. The output from the generator may then be appropriately scaled.

Gamma random variables are used in a wide variety of applications: for analytical modeling, in reliability theory and for statistical testing (the chi-squared random variable with n degrees of freedom has the $\Gamma(\frac{n}{2}, \frac{1}{2})$ distribution). See [6] or Chapter 17 of [4] for more details.

II. Use of the Subroutines

This extension to LLRANDOM is composed of two independent IBM System/360 Assembler-coded subroutines: CAUCHY for Cauchy-distributed variates and GAMA for gamma variates. The name GAMA was chosen so as not to conflict with the IBM mathematical library subprogram GAMMA which computes the gamma function (3).

The basic conventions for using GAMA and CAUCHY are the same as in the LLRANDOM package: the invoking statements

```
        CALL CAUCHY ( IX, X, N )
    and CALL GAMA ( A, IX, X, N )
```

will result in a vector X(1), ... , X(N) of Cauchy or Γ (A,1.0) pseudorandom variates, respectively. The argument IX is, in both cases, an integer seed to be used in the multiplicative congruential uniform generator employed by LLRANDOM. IX should be initialized just once in the calling program to some positive integer value and should not be altered thereafter.

The subroutine GAMA requires a source for normal and exponential deviates; these are obtained directly from the LLRANDOM package and so the statement "CALL OVFLOW" must appear once in the calling program to initialize LLRANDOM. As mentioned previously, the output from GAMA must be scaled if the scale parameter is other than one; the following set of statements will thus be required to generate a vector of 100 chi-squared variates with seven degrees of freedom:

```
DIMENSION X(100)
CALL OVFLOW
IX = 13726
...
CALL GAMA ( 3.5, IX, X, 100 )
```

```
      DO 50 I = 1,100
      X(I) = 2.0 * X(I)
   50 CONTINUE
      ...
      END
```

Cauchy variates are also often modified by location and scale parameters; since no expectations exist, however, we cannot refer to these parameters in terms of mean or variance. Subroutine CAUCHY is completely independent of LLRANDOM or any other subroutines so that the "CALL OVFLOW" statement is not necessary in this case. To use CAUCHY to produce a single variate C with location parameter T and scale parameter S we may use the statements

```
      ...
      IX = 217663541
      ...
      CALL CAUCHY ( IX, C, 1 )
      C = S * C + T
      ...
      END
```

Just as in LLRANDOM, linkage overhead between the calling program and GAMA or CAUCHY will be minimized if a vector of several variates is obtained at the same time instead of just a single one. The gain in this case can be as much as 50 microseconds per variate in average generation time, an improvement of up to 50%. In GAMA, several constants must be calculated for each different value of the shape parameter A; these constants are saved between calls so that they need not be recomputed. It will thus be more efficient to get several gamma variates with the same shape parameter before changing the A value, especially when A > 3.0 when the setup computations are extensive (see lines

174-246 of the program listing).

Note that the techniques used in GAMA and CAUCHY make use of so-called rejection methods so that the number of uniform (or exponential or normal) deviates needed to generate a single output deviate is random. When normal or exponential deviates are required by GAMA from LLRANDOM a vector of 10 deviates is called for; since not all of these may be used at the time they are generated, the balance are saved for the next call to GAMA. Thus, reinitializing the seed IX to its original value will not in general result in an exact repetition of the generated gamma sequence since the first few deviates will use the old normal or exponential deviates from the previous sequence. To achieve an exact repetition, the generator must be forced to repeat the initialization computations for the desired A value; at this time any remaining variates from LLRANDOM are discarded. An example of this might be

```
      DIMENSION G(100)
      CALL OVFLOW
      IX = 12345
      ...
      CALL GAMA ( A, IX, G, 100 )
      ...
C     REINITIALIZE GAMMA SEQUENCE
      CALL GAMA ( 1.0, IX, G, 1 )
      IX = 12345
      ...
      CALL GAMA ( A, IX, G, 100 )
      ...
      END
```

CAUCHY requires 552 bytes and, as mentioned previously, is completely independent of any other subprograms. CAUCHY uses the LLRANDOM multiplicative congruential uniform generator but this is coded in line when needed so as to preserve CAUCHY's independence. The average generation time per variate for subroutine CAUCHY on a System/360 Model 67 under OS/MVT was 67.5 microseconds when variates were generated in vectors of 100. The generation of variates one at a time increased the average time to 119.3 microseconds per variate.

Subroutine GAMA itself uses only 1988 bytes of memory but since it calls on LLRANDOM the total core requirement is 9342 bytes:

GAMA	1988	bytes
LLRANDOM	6189	bytes
Required IBM Functions	1165	bytes
Total	9342	bytes

Timing the gamma generator on a System/360 Model 67 was carried out using the TIME macro; Table 1 summarizes the observed times as a function of the shape parameter, A. Note that since special methods are employed when A is 0.5, 1.0, 1.5, 2.0 or 3.0, the times in these cases are considerably shorter than times for nearby values of A.

Shape Parameter A	Algorithm	Vector of 100 Variates	Single Variate
0.1	GS	324.0	364.0
0.3	GS	367.0	402.5
0.5	GA	70.4	207.7
0.8	GS	439.8	551.2
0.9	GS	459.0	611.0
1.0	GA	68.7	158.9
1.2	GF	300.1	385.0
1.4	GF	306.1	441.0
1.5	GA	141.7	215.8
1.8	GF	343.6	390.8
2.0	GA	142.5	203.6
2.1	GF	396.1	450.8
2.5	GF	434.7	468.5
2.9	GF	444.5	496.6
3.0	GA	206.7	237.1
3.1	GO	341.5	435.8
3.5	GO	336.2	373.4
4.0	GO	332.4	420.7
5.0	GO	307.7	363.2
8.0	GO	293.1	371.3
10.0	GO	289.4	312.5
20.0	GO	238.2	321.6
50.0	GO	197.7	284.2
100.0	GO	178.4	220.0
1000.0	GO	166.7	177.0
10000.0	GO	136.4	169.8
100000.0	GO	152.5	235.8

Table 1. Average generation times (microseconds) for gamma variates using subroutine GAMA.

III. Description of the Algorithms

This section describes the actual algorithms used in CAUCHY and GAMA. An understanding of the algorithms is not necessary for use of the package but they are set forth here both in the interest of completeness and in an effort to document the programs more fully. A single algorithm suffices for the Cauchy generator while GAMA uses one of four algorithms, depending on the value of A.

In the descriptions which follow, the letters U, N and E (with or without affixes) represent uniform, standard normal and unit exponential pseudorandom deviates, respectively. The phrase "Generate U" implies that U is the next sequential uniform variate in the linear congruential sequence; these variates are generated as needed by using the same multiplicative congruential scheme as used in LLRANDOM. The phrases "Generate N" or "Generate E" imply that normal or exponential variates are to be obtained by linking directly to LLRANDOM.

A. Cauchy Generator

The Cauchy generator is a combination decomposition-rejection method (see Knuth [5]). The Cauchy density is decomposed, as in Figure 1, into three subdensities: a uniform density between 0 and 1 (f_1), a wedge-shaped density (f_2) and a long tailed density (f_3).

The uniform density f_1 is sampled with probability $1/\pi$; in this case a uniform(0,1) variate is returned. The density f_2 is dealt with by using Marsaglia's almost-linear density algorithm, just as in Knuth's Algorithm L [5]. The density f_2 is sampled with probability $1/2 - 1/\pi$. The tail density f_3 is sampled by a rejection method with probability 1/2. The majorizing density for f_3 is $g(x) = 1 / x^2$, which is the density of the reciprocal of a uniform (0,1) variate.

Algorithm C below uses the fact that in the prime modulus congruential random number generator used in LLRANDOM the low order bits are uniformly distributed so that b_1 and b_2 select the proper sub-distribution in Step 1. This will not in general be the case for other congruential pseudo-random number generators.

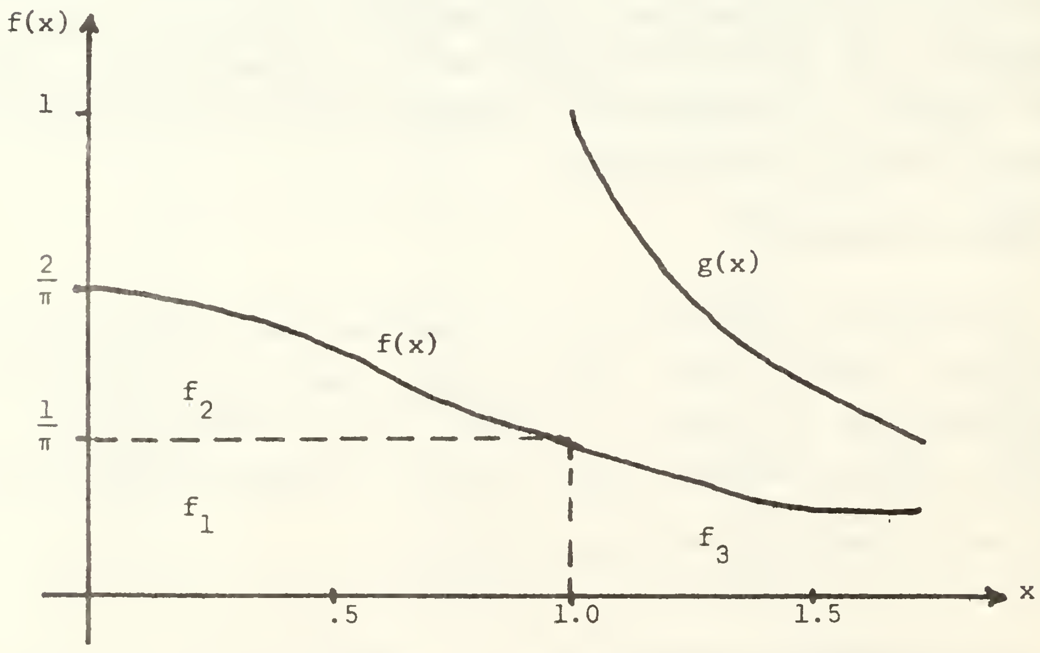

Figure 1. Decomposition of the Cauchy Density Function.

Algorithm C. Cauchy variates.

1. (Select subdensity) Generate U, setting aside the two low order bits b_1 and b_2. If $b_1 = 1$, go to Step 6.
2. (Sample box) If $U \le 0.6366197724 = 2/\pi$, generate a new variate U^*, set $x = U^*$ and go to Step 8.
3. (Sample wedge) Generate new variates U_1 and U_2. If $U_1 > U_2$, exchange U_1 and U_2. Set $x = U_1$.
4. (Easy rejection) If $U_2 \le 0.8284271247 = 2\sqrt{2} - 2$, go to Step 8.
5. (Hard rejection) If $U_2 - U_1 \le \frac{1}{1 + x^2}(2\sqrt{2} - 2)$, go to Step 8, otherwise go back to Step 3.
6. (Sample tail) Set $x = 1 / U$.
7. (Tail rejection) Generate a new variate U^*. If $U^* \le \frac{x^2}{1 + x^2}$ go to Step 8, otherwise generate a new U and go back to Step 6.
8. (Random sign) If $b_2 = 1$ set $x = - x$. Deliver x as the generated deviate.

It should be noted that there are several other methods for generating Cauchy variates: the ratio of independent standard normal deviates has the Cauchy distribution, as does the quantity

$$X = \tan\left[\pi\left(U - \frac{1}{2}\right)\right],$$

where U is uniform(0,1). These methods are both substantially slower than algorithm C, but another new method has an average time comparable to Algorithm C and is much easier to program. This second method requires an average of 2.55 uniform random variates per Cauchy variate (as compared with 2.47 for algorithm C) and it needs about 69 microseconds per variate on the System/360 Model 67. It is possible, however, that Algorithm CR will be better than algorithm C in some other implementation.

The method is essentially the technique devised by von Neumann to generate a random variate sin U, where U is uniform between 0 and 2π. Such variates are used in the polar method for generating normal random variables [8]. It does not seem to have been recognized that the method also generates tan U, which is the required Cauchy variate.

__Algorithm__ __CR__. Cauchy variates, ratio method.

1. (Get uniforms) Generate U_1 and U_2. Set $Y_1 = 2U_1 - 1$ and $Y_2 = 2U_2 - 1$.
2. (Rejection test) If $Y_1^2 + Y_2^2 > 1$ go back to Step 1.
3. (Take ratio) Deliver $x = Y_1 / Y_2$.

B. __Gamma Generator GS: A ≤ 1.0__

This method is due to Ahrens and is set forth in [1]. It is applicable only to values of A less than one and is markedly superior in execution time to the method of Johnk [3], which is the usual technique for generating variates of this type.

The method is a rejection method employing two different tests, one of which is chosen at random for any given variate: the power transform of a uniform(0,1) variate, $U^{1/A}$, is tested in the region $0 < x < 1$, while a suitable exponential, E, is tested when $x > 1$. The advantage of this method lies in the limited use of the library subprograms for the exponential and logarithm; average times range from 300 to 400 microseconds as compared with 600 to 800 for Johnk's method. Further discussion and proofs may be found in [1].

<u>Algorithm GS</u>. Gamma variates, $A < 1.0$.

1. (Select rejection test) Generate U and generate E and set $P = \frac{e+A}{e} U$. (Note that "e" is the base of the natural logarithms.) If $P \leq 1$ go to Step 2, otherwise go to Step 3.
2. (Small x test) Set $x = P^{1/A}$. If $x \leq E$, deliver x, otherwise go back to Step 1.
3. (Large x test) Set $x = -\ln\left[\frac{1}{A}\left\{\frac{e+A}{e} - P\right\}\right]$. If $(1 - A) \ln x \leq E$, deliver x, otherwise go back to Step 1.

C. <u>Gamma Generator GF: $1.0 \leq A \leq 3.0$</u>

A thus-far unpublished method devised by Professor G.S. Fishman of North Carolina University was communicated to the authors in private correspondence. It is valid for any $A > 1.0$ but its efficiency in terms of average time goes down as $\sqrt{A}$ so it is applied in GAMA only in the range where it is superior to the Dieter-Ahrens method GO described below.

The method is a rejection method based on the following theorem.

Theorem Let U be a uniform (0,1) random variable and let E be an exponential random variable with mean A. Let

$$g(x) = \left[\frac{x}{A}\right]^{A-1} e^{-x(1-1/A) - (A-1)} .$$

If $g(E) \geq U$, then E has conditionally the gamma distribution with shape parameter A, i.e.

$$f_E(\, x \mid U \leq g(E)\,) = \frac{x^{A-1} e^{-x}}{\Gamma(A)} .$$

Proof:

Unconditionally, E has density $h(x) = \frac{1}{A} e^{-x/A}$.

Therefore,

$$(4) \qquad f_E(x \mid U \leq g(E)\,) = \frac{h(x)\, \Pr\{U \leq g(E) \mid E=x\}}{\Pr\{U \leq g(E)\}} .$$

Now since U is uniformly distributed,

$$\Pr\{U \leq g(E) \mid E=x\} = g(x)$$

as long as $0 < g(x) < 1$; that this is true for every $x > 0$ may be readily verified by elementary calculus. Therefore,

$$\begin{aligned} (5) \qquad \Pr\{U \leq g(E)\} &= E[\; \Pr\{U \leq g(E) \mid E\}\;] \\ &= \int_0^\infty g(x)\; h(x)\; dx \\ &= \Gamma(A)\; e^{A-1} A^{-A} \\ &= C(A) \end{aligned}$$

Thus, in view of (4),

$$f_E(x \mid U \le g(E)) = \frac{h(x)\, g(x)}{C(A)}$$

$$= \frac{x^{A-1} e^{-x}}{\Gamma(A)}$$

The efficiency of the generator is governed by the probability that a given variate will pass the rejection test, $U \le g(E)$; from (5) it will be seen that this probability is just $C(A)$. When A is large we have from Stirling's approximation that $C(A) \doteq \sqrt{\frac{2\pi}{A e^2}}$, so that the method becomes more inefficient with increasing A, as noted above.

A slight modification to the method suggested by the theorem improves the efficiency slightly and we obtain

Algorithm GF. Gamma variates, $1.0 < A < 3.0$.

1. (Generate exponentials) Generate two independent exponential variates, E_1 and E_2.
2. (Rejection test) If $E_2 < (A-1)(E_1 - \ln E_1 - 1)$ then go back to Step 1.
3. (Acceptance) Deliver $x = A\, E_1$.

D. Gamma Generator GO: $A \ge 3.0$

This method was originally developed by Dieter and Ahrens and is fully described in [1] together with several other gamma generation techniques. Algorithm GO does not suffer the usual drawback of growing less efficient in generation time with increasing A; in fact, the method is more efficient for larger A values.

The basic idea here is to take advantage of the asymptotic normality of the gamma distribution by doing most of the sampling from a normal distribution; the right hand tail is sampled, when necessary, using a rejection method with the exponential distribution. The method can be applied to values of A greater than 2.533, but it is not as efficient as Fishman's technique for A < 3.0.

As mentioned previously, this algorithm requires the computation of several constants which depend only on A and which may be saved between calls; these calculations are described in step 0 of the specification below. Further discussion, illustrations and proofs are given in [1]; the version of GO here differs in a few minor details from the original Dieter and Ahrens technique.

Algorithm GO. Gamma variates, a > 3.0.

0. (Calculate constants) Compute:

$$m = A - 1;$$

$$s^2 = \sqrt{\frac{8A}{3}} + A; \qquad s = \sqrt{s^2};$$

$$d = \sqrt{6s^2}; \qquad b = d + m;$$

$$w = s^2 / m - 1; \qquad v = 2s^2 / (m\sqrt{A});$$

$$c = b + \ln \frac{s\_d}{b} - 2m - 3.7203285.$$

1. (Select normal/exponential) Generate U. If U ≤ 0.0095722652 go to Step 7.
2. (Normal sampling) Generate N and set x = sN + m.
3. (Check trial value) If x < 0 or x > b go back to Step 2,

otherwise generate a new variate U and set $S = N^2 / 2$. If $N > 0$ go to Step 5.

4. (Left-hand rejection) If $U < 1 + S(vN - w)$ go to Step 9, otherwise go to Step 6.
5. (Right-hand rejection) If $U < 1 - wS$ go to Step 9.
6. (Final normal rejection) If $\ln U < m \ln \frac{x}{m} + m - x + S$ go to Step 9; otherwise go back to step 1.
7. (Exponential) Generate E_1 and E_2 and set $x = b(1+E_1/d)$.
8. (Exponential rejection) If $m\left(\frac{x}{b} - \ln \frac{x}{m}\right) + c > E_2$ go back to Step 1.
9. (End) Deliver x as the gamma variate.

E. Ad Hoc Gamma Generators

This set of algorithms is based on the well-known fact that the sum of independent gamma variates with shape parameters A_1 and A_2 and equal scale parameters has the gamma distribution with shape parameter $A_1 + A_2$ and scale parameter equal to that of the summands. We may thus generate a gamma variate with integer shape parameter K by taking the sum of K independent exponentials. This will be more efficient than the previously discussed methods (Algorithms GF and GO) for moderate values of K; for the System/360 we take $K \leq 3$ to apply this ad hoc technique.

An obvious extension to this method is to allow for half-integral values of A by making use of the fact that the square of a standard normal random variable has the chi-squared distribution with one degree of freedom, i.e. $N^2/2$ has the gamma distribution with unit scale parameter and $A = 0.5$. We use this extension for $A = 0.5$ or 1.5.

The resulting algorithm is then

<u>Algorithm GA</u>. Gamma variates, integral or half-integral shape parameter A.

1. (Find K) Set K = [A], where [A] denotes the integral part of A. Set X = 0. If A - K = 0.5 set L = 1; if A - K = 0.0 set L = 0; otherwise Stop. (If the algorithm stops, an incorrect A value has been used.)
2. (Generate exponentials) If K = 0 go to Step 3, otherwise generate K exponentials $E_1, \ldots, E_K$ and set $X = E_1 + \ldots + E_K$.
3. (Generate normal) If L = 0 go to Step 4 otherwise generate N and set $X = X + N^2/2$.
4. (Deliver X) X is the desired variate.

IV. Summary and Comments

This work provides a convenient and useful extension to the LLRANDOM package, especially for users interested in statistical and reliability theory applications of digital simulation. The combination of the most efficient known gamma generation techniques with the new Cauchy method gives exceptionally good time characteristics at some cost in computer memory utilization.

The work may be extended at once to the generation of several other types of random variables. For example, the beta distribution with parameters A and B may be sampled by taking gamma variates X_1 and X_2 with respective shape parameters A and B and delivering

$$Z = X_1 / (X_1 + X_2)$$

as a beta variate. In this case considerable overhead in GAMA can result from shifting the shape parameter back and forth between A and B; for this reason obtaining vectors of gamma variates X_1 and X_2 is recommended, as in the following example:

```
        DIMENSION X1(50), X2(50), Z(50)
        ...
        CALL GAMA ( A, IX, X1, 50 )
        CALL GAMA ( B, IX, X2, 50 )
        DO 405 I = 1,50
        Z(I) = X1(I) / ( X1(I) + X2(I) )
    405 CONTINUE
        ...
        END
```

The t-Distribution may be sampled as the ratio of a standard normal and an independent chi-squared random variate, while the F-Distribution may be obtained by taking the ratio of two independent chi-squared variates divided by their respective degrees of freedom. (See pages 4 and 5 for an example of the generation of chi-squared variates.)

References

[1] Ahrens, J.H., and Dieter, U., Pseudo-Random Numbers, preprint edition of a forthcoming book.

[2] Gaver, D.P., and Thompson, G.L., Programming and Probability Models in Operations Research, Brooks/Cole, Monterey, California, 1973.

[3] Johnk, M.D., "Erzeugung von Betaverteilten und Gammaverteilten Zufallszahlen", Metrika, v. 8, p. 5-15, 1964.

[4] Johnson, N.L., and Kotz, S., Distributions in Statistics, Volume I: Continuous Univariate Distributions 1, Wiley, 1970.

[5] Knuth, D.E., The Art of Computer Programming, Volume II: Seminumerical Algorithms, Addison-Wesley, 1972.

[6] Lanchester, H.O., The X^2 Distribution, Wiley, 1969.

[7] Learmonth, G.P., and Lewis, P.A.W., Naval Postgraduate School Random Number Generator Package LLRANDOM, Naval Postgraduate School Technical Report NPS-55Lw-73061A, June 1973.

[8] Von Neumann, J., "Various Techniques Used in Connection with Random Digits," Monte Carlo Methods, National Bureau of Standards Applied Math Series 12, p. 36-38, 1951.

```
**** CAUCHY DEVIATE GENERATOR ****

*
*       PURPOSE:
*
*           GENERATION OF RANDOM VARIATES WITH THE CAUCHY DISTRIBUTION
*
*       USAGE:
*
*           CALL CAUCHY (IX, C, N)
*
*       PARAMETERS:
*
*       IX     SEED FOR RANDOM NUMBER GENERATOR (INTEGER*4). SHOULD BE
*              INITIALIZED TO ANY POSITIVE VALUE IN THE CALLING PROGRAM
*              AND NOT ALTERED THEREAFTER.
*
*       C      ARRAY TO HOLD THE GENERATED VARIATES (REAL*4). MUST BE
*              DIMENSIONED AT LEAST N.
*
*       N      NUMBER OF CAUCHY DEVIATES TO GENERATE (INTEGER*4).
*
*       METHOD:
*
*           A COMBINED DECOMPOSITION/REJECTION METHOD IS USED. ALL
*       SUBDISTRIBUTIONS CAN BE SAMPLED USING UNIFORM DEVIATES ONLY.
*
*       SUBROUTINES REQUIRED:
*
*           NONE
*
*
*       PROGRAMMER:     D.W. ROBINSON
*
*       DATE:           9 MAY 1974
*
```

**** CAUCHY DEVIATE GENERATOR ****

```
*
*         REGISTER ALLOCATION
*
*         R0     SAVE +/- BIT
*         R1     WORK REGISTER
*
*         R2     CONSTANT 4
*         R3     NUMBER OF DEVIATES (BYTES)
*         R4     BASE ADDRESS OF C ARRAY
*         R5     INDEX OF CURRENT RANDOM NUMBER IN C
*
*         R6,R7   SEED FOR GENERATOR
*         R8     UNIFORM MULTIPLIER = 16807
*         R9     EXPONENT CONSTANT = 40000001
*         R10    NORMALIZATICN COMPARAND = 40100000
*
*         R11    CONSTANT 1 (MASK)
*         R12    ADDRESS OF END OF MAIN LOOP
*
*         R13    ADDRESS OF IX IN CALLING PROGRAM
*
*         R14    RETURN ADDRESS
*         R15    BASE REGISTER
*
*
**********************************************************************
*
*         UNIFORM RANDOM NUMBER GENERATION MACRO
*
*         WITH THE CURRENT UNIFORM INTEGER IN R7 AND THE MULTIPLIER
*         IN R8, FINDS THE NEXT UNIFORM INTEGER AND PUTS IT INTO R7.
*
**********************************************************************
*
          MACRO
&A        RAND
&A        MR     R6,R8               GET NEXT UNIFORM
          SLDA   R6,1                R6 = REMAINDER; R7 = QUOTIENT
          SRL    R7,1                ADD QUOTIENT TO REMAINDER, THUS
          AR     R6,R7                 SIMULATING DIVISION BY 2 ** 31 - 1
          BNO    *+10                GO ON IF NO OVERFLOW
          A      R6,=F'2147483645'   FIXUP OVERFLOW.  ADD 2 ** 31 - 3
          AR     R6,R2               ADD FOUR MORE
          LR     R7,R6               PUT X(N) INTO R7
          MEND
```

```
**** CAUCHY DEVIATE GENERATOR ****

CAUCHY   CSECT                                                       CAU00850
         USING CAUCHY,R15    DEFINE BASE REGISTER                    CAU00860
         B     12(,R15)     BRANCH AROUND ID                         CAU00870
         DC    AL1(6)                                                CAU00880
         DC    CL6'CAUCHY'   MODULE NAME                             CAU00890
         STM   R14,R12,12(R13)  SAVE CALLING PROGRAM REGS            CAU00900
         ST    R13,SVAREA+4  CALLING SAVE ADDRESS IN OWN AREA        CAU00910
         LR    R2,R13        COPY CALLING SAVE ADDRESS TO R2         CAU00920
         LA    R13,SVAREA    OWN SAVE AREA IN R13                    CAU00930
         ST    R13,8(,R2)    FORWARD LINK                            CAU00940
*                                                                    CAU00950
*                                                                    CAU00960
         LM    R3,R5,0(R1)   GET PARAMETER ADDRESSES                 CAU00970
         LR    R13,R3        SAVE SEED ADDRESS                       CAU00980
         L     R7,0(,R3)     GET SEED VALUE                          CAU00990
         L     R3,0(,R5)     LOAD NUMBER OF DEVIATES TO GENERATE     CAU01000
         SLA   R3,2          CONVERT N TO BYTES                      CAU01010
         LA    R2,4          CONSTANT 4 FOR MAIN LOOP                CAU01020
         SR    R4,R2         BACK UP 4 IN CALLER'S ARRAY             CAU01030
         LR    R5,R2         INITIAL ARRAY INDEX                     CAU01040
         LM    R8,R12,LOOPCON  LOAD MAIN LOOP CONSTANTS              CAU01050
         CNOP  0,8           ALIGN BXLE LOOP FOR SPEED               CAU01060
*                                                                    CAU01070
*                                                                    CAU01080
MAINLOOP RAND  ,             GET FIRST UNIFORM                       CAU01090
*                            SAVE TWO BITS OF X(N)                   CAU01100
         LR    R0,R6         LAST BIT OF X(N) IN R0                  CAU01110
         LR    R1,R6         NEXT TO LAST BIT IN R1                  CAU01120
         SRL   R1,1                                                  CAU01130
         NR    R1,R11        TEST BIT IN R1; IF 0, SAMPLE FROM TAIL  CAU01140
         BZ    TAIL                                                  CAU01150
*                                                                    CAU01160
         C     R6,=F'1367130551'  SELECT RECTANGLE/WEDGE SAMPLING    CAU01170
         BH    WEDGE                                                 CAU01180
*                                                                    CAU01190
RECT     RAND  ,             GET NEXT UNIFORM                        CAU01200
SAMPL    SRL   R6,7          MAKE ROOM FOR EXPONENT                  CAU01210
         OR    R6,R9         "OR" ON THE EXPONENT                    CAU01220
         ST    R6,UNIF       STORE THE UNIFORM                       CAU01230
         LE    FR0,UNIF                                              CAU01240
         CR    R6,R10        TEST FOR NORMALIZATION                  CAU01250
         BCR   11,R12        QUIT IF NOT NEEDED                      CAU01260
         AE    FR0,=E'0.0'   NORMALIZE THE UNIFORM                   CAU01270
         BR    R12           GO TO END OF LOOP                       CAU01280
*                                                                    CAU01290
```

**** CAUCHY DEVIATE GENERATOR ****

```
WEDGE    RAND                                                                  CAU01300
         LR     R1,R6            SAVE FIRST UNIFORM                            CAU01310
         RAND                                                                  CAU01320
         CR     R6,R1            GET UNIFORM IN R6 < UNIFORM IN R1             CAU01330
         BNH    *+8                                                            CAU01340
         LR     R6,R1            EXCHANGE REGISTERS                            CAU01350
         LR     R1,R7                                                          CAU01360
         C      R1,=F'1779033703'  EASY REJECTION TEST                         CAU01370
         BL     SAMPL            ACCEPT WEDGE SAMPLE                           CAU01380
         SRL    R6,7             CONVERT MINIMUM UNIFORM TO REAL               CAU01390
         OR     R6,R9            "OR" ON THE EXPONENT                          CAU01400
         ST     R6,UNIF                                                        CAU01410
         SRL    R1,7             CONVERT MAXIMUM UNIFORM TO REAL               CAU01420
         OR     R1,R9            "OR" ON THE EXPONENT                          CAU01430
         ST     R1,U2                                                          CAU01440
         LE     FR0,UNIF         LOAD TRIAL VARIATE                            CAU01450
         CR     R6,R10           TEST FOR NORMALIZATION                        CAU01460
         BC     11,*+8                                                         CAU01470
         AE     FR0,=E'0.0'      NORMALIZE X                                   CAU01480
         LE     FR2,U2           GET FIRST COMPARAND FOR REJECTION TEST        CAU01490
         SER    FR2,FR0             U2 - X                                     CAU01500
         LER    FR4,FR0          FIND X ** 2                                   CAU01510
         MER    FR4,FR0                                                        CAU01520
         LCER   FR6,FR4          - X ** 2 IN FR6                               CAU01530
         AE     FR6,=E'1.0'      1 - X ** 2                                    CAU01540
         AE     FR4,=E'1.0'      1 + X ** 2                                    CAU01550
         DER    FR6,FR4          FIND QUOTIENT                                 CAU01560
         ME     FR6,=E'.82842712' CONSTANT IS 2 / (1 + SQRT(2) )               CAU01570
         CER    FR2,FR6          HARD REJECTION TEST                           CAU01580
         BCR    13,R12                                                         CAU01590
         B      WEDGE            GO BACK IF TEST FAILED                        CAU01600
*                                                                              CAU01610
```

**** CAUCHY DEVIATE GENERATOR ****

```
*
TAIL     SRL   R6,7             MAKE ROOM FOR EXPONENT
         OR    R6,R9            "OR" ON THE EXPONENT
         ST    R6,UNIF          STORE THE UNIFORM
         LE    FR0,=E'1.0'      GET 1 / UNIFORM
         DE    FR0,UNIF
         RAND  ,                GET ANOTHER UNIFORM FOR REJECTION TEST
         SRL   R6,7             MAKE ROOM FOR EXPONENT
         OR    R6,R9            "OR" ON THE EXPONENT
         ST    R6,UNIF
*
         LER   FR2,FR0          FIND X ** 2
         MER   FR2,FR0
         LER   FR4,FR2          GET 1 + X ** 2
         AE    FR4,=E'1.0'
         ME    FR4,UNIF         FIND COMPARAND FOR REJECTION TEST
         CER   FR4,FR2          REJECTION TEST
         BCR   13,R12
         RAND  ,                ANOTHER UNIFORM FOR NEXT PASS
         B     TAIL             GO BACK
*
*
ENDLOOP  NR    R0,R11           TEST SAVED BIT
         BZ    *+6              IF BIT = 0, QUIT
         LCER  FR0,FR0          IF BIT = 1, X = -X
         STE   FR0,0(R4,R5)     STORE VARIATE IN CALLER'S ARRAY
         BXLE  R5,R2,MAINLOOP    BRANCH BACK FOR ANCTHER VARIATE
*
*
         ST    R7,0(,R13)       SEND LAST SEED BACK TO CALLING PROGRAM
         L     R13,SVAREA+4     GET CALLING SAVE AREA ADDRESS
         LM    R14,R12,12(R13)   RESTORE CALLING PROG REGS
         BR    R14              RETURN
```

**** CAUCHY DEVIATE GENERATOR ****

```
*                                                                       CAU01960
*        DATA AREA                                                      CAU01970
*                                                                       CAU01980
SVAREA   DS    18F               SAVE AREA                              CAU01990
*                                                                       CAU02000
UNIF     DS    F                 TEMP STORAGE FOR UNIFORM               CAU02010
U2       DS    F                    RANDOM VARIATES                     CAU02020
*                                                                       CAU02030
LOOPCON  DC    F'16807'          MULTIPLIER FOR GENERATOR     => R8     CAU02040
         DC    X'40000001'       EXPONENT CONSTANT            => R9     CAU02050
         DC    X'40100000'       NORMALIZATION TEST CONSTANT  => R10    CAU02060
         DC    F'1'              MASK CONSTANT                => R11    CAU02070
         DC    AL4(ENDLOOP)      END OF LOOP ADDRESS          => R12    CAU02080
*                                                                       CAU02090
         LTORG                                                          CAU02100
*                                                                       CAU02110
*        REGISTER EQUATES                                               CAU02120
*                                                                       CAU02130
R0       EQU   0                                                        CAU02140
R1       EQU   1                                                        CAU02150
R2       EQU   2                                                        CAU02160
R3       EQU   3                                                        CAU02170
R4       EQU   4                                                        CAU02180
R5       EQU   5                                                        CAU02190
R6       EQU   6                                                        CAU02200
R7       EQU   7                                                        CAU02210
R8       EQU   8                                                        CAU02220
R9       EQU   9                                                        CAU02230
R10      EQU   10                                                       CAU02240
R11      EQU   11                                                       CAU02250
R12      EQU   12                                                       CAU02260
R13      EQU   13                                                       CAU02270
R14      EQU   14                                                       CAU02280
R15      EQU   15                                                       CAU02290
*                                                                       CAU02300
FR0      EQU   0                                                        CAU02310
FR2      EQU   2                                                        CAU02320
FR4      EQU   4                                                        CAU02330
FR6      EQU   6                                                        CAU02340
         END                                                            CAU02350
```

```
**** GAMMA DEVIATE GENERATOR ****

*
*       PURPOSE:
*
*           GENERATION OF PSEUDO-RANDOM GAMMA DEVIATES WITH
*       NON-INTEGRAL SHAPE PARAMETER A > 0 AND SCALE PARAMETER 1.
*
*       USAGE:
*
*           CALL GAMA (A, IX, G, N)
*
*       PARAMETERS:
*
*       A       GAMMA SHAPE PARAMETER (REAL*4). MUST BE > 0.
*
*       IX      SEED FOR GENERATOR (INTEGER*4). SHOULD BE INITIALIZED
*               IN THE CALLING PROGRAM TO ANY POSITIVE VALUE AND
*               NOT ALTERED THEREAFTER.
*
*       G       ARRAY TO HOLD THE GENERATED DEVIATES (REAL*4). SHOULD
*               BE DIMENSIONED AT LEAST N.
*
*       N       NUMBER OF GAMMA DEVIATES TO BE DELIVERED (INTEGER*4).
*
*
*       METHOD:
*
*           THREE DIFFERENT BASIC METHODS ARE USED, DEPENDING ON
*       THE VALUE OF A:
*
*           0 < A < 1   AHRENS SMALL PARAMETER METHOD (ALGORITHM "GS").
*
*           1 < A < 3   FISHMAN'S REJECTION METHOD (ALGORITHM "GF").
*
*           3 < A       DIETER-AHRENS NORMAL-EXPONENTIAL METHOD
*                       (ALGORITHM "GO").
*
*       WHEN A IS EXACTLY 0.5, 1.0, 1.5, 2.0 OR 3.0 AN AD HOC
*       METHOD BASED ON TAKING THE SUM OF INDEPENDENT EXPONENTIALS
*       IS USED.
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*        SUBROUTINES REQUIRED:
*
*            THE LEWIS AND LEARMONTH RANDOM NUMBER GENERATOR PACKAGE
*        LLRANDOM IS NEEDED.  THE FORTRAN BUILT-IN FUNCTIONS ALOG,
*        EXP AND SQRT ARE ALSO USED.
*
*        NOTES:
*
*        1.  IF A < 0.1, AN UNDERFLOW CONDITION IS LIKELY TO ARISE
*        BECAUSE THE GENERATED DEVIATES WILL BE TOO SMALL.  THE
*        FORTRAN STANDARD FIXUP IN THIS CASE IS TO SET THE GENERATED
*        DEVIATE TO ZERO; THIS MAY CAUSE PROBLEMS IF FURTHER DATA
*        TRANSFORMATIONS (E.G., LOGARITHMS) ARE PLANNED.
*
*        2.  THIS SUBROUTINE IS, IN GENERAL, MORE EFFICIENT IF A LARGE
*        NUMBER OF GAMMA DEVIATES IS GENERATED.
*
*        3.  BECAUSE SOME VECTORS OF NORMAL OR EXPONENTIAL DEVIATES
*        WILL BE SAVED BETWEEN CALLS BY METHODS GO, GS, OR GF, IT MAY
*        NOT BE POSSIBLE TO PRODUCE TWO COMPLETELY DIFFERENT SEQUENCES
*        OF DEVIATES WITH DIFFERENT SEEDS.
*
*
*        PROGRAMMER:  D.W. ROBINSON
*
*        DATE:        27 JANUARY 1975
*
*        VERSION:       1    ADDED 0.5, 1.5, 2.0 AND 3.0 METHODS
*
```

```
**** GAMMA DEVIATE GENERATOR ****

*                                                                      GMA 0720
*       REGISTER ALLOCATION                                            GMA 0730
*                                                                      GMA 0740
*       R0     LINKAGE                                                 GMA 0750
*       R1     LINKAGE                                                 GMA 0760
*                                                                      GMA 0770
*       R2     CONSTANT 4                      -|                      GMA 0780
*       R3     NO DEVIATES WANTED (BYTES)       |  MAIN                GMA 0790
*       R4     CALLER'S ARRAY ADDRESS           |   LOOP               GMA 0800
*       R5     ARRAY INDEX                     _|                      GMA 0810
*                                                                      GMA 0820
*       R6      (MULTIPLICATION)               -|  UNIFORM             GMA 0830
*       R7     IX (SEED)                        |  GENERATOR           GMA 0840
*       R8     MULTIPLIER = 16807               |  (GS, GO ONLY)       GMA 0850
*       R9     EXPONENT CONSTANT               _|                      GMA 0860
*                                                                      GMA 0870
*       R8     V(EXP) OR V(EXPON)              -|  (GF, GS             GMA 0880
*       R9     V(ALOG)                         _|   ONLY)              GMA 0890
*                                                                      GMA 0900
*       R10    CONSTANT 4                      -|  NORMAL/             GMA 0910
*       R11    ARRAY SIZE                       |   EXPONENTIAL        GMA 0920
*       R12    ARRAY INDEX                     _|  LOOP (GS,GO,GF)     GMA 0930
*                                                                      GMA 0940
*       R13    END OF BXLE LOOP (GO ONLY)                              GMA 0950
*                                                                      GMA 0960
*       R14    LINKAGE                                                 GMA 0970
*                                                                      GMA 0980
*       R15    BASE REGISTER                                           GMA 0990
*                                                                      GMA 1000
*                                                                      GMA 1010
*       FR2    HOLDS GENERATED DEVIATE                                 GMA 1020
*                                                                      GMA 1030
```

**** GAMMA DEVIATE GENERATOR ****

```
*                                                            GMA 1040
*         REGISTER EQUATES:                                  GMA 1050
R0        EQU   0                                            GMA 1060
R1        EQU   1                                            GMA 1070
R2        EQU   2                                            GMA 1080
R3        EQU   3                                            GMA 1090
R4        EQU   4                                            GMA 1100
R5        EQU   5                                            GMA 1110
R6        EQU   6                                            GMA 1120
R7        EQU   7                                            GMA 1130
R8        EQU   8                                            GMA 1140
R9        EQU   9                                            GMA 1150
R10       EQU   10                                           GMA 1160
R11       EQU   11                                           GMA 1170
R12       EQU   12                                           GMA 1180
R13       EQU   13                                           GMA 1190
R14       EQU   14                                           GMA 1200
R15       EQU   15                                           GMA 1210
*                                                            GMA 1220
FR0       EQU   0                                            GMA 1230
FR2       EQU   2                                            GMA 1240
FR4       EQU   4                                            GMA 1250
FR6       EQU   6                                            GMA 1260
```

```
**** GAMMA DEVIATE GENERATOR ****

*
*        LINKAGE / INITIALIZATION SECTION
*
GAMA     CSECT
         USING GAMA,R15          DEFINE BASE REGISTER
         B     10(,R15)          BRANCH AROUND ID
         DC    AL1(4)
         DC    CL4'GAMA'         MODULE IDENTIFIER
         STM   R14,R12,12(R13)     SAVE CALLING REGS
         ST    R13,SVAREA+4      CALLING SAVE ADDRESS IN OWN AREA
         LR    R2,R13            COPY CALLING AREA ADDRESS TO R2
         LA    R13,SVAREA        OWN SAVE AREA IN R13
         ST    R13,8(,R2)        FORWARD LINK
*
*
         LM    R2,R5,0(R1)       GET PARAMETER ADDRESSES
         LE    FR0,0(,R2)        GET SHAPE PARAMETER
         CE    FR0,AP            TEST FOR NEW "A" VALUE
         BNE   SETUP             IF SO, DO PRELIMINARY CALCULATIONS
GWAN     LA    R2,4              CONSTANT 4 FOR MAIN LOOP
         L     R7,0(,R3)         PUT SEED INTO R7
         L     R3,0(,R5)         GET NUMBER OF DEVIATES, N
         SLA   R3,2              CONVERT TO BYTES
         SR    R4,R2             BACKUP ONE IN CALLER'S ARRAY
         LR    R5,R2             INITIAL MAIN LOOP INDEX
         L     R6,METHOD         JUMP TO PROPER METHOD
         BR    R6
```

```
**** GAMMA DEVIATE GENERATOR ****

*                                                                       GMA 1560
*        SETUP AND CONSTANT CALCULATION                                 GMA 1570
*                                                                       GMA 1580
SETUP    LTER  FR0,FR0          TEST FOR VALID A                        GMA 1590
         BNP   THRU                                                     GMA 1600
         STE   FR0,AP           SAVE NEW SHAPE PARAMETER                GMA 1610
         CE    FR0,=E'0.5'      FIND PROPER SCALE INTERVAL              GMA 1620
         BE    S1               AD HOC METHOD FOR A = 0.5               GMA 1630
         CE    FR0,=E'1.0'                                              GMA 1640
         BL    SGS              METHOD "GS" FOR A < 1.                  GMA 1650
         BE    SEXPN            USE "EXPON" GENERATOR FOR A = 1.        GMA 1660
         CE    FR0,=E'1.5'                                              GMA 1670
         BE    S3               USE AD HOC METHOD FOR A = 1.5           GMA 1680
         CE    FR0,=E'2.0'                                              GMA 1690
         BE    S4               AD HOC METHOD FOR A = 2.0               GMA 1700
         CE    FR0,=E'3.0'                                              GMA 1710
         BL    SGF              USE METHOD "GF" FOR A < 3.              GMA 1720
         BE    S6               AD HOC METHOD FOR A = 3.0               GMA 1730
*                                                                       GMA 1740
*                                                                       GMA 1750
*        SET UP FOR LARGE PARAMETER METHOD, ALGORITHM "GO"              GMA 1760
SGO      LA    R0,GO            SET ADDRESS FOR SUBSEQUENT CALLS        GMA 1770
         ST    R0,METHOD                                                GMA 1780
         LA    R0,40            INITIALIZE RANDOM ARRAY INDEX           GMA 1790
         ST    R0,INX1                                                  GMA 1800
         CE    FR0,AGO          TEST FOR NEW SHAPE PARAMETER            GMA 1810
         BE    GWAN             GO AHEAD IF NOT                         GMA 1820
         STE   FR0,AGO          SAVE NEW SHAPE PARM                     GMA 1830
         LE    FR2,=E'1.0'      GET CONSTANT 1.                         GMA 1840
         SER   FR0,FR2          COMPUTE MU = A - 1.                     GMA 1850
         STE   FR0,MU                                                   GMA 1860
         DER   FR2,FR0          COMPUTE MUP = 1 / MU                    GMA 1870
         STE   FR2,MUP                                                  GMA 1880
*                                                                       GMA 1890
*        LINK TO SQRT FUNCTION FOR SQRT(A)                              GMA 1900
*                                                                       GMA 1910
         LA    R1,ARGLST1       LOAD ARGUMENT LIST                      GMA 1920
         LR    R8,R15           SAVE BASE REGISTER                      GMA 1930
         L     R15,VADDSR       ADDRESS OF SQRT FUNCTION                GMA 1940
         BALR  R14,R15                                                  GMA 1950
         LR    R15,R8           RESTORE BASE REGISTER                   GMA 1960
         LER   FR2,FR0          SAVE SQRT(A)                            GMA 1970
         ME    FR0,=E'1.6329932'  FIND NORMAL VARIANCE                  GMA 1980
         AE    FR0,AGO                                                  GMA 1990
         STE   FR0,SIGMA                                                GMA 2000
```

```
**** GAMMA DEVIATE GENERATOR ****

         DE     FR0,MU           FIND REJECTION CONSTANT "WM"           GMA 2010
         SE     FR0,=E'1.0'                                             GMA 2020
         STE    FR0,WM                                                  GMA 2030
         AE     FR2,=E'1.6329932'  FIND REJECTION CONSTANT "VP"         GMA 2040
         DE     FR2,MU                                                  GMA 2050
         ME     FR2,=E'2.0'                                             GMA 2060
         STE    FR2,VP                                                  GMA 2070
*                                                                       GMA 2080
*        LINK TO SQRT FUNCTION TO FIND NORMAL STD DEV                   GMA 2090
*                                                                       GMA 2100
         LA     R1,ARGLST2       LOAD ARGUMENT LIST ADDRESS             GMA 2110
         L      R15,VADDSR       ADDRESS OF SQRT FUNCTION               GMA 2120
         BALR   R14,R15                                                 GMA 2130
         LR     R15,R8           RESTORE BASE REGISTER                  GMA 2140
         STE    FR0,SIGMA        SAVE STD DEV                           GMA 2150
*                                                                       GMA 2160
         ME     FR0,=E'2.4494897'  FIND REJECTION CONSTANT "DP"         GMA 2170
         LE     FR2,=E'1.0'                                             GMA 2180
         DER    FR2,FR0                                                 GMA 2190
         STE    FR2,DP                                                  GMA 2200
         STE    FR0,D                                                   GMA 2210
*                                                                       GMA 2220
         AE     FR0,MU           FIND UPPER LIMIT FOR NORMAL METHOD, "B"  GMA 2230
         STE    FR0,B                                                   GMA 2240
         LE     FR2,=E'1.0'      COMPUTE BP = 1 / B                     GMA 2250
         DER    FR2,FR0                                                 GMA 2260
         STE    FR2,BP                                                  GMA 2270
*                                                                       GMA 2280
         LE     FR2,SIGMA        COMPUTE REJECTION CONSTANT "CONS"      GMA 2290
         ME     FR2,D                                                   GMA 2300
         DER    FR2,FR0          FIRST FIND VALUE FOR LOG FUNCTION      GMA 2310
         STE    FR2,CONS                                                GMA 2320
         LA     R1,ARGLST3       LOAD ARG LIST ADDRESS                  GMA 2330
         L      R15,VADDLG       ADDRESS OF ALOG FUNCTION               GMA 2340
         BALR   R14,R15                                                 GMA 2350
         LR     R15,R8           RESTORE BASE ADDRESS                   GMA 2360
*                                                                       GMA 2370
         LCER   FR0,FR0          COMPLETE COMPUTATION OF "CONS"         GMA 2380
         SE     FR0,B                                                   GMA 2390
         AE     FR0,MU                                                  GMA 2400
         AE     FR0,MU                                                  GMA 2410
         AE     FR0,=E'3.7203285'                                       GMA 2420
         STE    FR0,CONS                                                GMA 2430
         B      GWAN             DONE WITH INITIALIZATION. PROCEED TO   GMA 2450
*                                  GENERATION                           GMA 2460
```

```
**** GAMMA DEVIATE GENERATOR ****

*
*
*        SET UP FOR FISHMAN'S METHOD, ALGORITHM "GF"
SGF      LA    R0,GF              SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         SE    FR0,=E'1.0'        COMPUTE AMINUS = A - 1
         STE   FR0,AMINUS
         LA    R0,20              INITIALIZE RANDOM ARRAY INDEX
         ST    R0,INX2
         B     GWAN               DONE WITH INITIALIZATION.  PROCEED TO
*                                    GENERATION.
*
*
*        SET UP FOR SMALL PARAMETER METHOD. "GS"
*
SGS      LA    R0,GS              SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         LER   FR2,FR0            COMPUTE 1 - A
         LE    FR4,=E'1.0'
         SER   FR2,FR4
         LCER  FR2,FR2
         STE   FR2,AMIN1
         DER   FR4,FR0            COMPUTE 1 / A
         STE   FR4,AINV
         ME    FR0,=E'.36787944'  FIND (E + A) / E
         AE    FR0,=E'1.0'
         STE   FR0,BGS
         LA    R0,40              INITIALIZE EXPONENTIAL ARRAY INDEX
         ST    R0,INX3
         B     GWAN               DONE WITH INITIALIZATION. GO ON
*                                    TO GENERATION.
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*
*        SET UP FOR AD HOC METHODS
*
*
*        SET UP FOR CHI-SQUARED, 1 DEGREE OF FREEDOM ( A = 0.5 )
*
S1       LA    R0,CHISQ1         SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         B     GWAN              GO ON TO GENERATION
*
*        SET UP FOR EXPONENTIAL ( A = 1.0 )
*
SEXPN    LA    R0,EXPN           SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         B     GWAN              GO ON TO GENERATION
*
*        SET UP FOR CHI-SQUARED, 3 DEGREES OF FREEDOM ( A = 1.5 )
*
S3       LA    R0,CHISQ3         SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         LA    R0,40             INITIALIZE RANDOM ARRAY INDEX
         ST    R0,INX4
         B     GWAN              GO ON TO GENERATION
*
*        SET UP FOR 2 - ERLANG ( A = 2.0 )
*
S4       LA    R0,CHISQ4         SET ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         LA    R0,40             INITIALIZE RANDOM ARRAY INDEX
         ST    R0,INX4
         B     GWAN              GO ON TO GENERATION
*
*        SET UP FOR 3 - ERLANG ( A = 3.0 )
*
S6       LA    R0,CHISQ6         SAVE ADDRESS FOR SUBSEQUENT CALLS
         ST    R0,METHOD
         LA    R0,40             INITIALIZE RANDOM ARRAY INDEX
         ST    R0,INX5
         B     GWAN              GO ON TO GENERATION
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*        METHOD "GO" (DIETER-AHRENS)
*
GO       LM     R8,R13,GOCON     LOAD LOOPING CONSTANTS
         CNOP   0,8              ALIGN BXLE LOOP FOR SPEED
*
GOLOOP   MR     R6,R8            GET NEXT UNIFORM RANDOM DEVIATE.
         SLDA   R6,1             R6 = REMAINDER; R7 = QUOTIENT.
         SRL    R7,1             ADD QUOTIENT TO REMAINDER THUS
         AR     R6,R7              SIMULATING DIVISION BY 2 ** 31 - 1
         BNO    *+10             GO ON IF NO OVERFLOW
         A      R6,=F'2147483645'   FIXUP OVERFLOW.  ADD 2 ** 31 - 3
         AR     R6,R2            ADD 4 MORE
         LR     R7,R6            PUT X(N) INTO R7.
         C      R7,=F'20556283'  SELECT NORMAL OR EXPONENTIAL
         BL     GOEXP                SAMPLING
*
*        REJECTION SAMPLING FROM THE NORMAL DISTRIBUTION
*
GONURM   BXLE   R12,R10,GONTST INCREMENT NORMAL ARRAY INDEX.
*                                NORMAL ARRAY EXHAUSTED. REPLENISH IT.
         ST     R7,IX            SAVE CURRENT SEED VALUE.
         LR     R12,R15          SAVE BASE REGISTER
         LA     R13,SVAREA       SAVE AREA POINTER
         LA     R1,ARGLST4       ARGUMENT LIST ADDRESS
         L      R15,VADDNM       ADDRESS OF NORMAL GENERATOR
         BALR   R14,R15          LINK TO "NORMAL"
         LR     R15,R12          RESTORE BASE REGISTER
         LA     R13,ENDGO        RESTORE END OF LOOP REGISTER
         SR     R12,R12          SET NORMAL ARRAY INDEX TO START
         L      R7,IX            RESTORE SEED
         CNOP   0,8              ALIGN BXLE LOOP FOR SPEED
*
GONTST   LE     FR0,RNARRAY(R12)  LOAD NEXT NORMAL DEVIATE
         LER    FR2,FR0          TRIAL GAMMA VALUE:
         ME     FR2,SIGMA           X = NORMAL * SIGMA + MU
         AE     FR2,MU
         BNP    GONORM           REJECT X < 0
         CE     FR2,B
         BH     GONORM           REJECT X > B
*
         LER    FR4,FR0          S2 = 0.5 * S * S
         MER    FR4,FR0
         HER    FR4,FR4
*
```

**** GAMMA DEVIATE GENERATOR ****

```
*         GET A UNIFORM FOR NORMAL REJECTION TEST
          MR     R6,R8              GET NEXT UNIFORM
          SLDA   R6,1               R6 = REMAINDER; R7 = QUOTIENT
          SRL    R7,1               ADD QUOTIENT TO REMAINDER THUS
          AR     R6,R7                SIMULATING DIVISION BY 2 ** 31 - 1
          BNO    *+10               GO ON IF NO OVERFLOW
          A      R6,=F'2147483645'   FIXUP OVERFLOW.  ADD 2 ** 31 - 3
          AR     R6,R2              ADD 4 MORE
          LR     R7,R6              PUT X(N) INTO R7
          SRL    R6,7               MAKE ROOM FOR EXPONENT.
          OR     R6,R9              "OR" ON THE EXPONENT
          ST     R6,UNIF            SAVE THE UNIFORM.
          LTER   FR0,FR0            PERFORM THE PROPER REJECTION, DEPENDING
          BP     GOPOS                 ON THE SIGN OF THE NORMAL
*
GONEG     ME     FR0,VP             COMPUTE THE REJECTION VALUE:
          SE     FR0,WM                1 + S2 * (S * VP - WM)
          MER    FR0,FR4
          AE     FR0,=E'1.0'
          CE     FR0,UNIF           REJECTION TEST
          BCR    2,R13              GO TO LOOP END IF PASSED.
          B      GON2TST            FURTHER TEST IF NOT.
*
GOPOS     LCER   FR0,FR4            COMPUTE THE REJECTION VALUE:
          ME     FR0,WM                1 - S2 * WM
          AE     FR0,=E'1.0'
          CE     FR0,UNIF           REJECTION TEST
          BCR    2,R13              GO TO LOOP END IF PASSED.
*
GON2TST   SER    FR4,FR2            FIND PARTIAL SUM FOR REJECTION TEST:
          AE     FR4,MU                SUM = MU - X + S2
          STE    FR4,SUM
          STE    FR2,X              SAVE TRIAL GAMMA DEVIATE
          ME     FR2,MUP            GET LOG ARGUMENT, X / MU
          STE    FR2,LOG
*
*         LINK TO LOG SUBROUTINE TWICE
*
          STM    R12,R13,GOSAVE      SAVE PROGRAM REGS
          LR     R12,R15            SAVE BASE REGISTER
          LA     R13,SVAREA         SAVE AREA POINTER
          LA     R1,ARGLST5         ARGUMENT LIST ADDRESS
          L      R15,VADDLG         ADDRESS OF FORTRAN LOG FUNCTION
          BALR   R14,R15
          LR     R15,R12            RESTORE BASE REGISTER
```

**** GAMMA DEVIATE GENERATOR ****

```
*
         ME     FRO,MU            ADD MU * LOG (X / MU) TO SUM
         AE     FRO,SUM           GET REJECTION VALUE
         STE    FRO,SUM
*
         LA     R1,ARGLST6        SECOND LINK TO LOG FUNCTION
         L      R15,VADDLG        ADDRESS OF LOG FUNCTION
         BALR   R14,R15
         LR     R15,R12           RESTORE BASE REGISTER
         LM     R12,R13,GOSAVE      RESTORE OTHER REGS
*
         LE     FR2,X             RELOAD TRIAL GAMMA
         CE     FRO,SUM           FINAL REJECTION TEST
         BCR    13,R13            PASSED TEST. GO TO LOOP END.
         B      GOLOOP            FAILED TEST.  BRANCH BACK FOR ANOTHER
*                                      TRY.
*
*
*        REJECTION SAMPLING FROM THE EXPONENTIAL DISTRIBUTION.
*
GOEXP    ST     R7,IX             GET TWO EXPONENTIAL DEVIATES.  FIRST
*                                      SAVE SEED.
         STM    R12,R13,GOSAVE     SAVE PROGRAM REGS.
         LR     R12,R15           SAVE BASE REGISTER.
         LA     R13,SVAREA        SAVE AREA POINTER
         LA     R1,ARGLST7        ARGUMENT LIST ADDRESS.
         L      R15,VADDEX        ADDRESS OF EXPONENTIAL GENERATOR.
         BALR   R14,R15           LINK TO "EXPON"
         LR     R15,R12           RESTORE BASE REGISTER.
*
         LE     FRO,RNEXP         FIND TRIAL GAMMA VALUE:
         ME     FRO,DP               X = B * (1 + R * DP)
         AE     FRO,=E'1.0'
         ME     FRO,B
         STE    FRO,X             SAVE TRIAL GAMMA VALUE
         ME     FRO,MUP           GET LOG (X / MU)
         STE    FRO,LOG
         LA     R1,ARGLST5        LOAD ARGUMENT LIST ADDRESS
         L      R15,VADDLG        ADDRESS OF LOG FUNCTION.
         BALR   R14,R15           LINK TO "ALOG"
         LR     R15,R12           RESTORE BASE REGISTER
         LM     R12,R13,GOSAVE     RESTORE OTHER REGS
*
```

```
**** GAMMA DEVIATE GENERATOR ****

         LE    FR2,X           RELOAD TRIAL GAMMA VALUE
         LER   FR4,FR2         COMPLETE CALCULATION OF REJECTION VALUE.
         ME    FR4,BP            MU * (LOG - X * BP) + CONS
         SER   FR0,FR4
         ME    FR0,MU
         AE    FR0,CONS
         LCER  FR0,FR0
         CE    FR0,RNEXP+4     PERFORM REJECTION TEST
         BH    GOLOOP          BACK TO START IF FAILED.
*
*        END OF METHOD "GO" LOOP.
*           GENERATED DEVIATE IS IN FR2.
*
ENDGO    STE   FR2,0(R4,R5)    STORE DEVIATE IN CALLER'S ARRAY.
         BXLE  R5,R2,GOLOOP    BRANCH BACK FOR ANOTHER DEVIATE.
         ST    R12,INX1        SAVE LAST ARRAY INDEX
         B     THRU            ALL DONE.  QUIT.
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*
*        FISHMAN'S METHOD
*
GF       ST     R7,IX              SET UP SEED
         LM     R8,R12,GFCON       LOAD LOOP CONSTANTS
         LR     R7,R15             SHIFT BASE REGISTER
         DROP   R15
         USING  GAMA,R7
         LR     R15,R9             KEEP "ALOG" ADDRESS IN R15
         CNOP   0,8                ALIGN BXLE LOOP FOR SPEED
*
GFLOOP   BXLE   R12,R10,GFTST      GET NEXT PAIR OF EXPONENTIALS
*                                  EXPONENTIAL ARRAY EXHAUSTED, REPLENISH IT
         LA     R1,ARGLST4         LOAD ARGUMENT LIST ADDRESS
         LR     R15,R8             ADDRESS OF "EXPON"
         BALR   R14,R15            LINK TO EXPONENTIAL GENERATOR
         LR     R15,R9             RESTORE ALOG ADDRESS TO R15
         SR     R12,R12            SET ARRAY INDEX TO START
         CNOP   0,8                ALIGN BXLE LOOP FOR SPEED
*
GFTST    L      R6,RNARRAY(R12)  TAKE LOGARITHM OF ONE EXPONENTIAL
         ST     R6,GFLOG            DEVIATE
         LA     R1,ARGLST8         LOAD ARGUMENT LIST ADDRESS
         BALR   R14,R15            LINK TO "ALOG"
         LE     FR2,RNARRAY(R12)  FINISH COMPUTING REJECTION VALUE:
         LER    FR4,FR2             (A - 1) * (R - LN R - 1)
         SER    FR4,FR0
         SE     FR4,=E'1.0'
         ME     FR4,AMINUS
         CE     FR4,RNARRAY+20(R12)   REJECTION TEST
         BH     GFLOOP
*                                  DELIVER  A * R
         ME     FR2,AP
         STE    FR2,0(R4,R5)       STORE DEVIATE IN CALLER'S ARRAY
         BXLE   R5,R2,GFLOOP       BRANCH BACK FOR ANOTHER DEVIATE
         LR     R15,R7             RESTORE BASE REGISTER
         DROP   R7
         USING  GAMA,R15
         L      R7,IX              RELOAD SEED
         ST     R12,INX2           SAVE LAST ARRAY INDEX
         B      THRU               QUIT
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*         AD HOC METHODS
*
*         A = 0.5, 1.0, 1.5, 2.0 OR 3.0
*
*
*         CHI - SQUARED, 1 DEGREE OF FREEDOM ( A = 0.5 )
*
CHISQ1    LR    R12,R15          SAVE BASE REGISTER
          LA    R1,4(,R1)        SKIP OVER SHAPE PARAMETER IN ARG LIST
          L     R15,VADDNM       LINK TO "NORMAL"
          BALR  R14,R15
          LR    R15,R12          RESTORE BASE REGISTER
          L     R7,0(,R1)        GET SEED VALUE IN REG 7
          L     R7,0(,R7)
          CNOP  0,8              ALIGN BXLE LOOP FOR SPEED
*
CHLOOP1   LE    FR0,0(R4,R5)     GET NEXT NORMAL
          MER   FR0,FR0          SQUARE THE NORMAL
          HER   FR0,FR0             AND MULTIPLY BY 0.5
          STE   FR0,0(R4,R5)     PUT GAMMA DEVIATE INTO CALLER'S ARRAY
          BXLE  R5,R2,CHLOOP1    BRANCH BACK FOR NEXT NORMAL
          B     THRU             QUIT
*
*         EXPONENTIAL METHOD ( A = 1.0 )
*
EXPN      LR    R12,R15          SAVE BASE REGISTER
          LA    R1,4(,R1)        SKIP OVER SHAPE PARM IN ARG LIST
          L     R15,VADDEX       LINK DIRECTLY TO "EXPON"
          BALR  R14,R15
          LR    R15,R12          RESTORE BASE REGISTER
          L     R7,0(,R1)        GET SEED VALUE IN R7
          L     R7,0(,R7)
          B     THRU             QUIT.
*
```

**** GAMMA DEVIATE GENERATOR ****

```
*          CHI - SQUARED, 3 DEGREES OF FREEDOM ( A = 1.5 )                     GMA 5500
*                                                                              GMA 5510
CHISQ3    LR     R6,R15            SHIFT BASE REGISTER                         GMA 5520
          DROP   R15                                                           GMA 5530
          USING  GAMA,R6                                                       GMA 5540
          LA     R1,4(,R1)         SKIP OVER SHAPE PARAMETER IN ARG LIST       GMA 5550
          L      R15,VADDEX        LINK TO "EXPON"                             GMA 5560
          BALR   R14,R15                                                       GMA 5570
          L      R7,0(,R1)         GET LAST SEED VALUE USED                    GMA 5580
          L      R7,0(,R7)                                                     GMA 5590
          ST     R7,IX             SAVE SEED VALUE                             GMA 5600
          LM     R10,R12,CHICON3     LOAD LOOP CONSTANTS                       GMA 5610
          CNOP   0,8               ALIGN BXLE LOOP FOR SPEED                   GMA 5620
*                                                                              GMA 5630
CHLOOP3   BXLE   R12,R10,CH3COMP       GET NEXT NORMAL                         GMA 5640
*                                  NORMAL ARRAY EXHAUSTED. REPLENISH IT.       GMA 5650
          L      R15,VADDNM        PUT ADDRESS OF "NORMAL" INTO R15            GMA 5660
          LA     R1,ARGLST4        GET ARGUMENT LIST                           GMA 5670
          BALR   R14,R15           LINK TO "NORMAL"                            GMA 5680
          SR     R12,R12           RESET ARRAY INDEX                           GMA 5690
*                                                                              GMA 5700
CH3COMP   LE     FR0,RNARRAY(R12)  LOAD NEW NORMAL                             GMA 5710
          MER    FR0,FR0           SQUARE NORMAL                               GMA 5720
          HER    FR0,FR0              AND HALVE IT                             GMA 5730
          AE     FR0,0(R4,R5)      ADD EXPONENTIAL TO CHI-SQUARED IN REG 0     GMA 5740
          STE    FR0,0(R4,R5)      STORE GENERATED GAMMA IN CALLER'S ARRAY     GMA 5750
          BXLE   R5,R2,CHLOOP3     GO BACK FOR ANOTHER DEVIATE                 GMA 5760
*                                                                              GMA 5770
          L      R7,IX             LOAD LAST SEED VALUE                        GMA 5780
          ST     R12,INX4          SAVE RANDOM ARRAY INDEX                     GMA 5790
          LR     R15,R6            RESTORE BASE REGISTER                       GMA 5800
          B      THRU              QUIT                                        GMA 5810
```

**** GAMMA DEVIATE GENERATOR ****

```
*                                                                        GMA 5820
*         2 - ERLANG ( A = 2.0 )                                         GMA 5830
*                                                                        GMA 5840
CHISQ4    LR     R6,R15            SHIFT BASE REGISTER                   GMA 5850
          LA     R1,4(,R1)         SKIP OVER SHAPE PARAMETER IN ARG LIST GMA 5860
          L      R15,VADDEX        LINK TO "EXPON"                       GMA 5870
          BALR   R14,R15                                                 GMA 5880
          L      R7,0(,R1)         GET LAST SEED VALUE USED              GMA 5890
          L      R7,0(,R7)                                               GMA 5900
          ST     R7,IX             SAVE SEED VALUE                       GMA 5910
          LM     R10,R12,CHICON3   LOAD LOOP CONSTANTS                   GMA 5920
          CNOP   0,8               ALIGN BXLE LOOP FOR SPEED             GMA 5930
*                                                                        GMA 5940
CHLOOP4   BXLE   R12,R10,CH4COMP   GET NEXT EXPONENTIAL                  GMA 5950
*                                  EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT GMA 5960
          L      R15,VADDEX        LINK TO "EXPON"                       GMA 5970
          LA     R1,ARGLST4        GET ARGUMENT LIST                     GMA 5980
          BALR   R14,R15           LINK TO "EXPON"                       GMA 5990
          SR     R12,R12           RESET ARRAY INDEX TO ZERO             GMA 6000
*                                                                        GMA 6010
CH4COMP   LE     FR0,RNARRAY(R12)  LOAD NEW EXPONENTIAL                  GMA 6020
          AE     FR0,0(R4,R5)      ADD TO SECOND EXPONENTIAL             GMA 6030
          STE    FR0,0(R4,R5)      STORE GENERATED GAMMA IN CALLER'S ARRAY GMA 6040
          BXLE   R5,R2,CHLOOP4     GO BACK FOR NEXT DEVIATE              GMA 6050
*                                                                        GMA 6060
          L      R7,IX             LOAD LAST SEED VALUE                  GMA 6070
          ST     R12,INX4          SAVE RANDOM ARRAY INDEX               GMA 6080
          LR     R15,R6            RESTORE BASE REGISTER                 GMA 6090
          B      THRU              QUIT                                  GMA 6100
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*          3 - ERLANG ( A = 3.0 )
*
CHISQ6     LR     R6,R15           SHIFT BASE REGISTER
           LA     R1,4(,R1)        SKIP OVER SHAPE PARAMETER IN ARG LIST
           L      R15,VADDEX       LINK TO "EXPON"
           BALR   R14,R15
           L      R7,0(,R1)        GET LAST SEED VALUE USED
           L      R7,0(,R7)
           ST     R7,IX            SAVE SEED VALUE
           LM     R10,R12,CHICON6  LOAD LOOP CONSTANTS
           CNOP   0,8              ALIGN BXLE LOOP FOR SPEED
*
CHLOOP6    BXLE   R12,R10,CH6COMP  GET NEXT PAIR OF EXPONENTIALS
*                                  EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
           L      R15,VADDEX       LINK TO "EXPON"
           LA     R1,ARGLST4       GET ARGUMENT LIST
           BALR   R14,R15          LINK TO "EXPON"
           SR     R12,R12          RESET ARRAY INDEX
*
CH6COMP    LE     FR0,RNARRAY(R12)    LOAD NEW EXPONENTIAL
           AE     FR0,RNARRAY+20(R12)  ADD TWO INDEPENDENT EXPONENTIALS
           AE     FR0,0(R4,R5)
           STE    FR0,0(R4,R5)     SAVE GENERATED GAMMA IN CALLER'S ARRAY
           BXLE   R5,R2,CHLOOP6    GO BACK FOR NEXT DEVIATE
*
           L      R7,IX            LOAD LAST SEED VALUE
           ST     R12,INX5         SAVE RANDOM ARRAY INDEX
           LR     R15,R6           RESTORE BASE REGISTER
           DROP R6
           USING GAMA,R15
           B      THRU             QUIT
```

```
**** GAMMA DEVIATE GENERATOR ****

*
*         SMALL PARAMETER METHOD "GS" (AHRENS)
*
GS        LM     R8,R12,GSCON    LOAD LOOP CONSTANTS
          CNOP   0,8             ALIGN BXLE LOOP FOR SPEED
*
GSLOOP    MR     R6,R8           GET NEXT UNIFORM DEVIATE
          SLDA   R6,1            R6 = REMAINDER; R7 = QUOTIENT
          SRL    R7,1            ADD QUOTIENT TO REMAINDER THUS
          AR     R6,R7              SIMULATING DIVISION BY 2 ** 31 - 1
          BNO    *+10            GO ON IF NO OVERFLOW
          A      R6,=F'2147483645'  FIXUP OVERFLOW.  ADD 2 ** 31 - 3
          AR     R6,R2           ADD 4 MORE
          LR     R7,R6           PUT X(N) INTO R7
          SRL    R6,7            MAKE ROOM FOR EXPONENT
          OR     R6,R9           "OR" ON THE EXPONENT
          ST     R6,UNF          SAVE UNIFORM DEVIATE
          LE     FR0,UNF
          ME     FR0,BGS         FIND P = B * UNIFORM
          STE    FR0,P
*
          LM     R8,R9,GSVCON    LOAD FUNCTION ADDRESSES
          LR     R6,R15          SHIFT BASE REGISTER TO R6
          DROP   R15
          USING  GAMA,R6
*
*         SAMPLE FROM EXPONENTIAL DISTRIBUTION FOR REJECTION TEST
*
          BXLE   R12,R10,GSTST   GET NEXT EXPONENTIAL IN ARRAY
*                                EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
          ST     R7,IX           SAVE SEED VALUE
          LA     R1,ARGLST4      LOAD ARGUMENT LIST ADDRESS
          L      R15,VADDEX      LINK TO "EXPON"
          BALR   R14,R15
          SR     R12,R12         RESET ARRAY INDEX TO START
          LE     FR0,P           RELOAD P INTO FR0
          L      R7,IX           RESTORE SEED TO R7
          CNOP   0,8             ALIGN BXLE FOR SPEED
*
GSTST     CE     FR0,=E'1.0'     FIND REJECTION METHOD TO USE
          BH     XBIG
*
XLO       LA     R1,ARGLST9      FIND LOG (P). LOAD ARGUMENT LIST ADD
          LR     R15,R9          ADDRESS OF LOG FUNCTION
          BALR   R14,R15
```

```
**** GAMMA DEVIATE GENERATOR ****

         ME     FRO,AINV          GET LOG (P) / A
         STE    FRO,P
         LR     R15,R8            LINK TO EXPONENTIAL FUNCTION.
         LA     R1,ARGLST9        LOAD ARGUMENT LIST ADDRESS
         BALR   R14,R15           RESULT IS P ** (1 / A)
         CE     FRO,RNARRAY(R12)  REJECTION TEST
         BNH    ENDGS             QUIT IF OK,
         LM     R8,R9,GSCON       OTHERWISE GO BACK
         LR     R15,R6            RESET BASE REGISTER
         B.     GSLOOP
*
XBIG     LE     FR2,BGS           FIND (B - P) / A
         SER    FR2,FRO
         ME     FR2,AINV
         STE    FR2,P
         LA     R1,ARGLST9        NOW LINK TO LOG FUNCTION:
         LR     R15,R9            ADDRESS OF LOG FUNCTION
         BALR   R14,R15           RESULT IS LOG ( (B - P) / A )
         LCER   FRC,FRO           TRIAL GAMMA IS - LOG
         STE    FRO,P             NOW FIND LOG OF TRIAL VALUE
         LA     R1,ARGLST9        LOAD ARGUMENT LIST ADDRESS
         LR     R15,R9            ADDRESS OF LOG FUNCTION
         BALR   R14,R15
         ME     FRO,AMIN1         FINISH CALCULATION OF REJECTION VALUE
         CE     FRO,RNARRAY(R12)  REJECTION TEST
         LE     FRO,P             RELOAD TRIAL GAMMA VALUE
         BNH    ENDGS             QUIT IF OK
         LM     R8,R9,GSCON       OTHERWISE RESET LOOP CONSTANTS
         LR     R15,R6            AND CHANGE BASE REGISTER
         B      GSLOOP            AND GO BACK
*
*        END OF GSLOOP
*
*        GAMMA VARIATE VALUE IS IN FRO
ENDGS    STE    FRO,0(R4,R5)      STORE DEVIATE IN CALLER'S ARRAY
         LM     R8,R9,GSCON       RESET LOOP CONSTANTS
         LR     R15,R6            SHIFT BASE REGISTER
         BXLE   R5,R2,GSLOOP      BRANCH BACK FOR ANOTHER DEVIATE
         ST     R12,INX3          SAVE LAST ARRAY INDEX
         B      THRU              OTHERWISE QUIT.
         DROP   R6
         USING  GAMA,R15
```

**** GAMMA DEVIATE GENERATOR ****

```
*
*          END OF ROUTINE.
*
THRU       L      R13,SVAREA+4      RESTORE CALLING SAVE AREA.
           L      R1,24(,R13)       GET ARGUMENT LIST ADDRESS
           L      R4,4(,R1)         GET SEED ADDRESS
           ST     R7,0(,R4)         SEND BACK LAST SEED USED.
           LM     R14,R12,12(R13)     RESTORE CALLING REGS
           BR     R14               RETURN
           EJECT
           DS     0D
*
*          DATA AREA
*
SVAREA     DS     18F               SAVE AREA
*
*
AP         DC     E'-1.0'           OLD SHAPE PARAMETER
METHOD     DS     F                 ADDRESS FOR PROPER METHOD
*
VADDEX     DC     V(EXPON)          EXTERNAL EXPONENTIAL GENERATOR
VADDNM     DC     V(NORMAL)         EXTERNAL NORMAL GENERATOR
VADDLG     DC     V(ALOG)           LOGARITHM FUNCTION
VADDSR     DC     V(SQRT)           SQUARE ROOT FUNCTION
*
IX         DS     F                 RANDOM NUMBER SEED
RNARRAY    DS     10F               ARRAY FOR NORMAL OR EXPONENTIAL DEVIATES
NUM        DC     F'10'             NUMBER OF DEVIATES TO BE DELIVERED
*
*          CONSTANTS FOR METHOD "GO"
*
AGO        DC     E'5.0'            SHAPE PARAMETER
MU         DC     E'4.0'            NORMAL MEAN
SIGMA      DC     E'2.9413405'      NORMAL STD DEV
B          DC     E'11.204783'      UPPER LIMIT FOR NORMAL
MUP        DC     E'0.25'           1 / MU
BP         DC     E'.089247598'     1 / B
DP         DC     E'.13879668'      MISC
WM         DC     E'1.1628709'          CONSTANTS
VP         DC     E'1.9345306'              FOR
CONS       DC     E'-.12172460'               "GO"
```

**** GAMMA DEVIATE GENERATOR ****

```
*                                                                  GMA 7730
GOCON    DC    F'16807'       UNIFORM MULTIPLIER                   GMA 7740
         DC    X'40000001'    EXPONENT CONSTANT                    GMA 7750
         DC    F'4'           NORMAL ARRAY INDEX INCREMENT         GMA 7760
         DC    F'36'          INDEX LIMIT                          GMA 7770
INX1     DC    F'40'          ARRAY INDEX                          GMA 7780
         DC    AL4(ENDGO)     END OF "GO" LOOP                     GMA 7790
*                                                                  GMA 7800
D        DS    F              TEMP STORAGE                         GMA 7810
SUM      DS    F                  FOR                              GMA 7820
LOG      DS    F                    INTERMEDIATE                   GMA 7830
UNIF     DS    F                       RESULTS                     GMA 7840
X        DS    F              TRIAL GAMMA DEVIATE                  GMA 7850
GOSAVE   DS    2F             REGISTER STORAGE                     GMA 7860
RNEXP    DS    2F             ARRAY FOR EXPONENTIAL SAMPLING       GMA 7870
NGO1     DC    F'2'           NUMBER OF EXPONENTIALS               GMA 7880
*                                                                  GMA 7890
*        CONSTANTS FOR METHOD "GF"                                 GMA 7900
*                                                                  GMA 7910
AMINUS   DS    F              A - 1                                GMA 7920
GFCON    DC    V(EXPON)       ADDRESS OF EXPONENTIAL GENERATOR     GMA 7930
         DC    V(ALOG)        ADDRESS OF LOG FUNCTION              GMA 7940
         DC    F'4'           EXPONENTIAL ARRAY INDEX INCREMENT    GMA 7950
         DC    F'1o'          EXPONENTIAL ARRAY INDEX LIMIT        GMA 7960
INX2     DC    F'40'          EXPONENTIAL ARRAY INDEX              GMA 7970
GFLOG    DS    F              TEMP STORAGE                         GMA 7980
*                                                                  GMA 7990
*        CONSTANTS FOR METHOD "GS"                                 GMA 8000
*                                                                  GMA 8010
AINV     DS    F              1 / A                                GMA 8020
AMIN1    DS    F              1 - A                                GMA 8030
BGS      DS    F              (E + A) / E                          GMA 8040
GSCON    DC    F'16807'       UNIFORM MULTIPLIER                   GMA 8050
         DC    X'40000001'    EXPONENT CONSTANT                    GMA 8060
         DC    F'4'           EXPONENTIAL ARRAY INDEX INCREMENT    GMA 8070
         DC    F'36'          EXPONENTIAL ARRAY INDEX LIMIT        GMA 8080
INX3     DC    F'40'          EXPONENTIAL ARRAY INDEX              GMA 8090
GSVCON   DC    V(EXP)         EXTERNAL FUNCTION                    GMA 8100
         DC    V(ALOG)           ADRESSES                          GMA 8110
UNF      DS    F              TEMPORARY STORAGE                    GMA 8120
P        DS    F                  LOCATIONS                        GMA 8130
```

```
**** GAMMA DEVIATE GENERATOR ****

*
*
*          CONSTANTS FOR AD HOC METHODS
*
CHICON3   DC      F'4'                NORMAL ARRAY INDEX INCREMENT
          DC      F'36'               NORMAL ARRAY INDEX LIMIT
INX4      DC      F'40'               NORMAL ARRAY INDEX
*
CHICON6   DC      F'4'                ARRAY INDEX INCREMENT
          DC      F'16'               ARRAY INDEX LIMIT
INX5      DC      F'40'               ARRAY INDEX
*
*
*          ARGUMENT LISTS
*
ARGLST1   DC      X'FF'               CALL TO SQRT IN "GO" SET UP
          DC      AL3(AGO)
ARGLST2   DC      X'FF'               2ND CALL TO SQRT IN "GO" SET UP
          DC      AL3(SIGMA)
ARGLST3   DC      X'FF'               CALL TO ALOG IN "GO" SETUP
          DC      AL3(CONS)
ARGLST4   DC      AL4(IX)             CALLS TO REPLENISH RNARRAY
          DC      AL4(RNARRAY)
          DC      X'FF'
          DC      AL3(NUM)
ARGLST5   DC      X'FF'               CALL TO ALOG IN NORMAL SECTION OF "GO"
          DC      AL3(LOG)
ARGLST6   DC      X'FF'               CALL TO ALOG IN EXPON SECTION OF "GO"
          DC      AL3(UNIF)
ARGLST7   DC      AL4(IX)             CALL TO EXPONENTIAL GENERATOR IN "GO"
          DC      AL4(RNEXP)
          DC      X'FF'
          DC      AL3(NGO1)
ARGLST8   DC      X'FF'               CALL TO ALOG IN METHOD "GF"
          DC      AL3(GFLOG)
ARGLST9   DC      X'FF'               FUNCTION CALLS IN METHOD "GS"
          DC      AL3(P)
          LTORG
          END
```

INITIAL DISTRIBUTION LIST

	No. Copies
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12
Dean of Research Code 023 Naval Postgraduate School Monterey, CA 93940	1
Library (Code 0212) Naval Postgraduate School Monterey, CA 93940	2
Library (Code 55) Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, CA 93940	3
Marvin Denicoff Office of Naval Research Arlington, Virginia 22217	1
Dr. Thomas Varley Office of Naval Research Arlington, Virginia 22217	1
Dr. Bruce McDonald Office of Naval Research Arlington, Virginia 22217	1
Professor J. H. Ahrens Nova Scotia Technical College Department of Applied Mathematics Halixfax, Nova Scotia, Canada	1
Richard V. Andree Department of Information and Computer Sciences University of Oklahoma Norman, Oklahoma 73069	1
Julius Aronofsky University Computing Corporation 1500 UCC Tower P. O. Box 6228 Dallas, Texas 75222	1

Professor James N. Arvesen 1
Department of Mathematical Statistics
Columbia University
New York, New York 10027

George Atkins 1
Department of Computer Science
Southwestern State College
Weatherford, Oklahoma 73096

T. E. Bailey 1
Computing and Information Systems
Oklahoma State University
Stillwater, Oklahoma 74074

Lee J. Bain 1
Math Department
University of Missouri at Rolla
Rolla, Missouri

Jeff Bangert 1
Computation Center
Kansas University
Lawrence, Kansas 66044

R. B. Breitenbach 1
College of Business Administration
Oklahoma State University
Stillwater, Oklahoma 74074

Lyle Broemeling 1
Department of Math and Science
Oklahoma State University
Stillwater, Oklahoma 74074

David Campbell 1
Naval Weapons Systems Analysis Office
Bldg. 210 Second Deck
Washington Navy Yard
Washington, D. C. 20374

Gary Carlson 1
Computer Research Center
Bringham Young University
Provo, Utah 84601

John P. Chandler 1
Computing and Information Systems
Oklahoma State University
Stillwater, Oklahoma 74074

Claude Cohen 1
Computer Center
Northwestern University
2129 Sheridan Road
Evanston, Illinois 60201

Eli Cohen 1
Vogelback Computer Center
2129 Sheridan
Northwestern University
Evanston, Illinois 60201

Herbert T. Davis 1
Department of Mathematics
University of New Mexico
Albuquerque, New Mexico 87106

Arthur D. Dayton 1
Department of Statistics
Kansas State University
Manhattan, Kansas 66502

Hamed Eldin 1
Department of Industrial Engineering
Oklahoma State University
Stillwater, Oklahoma 74074

Professor William F. Fellner 1
Virginia Commonwealth University
Department of Biometry
MCV Station, Box 32
Richmond, Virginia 23298

Donald Fisher 1
Computing Information Science Dept.
Oklahoma State University
Stillwater, Oklahoma 74074

J. L. Folks 1
Department of Math and Science
Oklahoma State University
Stillwater, Oklahoma 74074

Professor A. V. Gafarian 1
Department of Industrial and Systems Engineering
University of Southern California
Los Angeles, CA 90007

Charles E. Gates 1
Institute of Statistics
Texas A & M University
College Station, Texas 77843

Dennis E. Grawoig 1
Georgia State University
33 Gilmer Street, N. E.
Atlanta, Georgia 30303

Joseph L. Gray 1
Oklahoma State University
University Computer Center
Stillwater, Oklahoma 74074

Robert Gumm 1
Computer Center
Oklahoma State University
Stillwater, Oklahoma 74074

G. E. Hedrick 1
Computer Information Science Department
Oklahoma State University
Stillwater, Oklahoma 74074

Dr. David C. Hoaglin 1
National Bureau of Economic Research, Inc.
575 Technology Square
Cambridge, Mass. 02139

William G. Hunter 1
Department of Statistics
University of Wisconsin
Madison, Wisconsin 53706

William J. Kennedy 1
Statistical Lab
Iowa State University
Ames, Iowa 50010

W. Kirby 1
U.S. Department of the Interior
Geological Survey
National Center, Mail Stop #430
Reston, Virginia 22090

Professor George Marsaglia 1
Department of Computing Science
McGill University
Montreal, P. Q., Canada

Ronald McNew 1
Department of Math And Statistics
Oklahoma State University
Stillwater, Oklahoma 74074

Robert Morrison 1
Department of Math and Statistics
Oklahoma State University
Stillwater, Oklahoma 74074

Wesley L. Nicholson 1
Mathematics and Physics Research
Battelle Northwest
P. O. Box 999
Richmond, Washington 99352

Billie J. Pease 1
Department of the Interior
Geological Survey
Computer Center Division
18th and C Streets, N. W.
Room 1452
Washington, D. C. 20242

William S. Peters 1
University of New Mexico
Albuquerque, New Mexico 87106

Norval F. Pohl 1
Department of Quantative Methods
University of Santa Clara
Santa Clara, CA

Julius Reichbach 1
Technological University
Mathematics Room 2004
Holland, Delft 8
Julianalaam 132, The Netherlands

David P. Rutten 1
Graduate School of Business
Indiana University
Bloomington, Indiana 47401

David J. Schumacher 1
Lockheed Missiles and Space Company
Sunnyvale, CA 94088

W. T. Stille 1
Eastman Kodak
343 State Street
Rochester, New York

Sundaram Swetharanyam 1
Main Campus Computer Center
McNeese State University
Lake Charles, Louisiana 70601

Jack Testerman 1
University of Southwestern Louisiana
Box 940
Lafayette, Louisiana 70501

Carolyn S. Thompson 1
Medical Center
University of Arkansas
Little Rock, Arkansas 72201

W. M. Usher 1
O.S.U. Computing Center
Oklahoma State University
Stillwater, Oklahoma 74074

James Van Doren 1
Computing Information Sciences Department
Oklahoma State University
Stillwater, Oklahoma 74074

Wray Wilkes 1
Computing Center
University of Arkansas
Fayetteville, Arkansas

Sing-Chou Wu 1
Computer Science and Statistics Department
California Polytechnic College
San Luis Obispo, California

Geoffrey Gates 1
400 Computer Center
Michigan State University
E. Lansing, Michigan 48823

W. V. Accola 1
Computer Center
Oklahoma State University
Stillwater, Oklahoma 74074

Professor Amrit L. Goel 1
427 Linil Hall
Syracuse University
Syracuse, New York 13210

Y. C. Lu 1
College of Agriculture
Oklahoma State University
Stillwater, Oklahoma 74074

John W. Meredith 1
Stephen F. Austin State University
Box 6163
Nacogdoches, Texas 75961

John M. Chambers 1
Bell Telephone Laboratories
Murray Hill, New Jersey 97974

Patrick L. Odell 1
Texas Technical University
Department of Mathematics
Lubbock, Texas 79409

Prof. W. Morven Gentlemen 1
Dept. Computer Science
University of Waterloo
Waterloo
Ontario, Canada

Prof. W. J. Hemmerle 1
Computer Lab
University of Rhode Island
Kingston, Rhode Island 02881

Alan S. Galbraith 1
Math Division
U. S. Army Research Office
Box CM
Duke Station
Durham, North Carolina 27706

Dean W. M. Woods, Code 024 1
Dean of Educational Development
Naval Postgraduate School
Monterey, CA 93940

Dean W. F. Koehler, Code 021 1
Dean of Programs
Naval Postgraduate School
Monterey, CA 93940

Captain D. W. Kiley, Code 03 1
Director of Programs
Naval Postgraduate School
Monterey, CA 93940

Professor D. G. Williams, Code 0211 1
Director, Computer Center
Naval Postgraduate School
Monterey, California 93940

Prof. D. E. Harrison, Jr., Code 61Hx 1
Prof. R. L. Kelly, Code 61Ke
Dept. of Physics and Chemistry
Naval Postgraduate School
Monterey, CA 93940

Prof. J. A. Galt 1
Department of Oceanography
Naval Postgraduate School
Monterey, CA 93940

Prof. R. E. Ball 1
Department of Aeronautics
Naval Postgraduate School
Monterey, CA 93940

Prof. G. L. Barksdale, Jr., Code 72 1
Prof. U. R. Kodres, Code 72Kr 1
Prof. V. M. Powers, Code 72Pw 1
Computer Science Group
Naval Postgraduate School
Monterey, CA 93940

Prof. L. Kovach, Code 53 1
Prof. T. Jayachandran, Code 53Jy 1
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93940

Prof. A. F. Andrus, Code 55As 1
Prof. D. R. Barr, Code 55Bn 1
Prof. G. G. Brown, Code 55Zr 1
Prof. G. Bradley, Code 55Bz 1
Prof. R. W. Butterworth, Code 55Bd 1
Prof. D. P. Gaver, Code 55Gv 1
Prof. J. K. Hartman, Code 55Hh 1
Prof. G. T. Howard, Code 55Hk 1
Prof. K. T. Marshall, Code 55Mt 1
Prof. W. M. Raike, Code 55Rj 1
Prof. F. R. Richards, Code 55Rh 1
Prof. R. H. Shudde, Code 55Su 1
Prof. N. F. Schneidewind, Code 55Ss 1
Prof. M. U. Thomas, Code 55To 1
Prof. J. B. Tysver, Code 55Ty 1
Prof. D. R. Whipple, Code 55Wp 1
Prof. P.A.W. Lewis, Code 55Lw 12
Department of Operations Research
and Administrative Sciences
Naval Postgraduate School
Monterey, CA 93940

Mr. G. P. Learmonth, Code 0211 1
Mathematician
Computer Center
Naval Postgraduate School
Monterey, CA 93940

LT. D. W. Robinson, Code 72Ro 12
Computer Science Group
Naval Postgraduate School
Monterey, CA 93940