# Chapter 5:
# Generating Random Numbers from Distributions

Refer to readings

# Review

## Inverse Transform

- Generate a number $\mathbf{u}_i$ between 0 and 1 (one U-axis) and then find the corresponding $\mathbf{x}_i$ coordinate by using $\mathbf{F^{-1}(\cdot)}$.
- One-to-one mapping between $\mathbf{u}_i$ and $\mathbf{x}_i$.

## Continuous Distributions

- General PDF
- Exponential ($\lambda$)
- Uniform (a,b)
- Weibull Distribution

## Discrete Distributions

- General PMF
- Bernoulli (p)
- Binomial (n,p)
- Geometric (p)

# 3. Convolution Generation

- Using random variables related to each other through some functional relationship.

- ***The convolution relationship***:

The distribution of the sum of two or more random variables is called the *convolution*.

Let $Y_i \sim G(y)$ be IID random variables.

$$X = \sum_{i=1}^{n} Y_i$$

Then the distribution of $X$ is said to be the convolution of $Y$.

# 3. Convolution Generation

Some common random variables with convolution:

- Binomial Variable = $\Sigma$ *iid* Bernoulli variables
- Negative Binomial = $\Sigma$ *iid* Geometric variables
- Erlang Variable = $\Sigma$ *iid* Exponential variables.
- Normal Variable = $\Sigma$ *iid* other Normal variables.
- Chi-squared Variable = $\Sigma$ *iid* Squared normal variables.

# 3. Convolution Generation

The Convolution Algorithm:

1. simply generates $Y_i \sim G(y)$
2. sum the generated random variables.
3. The result is the needed variable.

**Example**

Generate a random variable from Erlang Distribution with parameters $r$ and $\lambda$ .

# 3. Convolution Generation

<u>**Example**</u>

Generate a random variable X from Erlang Distribution with parameters *r* and λ .

From Probability theory:

Erlang Variable X with parameters ($r$, λ)

$= \sum_{\boldsymbol{r}}$ *iid* Exponential variables with parameter λ.

Then, generate ***r*** nubers: $Y_i$ exponentially distributed with rate parameter $\lambda$. Then add them to get one value of Erlang distribution

# 3. Convolution Generation

**Example**

Generate a random variate from an Erlang distribution having parameters $r = 3$ and $\lambda = 0.5$ using the following pseudorandom numbers

$$u_1 = 0.35,\ u_2 = 0.64, \text{ and } u_3 = 0.14,$$

Then, X~ Erlang($r$=3, $\lambda$ = 0.5)

$$X = Y_1 + Y_2 + Y_3$$

With $Y_1$ , $Y_2$ , $Y_3$ are all IID exponentially distributed with parameter $\lambda$.

# 3. Convolution Generation

**Example**

$$X = Y_1 + Y_2 + Y_3$$

For $Y_i$ exponential with parameter $\lambda$ =0.5

$$Y = -\frac{1}{\lambda}\ln(1-u)$$

- $u_1 = 0.35 \rightarrow Y_1 = -\frac{1}{0.5}\ln(1-0.35) = 0.8616$
- $u_2 = 0.64 \rightarrow Y_2 = -\frac{1}{0.5}\ln(1-0.65) = 2.0433$
- $u_3 = 0.14 \rightarrow Y_3 = -\frac{1}{0.5}\ln(1-0.14) = 0.3016$

$$X = Y_1 + Y_2 + Y_3 = 0.8616 + 2.0433 + 0.3016 = 3.2065$$

# 3. Convolution Generation

**Generating from a Poisson Distribution**:

Let X(t) represent the number of events happened in an interval of length *t*, where *t* is measured in hours. Suppose X(t) has a Poisson distribution with mean rate $\lambda$ event per hour.

$$P\{X = x\} = \frac{e^{-\lambda}\lambda^{x}}{x!} \quad \lambda > 0, \quad x = 0, 1, \ldots$$

$$\mathrm{E}[X] = \lambda$$

$$Var[X] = \lambda$$

***We need to generate number of events in one hour***

# 3. Convolution Generation

**Generating from a Poisson Distribution**:

From probability theory

If X(t) number of events with Poisson distribution $\lambda$ event per hour, ***then the time between two events is exponential with rate $\lambda$***

- Let $T_i$= is the time between event ($i$) and ($i$-1)

Then $T_i \sim \text{Exp}(\lambda)$

- Let $A_k$= is the occurrence time of event ($k$)

# 3. Convolution Generation

**Generating from a Poisson Distribution**:

This means that

- Event #1 happened at time $A_1 = T_1$
- Event #2 happened at time $A_2 = T_1 + T_2$
- Event #3 happened at time $A_3 = T_1 + T_2 + T_3$
- Event #4 happened at time $A_4 = T_1 + T_2 + T_3 + T_4$

$$A_k = \sum_{i=1}^{k} T_i$$

- Then $A_k$= is Erlang distributed with $r=k$ and $\lambda$
- **To generate number of events in one hour, generate $A_k$ until you reach $A_k > 1$ hour**

# 3. Convolution Generation

**Generating from a Poisson Distribution**:

**<u>Example:</u>**

Let $X(t)$ represent the number of customers that arrive to a bank in an interval of length $t$, where $t$ is measured in hours. Suppose $X(t)$ has a Poisson distribution with mean rate $\lambda = 4$ per hour. Generate the number of arrivals in 2 hours.

- Because the time between events $T$ will have an exponential distribution with mean $0.25 = 1/\lambda$. We generate exponential values and some them.

# 3. Convolution Generation

## Generating from a Poisson Distribution:

**Example:**

$$T_i = \frac{-1}{\lambda} \ln\,(1 - u_i) = -0.25 \ln\,(1 - u_i)$$

$$A_i = \sum_{k=1}^{i} T_k$$

we can compute $T_i$ and $A_i$ until $A_i$ goes over 2 hours.

Total number of arrivals in 2 hours

| $i$ | $u_i$ | $T_i$ | $A_i$ |
|---|---|---|---|
| 1 | 0.971 | 0.881 | 0.881 |
| 2 | 0.687 | 0.290 | 1.171 |
| 3 | 0.314 | 0.094 | 1.265 |
| 4 | 0.752 | 0.349 | 1.614 |
| 5 | 0.830 | 0.443 | 2.057 |

The arrival of last customer before 2 hours

Since the fifth arrival occurs after 2 hours, $X(2) = 4$

# 4. Acceptance/Rejection Method

- We need to get a sample from density function (PDF), $f(x)$
- The probability density function (PDF), $f(x)$, is complicated or has no closed form for CDF.

***Idea:***

- Replace $f(x)$ by a simple PDF, $w(x)$, which can be sampled from more easily.
- $w(x)$ is based on the development of a majorizing function for $f(x)$.

# 4. Acceptance/Rejection Method

- A majorizing function, $g(x)$, for $f(x)$, is a function such that $g(x) \geq f(x)$ for $-\infty < x < +\infty$

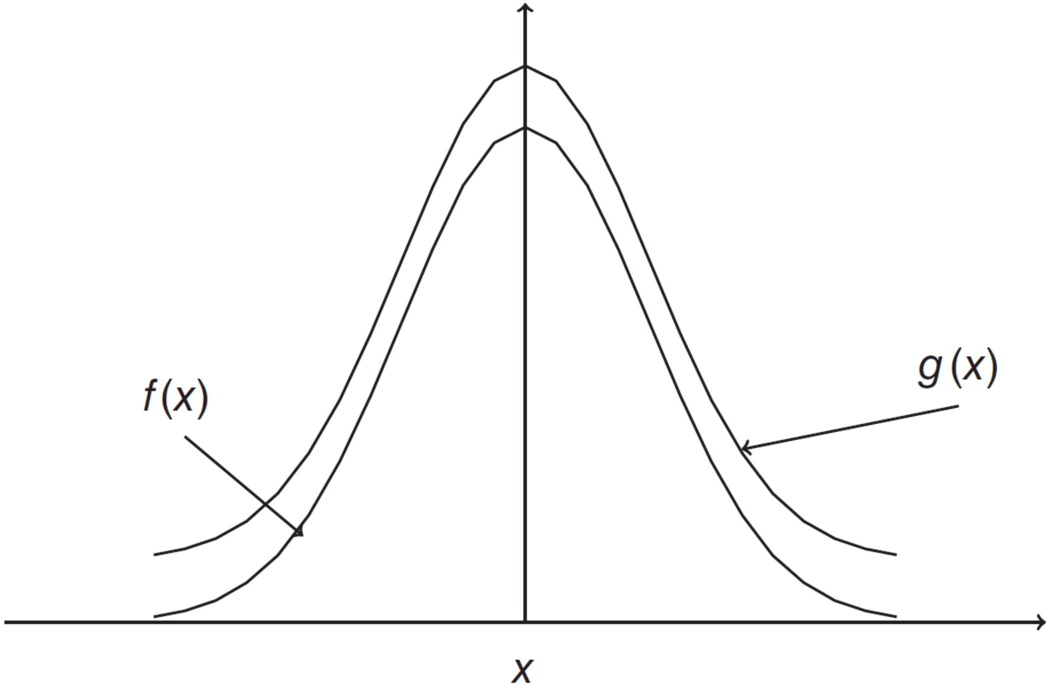



**Figure 2.5** Illustration of a majorizing function.

# 4. Acceptance/Rejection Method

- Transform the majorizing function, $g(x)$, to a density function

- majorizing function for $f(x)$, $g(x)$ must have finite area,

$$c = \int_{-\infty}^{+\infty} g(x)\, dx$$

**If $w(x)$ is defined as $w(x) = g(x)/c$, then $w(x)$ will be a PDF**

# 4. Acceptance/Rejection Method

The acceptance–rejection method for $f(x)$ :

- start by obtaining a random number $W$ from a simple function $w(x)$.
- $w(x)$ should be chosen to be easily sampled, for example, via the inverse transform method.
- Let $U \sim U(0, 1)$ and check if

$$\frac{f(W)}{g(W)} \geq U$$

Then W $\sim f(x)$

- Continue sampling of $U$ and $W$ until the condition is satisfied

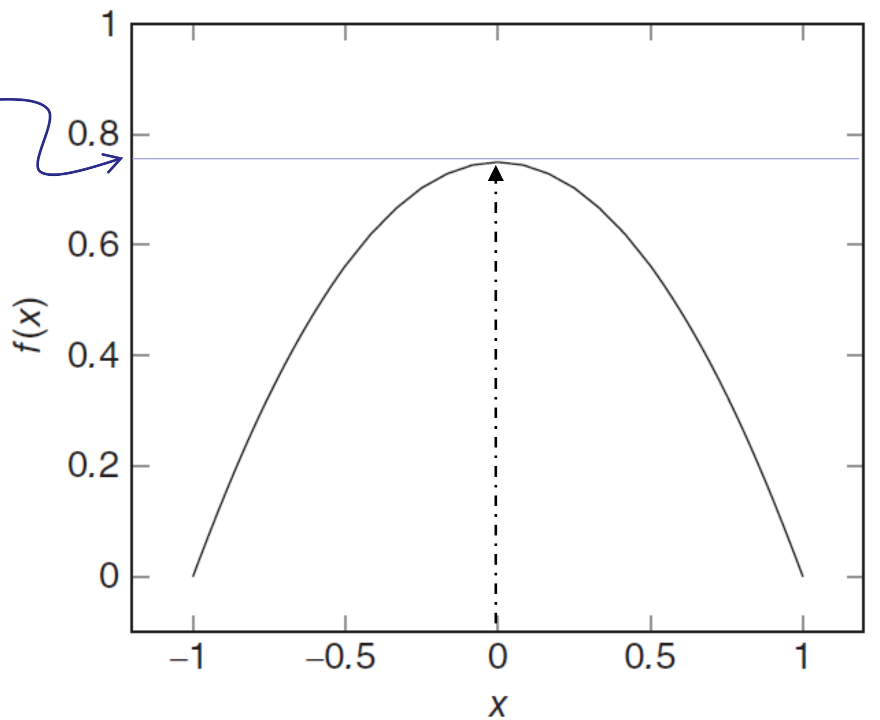
# 4. Acceptance/Rejection Method

**EXAMPLE 2.12 Acceptance–Rejection Method**

Consider the following PDF over the range $[-1, 1]$. Develop an acceptance/rejection-based algorithm for $f(x)$.

$$f(x) = \begin{cases} \frac{3}{4}\left(1 - x^2\right) & -1 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

*g(x)*




**Start**
choosing a simple function g(x) so that:
g(x) > f(x) for all x ∈[-1,1]

Let g(x) be the max f(x)

*Then*
g(x) = max{f(x)} = ¾

# 4. Acceptance/Rejection Method

$$c = \int_{-1}^{1} g(x)dx = \int_{-1}^{1} \frac{3}{4} dx = \frac{3}{2}$$

$$w(x) = \begin{cases} \frac{1}{2} & -1 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$

```
REPEAT
    Generate U_1 ~ U(0,1)
    W = -1 + 2U_1
    Generate U_2 ~ U(0,1)
    f = 3/4 (1 - W^2)
UNTIL U_2 × 3/4 ≤ f
RETURN W
```

**Second**

Find the constant ***c*** that makes the function g(x) a pdf function for all x between [-1,1], by integration g(x) for all x $\in$[-1,1]

*Then*

w(x) = g(x)/***c***

**Third**

Using U[0,1]: Generate **W** from w*(x)* and use it for *f*(**W**) and *g*(**W**)

**Last**

Decide using new U:

Accept$\rightarrow$ if *f*(**W**)/*g*(**W**) $\ge U_{new}$

**or**

Reject $\rightarrow$ if *f*(**W**)/*g*(**W**) $< U_{new}$

# 4. Acceptance/Rejection Method

| n | U1 | W | f(W) | U2 | g(W) | f(W)/g(W) | Acc./Rej. | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.622 | 0.243 | 0.706 | 0.311 | 0.75 | 0.941 | Accept | W ~ $f(x)$ |
| 2 | 0.943 | 0.885 | 0.162 | 0.964 | 0.75 | 0.216 | Reject | No |
| 3 | 0.851 | 0.702 | 0.381 | 0.827 | 0.75 | 0.508 | Reject | No |
| 4 | 0.592 | 0.183 | 0.725 | 0.186 | 0.75 | 0.966 | Accept | W ~ $f(x)$ |
| 5 | 0.084 | -0.833 | 0.230 | 0.165 | 0.75 | 0.307 | Accept | W ~ $f(x)$ |
| 6 | 0.936 | 0.873 | 0.179 | 0.684 | 0.75 | 0.238 | Reject | No |
| 7 | 0.016 | -0.969 | 0.046 | 0.768 | 0.75 | 0.062 | Reject | No |
| 8 | 0.219 | -0.562 | 0.513 | 0.667 | 0.75 | 0.685 | Accept | W ~ $f(x)$ |
| 9 | 0.091 | -0.818 | 0.248 | 0.257 | 0.75 | 0.331 | Accept | W ~ $f(x)$ |
| 10 | 0.238 | -0.524 | 0.544 | 0.280 | 0.75 | 0.725 | Accept | W ~ $f(x)$ |
| 11 | 0.057 | -0.886 | 0.162 | 0.318 | 0.75 | 0.215 | Reject | No |
| 12 | 0.236 | -0.528 | 0.541 | 0.270 | 0.75 | 0.721 | Accept | W ~ $f(x)$ |
| 13 | 0.119 | -0.762 | 0.315 | 0.890 | 0.75 | 0.419 | Reject | No |
| 14 | 0.375 | -0.250 | 0.703 | 0.163 | 0.75 | 0.938 | Accept | W ~ $f(x)$ |
| 15 | 0.012 | -0.976 | 0.035 | 0.685 | 0.75 | 0.047 | Reject | No |
| 16 | 0.664 | 0.328 | 0.669 | 0.904 | 0.75 | 0.892 | Reject | No |
| 17 | 0.375 | -0.249 | 0.703 | 0.015 | 0.75 | 0.938 | Accept | W ~ $f(x)$ |
| 18 | 0.126 | -0.749 | 0.330 | 0.776 | 0.75 | 0.439 | Reject | No |
| 19 | 0.550 | 0.100 | 0.742 | 0.395 | 0.75 | 0.990 | Accept | W ~ $f(x)$ |
| 20 | 0.868 | 0.736 | 0.343 | 0.570 | 0.75 | 0.458 | Reject | No |

# 4. Acceptance/Rejection Method

- **Quiz**

Consider the following pdf

$$f(x) = \frac{1}{32}(8 - x^3); \qquad -2 \le x \le 2$$

Use Acceptance/Rejection method to generate random numbers using

| n | U1 | W | f(W) | U2 | g(W) | f(W)/g(W) | Acc./Rej. | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.622 | | | 0.311 | | | | |
| 2 | 0.943 | | | 0.964 | | | | |
| 3 | 0.851 | | | 0.827 | | | | |
| 4 | 0.592 | | | 0.186 | | | | |
| 5 | 0.084 | | | 0.165 | | | | |

# 4. Acceptance/Rejection Method

**Best choices of majorizing function** g(x)

- Always choose the function g(x) to be simple to generate from using inverse transform
- Always choose the function g(x) to reduce rejection rate

*g(x)*
*Good choice*
*Less rejection rate*