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

W. Robinson
A. W. Lewis

Apr 1975

Approved for public release; distribution unlimited
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
Instructor of Computer Science

Peter A. W. Lewis, Professor
Department of Operations Research and Administrative Science

Reviewed by: Released by:

G. L. Barksdale, Jr. R. R. Possum
Chairman Dean of Research
Computer Science Group
NAVAL POSTGRADUATE SCHOOL
GENERATING GAMMA AND CAUCHY RANDOM VARIABLES:
AN EXTENSION TO THE NAVAL POSTGRADUATE SCHOOL
RANDOM NUMBER PACKAGE

D. W. ROBINSON
P. A. W. LEWIS

Naval Postgraduate School
Monterey, California 93940

Chief of Naval Research
Arlington, Virginia 22217

April 1975

58

Approved for public release; distribution unlimited.

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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. Use of the Subroutines</td>
<td>4</td>
</tr>
<tr>
<td>III. Description of the Algorithms</td>
<td>9</td>
</tr>
<tr>
<td>A. Cauchy Generator</td>
<td>9</td>
</tr>
<tr>
<td>B. Gamma Generator GS: A &lt; 1.0</td>
<td>12</td>
</tr>
<tr>
<td>C. Gamma Generator GF: 1.0 &lt; A &lt; 3.0</td>
<td>13</td>
</tr>
<tr>
<td>D. Gamma Generator GO: A &gt; 3.0</td>
<td>15</td>
</tr>
<tr>
<td>E. Ad Hoc Gamma Generators</td>
<td>17</td>
</tr>
<tr>
<td>IV. Summary and Conclusions</td>
<td>19</td>
</tr>
<tr>
<td>References</td>
<td>21</td>
</tr>
<tr>
<td>Listing for Subroutine CAUCHY</td>
<td>22</td>
</tr>
<tr>
<td>Listing for Subroutine GAMA</td>
<td>28</td>
</tr>
<tr>
<td>Distribution List</td>
<td>51</td>
</tr>
</tbody>
</table>
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 algorithms 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

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

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 \( A \) and scale parameter \( S \) has the density function

\[ f(x) = S^A x^{A-1} e^{-sx} / \Gamma(A), \]

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

\[ \Gamma(A) = \int_0^\infty x^{A-1} e^{-x} dx. \]

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

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

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

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 $\Gamma(n/2, 1/2)$ 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 $X(1), \ldots, X(N)$ of Cauchy or $\Gamma(1,0)$ 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
\ldots
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
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

```fortran
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 LLRANDDOM the total core requirement is 9342 bytes:

<table>
<thead>
<tr>
<th>Function</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAMA</td>
<td>1988 bytes</td>
</tr>
<tr>
<td>LLRANDDOM</td>
<td>6189 bytes</td>
</tr>
<tr>
<td>Required IBM Functions</td>
<td>1165 bytes</td>
</tr>
<tr>
<td>Total</td>
<td>9342 bytes</td>
</tr>
</tbody>
</table>

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, \( \lambda \). Note that since special methods are employed when \( \lambda \) 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 \( \lambda \).
<table>
<thead>
<tr>
<th>Shape Parameter $\alpha$</th>
<th>Algorithm</th>
<th>Vector of 100 Variates</th>
<th>Single Variate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>GS</td>
<td>324.0</td>
<td>364.0</td>
</tr>
<tr>
<td>0.3</td>
<td>GS</td>
<td>367.0</td>
<td>402.5</td>
</tr>
<tr>
<td>0.5</td>
<td>GA</td>
<td>70.4</td>
<td>207.7</td>
</tr>
<tr>
<td>0.8</td>
<td>GS</td>
<td>439.8</td>
<td>551.2</td>
</tr>
<tr>
<td>0.9</td>
<td>GS</td>
<td>459.0</td>
<td>611.0</td>
</tr>
<tr>
<td>1.0</td>
<td>GA</td>
<td>68.7</td>
<td>158.9</td>
</tr>
<tr>
<td>1.2</td>
<td>GF</td>
<td>300.1</td>
<td>385.0</td>
</tr>
<tr>
<td>1.4</td>
<td>GF</td>
<td>306.1</td>
<td>441.0</td>
</tr>
<tr>
<td>1.5</td>
<td>GA</td>
<td>141.7</td>
<td>215.8</td>
</tr>
<tr>
<td>1.8</td>
<td>GF</td>
<td>343.6</td>
<td>390.8</td>
</tr>
<tr>
<td>2.0</td>
<td>GA</td>
<td>142.5</td>
<td>203.6</td>
</tr>
<tr>
<td>2.1</td>
<td>GF</td>
<td>396.1</td>
<td>450.8</td>
</tr>
<tr>
<td>2.5</td>
<td>GF</td>
<td>434.7</td>
<td>468.5</td>
</tr>
<tr>
<td>2.9</td>
<td>GF</td>
<td>444.5</td>
<td>496.6</td>
</tr>
<tr>
<td>3.0</td>
<td>GA</td>
<td>206.7</td>
<td>237.1</td>
</tr>
<tr>
<td>3.1</td>
<td>GO</td>
<td>341.5</td>
<td>435.8</td>
</tr>
<tr>
<td>3.5</td>
<td>GO</td>
<td>336.2</td>
<td>373.4</td>
</tr>
<tr>
<td>4.0</td>
<td>GO</td>
<td>332.4</td>
<td>420.7</td>
</tr>
<tr>
<td>5.0</td>
<td>GO</td>
<td>307.7</td>
<td>363.2</td>
</tr>
<tr>
<td>8.0</td>
<td>GO</td>
<td>293.1</td>
<td>371.3</td>
</tr>
<tr>
<td>10.0</td>
<td>GO</td>
<td>289.4</td>
<td>312.5</td>
</tr>
<tr>
<td>20.0</td>
<td>GO</td>
<td>238.2</td>
<td>321.6</td>
</tr>
<tr>
<td>50.0</td>
<td>GO</td>
<td>197.7</td>
<td>284.2</td>
</tr>
<tr>
<td>100.0</td>
<td>GO</td>
<td>178.4</td>
<td>220.0</td>
</tr>
<tr>
<td>1000.0</td>
<td>GO</td>
<td>166.7</td>
<td>177.0</td>
</tr>
<tr>
<td>10000.0</td>
<td>GO</td>
<td>136.4</td>
<td>169.8</td>
</tr>
<tr>
<td>100000.0</td>
<td>GO</td>
<td>152.5</td>
<td>235.8</td>
</tr>
</tbody>
</table>

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 \( 1/n \); in this case a uniform \((0,1)\) variate is returned. The density \( f_2 \) is dealt with by using Marsaglia's almost-linear
density algorithms, just as in Knuth's Algorithm L [5]. The density $f_2$ is sampled with probability $1/2 - 1/x$. 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 LLRAND0M 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.](image-url)
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 $U_2 - U_1 \leq 1 - \frac{1 - x^2}{1 + x^2} (2/\sqrt{2} - 2)$, 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 $U^* \leq 1 - \frac{x^2}{1 + x^2}$ 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

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

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 $2\pi$. 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 $	an 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)
Implied 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 logarithms; 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 $P = e^{-A} U$. (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 GP: $1.0 \leq A \leq 3.0$

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 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 $\lambda$. Let

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

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

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

**Proof:**

Unconditionally, $E$ has density

$$h(x) = \frac{1}{\lambda} e^{-x/\lambda}.$$

Therefore,

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

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,

$$\Pr(U \leq g(E)) = E[ \Pr(U \leq g(E) \mid E) ]$$

$$= \int_0^1 g(x) h(x) \, dx$$

$$= \Gamma(A) \frac{A-1}{\lambda}$$

$$= C(A)$$

Thus, in view of (4),
\[ f_E(x) = \frac{h(x)q(x)}{C(A)} \]
\[ = \frac{A^{-1} - x}{\Gamma(A)} \]

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 \( C(A) \). When \( A \) is large we have from Stirling’s approximation that \( C(A) \approx \frac{2^{2} e^{-x}}{\sqrt{\pi x}} \), 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 \( E_2 < \frac{E_1 - 1}{2} \) then go back to Step 1.
3. (Acceptance) Deliver \( x = A^{E_1} \).

D. Gamma Generator GO: \( A \geq 3.0 \)

This method was originally developed by Dieter and Ahrens and is fully described in [1] together with several other gamma generation techniques. Algorithm 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:
   \[
   m = A - 1; \\
   s^2 = \sqrt{\frac{2A}{3}} + A; \quad s = \sqrt{s^2}; \\
   d = \sqrt{s^2}; \quad b = d + m; \\
   w = s^2 / m - 1; \quad v = 2s^2 / (m\sqrt{A}); \\
   c = b + \ln s d - 2m - 3.7203285. 
   \]

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 \( S = \frac{N^2}{2} \).
If \( N > 0 \) go to Step 5.

4. (Left-hand rejection) If \( U < 1 + S (\sqrt{N} - w) \) 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 \( \ln U < \ln \frac{x}{x + m - x + S} \) 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 \( \# \left( \frac{x - \ln x}{d} \right) + c > E_2 \) 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 \( \alpha_1 \) and \( \alpha_2 \) and equal scale parameters has the gamma distribution with shape parameter \( \alpha_1 + \alpha_2 \) 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 \( \alpha \) 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. \( \frac{N^2}{2} \) has the gamma distribution with unit scale parameter and \( \alpha = 0.5 \). We use this extension for \( \alpha = 0.5 \) or \( 1.5 \).
The resulting algorithm is then

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

1. (Find $K$) Set $K = \lfloor A \rfloor$, where $\lfloor A \rfloor$ 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 $X = E_i + \ldots + E_k$.

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

$$Z = \frac{X_1}{X_1 + X_2}$$

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 \textit{t-Distribution} may be sampled as the ratio of a standard normal and an independent chi-squared random variate, while the \textit{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


**** CAUCHY DEVIATE GENERATOR ****

* PURPOSE:                  CAU00020
  GENERATION OF RANDOM VARIATES WITH THE CAUCHY DISTRIBUTION  CAU00040
* USAGE:                   CAU00050
  CALL CAUCHY (IX, C, N)             CAU00060
* PARAMETERS:             CAU00070
  IX    SEED FOR RANDOM NUMBER GENERATOR (INTEGER*4). SHOULD BE  CAU00080
       INITIALIZED TO ANY POSITIVE VALUE IN THE CALLING PROGRAM  CAU00090
       AND NOT ALTERED THEREAFTER.                   CAU00100
  C     ARRAY TO HOLD THE GENERATED VARIATES (REAL*4). MUST BE  CAU00110
       DIMENSIONED AT LEAST N.                       CAU00120
  N     NUMBER OF CAUCHY DEVIATES TO GENERATE (INTEGER*4).      CAU00130
* METHOD:                 CAU00140
  A COMBINED DECOMPOSITION/REJECTION METHOD IS USED. ALL  CAU00150
  SUBDISTRIBUTIONS CAN BE SAMPLED USING UNIFORM DEVIATES ONLY.  CAU00160
* SUBROUTINES REQUIRED:  CAU00170
  NONE                      CAU00180
* PROGRAMMER:            CAU00190
  D.W. ROBINSON             CAU00200
* DATE:                   CAU00210
  9 MAY 1974              CAU00220
**** 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: NORMALIZATION COMPAREND = 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 **

- MA: RAND
- &A: 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

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
**** CAUCHY DEVIATE GENERATOR ****

CAUCHY
CSECT
using CAUCHY,R15
DEFINE BASE REGISTER
B
R12(R13)
BRANCH AROUND ID
DC
AL1(6)
MODULE NAME
DC
CL6'CAUCHY'
ST
R14,R12,12(R13)
SAVE CALLING PROGRAM REGS
ST
R13,SVAREA+4
CALCULATE SAVE ADDRESS IN OWN AREA
LR
R2,R13
COPY CALLING SAVE ADDRESS TO R2
LA
R13,SVAREA
OWN SAVE AREA IN R13
ST
R13,B1(R2)
FORWARD LINK

* *
LM
R3,R5,0(R1)
GET PARAMETER ADDRESSES
LR
R13,R3
SAVE SEED ADDRESS
L
R7,0(R3)
GET SEED VALUE
L
R3,0(R5)
LOAD NUMBER OF DEVIATES TO GENERATE
SLA
R3,2
CONVERT N TO BYTES
LA
R2,4
CONSTANT 4 FOR MAIN LOOP
SR
R4,R2
BACK UP 4 IN CALLER'S ARRAY
LR
R5,R2
INITIAL ARRAY INDEX
LM
R8,R12,LOOPCON
LOAD MAIN LOOP CONSTANTS
CNOP
0,8
ALIGN BXLE LOOP FOR SPEED

* *
MAINLOOP
RAND
GET FIRST UNIFORM
LR
R0,R6
SAVE TWO BITS OF X(N) IN R0
LR
R1,R6
NEXT TO LAST BIT IN R1
SRL
R1,1

NR
R1,R11
TEST BIT IN R1; IF 0, SAMPLE FROM TAIL
B2
TAIL

* 
C
R6=1367130551
SELECT RECTANGLE/WEDGE SAMPLING
BH
WEDGE

* 
REXT
RAND
GET NEXT UNIFORM
SRL
R6,7
MAKE ROOM FOR EXPONENT
OR
R6,R9
"OR" ON THE EXPONENT
ST
R6,UNIF
STORE THE UNIFORM
LE
FR0,UNIF

CR
R6,R10
TEST FOR NORMALIZATION
BCR
R12
QUIT IF NOT NEEDED
AE
FR0,E'0.0'
NORMALIZE THE UNIFORM
BR
R12
GO TO END OF LOOP

CAU00850
CAU00850
CAU00850
CAU00850
CAU00850
CAU00900
CAU00910
CAU00920
CAU00930
CAU00940
CAU00950
CAU00960
CAU00970
CAU00980
CAU00990
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 ****

<table>
<thead>
<tr>
<th><strong>WEDGE</strong></th>
<th><strong>RAND</strong></th>
<th><strong>R1,R6</strong></th>
<th><strong>SAVE FIRST UNIFORM</strong></th>
<th><strong>CAU01300</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RAND</strong></td>
<td><strong>R6,R1</strong></td>
<td><strong>GET UNIFORM IN R6 &lt; UNIFORM IN R1</strong></td>
<td><strong>CAU01310</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BNH</strong></td>
<td><strong>R6,R1</strong></td>
<td><strong>EXCHANGE REGISTERS</strong></td>
<td><strong>CAU01320</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LR</strong></td>
<td><strong>R1,R7</strong></td>
<td><strong>EASY REJECTION TEST</strong></td>
<td><strong>CAU01330</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BL</strong></td>
<td><strong>SAMPL</strong></td>
<td><strong>ACCEPT WEDGE SAMPLE</strong></td>
<td><strong>CAU01340</strong></td>
<td></td>
</tr>
<tr>
<td><strong>SRL</strong></td>
<td><strong>R6,7</strong></td>
<td><strong>CONVERT MINIMUM UNIFORM TO REAL</strong></td>
<td><strong>CAU01350</strong></td>
<td></td>
</tr>
<tr>
<td><strong>OR</strong></td>
<td><strong>R6,R9</strong></td>
<td><strong>ON THE EXponent</strong></td>
<td><strong>CAU01360</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ST</strong></td>
<td><strong>R6,UNIF</strong></td>
<td><strong>CONVERT MAXIMUM UNIFORM TO REAL</strong></td>
<td><strong>CAU01370</strong></td>
<td></td>
</tr>
<tr>
<td><strong>OR</strong></td>
<td><strong>R1,R9</strong></td>
<td><strong>ON THE EXponent</strong></td>
<td><strong>CAU01380</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ST</strong></td>
<td><strong>R1,U2</strong></td>
<td><strong>LOAD TRIAL VARIATE</strong></td>
<td><strong>CAU01390</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LE</strong></td>
<td><strong>FR0,UNIF</strong></td>
<td><strong>TEST FOR NORMALIZATION</strong></td>
<td><strong>CAU01400</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CR</strong></td>
<td><strong>R6,R10</strong></td>
<td><strong>GET FIRST COMPARAND FOR REJECTION TEST</strong></td>
<td><strong>CAU01410</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BC</strong></td>
<td><strong>11,</strong></td>
<td><strong>NORMALIZE X</strong></td>
<td><strong>CAU01420</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AE</strong></td>
<td><strong>FR0,E:0.0</strong></td>
<td><strong>U2 - X</strong></td>
<td><strong>CAU01430</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LE</strong></td>
<td><strong>FR2,U2</strong></td>
<td>**FIND X <strong>2</strong></td>
<td><strong>CAU01440</strong></td>
<td></td>
</tr>
<tr>
<td><strong>SER</strong></td>
<td><strong>FR2,FR0</strong></td>
<td><strong>FIND QUOTIENT</strong></td>
<td><strong>CAU01450</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LER</strong></td>
<td><strong>FR4,FR0</strong></td>
<td><strong>HARD REJECTION TEST</strong></td>
<td><strong>CAU01460</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LCER</strong></td>
<td><strong>FR0,FR4</strong></td>
<td><strong>GO BACK IF TEST FAILED</strong></td>
<td><strong>CAU01470</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AE</strong></td>
<td><strong>FR6,E:1.0</strong></td>
<td>**1 - X <strong>2</strong></td>
<td><strong>CAU01480</strong></td>
<td></td>
</tr>
<tr>
<td><strong>AE</strong></td>
<td><strong>FR6,E:1.0</strong></td>
<td>**1 + X <strong>2</strong></td>
<td><strong>CAU01490</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DAR</strong></td>
<td><strong>FR6,FR4</strong></td>
<td><strong>1 + SQRT(2)</strong></td>
<td><strong>CAU01500</strong></td>
<td></td>
</tr>
<tr>
<td><strong>ME</strong></td>
<td><strong>FR6,E:82842712</strong></td>
<td><strong>1/ (1 + SQRT(2))</strong></td>
<td><strong>CAU01510</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CER</strong></td>
<td><strong>FR2,FR6</strong></td>
<td><strong>13,R12</strong></td>
<td><strong>CAU01520</strong></td>
<td></td>
</tr>
<tr>
<td><strong>B</strong></td>
<td><strong>WEDGE</strong></td>
<td><strong>GO BACK IF TEST FAILED</strong></td>
<td><strong>CAU01530</strong></td>
<td></td>
</tr>
</tbody>
</table>
**** CAUCHY DEVIATE GENERATOR ****

| TAIL | SRL R6,7 | MAKE ROOM FOR EXPONENT | CAU01620 |
|      | OR R6,R9 | "OR" ON THE EXPONENT   | CAU01630 |
|      | ST R6,UNIF | STORE THE UNIFORM | CAU01640 |
|      | LE FRO=E'1.0' | GET 1 / UNIFORM | CAU01650 |
|      | DE FRO,UNIF |                | CAU01660 |
| RAND | 1 | GET ANOTHER UNIFORM FOR REJECTION TEST | CAU01670 |
| SRL R6,7 | MAKE ROOM FOR EXPONENT | CAU01680 |
| OR R6,R9 | "OR" ON THE EXPONENT | CAU01690 |
| ST R6,UNIF |                | CAU01700 |
|    | LER FR2,FRO | FIND X ** 2 | CAU01710 |
|    | MER FR2,FRO | GET 1 + X ** 2 | CAU01720 |
|    | LER FR4,FR2 |                | CAU01730 |
|    | AE FR4=E'1.0' | FIND COMPARAND FOR REJECTION TEST | CAU01740 |
|    | CER FR4,FR2 | REJECTION TEST | CAU01750 |
|    | BCR 13,R12 |                | CAU01760 |
|    | RAND | ANOTHER UNIFORM FOR NEXT PASS | CAU01770 |
|    | B TAIL | GO BACK | CAU01780 |
|    | ENDLOOP | TEST SAVED BIT | CAU01790 |
|    | NR R0,R11 |                | CAU01800 |
|    | BZ **6 | IF BIT = 0, QUIT | CAU01810 |
|    | LCE RFR0,FRO | IF BIT = 1, X = -X | CAU01820 |
|    | STE FRO,0(R4,R5) | STORE VARIATE IN CALLER'S ARRAY | CAU01830 |
|    | BXLE R5,R2,MAINLOOP | BRANCH BACK FOR ANOTHER VARIATE | CAU01840 |
|    | ST R7,04(R13) | SEND LAST SEED BACK TO CALLING PROGRAM | CAU01850 |
|    | L R13,AREA+4 | GET CALLING SAVE AREA ADDRESS | CAU01860 |
|    | LM R14,R12,12(R13) | RESTORE CALLING PROG REGS | CAU01870 |
|    | BR R14 | RETURN | CAU01880 |
**** CAUCHY DEVIATE GENERATOR ****

* DATA AREA
  ** SVAREA ** DS 18F SAVE AREA
  ** UNIF ** DS F TEMP STORAGE FOR UNIFORM RANDOM VARIATES
  ** U2 ** DS F

  ** LOOPCON **
  DC '16807' MULTIPLIER FOR GENERATOR => R8
  DC '40000001' EXPONENT CONSTANT => R9
  DC '40100000' NORMALIZATION TEST CONSTANT => R10
  DC '1' MASK CONSTANT => R11
  DC AL4(ENDLOOP) END OF LOOP ADDRESS => R12

* 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

CAU001960
CAU001970
CAU001980
CAU001990
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 ****

* SUBROUTINES REQUIRED:
  - THE LEWIS AND LEARMONTH RANDOM NUMBER GENERATOR PACKAGE LLRAN
  - DEEDED. 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 ****

* REGISTER ALLOCATION
  * R0  LINKAGE
  * R1  LINKAGE
  * R2  CONSTANT 4
  * R3  NO DEVIATES WANTED (BYTES)
  * R4  CALLER'S ARRAY ADDRESS
  * R5  ARRAY INDEX
  * R6  (MULTIPLICATION)
  * R7  IX (SEED)
  * R8  MULTIPLIER = 16807
  * R9  EXPONENT CONSTANT
  * R8  V(EXP) OR V(EXPON)
  * R9  V(ALOG)
  * R10 CONSTANT 4
  * R11 ARRAY SIZE
  * R12 ARRAY INDEX
  * R13 END OF BXLE LOOP (GO ONLY)
  * R14 LINKAGE
  * R15 BASE REGISTER
  * FR2 HOLDS GENERATED DEVIATE

* MAIN
  * LOOP
  * UNIFORM
  * GENERATOR
  * (GS, GO ONLY)
  * (GF, GS
  * ONLY)
  * NORMAL/
  * EXPONENTIAL
  * LOOP (GS, GO, GF)

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 ****

* LINKAGE / INITIALIZATION SECTION

**
GAMA
CSECT
USING GAMA,R15
B   101(R15)  DEFINE BASE REGISTER
DC  AL1(4)
DC CL4'GAMA'
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,R1,R2)  FORWARD LINK

**
LM R2,R5,0(R1)  GET PARAMETER ADDRESSES
LE FR0,01,R2)  GET SHAPE PARAMETER
CE FR0,AP  TEST FOR NEW "A" VALUE
BNE SETUP  IF SO, DO PRELIMINARY CALCULATIONS
GWA N
LA R2,4  CONSTANT 4 FOR MAIN LOOP
L R7,04,R3)  PUT SEED INTO R7
L R3,01,R5)  GET NUMBER OF DEVIATES, N
SLA R3,2  CONVERT TO BYTES
SR R4,R2  BACKUP ONE IN CALLER'S ARRAY
LR R5,R9  INITIAL MAIN LOOP INDEX
L R6,METHD  JUMP TO PROPER METHOD
BR R6

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
**GAMMA DEVIATE GENERATOR**

<table>
<thead>
<tr>
<th>SETUP AND CONSTANT CALCULATION</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SETUP</strong></td>
<td><strong>FRO,FRO</strong></td>
</tr>
<tr>
<td><strong>BNP</strong></td>
<td><strong>THRU</strong></td>
</tr>
<tr>
<td><strong>CE</strong></td>
<td><strong>FRO,+E‘0.5’</strong></td>
</tr>
<tr>
<td><strong>BE</strong></td>
<td><strong>S1</strong></td>
</tr>
<tr>
<td><strong>CE</strong></td>
<td><strong>FRO,+E‘1.0’</strong></td>
</tr>
<tr>
<td><strong>BE</strong></td>
<td><strong>SGS</strong></td>
</tr>
<tr>
<td><strong>CE</strong></td>
<td><strong>FRO,+E‘1.5’</strong></td>
</tr>
<tr>
<td><strong>BE</strong></td>
<td><strong>S3</strong></td>
</tr>
<tr>
<td><strong>CE</strong></td>
<td><strong>FRO,+E‘2.0’</strong></td>
</tr>
<tr>
<td><strong>BE</strong></td>
<td><strong>S4</strong></td>
</tr>
</tbody>
</table>

**SET UP FOR LARGE PARAMETER METHOD, ALGORITHM "GO"**

| **SGO** | **R0,GO** | **SET ADDRESS FOR SUBSEQUENT CALLS** |
| **ST** | **R0,METHOD** | **INITIALIZE RANDOM ARRAY INDEX** |
| **CE** | **R0,INX1** | **TEST FOR NEW SHAPE PARAMETER** |
| **BE** | **GWAN** | **GO AHEAD IF NOT** |
| **ST** | **FRO,AGO** | **SAVE NEW SHAPE PARM** |
| **LE** | **F2,+E‘1.0’** | **GET CONSTANT 1.0** |
| **SER** | **FRO,FR2** | **COMPUTE MU = A - 1.0** |
| **STE** | **FRO,MU** | **COMPUTE MUP = 1 / MU** |

**LINK TO SQRT FUNCTION FOR SQRT(A)**

| **LA** | **R1,ARGLIST1** | **LOAD ARGUMENT LIST** |
| **LR** | **R0,R15** | **SAVE BASE REGISTER** |
| **L** | **R15,VADDERS** | **ADDRESS OF SQRT FUNCTION** |
| **BALR** | **R14,R15** | **RESTORE BASE REGISTER** |
| **LR** | **R15,R8** | **SAVE SQRT(A)** |
| **LER** | **FRO,FR2** | **FIND NORMAL VARIANCE** |
| **ME** | **FRO,+E‘1.6329932’** | **FIND NORMAL VARIANCE** |
| **AE** | **FRO,AGO** | **FIND NORMAL VARIANCE** |
| **STE** | **FRO,SIGMA** | **FIND NORMAL VARIANCE** |
### GAMMA DEVIATE GENERATOR

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DE</td>
<td>FRO,MU               FIND REJECTION CONSTANT &quot;WM&quot;</td>
<td>GMA 2010</td>
</tr>
<tr>
<td>SE</td>
<td>FRO1=1.0</td>
<td></td>
</tr>
<tr>
<td>SE</td>
<td>FRO1=MU</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>FR21=1.6329932E01    FIND REJECTION CONSTANT &quot;VP&quot;</td>
<td>GMA 2040</td>
</tr>
<tr>
<td>DE</td>
<td>FR21=MU</td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>FR21=2.0</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FR21=VP</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>LINK TO SQRT FUNCTION TO FIND NORMAL STD DEV</td>
<td></td>
</tr>
<tr>
<td>LA</td>
<td>R11=ARGST2           LOAD ARGUMENT LIST ADDRESS</td>
<td>GMA 2100</td>
</tr>
<tr>
<td>L</td>
<td>R15=VADDX            ADDRESS OF SQRT FUNCTION</td>
<td></td>
</tr>
<tr>
<td>BALR</td>
<td>R14,R15</td>
<td></td>
</tr>
<tr>
<td>LR</td>
<td>R15,R8               RESTORE BASE REGISTER</td>
<td>GMA 2140</td>
</tr>
<tr>
<td>STE</td>
<td>FRO1=SIGMA           SAVE STD DEV</td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>FRO1=1.4494897E01    FIND REJECTION CONSTANT &quot;DP&quot;</td>
<td>GMA 2170</td>
</tr>
<tr>
<td>LE</td>
<td>FR21=1.0</td>
<td></td>
</tr>
<tr>
<td>DER</td>
<td>FR21=FRO</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FR21=DP</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FRO1=0</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>FRO1=MU              FIND UPPER LIMIT FOR NORMAL METHOD, &quot;B&quot;</td>
<td>GMA 2230</td>
</tr>
<tr>
<td>STE</td>
<td>FRO1=B</td>
<td></td>
</tr>
<tr>
<td>LE</td>
<td>FR21=1.0</td>
<td></td>
</tr>
<tr>
<td>DER</td>
<td>FR21=FRO</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FR21=BP</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>LE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FR21=SIGMA           COMPUTE REJECTION CONSTANT &quot;CONS&quot;</td>
<td>GMA 2290</td>
</tr>
<tr>
<td>ME</td>
<td>FR21=D</td>
<td></td>
</tr>
<tr>
<td>DER</td>
<td>FR21=FRO</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FR21=CONS</td>
<td></td>
</tr>
<tr>
<td>LA</td>
<td>R11=ARGST3           LOAD ARGUMENT LIST ADDRESS</td>
<td>GMA 2320</td>
</tr>
<tr>
<td>L</td>
<td>R15=VADDLX           ADDRESS OF ALOG FUNCTION</td>
<td></td>
</tr>
<tr>
<td>BALR</td>
<td>R14,R15</td>
<td></td>
</tr>
<tr>
<td>LK</td>
<td>R15,R8               RESTORE BASE ADDRESS</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>LCER</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F40=FRO              COMPLETE COMPUTATION OF &quot;CONS&quot;</td>
<td>GMA 2380</td>
</tr>
<tr>
<td>SE</td>
<td>FRO1=B</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>FRO1=MU</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>FRO1=MU</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>FRO1=3.7203285E01</td>
<td></td>
</tr>
<tr>
<td>STE</td>
<td>FRO1=CONS</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>GWAN                  DONE WITH INITIALIZATION. PROCEED TO GENERATION</td>
<td>GMA 2460</td>
</tr>
</tbody>
</table>
**** GAMMA DEVIATE GENERATOR ****

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA R0,GF</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td>GMA 2510</td>
</tr>
<tr>
<td>ST R0,METHOD</td>
<td>SET UP FOR FISHMAN'S METHOD, ALGORITHM &quot;GF&quot;</td>
<td>GMA 2620</td>
</tr>
<tr>
<td>SE FRO=E'1.0'</td>
<td>COMPUTE AMINUS = A - 1</td>
<td>GMA 2530</td>
</tr>
<tr>
<td>STE FRO,AMINUS</td>
<td></td>
<td>GMA 2540</td>
</tr>
<tr>
<td>LA R0,20</td>
<td>INITIALIZE RANDOM ARRAY INDEX</td>
<td>GMA 2550</td>
</tr>
<tr>
<td>ST R0,INX2</td>
<td>DONE WITH INITIALIZATION. PROCEED TO GENERATION.</td>
<td>GMA 2560</td>
</tr>
<tr>
<td>B GWAN</td>
<td></td>
<td>GMA 2570</td>
</tr>
<tr>
<td>SGS LA R0,GS</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td>GMA 2580</td>
</tr>
<tr>
<td>ST R0,METHOD</td>
<td>SET UP FOR SMALL PARAMETER METHOD. &quot;GS&quot;</td>
<td>GMA 2590</td>
</tr>
<tr>
<td>LE FR2,FRO</td>
<td>COMPUTE 1 - A</td>
<td>GMA 2600</td>
</tr>
<tr>
<td>SE R2,FR4</td>
<td></td>
<td>GMA 2610</td>
</tr>
<tr>
<td>LER FR2,FR0</td>
<td></td>
<td>GMA 2620</td>
</tr>
<tr>
<td>LRER FR2,FR2</td>
<td></td>
<td>GMA 2630</td>
</tr>
<tr>
<td>STE FR2,AMIN1</td>
<td></td>
<td>GMA 2640</td>
</tr>
<tr>
<td>DER FR4,FRO</td>
<td></td>
<td>GMA 2650</td>
</tr>
<tr>
<td>STE FR4,AINV</td>
<td></td>
<td>GMA 2660</td>
</tr>
<tr>
<td>ME FRO=E',36787944'</td>
<td>FIND (E + A) / E</td>
<td>GMA 2670</td>
</tr>
<tr>
<td>AE FRO=E',1.0'</td>
<td></td>
<td>GMA 2680</td>
</tr>
<tr>
<td>STE FRO,BGS</td>
<td></td>
<td>GMA 2690</td>
</tr>
<tr>
<td>LA R0,40</td>
<td></td>
<td>GMA 2700</td>
</tr>
<tr>
<td>ST R0,INX3</td>
<td></td>
<td>GMA 2710</td>
</tr>
<tr>
<td>B GWAN</td>
<td></td>
<td>GMA 2720</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GMA 2730</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
#### Gamma Deviate Generator

**SET UP FOR AD HOC METHODS**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>LA</td>
<td>RO,CHIS01</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 2880</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 2890</td>
</tr>
</tbody>
</table>

**SET UP FOR CHI-SQUARED, 1 DEGREE OF FREEDOM ( A = 0.5 )**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>LA</td>
<td>RO,CHIS01</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 2880</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 2890</td>
</tr>
</tbody>
</table>

**SET UP FOR EXPONENTIAL ( A = 1.0 )**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SEXPN</td>
<td>LA</td>
<td>RO,EXPN</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 2890</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 2890</td>
</tr>
</tbody>
</table>

**SET UP FOR CHI-SQUARED, 3 DEGREES OF FREEDOM ( A = 1.5 )**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S3</td>
<td>LA</td>
<td>RO,CHIS03</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 3000</td>
</tr>
<tr>
<td></td>
<td>LA</td>
<td>RO,40</td>
<td>INITIALIZE RANDOM ARRAY INDEX</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,INX4</td>
<td></td>
<td>GMA 3030</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 3040</td>
</tr>
</tbody>
</table>

**SET UP FOR 2 - ERLANG ( A = 2.0 )**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S4</td>
<td>LA</td>
<td>RO,CHIS04</td>
<td>SET ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 3070</td>
</tr>
<tr>
<td></td>
<td>LA</td>
<td>RO,40</td>
<td>INITIALIZE RANDOM ARRAY INDEX</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,INX4</td>
<td></td>
<td>GMA 3100</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 3120</td>
</tr>
</tbody>
</table>

**SET UP FOR 3 - ERLANG ( A = 3.0 )**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S6</td>
<td>LA</td>
<td>RO,CHIS06</td>
<td>SAVE ADDRESS FOR SUBSEQUENT CALLS</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,METHOD</td>
<td></td>
<td>GMA 3150</td>
</tr>
<tr>
<td></td>
<td>LA</td>
<td>RO,40</td>
<td>INITIALIZE RANDOM ARRAY INDEX</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>ST</td>
<td>RO,INX5</td>
<td></td>
<td>GMA 3180</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>GWAN</td>
<td>GO ON TO GENERATION</td>
<td>GMA 3190</td>
</tr>
</tbody>
</table>
**** 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 THE
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
LP R7,R6 PUT X(N) INTO R7.
C R7,=F'20556283' SELECT NORMAL OR EXPONENTIAL
BL G0EXP SAMPLING

* REJECTION SAMPLING FROM THE NORMAL DISTRIBUTION
* GONORM
BXLE R12,R16,GONTST INCREMENT NORMAL ARRAY INDEX.
ST R7,IX SAVE CURRENT SEED VALUE.
LR R12,K15 SAVE BASE REGISTER
LA R13,SAREA 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
LR R7,IX RESTORE SEED
CNOP 0,8 ALIGN BXLE LOOP FOR SPEED

* GONTST
LFR0,RNARRAY(R12) LOAD NEXT NORMAL DEVIATE
LER FR2,FR0 TRIAL GAMMA VALUE:
ME FR2,SI GAMA X = NORMAL * SIGMA + MU
AE FR2,MU GAMA
BNP GONORM REJECT X < 0
CE FR2,B GAMA REJECT X > B
* LER FR4,FK0 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
  SLD A 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,*F2147483645* FIXUP OVERFLOW. ADD 2 ** 31 - 3
  LR R7,R6 PUT X(N) INTO R7
  SRL R6,7 MAKE ROOM FOR EXPONENT.
  OR R6,R9 "OR" ON THE EXPONENT
  SI R6,UNIF SAVE THE UNIFORM.
  LTER FRO,FRO PERFORM THE PROPER REJECTION, DEPENDING
  BP GOPOS ON THE SIGN OF THE NORMAL

* GONEG ME FRO,VP COMPUTE THE REJECTION VALUE:
  SE FRO,WM 1 + S2 * (S * VP - WM)
  MER FRO,FR4
  AE FRO =E1.0* REJECTION TEST
  CER FRO,UNIF
  BCR 2,R13 GO TO LOOP END IF PASSED.
  B GON2ST FURTHER TEST IF NOT.

* GOPOS LCE R4,FR4 COMPUTE THE REJECTION VALUE:
  ME FRO,WM 1 - S2 * WM
  AE FRO,*E1.0* REJECTION TEST
  CCR 2,R13 GO TO LOOP END IF PASSED.

* GON2ST SER FR4,FR2 FIND PARTIAL SUM FOR REJECTION TEST:
  AE FR4,MO SUM = MU - X + S2
  STS FR4,SUM
  STE FR2,X SAVE TRIAL GAMMA DEVIATE
  ME FR2,MU 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
  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 GMA 4110
  AE FRO,SUM GET REJECTION VALUE GMA 4120
  STE FRO,SUM GMA 4130

* LA R1,ARGLST6 SECOND LINK TO LOG FUNCTION GMA 4140
  L R15,VADDLG ADDRESS OF LOG FUNCTION GMA 4150
  BALR R12,R15 RESTORE BASE REGISTER GMA 4160
  LR R15,R12 GOSAVE RESTORE OTHER REGS GMA 4170

* LE FRO,X RELOAD TRIAL GAMMA GMA 4180
  GE FRO,SUM FINAL REJECTION TEST GMA 4190
  BCR 13,R13 PASSED TEST. GO TO LOOP END. GMA 4200
  B GOLoopili FAILED TEST. BRANCH BACK FOR ANOTHER TRY. GMA 4210

* REJECTION SAMPLING FROM THE EXPONENTIAL DISTRIBUTION.
  * GOEXP ST R7,IX GET TWO EXPONENTIAL DEVIATES. FIRST GMA 4220
     SAVE SEED. GMA 4230
    STM R12,R13,GOSAVE SAVE PROGRAM REGS. GMA 4240
    LR R12,R15 SAVE BASE REGISTER. GMA 4250
    LA R13,SVAREA SAVE AREA POINTER. GMA 4260
    LA R1,ARGLST7 ARGUMENT LIST ADDRESS. GMA 4270
    L R15,VADDEX ADDRESS OF EXPONENTIAL GENERATOR. GMA 4280
    BALR R16,R15 LINK TO "EXPON" GMA 4290
    LR R15,R12 RESTORE BASE REGISTER. GMA 4300

* LE FRO,RNEXP FIND TRIAL GAMMA VALUE:
  ME FRO,DP X = B * (1 + R * DP) GMA 4310
  AE FRO,=E'1.0' GMA 4320
  ME FRO,B GMA 4330
  STE FRO,X SAVE TRIAL GAMMA VALUE GMA 4340
  ME FRO,MUP GET LOG (X / MU) GMA 4350
  STE FRO,LOG GMA 4360
  LA R1,ARGLST5 LOAD ARGUMENT LIST ADDRESS GMA 4370
  L R15,VADDLG ADDRESS OF LOG FUNCTION. GMA 4380
  BALR R14,R15 LINK TO "ALOG" GMA 4390
  LR R15,R12 RESTORE BASE REGISTER GMA 4400
  LM R12,R13,GOSAVE RESTORE OTHER REGS GMA 4410

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

LE   FR2,X      RELOAD TRIAL GAMMA VALUE.   GMA 4540
LER  FR4;FR2    COMPLETE CALCULATION OF REJECTION VALUE. GMA 4550
ME   FR4,BP     MU *(LOG - X * BP) + CONS.       GMA 4560
SER  FR0,FR4    GMA 4570
ME   FR0,MU     GMA 4580
AE   FR0,CONS   GMA 4590
LCER FR0,FR0    GMA 4600
CE   FR0,RNEXP+4 PERFORM REJECTION TEST. GMA 4610
BH   GOLOOP     BACK TO START IF FAILED.   GMA 4620
**   END OF METHOD "GO" LOOP. GMA 4630
**   GENERATED DEVIATE IS IN FR2. GMA 4640
**   ENDGO STRE FR2,O(R4,R5) STORE DEVIATE IN CALLER'S ARRAY. GMA 4650
BXLE R5,R2,GOLOOP BRANCH BACK FOR ANOTHER DEVIATE. GMA 4660
ST   R12,INX1   SAVE LAST ARRAY INDEX. GMA 4670
B    THRU ALL DONE. QUIT. GMA 4680

GMA 4690
GMA 4700
**** 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
    LR R1, ARGLST4 EXPONENTIAL ARRAY EXHAUSTED, REPLENISH IT
    R15, R8 LOAD ARGUMENT LIST ADDRESS
    BALR R14, R15 ADDRESS OF "EXPON"
    LR R15, R9 LINK TO EXPONENTIAL GENERATOR
    SR R12, R12 RESTORE ALOG ADDRESS TO R15
    CNOP 0, 8 SET ARRAY INDEX TO START
    ALIGN BXLE LOOP FOR SPEED
  * GFTST
    L R6, RNARRAY(R12) TAKE LOGARITHM OF ONE EXPONENTIAL
    R6, GFCON
    ST DEViate
    LR R1, ARGLST8 LOAD ARGUMENT LIST ADDRESS
    BALR R14, R15 LINK TO "ALOG"
    FR2, RNARRAY(R12) FINISH COMPUTING REJECTION VALUE:
    (A - 1) * (R - LN R - 1)
    LER FR4, FR2 GFTST
    SER FR4, FR0
    SE FR4, E'1.0'
    ME FR4, AMINUS
    CE FR4, RNARRAY+20(R12) REJECTION TEST
    BH GFCON
    ME FR2, AP
    STE FR2, (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
    LR 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 )
  * CHISQL
    LR R12,R15  SAVE BASE REGISTER
    LA R14,R11  SKIP OVER SHAPE PARAMETER IN ARG LIST
    BALR R14,R15 LINK TO "NORMAL"
    LR R15,R12  RESTORE BASE REGISTER
    L R7,01,R11 GET SEED VALUE IN REG 7
    CNOP 0,8  ALIGN BXLE LOOP FOR SPEED
  *
  * CHLOOPL
    LE FRO,0(R4,R5) GET NEXT NORMAL
    HER FRO,FRO SQUARE THE NORMAL
    SEL FRO,FRO AND MULTIPLY BY 0.5
    BXLE R5,R2,CHLOOPL PUT GAMMA DEVIATE INTO CALLER'S ARRAY
    B THRU BRANCH BACK FOR NEXT NORMAL
    QUIT

* EXPONENTIAL METHOD ( A = 1.0 )
  * EXPN
    LR R12,R15  SAVE BASE REGISTER
    LA R14,R11  SKIP OVER SHAPE PARAMETER IN ARG LIST
    L R15,VADDUX LINK DIRECTLY TO "EXPON"
    BALR R14,R15
    LR R15,R12  RESTORE BASE REGISTER
    L R7,01,R11 GET SEED VALUE IN R7
    B THRU QUIT.
**Gamma Deviate Generator**

* CHI-SQUARED, 3 DEGREES OF FREEDOM (A = 1.5)  
  **CHISQ3**
  - LR R6, R15
  - DROP R15
  - USING GAMA, R6
  - LA R1, 4(R1)
  - L R15, VADDDX
  - BALR R14, R15
  - L R7, 0(R1)
  - ST R7, IX
  - LM R10, R12, CHICON3
  - CNOP 0, 8

* CHLOOP3
  - BXLE R12, R10, CH3COMP

* CH3COMP
  - LE FRO, RNRARRAY(R12)
  - MER FRO, FRO
  - HER FRO, FRO
  - AE FRO, 0(R4, R5)
  - STE FRO, 0(R4, R5)
  - BXLE R5, R2, CHLOOP3

* L R7, IX
  - ST R12, IX4
  - 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 ****

* 2 - ERLANG ( A = 2.0 )

CHISQ4 LR R6,R15 SHIFT BASE REGISTER
LA R1,4l,R1) SKIP OVER SHAPE PARAMETER IN ARG LIST
L R15,VADEECX 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,CHICON3 LOAD LOOP CONSTANTS
CNOP 0,8 ALIGN BXLE LOOP FOR SPEED

* CHLOOP4 BXLE R12,R10,CH4COMP GET NEXT EXPONENTIAL

* EXPONENTIAL ARRAY EXHAUSTED. REPLENISH IT
L R15,VADEECX LINK TO "EXPON"
LA R1,AGRST4 GET ARGUMENT LIST
BALR R14, R15 LINK TO "EXPON"
SR R12,R12 RESET ARRAY INDEX TO ZERO

* CH4COMP LE FRO,RNARAY(R12) LOAD NEW EXPONENTIAL
AE FRO,01(R4,R5) ADD TO SECOND EXPONENTIAL
STE FRO,0IR4,R5) STORE GENERATED GAMMA IN CALLER'S ARRAY
BXLE R5,RZ,CHLOOP4 GO BACK FOR NEXT DEVIATE

* L R7,IX LOAD LAST SEED VALUE
ST R12,INX4 SAVE RANDOM ARRAY INDEX
LR R15,R6 RESTORE BASE REGISTER
B THRU 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 ****

* 3 - ERLANG ( A = 3.0 )

CHISQ6
LR R6,R15
LA R1,4(R1)
L R15,VADDEX
BALR R14,R15
L R7,0(R1)
ST R7,IX
LM R10,R12,CHICON6
GNDP 0:8

* CHLOOP6
BXLE R12,R10,CH6COMP
L R15,VADDEX
LA R1,ARGST4
BALR R14,R15
SR R12,R12

* CH6COMP
LE FRO,RNARRAY(R12)
AE FRO,RNARRAY+20(R12)
AE FRO,0(R4,R5)
STE FRO,0(R4,R5)
BXLE R5,R2,CHLOOP6

* L R7,IX
ST R12,INX5
LR R15,R6
DROP R6
USING GAMA,R15
B THRU

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 ****

* SMALL PARAMETER METHOD "GS" (AHRENS)

LM R8,R12,GS CON 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
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 FIND P = B * UNIFORM
STE FRO,P

* LM R8,R9,GS VCON 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
EXPOENTIAL 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 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 X816

* 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 | GMA 6890 |
| STE | FRO,P | | GMA 6900 |
| LR | R15,R8 | LINK TO EXPONENTIAL FUNCTION | GMA 6910 |
| LA | R1,ARGLST9 | LOAD ARGUMENT LIST ADDRESS | GMA 6920 |
| BALR | R14,R13 | RESULT IS P ** (1 / A) | GMA 6930 |
| CIE | FRO,RNARRAY(R12) | REJECTION TEST | GMA 6940 |
| BNP | ENDS | QUIT IF OK | GMA 6950 |
| LM | R8,R9,GCON | OTHERWISE GO BACK | GMA 6960 |
| LR | R15,R6 | RESET BASE REGISTER | GMA 6970 |
| B | GSGOOP | | GMA 6980 |

* XBIG | LE | FR2,BGS | FIND (B - P) / A | GMA 6990 |
| SER | FR2,FRO | | GMA 7000 |
| ME | FR2,AINV | | GMA 7010 |
| STE | FR2,P | | GMA 7020 |
| LA | R1,ARGLST9 | NOW LINK TO LOG FUNCTION: | GMA 7030 |
| LR | R15,R9 | ADDRESS OF LOG FUNCTION | GMA 7040 |
| BALR | R14,R15 | RESULT IS LOG (B - P) / A | GMA 7050 |
| LCLR | FRO,FRO | TRIAL GAMMA IS - LOG | GMA 7060 |
| STE | FRO,P | NOW FIND LOG OF TRIAL VALUE | GMA 7070 |
| LA | R1,ARGLST9 | LOAD ARGUMENT LIST ADDRESS | GMA 7080 |
| LR | R15,R9 | ADDRESS OF LOG FUNCTION | GMA 7090 |
| BALR | R14,R15 | | GMA 7100 |
| ME | FRO,AAMI | FINISH CALCULATION OF REJECTION VALUE | GMA 7110 |
| CIE | FRO,RNARRAY(R12) | REJECTION TEST | GMA 7120 |
| LE | FRO,P | RELOAD TRIAL GAMMA VALUE | GMA 7130 |
| BNP | ENDS | QUIT IF OK | GMA 7140 |
| LM | R8,R9,GCON | OTHERWISE RESET LOOP CONSTANTS | GMA 7150 |
| LR | R15,R6 | AND CHANGE BASE REGISTER | GMA 7160 |
| B | GSGOOP | AND GO BACK | GMA 7170 |

* * END OF GSGOOP *

** ** GAMMA ** **

| ENDGS | STE | FRO,0(R4,R5) | STORE DEVIATE IN CALLER'S ARRAY | GMA 7200 |
| LM | R8,R9,GCON | RESET LOOP CONSTANTS | GMA 7210 |
| LR | R15,R6 | SHIFT BASE REGISTER | GMA 7220 |
| BXLE | R5,R2,GSGOOP | BRANCH BACK FOR ANOTHER DEVIATE | GMA 7230 |
| ST | R12,INX3 | SAVE LAST ARRAY INDEX | GMA 7240 |
| B | THRU | OTHERWISE QUIT. | GMA 7250 |
| DROP | R6 | | GMA 7260 |
| USING | GAMA,R15 | | GMA 7270 |

| | | | GMA 7280 |
| | | | GMA 7290 |
| | | | GMA 7300 |
**** 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 R7,01,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 METHOD DC E'-1.0' OLD SHAPE PARAMETER
*          DS F ADDRESS FOR PROPER METHOD
*   VADDEX DC V(EXPON) EXTERNAL EXPONENTIAL GENERATOR
*   VALL DC V(NORMAL) EXTERNAL NORMAL GENERATOR
*   VADLS DC V(ALOG) LOGARITHM FUNCTION
*   VADDSR DC V(SQRT) SQUARE ROOT FUNCTION
*   IX RNARRAY DS 10F RANDOM NUMBER SEED
*       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'0.089247598' 1 / B
*      DP DC E'1.3879668' MISC
*      WM DC E'1.1628709' CONSTANTS
*      VP DC E'1.9345306' FOR "GO"
*      CONS DC E'1.2172460'
**** GAMMA DEVIATE GENERATOR ****

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCON</td>
<td>DC F'16807' UNIFORM MULTIPLIER</td>
<td>GMA 7730</td>
</tr>
<tr>
<td></td>
<td>DC X'40000001' EXPONENT CONSTANT</td>
<td>GMA 7740</td>
</tr>
<tr>
<td></td>
<td>DC F'4' NORMAL ARRAY INDEX INCREMENT</td>
<td>GMA 7750</td>
</tr>
<tr>
<td>INX1</td>
<td>DC F'36' INDEX LIMIT</td>
<td>GMA 7760</td>
</tr>
<tr>
<td></td>
<td>DC F'40' ARRAY INDEX</td>
<td>GMA 7770</td>
</tr>
<tr>
<td></td>
<td>DC AL4(ENDG) END OF &quot;GO&quot; LOOP</td>
<td>GMA 7780</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>DS F TEMP STORAGE</td>
<td>GMA 7790</td>
</tr>
<tr>
<td>SUM</td>
<td>DS F FOR</td>
<td>GMA 7800</td>
</tr>
<tr>
<td>LOG</td>
<td>DS F INTERMEDIATE</td>
<td>GMA 7810</td>
</tr>
<tr>
<td>UNF</td>
<td>DS F RESULTS</td>
<td>GMA 7820</td>
</tr>
<tr>
<td>X</td>
<td>DS F TRIAL GAMMA DEVIATE</td>
<td>GMA 7830</td>
</tr>
<tr>
<td>GOSAVE</td>
<td>DS 2F REGISTER STORAGE</td>
<td>GMA 7840</td>
</tr>
<tr>
<td>RNEXP</td>
<td>DS 2F ARRAY FOR EXPONENTIAL SAMPLING</td>
<td>GMA 7850</td>
</tr>
<tr>
<td>NGOI</td>
<td>DC F'2' NUMBER OF EXPONENTIALS</td>
<td>GMA 7860</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMUS</td>
<td>DS F A - 1</td>
<td>GMA 7870</td>
</tr>
<tr>
<td>GCON</td>
<td>DC V(EXPON) ADDRESS OF EXPONENTIAL GENERATOR</td>
<td>GMA 7880</td>
</tr>
<tr>
<td></td>
<td>DC V(ALOG) ADDRESS OF LOG FUNCTION</td>
<td>GMA 7890</td>
</tr>
<tr>
<td></td>
<td>DC F'4' EXPONENTIAL ARRAY INDEX INCREMENT</td>
<td>GMA 7900</td>
</tr>
<tr>
<td>INX2</td>
<td>DC F'36' EXPONENTIAL ARRAY INDEX LIMIT</td>
<td>GMA 7910</td>
</tr>
<tr>
<td>GCON</td>
<td>DC F'40' EXPONENTIAL ARRAY INDEX</td>
<td>GMA 7920</td>
</tr>
<tr>
<td></td>
<td>DS F TEMP STORAGE</td>
<td>GMA 7930</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>AINV</td>
<td>DS F 1 / A</td>
<td>GMA 7940</td>
</tr>
<tr>
<td>AMIN1</td>
<td>DS F 1 - A</td>
<td>GMA 7950</td>
</tr>
<tr>
<td>GCON</td>
<td>DC F'16807' UNIFORM MULTIPLIER</td>
<td>GMA 7960</td>
</tr>
<tr>
<td></td>
<td>DC X'40000001' EXPONENT CONSTANT</td>
<td>GMA 7970</td>
</tr>
<tr>
<td>INX3</td>
<td>DC F'4' EXPONENTIAL ARRAY INDEX INCENT</td>
<td>GMA 7980</td>
</tr>
<tr>
<td></td>
<td>DC F'36' EXPONENTIAL ARRAY INDEX LIMIT</td>
<td>GMA 7990</td>
</tr>
<tr>
<td></td>
<td>DC F'40' EXPONENTIAL ARRAY INDEX</td>
<td>GMA 8000</td>
</tr>
<tr>
<td></td>
<td>DC V(EXP) EXTERNAL FUNCTION</td>
<td>GMA 8010</td>
</tr>
<tr>
<td>UNF</td>
<td>DS F TEMPORARY STORAGE</td>
<td>GMA 8020</td>
</tr>
<tr>
<td>P</td>
<td>DS F LOCATIONS</td>
<td>GMA 8030</td>
</tr>
</tbody>
</table>
***** 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(AGQ) GMA 8130
DC X'FF' GMA 8130

ARGLST2 DC AL3(SIGA) GMA 8300
DC X'FF' GMA 8300

ARGLST3 DC AL3(CONS) GMA 8350
DC X'FF' GMA 8350
DC AL3(RNARRAY) CALL TO ALOG IN "GO" SETUP
DC X'FF' GMA 8350
DC AL3(NUM) GMA 8350

ARGLST4 DC X'FF' CALL TO ALOG IN NORMAL SECTION OF "GO"
DC AL3(LOG) GMA 8400
DC X'FF' GMA 8400

ARGLST5 DC AL3(UNIF) CALL TO ALOG IN EXPON SECTION OF "GO"
DC X'FF' GMA 8430
DC AL3(GFLOG) GMA 8430

ARGLST6 DC X'FF' CALL TO EXPONENTIAL GENERATOR IN "GO"
DC AL3(UNIF) GMA 8440
DC X'FF' GMA 8440
DC AL3(NGO1) GMA 8440

ARGLST7 DC X'FF' CALL TO ALOG IN METHOD "GF"
DC AL3(GFLOG) GMA 8470
DC X'FF' GMA 8470
DC AL3(P) FUNCTION CALLS IN METHOD "GS"

END

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
<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>No. Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defense Documentation Center</td>
<td>Cameron Station, Alexandria, Virginia 22314</td>
<td>12</td>
</tr>
<tr>
<td>Dean of Research</td>
<td>Code 023, Naval Postgraduate School, Monterey, CA 93940</td>
<td>1</td>
</tr>
<tr>
<td>Library (Code 0212)</td>
<td>Naval Postgraduate School, Monterey, CA 93940</td>
<td>2</td>
</tr>
<tr>
<td>Library (Code 55)</td>
<td>Department of Operations Research and Administrative Sciences, Naval Postgraduate School, Monterey, CA 93940</td>
<td>3</td>
</tr>
<tr>
<td>Marvin Denicoff</td>
<td>Office of Naval Research, Arlington, Virginia 22217</td>
<td>1</td>
</tr>
<tr>
<td>Dr. Thomas Varley</td>
<td>Office of Naval Research, Arlington, Virginia 22217</td>
<td>1</td>
</tr>
<tr>
<td>Dr. Bruce McDonald</td>
<td>Office of Naval Research, Arlington, Virginia 22217</td>
<td>1</td>
</tr>
<tr>
<td>Professor J. H. Ahrans</td>
<td>Nova Scotia Technical College, Department of Applied Mathematics, Halifax, Nova Scotia, Canada</td>
<td>1</td>
</tr>
<tr>
<td>Richard V. Andree</td>
<td>Department of Information and Computer Sciences, University of Oklahoma, Norman, Oklahoma 73069</td>
<td>1</td>
</tr>
<tr>
<td>Julius Aronofsky</td>
<td>University Computing Corporation, 1500 UCC Tower, P. O. Box 6228, Dallas, Texas 75222</td>
<td>1</td>
</tr>
</tbody>
</table>
Claude Cohen  
Computer Center  
Northwestern University  
2129 Sheridan Road  
Evanston, Illinois 60201

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

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

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

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

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

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

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

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

Charles E. Gates  
Institute of Statistics  
Texas A & M University  
College Station, Texas 77843
Dennis E. Grawoig
Georgia State University
33 Gilmer Street, N. E.
Atlanta, Georgia 30303

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

Robert Gumm
Computer Center
Oklahoma State University
Stillwater, Oklahoma 74074

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

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

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

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

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

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

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

Robert Morrison
Department of Math and Statistics
Oklahoma State University
Stillwater, Oklahoma 74074
Wesley L. Nicholson  
Mathematics and Physics Research  
Battelle Northwest  
P. O. Box 999  
Richmond, Washington 99352

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

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

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

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

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

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

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

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

Jack Testerman  
University of Southwestern Louisiana  
Box 940  
Lafayette, Louisiana 70501
Carolyn S. Thompson  
Medical Center  
University of Arkansas  
Little Rock, Arkansas 72201

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

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

Wray Wilkes  
Computing Center  
University of Arkansas  
Pattystown, Arkansas

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

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

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

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

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

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

John M. Chambers  
Bell Telephone Laboratories  
Murray Hill, New Jersey 97974
Patrick L. Odell
Texas Technical University
Department of Mathematics
Lubbock, Texas 79409

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

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

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

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

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

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

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

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

Prof. J. A. Galt
Department of Oceanography
Naval Postgraduate School
Monterey, CA 93940
Prof. R. E. Ball  
Department of Aeronautics  
Naval Postgraduate School  
Monterey, CA 93940  

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

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

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

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

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