

AD A057230

AD No. _____
DDC FILE COPY

(14)

NPS-72R075041

(2)

LEVEL

NAVAL POSTGRADUATE SCHOOL
Monterey, California



DDC
RECEIVED
AUG 9 1978
B

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

W./Robinson

A. W./Lewis

(11)

Apr 1975

(12)

5

4

Approved for public release; distribution unlimited

78 07 4 008

251 450

fu

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.

Reproduction of all or part of this report is authorized.

This report was prepared by:

David W. Robinson

DAVID W. ROBINSON

Instructor of Computer Science

Peter A. W. Lewis

PETER A. W. LEWIS, Professor

Department of Operations Research
and Administrative Science

Reviewed by:

Released by:

G. L. Barksdale, Jr.

G. L. BARKSDALE, JR.

Chairman

Computer Science Group

R. R. Fossum

R. R. FOSSUM

Dean of Research

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
		16a. 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		

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.

ACCESS		
NTIS		☒
DDI		☐
DDP		☐
DDI		☐
BY		
DISTRIBUTION		
Dist.	AVAIL	SPECIAL
A		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

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.

NONUNIFORM RANDOM NUMBER PACKAGE

TABLE OF CONTENTS

	Page
I. Introduction	1
II. Use of the Subroutines	4
III. Description of the Algorithms	9
A. Cauchy Generator	9
B. Gamma Generator GS: $\lambda < 1.0$	12
C. Gamma Generator GF: $1.0 < \lambda < 3.0$	13
D. Gamma Generator GO: $\lambda > 3.0$	15
E. Ad Hoc Gamma Generators	17
IV. Summary and Conclusions	19
References	21
Listing for Subroutine CAUCHY	22
Listing for Subroutine GAMA	28
Distribution List	51

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

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

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 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 λ and scale parameter s has the density function

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

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

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

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

$$E[X] = \lambda / s,$$

$$V[X] = \lambda / s^2 .$$

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

We note that if X has a $\Gamma(\lambda, 1)$ distribution then X/s has a $\Gamma(\lambda, s)$ 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 $X(1), \dots, X(N)$ of Cauchy or $\Gamma(A, 1.0)$ 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	<u>1165</u> 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, λ . Note that since special methods are employed when λ 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 λ .

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 λ .

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.

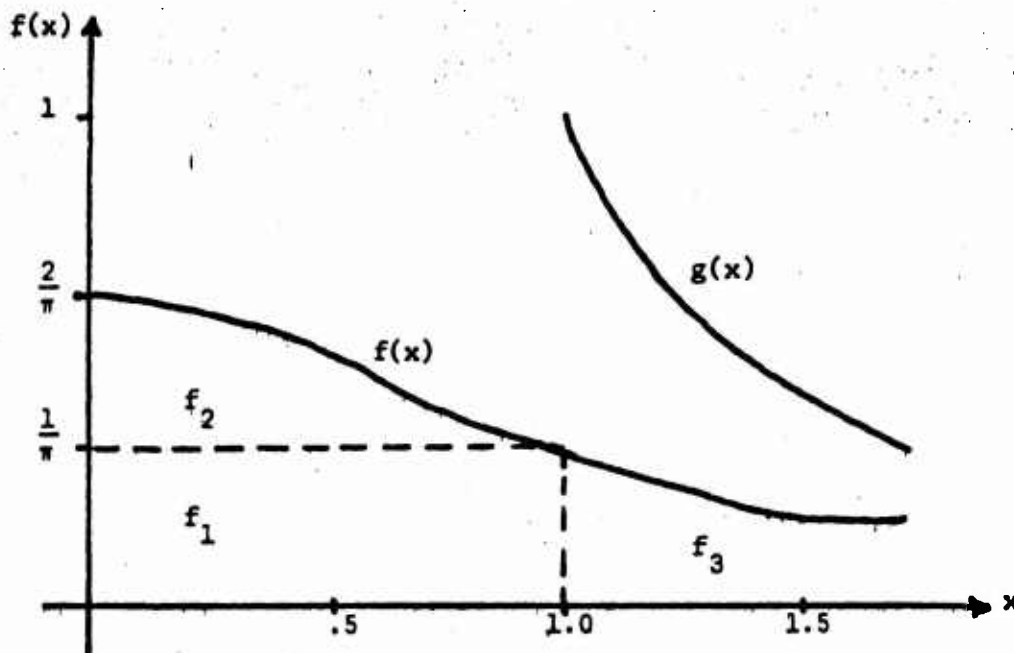


Figure 1. Decomposition of the Cauchy Density Function.

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 $U \leq 0.6366197724 = 2/\pi$, 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 $U_2 \leq 0.8284271247 = 2\sqrt{2} - 2$, go to Step 8.
5. (Hard rejection) If $U_2 - U_1 \leq \frac{1 - x^2}{1 + x^2} (2\sqrt{2} - 2)$, 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 $U^* \leq \frac{x^2}{1 + x^2}$ 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 $Y_1 = 2 U_1 - 1$ and $Y_2 = 2 U_2 - 1$.
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: $A \leq 1.0$

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].

Algorithm GS. 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. Gamma Generator GF: $1.0 \leq A \leq 3.0$

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 G0 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 λ . Let

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

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

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

Proof:

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

Therefore,

$$(4) \quad f_E(x | U \leq g(E)) = \frac{h(x) \Pr\{U \leq g(E) | E=x\}}{\Pr\{U \leq g(E)\}}.$$

Now since U is uniformly distributed,

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

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

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

Thus, in view of (4),

$$f_E(x|U \leq g(E)) = \frac{h(x)g(x)}{C(\lambda)}$$

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

The efficiency of the generator is governed by the probability that a given variate will pass the rejection test, $U \leq g(E)$; from (5) it will be seen that this probability is just $C(\lambda)$. When λ is large we have from Stirling's approximation that $C(\lambda) \approx \sqrt{\frac{2\pi}{\lambda}} \frac{e^{-\lambda}}{\lambda^{\lambda-1/2}}$, so that the method becomes more inefficient with increasing λ , as noted above.

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

Algorithm GF. Gamma variates, $1.0 < \lambda < 3.0$.

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

D. Gamma Generator GQ: $\lambda \geq 3.0$

This method was originally developed by Dieter and Ahrens and is fully described in [1] together with several other gamma generation techniques. Algorithm GQ does not

suffer the usual drawback of growing less efficient in generation time with increasing λ ; in fact, the method is more efficient for larger λ 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 λ greater than 2.533, but it is not as efficient as Fishman's technique for $\lambda < 3.0$.

As mentioned previously, this algorithm requires the computation of several constants which depend only on λ 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, $\lambda > 3.0$.

0. (Calculate constants) Compute:

$$m = \lambda - 1;$$

$$s^2 = \sqrt{\frac{8\lambda}{3}} + \lambda; \quad s = \sqrt{s^2};$$

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

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

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

1. (Select normal/exponential) Generate U . If $U \leq 0.0095722652$ 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 $m (\frac{x}{d} - \ln \frac{x}{m}) + c > E_2$ 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

Algorithm GA. Gamma variates, integral or half-integral shape parameter λ .

1. (Find K) Set $K = [\lambda]$, where $[\lambda]$ denotes the integral part of λ . Set $X = 0$. If $\lambda - K = 0.5$ set $L = 1$; if $\lambda - K = 0.0$ set $L = 0$; otherwise Stop. (If the algorithm stops, an incorrect λ value has been used.)
2. (Generate exponentials) If $K = 0$ go to Step 3, otherwise generate K exponentials $E_1, \dots, E_K$ and set
$$X = E_1 + \dots + E_K.$$
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

```

CAU000020
CAU000030
CAU000040
CAU000050
CAU000060
CAU000070
CAU000080
CAU000090
CAU000100
CAU000110
CAU000120
CAU000130
CAU000140
CAU000150
CAU000160
CAU000170
CAU000180
CAU000190
CAU000200
CAU000210
CAU000220
CAU000230
CAU000240
CAU000250
CAU000260
CAU000270
CAU000280
CAU000290
CAU000300
CAU000310
CAU000320
CAU000330
CAU000340
CAU000350

```

CAU00370
CAU00380
CAU00390
CAU00400
CAU00410
CAU00420
CAU00430
CAU00440
CAU00450
CAU00460
CAU00470
CAU00480
CAU00490
CAU00500
CAU00510
CAU00520
CAU00530
CAU00540
CAU00550
CAU00560
CAU00570
CAU00580
CAU00590
CAU00600
CAU00610
CAU00620
CAU00630
CAU00640
CAU00650
CAU00660
CAU00670
CAU00680
CAU00690
CAU00700
CAU00710
CAU00720
CAU00730
CAU00740
CAU00750
CAU00760
CAU00770
CAU00780
CAU00790
CAU00800
CAU00810

*****
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     NORMALIZATION 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
RAND
MR      R6,R8
SLDA   R6,1
SRL    R7,1
AR     R6,R7
BNO    #+10
A      R6,=F'2147483645'
LR     R6,R2
MEND   LR,R6
      PUT X(N) INTO R7
      ADD FOUR MORE
      GO ON IF NO OVERFLOW.
      ADD 2 ** 31 - 3
      SIMULATING DIVISION BY 2 ** 31 - 1
      ADD QUOTIENT TO REMAINDER, THUS
      R6 = REMAINDER; R7 = QUOTIENT
      GET NEXT UNIFORM

*****
EA
EA

```

**** CAUCHY DEVIATE GENERATOR ****

```

CAUCHY      CSECT      USING      CAUCHY,R15      DEFINE BASE REGISTER
B           DC          I2(,R13)      BRANCH AROUND ID
DC          DC          AL1(6)
DC          DC          CL6,CAUCHY,      MODULE NAME
ST          ST          R14,R12,I2(R13)  SAVE CALLING PROGRAM REGS
LR          LR          R13,SVAREA+4     CALLING SAVE ADDRESS IN OWN AREA
LA          LA          R2,R13          COPY CALLING SAVE ADDRESS TO R2
ST          ST          R13,SVAREA      OWN SAVE AREA IN R13
LM          LM          R3,R5,0(R1)      GET PARAMETER ADDRESSES
LR          LR          R7,0(R3)         SAVE SEED ADDRESS
L           L           R3,0(,R5)       GET SEED VALUE
SLA        SLA        R2,4             LOAD NUMBER OF DEVIATES TO GENERATE
LA         LA         R4,R2            CONVERT N TO BYTES
LR         LR         R5,R2            CONSTANT 4 FOR MAIN LOOP
LM         LM         R8,R12,LOOPCON    BACK UP 4 IN CALLER'S ARRAY
CNOP       CNOP       0,8             INITIAL ARRAY INDEX
                                LOAD MAIN LOOP CONSTANTS
                                ALIGN BXLE LOOP FOR SPEED

* * *
* * * MAINLOOP RAND
* * *
LR          LR          R0,R6
LR          LR          R1,R6
NR          NR          R1,R11
BZ         BZ         R1,TAIL
                                GET FIRST UNIFORM
                                SAVE TWO BITS OF X(N)
                                LAST BIT OF X(N) IN R0
                                NEXT TO LAST BIT IN R1
                                TEST BIT IN R1; IF 0, SAMPLE FROM TAIL

* * *
* * * RECT SAMPL
* * *
C           C           R6,=F'1367130551' SELECT RECTANGLE/WEDGE SAMPLING
BH         BH         WEDGE
                                GET NEXT UNIFORM
                                MAKE ROOM FOR EXPONENT
                                "OR" ON THE EXPONENT
                                STORE THE UNIFORM
                                TEST FOR NORMALIZATION
                                QUIT IF NOT NEEDED
                                NORMALIZE THE UNIFORM
                                GO TO END OF LOOP
BR         BR         R12

```

CAU000850
CAU000860
CAU000870
CAU000880
CAU000890
CAU000900
CAU000910
CAU000920
CAU000930
CAU000940
CAU000950
CAU000960
CAU000970
CAU000980
CAU000990
CAU01000
CAU01010
CAU01020
CAU01030
CAU01040
CAU01050
CAU01060
CAU01070
CAU01080
CAU01090
CAU01100
CAU01110
CAU01120
CAU01130
CAU01140
CAU01150
CAU01160
CAU01170
CAU01180
CAU01190
CAU01200
CAU01210
CAU01220
CAU01230
CAU01240
CAU01250
CAU01260
CAU01270
CAU01280
CAU01290

**** CAUCHY DEVIATE GENERATOR ****

```

WEDGE      RAND      R1,R6      SAVE FIRST UNIFORM
LR          RAND      R6,R1      GET UNIFORM IN R6 < UNIFORM IN R1
CR          CR        *+8        EXCHANGE REGISTERS
BNH         LR        R1,R7      EASY REJECTION TEST
LR          C         R1,R7      ACCEPT WEDGE SAMPLE
C           SRL       R6,7       CONVERT MINIMUM UNIFORM TO REAL
SRL         OR        R6,R9      "OR" ON THE EXPONENT
OR          ST        R6,UNIF    CONVERT MAXIMUM UNIFORM TO REAL
ST          OR        R1,R9      "OR" ON THE EXPONENT
OR          ST        R1,U2      LOAD TRIAL VARIATE
LE          CR        R6,R10     TEST FOR NORMALIZATION
CR          BC        11,*+8     NORMALIZE X
BC          AE        FR0,=E,0.0' GET FIRST COMPARAND FOR REJECTION TEST
AE          SER       FR2,U2     U2 - X
SER         LER       FR2,FR0    FIND X ** 2
LER         MER       FR4,FR0    - X ** 2 IN FR6
MER         LGER      FR4,FR4    1 - X ** 2
LGER        AE        FR6,=E,1.0' 1 + X ** 2
AE          AER       FR4,=E,1.0' FIND QUOTIENT
AER         MER       FR6,FR4    HARD REJECTION TEST
MER         CCR       FR6,=E,1.0' 1 - 82842712
CCR         BCR       FR2,FR6    GO BACK IF TEST FAILED
BCR         B        13,R12     WEDGE
B           B        WEDGE

```

CAU01300
CAU01310
CAU01320
CAU01330
CAU01340
CAU01350
CAU01360
CAU01370
CAU01380
CAU01390
CAU01400
CAU01410
CAU01420
CAU01430
CAU01440
CAU01450
CAU01460
CAU01470
CAU01480
CAU01490
CAU01500
CAU01510
CAU01520
CAU01530
CAU01540
CAU01550
CAU01560
CAU01570
CAU01580
CAU01590
CAU01600
CAU01610

**** CAUCHY DEVIATE GENERATOR ****

*
*TAIL

SRL R6,7
OR R6,R9
ST R6,UNIF
LE FR0,=E,1.0
DE FR0,UNIF
RAND
SRL
OR R6,7
ST R6,R9
R6,UNIF

MAKE ROOM FOR EXPONENT
"OR" ON THE EXPONENT
STORE THE UNIFORM
GET 1 / UNIFORM

GET ANOTHER UNIFORM FOR REJECTION TEST
MAKE ROOM FOR EXPONENT
"OR" ON THE EXPONENT

*

LER FR2,FR0
MER FR2,FR0
LA FR4,FR2
ME FR4,=E,1.0
CE FR4,UNIF
BCR FR4,FR2
RAND 13,R12
B TAIL

FIND X ** 2

GET 1 + X ** 2

FIND COMPARAND FOR REJECTION TEST
REJECTION TEST

ANOTHER UNIFORM FOR NEXT PASS
GO BACK

*

*
*ENDLOOP

NR R0,R11
BZ *+6
LCER FR0,FR0
STE FR0,0(R4,R5)
BXLE R5,R2,MAINLOOP

TEST SAVED BIT
IF BIT = 0, QUIT
IF BIT = 1, X = -X
STORE VARIATE IN CALLER'S ARRAY
BRANCH BACK FOR ANOTHER VARIATE

**

ST R7,0(R13)
LM R13,SVAREA+4
BR R14,R12,12(R13)
R14 RESTORE CALLING PROG REGS
RETURN

CAU01620
CAU01630
CAU01640
CAU01650
CAU01660
CAU01670
CAU01680
CAU01690
CAU01700
CAU01710
CAU01720
CAU01730
CAU01740
CAU01750
CAU01760
CAU01770
CAU01780
CAU01790
CAU01800
CAU01810
CAU01820
CAU01830
CAU01840
CAU01850
CAU01860
CAU01870
CAU01880
CAU01890
CAU01900
CAU01910
CAU01920
CAU01930
CAU01940

*** CAUCHY DEVIATE GENERATOR ***

SVAREA	DS	18F	SAVE AREA
UNIF	DS	F	TEMP STORAGE FOR UNIFORM
U2	DS	F	RANDOM VARIATES
LOOPCON	DC	F*16807,	MULTIPLIER FOR GENERATOR
	DC	X*400000001,	EXPONENT CONSTANT
	DC	X*401000000,	NORMALIZATION TEST CONSTANT
	DC	F*1,	MASK CONSTANT
	DC	AL4(ENDDLOOP)	END OF LOOP ADDRESS

LTORG

REGISTER EQUATES

R0	EQU	0
R1	EQU	1
R2	EQU	2
R3	EQU	3
R4	EQU	4
R5	EQU	5
R6	EQU	6
R7	EQU	7
R8	EQU	8
R9	EQU	9
R10	EQU	10
R11	EQU	11
R12	EQU	12
R13	EQU	13
R14	EQU	14
R15	EQU	15
FR0	EQU	0
FR2	EQU	2
FR4	EQU	4
FR6	EQU	6
	END	

CAU01960
CAU01970
CAU01980
CAU01990
CAU02000
CAU02010
CAU02020
CAU02030
CAU02040
CAU02050
CAU02060
CAU02070
CAU02080
CAU02090
CAU02100
CAU02110
CAU02120
CAU02130
CAU02140
CAU02150
CAU02160
CAU02170
CAU02180
CAU02190
CAU02200
CAU02210
CAU02220
CAU02230
CAU02240
CAU02250
CAU02260
CAU02270
CAU02280
CAU02290
CAU02300
CAU02310
CAU02320
CAU02330
CAU02340
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.

GMA 0020
GMA 0030
GMA 0040
GMA 0050
GMA 0060
GMA 0070
GMA 0080
GMA 0090
GMA 0100
GMA 0110
GMA 0120
GMA 0130
GMA 0140
GMA 0150
GMA 0160
GMA 0170
GMA 0180
GMA 0190
GMA 0200
GMA 0210
GMA 0220
GMA 0230
GMA 0240
GMA 0250
GMA 0260
GMA 0270
GMA 0280
GMA 0290
GMA 0300
GMA 0310
GMA 0320
GMA 0330
GMA 0340
GMA 0350
GMA 0360
GMA 0370
GMA 0380
GMA 0390
GMA 0400

**** GAMMA DEVIATE GENERATOR ****

GMA 0410
GMA 0420
GMA 0430
GMA 0440
GMA 0450
GMA 0460
GMA 0470
GMA 0480
GMA 0490
GMA 0500
GMA 0510
GMA 0520
GMA 0530
GMA 0540
GMA 0550
GMA 0560
GMA 0570
GMA 0580
GMA 0590
GMA 0600
GMA 0610
GMA 0620
GMA 0630
GMA 0640
GMA 0650
GMA 0660
GMA 0670
GMA 0680
GMA 0690
GMA 0700

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

FR2 HOLD'S GENERATED DEVIATE

NORMAL /
EXPOENTIAL
LOOP (GS.GQ.GE)

GMA	0720
GMA	0730
GMA	0740
GMA	0750
GMA	0760
GMA	0770
GMA	0780
GMA	0790
GMA	0800
GMA	0810
GMA	0820
GMA	0830
GMA	0840
GMA	0850
GMA	0860
GMA	0870
GMA	0880
GMA	0890
GMA	0900
GMA	0910
GMA	0920
GMA	0930
GMA	0940
GMA	0950
GMA	0960
GMA	0970
GMA	0980
GMA	0990
GMA	1000
GMA	1010
GMA	1020
GMA	1030

*** GAMMA DEVIATE GENERATOR ***

* R0
* R1
* R2
* R3
* R4
* R5
* R6
* R7
* R8
* R9
* R10
* R11
* R12
* R13
* R14
* R15
* FR0
* FR2
* FR4
* FR6

REGISTER EQUATES:

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
2
4
6

1040
1050
1060
1070
1080
1090
1100
1110
1120
1130
1140
1150
1160
1170
1180
1190
1200
1210
1220
1230
1240
1250
1260

GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA
GMA

*** GAMMA DEVIATE GENERATOR ***

*

*

*

GAMA

LINKAGE / INITIALIZATION SECTION

```

CSECT GAMA,R15
USING IOI,R15)
BC ALI(4)
DC CL4,GAMA,12(R13)
ST R14,R12,SVAREA+4
LR R13,SVAREA
LA R2,R13
ST R13,SVAREA
ST R13,8(R2)

                DEFINE BASE REGISTER
                BRANCH AROUND IO
                MODULE IDENTIFIER
                SAVE CALLING REGS IN OWN AREA
                CALLING SAVE AREA ADDRESS TO R2
                COPY CALLING AREA ADDRESS TO R2
                OWN SAVE AREA IN R13
                FORWARD LINK

                GET PARAMETER ADDRESSES
                GET SHAPE PARAMETER
                TEST FOR NEW "A" VALUE
                IF SO, DO PRELIMINARY CALCULATIONS
                CONSTANT 4 FOR MAIN LOOP
                PUT SEED INTO R7 DEVIATES, N
                GET NUMBER OF BYTES
                CONVERT ONE IN CALLER'S ARRAY
                BACKUP ONE IN LOOP INDEX
                INITIAL MAIN LOOP INDEX
                JUMP TO PROPER METHOD

R2,R5,0(R1)
FR0,0(R2)
FR0,AP
SETUP
R2,4
R7,0(R3)
R3,0(R5)
R4,R2
R5,R2
R6,METHOD
R6

```

*

*

GMAN

GMA 1280
GMA 1290
GMA 1300
GMA 1310
GMA 1320
GMA 1330
GMA 1340
GMA 1350
GMA 1360
GMA 1370
GMA 1380
GMA 1390
GMA 1400
GMA 1410
GMA 1420
GMA 1430
GMA 1440
GMA 1450
GMA 1460
GMA 1470
GMA 1480
GMA 1490
GMA 1500
GMA 1510
GMA 1520
GMA 1530
GMA 1540

GMA	1560
GMA	1570
GMA	1580
GMA	1590
GMA	1600
GMA	1610
GMA	1620
GMA	1630
GMA	1640
GMA	1650
GMA	1660
GMA	1670
GMA	1680
GMA	1690
GMA	1700
GMA	1710
GMA	1720
GMA	1730
GMA	1740
GMA	1750
GMA	1760
GMA	1770
GMA	1780
GMA	1790
GMA	1800
GMA	1810
GMA	1820
GMA	1830
GMA	1840
GMA	1850
GMA	1860
GMA	1870
GMA	1880
GMA	1890
GMA	1900
GMA	1910
GMA	1920
GMA	1930
GMA	1940
GMA	1950
GMA	1960
GMA	1970
GMA	1980
GMA	1990
GMA	2000

```

***
$GO
SET UP FOR LARGE PARAMETER METHOD, ALGORITHM "GO"
LA RO,GO
ST RO,METHOD
LA RO,40
RO,INX1
FRO,AGO
GWAN
FRO,AGO
FRO,=E,1.0,
FRO,FR2
FRO,MU
FRO,FR0
FRO,MUP
STE
DERE
STE

```

```

***
LINK TO SQR T FUNCTION FOR SQR T(A)

LA      R1,ARGLS T1      LOAD ARGUMENT LIST
LR      R8,R15           SAVE BASE REGISTER
L        R15,VADDSR      ADDRESS OF SQR T FUNCTION
BALR    R14,R15
LR      R15,R8           RESTORE BASE REGISTER
LR      FR2,FR0          SAVE SQR T(A)
ME      FR0,=E'1.6329932' FIND NORMAL VARIANCE
AE      FR0,AGO
STE     FR0,SIGMA

```

**** GAMMA DEVIATE GENERATOR ****

```

DE      FRO,MU      FIND REJECTION CONSTANT "WM"
SE      FRO,=E,1.0'
STE     FRO,WM
AE      FR2,=E,1.6329932'  FIND REJECTION CONSTANT "VP"
DE      FR2,MU
ME      FR2,=E,2.0'
STE     FR2,VP

LINK TO SORT FUNCTION TO FIND NORMAL STD DEV
* * *
LA      R1,ARGLST2      LOAD ARGUMENT LIST ADDRESS
BALR    R15,VADDSR      ADDRESS OF SORT FUNCTION
LR      R14,R15
STE     R15,R8          RESTORE BASE REGISTER
FRO,SIGMA      SAVE STD DEV
*
ME      FRO,=E,2.4494897'  FIND REJECTION CONSTANT "DP"
LE      FR2,=E,1.0'
DER     FR2,FRO
STE     FR2,DP
FRO,D
*
AE      FRO,MU
STE     FRO,B
LE      FR2,=E,1.0'
DER     FR2,FRO
STE     FR2,BP
FIND UPPER LIMIT FOR NORMAL METHOD, "B"
*
*      COMPUTE BP = 1 / B
*
LE      FR2,SIGMA
ME      FR2,D
DER     FR2,FRO
STE     FR2,CONS
LA      R1,ARGLST3
BALR    R15,VADDLG
LR      R14,R15
FRO,R8
*
LCER    FRO,FRO
SE      FRO,B
AE      FRO,MU
AE      FRO,MU
AE      FRO,=E,3.7203285'
STE     FRO,CONS
B      GMAN
*
*      DONE WITH INITIALIZATION. PROCEED TO
*      GENERATION

```

GMA 2010
GMA 2020
GMA 2030
GMA 2040
GMA 2050
GMA 2060
GMA 2070
GMA 2080
GMA 2090
GMA 2100
GMA 2110
GMA 2120
GMA 2130
GMA 2140
GMA 2150
GMA 2160
GMA 2170
GMA 2180
GMA 2190
GMA 2200
GMA 2210
GMA 2220
GMA 2230
GMA 2240
GMA 2250
GMA 2260
GMA 2270
GMA 2280
GMA 2290
GMA 2300
GMA 2310
GMA 2320
GMA 2330
GMA 2340
GMA 2350
GMA 2360
GMA 2370
GMA 2380
GMA 2390
GMA 2400
GMA 2410
GMA 2420
GMA 2430
GMA 2440
GMA 2450
GMA 2460

**** GAMMA DEVIATE GENERATOR ****

*
*
* SGF

```

SET UP FOR FISHMAN'S METHOD, ALGORITHM "GF"
LA RO,GF
ST RO,METHOD
SE FR0,=E,1.0,
STE FR0,AMINUS
LA RO,20
ST RO,INX2
B GWAN

```

COMPUTE AMINUS = A - 1
INITIALIZE RANDOM ARRAY INDEX
DONE WITH INITIALIZATION. PROCEED TO
GENERATION.

*
*
*
*
* SGF

```

SET UP FOR SMALL PARAMETER METHOD. "GS"
LA RO,GS
ST RO,METHOD
LER FR2,FR0
LE FR4,=E,1.0,
LCER FR2,FR2
STE FR2,AMIN1
DER FR4,FR0
ME FR4,AINV
AE FR0,=E,1.0,
STE FR0,BGS
LA RO,40
ST RO,INX3
B GWAN

```

SET ADDRESS FOR SUBSEQUENT CALLS
COMPUTE 1 - A
COMPUTE 1 / A
FIND (E + A) / E
INITIALIZE EXPONENTIAL ARRAY INDEX
DONE WITH INITIALIZATION. GO ON
TO GENERATION.

*

GMA 2480
GMA 2490
GMA 2500
GMA 2510
GMA 2520
GMA 2530
GMA 2540
GMA 2550
GMA 2560
GMA 2570
GMA 2580
GMA 2590
GMA 2600
GMA 2610
GMA 2620
GMA 2630
GMA 2640
GMA 2650
GMA 2660
GMA 2670
GMA 2680
GMA 2690
GMA 2700
GMA 2710
GMA 2720
GMA 2730
GMA 2740
GMA 2750
GMA 2760
GMA 2770
GMA 2780

```

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

```

GMA	2800
GMA	2810
GMA	2820
GMA	2830
GMA	2840
GMA	2850
GMA	2860
GMA	2870
GMA	2880
GMA	2890
GMA	2900
GMA	2910
GMA	2920
GMA	2930
GMA	2940
GMA	2950
GMA	2960
GMA	2970
GMA	2980
GMA	2990
GMA	3000
GMA	3010
GMA	3020
GMA	3030
GMA	3040
GMA	3050
GMA	3060
GMA	3070
GMA	3080
GMA	3090
GMA	3100
GMA	3110
GMA	3120
GMA	3130
GMA	3140
GMA	3150
GMA	3160
GMA	3170
GMA	3180
GMA	3190

```

** *
** *
** *
METHOD "GO" (DIETER-AHRENS)
LM      R8,R13,GOCON    LOAD LOOPING CONSTANTS
CNOP    0,8             ALIGN BXLE LOOP FOR SPEED

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
R6:=F'2147483645'      FIXUP OVERFLOW.   ADD 2 ** 31 - 3
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
BXLE    R12,R10,GONTST INCREMENT NORMAL ARRAY INDEX.
                                NORMAL ARRAY EXHAUSTED. REPLENISH IT.
ST      R7,IX           SAVE CURRENT SEED VALUE.
LA      R12,K15          SAVE BASE REGISTER
LA      R13,SVAREA       SAVE AREA POINTER
LA      R15,VADDNM       ARGUMENT OF "NORMAL"
BALR    R14,R15          ADDRESS OF "NORMAL" LINK
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      FRO,RNARRAY(R12)  LOAD NEXT NORMAL DEVIATE
LER     FR2,FRO           TRIAL GAMMA VALUE:
ME      FR2,SIGMA         X = NORMAL * SIGMA + MU
AE      FR2,MU            REJECT X < 0
GNORM   GONORM            REJECT X > B
CE      FR2,B             S2 = 0.5 * S * S
BH      GONORM

LER     FR4,FKO           FR4,FR0
MER     FR4,FR0           FR4,FR4
HER     FR4,FR4

```

*** GAMMA DEVIATE GENERATOR ***

```

*
GET A UNIFORM FOR NORMAL REJECTION TEST
MR,R8      GET NEXT UNIFORM
SLDA       R6,R8
SRL        R6,1
AR         R7,1
BNO        R6,R7
A          R6,R7
AR         R6,R7
LR         R6,R7
SRL        R6,7
OR         R6,R9
ST         R6,R9
LTER       R6,R9
BP         R6,R9
GO ON IF NO OVERFLOW. ADD 2 ** 31 - 1
R6,=F'2147483645.
ADD 4 MORE
PUT X(N) INTO R7
MAKE ROOM FOR EXPONENT.
"OR" ON THE EXPONENT
SAVE THE UNIFORM.
PERFORM THE PROPER REJECTION, DEPENDING
ON THE SIGN OF THE NORMAL

* GONEG
ME         FR0,VP
SE         FR0,WM
MER        FR0,FR4
AE         FR0,=E,1.0.
CE         FR0,UNIF
BCR        2,R13
B          GON2TST

* GOPOS
LCER       FR0,FR4
ME         FR0,WM
AE         FR0,=E,1.0.
CE         FR0,UNIF
BCR        2,R13

* GON2TST
SER        FR4,FR2
AE         FR4,MU
STE        FR4,SUM
ME         FR2,X
STE        FR2,MUP
STE        FR2,LOG

LINK TO LOG SUBROUTINE TWICE

STM        R12,R13,GOSAVE
LR         R12,R15
LA         R13,SVAREA
LA         R1,ARGLST5
L          R15,VADDLG
BALR       R14,R15
LR         R15,R12

RESTORE BASE REGISTER

COMPUTE THE REJECTION VALUE:
1 + S2 * (S * VP - WM)

REJECTION TEST
GO TO LOOP END IF PASSED.
FURTHER TEST IF NOT.

COMPUTE THE REJECTION VALUE:
1 - S2 * WM

REJECTION TEST
GO TO LOOP END IF PASSED.

FIND PARTIAL SUM FOR REJECTION TEST:
SUM = MU - X + S2

SAVE TRIAL GAMMA DEVIATE
GET LOG ARGUMENT, X / MU

GMA 3660
GMA 3670
GMA 3680
GMA 3690
GMA 3700
GMA 3710
GMA 3720
GMA 3730
GMA 3740
GMA 3750
GMA 3760
GMA 3770
GMA 3780
GMA 3790
GMA 3800
GMA 3810
GMA 3820
GMA 3830
GMA 3840
GMA 3850
GMA 3860
GMA 3870
GMA 3880
GMA 3890
GMA 3900
GMA 3910
GMA 3920
GMA 3930
GMA 3940
GMA 3950
GMA 3960
GMA 3970
GMA 3980
GMA 3990
GMA 4000
GMA 4010
GMA 4020
GMA 4030
GMA 4040
GMA 4050
GMA 4060
GMA 4070
GMA 4080
GMA 4090
GMA 4100

```

**** 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
*      LM      R12,R13,GOSAVE      RESTORE BASE REGISTER
*
*      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 OF 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
*

```

GMA 4110
GMA 4120
GMA 4130
GMA 4140
GMA 4150
GMA 4160
GMA 4170
GMA 4180
GMA 4190
GMA 4200
GMA 4210
GMA 4220
GMA 4230
GMA 4240
GMA 4250
GMA 4260
GMA 4270
GMA 4280
GMA 4290
GMA 4300
GMA 4310
GMA 4320
GMA 4330
GMA 4340
GMA 4350
GMA 4360
GMA 4370
GMA 4380
GMA 4390
GMA 4400
GMA 4410
GMA 4420
GMA 4430
GMA 4440
GMA 4450
GMA 4460
GMA 4470
GMA 4480
GMA 4490
GMA 4500
GMA 4510
GMA 4520
GMA 4530

**** GAMMA DEVIATE GENERATOR ****

```

LE      FR2,X
ME      FR4,FR2
SER     FR4,BP
ME      FR0,FR4
AE      FR0,MU
LCER    FR0,CONS
CE      FR0,FR0
BH      FR0,RNEXP+4
        GOLOOP

        RELOAD TRIAL GAMMA VALUE
        COMPLETE CALCULATION OF REJECTION VALUE.
        MU * (LOG - X * BP) + CONS

        PERFORM REJECTION TEST
        BACK TO START IF FAILED.

        END OF METHOD "GO" LOOP.
        GENERATED DEVIATE IS IN FR2.

STE      FR2,0(R4,R5)
BXLE     R5,R2,GOLOOP
ST       R12,INX1
B        THRU

        STORE DEVIATE IN CALLER'S ARRAY.
        BRANCH BACK FOR ANOTHER DEVIATE.
        SAVE LAST ARRAY INDEX
        ALL DONE. QUIT.

GMA 4540
GMA 4550
GMA 4560
GMA 4570
GMA 4580
GMA 4590
GMA 4600
GMA 4610
GMA 4620
GMA 4630
GMA 4640
GMA 4650
GMA 4660
GMA 4670
GMA 4680
GMA 4690
GMA 4700

```

ENDGO

**** GAMMA DEVIATE GENERATOR ****

```

*
*
*
*
* GF
FISHMAN'S METHOD
ST      R7,IX      SET UP SEED
LM      R8,R12,GFCN LOAD LOOP CONSTANTS
LR      R7,R15     SHIFT BASE REGISTER
DROP    R15
USING   GAMA,R7
LR      R15,R9
CNOP    0,8
*
* GFLOOP
*
BXLE    R12,R10,GFTST
LA      R1,ARGLST4
LR      R15,R8
BALR    R14,R15
LR      R15,R9
SR      R12,R12
CNOP    0,8
*
* 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 "ALUG"
LE      R12,RNARRAY(R12) FINISH COMPUTING REJECTION VALUE:
SER      R4,FR2      (A - 1) * (R - LN R - 1)
SE      R4,FR0
ME      R4,=E,1.0'
CE      R4,AMINUS
BH      R4,RNARRAY+20(R12) REJECTION TEST
*
ME      R2,AP
ST      R2,0(R4,R5)
BXLE    R5,R2,GFLOOP
LR      R15,R7
DROP    R7
USING   GAMA,R15
L        R7,IX
ST      R12,INX2
B        THRU

```

GMA 4720
 GMA 4730
 GMA 4740
 GMA 4750
 GMA 4760
 GMA 4770
 GMA 4780
 GMA 4790
 GMA 4800
 GMA 4810
 GMA 4820
 GMA 4830
 GMA 4840
 GMA 4850
 GMA 4860
 GMA 4870
 GMA 4880
 GMA 4890
 GMA 4900
 GMA 4910
 GMA 4920
 GMA 4930
 GMA 4940
 GMA 4950
 GMA 4960
 GMA 4970
 GMA 4980
 GMA 4990
 GMA 5000
 GMA 5010
 GMA 5020
 GMA 5030
 GMA 5040
 GMA 5050
 GMA 5060
 GMA 5070
 GMA 5080
 GMA 5090
 GMA 5100
 GMA 5110
 GMA 5120
 GMA 5130

5150 GMA
5160 GMA
5170 GMA
5180 GMA
5190 GMA
5200 GMA
5210 GMA
5220 GMA
5230 GMA
5240 GMA
5250 GMA
5260 GMA
5270 GMA
5280 GMA
5290 GMA
5300 GMA
5310 GMA
5320 GMA
5330 GMA
5340 GMA
5350 GMA
5360 GMA
5370 GMA
5380 GMA
5390 GMA
5400 GMA
5410 GMA
5420 GMA
5430 GMA
5440 GMA
5450 GMA
5460 GMA
5470 GMA
5480 GMA
5490 GMA

42

**** GAMMA DEVIATE GENERATOR ****

```

*      CHI - SQUARED, 3 DEGREES OF FREEDOM ( A = 1.5 )
*      CHISQ3
*      LR      R6,R15      SHIFT BASE REGISTER
*      DROP    R15
*      USING   GAMA,R6
*      LA      R14,(R1)
*      BALR    R15,VADDEX
*      BALR    R14,R15
*      L       R7,0(R1)
*      L       R7,0(R7)
*      ST      R7,IX
*      LM      R10,R12,CHICON3
*      CNOP    0,8
*      CHLOOP3 BXLE      R12,R10,CH3COMP
*      LA      R15,VADDNM
*      BALR    R14,ARG1ST4
*      SR      R12,R15
*      CH3COMP LE      R0,RNARRAY(R12)
*      MER     R0,FRO
*      AE      R0,FRO
*      STE     R0,0(R4,R5)
*      BXLE   R5,R2,CHLOOP3
*
*      LR      R7,IX
*      ST      R12,INX4
*      LR      R15,R6
*      B       THRU

```

GMA 5500
 GMA 5510
 GMA 5520
 GMA 5530
 GMA 5540
 GMA 5550
 GMA 5560
 GMA 5570
 GMA 5580
 GMA 5590
 GMA 5600
 GMA 5610
 GMA 5620
 GMA 5630
 GMA 5640
 GMA 5650
 GMA 5660
 GMA 5670
 GMA 5680
 GMA 5690
 GMA 5700
 GMA 5710
 GMA 5720
 GMA 5730
 GMA 5740
 GMA 5750
 GMA 5760
 GMA 5770
 GMA 5780
 GMA 5790
 GMA 5800
 GMA 5810

**** GAMMA DEVIATE GENERATOR ****

```

*
*
* CHISQ4
  2 - ERLANG ( A = 2.0 )
  LR R6,R15      SHIFT BASE REGISTER
  LA R14(,R1)    SKIP OVER SHAPE PARAMETER IN ARG LIST
  L  R15,VADDEX  LINK TO "EXPON"
  BALR R14,R15
  L  R7,O(,R1)   GET LAST SEED VALUE USED
  L  R7,O(,R7)
  ST R7,IX
  LM R10,R12,CHICON3  SAVE SEED VALUE
  CNOP 0,8        LOAD LOOP CONSTANTS
                   ALIGN BXLE LOOP FOR SPEED
* CHLOOP4
  BXLE R12,R10,CH4COMP  GET NEXT EXPONENTIAL
*
  L  R15,VADDEX  EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
  LA R1,ARGLST4  LINK TO "EXPON"
  BALR R14,R15   GET ARGUMENT LIST
  SR  R12,R12    LINK TO "EXPON"
                   RESET ARRAY INDEX TO ZERO
* CH4COMP
  LE  FRO,RNARRAY(R12)  LOAD NEW EXPONENTIAL
  AE  FRO,O(R4,R5)      ADD TO SECOND EXPONENTIAL
  STE FRO,O(R4,R5)      STORE GENERATED GAMMA IN CALLER'S ARRAY
  BXLE R5,R2,CHLOOP4   GO BACK FOR NEXT DEVIATE
*
  L  R7,IX
  ST R12,INX4
  LR R15,R6
  B  THRU
                   LOAD LAST SEED VALUE
                   SAVE RANDOM ARRAY INDEX
                   RESTORE BASE REGISTER
                   QUIT

```

GMA 5820
 GMA 5830
 GMA 5840
 GMA 5850
 GMA 5860
 GMA 5870
 GMA 5880
 GMA 5890
 GMA 5900
 GMA 5910
 GMA 5920
 GMA 5930
 GMA 5940
 GMA 5950
 GMA 5960
 GMA 5970
 GMA 5980
 GMA 5990
 GMA 6000
 GMA 6010
 GMA 6020
 GMA 6030
 GMA 6040
 GMA 6050
 GMA 6060
 GMA 6070
 GMA 6080
 GMA 6090
 GMA 6100

**** GAMMA DEVIATE GENERATOR ****

```

*
*
* CHISQ6
  3 - ERLANG ( A = 3.0 )
  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
  LM R10,R12,CHICON6  SAVE SEED VALUE
  CNOP 0,8        LOAD LOOP CONSTANTS
                  ALIGN BXLE LOOP FOR SPEED
* CHLOOP6
  BXLE R12,R10,CH6COMP  GET NEXT PAIR OF EXPONENTIALS
*
  LA R15,VADDEX  EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
  BALR R1,ARGLIST4  LINK TO "EXPON"
  SR R14,R15      GET ARGUMENT LIST
  R12,R12        LINK TO "EXPON"
                  RESET ARRAY INDEX
* CH6COMP
  LE FRO,RNARRAY(R12)  LOAD NEW EXPONENTIAL
  AE FRO,RNARRAY+20(R12)  ADD TWO INDEPENDENT EXPONENTIALS
  AE FRO,0(R4,R5)
  STE FRO,0(R4,R5)
  BXLE R5,R2,CHLOOP6  SAVE GENERATED GAMMA IN CALLER'S ARRAY
                        GO BACK FOR NEXT DEVIATE
*
  L  R7,IX
  ST R12,INX5
  LR R15,R6
  DROP R6
  USING GAMA,R15
  B THRU
  QUIT

```

GMA 6110
 GMA 6120
 GMA 6130
 GMA 6140
 GMA 6150
 GMA 6160
 GMA 6170
 GMA 6180
 GMA 6190
 GMA 6200
 GMA 6210
 GMA 6220
 GMA 6230
 GMA 6240
 GMA 6250
 GMA 6260
 GMA 6270
 GMA 6280
 GMA 6290
 GMA 6300
 GMA 6310
 GMA 6320
 GMA 6330
 GMA 6340
 GMA 6350
 GMA 6360
 GMA 6370
 GMA 6380
 GMA 6390
 GMA 6400
 GMA 6410
 GMA 6420

**** GAMMA DEVIATE GENERATOR ****

```

* * * * *
* GS
* GSLOOP
  SMALL PARAMETER METHOD "GS" (AHRENS)
  LM R8,R12,GSCON LOAD LOOP CONSTANTS
  CNOP 0,8 ALIGN BXLE LOOP FOR SPEED

  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
  BND *+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 FRO,UNF
  ME FRO,BGS
  STE FRO,P FIND P = 8 * UNIFORM

*
  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
  ST R7,IX EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
  LA R1,ARGLST4 SAVE SEED VALUE
  L R15,VADDEX LOAD ARGUMENT LIST ADDRESS
  BALR R14,R15 LINK TO "EXPON"
  SR R12,R12 RESET ARRAY INDEX TO START
  LE FRO,P RELOAD P INTO FRO
  L R7,IX RESTORE SEED TO R7
  CNOP 0,8 ALIGN BXLE FOR SPEED

* GSTST
  CE FRO,=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

```

GMA 6440
 GMA 6450
 GMA 6460
 GMA 6470
 GMA 6480
 GMA 6490
 GMA 6500
 GMA 6510
 GMA 6520
 GMA 6530
 GMA 6540
 GMA 6550
 GMA 6560
 GMA 6570
 GMA 6580
 GMA 6590
 GMA 6600
 GMA 6610
 GMA 6620
 GMA 6630
 GMA 6640
 GMA 6650
 GMA 6660
 GMA 6670
 GMA 6680
 GMA 6690
 GMA 6700
 GMA 6710
 GMA 6720
 GMA 6730
 GMA 6740
 GMA 6750
 GMA 6760
 GMA 6770
 GMA 6780
 GMA 6790
 GMA 6800
 GMA 6810
 GMA 6820
 GMA 6830
 GMA 6840
 GMA 6850
 GMA 6860
 GMA 6870
 GMA 6880

**** GAMMA DEVIATE GENERATOR ****

```

ME STE      FR0,AINV      GET LOG (P) / A
LR   R15,R8    LINK TO EXPONENTIAL FUNCTION.
LA   R1,ARGLST9  LOAD ARGUMENT LIST ADDRESS
BALR R14,R15    RESULT IS P*(1 / A)
CE   R14,RNARRAY(R12) REJECTION TEST
BNH  ENDS      QUIT IF OK,
LM   R8,R9,GSCON OTHERWISE GO BACK
LR   R15,R6     RESET BASE REGISTER
B.   GSLOOP

*XBIG
LE   FR2,BGS
SER  FR2,FR0
ME   FR2,AINV
STE  FR2,P
LA   R1,ARGLST9
LR   R15,R9
BALR R14,R15
LCER FR0,FR0
STE  FR0,P,ARGLST9
LA   R15,R9
LR   R14,R15
BALR R14,AMIN1
ME   FR0,RNARRAY(R12) FINISH CALCULATION OF REJECTION VALUE
CE   FR0,P      REJECTION TEST
LE   ENDS      RELOAD TRIAL GAMMA VALUE
BNH  R8,R9,GSCON QUIT IF OK
LM   R15,R6     OTHERWISE RESET LOOP CONSTANTS
LR   R15,R6     AND CHANGE BASE REGISTER
B.   GSLOOP     AND GO BACK

END OF GSLOOP

GAMMA VARIATE VALUE IS IN FR0
STE  FR0,0(R4,R5) STORE DEVIATE IN CALLER'S ARRAY
LM   R8,R9,GSCON RESET LOOP CONSTANTS
LR   R15,R6     RESET BASE REGISTER
BXLE R5,R2,GSLOOP SHIFT BASE REG FOR ANOTHER DEVIATE
ST   R12,INX3   SAVE LAST ARRAY INDEX
B    THRU       OTHERWISE QUIT.
DROP R6
USING GAMA,R15

```

GMA 6890
 GMA 6900
 GMA 6910
 GMA 6920
 GMA 6930
 GMA 6940
 GMA 6950
 GMA 6960
 GMA 6970
 GMA 6980
 GMA 6990
 GMA 7000
 GMA 7010
 GMA 7020
 GMA 7030
 GMA 7040
 GMA 7050
 GMA 7060
 GMA 7070
 GMA 7080
 GMA 7090
 GMA 7100
 GMA 7110
 GMA 7120
 GMA 7130
 GMA 7140
 GMA 7150
 GMA 7160
 GMA 7170
 GMA 7180
 GMA 7190
 GMA 7200
 GMA 7210
 GMA 7220
 GMA 7230
 GMA 7240
 GMA 7250
 GMA 7260
 GMA 7270
 GMA 7280
 GMA 7290
 GMA 7300

**** GAMMA DEVIATE GENERATOR ****

```

* * *
* * * THRU
* * * END OF ROUTINE.
* * * 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 OD
* * * DS
* * * DATA AREA
* * * SVAREA DS 18F SAVE AREA
* * *
* * * AP E'-1.0' OLD SHAPE PARAMETER
* * * METHOD DS F ADDRESS FOR PROPER METHOD
* * * VADDEX V(EXPON) EXTERNAL EXPONENTIAL GENERATOR
* * * VADNOR V(NORMAL) EXTERNAL NORMAL GENERATOR
* * * VADLOG V(ALOG) LOGARITHM FUNCTION
* * * VADDSR V(SQRT) SQUARE ROOT FUNCTION
* * *
* * * IX RANDOM NUMBER SEED
* * * RNARRAY DS F ARRAY FOR NORMAL OR EXPONENTIAL DEVIATES
* * * NUM DS F.10' NUMBER OF DEVIATES TO BE DELIVERED
* * *
* * * CONSTANTS FOR METHOD "GO"
* * *
* * * DC E'5.0' SHAPE PARAMETER
* * * DC E'4.0' NORMAL MEAN
* * * DC E'2.9413405' NORMAL STD DEV
* * * DC E'11.204783' UPPER LIMIT FOR NORMAL
* * * DC E'0.25' I / MU
* * * DC E'.089247598' I / B
* * * DC E'.13879668' MISC
* * * DC E'.1628709' CONSTANTS
* * * DC E'.19345306' FOR
* * * DC E'-.12172460' "GO"

```

GMA 7320
 GMA 7330
 GMA 7340
 GMA 7350
 GMA 7360
 GMA 7370
 GMA 7380
 GMA 7390
 GMA 7400
 GMA 7410
 GMA 7420
 GMA 7430
 GMA 7440
 GMA 7450
 GMA 7460
 GMA 7470
 GMA 7480
 GMA 7490
 GMA 7500
 GMA 7510
 GMA 7520
 GMA 7530
 GMA 7540
 GMA 7550
 GMA 7560
 GMA 7570
 GMA 7580
 GMA 7590
 GMA 7600
 GMA 7610
 GMA 7620
 GMA 7630
 GMA 7640
 GMA 7650
 GMA 7660
 GMA 7670
 GMA 7680
 GMA 7690
 GMA 7700
 GMA 7710
 GMA 7720

*** GAMMA DEViate GENERATOR ***

* GOCON	DC	F'16807'	UNIFORM MULTIPLIER
	DC	X'400000001'	EXPONENT CONSTANT
	DC	F'4'	NORMAL ARRAY INDEX INCREMENT
INX1	DC	F'36'	INDEX LIMIT
	DC	F'40'	ARRAY INDEX
	DC	AL4(ENDGO)	END OF "GO" LOOP
* D	DS	F	TEMP STORAGE
SUM	DS	F	FOR
LOG	DS	F	INTERMEDIATE
UNIF	DS	F	RESULTS
X	DS	F	TRIAL GAMMA DEViate
GOSAVE	DS	2F	REGISTER STORAGE
RNEXP	DS	2F	ARRAY FOR EXPONENTIAL SAMPLING
NGO1	DC	F'2'	NUMBER OF EXPONENTIALS
* *			
* *			
AMINUS			
GFCON	DS	F	CONSTANTS FOR METHOD "GF"
	DC	V(EXPON)	A - 1
	DC	V(ALOG)	ADDRESS OF EXPONENTIAL GENERATOR
	DC	F'4'	ADDRESS OF LOG FUNCTION
	DC	F'10'	EXPONENTIAL ARRAY INDEX INCREMENT
	DC	F'40'	EXPONENTIAL ARRAY INDEX LIMIT
INX2	DC	F	EXPONENTIAL ARRAY INDEX
GFLG	DS	F	TEMP STORAGE
* *			
* *			
AINV	DS	F	CONSTANTS FOR METHOD "GS"
AMIN1	DS	F	1 / A
BGS	DS	F	1 / A
GSCON	DC	F'16807'	(E + A) / E
	DC	X'400000001'	UNIFORM MULTIPLIER
	DC	F'4'	EXPONENT CONSTANT
	DC	F'36'	EXPONENTIAL ARRAY INDEX INCREMENT
	DC	F'40'	EXPONENTIAL ARRAY INDEX
INX3	DC	V(EXP)	EXPONENTIAL FUNCTION
GSVCON	DC	V(ALOG)	EXTERNAL ADDRESSES
UNF	DC	F	TEMPORARY STORAGE
P	DS	F	LOCATIONS

GMA 7730
GMA 7740
GMA 7750
GMA 7760
GMA 7770
GMA 7780
GMA 7790
GMA 7800
GMA 7810
GMA 7820
GMA 7830
GMA 7840
GMA 7850
GMA 7860
GMA 7870
GMA 7880
GMA 7890
GMA 7900
GMA 7910
GMA 7920
GMA 7930
GMA 7940
GMA 7950
GMA 7960
GMA 7970
GMA 7980
GMA 7990
GMA 8000
GMA 8010
GMA 8020
GMA 8030
GMA 8040
GMA 8050
GMA 8060
GMA 8070
GMA 8080
GMA 8090
GMA 8100
GMA 8110
GMA 8120
GMA 8130

* * *		CONSTANTS FOR AD HOC METHODS				GMA 8150
* CHICON3	F'4'	NORMAL	ARRAY INDEX INCREMENT			GMA 8160
INX4	F'36'	NORMAL	ARRAY INDEX LIMIT			GMA 8170
* INX4	F'40'	NORMAL	ARRAY INDEX			GMA 8180
* CHICON6	F'4'	ARRAY INDEX INCREMENT				GMA 8190
INX5	F'16'	ARRAY INDEX LIMIT				GMA 8200
* INX5	F'40'	ARRAY INDEX				GMA 8210
* * *						GMA 8220
* ARGST1	X'FF'	CALL TO SQRT IN "GO" SET UP				GMA 8230
ARGST2	AL3(AGO)	2ND CALL TO SQRT IN "GO" SET UP				GMA 8240
ARGST3	X'FF'	CALL TO ALOG IN "GO" SETUP				GMA 8250
ARGST4	AL3(CONS)	CALLS TO REPLENISH RNARRAY				GMA 8260
	AL4(IX)					GMA 8270
	AL4(RNARRAY)					GMA 8280
	X'FF'					GMA 8290
ARGST5	AL3(NUM)	CALL TO ALOG IN NORMAL SECTION OF "GO"				GMA 8300
ARGST6	AL3(LOG)	CALL TO ALOG IN EXPON SECTION OF "GO"				GMA 8310
ARGST7	X'FF'	CALL TO EXPONENTIAL GENERATOR IN "GO"				GMA 8320
	AL3(UNIF)					GMA 8330
	AL4(IX)					GMA 8340
	AL4(RNEXP)					GMA 8350
	X'FF'					GMA 8360
	AL3(NGO1)					GMA 8370
ARGST8	X'FF'	CALL TG ALOG IN METHOD "GF"				GMA 8380
ARGST9	AL3(GFLOG)	FUNCTION CALLS IN METHOD "GS"				GMA 8390
	X'FF'					GMA 8400
	AL3(P)					GMA 8410
						GMA 8420
						GMA 8430
						GMA 8440
						GMA 8450
						GMA 8460
						GMA 8470
						GMA 8480
						GMA 8490
						GMA 8500
						GMA 8510
						GMA 8520
						GMA 8530
						GMA 8540
						GMA 8550
						GMA 8560
						GMA 8570
						GMA 8580
						GMA 8590
						GMA 8600
						GMA 8610
						GMA 8620
						GMA 8630
						GMA 8640
						GMA 8650
						GMA 8660
						GMA 8670
						GMA 8680
						GMA 8690
						GMA 8700
						GMA 8710
						GMA 8720
						GMA 8730
						GMA 8740
						GMA 8750
						GMA 8760
						GMA 8770
						GMA 8780
						GMA 8790
						GMA 8800
						GMA 8810
						GMA 8820
						GMA 8830
						GMA 8840
						GMA 8850
						GMA 8860
						GMA 8870
						GMA 8880
						GMA 8890
						GMA 8900
						GMA 8910
						GMA 8920
						GMA 8930
						GMA 8940
						GMA 8950
						GMA 8960
						GMA 8970
						GMA 8980
						GMA 8990
						GMA 9000
						GMA 9010
						GMA 9020
						GMA 9030
						GMA 9040
						GMA 9050
						GMA 9060
						GMA 9070
						GMA 9080
						GMA 90

GMA	8150
GMA	8160
GMA	8170
GMA	8180
GMA	8190
GMA	8200
GMA	8210
GMA	8220
GMA	8230
GMA	8240
GMA	8250
GMA	8260
GMA	8270
GMA	8280
GMA	8290
GMA	8300
GMA	8310
GMA	8320
GMA	8330
GMA	8340
GMA	8350
GMA	8360
GMA	8370
GMA	8380
GMA	8390
GMA	8400
GMA	8410
GMA	8420
GMA	8430
GMA	8440
GMA	8450
GMA	8460
GMA	8470
GMA	8480
GMA	8490
GMA	8500
GMA	8510
GMA	8520
GMA	8530

INITIAL DISTRIBUTION LIST

	<u>No. Copies</u>
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12
Dean of Research Code Q23 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 Halifax, 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 Department of Mathematical Statistics Columbia University New York, New York 10027	1
George Atkins Department of Computer Science Southwestern State College Weatherford, Oklahoma 73096	1
T. E. Bailey Computing and Information Systems Oklahoma State University Stillwater, Oklahoma 74074	1
Lee J. Bain Math Department University of Missouri at Rolla Rolla, Missouri	1
Jeff Bangert Computation Center Kansas University Lawrence, Kansas 66044	1
R. B. Breitenbach College of Business Administration Oklahoma State University Stillwater, Oklahoma 74074	1
Lyle Broemeling Department of Math and Science Oklahoma State University Stillwater, Oklahoma 74074	1
David Campbell Naval Weapons Systems Analysis Office Bldg. 210 Second Deck Washington Navy Yard Washington, D. C. 20374	1
Gary Carlson Computer Research Center Brigham Young University Provo, Utah 84601	1
John P. Chandler Computing and Information Systems Oklahoma State University Stillwater, Oklahoma 74074	1

Claude Cohen Computer Center Northwestern University 2129 Sheridan Road Evanston, Illinois 60201	1
Eli Cohen Vogelback Computer Center 2129 Sheridan Northwestern University Evanston, Illinois 60201	1
Herbert T. Davis Department of Mathematics University of New Mexico Albuquerque, New Mexico 87106	1
Arthur D. Dayton Department of Statistics Kansas State University Manhattan, Kansas 66502	1
Hamed Eldin Department of Industrial Engineering Oklahoma State University Stillwater, Oklahoma 74074	1
Professor William F. Fellner Virginia Commonwealth University Department of Biometry MCV Station, Box 32 Richmond, Virginia 23298	1
Donald Fisher Computing Information Science Dept. Oklahoma State University Stillwater, Oklahoma 74074	1
J. L. Folks Department of Math and Science Oklahoma State University Stillwater, Oklahoma 74074	1
Professor A. V. Gafarian Department of Industrial and Systems Engineering University of Southern California Los Angeles, CA 90007	1
Charles E. Gates Institute of Statistics Texas A & M University College Station, Texas 77843	1

Dennis E. Grawoig Georgia State University 33 Gilmer Street, N. E. Atlanta, Georgia 30303	1
Joseph L. Gray Oklahoma State University University Computer Center Stillwater, Oklahoma 74074	1
Robert Gumm Computer Center Oklahoma State University Stillwater, Oklahoma 74074	1
G. E. Hedrick Computer Information Science Department Oklahoma State University Stillwater, Oklahoma 74074	1
Dr. David C. Hoaglin National Bureau of Economic Research, Inc. 575 Technology Square Cambridge, Mass. 02139	1
William G. Hunter Department of Statistics University of Wisconsin Madison, Wisconsin 53706	1
William J. Kennedy Statistical Lab Iowa State University Ames, Iowa 50010	1
W. Kirby U.S. Department of the Interior Geological Survey National Center, Mail Stop #430 Reston, Virginia 22090	1
Professor George Marsaglia Department of Computing Science McGill University Montreal, P. Q., Canada	1
Ronald McNew Department of Math And Statistics Oklahoma State University Stillwater, Oklahoma 74074	1
Robert Morrison Department of Math and Statistics Oklahoma State University Stillwater, Oklahoma 74074	1

Wesley L. Nicholson Mathematics and Physics Research Battelle Northwest P. O. Box 999 Richmond, Washington 99352	1
Billie J. Pease Department of the Interior Geological Survey Computer Center Division 18th and C Streets, N. W. Room 1452 Washington, D. C. 20242	1
William S. Peters University of New Mexico Albuquerque, New Mexico 87106	1
Norval F. Pohl Department of Quantative Methods University of Santa Clara Santa Clara, CA	1
Julius Reichbach Technological University Mathematics Room 2004 Holland, Delft 8 Julianalaam 132, The Netherlands	1
David P. Rutten Graduate School of Business Indiana University Bloomington, Indiana 47401	1
David J. Schumacher Lockheed Missiles and Space Company Sunnyvale, CA 94088	1
W. T. Stille Eastman Kodak 343 State Street Rochester, New York	1
Sundaram Swetharanyam Main Campus Computer Center McNeese State University Lake Charles, Louisiana 70601	1
Jack Testerman University of Southwestern Louisiana Box 940 Lafayette, Louisiana 70501	1

Carolyn S. Thompson Medical Center University of Arkansas Little Rock, Arkansas 72201	1
W. M. Usher O.S.U. Computing Center Oklahoma State University Stillwater, Oklahoma 74074	1
James Van Doren Computing Information Sciences Department Oklahoma State University Stillwater, Oklahoma 74074	1
Wray Wilkes Computing Center University of Arkansas Fayetteville, Arkansas	1
Sing-Chou Wu Computer Science and Statistics Department California Polytechnic College San Luis Obispo, California	1
Geoffrey Gates 400 Computer Center Michigan State University E. Lansing, Michigan 48823	1
W. V. Accola Computer Center Oklahoma State University Stillwater, Oklahoma 74074	1
Professor Amrit L. Goel 427 Linil Hall Syracuse University Syracuse, New York 13210	1
Y. C. Lu College of Agriculture Oklahoma State University Stillwater, Oklahoma 74074	1
John W. Meredith Stephen F. Austin State University Box 6163 Nacogdoches, Texas 75961	1
John M. Chambers Bell Telephone Laboratories Murray Hill, New Jersey 97974	1

Patrick L. Odell Texas Technical University Department of Mathematics Lubbock, Texas 79409	1
Prof. W. Morven Gentlemen Dept. Computer Science University of Waterloo Waterloo Ontario, Canada	1
Prof. W. J. Hemmerle Computer Lab University of Rhode Island Kingston, Rhode Island 02881	1
Alan S. Galbraith Math Division U. S. Army Research Office Box CM Duke Station Durham, North Carolina 27706	1
Dean W. M. Woods, Code 024 Dean of Educational Development Naval Postgraduate School Monterey, CA 93940	1
Dean W. F. Koehler, Code 021 Dean of Programs Naval Postgraduate School Monterey, CA 93940	1
Captain D. W. Kiley, Code 03 Director of Programs Naval Postgraduate School Monterey, CA 93940	1
Professor D. G. Williams, Code 0211 Director, Computer Center Naval Postgraduate School Monterey, California 93940	1
Prof. D. E. Harrison, Jr., Code 61Hx Prof. R. L. Kelly, Code 61Ke Dept. of Physics and Chemistry Naval Postgraduate School Monterey, CA 93940	1
Prof. J. A. Galt Department of Oceanography Naval Postgraduate School Monterey, CA 93940	1

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	