Random Variate Generation

Non-uniform RV

Radu Trîmbițaș

Faculty of Math and CS-UBB

1st Semester 2011-2012

Topics I

- General principles
  - Inverse Transform Method
  - Acceptance-Rejection Method
  - Composition Method
  - Convolution Method
  - Translation and Other Simple Transforms (Characterization)
- Continuous Distributions
  - Inverse Transform by Numerical Solution
  - Specific Continuous Distribution
- Discrete Distribution
  - Look-up Tables
  - Alias Method
  - Empirical Distribution
  - Specific Discrete Distributions

Topics II

- Multivariate Distribution
    - General Methods
    - Special Distributions
- Stochastic Processes
    - Point Processes
    - Time-Series Models and Gaussian Processes

Introduction

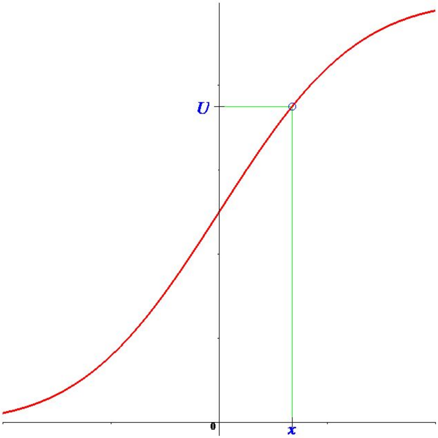
- The basic problem is to generate a random variable X, whose distribution is completely known and nonuniform
- RV generators use as starting point random numbers distributed $U[0,1]$ - so we need a good RN generator
- Assume RN generates a sequence $\{U_1, U_2, \dots\}$ IID
- For a given distribution there exists more than one method
- **Assumption**: a uniform RNG is available, and a call $RN(0,1)$ produce a uniform r.n., independent of all variates generated by previous calls

Choice Criteria

1. *Exactness* – a generator is exact if the distribution of variates has the exact form desired; the opposite approximative generator
2. *Mathematical validity* – does it give what it is supposed to?
3. *Speed* – initial setup time + variable generation time the relative contribution of these factors depends on application
4. *Space* – computer memory requirements of the generator; short algorithms, but some of them make use of extensive tables, important when if different tables need to be held simultaneously in memory
5. *Simplicity*, both algorithmic and implementational
6. *Parametric stability* – is it uniformly fast for all input parameters (e.g. will it take longer to generate PP as rate increases?)

Inverse Transform Method (Continuous Case)

X, F cdf of X, f pdf of X
Let $U := RN(0,1)$
return $X := F^{-1}(U)$



Example - Exponential distribution

$X \sim Exp(a)$

$$F(x) = \begin{cases} 1 - \exp\left(\frac{x}{a}\right), & x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Solving $u = F(x)$ for x yields

$$x = F^{-1}(u) = -a \ln(1-u) \tag{2}$$

Generate u rv $U[0,1]$, then apply (2) to obtain X having cdf (1).

Example

Consider the case $a = 1$ (see Figure 2). The cdf for $x > 0$ is $F(x) = 1 - \exp(-x)$. Two random variates has been generated using (2). The first r.n. generated is $u_1 = 0.7505$ and the corresponding x is $x_1 = -\ln(1 - 0.7505) = 1.388\,3$. Similarly, the random number $u_2 = 0.1449$ generates the exponential variate $x_2 = -\ln(1 - 0.1449) = 0.156\,54$.

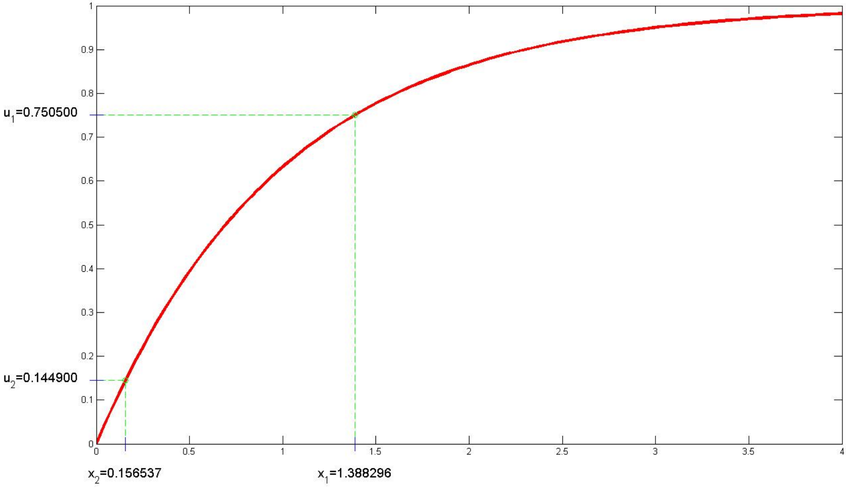


Figure: Inverse transform for exponential distribution

Inverse Transform Method (Discrete Case) I

- Suppose X has the distribution $\begin{pmatrix} x_i \\ p_i \end{pmatrix}$. The cdf is

$$F(x) = P(X \le x) = \sum_{i: x_i \le x} p_i.$$

- We "define" the inverse by

$$F^{-1}(u) = \min\{x : u \le F(x)\}$$

- The method still works despite the discontinuities of F (see Figure 3)

$U := RN(0,1);\ i := 1;$
`while` $(F(x_i) < U)\{i := i + 1\}$
`return` $X = x_i$

- Because the method uses a linear search, it can be ineficient if n is large. More efficient methods are required.

Inverse Transform Method (Discrete Case) II

- If a table of x_i values with the corresponding $F(x_i)$ values are stored, the method is called *table look-up method*. The method compares U with each $F(x_i)$, returning, as X, the first x_i encountered for which $F(x_i) \geq U$.

Inverse Transform Method (Discrete Case)

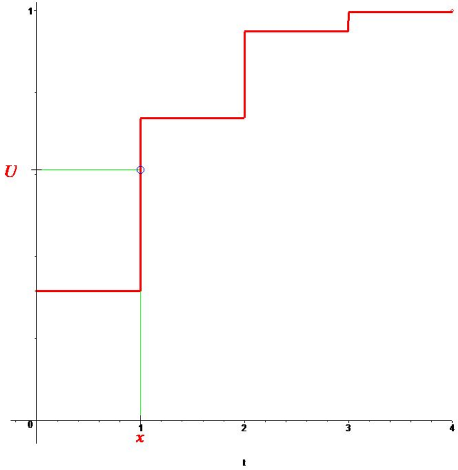


Figure: Inverse transform method - Bin(4,0.25)

Example - Binomial Distribution

Example

$X \sim Bin(4, 0.25)$. The possible values of X are $x_i = i$, $i = 0, \ldots, 4$, and the values of F are given in Table 1. Suppose $U = 0.6122$ is a given random number. Looking along the rows of $F(x_i)$ values, we see that $F(x_0) = 0.3164 < U = 0.6122 < F(x_1) = 0.7383$. Thus x_1 is the first x_i such that $U \leq F(x_i)$; therefore $X = 1$. (see Figure 3).

i	0	1	2	3	4
p_i	0.3164	0.4219	0.2109	0.0469	0.0039
$F(x_i)$	0.3164	0.7383	0.9492	0.9961	1.0000

Table: Distribution of $Bin(4, 0.25)$

Inverse Transform Method - Correctness

Constructive proof:

Theorem

If $U \sim U[0,1]$, then the random variable $X = F^{-}(U)$ has the distribution function F, where F^{-} is the inverse function of F defined as

$$F^{-}(p) = \inf\{x : F(x) \geq p\}, \qquad 0 < p < 1.$$

Proof.

First, we have $F^{-}(F(x)) \leq x$ for $x \in \mathbb{R}$ and $F(F^{-}(u)) \geq u$ for $0 < u < 1$. Thus

$$P(X \leq x) = P(F^{-}(U) \leq x) = P(U \leq F(x)) = F(x).$$

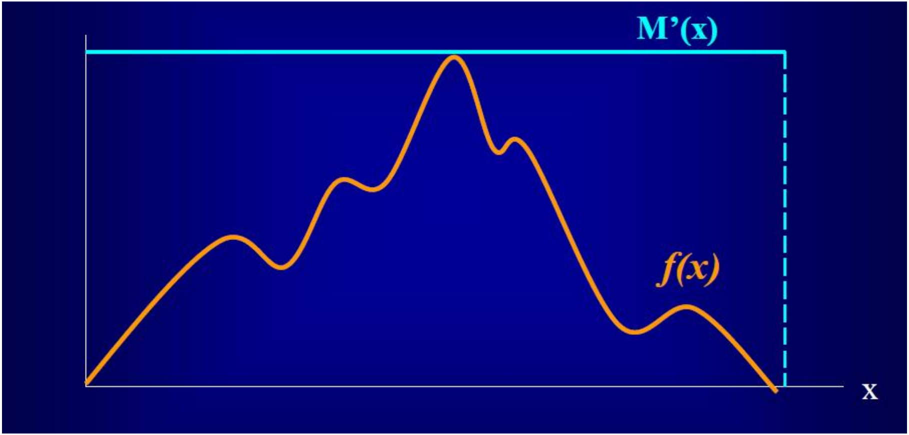
$\square$

Acceptance-Rejection Method

- X has density $f(x)$ with bounded support
- If F is hard (or impossible) to invert, too messy ... what to do?
- Generate Y from a more manageable distribution and accept as coming from f with a certain probability

Acceptance-Rejection Intuition

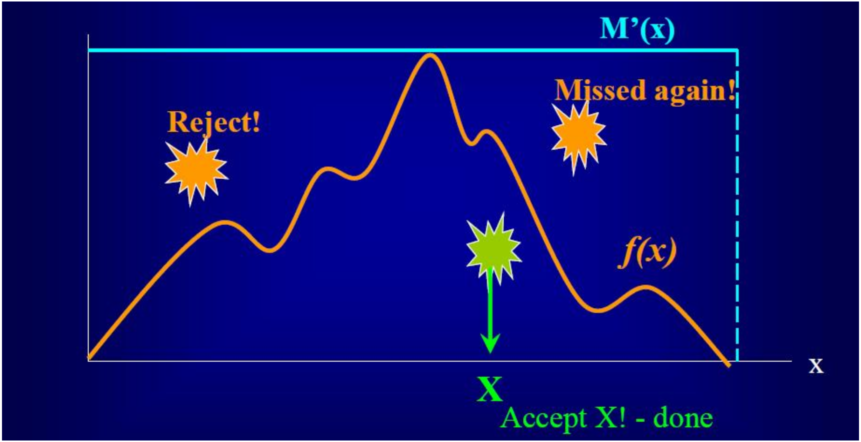
Density $f(x)$ is really ugly ... Say, Orange!



M' is a "Nice" Majorizing function..., Say Uniform

Acceptance-Rejection Intuition

Throw darts at rectangle under M' until hit f



Prob$\{$Accept $X\}$ is proportional to height of $f(X)$ - called *trial ratio*

Acceptance-Rejection Correctness

The basic idea comes from the observation that if f is the target density, we have

$$f(x) = \int_0^{f(x)} 1 du.$$

Thus, f can be thought as the marginal density of the joint distribution

$$(X, U) \sim Unif\{(x, u) : 0 < u < f(x)\},$$

where U is called an auxiliary variable.

Theorem

Let $X \sim f(x)$ and let $g(y)$ be a density function that satisfies $f(x) \leq Mg(x)$ for some constant $M \geq 1$. To generate a random variable $X \sim f(x)$: (1) Generate $Y \sim g(y)$ and $U \sim Unif[0, 1]$ independently; (2) If $U \leq f(Y)/Mg(Y)$ set $X = Y$; otherwise return to step (1).

Acceptance-Rejection Proof

Proof.

The generated random variable X has distribution

$$\begin{aligned}
P(X \leq x) &= P(Y \leq x | U \leq f(Y)/Mg(Y)) \\
&= \frac{P(Y \leq x, U \leq f(Y)/Mg(Y))}{P(U \leq f(Y)/Mg(Y))} \\
&= \frac{\int_{-\infty}^{x} \int_{0}^{f(y)/Mg(y)} 1 \cdot du \cdot g(y) dy}{\int_{-\infty}^{\infty} \int_{0}^{f(y)/Mg(y)} 1 \cdot du \cdot g(y) dy} \\
&= \frac{\int_{-\infty}^{x} f(y)/\left(Mg(y)\right) \cdot g(y) dy}{\int_{-\infty}^{\infty} f(y)/\left(Mg(y)\right) \cdot g(y) dy} \\
&= \int_{-\infty}^{x} f(y) dy,
\end{aligned}$$

which is the desired distribution. $\square$

Example - Gamma distribution

Example

We want to generate $\gamma(b, 1)$, for $b > 1$ (see [Fishman, 1996]). The pdf is

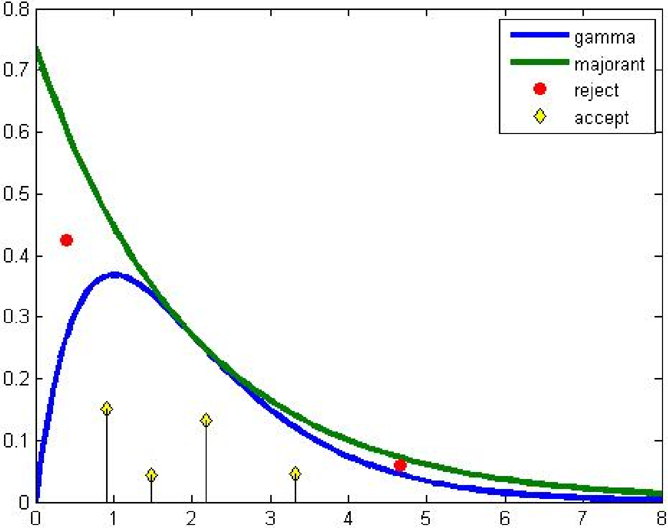
$$f(x) = x^{b-1} \exp(x)/\Gamma(b), \; x > 0.$$

The majorizing function is $e(x) = K \exp(-x/b)/b$. If

$$K = \frac{b^b \exp(1-b)}{\Gamma(b)}$$

then $e(x) \geq f(x)$ for $x \geq 0$. The method is convenient for b not too large. Figure 4 illustrates the generation.

Example - Gamma distribution



Composition Method I

- Can be used when F can be expressed as a convex combination of other distributions F_i, where we hope to be able to sample from F_i more easily than from F directly.

$$F(x) = \sum_{i=1}^{\infty} p_i F_i(x) \text{ and } f(x) = \sum_{i=1}^{\infty} p_i f_i(x)$$

- p_i is the probability of generating from F_i
- Algorithm
  1. Generate positive random integer J such that

$$P\{J = j\} = p_j, \text{ for } j = 1, 2, \dots$$

  2. Return X with distribution function F_j

Composition Method II

- Think of Step 1 as generating J with mass function p_J

$$P(X \le x) = \sum_{j=1}^{\infty} P(X \le x | J = j) P(J = j) = \sum_{j=1}^{\infty} F_j(x) p_j = F(x).$$

Example

The double exponential (or Laplace) distribution has density $f(x) = \frac{1}{2} e^{-|x|}$, $x \in \mathbb{R}$ (Figure 5), We can express the density as

$$f(x) = 0.5 e^x I_{(-\infty,0)} + 0.5 e^{-x} I_{(0,\infty)},$$

I_A indicator of A. f convex combination of $f_1(x) = e^x I_{(-\infty,0)}$ and $f_2(x) = e^{-x} I_{(0,\infty)}$. We can generate X with density f by composition. First generate $U_1, U_2 \sim U[0,1]$. If $U_1 \le 0.5$, return $X = \ln U_2$, else return $X = -\ln U_2$.

Composition Method III

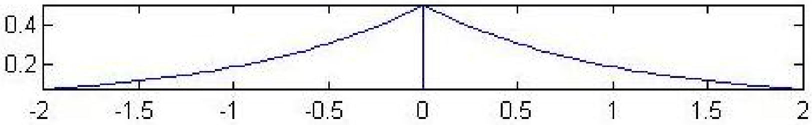


Figure: Double-exponential density

Convolution

- Suppose Y_i, $i = 1, \ldots, n$ IID rv and $X = Y_1 + Y_2 + \cdots + Y_n$
- Algorithm Y_i, $i = 1, \ldots, n$ IID rv with cdf G
  1. Generate Y_i, $i = 1, \ldots, n$
  2. Return $X = Y_1 + Y_2 + \cdots + Y_n$
- The distribution of X is the m-fold convolution of G
- In probability theory, the probability distribution of the sum of two or more independent random variables is the convolution of their individual distributions

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,\mathrm{d}\tau$$

Convolution- Examples

Examples

1. Y_i, $i = 1, \ldots, n$ IID $\chi^2(1,1)$; $X = Y_1 + Y_2 + \cdots + Y_n$ is distributed $\chi^2(n,1)$
2. The m-Erlang rv with mean β is the sum of m IID exponential rvs with common mean β/m. Thus we generate first $Y_1, \ldots, Y_m$ IID $Exp(\beta/m)$, then return $X = Y_1 + Y_2 + \cdots + Y_n$
3. If X_i has a $\Gamma(a_i, \lambda)$ distribution for $i = 1, 2, ..., n$, i.r.v., then

$$\sum_{i=1}^{n} X_i \sim \Gamma\left(\sum_{i=1}^{n} a_i, \lambda\right)$$

Translation and Other Simple Transforms

- Often a random variable can be obtained by some elementary transformation of another
- lognormal variable is an exponential of a normal variable
- $\chi^2(1)$ is a standard normal variable squared
- More elementary, location-scale models – if X is a crv with pdf f then $Y = aX + b$, $a > 0, b \in \mathbb{R}$, then Y has the density

$$g(y) = a^{-1} f\left(\frac{y-b}{a}\right)$$

Summary

if CDF invertible *then* **inversion**
else if CDF or PDF is a sum of other CDFs or PDFs *then* **composition**
else if rv is a sum of iid rv's *then* **convolution**
else if rv is related to other rv then **characterization**
else if a majorising function exists then **acceptance-rejection**
else use empirical inversion

Ratio of Uniforms I

- *Ratio of uniforms* method is based on a relationship among the r.v. U, V, and V/U.
- If $p > 0$, real, and if (U, V) is uniformly distributed over the set

$$S = \left\{ (u, v) : 0 \leq u \leq \sqrt{pf\left(\frac{v}{u}\right)} \right\},$$

  then V/U has the pdf f.
- **Proof:** the joint density function of (U, V) is

$$f_{U,V}(u, v) = \frac{1}{s}, \qquad (u, v) \in S$$

  where s is the measure of S.

Ratio of Uniforms II

- Let $Y = U$, $Z = V/U$: the jacobian J of this transform is

$$J = \begin{vmatrix} \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \end{vmatrix} = \begin{vmatrix} 1 & 0 \\ z & y \end{vmatrix} = y$$

- The joint distribution of (Y, Z) is

$$f_{Y,Z}(y,z) = |J|\, f_{U,V}(u,v) = \frac{y}{s}, \qquad 0 \le y \le \sqrt{pf(z)}, 0 < z < \infty$$

- The density of Z is

$$f_Z(z) = \int_0^{\sqrt{pf(z)}} f_{Y,Z}(y,z)dy = \int_0^{\sqrt{pf(z)}} \frac{y}{s}dy = \frac{p}{2s}f(z) = f(z),$$

since $\int f_Z$ and $\int f$ equate 1 $\Longrightarrow s = p/2$.

Ratio of Uniforms III

- **Idea**: to generate (u, v) uniformly in S, choose a majorizing region T that contains S, generate a point (u, v) uniformly in T, and accept it if
$$u^2 \le pf\left(\frac{v}{u}\right);$$
otherwise generate a new point and try again, etc.
- S is defined parametrically by
$$u(z) = \sqrt{pf(z)}, \qquad v(z) = z\sqrt{pf(z)}$$

Ratio of Uniforms IV

- If $f(x)$ and $x^2 f(x)$ are bounded, then a good choice for T is the rectangle
$$T = \{(u, v) : 0 \leq u \leq u^*, v_* \leq v \leq v^*\}$$
where
$$u^* = \sup_z \sqrt{pf(z)}$$
$$v_* = \inf_z z\sqrt{pf(z)}$$
$$v^* = \sup_z z\sqrt{pf(z)}$$

Algorithm

1. Generate $U \sim U(0, u^*)$ and $V \sim U(v_*, v^*)$ independently
2. Set $Z = V/U$
3. If $U^2 \leq pf(z)$, the return Z. Otherwise, go to step 1.

Ratio of Uniforms V

Probability of acceptance

$$\frac{s}{t} = \frac{p/2}{u^*(v^* - v_*)}$$

$t-$area of T

Continuous Distributions

Inverse Transform by Numerical Solution

- Solve the equation $F(X) = U$, or equivalently $\varphi(X) := F(X) - U = 0$, numerically for X
- Methods: bisection, false position, secant, Newton
- Problem: find starting values

Bisection

$a := -1;$
while $F(a) > U$ **do**
  $a := 2 * a;$
$b := 1;$
while $F(b) < U$ **do**
  $b := 2 * b;$
while $b - a > \delta$ **do**
  $X := (a + b)/2;$
  if $F(x) \leq U$ **then**
    $a := X;$
  else
    $b := X;$

Newton

For unimodal densities with known mode, X_m the following alternative is quicker

$Y_m := F(X_m);\ U := RN(0,1);$
$X := X_m;\ Y := Y_m;\ h := Y - U;$
while $|h| > \delta$ **do**
  $X := X - h/f(X);$
  $h := F(X) - U;$
return X

- Convergence is guaranted for unimodal densities because $F(x)$ is convex for $x \in (-\infty, X_m)$ and concave for $x \in (X_m, \infty)$
- The tolerance criterion guaranteed that guarantees that $F(X)$ is close to U (within δ), but it does not guarantee that X is close to the exact solution of $F(X) = U$

Uniform U(a,b), a<b

- pdf

$$f(x) = \begin{cases} \frac{1}{b-a}, & x \in (a, b) \\ 0, & \text{otherwise} \end{cases}$$

- cdf

$$F(x) = \begin{cases} 0, & x \leq a \\ \dfrac{x-a}{b-a}, & a < x < b \\ 1, & x \geq b \end{cases}$$

- generator: inverse transform method

$U := RN(0, 1)$;
return $X := a + (b - a)U$

Exponential Exp(a), a>0

- pdf

$$f(x) = \begin{cases} \frac{1}{a}\exp\left(-\frac{x}{a}\right), & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

- cdf

$$F(x) = \begin{cases} 1 - \exp\left(-\frac{x}{a}\right), & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

- generator: inverse transform method

$U := RN(0,1)$;
return $X := -a\ln(1-U)$;

Weibull Weib(a,b), a,b>0

- pdf

$$f(x) = \begin{cases} ba^{-b}x^{b-1}\exp\left[-\left(\frac{x}{a}\right)^b\right], & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

- cdf

$$f(x) = \begin{cases} 1-\exp\left[-\left(\frac{x}{a}\right)^b\right], & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

- generator: inverse transform method

$U := RN(0,1);$

return $X := a\left[-\ln(1-U)\right]^{1/b}$

- Notes:
  1. Some references replace $1-U$ by U since $1-U \sim U(0,1)$; this is not recommended
  2. for $b = 1$ exponential distribution
  3. $X \sim Weib(a,b) \Longrightarrow X^b \sim Exp(a^b)$;
     $E \sim Exp(a^b) \Longrightarrow E^{1/b} \sim Weib(a,b)$

Extreme Value EXTREME(mu,sigma)

- pdf

$$f(x) = \sigma^{-1} \exp\left(-\frac{x-\mu}{\sigma}\right) \exp\left[-\exp\left(-\frac{x-\mu}{\sigma}\right)\right], \quad x \in \mathbb{R}$$

- cdf

$$F(x) = \exp\left[-\exp\left(-\frac{x-\mu}{\sigma}\right)\right], \quad x \in \mathbb{R}$$

- generator: inverse transform method

$U := RN(0,1);$
return $X := -\sigma \ln\left[-\ln(U)\right] + \mu$

Gamma Gam(a,b), a,b>0 I

- pdf

$$f(x) = \begin{cases} \dfrac{(x/b)^{a-1}}{b\Gamma(a)} \exp\left(-\dfrac{x}{b}\right), & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

- cdf - improper integral
- generator: no single method satisfactory for all values of a. Generators cover different ranges of a
- Generator 1 (Ahrens&Dieter) acceptance-rejection, requires $a \in (0, 1)$

Gamma Gam(a,b), a,b>0 II

Setup : $\beta := (e + a)/e;$

$$
\begin{array}{l}
\text{while(true)}\{ \\
\quad U := RN(0,1);\ W := \beta U; \\
\quad \text{if } (W < 1)\{ \\
\qquad Y := W^{1/a};\ V = RN(0,1); \\
\qquad \text{if } (V \leq e^{-Y}) \text{ return } X = bY;\} \\
\quad \text{else } \{ \\
\qquad Y := -\ln\left(\left(\beta - W\right)/a\right);\ V := RN(0,1); \\
\qquad \text{if } (V \leq Y^{b-1}) \text{ return } X := bY\} \\
\}
\end{array}
$$

- Generator 2: (Cheng) acceptance-rejection; requires $a > 1$

Gamma Gam(a,b), a,b>0 III

$$\text{Setup}: \ \alpha := (2a-1)^{-1/2}; \ \beta := a - \ln 4; \ \gamma := a + \alpha^{-1}; \delta := 1 + \ln 4.5;$$

```
while (true){
    U_1 := RN(0,1);  U_2 := RN(0,1);

    V := α ln (U_1/(1 − U_1)) ;  Y := a e^V;
    Z := U_1^2 U_2;  W := β + γV − Y;

    if (W + δ − 4.5Z ≥ 0)
        return X := bY;
    else{
        if (W ≥ ln Z) return X := bY; }

}
```

- Generator 3 (Fishman) acceptance-rejection; requires $a > 1$ and it is simple and efficient for values of $a < 5$.

$$
\begin{array}{l}
\text{while (true) } \{ \\
\quad U_1 := RN(0,1);\ \ U_2 := RN(0,1);\ \ V_1 := -\ln U_1;\ \ V_2 := -\ln U_2; \\
\quad \text{if } \left(V_2 > (a-1)\left(V_1 - \ln V_1 - 1\right)\right) \text{ return } X := bV_1; \\
\}
\end{array}
$$

Erlang ERL(m,k), m>0, k natural

- pdf – same as $GAM(k, m/k)$
- cdf – improper integral
- generator 1. If $X \sim ERL(m, k)$, it is the sum of k i.r.v. $Exp(m/k)$

$$U_1 := RN(0,1);\ U_2 := RN(0,1);\ \ldots;\ U_k := RN(0,1);$$
$$\text{return } X := -(m/k)\ln\left((1-U_1)(1-U_2)\ldots(1-U_k)\right)$$

- generator 2. generates $GAM(k, m/k)$
- generator 1 efficient for $k < 10$. For larger values of k, generator 2 is faster and not affected by error caused by multiplication of quantities < 1.

Normal I

- pdf

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \quad x \in \mathbb{R}$$

- cdf improper integral
- generator 1. Box-Muller

$U_1 := RN(0,1);\ \ U_2 := RN(0,1);$
return $X_1 := \sqrt{-2\ln U_1}\cos U_2$ and $X_2 := \sqrt{-2\ln U_1}\sin U_2$

- If $X_1, X_2 \sim N(0,1)$, then
  $D^2 = X_1^2 + X_2^2 \sim \chi^2(2) \equiv Exp(2) \Longrightarrow D = \sqrt{-2\ln U}$
- $X_1 = D\cos\omega,\ X_2 = D\sin\omega,\ \ \omega = 2\pi U_2;$
- generator 2. Polar Method

Normal II

while(true){
$U_1 := RN(0,1);\ U_2 := RN(0,1);$
$V_1 := 2U_1 - 1;\ V_2 := 2U_2 - 1;\ W := V_1^2 + V_2^2;$
if $(W < 1)${
$Y := \left[(-2 \ln W) / W\right]^{1/2};$
return $X_1 := \mu + \sigma V_1 Y$ and $X_2 := \mu + \sigma V_2 Y$
}
}

Beta BETA(p,q) I

- pdf

$$f(x) = \begin{cases} \dfrac{x^{p-1}(1-x)^{q-1}}{B(p,q)}, & x \in (0,1) \\ 0, & \text{otherwise} \end{cases}$$

- cdf – improper integral
- generator 1. If $G_1 \sim GAM(p,a)$, $G_2 \sim GAM(q,a)$, independent, then $X = G_1/(G_1+G_2) \sim BETA(p,q)$
- generator 2. (Cheng) acceptance-rejection, for $p,q>1$

$$
\begin{aligned}
&\text{setup}: \ \alpha := p+q;\ \beta := \sqrt{(\alpha-2)\,/\,(2pq-\alpha)};\ \gamma := p+\beta^{-1};\\
&\text{do}\{\\
&\qquad U_1 := RN(0,1);\ U_2 := RN(0,1);\\
&\qquad V := \beta \ln\left(U_1\,/\,(1-U_1)\right);\ W := pe^{V};\\
&\}\text{while}\ \left(\alpha \ln\left[\alpha\,/\,(q+W)\right] + \gamma V - \ln 4 < \ln\left(U_1^2 U_2\right)\right)\\
&\text{return}\ X := W\,/\,(q+W);
\end{aligned}
$$

Beta BETA(p,q) II

- generator 3. (Jöhnk) acceptance-rejection for $p, q < 1$

do {
  $U := RN(0,1);\ V := RN(0,1);$

  $Y := U^{1/p};\ Z := V^{1/q};$

} while$(Y + Z > 1)$

return $X := Y/(Y + Z)$

Discrete Distributions

Look-up Tables I

- General methods work for discrete distributions, but with modifications
- *look-up table method* and *alias method*
- Suppose distribution has the form

$$p_i = P(X = X_i), \qquad P_i = \sum_{j=1}^{i} p_j = P(X \leq x_i), \ i = 1, \dots, n$$

- If the table is large, look-up procedure is slow, to find i we need i steps
- Acceleration: binary search, hashing, etc.
- When the number of points is infinite we need an appropriate cutoff, for example

$$P_n > 1 - \delta = 1 - 10^c$$

c must be selected carefully

Look-up Tables II

- Look-up by binary search

$U := RN(0,1);\ A := 0; B := n;$

while $(A < B - 1)\{$

$\quad i := \text{trunc}((A + B)/2);$

$\quad$ if $(U > P_i)\ A := i$

$\quad$ else $B = i;$

$\}$

return $X := X_i$

- An alternative is to make a table of starting points aproximately every (n/m)th entry, in the same way that the letters of the alphabet form convenient starting points for search in a dictionary

Look-up Tables III

- Look-up by indexed search -setup

$i := 0;$
$\text{for } (j := 0 \ to \ m-1)\{$
$\quad \text{while } (P_i < j/m)\{i := i+1\}$
$Q_j := i;$
$\}$

- search

$U := RN(0,1); \ j := \text{trunc}(mU); \ i := Q_j;$
$\text{while } (U \geq P_i) \ i := i+1;$
$\text{return } X := X_i$

Alias Method I

- X has the range $S_n = \{0, 1, \ldots, n\}$
- From the given $p(i)$'s we compute two arrays of length $n+1$
  1. cutoff values $F_i \in [0, 1]$, $i = 0, 1, \ldots, n$
  2. aliases $L_i \in S_n$ for $i = 0, 1, \ldots, n$
- Setup for the alias method Walker (1977)
  1. Set $L_i = i$, $F_i = 0$, $b_i = p_i - 1/(n+1)$, for $i = 0, 1, \ldots, n$
  2. For $i = 0, 1, \ldots, n$ do the following steps
     1. Let $c = \min\{b_0, b_1, \ldots, b_n\}$ and k be the index of this minimal b_j. (Ties can be broken arbitrarily)
     2. Let $d = \max\{b_0, b_1, \ldots, b_n\}$ and m be the index of this maximal b_j. (Ties can be broken arbitrarily)
     3. If $\sum_{j=0}^{n} |b_j| < \varepsilon$, stop the algorithm.
     4. Let $L_k = m$, $F_k = 1 + c(n+1)$, $b_k = 0$, and $b_m = c + d$.
- Setup for the alias method Kronmal and Peterson (1979)
  1. Set $F_i = (n+1)p(i)$ for $i = 0, 1, \ldots, n$

Alias Method II

2. Define the sets $G = \{i : F_i \geq 1\}$ and $S = \{i : F_i < 1\}$
3. Do the following steps until S becomes empty:
    1. Remove an element k from G and remove an element m from S
    2. Set $L_m = k$ and replace F_k by $F_k - 1 + F_m$
    3. If $F_k < 1$, put k into S; otherwise, put k back into G

- The cuttof and the aliases are not unique
- The alias method
    1. Generate $I \sim DU(0, n)$ and $U \sim (0, 1)$ independent of I
    2. If $U \leq F_I$ return $X = I$, else return $X = L_I$.

Alias Method - Example I

- Consider the RV; the range is S_3

$$X : \begin{pmatrix} 0 & 1 & 2 & 3 \\ 0.1 & 0.4 & 0.2 & 0.3 \end{pmatrix}$$

- The first setup algorithm leads to

i	0	1	2	3
$p(i)$	0.1	0.4	0.2	0.3
F_i	0.4	0.0	0.8	0.0
L_i	1	1	3	3

- If step 1 of the algorithm produces $I = 2$, the probability is $F_2 = 0.8$; we would keep $X = I = 2$, and with probability $1 - F_2 = 0.2$ we would return $X = L_2 = 3$ instead.

Alias Method - Example II

- Since 2 is not the alias of anything, the algorithm returns $X = 2$ if and only if $I = 2$ in step 1 and $U \leq 0.8$ in step 2

$$
\begin{aligned}
P(X = 2) = P(I = 2 \wedge U \leq 0.8) = \\
= P(I = 2)P(U \leq 0.8) = 0.25 \cdot 0.8 = 0.2
\end{aligned}
$$

- $X = 3$ is returned when
    - if $I = 2$, since $F_3 = 0$, we return $X = L_3 = 3$
    - if $I = 2$, we return $X = L_2 = 3$ with probability $1 - F_2 = 0.2$.

$$
\begin{aligned}
P(X = 3) = P(I = 3) + P(I = 2 \wedge U > F_2) \\
= 0.25 +).25 \cdot 0.2 = 0.3
\end{aligned}
$$

Alias Method - Infinite Case

- In this case can be combine with composition method
- If $X \in \mathbb{N}$, we find an n such that $q = \sum_{i=0}^{n} p(i)$ is close to 1, and $P(X \in S_n)$ is hight.
- Since

$$p(i) = q\left[\frac{p(i)}{q} I_{S_n}(i)\right] + (1-q)\left[\frac{p(i)}{1-q}\left(1 - I_{S_n}(i)\right)\right]$$

we obtain the following algorithm

1. Generate $U \sim U(0,1)$. If $U \leq q$ go to step2, otherwise go to step 3;
2. Use the alias method to return X on S_n with probability mass function $p(i)/q$ for $i = 0, 1, \ldots, n$.
3. Use any other method to return X on $\{n+1, n+2, \ldots\}$ with probability mass function $p(i)/(1-q)$ for $i = n+1, n+2, \ldots$

Empirical Distribution

- $x_1, x_2, \ldots, x_n$ is a sample of size n. Assume that each value has the same probability of occuring

$$P(X = x_i) = \frac{1}{n} \qquad i = 1, 2, \ldots, n$$

- We generate variates from this distribution using

$U := RN(0,1);\ i := \mathrm{trunc}(nU) + 1;$
return $X = x_i$

Sampling without replacement; Permutations

- The next algorithm samples $m \le n$ items from the random sample $x_1, x_2, \ldots, x_n$ of size n, without replacement

for $(j = 1 \text{ to } m)\{$
$U := RN(0,1);\ i := trunc\left[(n - j + 1)\, U\right] + j$
$a := a_j;\ a_j := a_i;\ a_i := a; \}$
return $a_1, a_2, \ldots, a_m$

- The routine swaps each entry with one drawn from the remaining list; at the end of the call the entries of the first m positions (i.e. $a_1, a_2, \ldots, a_m$) contains the elements sampled without replacement
- For $m = n$ we generate random permutations of the initial sample

Bernoulli BER(p)

- pmf $\begin{pmatrix} 0 & 1 \\ q = 1 - p & p \end{pmatrix}$
- generator elementary look-up table

$U := RN(0, 1);$
if $(U \leq p)$ then return $X = 1$ else return $X = 0$

Discrete uniform DU(i,j)

- $i, j \in \mathbb{N}$; pmf

$$p(x) = \begin{cases} \frac{1}{j-i+1}, & x \in \{i, i+1, \dots, j\} \\ 0, & \text{otherwise} \end{cases}$$

- generator: inverse transform method

$U := RN(0,1);$

return $X = i + \lfloor (j - i + 1)\, U \rfloor$

Binomial BIN(n,p)

- $n \in \mathbb{N}$, $p \in (0,1)$
- pmf

$$p(x) = \begin{cases} \binom{n}{x} p^x q^{n-x} & x = 0, 1, \ldots, n \\ 0, & \text{otherwise} \end{cases}$$

- Generator: special property $X \sim BIN(n,p)$ is the sum of n independent $BER(p)$. The generation time increases linearly with n. For large n (>20) use general methods.

$X := 0;$
for $(i = 1$ to $n)\{$
  $B := BER(p);\ X := X + B;\}$
return X

Geometric GEOM(p)

- $p \in (0, 1)$
- pmf

$$p(x) = \begin{cases} p(1-p)^x, & x \in \mathbb{N} \\ 0, & \text{otherwise} \end{cases}$$

- generator: the cdf is invertible

$$\begin{aligned}
&\text{Setup}: \; a := 1/\ln(1-p);\\
&U := RN(0,1);\\
&\text{return } X = \text{trunc}(a \ln U);
\end{aligned}$$

- $n \in \mathbb{N}$, $p \in (0,1)$
- pmf
$$p(x) = \begin{cases} \binom{n+x-1}{x} p^n (1-p)^x, & x \in \mathbb{N} \\ 0, & \text{otherwise} \end{cases}$$
- generator: X is the sum of n independent $GEOM(p)$ variables

$X := 0;$
for $(i = 1 \text{ to } n)\{$
  $Y := GEOM(p);\ X := X + Y; \}$
return X

- Time increase linearly with n. One of the general method will be preferable for large n (>10 say).

Hypergeometric HYP(a,b)

- $a, b \in \mathbb{N}^*$
- pmf

$$p(x) = \begin{cases} \dfrac{\binom{a}{x}\binom{b}{n-x}}{\binom{a+b}{n}}, & x = 0, 1, \ldots, n \\ 0, & \text{otherwise} \end{cases}$$

- generator: inverse transform method [Fishman, 1996]

Setup : $\alpha := p_0 = [b!(a+b-n)!] \,/\, [(b-n)!\,(a+b)!]$;
$A := \alpha;\ \ B := \alpha;\ \ X := 0;$
$U := RN(0,1);$
while$(U > A)\{$
$\quad X := X + 1;\ \ B :=$
$B(a-X)(n-X)/\,[(X+1)\,(b-n+X+1)]\,;\ \ A := A + B;$
$\}$
return X

Poisson I

- $POIS(\lambda)$, $\lambda > 0$
- pmf

$$p(x) = \begin{cases} \dfrac{\lambda^x e^{-\lambda}}{x!}, & x \in \mathbb{N} \\ 0, & \text{otherwise} \end{cases}$$

- Generator 1: The direct method is to count the number of events in an appropriate time period as indicated above:

$$\begin{aligned}
&\text{Setup} : a := e^{-\lambda};\\
&p := 1;\ X = -1;\\
&\text{while}(p > a)\{\\
&\qquad U := RN(0,1);\ p := pU;\ X := X + 1;\}\\
&\text{return } X
\end{aligned}$$

- Generator 2:

Poisson II

Setup : $a := \pi\sqrt{\lambda/3}$; $b := a/\lambda$; $c := 0.767 - 3.36/\lambda$; $d :=$
$\ln c - \ln b - \lambda$;
do{
    do{
        $U := RN(0,1)$; $Y := \left[a - \ln\left((1-U)/U\right)\right]/b$;
    }while($Y \le 1/2$)
    $X :=$ trunc$(Y + 1/2)$; $V := RN(0,1)$;
}while$\left(a + bY + \ln\left[V/\left(1+e^{a-bY}\right)^2\right] > d + X\ln\lambda - \ln X!\right)$
return X

- Generator 3: For large λ, $\lambda^{-1/2}(X-\lambda)$ tends to the standard normal. For large $\lambda(>20)$ we have the following:

Setup : $a := \lambda^{1/2}$;
$Z := N(0,1)$;
$X := \max\left[0, trunc\left(0.5 + \lambda + aZ\right)\right]$
return X

Multivariate Distributions

General Methods

- Not as well developed as univariate methods.
- Key requirement: to ensure an appropriate correlation structure among the components of the multivariate vector.
- **Conditional sampling:** $X = (X_1, X_2, \ldots, X_n)^T$ random vector with joint distribution $F(x_1, \ldots, x_n)$.
  - Suppose distribution of X_j given that $X_i = x_i$, for $i = 1, 2, \ldots, j-1$, is known for each j.
  - X can be built one component at a time, with each component obtained by sampling from an univariate distribution

Generate x_1 from the distribution $F_1(x)$

Generate x_2 from the distribution $F_2(x|X_1 = x_1)$

...

Generate x_n

from the distribution $F_n(x|X_1 = x_1, X_2 = x_2, \ldots, X_{n-1} = x_{n-1})$

- $X \sim MVN(\mu, \Sigma)$, μ $n \times 1$ vector, Σ $n \times n$ positive definite matrix

$$f(x) = (2\pi|\Sigma|)^{-n/2} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right], \; x_i \in \mathbb{R}, i = 1, \dots$$

- Generator: compute first the Choleski decomposition of Σ, $\Sigma = LL^T$ then generate $\mathbf{Z} = (Z_1, \dots, Z_n)^T$, $\mathbf{X} = L\mathbf{Z} + \mu$, $Z_i \sim N(0,1)$

for $(i = 1 \text{ to } n)$ $Z_i := N(0,1)$;

for $(i = 1 \text{ to } n)$ $\{$
    $X_i := \mu_i$;
    for $(j = 1 \text{ to } i)$ $X_i := X_i + L_{ij} Z_j$
$\}$
return $X = (X_1, \dots, X_n)$

Uniform Distribution on the n-Dimensional Sphere

- Components of $MVN(0, I)$ are treated as equally likely directions in $\mathbb{R}^n$
- Generator:

$S := 0;$
for $(i = 1 \text{ to } n)\{$
  $Z_i := N(0, 1);\ S := S + Z_i^2;$
$\}$
$S := \sqrt{S}$
for $(i := 1 \text{ to } n)\ X_i := Z_i / S;$
return $\mathbf{X} = (X_1, \ldots, X_n)$

Order Statistics I

- Sample $X_1, X_2, \ldots, X_n$ arranged in ascending order

$$X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$$

- Generation and reordering, time $O(n \log n)$
- If X generated by $X = F^{-1}(U)$, the sample can be generated in order from the order statistics of the uniform sample

$$U_{(1)} \leq U_{(2)} \leq \cdots \leq U_{(n)}$$

- Based on
  1. $U_{(n)}$ has an invertible distribution
  2. $U_{(1)}, U_{(2)}, \ldots, U_{(i)}$ are the order statistics of a sample of size i drawn from the distribution $U(0, U_{(i+1)})$.

$U := RN(0,1);\ U_{(n)} = U^{1/n};$
for $(i = n-1 \text{ downto } 1)\{$
  $U := RN(0,1);$

  $U_{(i)} := U_{(i+1)} U^{1/i};$
$\}$

Alternative way:

$E_1 := EXP(1);\ S_1 := E_1;$
for $(i = 2 \text{ to } n+1)\{$
  $E_i := EXP(1);\ S_i := S_{i-1} + E_i;$
$\}$
for $(i = 1 \text{ to } n)\ U_{(i)} := S_i / S_{n+1}$

Point Processes

- *Point process* = a sequence of points $t_0 = 0,\ t_1,\ \ldots$ in time
- the time intervals $x_i = t_i - t_{i-1}$ are usually random

Examples

1. t_i are arrival times of customers, x_i are interarrival times;
2. t_i moments at breakdowns, x_i lifetimes

Poisson Processes

- x_i independent $EXP(1/\lambda)$ variables$\Longrightarrow$ (t_i) Poisson process with rate λ
- to generate next time point t_i assuming that t_{i-1} has already been generated

$U := RN(0,1);$
$return\ t_i := t_{i-1} - \lambda^{-1} \ln U$

Nonstationary Poisson Processes

- The rate $\lambda = \lambda(t)$ varies with time.
- Suppose the cummulative rate

$$\Lambda(t) = \int_0^t \lambda(u) \mathrm{d}\, u$$

  is invertible with inverse $\Lambda^{-1}(.)$
- assume that previous moment s_{i-1} has been already generated; next moment t_i given by

$U := RN(0,1);\ s_i := s_{i-1} - \ln U;$
return $t_i = \Lambda^{-1}(s_i)$

Nonstationary Poisson Processes - Thinning

- Analog to acceptance-rejection method
- $\lambda_M = \max_t \lambda(t)$

$t := t_{i-1}$;
do{
    $U := RN(0,1);\ t := t - \lambda_M^{-1} \ln U;\ V = RN(0,1)$;
}while $(V > \lambda(t)/\lambda_M)$
return $t_i = t$

- **Discrete-time Markov chain**: $t = 0, 1, 2, \ldots,$ states set $X \in \{1, 2, \ldots, n\}$
- Given $X_t = i$, the next state X_{i+1} is selected according to

$$P(X_{t+1} = j | X_t = i) = p_{ij}, \qquad j = 1, \ldots, n$$

- **Continuous-time Markov chain**: assume system has just entered state i at time t_k. Then the next change of state occurs at $t_{k+1} = t_k + EXP(1/\lambda_i)$. The state entered is j with probability p_{ij}, $j = 1, 2, \ldots, n$

Time-Series Models and Gaussian Processes

- *Gaussian process* = stochastic process $X(t)$ all of whose joint distribution are multivariate normal (i.e. $X_{t_1}, X_{t_2}, \ldots, X_{t_r}$ is multivariate normal for any given set of times $t_1, t_2, \ldots, t_r$)
- A *moving average process* X_t is defined by

$$X_t = Z_t + \beta_1 Z_{t-1} + \cdots + \beta_q Z_{t-q}, \qquad t = 1, 2, 3, \ldots,$$

where Z's are independent $N(0, \sigma^2)$ normal variates and the β's are user-prescribed coefficients. The X's can be generated directly from this definition.

Autoregressive Processes

- defined by

$$X_t = \alpha_1 X_{t-1} + \cdots + \alpha_p X_{t-p} + Z_t, \qquad t = 1, 2, 3, \ldots,$$

where Z's indep. $N(0,1)$ r.v.and α's user-prescribed coefficients
- The X's can be generated from definition; the initial values X_0, X_{-1}, $\ldots$, X_{1-p} need to be obtained

$$(X_0, X_{-1}, \ldots, X_{1-p}) \sim MVN(0, \Sigma),$$

where Σ satisfies

$$\Sigma = A\Sigma A^T + B \tag{3}$$

with

$$A = \begin{bmatrix} \alpha_1 & \alpha_2 & \ldots & \alpha_{p-1} & \alpha_p \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} \sigma^2 & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix}$$

References

- Luc Devroye, *Non-uniform Random Variate Generation*, Springer, 1986
- Averill M. Law, *Simulation Modeling and Analysis*, McGraw-Hill, 2007
- J. Banks, J. S. Carson II, B. L. Nelson, D. M. Nicol, *Discrete-Event System Simulation*, Prentice Hall, 2005
- J. Banks (ed), *Handbook of Simulation*, Wiley, 1998, Chapter 8
- G. S. Fishman, *Monte Carlo. Concepts, Algorithms and Applications*, Springer, 1996
- Kromal & Peterson, Statistical Computing 33.4, pp.214-218