Lecture 3:
Random number generation

Pseudorandom numbers

Pseudo Random Number Generators:

- Create random numbers using a mathematical algorithm
- Classical examples: *Congruential generators*
- Come as embedded functions in software or can be linked as separate objects to the program code.
- The numbers are not truly random; attention must be made to the type of application.

First step: Generating U[0,1]

- Linear congruential generator

Define a sequence $\{x_k\}$ of integers according to

$$x_{k+1} = (a \cdot x_k + c) \bmod m, \quad k \geq 0$$

where x_0 is called **seed ,** "mod m" means that x_k is the remainder after division by m

- The result is an integer in the interval $[0, m - 1]$

a and c are constants in $[0, m)$, need to be carefully selected

- To obtain U[0,1] , x_i are scaled, i.e.

$$x_i := x_i / m$$

First step: Generating U[0,1]

Generated numbers will get into a "loop" with a certain *period*

Example:

Let $x_0 = a = c = 7$ and $m = 10$

➔

$$
\begin{aligned}
x_1 &= (7 \cdot 7 + 7) \bmod 10 = 56 \bmod 10 = 6 \\
x_2 &= (7 \cdot 6 + 7) \bmod 10 = 49 \bmod 10 = 9 \\
x_3 &= (7 \cdot 9 + 7) \bmod 10 = 70 \bmod 10 = 0 \\
x_4 &= (7 \cdot 0 + 7) \bmod 10 = 7 \bmod 10 = 7 \\
x_5 &= (7 \cdot 7 + 7) \bmod 10 = 56 \bmod 10 = 6 \\
&\ldots
\end{aligned}
$$

The period is thus 4 in this case

First step: Generating U[0,1]

Comments:

- Obviously, period can not be larger than m
- Period and other constants should be carefully chosen, m is typically very large
- Seed defines the sequence of random numbers, if seed is fixed by program – same sequence will be produced
- Other methods for generating U[0,1] are available (i.e. generalized feedback shift register)

Generation U[a,b]

- U(0,1) can be transformed to U(a, *b*):

$$X = a + U \cdot (b - a)$$

- *U* can also be transformed to *discrete* uniform distribution on the integers (1, …, *n*) by

$$X = [n \cdot U] + 1$$

where [·] depicts the integer part.

Question and exercise:

- Why do we need to add “1”?
- How can *U* be transformed to a random variable *Y* with a discrete uniform distribution on the integers (50, 55, 60) ?

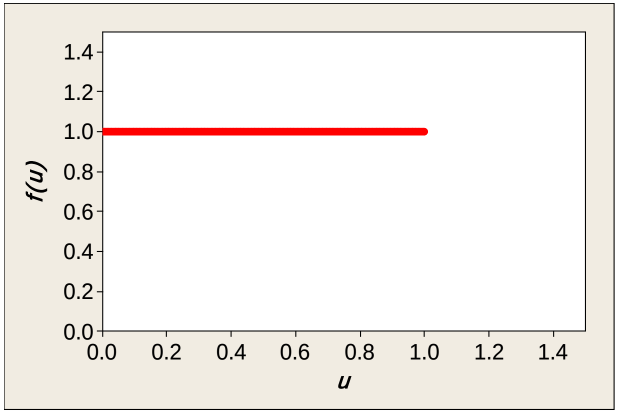
Generation of nonuniform random numbers

U(0,1)

- Let U be a random variable uniformly distributed on (0,1)
- Let F_U be its cumulative distribution function, i.e.

$$F_U(u) = P(U \le u)$$

- The probability density function (pdf) of U is



$$f_U(u) = \begin{cases} 1 & 0 < u < 1 \\ 0 & u \notin (0,1) \end{cases}$$

Inverse CDF method

Let X be a random variable with CDF F_X .

Set $Y = F_X^{-1}(U)$ where U is U(0,1)

The CDF of Y is now

$$
\begin{aligned}
&F_Y(y) = P(Y \leq y) = P\left(F_X^{-1}(U) \leq y\right) = \\
&= P\left(F_X\left(F_X^{-1}(U)\right) \leq F_X(y)\right) = P(U \leq F_X(y)) = \\
&= F_U(F_X(y)) = F_X(y)
\end{aligned}
$$

as $0 \leq F_X(y) \leq 1$ and $F_U(u) = u$ for $0 \leq u \leq 1$

➔ ***Y* has the same probability distribution as *X* !**

Inverse CDF method

- If U is U(0,1) then a realization of a random variable X with CDF F_X can be obtained by

$$X = F_X^{-1}(U)$$

provided F_X^{-1} can be evaluated

- The realization U comes from a RNG

Example

Let X be exponentially distributed, i.e. with pdf

$$f_X(x)=\begin{cases}\lambda e^{-\lambda x} & x\geq 0\\ 0 & x<0\end{cases} \quad ; \lambda = 1/E(X)>0$$

$$\Rightarrow \quad F_X(x)=\int_{-\infty}^{x} f_X(t)dt=\int_0^x \lambda e^{-\lambda t}dt=\left[-e^{-\lambda t}\right]_0^x=1-e^{-\lambda x}$$

for $x \geq 0$ [$F_X(x) = 0$ for $x < 0$]

Inverse CDF method

Example(cont.)

To find F_X^{-1} solve for x the equation

$$y = 1 - e^{-\lambda x}$$
$$\Rightarrow e^{-\lambda x} = 1 - y$$
$$\Rightarrow x = -\frac{\ln(1-y)}{\lambda}$$
$$\Rightarrow F_X^{-1}(y) = -\frac{\ln(1-y)}{\lambda}$$

Thus the transform from U to X becomes $$X = -\frac{\ln(1-U)}{\lambda}$$

Inverse CDF method – discrete variables

1. Define distribution $P(X=x_i)=p_i$
2. Generate *U* from U(0,1)
3. If $U \le p_0$, deliver $X=x_0$
4. If $U \le p_0 + p_1$, deliver $X=x_1$
5. …
6. Repeat procedure from step 2

Inverse CDF method

1. When the inverse cumulative distribution can be explicitly derived $\rightarrow$ No problem!

2. When not $\rightarrow$ Numerical solution necessary $\rightarrow$ Usually time-consuming

- Unfortunately, situation 2 is quite typical, ex.: normally distributed random variables

Generating N(0,1)

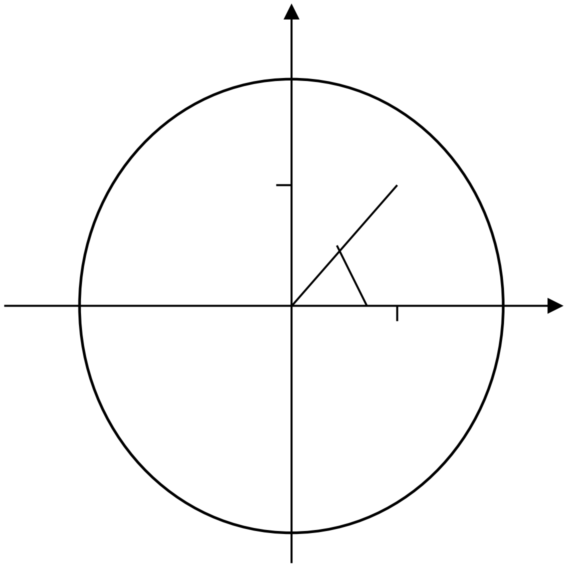
Assume

- $\Theta \in U(0, 2\pi)$
- $D \in U(0,1)$

Algorithm 1

1. Generate D and Θ
2. Generate X_1 and X_2 as

$$X_1 = \sqrt{-2\ln D}\cos\Theta$$
$$X_2 = \sqrt{-2\ln D}\sin\Theta$$



X_1 and X_2 are independent and normally distributed (see proof…)

Acceptance/rejection methods

- Idea: to generate Y with PDF f_y similar to some known PDF f_x
- Requirement: There should exist constant *c* such that

$$cf_Y(x) \geq f_X(x) \quad \textit{for all } x$$

- $f_Y(x)$ majorizing density, proposal density
- $f_X(x)$ target density
- c majorizing constant

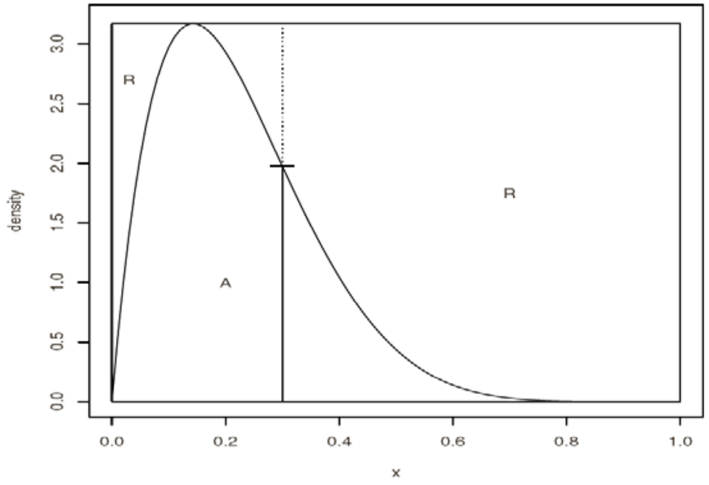


Fig. 7.1. Beta (2, 7) Density with a Uniform Majorizing Density

Acceptance/rejection methods

Algorithm

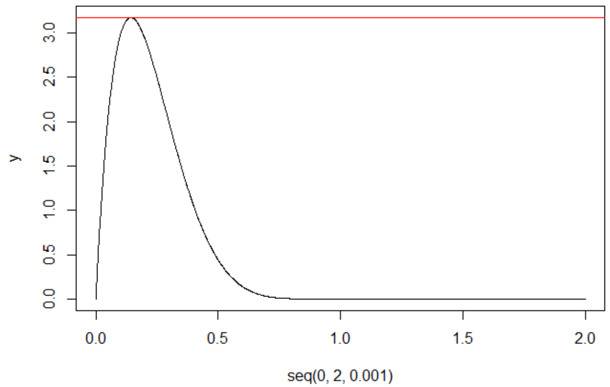
1. Generate Y from distribution with density f_Y
2. Generate U from U(0,1)
3. If $U \le \frac{f_X(Y)}{c f_Y(Y)}$, take Y else return to step 2

- It can be seen that variables obtained are from f_X
- Larger *c* lead to larger rejection rates ***R***
- The value of *c* should be as small as possible (minimize ***R***)
- The method works for multivariate random cases, but the rejection proportion can be high (curse of dimensionality)

Acceptance/rejection methods

- Generation beta(2,7)

```
> y=dbeta(seq(0,2,0.001), 2,7)
> max(y)
[1] 3.172554
```



- Algorithm
  1. Generate Y from *U[0,1]*
  2. Generate U from *U[0,1]*
  3. If $U \leq \frac{dBeta(Y|sh_1=2,sh_2=7)}{3.173 \cdot 1}$, take Y else return to step 2

- Observe acceptance and rejection areas
- One could take $c = 4$→ what are consequences?

Generating multivariate normal

- Acceptance/rejection is difficult to apply
  - Difficult to determine majorizing density
  - High rejection rate

Suppose we need to generate $N(\mu,\Sigma)$:

1. Take i.i.d. $N(0,1)$ sequence $X=(X_1,\dots X_n)$
2. Compute Cholesky factor or matrix square root, i.e. matrix A: $AA^T= \Sigma$
3. Compute Y as $\mu+AX$

Observe: $EY= \mu$, $cov(Y)=AA^T$

Random numbers in R

- Use d for density p for CDF q for quantiles and r for simulation:

(ex: rnorm pnorm dnorm qnorm)

Distribution	R name	additional arguments
beta	beta	shape1, shape2, ncp
binomial	binom	size, prob
Cauchy	cauchy	location, scale
chi-squared	chisq	df, ncp
exponential	exp	rate
F	f	df1, df2, ncp
gamma	gamma	shape, scale
geometric	geom	prob
hypergeometric	hyper	m, n, k
log-normal	lnorm	meanlog, sdlog
logistic	logis	location, scale
negative binomial	nbinom	size, prob
normal	norm	mean, sd
Poisson	pois	lambda
Student's t	t	df, ncp
uniform	unif	min, max
Weibull	weibull	shape, scale
Wilcoxon	wilcox	m, n

Recommended reading

- Chapter 7.1-7.3