SAMPLING FROM THE GAMMA DISTRIBUTION
ON A COMPUTER

George S. Fishman

Technical Report No. 75-2
May, 1975

TR-75-2

Curriculum in Operations Research
and Systems Analysis

University of North Carolina at Chapel Hill

This research was supported by the Office of Naval Research under contract
N00014-67-A-0321-0000

Reproduction in whole or in part is permitted for any purpose of the United States government.
Abstract

This paper describes a method of generating gamma variates that appears to be less costly than a recently suggested method in [3]. For large shape parameter \( \alpha \), the cost of computation is proportional to \( \sqrt{\alpha} \), whereas the method in [3] is proportional to \( \alpha \). Experimentation in [2] indicates that for small \( \alpha \), the method suggested here also dominates other methods recently suggested in [1], albeit those methods dominate for large \( \alpha \). The method suggested here uses the rejection technique.
1. Introduction

This paper describes a new technique (method 1) for sampling from the gamma distribution on a digital computer and compares it with an alternative technique (method 2) that Wallace has suggested in [3]. A gamma variate \( X \) has the probability density function (p.d.f.)

\[
f(x) = \begin{cases} \frac{x^{a-1} e^{-x/a}}{\Gamma(a)} & 0 \leq x \leq a, \quad a > 0 \\ 0 & \text{elsewhere.} \end{cases}
\]

Both methods use the rejection method and apply for \( \alpha > 1 \).

2. Rejection Method

Let \( X \) be a nonnegative valued continuous random variable with bounded p.d.f. representable in the form

\[
f(x, \alpha) = \begin{cases} c(\alpha, \beta) \alpha(\alpha, \beta) g(x, \alpha, \beta) h(x, \alpha, \beta) & 0 \leq x \leq \infty \\ 0 & \text{elsewhere} \end{cases}
\]

\[
0 \leq h(x, \alpha, \beta), \quad \int h(x, \alpha, \beta) dx = 1,
\]

\[
0 < g(x, \alpha, \beta) < \infty, \quad a(\alpha, \beta) > 1/g(x, \alpha, \beta),
\]

\[
1/c(\alpha, \beta) = a(\alpha, \beta) \int g(x, \alpha, \beta) h(x, \alpha, \beta) dx.
\]

Let \( X' \) denote a random variable with p.d.f. \( h \) and let \( U \) be a uniform deviate on \((0, 1)\). If \( U < a(\alpha, \beta)g(X', \alpha, \beta) \) then \( X' \) has the p.d.f.

\[\text{I am grateful to Mr. Hunter McDaniel for programming methods 1 and 2 in PL/1.}\]

\[\text{Here we assume a unit scale parameter without loss of generality.}\]
\( f_X \) in (2). This result follows from

\[
f_X(x|U \leq a(\alpha, \beta)g(x, \alpha, \beta)) = \frac{\text{pr}[U \leq a(\alpha, \beta)g(x, \alpha, \beta)|X' = x]h(x, \alpha, \beta)}{\text{pr}[U \leq a(\alpha, \beta)g(x, \alpha, \beta)]}
\]

(3)

\[= f_X(x, \alpha).\]

Since

(4) \( \text{pr}[U \leq a(\alpha, \beta)g(x, \alpha, \beta)] = 1/c(\alpha, \beta) \)

\( c(\alpha, \beta) \) denotes the mean number of trials to obtain an \( X \) from (2). For a given \( X' \) from a specified \( h \) we want the probability of success to be as close to unity as possible. This feature requires

(5) \( 1/a^*(\alpha, \beta) = \max_x g(x, \alpha, \beta). \)

For any \( X' \) we want (4) to be as large as possible, which implies.

\[
c(\alpha, \beta^*) = \min_{\beta} c(\alpha, \beta) = \min_{\beta} [1/a^*(\alpha, \beta)Eg(X', \alpha, \beta)]
\]

\[= \min_{\beta} \max_x [g(x, \alpha, \beta)/Eg(X', \alpha, \beta)]
\]

\[Eg(X', \alpha, \beta) = \int_0^\infty g(x, \alpha, \beta)h(x, \alpha, \beta)dx.\]

The distinction between methods 1 and 2 lies in the choice of \( h \).

Table 1 shows relevant quantities for each proposal. To make an appropriate
comparison between methods we need to consider the mean number of trials $c_j(\alpha, \beta^*)$ for each and the mean number of required random numbers.

### Table 1

#### Gamma Generation* for $\alpha \geq 1$

<table>
<thead>
<tr>
<th>Method</th>
<th>$h_i(x, \alpha, \beta)$</th>
<th>$g_i(x, \alpha, \beta)$</th>
<th>$a_i(\alpha, \beta)$</th>
<th>$\beta_i^*$</th>
<th>$c_i(\alpha, \beta^*)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\beta^{-1} e^{-x/\beta}$</td>
<td>$x^{\alpha-1} e^{(1/\beta-1)x}$</td>
<td>$(e/\alpha)^{\alpha-1}$</td>
<td>$\alpha$</td>
<td>$\alpha e^{1-\alpha}/\Gamma(\alpha)$</td>
</tr>
<tr>
<td>2</td>
<td>$x^{\gamma-1} e^{-x[(1-\beta)\gamma+8x]} \gamma(\gamma+1)$</td>
<td>$x^{\gamma'}/(1-\beta)\gamma+8x$</td>
<td>$\gamma(1-\beta)\left[\beta(1-\gamma')\right]^{\gamma'}$</td>
<td>$\gamma'$</td>
<td>$\Gamma(\gamma)\gamma^{1-\gamma'}/\Gamma(\gamma)$</td>
</tr>
<tr>
<td></td>
<td>$\gamma = \langle \theta \rangle$</td>
<td>$\gamma' = \alpha - \gamma$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* $\langle \theta \rangle$ denotes the largest integer in $\theta$.

### 3. Method 1

Conceptually, method 1 implies 4 steps:

1. Generate an exponential deviate $X'$.
2. Generate a uniform deviate $U$.
3. If $U \leq (X'/e^{X'+1})^{\alpha-1}$ then $X = \alpha X'$ has the p.d.f. in (2).
4. Otherwise, return to step 1.

If we use the inverse transform method to generate $X'$ then each trial requires 2 random numbers. Therefore, the mean number of random numbers needed to generate $X$ from (2) is $2\alpha e^{1/\Gamma(\alpha)e^{\alpha-1}}$. For large $\alpha$ this quantity is approximately $e(2\alpha/\pi)^{1/2}$, an appealing result. Notice that for large integral $\alpha$, using method 1

$^+$ An exponential deviate has unit mean.
requires fewer random numbers than the conventional method which uses

\[ X = \frac{1}{b} \prod_{i=1}^{b} (U_i), \]

\[ U_1, \ldots, U_b \text{ being a sequence of independent uniform deviates. For small integral } \alpha \text{ one can show that (7) is superior. Our experiments indicate that method one prevails for nonintegral } \alpha < 7 \text{ and all } \alpha > 7. \]

4. Method 2

For integral \( \alpha \) method 2 uses (7). For nonintegral \( \alpha \) the 6 steps are:

1. Generate a uniform deviate \( U \).
2. If \( U \leq 1 - \alpha + \alpha \) generate \( X' \) from (7) using \( b = \alpha \).
3. Otherwise, generate \( X' \) from (7) using \( b = \alpha + 1 \).
4. Generate a uniform deviate \( U \).
5. If \( U \leq (X'/\gamma)'(1-\gamma'X'/\gamma) \) then \( X' \) has the p.d.f. (2).
6. Otherwise, go to step 1.

These steps require \( \alpha + 2 \) random numbers on average per trial. Therefore, for nonintegral \( \alpha \), method 2 uses \( (\alpha + 2) \gamma(\gamma)'(1-\gamma'x'/\gamma) / \Gamma(\alpha) \) random numbers on average. This quantity is approximately \( \alpha + 2 \) for \( \alpha > 5 \).

5. Comparison of Methods

PL/1 programs were prepared using algorithm GI for method 1 and using the steps given in [3] for method 2.

Algorithm GI

Given: \( \alpha \)
1. \( \alpha' = \alpha - 1 \).
2. Generate a uniform deviate \( U \).
(continued)
3. \( V \leftarrow -\ln U. \)
4. Generate a uniform deviate \( U. \)
5. \( W \leftarrow -\ln U. \)
6. If \( W \geq \alpha (V - \ln V - 1) \), \( X \leftarrow \alpha V \) and return with \( X. \)
7. Otherwise, go to 2.

Table 2 displays the results for generation of 10,000 gamma variates for each selected value of \( \alpha. \)

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>Method 1</th>
<th>Method 2</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.25</td>
<td>723</td>
<td>1093</td>
<td>.661</td>
</tr>
<tr>
<td>2.25</td>
<td>988</td>
<td>1300</td>
<td>.760</td>
</tr>
<tr>
<td>3.25</td>
<td>1193</td>
<td>1542</td>
<td>.774</td>
</tr>
<tr>
<td>4.25</td>
<td>1352</td>
<td>1787</td>
<td>.757</td>
</tr>
<tr>
<td>5.25</td>
<td>1541</td>
<td>2039</td>
<td>.756</td>
</tr>
</tbody>
</table>

Based on these results we computed expressions for \( T_1 \), the mean CPU time for method 1, as a function of \( \alpha. \) These expressions are in microseconds.

\[
T_1 = 140 + 624\alpha / \Gamma(\alpha)e^{\alpha - 1}
\]

\[
T_2 = -88 + (752 + 254\alpha)\Gamma(\alpha)^{1-\alpha}/\Gamma(\alpha).
\]

\( \dagger \) The programs were run on the IBM 360/75 computer at the University of North Carolina Computer Center at Chapel Hill as single stream inputs. This procedure minimized the error due to monitoring in a multiprogram mode.
For large $\alpha$ $T_1/T_2 \sim 624/254\sqrt{\alpha} = 2.46/\sqrt{\alpha}$. For example, $\alpha = 30$ gives $T_1/T_2 \sim 0.45$ and $\alpha = 50$, $T_1/T_2 \sim 0.35$.

One modification to method 1 makes it at least as good as method 2 for all integral $\alpha$, while preserving its superiority for nonintegral $\alpha$. Experimentation with method 1 revealed that it is superior to method 2 for all $\alpha > 7$. Addition of the statement:

0. If $\alpha \leq 7$ and $<\alpha> = \alpha$, return with $X = -\ln(\Pi U_i)^{\alpha}$

prior to statement 1 in algorithm G1 modifies the flow appropriately.

5. New Prospects

Upon conclusion of the work presented here the writer learned of research by Dieter and Ahrens in [1] on gamma generation 1) using a truncated noncentral Cauchy distribution for $h$ and 2) exploiting the relationship between the gamma and normal distributions for large $\alpha$. The most notable feature of their work is that computation time goes to a fixed limit as $\alpha$ increases. Although this property makes the Dieter and Ahrens procedures more attractive for large $\alpha$, Robinson and Lewis [2] have recently prepared a gamma generation program in which a variant of algorithm G1 dominates all competitors for $1.2 < \alpha < 2.9$. Since this is a commonly encountered range in practice the significance of method 1 remains.

Since the work in [2] generates exponential variates by a more efficient method than inverse transformation does, it is not presently clear to the writer what the range of superiority would be using algorithm G1. This issue is a legitimate concern since simulation languages such as SIMSCRIPT and SIMPL/1 use the inverse approach.
References


This paper describes a method of generating gamma variates that appears to be less costly than a recently suggested method in [3]. For large shape parameter $\alpha$ the cost of computation is proportional to $\sqrt{\alpha}$, whereas the method in [3] is proportional to $\alpha$. Experimentation in [2] indicates that for small $\alpha$ the method suggested here also dominates methods recently suggested in [1], albeit those methods dominate for large $\alpha$. The method suggested here uses the rejection technique.