Monte Carlo methods for Statistical Inference: Generation of Random Numbers

Hung Chen

hchen@math.ntu.edu.tw

Department of Mathematics

National Taiwan University

25th February 2004

Meet at NS 104 On Wednesday from 9:10 to 12.

Outline

- Introduction
  - Inverse CDF Method
  - Generate Discrete Random Variates
  - Limitation
- Classical Uniform Variate Generator
  - Lehmer congruential generator (LCG)
  - Tausworthe feedback shift register generator
  - Combination of random number generators
- Non-uniform Variate Generation
  - Simulating Stochastic Processes
    - Poisson process

- ∗ **Brownian motion**
- **– Acceptance/Rejection Methods**
- **– Bayesian Analysis**
- **– Simulating Multivariate Random Variates**

- **Gibbs Sampling and MCMC**
  - **– Brief Introduction on Markov Chain**
  - **– MCMC**
  - **– Gibbs sampling**

Classical Uniform Variate Generator

Simulation is used heavily when analytical study of a statistical procedure becomes intractable.

- **Simulation of random variables and random processes using computers is among the fastest growing areas of computational statistics.**
- **Many statistical techniques rely on simulating random variables.**
  - **One traditional area is the use of random numbers to sample from a population.**
  - **More recent applications include simulation of high-dimensional, complex stochastic systems that are beyond analytical studies.**

- **In many practical situations the probability distributions are far too complicated to analyze and often it is easier to simulate these distributions on computers and the resulting samples can be analyzed instead.**

- **The study of a random variable through simulations is becoming a powerful tool in the hands of the statisticians.**

Monte Carlo experimentation is the use of simulated random numbers to estimate some functional of a probability distribution.

- **Building block in any simulation study is** *non-uniform variate generation*.
  - **Many algorithms are available.**

– **Example: Generate normal random variable.**

  * **Box-Muller method (Polar method)**
    If X and Y are independent and standard normal random variables then for

$$\theta = \tan^{-1}\left(\frac{Y}{X}\right), \quad R = \sqrt{X^2 + Y^2}$$

    θ is uniform in $[0, 2\pi]$ and R^2 is exponential with mean 2.
    (0) U_1, U_2 **iid** $U(0,1)$
    (1) $X_1 = (-2\ln U_1)^{1/2}\cos(2\pi U_2)$
    (2) $X_2 = (-2\ln U_1)^{1/2}\sin(2\pi U_2)$
    (3) X_1, X_2 **iid** $N(0,1)$
  * **Inverse method**
    If $X \sim F$, then $F(X) \sim U(0,1)$.

- **In the above methods, it assumes that we can produce an** $endless$ **flow of a iid** $uniform$ **random variate generators.**
  - ∗ **On the computer, we generally settle for** $pseudo$-$random$ **numbers, that is, numbers that appear to be random but actually deterministic.**

- **CDF transformation method**
  $X = F^{-}(U)$**,** $U \sim U(0,1)$ **where**

$$F^{-}(u) = \inf\{x \mid F(x) \geq u\}$$

  is the generalized inverse of the cdf F**.**
  - **For a standard exponential random variable, the transformation**

$$X = \log(U)$$

    yields one exponential for each uniform variable.

- **How to simulate the process of flipping a coin with probability of head p?**

- **For a discrete random variable, although the inverse of the cdf does not exist, the inverse cdf method can still be used.**
  - **The value of the discrete random variable is chosen as the smallest value within its countable range such that the cdf is no less than the value of the uniform variate.**
- **For a multivariate random variable, the inverse cdf method yields a level curve in the range of the random variable; hence, the method is not directly useful for multivariate random variable.**
  - **Multivariate random variates can be generated us-**

ing the inverse cdf method first on a univariate marginal and then on a sequence of univariate conditionals.

Discrete Random Variables

A discrete random variable takes only a countable number of values with pre-defined probabilities.

- **A discrete random variable is characterized by its probability mass function defined as** $P(x_1) = p_1, P(x_2) = p_2, \ldots, P(x_n) = p_n, \ldots$ **such that such that for all** i**,** $0 \leq p_i \leq 1$**, and** $\sum_i p_i = 1$**.**
- **Commonly used discrete random variables are binomial, Poisson, geometric and negative-binomial. As an**
- **How do we generate a Poisson random variable with parameter** λ**?**

The probability mass function is given by:

$$p_i = \exp(-\lambda)\frac{\lambda^i}{i!}, \quad i = 0, 1, 2, \ldots.$$

Note that

$$\frac{P(X = i+1)}{P(X = i)} = \frac{\lambda}{i+1}.$$

$F_X(i+1)$ **can be written in the following interative form:**

$$F_X(i+1) = F_X(i) + P(X = i)\frac{\lambda}{i+1}.$$

The algorithm is

i. Generate U **according to** $U[0, 1]$**.**

ii. Set $i = 0$**,** $p = \exp(-\lambda)$**, and** $F = p$**.**

iii. If $U < F$**, set** $X = i$ **and stop.**

iv. Set $p = \lambda p/(i+1)$**,** $F = F + p$**,** $i = i + 1$

v. Go to Step (iii).

Definition For a given random variable, with a specified probability mass function $\{(x_i, p_i), i = 0, 1, 2, \ldots\}$**, the process of selecting a value** x_i **with probability** p_i **is called** $Simulation$**. If this selection is performed many times, generating a sequence** $\{X_j\}$**, then**

$$\frac{1}{n}\sum_{j=1}^{n} I_{X_j}(\{x_i\}) \to p_i.$$

Uniform Random Number Generation

- **Use algebraic methods to generate sequences of numbers that mimic the behavior of a uniform random variable.**
  - **These numbers are called pseudorandom numbers.**
  - **A uniform pseudorandom number generator is a mapping f that, starting from an initial value x_0, generates a sequence**

    $$x_0, f(x_0), f(f(x_0)), f(f(f(x_0))), \ldots.$$

    Since f is computed on a computer (without the use of random number generator!!), it is a deterministic mapping. That is, given x_0 the remaining

sequence is fixed everytime the sequence is computed.

The elements of such a sequence should have the following properties:

1. **The patterns between the numbers the appearing in a sequence should be minimized.**
2. **The correlation between the neighboring elements should be reasonably small.**
3. **The values should be distributed nearly uniformly over the whole the range of possible values.**
4. **The sequences should have large periods, where a period is defined to be duration after which a sequence repeats itself.**
5. **There exist a set of goodness of fit tests for test-**

ing the probability distributions associated with the observed random variables. The elements of a pseudorandom sequence should provide a reasonable performance in these goodness of fit tests.

- **No random number generator is capable of generating (a) uniform and (b) independent variates.**
- **Slight** $defect$ **in RNG may have dramatic effect on whole simulation study.**
  - **Deng and Chhikara (1992)**
  - **If** $U_1, U_2, \ldots, U_n$ **iid** $\sim U(0,1)$**,**

$$Z_n = \frac{\sum_{i=1}^{n} U_i - \frac{n}{2}}{\sqrt{n/12}} \approx N(0,1).$$

What if the assumption of iid and/or "$U(0,1)$**" fail?**

- **Classical uniform variate generator**
  Linear congruential generator [Lehmer (1951)]
  - $X_i = BX_{i-1} + A$ **mod** n**.**
  - $U_i = X_i/m$
  - **LCG has been used in almost all computer systems and packages.**
  - **Popular LCG (e.g., IMSL, SAS)**

  (a) $B = 16807$**,** $A = 0$**,** $m = 2^{31} - 1$**.**

  (b) Its period is $m - 1 \approx 2.1 \cdot 10^9$**.**
  - **Comments**
    - **Period (**$\leq m$**) depends on** B, A, m, X_0**.**
    - **The period is too short by today's standard. Large-scale simulation study is more and more common.**

  - * **uniformity in** 1**-dimensional space**
  - * **LCG cannot generate set of all lattice points in** k **space,** S_k**, for** $k \geq 2$**.**
  - * **Consider** $S_2 = \{(I, J) \mid 0 \leq I, J \leq m\}$ **and do plots of** (U_i, U_{i+1})**,** $i = 0, 1, 2, \ldots$
  - * **Insert p7 and 8 of Deng's note.**

- **Feedback shift register [Tausworthe (1965)]**
  - $a_j = \sum_{i=1}^{k} c_i a_{j-i}$**(mod** 2**) where** $a_i, c_i \in \{0, 1\}$**,** $c_k = 1$
  - **The** m**th random variate is the** d **bits binary num-**

ber

$$0.a_0a_1\ldots a_{d-1} \text{ base } 2$$
$$0.a_da_{d+1}\ldots a_{2d-1} \text{ base } 2$$
$$\vdots$$
$$0.a_{md}a_{md+1}\ldots a_{md+d-1} \text{ base } 2$$
$$\ldots$$

- **It can have an extremely long period, $2^k - 1$, (if c_i's are properly selected) for a large k,**
- **good theoretical k-space uniformity**
- $Poor$ **empirical performance**

- **Combination generators:**
  Wichmann and Hill (1982): Add three LCGs and take its fractional part.
  - $X_i = AX_{i-1}$ **mod** m_1

- $Y_i = BY_{i-1}$ **mod** m_2
- $Z_i = CZ_{i-1}$ **mod** m_3
- $U_i = X_i/m_1 + Y_i/m_2 + Z_i/m_3$ **mod** 1

Comments:

- **Period is** $LCM(m_1 - 1, m_2 - 1, m_3 - 1)$**.**
  For $m_1 = 30269, m_2 = 30307, m_3 = 30323$**, its period is** 6.95×10^{12}**.**
- **About** 3000 **times longer period than LCG-**16807**.**
- **About three times slower than LCG.**
- **No theoretical justification for uniformity provided.**

- **Statistical justification given in Deng and George (1990)**
  - **Suppose that** X_1 **and** X_2 **are independent r.v. over**

$[0, 1]$ **with pdfs** $f_1(x_1)$ **and** $f_2(x_2)$ **respectively.**

- $\mid f_1(x_1) - 1 \mid \leq \epsilon_1$**,** $\mid f_2(x_2) - 1 \mid \leq \epsilon_2$
- **Let** $Y = X_1 + X_2$ **mod** 1 **and denote its pdf by** $f(y)$**.**
- **Conclusion:** $\mid f(y) - 1 \mid \leq \epsilon_1 \epsilon_2$**.**

In general, $Y = \sum_{i=1}^{n} X_i$ **mod** 1 **and denote its pdf by** $f(y)$**. Then**

$$\mid f(y) - 1 \mid \leq \prod_{i=1}^{n} \epsilon_i.$$

Exponential and Poisson RVs

The exponential density function is defined by

$$f(x) = \begin{cases} \lambda \exp(-\lambda x), & \text{if } 0 \leq x < \infty, \\ 0, & \text{otherwise.} \end{cases}$$

Here λ is any positive constant, depending on the experiment.

- **The exponential density is often used describe experiments involving a question of the form: How long until something happens?**
- **For example, the exponential density is often used to study the time between emissions of particles from a radioactive source.**
- **"Memoryless" property:**
  Let T be an exponentially distributed random vari-

able with parameter λ.
It says that $P(T > r + s \mid T > r) = P(T > s)$.

There is a very important relationship between the exponential density and the Poisson distribution.

- Define $X_1, X_2, \ldots$ to be a sequence of independent exponentially distributed random variables with parameter λ.
- Think of X_i as denoting the amount of time between the ith and $(i+1)$st emissions of a particle by a radioactive source.
- Consider a time interval of length t, and we let Y denote the random variable which counts the number of emissions that occur in this time interval.
- Find the distribution function of Y (clearly, Y is a

discrete random variable).

- **Let S_n denote the sum $X_1 + X_2 + \ldots + X_n$, then it is easy to see that**

$$\begin{aligned} P(Y = n) &= P(S_n \leq t \text{ and } S_{n+1} > t) \\ &= P(S_n \leq t) - P(S_{n+1} \leq t). \end{aligned}$$

- **The density of S_n is given by the following formula:**

$$g_n(x) = \begin{cases} \lambda \frac{(\lambda x)^{n-1}}{(n-1)!} \exp(-\lambda x), & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

  It is a gamma density with parameters λ and n.

- **It is easy to show by induction on n that the cumu-**

lative distribution function of S_n is given by:

$$G_n(x) = \begin{cases} 1 - \exp(-\lambda x)\left(1 + \frac{\lambda x}{1!} + \cdots + \frac{(\lambda x)^{n-1}}{(n-1)!}\right), & \text{if } x > 0, \\ 0, & \text{otherwis} \end{cases}$$

We recognize easily that it is the probability of taking on the value n by a Poisson-distributed random variable, with parameter λt.

- The above relationship will allow us to simulate a Poisson distribution, once we have found a way to simulate an exponential density.
- To simulate a Poisson random variable W with parameter λ, we
  - Generate a sequence of values of an exponentially distributed random variable with the same param-

eter.

- **Keep track of the subtotals S_k of these values.**
- **We stop generating the sequence when the subtotal first exceeds λ.**
- **Assume that we find that $S_n \leq \lambda < S_{n+1}$. Then the value n is returned as a simulated value for W.**

Simulating Poisson Processes

A point process consisting of randomly occurring points in the plane is said to be a two-dimensional Poisson process having rate λ, if

1. **the number of points in any given region of area A is Poisson distributed with mean λA; and**
2. **the number of points in disjoint regions are independent.**

Let O be the origin in R^2 and R_i be the ith nearest Poisson point to O, $i \geq 1$ ($R_0 = O$).
It can be shown that

- **$(\pi R_i^2 - \pi R_{i-1}^2)$ are exponentially distributed with rate λ.**

- **By symmetry, the respective angles of the Poisson points are independent and uniform** $[0, 2\pi]$.

The following algorithm simulates a two-dimensional Poisson process in a ball of radius r **centered at** O, $C(r)$.

1. **Generate independent exponentials** $X_1, X_2, \ldots$ **with rate** 1**, stopping at**
$$N = \min\left\{n : \frac{X_1 + X_2 + \cdots + X_n}{\lambda\pi} > r^2\right\}$$
2. **if** $N = 1$**, stop, there are no points in** $C(r)$**. Otherwise, for** $i = 1, 2, \ldots, N-1$**, set**
$$R_i = \sqrt{(X_1 + X_2 + \cdots + X_i)/\lambda\pi}.$$
3. **Generate independent uniform** $[0, 1]$ **random variables** $U_1, U_2, \ldots, U_{N-1}$.

4. **Return the $N-1$ Poisson points in $C(r)$ whose polar coordinates are $(R_i, 2\pi U_i)$; $i = 1, \ldots, N-1$.**

Brownian motion

Finance Application:

As you may know something about the celebrated Black-Scholes formula of finance. The problem addressed by the formula is determining how much an "option" should cost. This option is called the "call" options.

- **A call option on a certain stock is the right to buy a share of the stock at a certain fixed price (the** $strike\ price$**) at a certain fixed time in the future (the** $maturity\ date$**).**
- **If I buy a call option from you, I am paying you a certain amount of money in return for the right to force you to sell me a share of the stock, if I want it, at the strike price,** K**, on the maturity date,** t_1**.**

- **The problem is, what is the $right$ amount of money for me to pay for this right?**
  - **The meaning of the term right here relates to the economic term arbitrage.**
  - **An arbitrage opportunity is the opportunity to make money instantly and without risk. That is, you get some money for sure, right now.**
  - **Such free lunches are not supposed to exist, or at least should be rare and short-lived.**
    The basic reason for believing this is that many people are looking for such opportunities to make money.
    - **If the price of commodity A were so low, for example, that some clever financial transaction in-**

volving buying commodity A and perhaps selling some others were guaranteed to make an instantaneous profit, then many eager arbitrage seekers would try to perform the transaction many times.

- **The resulting increased demand for commodity A would cause its price to increase, thereby destroying the arbitrage opportunity.**

- **It assume that there is a financial instrument called bond such that its "interest rate" or the "riskless" rate of return be r, that is, \$1 in a riskless investment today becomes \$$\exp(rt)$ at time t.**

$$\frac{dB(t)}{dt} = rB(t),$$

where $B(t)$ is the bond price at time t.

- **Let the stock price at time t be $X(t)$.**
- **A little thought shows that the value of the option at time t_1 is the random variable $(X(t_1) - K)_+$, since it makes sense for me to exercise the option if and only if $X(t_1) > K$.**
- **Let $Y(t)$ denote the magic, no-arbitrage price for the option that we are seeking.**
  Assume that $Y(t)$ may be expressed as some function $f(X(t), t)$ of $X(t)$ and t; our goal is to determine the function f.
- **Assume a simple probabilistic model for the evolution of the stock price: suppose X is the geometric Brownian motion having stochastic differential**

$$dX = \mu X dt + \sigma X dW.$$

Thus, X is the exponential of a Brownian motion with drift.

- **Note that the riskless investments change as exp(linear function), and stocks change as exp(Brownian motion).**
  What we are really assuming is that returns, that is, proportional changes in the stock price, are stationary and independent over different time intervals.

The formulation of this process was inspired by the physical phenomenon of Brownian motion, which is the irregular jiggling sort of movement exhibited by a small particle suspended in a fluid, named after the botanist Robert Brown who observed and studied it in 1827.

- **A physical explanation of Brownian motion was given**

by Einstein, who analyzed Brownian motion as the cumulative effect of innumerable collisions of the suspended particle with the molecules of the fluid.

- Einstein's analysis provided historically important support for the atomic theory of matter, which was still a matter of controversy at the time-shortly after 1900.
- The mathematical theory of Brownian motion was given a firm foundation by Norbert Wiener in 1923; the mathematical model we will study is also known as the "Wiener process."

Brownian motion and diffusions are used all the time in models in all sorts of fields, such as finance (in modeling the prices of stocks, for example), economics,

queueing theory, engineering, and biology.

- **Just as a pollen particle is continually buffeted by collisions with water molecules, the price of a stock is buffeted by the actions of many individual investors.**

Construction of Brownian motion on the time interval $[0, 1]$**:**

- **Connect-the-dots approach: At each stage of the construction we obtain a more and more detailed picture of a sample path.**
- $W(0) = 0$
- **For** $W(1)$**, we generate a** $N(0, 1)$ **random variable** Z_1 **and take** Z_1 **to be** $W(1)$ **since** $W(1) \sim N(0, 1)$**.**

- **Given that the path passes through the two points $(0,0)$ and $(1,Z_1)$, the conditional expectation is the linear interpolation $X^{(0)}(t) = Z_1 t$.**
  This will be our first crude approximation to a sample path.
- **Next let's simulate a value for $W(1/2)$.**
  - **Given the values we have already generated for $W(0)$ and $W(1)$, we know that $W(1/2) \sim N(Z_1/2, (1/2)(1/$**
  - **Generate another independent standard random variable Z_2 and take $W(1/2)$ to be $X^{(0)}(1/2) + (1/2)Z_2$.**
  - **Define the approximation $X^{(1)}$ to be the piecewise linear path joining the three points $(0,0)$, $(1/2, W(1/2))$, and $(1, W(1))$.**

- **Simulate $W(1/4)$ and $W(3/4)$.**
  - $E(W(t) \mid W(0), W(1/2), W(1)) = X^{(1)}(t)$
  - **Conditional variance of both $W(1/4)$ and $W(3/4)$ is $(1/4)(1/4)/(1/2) = 1/8$.**
  - **Generate two more independent standard random variables Z_3 and Z_4, and define**

$$W(1/4) = X^{(1)}(1/4) + \frac{1}{\sqrt{8}} Z_3,$$

$$W(3/4) = X^{(1)}(3/4) + \frac{1}{\sqrt{8}} Z_4.$$

  - **The approximation $X^{(2)}$ to be the piecewise linear interpolation of the simulated values we have obtained for the times 0, $1/4$, $1/2$, $3/4$, and 1.**
- **In general, to get from $X^{(n)}$ to $X^{(n+1)}$, we generate**

2^n **new standard normal random variables** $Z_{2^n+1}, Z_{2^n+2}, \ldots,$ **multiply these by the appropriate conditional standard deviation** $\sqrt{2^{-n-2}} = 2^{-(n/2)-1}$**, and add to the values** $X^{(n)}(1/2^{n+1}), X^{(n)}(3/2^{n+1}), \ldots, X^{(}n)(1-1/2^{n+1})$ **to get the new values** $X^{(n+1)}(1/2^{n+1}), X^{(n+1)}(3/2^{n+1}), \ldots, X$ $1)(1-1/2^{n+1})$**.**

- **Claim. With probability** 1**, the sequence of functions** $X^{(1)}, X^{(2)}, \ldots$ **converges uniformly over the interval** $[0,1]$**.**
  - **The limit of a uniformly convergent sequence of continuous functions is a continuous function.**
  - **To appreciate the need for uniformity of convergence in order to be guaranteed that the limit function is continuous, recall the following stan-**

dard example.

For $n = 1, 2, \ldots$, consider the function t^n for $t \in [0, 1]$. Then as $n \to 1$, this converges to 0 for all $t < 1$ whereas it converges to 1 for $t = 1$, so that the limit is not a continuous function.

- **Define the maximum difference M_n between $X^{(n+1)}$ and $X^{(n)}$ by**

$$M_n = \max_{t \in [0,1]} \mid X^{(n+1)}(t) - X^{(n)}(t) \mid .$$

- **Note that if $\sum M_n < \infty$, then the sequence of functions $X^{(1)}, X^{(2)}, \ldots$ converges uniformly over $[0, 1]$.**
- **It is sufficient to show that $P\{\sum M_n < \infty\} = 1$.**

– **Observe that**

$$M_n = 2^{-n/2-1} \max\{\mid Z_{2^n+1} \mid, \mid Z_{2^n+2} \mid, \ldots, \mid Z_{2^{n+1}} \mid\}.$$

– **Note that**

$$\sum_{n=1}^{\infty} P\{\mid Z_n \mid > \sqrt{c \log n}\} \leq 2\frac{1}{\sqrt{2\pi}} \sum_{n=1}^{\infty} \frac{e^{-(1/2)c \log n}}{\sqrt{c \log n}}$$

$$= \frac{2}{\sqrt{c\pi}} \sum_{n=1}^{\infty} \frac{1}{n^{c/2} (\log n)^{1/2}}$$

which is finite for $c > 2$**.**

– **By the Borel-Cantelli lemma,**

$$P\{\mid Z_n \mid > \sqrt{c \log n} \text{ infinitely often}\} = 0.$$

– **Taking** $c > 2$**, the fact implies that with probability** 1**,**

$$M_n \leq 2^{-n/2-1} \sqrt{c \log(2^{n+1})}$$

holds for all sufficiently large n.
We have $\sum M_n < \infty$ with probability 1, which completes the proof of the above claim.

Acceptance/Rejection Method

This method assumes that we have a method for simulating from some density function g and our task is utilize samples from g to simulate from a given density function f.
g can be fairly arbitrary except for one condition mentioned below.

- **The basic idea is to simulate from g and accept the samples with probability proportional to the ratio f/g.**
  - **Requirement: Let C be a constant such that**

$$\frac{f(Y)}{g(Y)} \leq C; \quad \text{for all } Y.$$

- **Simulation procedure:**

(1) Simulate Y from the density g and simulate U from uniform $[0, 1]$.

(2) If $U \leq f(Y)/[Cg(Y)]$ then $X = Y$ else go to step 1.

Validity of Acceptance/Rejection Method

- **Let X be the value obtained and n be the number of iterations required to reach this value.**

$$
\begin{aligned}
P(X \le x) &= P(Y_n \le x) = P\left(Y \le x \mid U \le \frac{f(Y_n)}{Cg(Y_n)}\right) \\
&= \frac{P\left(Y \le x, U \le \frac{f(Y_n)}{Cg(Y_n)}\right)}{P\left(Y \le x, U \le \frac{f(Y_n)}{Cg(Y_n)}\right)} \\
&= \frac{\int_{-\infty}^{x} \int_{0}^{f(y)/Cg(y)} 1 du g_Y(y) du dy}{\int_{-\infty}^{\infty} \int_{0}^{f(y)/Cg(y)} 1 du g_Y(y) du dy} \\
&= \frac{\int_{-\infty}^{x} \frac{f(y)}{Cg(y)} g_Y(y) dy}{\int_{-\infty}^{\infty} \frac{f(y)}{Cg(y)} g_Y(y) dy},
\end{aligned}
$$

since Y and U are independent random variables. (Their joint density function is the product of the marginals $g(y) \times 1$)

- As $x \to \infty$, the left side goes to 1 and the integral on the right side also goes to 1.
  Therefore,

$$C \int_{-\infty}^{\infty} \frac{f(y)}{Cg(y)} g_Y(y) dy = 1$$

  and $P(X \leq x) = \int_{-\infty}^{x} f(Y) dY$. We conclude that X is random with probability density f.

Efficiency: For a given value of Y we accept Y by generating a uniform U and comparing U with $f(Y)/Cg(Y)$.

- Accept Y with probability $f(Y)/Cg(Y)$.

- **Each iteration in the loop involves independent realizations, we can compute the probability of accepting Y as X according to**

$$P\left(U \leq \frac{f(Y)}{Cg(Y)}\right) = K = \frac{1}{C}.$$

  If C is large then the process, of generating samples from f using this method, will be slow.

- **What is the distribution of n?**

Illustration: Use acceptance/rejection method to generate sample from standard normal density function.

- **Find g with support on $(-\infty, \infty)$.**
  - **Sampling from the standard exponential density function ($g(x) = \exp(-x)$) can be done quickly.**

- **Note that the support of g is on $[0, \infty)$ and f is symmetric at 0. Convert the problem to the generation of half-normal variate.**

- **Generate $X = \mid Z \mid$ with density function**

$$f(x) = \frac{2}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right), \quad x \geq 0.$$

- **Determine C.**
  - **The bound on the ratio of f to g:**

$$\frac{f(x)}{g(x)} = \sqrt{\frac{2e}{\pi}} \exp\left(-\frac{(x-1)^2}{2}\right) \leq \sqrt{\frac{2e}{\pi}} = C.$$

  - $f(x)/Cg(x) = \exp(-(x-1)^2/2)$**.**

- **Algorithm 1**

1. **Generate Y, an exponential random variable with mean 1, and U, a uniform $[0,1]$ random variable.**
2. **If $U \leq \exp(-(Y-1)^2/2)$ set $X = Y$, otherwise return to (1).**

- **Algorithm 2**
  Observe that $-\log(U) \geq (Y-1)^2/2$ and $\log(U)$ is exponential with rate 1.

  1. **Generate Y_1 and Y_2, two samples from exponential random variable with mean 1.**
  2. **If $Y_2 \geq (Y_1 - 1)^2/2$ set $X = Y_1$, otherwise return to (1).**

  Having generated a random variable which is the absolute value of a standard normal, we can generate sample from standard normal.

1. **Generate U a uniform random variable the algorithm described above.**
2. **If $U \in (0, 1/2]$ set $Z = X$, else set $Z = -X$.**

Example on R-programming:
Generate deviates from a beta distribution with parameters α and β.

$$f(x) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1}.$$

- **It has a finite support $[0, 1]$.**
- **Choose g as a uniform.**
- **Need to find the mode f. Solve**

$$\frac{\alpha - 1}{x} - \frac{\beta - 1}{1 - x} = 0$$

and obtain $xmode = (\alpha - 1)/(\alpha + \beta - 2)$.

- $C = (xmode)^{\alpha-1}(1-xmode)^{\beta-1}\Gamma(\alpha+\beta)/(\Gamma(\alpha)\Gamma(\beta))$

R-program

- `alpha<- 2; beta<- 7; nsimu<- 1000`
- `xmode<- (alpha -1)/(alpha+ beta -2)`
- `dmax<- xmode^(alpha -1)*(1-xmode)^(beta-1)* gamma(alpha+beta)/(gamma(alpha)*gamma(beta))`
- `y<- runif(nsimu)`
- `x<- na.omit(ifelse(runif(nsimu)<=dbeta(y,alpha, beta)/dmax,y,NA))`

Note that dmax≈ 3.18 in this case, we expect to get around $1000/3.18$ deviates.

Remarks

- **No clear rule to find g.**

- $g(y)$ **should be** $similar$ **and** $dominate$ $f(y)$**.**

- **The constant** C **maynot be easy to find.**
  - **As an example, how do we determine** C **for the posterior distribution**

$$p(\theta \mid y) \propto (2+\theta)^{125}(1-\theta)^{38}\theta^{34}$$

If it is hard to apply rejection method, what can be used?

Simulating Multivariate Random Variates

With multivariate distributions, one is often faced with enormous problems for random variate generation.

- **Von Neumann's rejection method [von Neumann 1963] requires a case-by-case study.**
  - **It is difficult to determine a usable majorizing density.**
- **The conditional method (generate one random variate; generate the next one conditional on the first one, and so forth) requires often difficult-to-compute marginal densities.**
  Consider the generation of multivariate normal with mean 0 and variance-covariance matrix $\Sigma = (\sigma_{ij})_{p \times p}$.

- $\mathbf{X}_{p\times 1}$ **has a multivariate normal distribution iff** $\mathbf{X}$ **can be written as**

$$\mathbf{X} = \mu + \mathbf{AZ}$$

where $\mu_{p\times p}$**,** $\mathbf{A}$ **are constant and** $\mathbf{X} = (X_1, \ldots, X_p)^T$ **where the** Z_j **are independent standard normal variables.**

- $\Sigma = \mathbf{A}\mathbf{A}^T$

  $\mathbf{A}$ **is nonsingular iff** Σ **is positive definite.**

- **By the spectral decomposition theorem, there exists** P **orthogonal such that** $\Sigma = \mathrm{P}^T\mathrm{DP}$**.**

  Here D **is the diagonal matrix whose diagonal entries are nonnegative eigenvalues of** Σ**.**

- **If** $rank(\Sigma) = p$**,** $\Sigma^{1/2} = \mathbf{P}\mathbf{D}^{1/2}\mathbf{P}^T$**.**

  Useful R-command:

  - **solve: Solve a system of equations.**
  - **eigen: Computes eigenvalues and eigenvectors.**
  - **backsolve: Solve an upper or lower Triangular System.**
  - **chol: Compute the Cholesky factorization of a real symmetric positive-definite square matrix.**
  - **qr: The QR decomposition of a matrix**
- **Write** $\mathbf{X} = ((\mathbf{X}^{(1)})^T, (\mathbf{X}^{(2)})^T)^T$**, the conditional distribution of** $\mathbf{X}^{(2)}$ **given** $\mathbf{X}^{(1)} = \mathbf{x}^{(1)}$ **is normal with mean** $\mu^{(2)} + \Sigma_{21}\Sigma_{11}^{-1}(\mathbf{x}^{(1)} - \mu^{(1)})$ **and variance** $\Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{21}$**.**
- X_1 **is generated as** $N(\mu_1, \sigma_{11})$**,**
- X_2 **is generated as** $N(\mu_2 + \sigma_{12}X_1/\sigma_{11}, \sigma_{22} - \sigma_{12}^2/\sigma_{11})$**,**

and so on,

- **Generate multivariate random variates by use of either iid univariates followed by a transformation.**