Two New Ratio-of-Uniforms Gamma Random Number Generators

by

B. Xi, K.M. Tan and C. Liu
Purdue University

Technical Report #10-03

Department of Statistics
Purdue University

October 2010

◇

Two New Ratio-of-Uniforms Gamma Random Number Generators

Bowei Xi, Kean Ming Tan, Chuanhai Liu,

Department of Statistics,
Purdue University

Abstract: Two simple algorithms to generate gamma random numbers are proposed in this article. Both algorithms use the ratio-of-uniforms method and are based on logarithmic transformations of the gamma random variable. One algorithm applies to all positive shape parameter value without limitation. It has good performance compared with other existing algorithms for both shape parameter values greater than 1 and smaller than 1. The other algorithm is limited to shape parameter smaller or equal to 1, but it has better performance compared with the first algorithm in that limited shape parameter range. Furthermore, the proposed methods will gain more efficiency if the logarithmic scale is used for the generated Gamma random numbers.

Key words and phrases: Random Number Generation, Ratio-of-Uniforms, Simulation

1 Introduction

The Gamma distribution is an important distribution. The standard gamma random numbers are used to generate random numbers from other distributions, such as chi-square, Student-t, F, and Beta distributions. A fast and simple gamma random number generator is an important component of statistical softwares. In the past many articles proposed different algorithms to generate gamma random numbers; see Devroye (1980) and Tanizaki (2008) for a comprehensive survey. However the algorithms proposed in the past apply to only limited range of shape parameter values, either shape parameter greater than 1 (e.g., [2, 8]) or smaller than 1 (e.g., [1]). Ahrens and Dieter (1974) mentioned that gamma random numbers with shape parameter greater than 1 can be generated as the sum of two independent gamma variates: gamma($n+\alpha$,1)=gamma(n,1)+gamma(α,1), where n is an integer and $\alpha < 1$. Marsaglia and Tsang (2000a) proposed an algorithm to generate gamma variates with shape parameter greater than 1. They suggested that gamma variates with shape parameter smaller than 1 can be generated as gamma(α,1)=gamma($\alpha+1$,1)×unif$(0,1)^{1/\alpha}$. However a gamma random number generator slows down significantly due to such additional step, because one more uniform random number needs to be generated.

Cheng and Feast (1980) used the ratio of uniforms method and a power transformation of the gamma random variable to generate gamma random numbers with increased shape parameter range: shape parameter greater than $1/n$, where n is an integer. Cheng and Feast (1980) provided two algorithms for $n = 2$ and $n = 4$. The ratio of uniforms method proposed by Kinderman and Monahan (1977) is based on the following result. Let $h(t)$ be a non-negative function with finite integral M_h. If we generate (u, v) uniformly over region $\{(u, v) : 0 \le u \le \sqrt{h(v/u)}\}$, then v/u has density $h(t)/M_h$. Often rejection sampling is needed to generate (u, v) uniformly over the region. When applying the ratio of uniforms method to generate gamma random numbers, $h(t)/M_h$ is the density of gamma random variable to the power of $1/n$ in Cheng and Feast (1980). The power transformation of gamma random variable is a critical step. It controls the acceptance rate when sampling from the uniform region. The transformation also allows the algorithm to be applied to increased shape parameter range [13].

In this article we propose two algorithms to generate gamma random numbers using the ratio of uniforms method. Both algorithms are based on logarithmic transformations of gamma random variable. One simple algorithm applies to all positive shape parameter value without any limitation. The second algorithm is limited to shape parameter smaller or equal to 1, but it has better performance in that range.

The remaining of the article is organized as follows. Section 2 and Section 3 presents the algorithm that applies to all shape parameter value. Section 4 presents the algorithm that applies to shape parameter smaller or equal to 1. We compare the proposed algorithms with the algorithms adopted by R and Matlab and Cheng and Feast (1980) with $n = 2$ and $n = 4$. Finally, Section 5 summarizes the timing results.

2 The Algorithm for All Positive Shape Parameter

If a random variable X follows a gamma distribution with scale parameter equal to 1 and shape parameter α, $X \sim gamma(\alpha, 1)$, then $\beta X \sim gamma(\alpha, \beta)$. Hence the proposed random number generators focus on gamma distributions with scale parameter equal to 1. Assume $X \sim gamma(\alpha, 1)$ with density $f(x) = x^{\alpha-1}e^{-x}/\Gamma(\alpha)$. Then $E(X) = V(X) = \alpha$. We consider the transformed random variable $T = \sqrt{\alpha}\ln(\frac{X}{\alpha})$, which has density function $g(t) = \frac{1}{\Gamma(\alpha)}\alpha^{\alpha-\frac{1}{2}}e^{\sqrt{\alpha}t-\alpha e^{\frac{t}{\sqrt{\alpha}}}}$. Applying Taylor expansion with respect to $\frac{X}{\alpha}$, we have $E(T) \simeq 0$ and $V(T) \simeq 1$ for large α. For large α, the distribution of T is approximately normal with mean 0 and unit variance. It will be seen that this transformation also works well for small α. The proposed algorithm will first generate random numbers of T, and then compute $X = \alpha e^{\frac{T}{\sqrt{\alpha}}}$. We note that when random numbers on the logarithmic scale is preferred, additional CPU time can be saved by our proposed generators.

Using the ratio of uniforms method to generate random numbers of T, we simulate (U, V) uniformly over the region

$$\mathcal{C} = \{(u, v) : 0 \le u \le \sqrt{h(t)}, -\infty < t = \frac{v}{u} < \infty\},$$

where $h(t) = e^{\sqrt{\alpha}t-\alpha e^{\frac{t}{\sqrt{\alpha}}}+\alpha}$. Then $T = \frac{V}{U}$ has the desired density $g(t)$. The size of region $\mathcal{C}$ is

$$|\mathcal{C}| = \frac{\Gamma(\alpha)e^{\alpha}}{2\alpha^{\alpha-\frac{1}{2}}}. \tag{2.1}$$

In order to simulate (U, V) uniformly over region $\mathcal{C}$, we use the upper and lower bounds of u and v to define a rectangular region that covers $\mathcal{C}$ and apply the accept-reject method. Since $\max\sqrt{h(t)} = 1$, the range of u can be easily defined: $0 \le u \le 1$.

Note that $v \le t\sqrt{h(t)}$ if $t > 0$ and $v \ge t\sqrt{h(t)}$ if $t < 0$. Let $v_{\max}(\alpha) = \max_{\,t>0}\ t\sqrt{h_\alpha(t)}$ and $v_{\min}(\alpha) = \min_{\,t<0}\ t\sqrt{h_\alpha(t)}$. The properties of $v_{\max}$ and $v_{\min}$ as functions of α, or equivalently $\ln\alpha$ ($\alpha > 0$), help us find an upper bound and a lower bound of v. These properties are summarized into the following two theorems with their proofs given in Appendix A.

Theorem 1 (Properties of the log of $v_{\max}$). *Let $\theta = \ln\alpha$ and let $s(\theta) = \max_{t>0}\ln(t\sqrt{h_\alpha(t)})$. Then, as a function of $\theta \in \mathcal{R}$, $s(\theta)$ is strictly increasing and concave. Furthermore,*

$$\lim_{\theta\to-\infty}\frac{s(\theta)}{\theta} = \frac{1}{2} \tag{2.2}$$

and

$$\lim_{\theta\to\infty} s(\theta) = \ln\sqrt{\frac{2}{e}}. \tag{2.3}$$

Theorem 2 (Properties of the log of the absolute value of $v_{\min}$). *Let $w(\theta) = \max_{t<0}\ln(|t|\sqrt{h_\alpha(t)})$ and $\theta = \ln\alpha$. Then, as a function of $\theta \in \mathcal{R}$, $w(\theta)$ is strictly decreasing and convex. Furthermore,*

$$\lim_{\theta\to-\infty}\frac{w(\theta)}{\theta} = -\frac{1}{2} \tag{2.4}$$

and

$$\lim_{\theta\to\infty} w(\theta) = \ln\sqrt{\frac{2}{e}}. \tag{2.5}$$

Note that

$$v_{\max}(\alpha) = e^{s(\theta)} \tag{2.6}$$

and

$$v_{\min}(\alpha) = (-1) \times e^{w(\theta)} \tag{2.7}$$

are monotone functions of $\theta = \ln \alpha$. If we obtain upper bounds of $s(\theta)$ and $w(\theta)$, we have the upper and lower bounds of v. Since $s(\theta)$ is strictly increasing and concave, we can use its tangent lines to construct an upper bound. Since $w(\theta)$ is strictly decreasing and convex, we can select a set of points on the curve $w(\theta)$ and use the intervening line segments as an upper bound. The two curves $s(\theta)$ and $w(\theta)$ are shown in Figure 2.1.

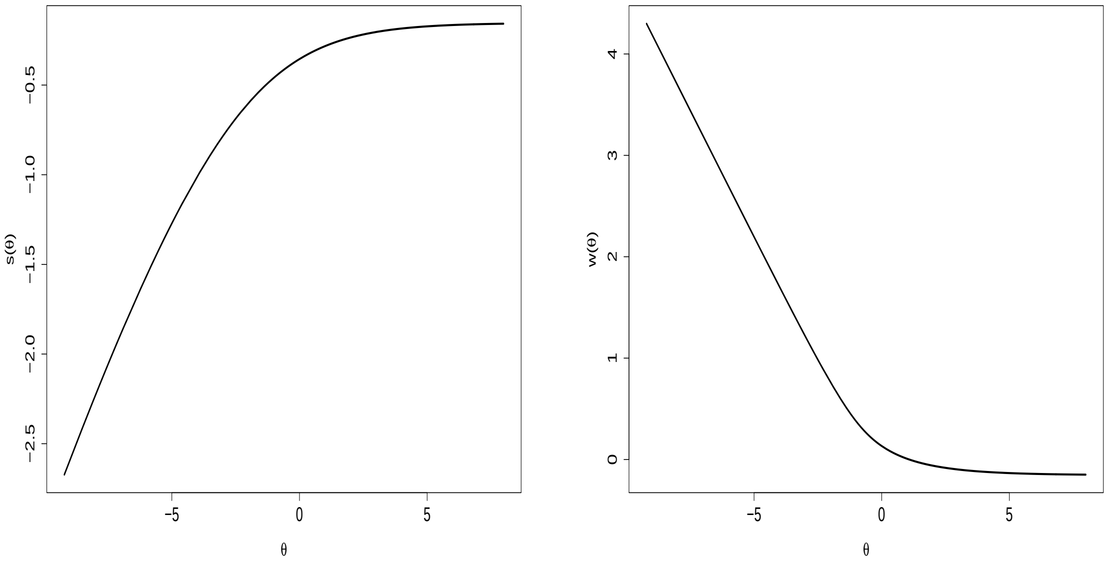

Figure 2.1: Left: $s(\theta)$ against θ; Right: $w(\theta)$ against θ.

3 Upper and Lower Bounds of Ratio of Uniforms Region

Let $b_s(\theta)$ be an upper bound of $s(\theta)$ and $b_w(\theta)$ be an upper bound of $w(\theta)$ such that $s(\theta) \leq b_s(\theta) \leq s(\theta)+\delta$ and $w(\theta) \leq b_w(\theta) \leq w(\theta) + \delta$, where constant $\delta > 0$. Then $-e^{b_w(\theta)}$ is a lower bound of $v_{\min}(\alpha)$ and $e^{b_s(\theta)}$ is an upper bound of $v_{\max}(\alpha)$. Both of the following rectangles cover region $\mathcal{C}$:

$$\mathcal{B}_1 = \{(u, v) : 0 \leq u \leq 1, -e^{b_w(\theta)} \leq v \leq e^{b_s(\theta)}\}$$

and

$$\mathcal{B}_2 = \{(u, v) : 0 \leq u \leq 1, -e^{w(\theta)} \leq v \leq e^{s(\theta)}\}.$$

Define the rate $r(\theta)$ as the following:

$$r(\theta) = \frac{|\mathcal{B}_2|}{|\mathcal{B}_1|} = \frac{e^{s(\theta)} + e^{w(\theta)}}{e^{b_s(\theta)} + e^{b_w(\theta)}}. \tag{3.1}$$

Hence the rate $r(\theta) \geq \frac{e^{s(\theta)}+e^{w(\theta)}}{(e^{s(\theta)}+e^{w(\theta)})e^{\delta}} = e^{-\delta}$. Given a pre-specified minimum rate R such as 0.99 or 0.90, we let $\delta = -\ln R$ to construct $b_s(\theta)$ and $b_w(\theta)$.

3.1 The Lower Bound

There is a simple and tight lower bound of v for small shape parameter α. Since $\ln x \leq x - 1$, $\forall t < 0$ we have

$$v \geq t\sqrt{h(t)} = te^{\frac{\sqrt{\alpha}}{2}t - \frac{\alpha}{2}e^{t/\sqrt{\alpha}} + \frac{\alpha}{2}} \geq te^{\frac{\sqrt{\alpha}}{2}t + \frac{\alpha}{2}} \geq te^{\frac{\alpha}{2}}\frac{2}{e\sqrt{\alpha}|t|}.$$

Hence a tight lower bound for small values of shape parameter α is given by

$$v \geq -\frac{2}{\sqrt{\alpha}}e^{\frac{\alpha}{2}-1}. \tag{3.2}$$

However because of the factor $e^{\frac{\alpha}{2}-1}$, this lower bound will become powerless for large α. For larger α we switch to a lower bound based on the intervening line segments connecting the points on the curve $w(\theta)$.

Note $\theta = \ln \alpha$. Let

$$k(\theta) = \ln\left(\frac{2}{\sqrt{\alpha}}e^{\frac{\alpha}{2}-1}\right) = (\ln 2 - 1) - \frac{\theta}{2} + \frac{e^{\theta}}{2}. \tag{3.3}$$

Meanwhile Equations (5.7) and (5.8) are used to construct $b_w(\theta)$:

$$w(\theta) = \frac{e^{\theta}}{2} + \frac{\theta}{2} + \ln y - \frac{e^{\theta}}{2}(y + e^{-y}) = \frac{1 - y - e^{-y}}{y(1 - e^{-y})} + \frac{\ln(2y)}{2} - \frac{\ln(1 - e^{-y})}{2},$$

$$e^{\theta} = \frac{2}{y(1 - e^{-y})}.$$

The algorithm to construct $b_w(\theta)$ is the following:

Step 1. Solve $k(\theta) - w(\theta) = \delta$ for θ^*. This is equivalently to solve the following equation for y^*:

$$\ln(1 - e^{-y}) - 1 + \frac{y + e^{-y}}{y(1 - e^{-y})} = \delta.$$

Obtain $\theta^* = \theta^*(y^*)$ using Equation 5.8. There exists a positive integer M such that $\ln\sqrt{\frac{2}{e}} + M\delta \leq w(\theta^*) \leq \ln\sqrt{\frac{2}{e}} + (M+1)\delta$. Let $\theta^L_{M+1} = \theta^*$ and $y^L_{M+1} = y^*$.

Step 2. Solve these M equations for θ^L_k: $w(\theta^L_k) = \ln\sqrt{\frac{2}{e}} + k\delta$, $k = 1, ..., M$. This is equivalent to solve the following equations for y^L_k and obtain $\theta^L_k = \theta^L_k(y^L_k)$ using Equation 5.8:

$$\frac{1 - y - e^{-y}}{y(1 - e^{-y})} + \frac{\ln(2y)}{2} - \frac{\ln(1 - e^{-y})}{2} = \ln\sqrt{\frac{2}{e}} + k\delta.$$

The function $b_w(\theta)$ has the following structure:

1. $\forall\ \theta \geq \theta_1^L$, $b_w(\theta) = \ln\sqrt{\frac{2}{e}} + \delta$;

2. $\forall\ \theta_{k+1}^L \leq \theta < \theta_k^L$ $(k = 1, ..., M)$, $b_w(\theta) = \frac{w(\theta_{k+1}^L)-w(\theta_k^L)}{\theta_{k+1}^L-\theta_k^L} \times \theta + \frac{\theta_k^L w(\theta_{k+1}^L)-\theta_{k+1}^L w(\theta_k^L)}{\theta_k^L-\theta_{k+1}^L}$;

3. $\forall\ \theta < \theta_{M+1}^L$, $b_w(\theta) = k(\theta) = \ln(\frac{2}{\sqrt{\alpha}} e^{\frac{\alpha}{2}-1})$.

A lower bound of $v_{\min}(\alpha)$ is $B_{\min}(\alpha) = -e^{b_w(\theta)}$. It is straightforward to show that $k(\theta) - w(\theta)$ is strictly increasing since $k'(\theta) - w'(\theta) > 0$. Then there is a unique solution for the equation in Step 1 of the algorithm. Because $w(\theta)$ is strictly decreasing, the equations in Step 2 have unique solutions. The distance between $b_w(\theta)$ and $w(\theta)$ is smaller or equal to $w(\theta_k^L) - w(\theta_{k+1}^L) = k(\theta_{M+1}^L) - w(\theta_{M+1}^L) = \delta$, $k = 1, ..., M$. $b_w(\theta)$ constructed from the above algorithm has the desired property: $w(\theta) \leq b_w(\theta) \leq w(\theta) + \delta$.

3.2 The Upper Bound

We use the tangent lines of $s(\theta)$ to form a piece-wise linear function $b_s(\theta)$ that bounds $s(\theta)$. Using Equations 5.1, 5.2, and 5.3, we re-write $s(\theta)$, $s'(\theta)$, and θ as functions of y:

$$S(y) = s(\theta) = \frac{1 + y - e^y}{y(e^y - 1)} + \frac{\ln(2y)}{2} - \frac{\ln(e^y - 1)}{2} \tag{3.4}$$

$$P(y) = s'(\theta) = \frac{1 + y - e^y}{y(e^y - 1)} + \frac{1}{2} \tag{3.5}$$

$$T(y) = \theta = \ln 2 - \ln y - \ln(e^y - 1) \tag{3.6}$$

Note $\alpha(\theta) = e^\theta$. Let

$$c(\theta) = \frac{e^{w(\theta)}}{e^{s(\theta)} + e^{k(\theta)}} = \frac{e^{w(\theta)}}{e^{s(\theta)} + \frac{2}{\sqrt{\alpha(\theta)}} e^{\frac{\alpha(\theta)}{2}-1}} \tag{3.7}$$

The algorithm to construct $b_s(\theta)$ is the following:

Step 1. Use the constant $\ln\sqrt{\frac{2}{e}}$ to bound the largest θs. Solve $s(\theta) = \ln\sqrt{\frac{2}{e}} - \delta$ for θ_1^U, where θ_1^U is the first change point of the piece-wise linear function $b_s(\theta)$. This is equivalent to solve

$$S(y) = \ln\sqrt{\frac{2}{e}} - \delta$$

for y_1^U and obtain θ_1^U using Equation 3.6.

Step 2. Find a tangent line of $s(\theta)$ that passes through the point $(\theta_1^U, \ln\sqrt{\frac{2}{e}})$. Solve $s'(\theta) \times \theta_1^U + s(\theta) - s'(\theta)\theta = s(\theta_1^U) + \delta$ for θ_1^T. The tangent line at θ_1^T is used to form $b_s(\theta)$. This is equivalent to solve

$$P(y) \times \theta_1^U + S(y) - P(y) \times T(y) = \ln\sqrt{\frac{2}{e}}$$

for y_1^T $(y_1^T > y_1^U)$ and obtain θ_1^T $(\theta_1^T < \theta_1^U)$ using Equation 3.6. If $\theta_1^T < \theta_{M+1}^L$, compute $c(\theta_1^T)$. If $c(\theta_1^T) \geq R$, stop; otherwise proceed to Step 3.

Step 3. (1) Given y_k^T and θ_k^T, find the next change point θ_{k+1}^U ($k \geq 1$) where the distance between the tangent line at θ_k^T and $s(\theta)$ reaches δ. Solve $s'(\theta_k^T) \times \theta + s(\theta_k^T) - s'(\theta_k^T)\theta_k^T = s(\theta) + \delta$ for θ_{k+1}^U. This is equivalent to solve

$$P(y_k^T) \times T(y) + S(y_k^T) - P(y_k^T) \times T(y_k^T) = S(y) + \delta$$

for y_{k+1}^U ($y_{k+1}^U > y_k^T$) and obtain θ_{k+1}^U ($\theta_{k+1}^U < \theta_k^T$) using Equation 3.6.

(2) Given y_{k+1}^U and θ_{k+1}^U, find the next tangent line that passes through the point $(\theta_{k+1}^U, s(\theta_{k+1}^U) + \delta)$. Solve $s'(\theta) \times \theta_{k+1}^U + s(\theta) - s'(\theta)\theta = s(\theta_{k+1}^U) + \delta$ for θ_{k+1}^T. Equivalently, we solve

$$P(y) \times \theta_{k+1}^U + S(y) - P(y) \times T(y) = S(y_{k+1}^U) + \delta$$

for y_{k+1}^T ($y_{k+1}^T > y_{k+1}^U$) and obtain θ_{k+1}^T ($\theta_{k+1}^T < \theta_{k+1}^U$) using Equation 3.6. If $\theta_{k+1}^T < \theta_{M+1}^L$, compute $c(\theta_{k+1}^T)$. If $c(\theta_{k+1}^T) \geq R$, stop; otherwise let $k = k + 1$ and repeat Step 3.

Assume that the above algorithm produces N tangent lines to form $b_s(\theta)$, each connecting with $s(\theta)$ at θ_k^T ($k = 1, ..., N$) where $\theta_{k+1}^T < \theta_k^T$. The function $b_s(\theta)$ is piece-wise linear with N change points θ_k^U, $k = 1, ..., N$. Note that $\theta_{k+1}^U < \theta_k^U$. Let $\theta_{N+1}^U = -\infty$. Then $b_s(\theta)$ has the following structure:

1. $\forall\ \theta > \theta_1^U$, $b_s(\theta) = \ln \sqrt{\frac{2}{e}}$;
2. $\forall\ \theta_{k+1}^U < \theta \leq \theta_k^U$ ($k = 1, ..., N$), $b_s(\theta) = s'(\theta_k^T) \times \theta + s(\theta_k^T) - s'(\theta_k^T)\theta_k^T$.

An upper bound of $v_{\max}(\alpha)$ is $B_{\max}(\alpha) = e^{b_s(\theta)}$. Given a θ_0, to compute the value of $w(\theta_0)$ in $c(\theta_0)$, we use Equation 5.8 to obtain the corresponding y_0 and plug both y_0 and θ_0 into Equation 5.7. Since $s(\theta)$ is strictly increasing, the equation in Step 1 has a unique solution. Write the tangent line at θ^T as $\ell(\theta) = s'(\theta^T) \times \theta + s(\theta^T) - s'(\theta^T)\theta^T$. The distance between $\ell(\theta)$ and $s(\theta)$ is strictly increasing $\forall\ \theta > \theta^T$ and strictly decreasing $\forall\ \theta < \theta^T$. In addition $s(\theta)$ is strictly concave. Hence the equations in Step 2 and Step 3 have unique solutions. For all $\theta \geq \theta_N^T$, $b_s(\theta) - s(\theta)$ reaches the maximum value δ at the change points θ_k^U. For all $\theta < \theta_N^T$, $b_s(\theta) - s(\theta)$ might be greater than δ but the rate $r(\theta)$ is still greater than R: Since $k(\theta) - w(\theta)$ is strictly increasing, it is straightforward to derive that $c(\theta)$ is strictly decreasing. Let $\ell_N(\theta)$ be the tangent line at θ_N^T. Define

$$c^*(\theta) = \frac{e^{w(\theta)}}{e^{\ell_N(\theta)} + \frac{2}{\sqrt{\alpha(\theta)}} e^{\frac{\alpha(\theta)}{2} - 1}}.$$

$c^*(\theta)$ is also strictly decreasing. Then $\forall\ \theta < \theta_N^T$,

$$r(\theta) = \frac{e^{w(\theta)} + e^{s(\theta)}}{e^{b_s(\theta)} + e^{b_w(\theta)}} = \frac{e^{w(\theta)} + e^{s(\theta)}}{e^{\ell_N(\theta)} + e^{k(\theta)}} \geq c^*(\theta) \geq c^*(\theta_N^T) = c(\theta_N^T) \geq R.$$

3.3 Acceptance Rate

We took $R = 0.9$ and implemented the algorithms in Sections 3.1 and 3.2 to obtain $b_w(\theta)$ and $b_s(\theta)$. The precision of the parameter values is 10^{-15}. Note that $\delta = -\ln R$. The resultant functions $b_w(\theta)$ and $b_s(\theta)$ are shown in Tables 3.1 and 3.2. These two functions have only 3 and 4 segments. We note that more segments are needed for larger R. For example, for $R = 0.95$ $b_w(\theta)$ needs 7 segments and $b_s(\theta)$ has 4 segments. However more segments will increase the initial parameter set-up time. Hence we use the following functions with $R = 0.9$.

Range of θ	$b_w(\theta)$
$\theta \geq 1.764216686288215$	$\ln\sqrt{\frac{2}{e}} - \ln R = -0.048065894062201$
$0.521223243207446 \leq \theta < 1.764216686288215$	$-0.084763530978316 \times \theta + 0.101475344169199$
$0.209314923020777 \leq \theta < 0.521223243207446$	$-0.135460234584798 \times \theta + 0.127899644442896$
$\theta < 0.209314923020777$	$\ln\frac{2}{e} - \frac{\theta}{2} + \frac{e^\theta}{2} = -0.306852819440055 - \frac{\theta}{2} + \frac{\alpha}{2}$

Table 3.1: The function $b_w(\theta)$ based on $R = 0.9$. $B_{\min}(\alpha) = -e^{b_w(\theta)}$ is a lower bound of v.

Range of θ	$b_s(\theta)$
$\theta > 1.448931546292675$	$\ln\sqrt{\frac{2}{e}} = -0.153426409720027$
$-3.333189906461192 < \theta \leq 1.448931546292675$	$0.124651796958072 \times \theta - 0.334038330634647$
$\theta \leq -3.333189906461192$	$0.306252995504409 \times \theta + 0.271272951361260$

Table 3.2: The function $b_s(\theta)$ based on $R = 0.9$. $B_{\max}(\alpha) = e^{b_s(\theta)}$ is an upper bound of v.

Both the exact (or best) acceptance rate based on the exact lower and upper bounds and the (actual) acceptance rate based on the functions in Tables 3.1 and 3.2 are shown in Figure 3.1. The uniform region and its cover for various αs are shown in Figure 3.2. Using the bounds $b_w(\theta)$ and $b_s(\theta)$ in Tables 3.1 and 3.2, Algorithm 3.1 demonstrates how to generate Gamma random numbers using the proposed approach. When $\alpha < 0.01$, the algorithm returns the generated random numbers on logarithmic scale.

4 The Algorithm for Small Shape Parameter

We consider another transformation similar to the one used in Algorithm 3.1 for $\alpha \leq 1$: $T^* = \alpha \ln X$ and $X = e^{T^*/\alpha}$. Its density function is $g^*(t) = e^{t - e^{t/\alpha}}/(\alpha\Gamma(\alpha))$. Although the ratio of uniforms algorithm based on T^* is limited to $\alpha \leq 1$, it is faster than Algorithm 3.1 for small shape parameter values. Let $h^*(t) = e^{t - e^{t/\alpha}}$. The uniform region is

$$\mathcal{C}^* = \{(u, v) : 0 \leq u \leq \sqrt{h^*(t)}, -\infty < t = \frac{v}{u} < \infty\}.$$

The size of region $\mathcal{C}^*$ is

$$|\mathcal{C}^*| = \frac{\Gamma(\alpha)\alpha}{2}. \tag{4.1}$$

Because $\max\sqrt{h^*(t)} = (\frac{\alpha}{e})^{\alpha/2}$, we have

$$0 \leq u \leq \left(\frac{\alpha}{e}\right)^{\alpha/2}. \tag{4.2}$$

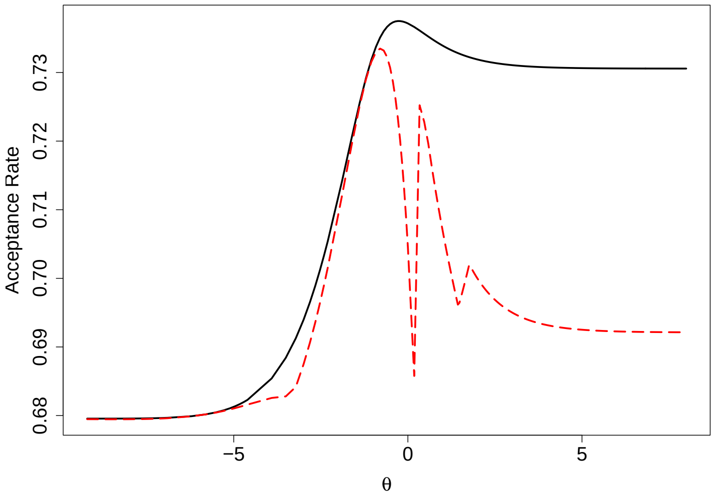

Figure 3.1: Algorithm 3.1: Maximum acceptance rate (solid line) and acceptance rate using the lower and upper bounds (dashed line) specified by Tables 3.1 and 3.2.

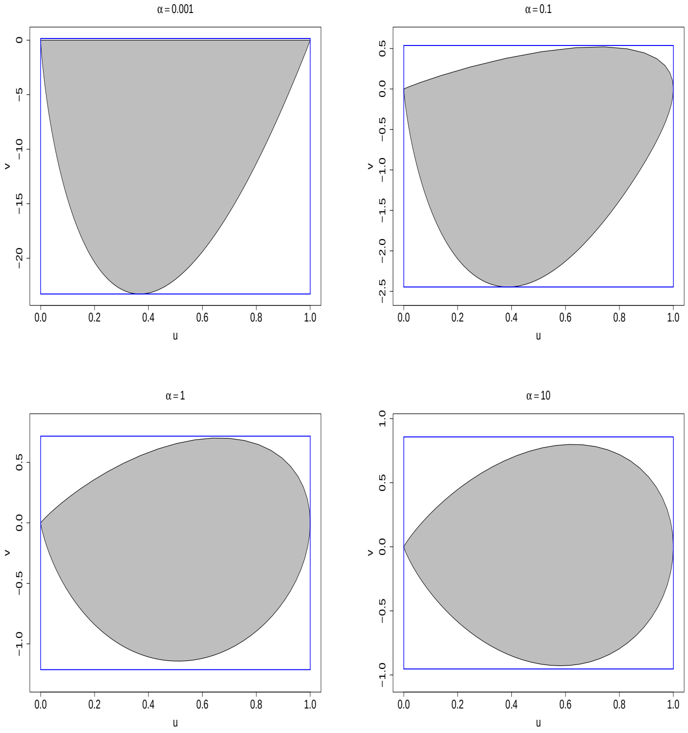

Figure 3.2: Region $\mathcal{C}$ and the upper and lower bounds for $\alpha = 0.001$, 0.1, 1, and 10.

Using the inequality $\ln x \le x - 1$, we have $\forall t > 0$

$$t\sqrt{h^*(t)} = te^{\frac{t}{2}-\frac{1}{2}e^{t/\alpha}} \le te^{\frac{t}{2}-\frac{et}{2\alpha}} \le \frac{2\alpha}{e(e-\alpha)}.$$

Algorithm 3.1 Generate n Gamma Random Numbers

```
$\theta = \ln(\alpha)$ and $c = \sqrt{\alpha}$
Compute $b_s(\theta)$ and B.max = $\exp\{b_s(\theta)\}$
Compute $b_w(\theta)$ and B.min = $-\exp\{b_w(\theta)\}$
$k = 1$
while $k \leq n$ do
  $u$ = Uniform(0, 1)
  $v$ = Uniform(0, 1)×(B.max−B.min)+B.min
  $t = v/u$ and $t_1 = e^{t/c+\theta}$
  if $2 \times \ln(u) \leq \alpha + c \times t - t_1$ then
    Deliver $t_1$ if $\alpha \geq 0.01$; Otherwise deliver $t/c + \theta$.
    $k = k + 1$
  end if
end while
```

For all $t < 0$, we have

$$t\sqrt{h^*(t)} \;\geq\; te^{\frac{t}{2}} \;\geq\; -\frac{2}{e}.$$

Therefore we have the range for v:

$$-\frac{2}{e} \leq v \leq \frac{2\alpha}{e(e-\alpha)}. \tag{4.3}$$

Equations 4.2 and 4.3 define a rectangle that covers region $\mathcal{C}^*$. The uniform region $\mathcal{C}^*$ and its cover are similar to the ones shown in Figure 3.2. We have a simple algorithm (Algorithm 4.1) to generate Gamma random numbers for $\alpha \leq 1$. Again when $\alpha < 0.01$ the algorithm returns the generated random numbers on logarithmic scale. The resulting acceptance rate is shown in Figure 4.1. Algorithm 4.1 reaches maximum acceptance rate 0.7554 when $\alpha = 0.33$.

Algorithm 4.1 Generate n Gamma Random Numbers given $\alpha \leq 1$

```
u.max=$(\alpha/e)^{\alpha/2}$, v.min=$-2/e$, v.max=$2\alpha/e/(e-\alpha)$.
$k = 1$
while $k \leq n$ do
  $u$ = u.max×Uniform(0, 1)
  $t$ = (Uniform(0, 1)×(v.max−v.min)+v.min)/$u$
  $t_1 = e^{t/\alpha}$
  if $2 \times \ln(u) \leq t - t_1$ then
    Deliver $t_1$ if $\alpha \geq 0.01$; Otherwise deliver $t/\alpha$.
    $k = k + 1$
  end if
end while
```

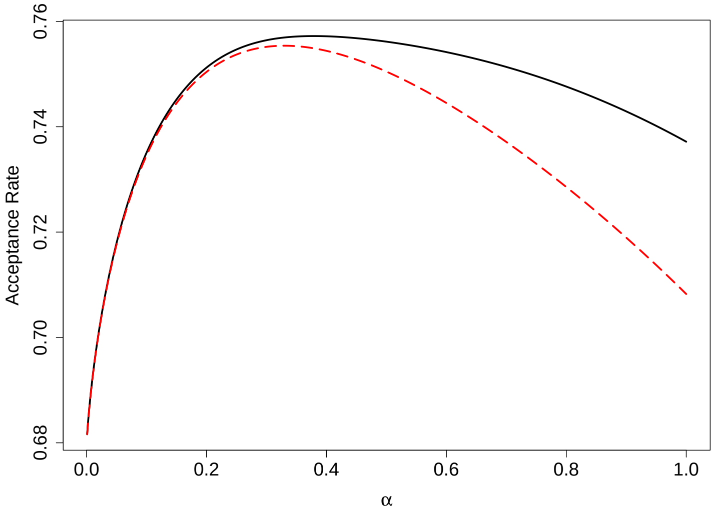

Figure 4.1: Algorithm 4.1: Maximum acceptance rate (solid line) and acceptance rate using the lower and upper bounds (dashed line) specified by Equations 4.2 and 4.3.

5 Summary: Timing Results

In this simulation study we generate 50 million Gamma random numbers for each of a sequence of selected shape parameter values. We compare our algorithms with the following algorithms.

- CF80GT – Cheng and Feast (1980) with $n = 2$.
- CF80GBH – Cheng and Feast (1980) with $n = 4$.
- AD82 ($\alpha \geq 1$) – Ahrens and Dieter (1982).
- AD74 ($\alpha < 1$) – Ahrens and Dieter (1974).
- MT00 – Marsaglia and Tsang (2000a). Gamma random numbers for $\alpha < 1$ are generated as gamma$(\alpha, 1)$=gamma$(\alpha + 1, 1)$×unif(0,1)$^{1/\alpha}$.
- ROU1 – Algorithm 3.1.
- ROU2 – Algorithm 4.1.

The algorithms proposed by Ahrens and Dieter (1982) and Ahrens and Dieter (1974) are adopted by R, an open source statistical computing software. The algorithm proposed by Marsaglia and Tsang (2000a) is adopted by Matlab. In our simulation study, the uniform random numbers were generated by the Mersenne Twister algorithm, which is one of the algorithms adopted by R and Matlab. The Mersenne Twister algorithm has a period $2^{19937} - 1$ and is a popular high quality and efficient uniform random number generator. The algorithms proposed by Ahrens and Dieter (1982) and Marsaglia and Tsang (2000a) require normal and exponential random numbers to achieve high acceptance rates. The Ziggurat algorithm, the algorithm adopted by Matlab, was used to generate normal and exponential random numbers. We used an independent random integer to select the rectangle in the Ziggurat algorithm as recommended by Thomas et. al. (2007). The computer used in the simulation is a 2.4 GHz Intel Core 2 Duo processor MacBook. The computer runs Mac Os/X Snow Leopard with 3MB L2 cache, 4GB memory, and 800MHz frontside bus. All the algorithms were coded in C++. The GNU g++ compiler was used to obtain the timing results.

α	0.01	0.25	0.5	0.8	1	1.25	3	5	10	100
CF80GT			23.94	18.80	18.05	17.56	16.66	16.46	16.34	16.24
CF80GBH		22.17	17.79	16.93	16.76	16.67	16.60	16.67	16.71	16.81
MT00	20.39	20.31	20.25	20.20	13.38	13.29	13.09	13.04	13.02	12.95
AD82					16.51	16.00	15.95	11.36	10.98	10.51
AD74	9.15	12.55	13.68	19.32	10.96					
ROU1	15.44	14.86	14.48	14.07	13.71	13.31	12.90	14.28	14.31	14.42
ROU2	14.50	12.54	12.61	13.44	13.42					

Table 5.1: Timing results (in seconds) with parameter set-up installed

α	0.01	0.25	0.5	0.8	1	1.25	3	5	10	100
CF80GT			28.68	22.61	18.83	21.16	20.08	19.88	19.69	19.65
CF80GBH		25.68	19.40	19.72	18.24	18.56	19.42	19.41	19.51	19.60
MT00	20.64	20.56	20.51	20.42	13.72	13.64	13.40	13.36	13.29	13.25
AD82					16.56	16.06	16.03	11.46	11.08	10.59
AD74	9.23	12.87	13.98	19.67	11.08					
ROU1	22.73	23.26	22.45	21.94	21.60	20.63	20.00	20.00	18.74	18.92
ROU2	19.32	16.97	17.05	18.03	18.17					

Table 5.2: Timing results (in seconds) without parameter set-up installed

Table 5.2 presents the timing results where the parameters are computed for every draw of random number. Table 5.1 presents the timing results where the parameters are set up initially and installed for repeated draws of random numbers. MT00, AD82, and AD74 need little extra time for the initial parameter set-up step. ROU1 and ROU2 slow down if the parameters are computed for every draw as shown in Table 5.2. When the parameters are installed for repeated draws, ROU1 and ROU2 have quite good performance as shown in Table 5.1. ROU1 and ROU2 are faster than CF80GT and CF80GBH. Both are faster than MT00 for $\alpha < 1$. ROU1 has high acceptance rate for α between 1.25 and 3, and is comparable to MT00 in that range. AD82 reaches high acceptance rate for $\alpha \geq 4$. ROU1 is faster than AD82 for $\alpha < 4$. AD74 has the lowest acceptance rate at around $\alpha = 0.8$. ROU2 is faster than AD74 for α between 0.25 and 0.8.

Appendix A: The proofs of Theorems 1 and 2

A.1 The proof of Theorem 1

Let $y = t/\sqrt{\alpha}$ and write

$$k(y, \alpha) = 2\ln(t\sqrt{h_\alpha(t)}) = \alpha + \ln\alpha + 2\ln y + \alpha y - \alpha e^y \qquad (y > 0)$$

Then

$$s(\theta) = \frac{1}{2}\max_{y>0} k(y, \alpha)$$

where $\theta = \ln\alpha \in \mathcal{R}$. It follows from

$$\frac{\partial 2k(y,\alpha)}{\partial y} = \frac{2}{y} + \alpha(1 - e^y) \qquad (y > 0; \alpha > 0)$$

that the function $s(\theta)$ is given by

$$2s(\theta) = e^\theta + \theta + 2\ln y + e^\theta(y - e^y) \qquad (\theta \in \mathcal{R}) \tag{5.1}$$

where y is a function of θ defined implicitly by

$$e^\theta = \frac{2}{y(e^y - 1)} \tag{5.2}$$

a one-to-one mapping: $\theta \mapsto y$ from $\mathcal{R}$ to $\{y : y > 0\}$.

Routine algebraic operations lead to the first-order derivative of $2s(\theta)$ with respect to θ:

$$\begin{aligned}\frac{\partial 2s(\theta)}{\partial\theta} &= 1 + e^\theta(1 + y - e^y) + \left[\frac{2}{y} + e^\theta(1 - e^y)\right]\frac{1}{\frac{\partial\theta}{\partial y}} \\ &\overset{(5.2)}{=} 1 + e^\theta(1 + y - e^y),\end{aligned} \tag{5.3}$$

where

$$\frac{\partial\theta}{\partial y} = -\frac{1}{y} - \frac{e^y}{e^y - 1}. \tag{5.4}$$

To show that (5.3) is positive for all $y > 0$ (or all $\theta \in \mathcal{R}$), write (5.3) in terms of y based on (5.2)

$$1 - \frac{2}{y} + \frac{2}{e^y - 1} = \frac{ye^y - 2e^y + 2 + y}{y(e^y - 1)} \tag{5.5}$$

The numerator of (5.5) has the first-order derivative

$$ye^y - e^y + 1$$

which is strictly increasing in y because its first-order derivative ye^y is positive for all $y > 0$. Note that The numerator of (5.5) tends to zero as y goes to zero. This implies that the numerator of (5.5) is positive for all y. The denominator of (5.5) is also positive for all $y > 0$, due to the fact $e^y > 1$. As a result, (5.5) or equivalently (5.3) is positive for all $y > 0$. That is, $2s(\theta)$ and, thereby, $s(\theta)$ is increasing.

The concavity of $s(\theta)$ or $2s(\theta)$ is to be proved by showing that its second-order derivative or the first-order derivative of (5.3) or (5.5) with respect to θ is negative. The first-order derivative of (5.5) with respect to θ is given by

$$2\left[\frac{1}{y^2} - \frac{e^y}{(e^y-1)^2}\right]\frac{1}{\frac{\partial\theta}{\partial y}}$$

Since $\frac{\partial\theta}{\partial y} < 0$, the concavity result follows if the bracketed term in the above expression is positive. Note that the bracketed term

$$\frac{1}{y^2} - \frac{e^y}{(e^y-1)^2} = \left[\frac{1}{y} + \frac{e^{y/2}}{(e^y-1)}\right]\left[\frac{1}{y} - \frac{e^{y/2}}{(e^y-1)}\right]$$

has the same sign as the second factor

$$\frac{1}{y} - \frac{e^{y/2}}{(e^y-1)} = \frac{e^y - 1 - ye^{y/2}}{y(e^y-1)}. \tag{5.6}$$

The numerator of (5.6) is positive because (i) its first-order derivative,

$$e^{y/2}\left[e^{y/2} - 1 - y/2\right],$$

is positive for all $y > 0$, and (ii) it approaches to zero as $y \to 0$. Note again that $y(e^y - 1)$, the denominator of (5.6), is positive. Thus, the function $s(\theta)$ is strictly concave.

The identities (2.3) and (2.2) can be easily established by applying standard algebraic operations to (5.1) and (5.2).

A.2 The proof of Theorem 2

The proof is similar to that of Theorem 1. Let $y = -t/\sqrt{\alpha}$ and write

$$k(y,\alpha) = 2\ln(|t|\sqrt{h_\alpha(t)}) = \alpha + \ln\alpha + 2\ln y - \alpha y - \alpha e^{-y} \qquad (y > 0)$$

Then

$$w(\theta) = \frac{1}{2}\max_{y>0} k(y,\alpha)$$

where $\theta = \ln\alpha \in \mathcal{R}$. It follows from

$$\frac{\partial k(y,\alpha)}{\partial y} = \frac{2}{y} - \alpha(1 - e^{-y}) \qquad (y > 0; \alpha > 0)$$

that the function $w(\theta)$ is given by

$$2w(\theta) = e^\theta + \theta + 2\ln y - e^\theta(y + e^{-y}) \qquad (\theta \in \mathcal{R}) \tag{5.7}$$

where y is a function of θ defined implicitly by

$$e^\theta = \frac{2}{y(1 - e^{-y})} \tag{5.8}$$

a one-to-one mapping: $\theta \mapsto y$ from $\mathcal{R}$ to $\{y : y > 0\}$.

The first-order derivative of $2w(\theta)$ is negative and the second-order derivative of $2w(\theta)$ is positive $\forall y > 0$. Therefore it is strictly decreasing and convex. The identities (2.5) and (2.4) can be easily established by applying standard algebraic operations to (5.7) and (5.8).

References

[1] Ahrens JH, Dieter U (1974). "Computer methods for sampling from gamma, beta, Poisson and binomial distributions." *Computing*, **12**, 223-246.

[2] Ahrens JH, Dieter U (1982). "Generating gamma variates by a modified rejection technique." *Communications of the ACM*, **25**, 47-54.

[3] Cheng RCH, Feast GM (1979). "Some simple gamma variate generators." *Journal of the Royal Statistical Society, Series C*, **28**, 290-295.

[4] Cheng RCH, Feast GM (1980). "Gamma variate generators with increased shape parameter range." *Communications of the ACM*, **23**, 389-395.

[5] Devroye L (1986). *Non-Uniform Random Variate Generation.* Springer-Verlag, New York, NY.

[6] Doornik J (2005). "An Improved Ziggurat Method to Generate Normal Random Samples." www.doornik.com/research.html.

[7] Kinderman AJ, Monahan JF (1977). "Computer generation of random variables using the ratio of uniform deviates." *ACM Transactions on Mathematical Software*, **3**, 257-260.

[8] Marsaglia G, Tsang WW (2000). "A Simple Method for Generating Gamma Variables." *ACM Transactions on Mathematical Software*, **26**, 363-372.

[9] Marsaglia G, Tsang WW (2000). "The Ziggurat Method for Generating Random Variables." *Journal of Statistical Software*, **5**, 1-7.

[10] Tanizaki H (2008). "A Simple Gamma Random Number Generator for Arbitrary Shape Parameters." *Economics Bulletin*, **3**, 1-10.

[11] Tadikamalla PR (1978). "Computer Generation of Gamma Random Variables II." *Communications of the ACM*, **21**, 925-928.

[12] Thomas DB, Luk W, Leong P, Villasenor JD (2007). "Gaussian Random Number Generators." *ACM Computing Surveys*, **39**, Article 11.

[13] Wakefield, JC, Gelfand, AE, Smith, AFM (1991). "Efficient Generation of Random Variates via the Ratio-of-Uniforms Method." *Statistics and Computing*, **1**, 129-133.