5.6.4 Using $U(0,1)$: Discrete Type

In Sections 5.6.2 and 5.6.3, the random number generators from continuous distributions are discussed, i.e., the transformation of variables in Section 5.6.2 and the inverse transform method in Section 5.6.3 are utilized.

Based on the uniform random draw between zero and one, in this section we deal with some discrete distributions and consider generating their random numbers.

As a representative random number generation method, we can consider utilizing the inverse transform method in the case of discrete random variables.

Suppose that a discrete random variable X can take x_1, x_2, $\cdots$, x_n, where the probability which X takes x_i is given by $f(x_i)$, i.e., $P(X = x_i) = f(x_i)$.

Generate a uniform random draw u, which is between zero and one.

Consider the case where we have $F(x_{i-1}) \leq u < F(x_i)$, where $F(x_i) = P(X \leq x_i)$ and $F(x_0) = 0$.

Then, the random draw of X is given by x_i.

References

Ahrens, J.H. and Dieter, U., 1980, “Sampling from Binomial and Poisson Distributions: A Method with Bounded Computation Times,” *Computing*, Vol.25, pp.193 – 208.

Ahrens, J.H. and Dieter, U., 1988, “Efficient, Table-Free Sampling Methods for the Exponential, Cauchy and Normal Distributions,” *Communications of the ACM*, Vol.31, pp.1330 – 1337.

Bays, C. and Durham, S.D., 1976, “Improving a Poor Random Number Generator,” *ACM Transactions on Mathematical Software*, Vol.2, pp.59 – 64.

Box, G.E.P. and Muller, M.E., 1958, “A Note on the Generation of Random Normal Deviates,” *Annals of Mathematical Statistics*, Vol.29, No.2, pp.610 – 611.

Cheng, R.C.H., 1998, “Random Variate Generation,” in *Handbook of Simulation*, Chap.5, edited by Banks, J., pp.139 – 172, John Wiley & Sons.

De Matteis, A. and Pagnutti, S., 1993, “Long-Range Correlation Analysis of the Wichmann-Hill Random Number Generator,” *Statistics and Computing*, Vol.3, pp.67 – 70.

Fishman, G.S., 1996, *Monte Carlo: Concepts, Algorithms, and Applications*, Springer-Verlag.

Gentle, J.E., 1998, *Random Number Generation and Monte Carlo Methods*, Springer-Verlag.

Hastings, C., 1955, *Approximations for Digital Computers*, Princeton University Press.

Hill, I.D and Pike, A.C., 1967, "Algorithm 2999: Chi-Squared Integral," *Communications of the ACM*, Vol.10, pp.243 – 244.

Hogg, R.V. and Craig, A.T., 1995, *Introduction to Mathematical Statistics* (Fifth Edition), Prentice Hall.

Johnson, N.L. and Kotz, S., 1970a, *Continuous Univariate Distributions*, Vol.1, John Wiley & Sons.

Johnson, N.L. and Kotz, S., 1970b, *Continuous Univariate Distributions*, Vol.2, John Wiley & Sons.

Kachitvichyanukul, V. and Schmeiser, B., 1985, "Computer Generation of Hypergeometric Random Variates," *Journal of Statistical Computation and Simulation*, Vol.22, pp.127 – 145.

Kennedy, Jr. W.J. and Gentle, J.E., 1980, *Statistical Computing* (Statistics: Textbooks and Monographs, Vol.33), Marcel Dekker.

Knuth, D.E., 1981, *The Art of Computer Programming, Vol.2: Seminumerical Algorithms* (Second Edition), Addison-Wesley, Reading, MA.

Kotz, S. and Johnson, N.L., 1982, *Encyclopedia of Statistical Sciences*, Vol.2, pp.188 – 193, John Wiley & Sons.

Kotz, S., Balakrishman, N. and Johnson, N.L., 2000a, *Univariate Discrete Distributions* (Second Edition), John Wiley & Sons.

Kotz, S., Balakrishman, N. and Johnson, N.L., 2000b, *Continuous Univariate Distributions, Vol.1* (Second Edition), John Wiley & Sons.

Kotz, S., Balakrishman, N. and Johnson, N.L., 2000c, *Continuous Univariate Dis-*

tributions, *Vol.2* (Second Edition), John Wiley & Sons.

Kotz, S., Balakrishman, N. and Johnson, N.L., 2000d, *Discrete Multivariate Distributions* (Second Edition), John Wiley & Sons.

Kotz, S., Balakrishman, N. and Johnson, N.L., 2000e, *Continuous Multivariate Distributions*, *Vol.1* (Second Edition), John Wiley & Sons.

Law, A.M. and Kelton, W.D., 2000, *Simulation Modeling and Analysis* (Third Edition), McGraw-Hill Higher Education.

L'Ecuyer, P., 1988, "Efficient and Portable Combined Random Number Generators," *Communications of the ACM*, Vol.31, No.6, pp.742 – 749.

L’Ecuyer, P., 1990, “Random Numbers for Simulation,” *Communications of the ACM*, Vol.33, No.10, pp.85 – 97.

L’Ecuyer, P., 1998, “Random Number Generation,” in *Handbook of Simulation*, Chap. 4, edited by Banks, J., pp.93 – 137, John Wiley & Sons.

Marsaglia, G., 1964, “Generating a Variable from the Tail of the Normal Distribution,” *Technometrics*, Vol.6, pp.101 – 102.

Marsaglia, G., MacLaren, M.D. and Bray, T.A., 1964, “A Fast Method for Generating Normal Random Variables,” *Communications of the ACM*, Vol.7, pp.4 – 10.

Marsaglia, G. and Zaman, A., 1994, “Rapid Evaluation of the Inverse of the Normal Distribution Function,” *Statistics and Probability Letters*, Vol.19, No.2, pp.259 – 266.

Niederreiter, H., 1992, *Random Number Generation and Quasi-Monte Carlo Methods* (CBMS-NFS Regional Conference Series in Applied Mathematics 63), Society for Industrial and Applied Mathematics.

Odeh, R.E. and Evans, J.O., 1974, “Algorithm AS 70: The Percentage Points of the Normal Distribution,” *Applied Statistics*, Vol.23, No.1, pp.96 – 97.

Odell, P.L. and Feiveson, A.H., 1966, “A Numerical Procedure to Generate a Simple

Covariance Matrix," *Journal of the American Statistical Association*, Vol.61, No.313, pp.199 – 203.

Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P., 1992a, *Numerical Recipes in C: The Art of Scientific Computing* (Second Edition), Cambridge University Press.

Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P., 1992b, *Numerical Recipes in Fortran: The Art of Scientific Computing* (Second Edition), Cambridge University Press.

Ripley, B.D., 1987, *Stochastic Simulation*, John Wiley & Sons.

Robert, C.P. and Casella, G., 1999, *Monte Carlo Statistical Methods*, Springer-Verlag.

Ross, S.M., 1997, *Simulation* (Second Edition), Academic Press.

Rubinstein, R.Y., 1981, *Simulation and the Monte Carlo Method*, John Wiley & Sons.

Rubinstein, R.Y. and Melamed, B., 1998, *Modern Simulation and Modeling*, John Wiley & Sons.

Schmeiser, B. and Kachitvichyanukul, V., 1990, "Noninverse Correlation Induction: Guidelines for Algorithm Development," *Journal of Computational and*

Applied Mathematics, Vol.31, pp.173 – 180.

Shibata, Y., 1981, *Normal Distribution* (in Japanese), Tokyo University Press.

Smith, W.B and Hocking, R.R., 1972, “Algorithm AS53: Wishart Variate Generator,” *Applied Statistics*, Vol.21, No.3, pp.341 – 345.

Stadlober, E., 1990, “The Ratio of Uniforms Approach for Generating Discrete Random Variates,” *Journal of Computational and Applied Mathematics*, Vol.31, pp.181 – 189.

Takeuchi, K., 1989, *Dictionary of Statistics* (in Japanese), Toyo-Keizai.

Thompson, J.R., 2000, *Simulation: A Modeler’s Approach*, Jhon Wiley & Sons.

Wichmann, B.A. and Hill, I.D., 1982, “Algorithm AS183: An Efficient and Portable Pseudo-random Number Generator,” *Applied Statistics*, Vol.31, No.2, pp.188 – 190.

Wichmann, B.A. and Hill, I.D., 1984, “Correction of Algorithm AS183: An Efficient and Portable Pseudo-random Number Generator,” *Applied Statistics*, Vol.33, No.2, p.123.

Zellner, A., 1971, *An Introduction to Bayesian Inference in Econometrics*, John Wiley & Sons.

5.7 Sampling Method II: Random Number Generation

5.7.1 Rejection Sampling (棄却法)

We want to generate random draws from $f(x)$, called the **target density (目的密度)**, but we consider the case where it is hard to sample from $f(x)$.

Now, suppose that it is easy to generate a random draw from another density $f_*(x)$, called the **sampling density (サンプリング密度)** or **proposal density (提案密度)**.

In this case, random draws of X from $f(x)$ are generated by utilizing the random draws sampled from $f_*(x)$.

Let x be the the random draw of X generated from $f(x)$.

Suppose that $q(x)$ is equal to the ratio of the target density and the sampling density, i.e.,

$$q(x) = \frac{f(x)}{f_*(x)}. \tag{1}$$

Then, the target density is rewritten as:

$$f(x) = q(x) f_*(x).$$

Based on $q(x)$, the acceptance probability is obtained.

Depending on the structure of the acceptance probability, we have three kinds of sampling techniques, i.e., **rejection sampling (棄却法)** in this section, **impor-**

tance resampling (重点的リサンプリング法) in Section 5.7.2 and the **Metropolis-Hastings algorithm (メトロポリス－ハスティング・アルゴリズム)** in Section 5.7.4.

See Liu (1996) for a comparison of the three sampling methods.

Thus, to generate random draws of x from $f(x)$, the functional form of $q(x)$ should be known and random draws have to be easily generated from $f_*(x)$.

In order for rejection sampling to work well, the following condition has to be satisfied:

$$q(x) = \frac{f(x)}{f_*(x)} < c,$$

where c is a fixed value.

That is, $q(x)$ has an upper limit.

As discussed below, $1/c$ is equivalent to the acceptance probability.

If the acceptance probability is large, rejection sampling computationally takes a lot of time.

Under the condition $q(x) < c$ for all x, we may minimize c.

That is, since we have $q(x) < \sup_x q(x) \leq c$, we may take the supremum of $q(x)$ for c.

Thus, in order for rejection sampling to work efficiently, c should be the supremum of $q(x)$ with respect to x, i.e., $c = \sup_x q(x)$.

Let x^* be the random draw generated from $f_*(x)$, which is a candidate of the random draw generated from $f(x)$.

Define $\omega(x)$ as:

$$\omega(x) = \frac{q(x)}{\sup_z q(z)} = \frac{q(x)}{c},$$

which is called the **acceptance probability (採択確率)**.

Note that we have $0 \leq \omega(x) \leq 1$ when $\sup_z q(z) = c < \infty$.

The supremum $\sup_z q(z) = c$ has to be finite.

This condition is sometimes too restrictive, which is a crucial problem in rejection sampling.

A random draw of X is generated from $f(x)$ in the following way:

(i) Generate x^* from $f_*(x)$ and compute $\omega(x^*)$.

(ii) Set $x = x^*$ with probability $\omega(x^*)$ and go back to (i) otherwise.

In other words, generating u from a uniform distribution between zero and one, take $x = x^*$ if $u \leq \omega(x^*)$ and go back to (i) otherwise.

The above random number generation procedure can be justified as follows.

Let U be the uniform random variable between zero and one, X be the random variable generated from the target density $f(x)$,

X^* be the random variable generated from the sampling density $f_*(x)$, and x^* be the realization (i.e., the random draw) generated from the sampling density $f_*(x)$. Consider the probability $P(X \leq x | U \leq \omega(x^*))$, which should be the cumulative distribution of X, $F(x)$, from Step (ii).

The probability $P(X \leq x | U \leq \omega(x^*))$ is rewritten as follows:

$$P(X \leq x | U \leq \omega(x^*)) = \frac{P(X \leq x, U \leq \omega(x^*))}{P(U \leq \omega(x^*))},$$

where the numerator is represented as:

$$P(X \leq x, U \leq \omega(x^*)) = \int_{-\infty}^{x} \int_{0}^{\omega(t)} f_{u,*}(u,t)\, \mathrm{d}u\, \mathrm{d}t = \int_{-\infty}^{x} \int_{0}^{\omega(t)} f_u(u) f_*(t)\, \mathrm{d}u\, \mathrm{d}t$$

$$= \int_{-\infty}^{x} \Big(\int_{0}^{\omega(t)} f_u(u)\, \mathrm{d}u \Big) f_*(t)\, \mathrm{d}t = \int_{-\infty}^{x} \Big(\int_{0}^{\omega(t)} \mathrm{d}u \Big) f_*(t)\, \mathrm{d}t$$

$$= \int_{-\infty}^{x} \Big[u \Big]_{0}^{\omega(t)} f_*(t)\, \mathrm{d}t = \int_{-\infty}^{x} \omega(t) f_*(t)\, \mathrm{d}t = \int_{-\infty}^{x} \frac{q(t)}{c} f_*(t)\, \mathrm{d}t = \frac{F(x)}{c},$$

and the denominator is given by:

$$P(U \le \omega(x^*)) = P(X \le \infty, U \le \omega(x^*)) = \frac{F(\infty)}{c} = \frac{1}{c}.$$

In the numerator, $f_{u,*}(u, x)$ denotes the joint density of random variables U and X^*. Because the random draws of U and X^* are independently generated in Steps (i) and (ii) we have $f_{u,*}(u, x) = f_u(u) f_*(x)$, where $f_u(u)$ and $f_*(x)$ denote the marginal density of U and that of X^*.

The density function of U is given by $f_u(u) = 1$, because the distribution of U is assumed to be uniform between zero and one.

Thus, the first four equalities are derived.

Furthermore, in the seventh equality of the numerator, since we have:

$$\omega(x) = \frac{q(x)}{c} = \frac{f(x)}{cf_*(x)},$$

$\omega(x)f_*(x) = f(x)/c$ is obtained.

Finally, substituting the numerator and denominator shown above, we have the following equality:

$$P(X \leq x|U \leq \omega(x^*)) = F(x).$$

Thus, the rejection sampling method given by Steps (i) and (ii) is justified.

The rejection sampling method is the most efficient sampling method in the sense of precision of the random draws, because using rejection sampling we can generate mutually independently distributed random draws.

However, for rejection sampling we need to obtain the c which is greater than or equal to the supremum of $q(x)$.

If the supremum is infinite, i.e., if c is infinite, $\omega(x)$ is zero and accordingly the candidate x^* is never accepted in Steps (i) and (ii).

Moreover, as for another remark, note as follows.

Let N_R be the average number of the rejected random draws.

We need $(1 + N_R)$ random draws in average to generate one random number from $f(x)$.

In other words, the acceptance rate is given by $1/(1+N_R)$ in average, which is equal to $1/c$ in average because of $P(U \leq \omega(x^*)) = 1/c$.

Therefore, to obtain one random draw from $f(x)$, we have to generate $(1 + N_R)$ random draws from $f_*(x)$ in average.

See, for example, Boswell, Gore, Patil and Taillie (1993), O'Hagan (1994) and Geweke (1996) for rejection sampling.

To examine the condition that $\omega(x)$ is greater than zero, i.e., the condition that the supremum of $q(x)$ exists, consider the case where $f(x)$ and $f_*(x)$ are distributed as $N(\mu, \sigma^2)$ and $N(\mu_*, \sigma_*^2)$, respectively.

$q(x)$ is given by:

$$
\begin{aligned}
q(x) = \frac{f(x)}{f_*(x)} &= \frac{(2\pi\sigma^2)^{-1/2} \exp\Bigl(-\dfrac{1}{2\sigma^2}(x-\mu)^2\Bigr)}{(2\pi\sigma_*^2)^{-1/2} \exp\Bigl(-\dfrac{1}{2\sigma_*^2}(x-\mu_*)^2\Bigr)} \\
&= \frac{\sigma_*}{\sigma} \exp\Bigl(-\frac{1}{2\sigma^2}(x-\mu)^2 + \frac{1}{2\sigma_*^2}(x-\mu_*)^2\Bigr) \\
&= \frac{\sigma_*}{\sigma} \exp\left(-\frac{1}{2}\frac{\sigma_*^2-\sigma^2}{\sigma^2\sigma_*^2}\Bigl(x - \frac{\mu\sigma_*^2 - \mu_*\sigma^2}{\sigma_*^2-\sigma^2}\Bigr)^2 + \frac{1}{2}\frac{(\mu-\mu_*)^2}{\sigma_*^2-\sigma^2}\right).
\end{aligned}
$$

If $\sigma_*^2 < \sigma^2$, $q(x)$ goes to infinity as x is large.

In the case of $\sigma_*^2 > \sigma^2$, the supremum of $q(x)$ exists, which condition implies that $f_*(x)$ should be more broadly distributed than $f(x)$.

In this case, the supremum is obtained as:

$$c = \sup_x q(x) = \frac{\sigma_*}{\sigma} \exp\Big(\frac{1}{2}\frac{(\mu - \mu_*)^2}{\sigma_*^2 - \sigma^2}\Big).$$

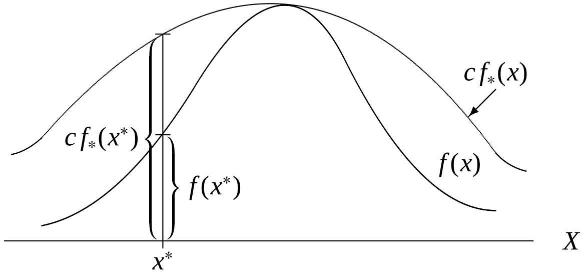
When $\sigma^2 = \sigma_*^2$ and $\mu = \mu_*$, we have $q(x) = 1$, which implies $\omega(x) = 1$.

That is, a random draw from the sampling density $f_*(x)$ is always accepted as a random draw from the target density $f(x)$, where $f(x)$ is equivalent to $f_*(x)$ for all x.

If $\sigma^2 = \sigma_*^2$ and $\mu \neq \mu_*$, the supremum of $q(x)$ does not exists.

Accordingly, the rejection sampling method does not work in this case.

Figure 1: Rejection Sampling



From the definition of $\omega(x)$, we have the inequality $f(x) \leq cf_*(x)$.

$cf_*(x)$ and $f(x)$ are displayed in Figure 1.

The ratio of $f(x^*)$ and $cf_*(x^*)$ corresponds to the acceptance probability at x^*, i.e., $\omega(x^*)$.

Thus, for rejection sampling, $cf_*(x)$ has to be greater than or equal to $f(x)$ for all x, which implies that the sampling density $f_*(x)$ needs to be more widely distributed than the target density $f(x)$.

Finally, note that the above discussion holds without any modification even though $f(x)$ is a kernel of the target density, i.e., even though $f(x)$ is proportional to the target density, because the constant term is canceled out between the numerator and denominator (remember that $\omega(x) = q(x)/\sup_z q(z)$).

Normal Distribution: $N(0,1)$: First, denote the half-normal distribution by:

$$f(x) = \begin{cases} \dfrac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

The half-normal distribution above corresponds to the positive part of the standard normal probability density function.

Using rejection sampling, we consider generating standard normal random draws based on the half-normal distribution.

We take the sampling density as the exponential distribution:

$$f_*(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise,} \end{cases}$$

where $\lambda > 0$. Since $q(x)$ is defined as $q(x) = f(x)/f_*(x)$, the supremum of $q(x)$ is given by:

$$c = \sup_x q(x) = \frac{2}{\lambda\sqrt{2\pi}} e^{\frac{1}{2}\lambda^2}.$$

which depends on parameter λ.

Remember that $P(U \leq \omega(x^*)) = 1/c$ corresponds to the acceptance probability.

Since we need to increase the acceptance probability to reduce computational time,

we want to obtain the λ which minimizes $\sup_x q(x)$ with respect to λ.

Solving the minimization problem, $\lambda = 1$ is obtained.

Substituting $\lambda = 1$, the acceptance probability $\omega(x)$ is derived as:

$$\omega(x) = e^{-\frac{1}{2}(x-1)^2},$$

for $0 < x < \infty$.

Remember that $-\log U$ has an exponential distribution with $\lambda = 1$ when $U \sim U(0, 1)$.

Therefore, the algorithm is represented as follows.

(i) Generate two independent uniform random draws u_1 and u_2 between zero and one.

(ii) Compute $x^* = -\log u_2$, which indicates the exponential random draw generated from the target density $f_*(x)$.

(iii) Set $x = x^*$ if $u_1 \leq \exp(-\frac{1}{2}(x^* - 1)^2)$, i.e., $-2\log(u_1) \geq (x^* - 1)^2$, and return to (i) otherwise.

x in Step (iii) yields a random draw from the half-normal distribution.

To generate a standard normal random draw utilizing the half-normal random draw above, we may put the positive or negative sign randomly with x.
Therefore, the following Step (iv) is additionally put.

(iv) Generate a uniform random draw u_3 between zero and one, and set $z = x$ if $u_3 \leq 1/2$ and $z = -x$ otherwise.

z gives us a standard normal random draw.
Note that the number of iteration in Step (iii) is given by $c = \sqrt{2e/\pi} \approx 1.3155$ in average, or equivalently, the acceptance probability in Step (iii) is $1/c \approx 0.7602$.
The source code for this standard normal random number generator is shown in

snrnd6(ix,iy,rn).

snrnd6(ix,iy,rn)

```
        subroutine snrnd6(ix,iy,rn)
c
c  Use "snrnd6(ix,iy,rn)"
c  together with "urnd(ix,iy,rn)".
c
c  Input:
c    ix, iy:   Seeds
c  Output:
c    rn: Normal Random Draw N(0,1)
c
      1 call urnd(ix,iy,rn1)
        call urnd(ix,iy,rn2)
        y=-log(rn2)
        if( -2.*log(rn1).lt.(y-1.)**2 ) go to 1
        call urnd(ix,iy,rn3)
          if(rn3.le.0.5) then
        rn= y
          else
        rn=-y
          endif
        return
        end
```

Note that `snrnd6(ix,iy,rn)` should be used together with `urnd(ix,iy,rn)`. Thus, utilizing rejection sampling, we have the standard normal random number generator, which is based on the half-normal distribution.

Gamma Distribution: $G(\alpha, 1)$ **for** $0 < \alpha \leq 1$ **and** $1 < \alpha$**:** In this section, utilizing rejection sampling we show an example of generating random draws from the gamma distribution with parameters α and $\beta = 1$, i.e., $G(\alpha, 1)$.

When $X \sim G(\alpha, 1)$, the density function of X is given by:

$$f(x) = \begin{cases} \dfrac{1}{\Gamma(\alpha)} x^{\alpha-1} e^{-x}, & \text{for } 0 < x < \infty, \\ \\ 0, & \text{otherwise.} \end{cases}$$

Ahrens and Dieter (1974) consider the case of $0 < \alpha \leq 1$, which is discussed in this section.

The case of $\alpha > 1$ will be discussed later.

Using the rejection sampling, the composition method and the inverse transform method, we consider generating random draws from $G(\alpha, 1)$ for $0 < \alpha \leq 1$.

The sampling density is taken as:

$$f_*(x) = \frac{e}{\alpha + e}\alpha x^{\alpha-1} I_1(x) + \frac{\alpha}{\alpha + e} e^{-x+1} I_2(x),$$

where both $I_1(x)$ and $I_2(x)$ denote the indicator functions defined as:

$$I_1(x) = \begin{cases} 1, & \text{if } 0 < x \leq 1, \\ 0, & \text{otherwise,} \end{cases} \qquad I_2(x) = \begin{cases} 1, & \text{if } 1 < x, \\ 0, & \text{otherwise.} \end{cases}$$

Random number generation from the sampling density above utilizes the composition method and the inverse transform method.

The cumulative distribution related to $f_*(x)$ is given by:

$$F_*(x) = \begin{cases} \dfrac{e}{\alpha + e} x^{\alpha}, & \text{if } 0 < x \le 1, \\ \dfrac{e}{\alpha + e} + \dfrac{\alpha}{\alpha + e}(1 - e^{-x+1}), & \text{if } x > 1. \end{cases}$$

Note that $0 < \alpha \le 1$ is required because the sampling density for $0 < x \le 1$ has to satisfy the property that the integration is equal to one.

The acceptance probability $\omega(x) = q(x)/\sup_z q(z)$ for $q(x) = f(x)/f_*(x)$ is given by:

$$\omega(x) = e^{-x} I_1(x) + x^{\alpha-1} I_2(x).$$

Moreover, the mean number of trials until success, i.e., $c = \sup_z q(z)$ is represented as:

$$c = \frac{\alpha + e}{\alpha e \Gamma(\alpha)},$$

which depends on α and is not greater than 1.39.

Note that $q(x)$ takes a maximum value at $x = 1$.

The random number generation procedure is given by:

(i) Generate a uniform random draw u_1 from $U(0, 1)$, and set $x^* = ((\alpha/e+1)u_1)^{1/\alpha}$ if $u_1 \leq e/(\alpha + e)$ and $x^* = -\log((1/e + 1/\alpha)(1 - u_1))$ if $u_1 > e/(\alpha + e)$.

(ii) Obtain $\omega(x^*) = e^{-x^*}$ if $u_1 \leq e/(\alpha + e)$ and $\omega(x^*) = x^{*\alpha-1}$ if $u_1 > e/(\alpha + e)$.

(iii) Generate a uniform random draw u_2 from $U(0,1)$, and set $x = x^*$ if $u_2 \leq \omega(x^*)$ and return to (i) otherwise.

In Step (i) a random draw x^* from $f_*(x)$ can be generated by the inverse transform method discussed in Section 5.6.3.

——— gammarnd2(ix,iy,alpha,rn) ———

```
      subroutine gammarnd2(ix,iy,alpha,rn)
c
c  Use "gammarnd2(ix,iy,alpha,rn)"
c  together with "urnd(ix,iy,rn)".
c
c  Input:
```

```
c    ix, iy: Seeds
c    alpha:  Shape Parameter (0<alpha \le 1)
c  Output:
c    rn: Gamma Random Draw
c        with Parameters alpha and beta=1
c
      e=2.71828182845905
    1 call urnd(ix,iy,rn0)
      call urnd(ix,iy,rn1)
            if( rn0.le.e/(alpha+e) ) then
      rn=( (alpha+e)*rn0/e )**(1./alpha)
      if( rn1.gt.e**(-rn) ) go to 1
            else
      rn=-log((alpha+e)*(1.-rn0)/(alpha*e))
      if( rn1.gt.rn**(alpha-1.) ) go to 1
            endif
      return
      end
```

Note that `gammarnd2(ix,iy,alpha,rn)` should be used with `urnd(ix,iy,rn)`.

In `gammarnd2(ix,iy,alpha,rn)`, the case of $0 < \alpha \leq 1$ has been shown.

Now, using rejection sampling, the case of $\alpha > 1$ is discussed in Cheng (1977, 1998).

The sampling density is chosen as the following cumulative distribution:

$$F_*(x) = \begin{cases} \dfrac{x^\lambda}{\delta + x^\lambda}, & \text{for } x > 0, \\ \\ 0, & \text{otherwise,} \end{cases}$$

which is sometimes called the **log-logistic distribution**.

Then, the probability density function, $f_*(x)$, is given by:

$$f_*(x) = \begin{cases} \dfrac{\lambda\delta x^{\lambda-1}}{(\alpha + x^\lambda)^2}, & \text{for } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

By the inverse transform method, the random draw from $f_*(x)$, denoted by x, is generated as follows:

$$x = \Big(\frac{\delta u}{1-u}\Big)^{1/\lambda},$$

where u denotes the uniform random draw generated from $U(0,1)$.

For the two parameters, $\lambda = \sqrt{2\alpha - 1}$ and $\delta = \alpha^\lambda$ are chosen, taking into account minimizing $c = \sup_x q(x) = \sup_x f(x)/f_*(x)$ with respect to δ and λ (note that λ and δ are approximately taken, since it is not possible to obtain the explicit solution of δ and λ).

Then, the number of rejections in average is given by:

$$c = \frac{4\alpha^\alpha e^{-\alpha}}{\Gamma(\alpha)\sqrt{2\alpha - 1}},$$

which is computed as:

$$1.47 \text{ when } \alpha = 1, \qquad 1.25 \text{ when } \alpha = 2, \qquad 1.17 \text{ when } \alpha = 5,$$
$$1.15 \text{ when } \alpha = 10, \qquad 1.13 \text{ when } \alpha = \infty.$$

Thus, the average number of rejections is quite small for all α.

The random number generation procedure is given by:

(i) Set $a = 1/\sqrt{2\alpha - 1}$, $b = \alpha - \log 4$ and $c = \alpha + \sqrt{2\alpha - 1}$.

(ii) Generate two uniform random draws u_1 and u_2 from $U(0,1)$.

(iii) Set $y = a \log \dfrac{u_1}{1-u_1}$, $x^* = \alpha e^y$, $z = u_1^2 u_2$ and $r = b + cy - x$.

(iv) Take $x = x^*$ if $r \geq \log z$ and return to (ii) otherwise.

To avoid evaluating the logarithm in Step (iv), we put Step (iii)' between Steps (iii) and (iv), which is as follows:

(iii)' Take $x = x^*$ if $r \geq 4.5z - d$ and go to (iv) otherwise.

d is defined as $d = 1 + \log 4.5$, which has to be computed in Step (i).

Note that we have the relation: $\theta z - (1 + \log\theta) \geq \log z$ for all $z > 0$ and any given $\theta > 0$, because $\log z$ is a concave function of z. According to Cheng (1977), the choice of θ is not critical and the suggested value is $\theta = 4.5$, irrespective of α.

The source code for Steps (i) – (iv) and (iii)' is given by `gammarnd3(ix,iy,alpha,rn)`.

——— `gammarnd3(ix,iy,alpha,rn)` ———

```
      subroutine gammarnd3(ix,iy,alpha,rn)
c
c  Use "gammarnd3(ix,iy,alpha,rn)"
c  together with "urnd(ix,iy,rn)".
c
```

```
c  Input:
c    ix, iy: Seeds
c    alpha:  Shape Parameter (1<alpha)
c  Output:
c    rn: Gamma Random Draw
c        with Parameters alpha and beta=1
c
      e=2.71828182845905
      a=1./sqrt(2.*alpha-1.)
      b=alpha-log(4.)
      c=alpha+sqrt(2.*alpha-1.)
      d=1.+log(4.5)
    1 call urnd(ix,iy,u1)
      call urnd(ix,iy,u2)
      y=a*log(u1/(1.-u1))
      rn=alpha*(e**y)
      z=u1*u1*u2
      r=b+c*y-rn
      if( r.ge.4.5*z-d ) go to 2
      if( r.lt.log(z)  ) go to 1
```

```
26:       2 return
27:         end
```

Note that `gammarnd3(ix,iy,alpha,rn)` requires `urnd(ix,iy,rn)`.

Line 24 corresponds to Step (iii)’, which gives us a fast acceptance.

Taking into account a recent progress of a personal computer, we can erase Lines 17 and 24 from `gammarnd3`, because evaluating the `if(...)` sentences in Lines 24 and 25 sometimes takes more time than computing the logarithm in Line 25.

Thus, using both `gammarnd2` and `gammarnd3`, we have the gamma random number generator with parameters $\alpha > 0$ and $\beta = 1$.

5.7.2 Importance Resampling (重点的リサンプリング)

The **importance resampling** method also utilizes the sampling density $f_*(x)$, where we should choose the sampling density from which it is easy to generate random draws.

Let x_i^* be the *i*th random draw of x generated from $f_*(x)$.

The acceptance probability is defined as:

$$\omega(x_i^*) = \frac{q(x_i^*)}{\sum_{j=1}^{n} q(x_j^*)},$$

where $q(\cdot)$ is represented as equation (1).

To obtain a random draws from $f(x)$, we perform the following procedure:

(i) Generate x_j^* from the sampling density $f_*(x)$ for $j = 1, 2, \cdots, n$.

(ii) Compute $\omega(x_j^*)$ for all $j = 1, 2, \cdots, n$.

(iii) Generate a uniform random draw u between zero and one and take $x = x_j^*$ when $\Omega_{j-1} \leq u < \Omega_j$, where $\Omega_j = \sum_{i=1}^{j} \omega(x_i^*)$ and $\Omega_0 \equiv 0$.

The x obtained in Step (iii) represents a random draw from the target density $f(x)$.
In Step (ii), all the probability weights $\omega(x_j^*)$, $j = 1, 2, \cdots, n$, have to be computed for importance resampling.

Thus, we need to generate n random draws from the sampling density $f_*(x)$ in advance.

When we want to generate more random draws (say, N random draws), we may repeat Step (iii) N times.

In the importance resampling method, there are n realizations, i.e., x_1^*, x_2^*, $\cdots$, x_n^*, which are mutually independently generated from the sampling density $f_*(x)$.

The cumulative distribution of $f(x)$ is approximated by the following empirical distribution:

$$P(X \leq x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t = \int_{-\infty}^{x} \frac{f(t)}{f_*(t)} f_*(t)\,\mathrm{d}t = \frac{\int_{-\infty}^{x} q(t) f_*(t)\,\mathrm{d}t}{\int_{-\infty}^{\infty} q(t) f_*(t)\,\mathrm{d}t}$$

$$\approx \frac{(1/n)\sum_{i=1}^{n} q(x_i^*)I(x, x_i^*)}{(1/n)\sum_{j=1}^{n} q(x_j^*)} = \sum_{i=1}^{n} \omega(x_i^*)I(x, x_i^*),$$

where $I(x, x_i^*)$ denotes the indicator function which satisfies $I(x, x_i^*) = 1$ when $x \geq x_i^*$ and $I(x, x_i^*) = 0$ otherwise.

$P(X = x_i^*)$ is approximated as $\omega(x_i^*)$.

See Smith and Gelfand (1992) and Bernardo and Smith (1994) for the importance resampling procedure.

As mentioned in Section 5.7.1, for rejection sampling, $f(x)$ may be a kernel of the target density, or equivalently, $f(x)$ may be proportional to the target density.

Similarly, the same situation holds in the case of importance resampling.

That is, $f(x)$ may be proportional to the target density for importance resampling, too.

To obtain a random draws from $f(x)$, importance resampling requires n random draws from the sampling density $f_*(x)$, but rejection sampling needs $(1+N_R)$ random draws from the sampling density $f_*(x)$.

For importance resampling, when we have n different random draws from the sampling density, we pick up one of them with the corresponding probability weight.

The importance resampling procedure computationally takes a lot of time, because we have to compute all the probability weights Ω_j, $j = 1, 2, \cdots, n$, in advance even when we want only one random draw.

When we want to generate N random draws, importance resampling requires n random draws from the sampling density $f_*(x)$, but rejection sampling needs $n(1 + N_R)$ random draws from the sampling density $f_*(x)$.

Thus, as N increases, importance resampling is relatively less computational than rejection sampling.

Note that $N < n$ is recommended for the importance resampling method.

In addition, when we have N random draws from the target density $f(x)$, some of the random draws take the exactly same values for importance resampling, while all the random draws take the different values for rejection sampling. Therefore, we can see that importance resampling is inferior to rejection sampling in the sense of precision of the random draws.

Normal Distribution: $N(0,1)$**:** Again, we consider an example of generating standard normal random draws based on the half-normal distribution:

$$f(x) = \begin{cases} \dfrac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

We take the sampling density as the following exponential distribution:

$$f_*(x) = \begin{cases} e^{-x}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise}, \end{cases}$$

which is exactly the same sampling density as in Section 5.7.1.

Given the random draws x_i^*, $i = 1, \cdots, n$, generated from the above exponential density $f_*(x)$, the acceptance probability $\omega(x_i^*)$ is given by:

$$\omega(x_i^*) = \frac{q(x_i^*)}{\sum_{j=1}^n q(x_j^*)} = \frac{f(x_i^*)/f_*(x_i^*)}{\sum_{j=1}^n f(x_j^*)/f_*(x_j^*)} = \frac{\exp(-\frac{1}{2}x_i^{*2} + x_i^*)}{\sum_{j=1}^n \exp(-\frac{1}{2}x_j^{*2} + x_j^*)}.$$

Therefore, a random draw from the half-normal distribution is generated as follows.

(i) Generate uniform random draws $u_1, u_2, \cdots, u_n$ from $U(0,1)$.

(ii) Obtain $x_i^* = -\log(u_i)$ for $i = 1, 2, \cdots, n$.

(iii) Compute $\omega(x_i^*)$ for $i = 1, 2, \cdots, n$.

(iv) Generate a uniform random draw v_1 from $U(0,1)$.

(v) Set $x = x_j^*$ when $\Omega_{j-1} \le v_1 < \Omega_j$ for $\Omega_j = \sum_{i=1}^{j} \omega(x_i^*)$ and $\Omega_0 = 0$.

x is taken as a random draw generated from the half-normal distribution $f(x)$.
In order to have a standard normal random draw, we additionally put the following step.

(vi) Generate a uniform random draw v_2 from $U(0,1)$, and set $z = x$ if $v_2 \leq 1/2$ and $z = -x$ otherwise.

z represents a standard normal random draw.

Note that Step (vi) above corresponds to Step (iv) in Section 5.7.1.

Steps (i) – (vi) shown above represent the generator which yields one standard normal random draw.

When we want N standard normal random draws, Steps (iv) – (vi) should be repeated N times.

In Steps (iv) and (v), a random draw from $f(x)$ is generated based on Ω_j for $j =$

$1, 2, \cdots, n$.

Gamma Distribution: $G(\alpha, 1)$ **for** $0 < \alpha \leq 1$**:** When $X \sim G(\alpha, 1)$, the density function of X is given by:

$$f(x) = \begin{cases} \dfrac{1}{\Gamma(\alpha)} x^{\alpha-1} e^{-x}, & \text{for } 0 < x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

The sampling density is taken as:

$$f_*(x) = \frac{e}{\alpha + e} \alpha x^{\alpha-1} I_1(x) + \frac{\alpha}{\alpha + e} e^{-x+1} I_2(x),$$

which is the same function as in `gammarnd2` of Section 5.7.1, where both $I_1(x)$ and $I_2(x)$ denote the indicator functions defined in Section 5.7.1.

The probability weights are given by:

$$
\begin{aligned}
\omega(x_i^*) &= \frac{q(x_i^*)}{\sum_{j=1}^n q(x_j^*)} = \frac{f(x_i^*)/f_*(x_i^*)}{\sum_{j=1}^n f(x_j^*)/f_*(x_j^*)} \\
&= \frac{x_i^{*\alpha-1}e^{-x_i^*}/(x_i^{*\alpha-1}I_1(x_i^*) + e^{-x_i^*}I_2(x_i^*))}{\sum_{j=1}^n x_j^{*\alpha-1}e^{-x_j^*}/(x_j^{*\alpha-1}I_1(x_j^*) + e^{-x_j^*}I_2(x_j^*))},
\end{aligned}
$$

for $i = 1, 2, \cdots, n$.

The cumulative distribution function of $f_*(x)$ is represented as:

$$F_*(x) = \begin{cases} \dfrac{e}{\alpha + e} x^{\alpha}, & \text{if } 0 < x \leq 1, \\[2ex] \dfrac{e}{\alpha + e} + \dfrac{\alpha}{\alpha + e}(1 - e^{-x+1}), & \text{if } x > 1. \end{cases}$$

Therefore, x_i^* can be generated by utilizing both the composition method and the inverse transform method.

Given x_i^*, compute $\omega(x_i^*)$ for $i = 1, 2, \cdots, n$, and take $x = x_i^*$ with probability $\omega(x_i^*)$.

Summarizing above, the random number generation procedure for the gamma distribution is given by:

(i) Generate uniform random draws u_i, $i = 1, 2, \cdots, n$, from $U(0,1)$, and set $x_i^* = ((\alpha/e + 1)u_i)^{1/\alpha}$ and $\omega(x_i^*) = e^{-x_i^*}$ if $u_i \le e/(\alpha + e)$ and take $x_i^* = -\log((1/e+1/\alpha)(1-u_i))$ and $\omega(x_i^*) = x_i^{*\alpha-1}$ if $u_i > e/(\alpha+e)$ for $i = 1, 2, \cdots, n$.

(ii) Compute $\Omega_i = \sum_{j=1}^{i} \omega(x_j^*)$ for $i = 1, 2, \cdots, n$, where $\Omega_0 = 0$.

(iii) Generate a uniform random draw v from $U(0,1)$, and take $x = x_i^*$ when $\Omega_{i-1} \le v < \Omega_i$.

As mentioned above, this algorithm yields one random draw.

If we want N random draws, Step (iii) should be repeated N times.

Beta Distribution: The beta distribution with parameters α and β is of the form:

$$f(x) = \begin{cases} \dfrac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}, & \text{for } 0 < x < 1, \\ 0, & \text{otherwise.} \end{cases}$$

The sampling density is taken as:

$$f_*(x) = \begin{cases} 1, & \text{for } 0 < x < 1, \\ 0, & \text{otherwise,} \end{cases}$$

which represents the uniform distribution between zero and one.

The probability weights $\omega(x_i^*)$, $i = 1, 2, \cdots, n$, are given by:

$$\omega(x_i^*) = \frac{q(x_i^*)}{\sum_{j=1}^n q(x_j^*)} = \frac{f(x_i^*)/f_*(x_i^*)}{\sum_{j=1}^n f(x_j^*)/f_*(x_j^*)} = \frac{x_i^{*\alpha-1}(1-x_i^*)^{\beta-1}}{\sum_{j=1}^n x_j^{*\alpha-1}(1-x_j^*)^{\beta-1}}.$$

Therefore, to generate a random draw from $f(x)$, first generate x_i^*, $i = 1, 2, \cdots, n$, from $U(0, 1)$, second compute $\omega(x_i^*)$ for $i = 1, 2, \cdots$,n, and finally take $x = x_i^*$ with probability $\omega(x_i^*)$.

We have shown three examples of the importance resampling procedure in this section.

One of the advantages of importance resampling is that it is really easy to construct a Fortran source code.

However, the disadvantages are that (i) importance resampling takes quite a long time because we have to obtain all the probability weights in advance and (ii) importance resampling requires a great amount of storages for x_i^* and Ω_i for $i = 1, 2, \cdots, n$.

5.7.3 Metropolis-Hastings Algorithm (メトロポリスーハスティングス・アルゴリズム)

This section is based on Geweke and Tanizaki (2003), where three sampling distributions are compared with respect to precision of the random draws from the target density $f(x)$.

The **Metropolis-Hastings algorithm** is also one of the sampling methods to generate random draws from any target density $f(x)$, utilizing sampling density $f_*(x)$, even in the case where it is not easy to generate random draws from the target density.

Let us define the acceptance probability by:

$$\omega(x_{i-1}, x^*) = \min\Big(\frac{q(x^*)}{q(x_{i-1})}, 1\Big) = \min\Big(\frac{f(x^*)/f_*(x^*)}{f(x_{i-1})/f_*(x_{i-1})}, 1\Big),$$

where $q(\cdot)$ is defined as equation (1).

By the Metropolis-Hastings algorithm, a random draw from $f(x)$ is generated in the following way:

(i) Take the initial value of x as x_{-M}.

(ii) Generate x^* from $f_*(x)$ and compute $\omega(x_{i-1}, x^*)$ given x_{i-1}.

(iii) Set $x_i = x^*$ with probability $\omega(x_{i-1}, x^*)$ and $x_i = x_{i-1}$ otherwise.

(iv) Repeat Steps (ii) and (iii) for $i = -M+1, -M+2, \cdots, 1$.

In the above algorithm, x_1 is taken as a random draw from $f(x)$.

When we want more random draws (say, N), we replace Step (iv) by Step (iv)', which is represented as follows:

(iv)' Repeat Steps (ii) and (iii) for $i = -M+1, -M+2, \cdots, N$.

When we implement Step (iv)', we can obtain a series of random draws x_{-M}, x_{-M+1}, $\cdots$, x_0, x_1, x_2, $\cdots$, x_N, where x_{-M}, x_{-M+1}, $\cdots$, x_0 are discarded from further consideration.

The last N random draws are taken as the random draws generated from the target density $f(x)$.

Thus, N denotes the number of random draws.

M is sometimes called the **burn-in period**.

We can justify the above algorithm given by Steps (i) – (iv) as follows.

The proof is very similar to the case of rejection sampling in Section 5.7.1.

We show that x_i is the random draw generated from the target density $f(x)$ under the assumption x_{i-1} is generated from $f(x)$.

Let U be the uniform random variable between zero and one, X be the random variable which has the density function $f(x)$ and x^* be the realization (i.e., the random draw) generated from the sampling density $f_*(x)$.

Consider the probability $P(X \leq x|U \leq \omega(x_{i-1}, x^*))$, which should be the cumulative distribution of X, i.e., $F(x)$.

The probability $P(X \leq x|U \leq \omega(x_{i-1}, x^*))$ is rewritten as follows:

$$P(X \leq x|U \leq \omega(x_{i-1}, x^*)) = \frac{P(X \leq x, U \leq \omega(x_{i-1}, x^*))}{P(U \leq \omega(x_{i-1}, x^*))},$$

where the numerator is represented as:

$$\begin{aligned}
P(X \leq x, U \leq \omega(x_{i-1}, x^*)) &= \int_{-\infty}^{x} \int_{0}^{\omega(x_{i-1},t)} f_{u,*}(u,t)\, \mathrm{d}u\, \mathrm{d}t \\
&= \int_{-\infty}^{x} \int_{0}^{\omega(x_{i-1},t)} f_u(u) f_*(t)\, \mathrm{d}u\, \mathrm{d}t = \int_{-\infty}^{x} \Big(\int_{0}^{\omega(x_{i-1},t)} f_u(u)\, \mathrm{d}u \Big) f_*(t)\, \mathrm{d}t \\
&= \int_{-\infty}^{x} \Big(\int_{0}^{\omega(x_{i-1},t)} \mathrm{d}u \Big) f_*(t)\, \mathrm{d}t = \int_{-\infty}^{x} \Big[u \Big]_{0}^{\omega(x_{i-1},t)} f_*(t)\, \mathrm{d}t \\
&= \int_{-\infty}^{x} \omega(x_{i-1}, t) f_*(t)\, \mathrm{d}t = \int_{-\infty}^{x} \frac{f_*(x_{i-1}) f(t)}{f(x_{i-1})}\, \mathrm{d}t = \frac{f_*(x_{i-1})}{f(x_{i-1})} F(x)
\end{aligned}$$

and the denominator is given by:

$$P(U \leq \omega(x_{i-1}, x^*)) = P(X \leq \infty, U \leq \omega(x_{i-1}, x^*)) = \frac{f_*(x_{i-1})}{f(x_{i-1})} F(\infty) = \frac{f_*(x_{i-1})}{f(x_{i-1})}.$$

The density function of U is given by $f_u(u) = 1$ for $0 < u < 1$.

Let X^* be the random variable which has the density function $f_*(x)$.

In the numerator, $f_{u,*}(u, x)$ denotes the joint density of random variables U and X^*.

Because the random draws of U and X^* are independently generated, we have $f_{u,*}(u, x) = f_u(u) f_*(x) = f_*(x)$.

Thus, the first four equalities are derived.

Substituting the numerator and denominator shown above, we have the following equality:

$$P(X \leq x | U \leq \omega(x_{i-1}, x^*)) = F(x).$$

Thus, the x^* which satisfies $u \leq \omega(x_{i-1}, x^*)$ indicates a random draw from $f(x)$.

We set $x_i = x_{i-1}$ if $u \leq \omega(x_{i-1}, x^*)$ is not satisfied. x_{i-1} is already assumed to be a random draw from $f(x)$.

Therefore, it is shown that x_i is a random draw from $f(x)$.

See Gentle (1998) for the discussion above.

As in the case of rejection sampling and importance resampling, note that $f(x)$ may be a kernel of the target density, or equivalently, $f(x)$ may be proportional to the target density.

The same algorithm as Steps (i) – (iv) can be applied to the case where $f(x)$ is

proportional to the target density, because $f(x^*)$ is divided by $f(x_{i-1})$ in $\omega(x_{i-1}, x^*)$.
As a general formulation of the sampling density, instead of $f_*(x)$, we may take the sampling density as the following form: $f_*(x|x_{i-1})$, where a candidate random draw x^* depends on the (i – 1)th random draw, i.e., x_{i-1}.
For choice of the sampling density $f_*(x|x_{i-1})$, Chib and Greenberg (1995) pointed out as follows.
$f_*(x|x_{i-1})$ should be chosen so that the chain travels over the support of $f(x)$, which implies that $f_*(x|_{i-1})$ should not have too large variance and too small variance, compared with $f(x)$.

See, for example, Smith and Roberts (1993), Bernardo and Smith (1994), O'Hagan (1994), Tierney (1994), Geweke (1996), Gamerman (1997), Robert and Casella (1999) and so on for the Metropolis-Hastings algorithm.

As an alternative justification, note that the Metropolis-Hastings algorithm is formulated as follows:

$$f_i(u) = \int f^*(u|v) f_{i-1}(v)\,\mathrm{d}v,$$

where $f^*(u|v)$ denotes the transition distribution, which is characterized by Step (iii).

x_{i-1} is generated from $f_{i-1}(\cdot)$ and x_i is from $f^*(\cdot|x_{i-1})$.

x_i depends only on x_{i-1}, which is called the **Markov property**.

The sequence $\{\cdots, x_{i-1}, x_i, x_{i+1}, \cdots\}$ is called the **Markov chain**.

The Monte Carlo statistical methods with the sequence $\{\cdots, x_{i-1}, x_i, x_{i+1}, \cdots\}$ is called the **Markov chain Monte Carlo (MCMC)**.

From Step (iii), $f^*(u|v)$ is given by:

$$f^*(u|v) = \omega(v,u) f_*(u|v) + \Big(1 - \int \omega(v,u) f_*(u|v)\,\mathrm{d}u\Big) p(u), \tag{2}$$

where $p(x)$ denotes the following probability function:

$$p(u) = \begin{cases} 1, & \text{if } u = v, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, x is generated from $f_*(u|v)$ with probability $\omega(v,u)$ and from $p(u)$ with probability $1 - \int \omega(v,u) f_*(u|v)\,\mathrm{d}u$.

Now, we want to show $f_i(u) = f_{i-1}(u) = f(u)$ as i goes to infinity, which implies that both x_i and x_{i-1} are generated from the invariant distribution function $f(u)$ for sufficiently large i.

To do so, we need to consider the condition satisfying the following equation:

$$f(u) = \int f^*(u|v) f(v)\,\mathrm{d}v. \tag{3}$$

Equation (3) holds if we have the following equation:

$$f^*(v|u) f(u) = f^*(u|v) f(v), \tag{4}$$

which is called the **reversibility condition**.

By taking the integration with respect to v on both sides of equation (4), equation (3) is obtained.

Therefore, we have to check whether the $f^*(u|v)$ shown in equation (2) satisfies equation (4).

It is straightforward to verify that

$$\omega(v,u) f_*(u|v) f(v) = \omega(u,v) f_*(v|u) f(u),$$
$$\Big(1 - \int \omega(v,u) f_*(u|v)\,\mathrm{d}u\Big) p(u) f(v) = \Big(1 - \int \omega(u,v) f_*(v|u)\,\mathrm{d}v\Big) p(v) f(u).$$

Thus, as i goes to infinity, x_i is a random draw from the target density $f(\cdot)$.

If x_i is generated from $f(\cdot)$, then x_{i+1} is also generated from $f(\cdot)$.

Therefore, all the x_i, x_{i+1}, x_{i+2}, $\cdots$ are taken as random draws from the target density $f(\cdot)$.

The requirement for uniform convergence of the Markov chain is that the chain should be **irreducible** and **aperiodic**.

See, for example, Roberts and Smith (1993).

Let $C_i(x_0)$ be the set of possible values of x_i from starting point x_0.

If there exist two possible starting values, say x^* and x^{**}, such that $C_i(x^*) \cap C_i(x^{**}) = \emptyset$ (i.e., empty set) for all i, then the same limiting distribution cannot be reached from both starting points.

Thus, in the case of $C_i(x^*) \cap C_i(x^{**}) = \emptyset$, the convergence may fail.

A Markov chain is said to be **irreducible** if there exists an i such that $P(x_i \in C|x_0) > 0$ for any starting point x_0 and any set C such that $\int_C f(x)\, \mathrm{d}x > 0$.

The irreducible condition ensures that the chain can reach all possible x values from any starting point.

Moreover, as another case in which convergence may fail, if there are two disjoint set C^1 and C^2 such that $x_{i-1} \in C^1$ implies $x_i \in C^2$ and $x_{i-1} \in C^2$ implies $x_i \in C^1$, then the chain oscillates between C^1 and C^2 and we again have $C_i(x^*) \cap C_i(x^{**}) = \emptyset$

for all i when $x^* \in C^1$ and $x^{**} \in C^2$.

Accordingly, we cannot have the same limiting distribution in this case, either.

It is called **aperiodic** if the chain does not oscillate between two sets C^1 and C^2 or cycle around a partition $C^1, C^2, \cdots, C^r$ of r disjoint sets for $r > 2$.

See O'Hagan (1994) for the discussion above.

For the Metropolis-Hastings algorithm, x_1 is taken as a random draw of x from $f(x)$ for sufficiently large M.

To obtain N random draws, we need to generate $M + N$ random draws.

Moreover, clearly we have $\text{Cov}(x_{i-1}, x_i) > 0$, because x_i is generated based on x_{i-1}

in Step (iii).

Therefore, for precision of the random draws, the Metropolis-Hastings algorithm gives us the worst random number of the three sampling methods. i.e., rejection sampling in Section 5.7.1, importance resampling in Section 5.7.2 and the Metropolis-Hastings algorithm in this section.

Based on Steps (i) – (iii) and (iv)', under some conditions the basic result of the Metropolis-Hastings algorithm is as follows:

$$\frac{1}{N}\sum_{i=1}^{N} g(x_i) \longrightarrow \mathrm{E}(g(x)) = \int g(x) f(x)\,\mathrm{d}x, \qquad \text{as } N \longrightarrow \infty,$$

where $g(\cdot)$ is a function, which is representatively taken as $g(x) = x$ for mean and

$g(x) = (x - \overline{x})^2$ for variance.

$\overline{x}$ denotes $\overline{x} = (1/N) \sum_{i=1}^{N} x_i$.

Thus, it is shown that $(1/N) \sum_{i=1}^{N} g(x_i)$ is a consistent estimate of $\mathrm{E}(g(x))$, even though $x_1, x_2, \cdots, x_N$ are mutually correlated.

As an alternative random number generation method to avoid the positive correlation, we can perform the case of $N = 1$ as in the above procedures (i) – (iv) N times in parallel, taking different initial values for x_{-M}.

In this case, we need to generate $M+1$ random numbers to obtain one random draw from $f(x)$.

That is, N random draws from $f(x)$ are based on $N(1 + M)$ random draws from $f_*(x|x_{i-1})$.

Thus, we can obtain mutually independently distributed random draws.

For precision of the random draws, the alternative Metropolis-Hastings algorithm should be similar to rejection sampling.

However, this alternative method is too computer-intensive, compared with the above procedures (i) – (iii) and (iv)’, which takes more time than rejection sampling in the case of $M > N_R$.

Furthermore, the sampling density has to satisfy the following conditions:

(i) we can quickly and easily generate random draws from the sampling density and

(ii) the sampling density should be distributed with the same range as the target density.

See, for example, Geweke (1992) and Mengersen, Robert and Guihenneuc-Jouyaux (1999) for the MCMC convergence diagnostics.
Since the random draws based on the Metropolis-Hastings algorithm heavily depend on choice of the sampling density, we can see that the Metropolis-Hastings algorithm has the problem of specifying the sampling density, which is the crucial

criticism.

Several generic choices of the sampling density are discussed by Tierney (1994) and Chib and Greenberg (1995).

We can consider several candidates for the sampling density $f_*(x|x_{i-1})$, i.e., Sampling Densities I – III.

3.4.1.1 Sampling Density I (Independence Chain) For the sampling density, we have started with $f_*(x)$ in this section.

Thus, one possibility of the sampling density is given by: $f_*(x|x_{i-1}) = f_*(x)$, where

$f_*(\cdot)$ does not depend on x_{i-1}.

This sampling density is called the **independence chain**.

For example, it is possible to take $f_*(x) = N(\mu_*, \sigma_*^2)$, where μ_* and σ_*^2 are the hyper-parameters.

Or, when x lies on a certain interval, say (a, b), we can choose the uniform distribution $f_*(x) = 1/(b-a)$ for the sampling density.

3.4.1.2 Sampling Density II (Random Walk Chain) We may take the sampling density called the **random walk chain**, i.e., $f_*(x|x_{i-1}) = f_*(x - x_{i-1})$.

Representatively, we can take the sampling density as $f_*(x|x_{i-1}) = N(x_{i-1}, \sigma_*^2)$, where σ_*^2 denotes the hyper-parameter.

Based on the random walk chain, we have a series of the random draws which follow the random walk process.

3.4.1.3 Sampling Density III (Taylored Chain)

The alternative sampling distribution is based on approximation of the log-kernel (see Geweke and Tanizaki (1999, 2001, 2003)), which is a substantial extension of the **Taylored chain** discussed in Chib, Greenberg and Winkelmann (1998).

Let $p(x) = \log(f(x))$, where $f(x)$ may denote the kernel which corresponds to the target density.

Approximating the log-kernel $p(x)$ around x_{i-1} by the second order Taylor series expansion, $p(x)$ is represented as:

$$p(x) \approx p(x_{i-1}) + p'(x_{i-1})(x - x_{i-1}) + \frac{1}{2}p''(x_{i-1})(x - x_{i-1})^2, \tag{5}$$

where $p'(\cdot)$ and $p''(\cdot)$ denote the first- and second-derivatives.

Depending on the values of $p'(x)$ and $p''(x)$, we have the four cases, i.e., Cases 1 – 4, which are classified by (i) $p''(x) < -\epsilon$ in Case 1 or $p''(x) \geq -\epsilon$ in Cases 2 – 4 and (ii) $p'(x) < 0$ in Case 2, $p'(x) > 0$ in Case 3 or $p'(x) = 0$ in Case 4.

Geweke and Tanizaki (2003) suggested introducing ϵ into the Taylored chain discussed in Geweke and Tanizaki (1999, 2001).
Note that $\epsilon = 0$ is chosen in Geweke and Tanizaki (1999, 2001).
To improve precision of random draws, ϵ should be a positive value, which will be discussed later in detail (see Remark 1 for ϵ).

Case 1: $\boldsymbol{p''(x_{i-1}) < -\epsilon}$: Equation (5) is rewritten by:

$$p(x) \approx p(x_{i-1}) - \frac{1}{2}\Big(\frac{1}{-1/p''(x_{i-1})}\Big)\Big(x - (x_{i-1} - \frac{p'(x_{i-1})}{p''(x_{i-1})})\Big)^2 + r(x_{i-1}),$$

where $r(x_{i-1})$ is an appropriate function of x_{i-1}.

Since $p''(x_{i-1})$ is negative, the second term in the right-hand side is equivalent to the exponential part of the normal density.

Therefore, $f_*(x|x_{i-1})$ is taken as $N(\mu_*, \sigma_*^2)$, where $\mu_* = x_{i-1} - p'(x_{i-1})/p''(x_{i-1})$ and $\sigma_*^2 = -1/p''(x_{i-1})$.

Case 2: $\boldsymbol{p''(x_{i-1}) \geq -\epsilon}$ **and** $\boldsymbol{p'(x_{i-1}) < 0}$: Perform linear approximation of $p(x)$.

Let x^+ be the nearest mode with $x^+ < x_{i-1}$.

Then, $p(x)$ is approximated by a line passing between x^+ and x_{i-1}, which is written as:

$$p(x) \approx p(x^+) + \frac{p(x^+) - p(x_{i-1})}{x^+ - x_{i-1}}(x - x^+).$$

From the second term in the right-hand side, the sampling density is represented as the exponential distribution with $x > x^+ - d$, i.e., $f_*(x|x_{i-1}) = \lambda \exp\Big(-\lambda(x - (x^+ - d))\Big)$ if $x^+ - d < x$ and $f_*(x|x_{i-1}) = 0$ otherwise, where λ is defined as:

$$\lambda = \left| \frac{p(x^+) - p(x_{i-1})}{x^+ - x_{i-1}} \right|.$$

d is a positive value, which will be discussed later (see Remark 2 for d).

Thus, a random draw x^* from the sampling density is generated by $x^* = w +$

$(x^+ - d)$, where w represents the exponential random variable with parameter λ.

<u>**Case 3:**</u> $\boldsymbol{p''(x_{i-1}) \geq -\epsilon}$ **and** $\boldsymbol{p'(x_{i-1}) > 0}$: Similarly, perform linear approximation of $p(x)$ in this case.

Let x^+ be the nearest mode with $x_{i-1} < x^+$.

Approximation of $p(x)$ is exactly equivalent to that of Case 2.

Taking into account $x < x^+ + d$, the sampling density is written as: $f_*(x|x_{i-1}) = \lambda \exp\Big(-\lambda((x^+ + d) - x)\Big)$ if $x < x^+ + d$ and $f_*(x|x_{i-1}) = 0$ otherwise.

Thus, a random draw x^* from the sampling density is generated by $x^* = (x^+ + d) - w$, where w is distributed as the exponential random variable with parameter λ.

Case 4: $\boldsymbol{p''(x_{i-1}) \geq -\epsilon}$ **and** $\boldsymbol{p'(x_{i-1}) = 0}$: In this case, $p(x)$ is approximated as a uniform distribution at the neighborhood of x_{i-1}.

As for the range of the uniform distribution, we utilize the two appropriate values x^+ and x^{++}, which satisfies $x^+ < x < x^{++}$.

When we have two modes, x^+ and x^{++} may be taken as the modes.

Thus, the sampling density $f_*(x|x_{i-1})$ is obtained by the uniform distribution on the interval between x^+ and x^{++}, i.e., $f_*(x|x_{i-1}) = 1/(x^{++} - x^+)$ if $x^+ < x < x^{++}$ and $f_*(x|x_{i-1}) = 0$ otherwise.

Thus, for approximation of the kernel, all the possible cases are given by Cases 1 – 4, depending on the values of $p'(\cdot)$ and $p''(\cdot)$.

Moreover, in the case where x is a vector, applying the procedure above to each element of x, Sampling III is easily extended to multivariate cases.

Finally, we discuss about ϵ and d in the following remarks.

Remark 1: ϵ in Cases 1 – 4 should be taken as an appropriate positive number.

It may seem more natural to take $\epsilon = 0$, rather than $\epsilon > 0$.

The reason why $\epsilon > 0$ is taken is as follows.

Consider the case of $\epsilon = 0$.

When $p''(x_{i-1})$ is negative and it is very close to zero, variance σ_*^2 in Case 1 becomes extremely large because of $\sigma_*^2 = -1/p''(x_{i-1})$.

In this case, the obtained random draws are too broadly distributed and accordingly they become unrealistic, which implies that we have a lot of outliers.

To avoid this situation, ϵ should be positive.

It might be appropriate that ϵ should depend on variance of the target density, because ϵ should be small if variance of the target density is large.

Thus, in order to reduce a number of outliers, $\epsilon > 0$ is recommended.

Remark 2: For d in Cases 2 and 3, note as follows.

As an example, consider the unimodal density in which we have Cases 2 and 3.

Let x^+ be the mode.

We have Case 2 in the right-hand side of x^+ and Case 3 in the left-hand side of x^+.

In the case of $d = 0$, we have the random draws generated from either Case 2 or 3.

In this situation, the generated random draw does not move from one case to another.

In the case of $d > 0$, however, the distribution in Case 2 can generate a random draw in Case 3.

That is, for positive d, the generated random draw may move from one case to another, which implies that the irreducibility condition of the MH algorithm is guaranteed.

Normal Distribution: $N(0,1)$**:** As in Sections 5.7.1 and 5.7.2, we consider an example of generating standard normal random draws based on the half-normal distribution:

$$f(x) = \begin{cases} \dfrac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

As in Sections 5.7.1 and 5.7.2, we take the sampling density as the following exponential distribution:

$$f_*(x) = \begin{cases} e^{-x}, & \text{for } 0 \le x < \infty, \\ 0, & \text{otherwise,} \end{cases}$$

which is the independence chain, i.e., $f_*(x|x_{i-1}) = f_*(x)$.

Then, the acceptance probability $\omega(x_{i-1}, x^*)$ is given by:

$$\begin{aligned}\omega(x_{i-1}, x^*) &= \min\Big(\frac{f(x^*)/f_*(x^*)}{f(x_{i-1})/f_*(x_{i-1})}, 1\Big)\\ &= \min\Big(\exp(-\frac{1}{2}x^{*2} + x^* + \frac{1}{2}x_{i-1}^2 - x_{i-1}), 1\Big).\end{aligned}$$

Utilizing the Metropolis-Hastings algorithm, the standard normal random number generator is shown as follows:

(i) Take an appropriate initial value of x as x_{-M} (for example, $x_{-M} = 0$).

(ii) Set $y_{i-1} = |x_{i-1}|$.

(iii) Generate a uniform random draw u_1 from $U(0,1)$ and compute $\omega(y_{i-1}, y^*)$ where $y^* = -\log(u_1)$.

(iv) Generate a uniform random draw u_2 from $U(0,1)$, and set $y_i = y^*$ if $u_2 \leq \omega(y_{i-1}, y^*)$ and $y_i = y_{i-1}$ otherwise.

(v) Generate a uniform random draw u_3 from $U(0,1)$, and set $x_i = y_i$ if $u_3 \leq 0.5$ and $x_i = -y_i$ otherwise.

(vi) Repeat Steps (ii) – (v) for $i = -M+1, -M+2, \cdots, 1$.

y_1 is taken as a random draw from $f(x)$. M denotes the burn-in period.

If a lot of random draws (say, N random draws) are required, we replace Step (vi) by Step (vi)' represented as follows:

(vi)' Repeat Steps (ii) – (v) for $i = -M + 1, -M + 2, \cdots, N$.

In Steps (ii) – (iv), a half-normal random draw is generated.

Note that the absolute value of x_{i-1} is taken in Step (ii) because the half-normal random draw is positive.

In Step (v), the positive or negative sign is randomly assigned to y_i.

Gamma Distribution: $G(\alpha, 1)$ **for** $0 < \alpha \leq 1$**:** When $X \sim G(\alpha, 1)$, the density function of X is given by:

$$f(x) = \begin{cases} \dfrac{1}{\Gamma(\alpha)} x^{\alpha-1} e^{-x}, & \text{for } 0 < x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

As in `gammarnd2` of Sections 5.7.1 and `gammarnd4` of 5.7.2, the sampling density is taken as:

$$f_*(x) = \frac{e}{\alpha + e} \alpha x^{\alpha-1} I_1(x) + \frac{\alpha}{\alpha + e} e^{-x+1} I_2(x),$$

where both $I_1(x)$ and $I_2(x)$ denote the indicator functions defined in Section 5.7.1.

Then, the acceptance probability is given by:

$$\omega(x_{i-1}, x^*) = \min\Big(\frac{q(x^*)}{q(x_{i-1})}, 1\Big) = \min\Big(\frac{f(x^*)/f_*(x^*)}{f(x_{i-1})/f_*(x_{i-1})}, 1\Big)$$
$$= \min\Big(\frac{x^{*\alpha-1}e^{-x^*}/(x^{*\alpha-1}I_1(x^*) + e^{-x^*}I_2(x^*))}{x_{i-1}^{\alpha-1}e^{-x_{i-1}}/(x_{i-1}^{\alpha-1}I_1(x_{i-1}) + e^{-x_{i-1}}I_2(x_{i-1}))}, 1\Big).$$

As shown in Section 5.7.1, the cumulative distribution function of $f_*(x)$ is represented as:

$$F_*(x) = \begin{cases} \dfrac{e}{\alpha + e}x^{\alpha}, & \text{if } 0 < x \leq 1, \\ \dfrac{e}{\alpha + e} + \dfrac{\alpha}{\alpha + e}(1 - e^{-x+1}), & \text{if } x > 1. \end{cases}$$

Therefore, a candidate of the random draw, i.e., x^*, can be generated from $f_*(x)$, by utilizing both the composition method and the inverse transform method.

Then, using the Metropolis-Hastings algorithm, the gamma random number generation method is shown as follows.

(i) Take an appropriate initial value as x_{-M}.

(ii) Generate a uniform random draw u_1 from $U(0, 1)$, and set $x^* = ((\alpha/e+1)u_1)^{1/\alpha}$ if $u_1 \leq e/(\alpha + e)$ and $x^* = -\log((1/e + 1/\alpha)(1 - u_1))$ if $u_1 > e/(\alpha + e)$.

(iii) Compute $\omega(x_{i-1}, x^*)$.

(iv) Generate a uniform random draw u_2 from $U(0, 1)$, and set $x_i = x^*$ if $u_2 \leq$

$\omega(x_{i-1}, x^*)$ and $x_i = x_{i-1}$ otherwise.

(v) Repeat Steps (ii) – (iv) for $i = -M + 1, -M + 2, \cdots, 1$.

For sufficiently large M, x_1 is taken as a random draw from $f(x)$. u_1 and u_2 should be independently distributed.

M denotes the burn-in period. If we need a lot of random draws (say, N random draws), replace Step (v) by Step (v)', which is given by:

(v)' Repeat Steps (ii) – (iv) for $i = -M + 1, -M + 2, \cdots, N$.

Beta Distribution: The beta distribution with parameters α and β is of the form:

$$f(x) = \begin{cases} \dfrac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}, & \text{for } 0 < x < 1, \\ 0, & \text{otherwise.} \end{cases}$$

The sampling density is taken as:

$$f_*(x) = \begin{cases} 1, & \text{for } 0 < x < 1, \\ 0, & \text{otherwise,} \end{cases}$$

which represents the uniform distribution between zero and one.

The probability weights $\omega(x_i^*)$, $i = 1, 2, \cdots, n$, are given by:

$$\omega(x_{i-1}, x^*) = \min\Big(\frac{f(x^*)/f_*(x^*)}{f(x_{i-1})/f_*(x_{i-1})}, 1\Big) = \min\Big(\Big(\frac{x^*}{x_{i-1}}\Big)^{\alpha-1}\Big(\frac{1-x^*}{1-x_{i-1}}\Big)^{\beta-1}, 1\Big).$$

Then, utilizing the Metropolis-Hastings algorithm, the random draws are generated as follows.

(i) Take an appropriate initial value as x_{-M}.

(ii) Generate a uniform random draw x^* from $U(0,1)$, and compute $\omega(x_{i-1}, x^*)$.

(iii) Generate a uniform random draw u from $U(0,1)$, which is independent of x^*, and set $x_i = x^*$ if $u \leq \omega(x_{i-1}, x^*)$ and $x_i = x_{i-1}$ if $u > \omega(x_{i-1}, x^*)$.

(iv) Repeat Steps (ii) and (iii) for $i = -M+1, -M+2, \cdots, 1$.

For sufficiently large M, x_1 is taken as a random draw from $f(x)$.

M denotes the burn-in period.

If we want a lot of random draws (say, N random draws), replace Step (iv) by Step (iv)', which is represented as follows:

(iv)' Repeat Steps (ii) and (iii) for $i = -M+1, -M+2, \cdots, N$.