Lecture Notes 6
Generating Random Samples

Prof. Marvin Nakayama

- Generate observations from probability distributions
  - Inverse-transform method
  - Convolution method
  - Composition method
  - Acceptance-rejection method
  - Other special methods
- Generate stochastic processes
- Generate observations from multivariate distributions
- Be sure to check course webpage for reading assignment and HW

Introduction

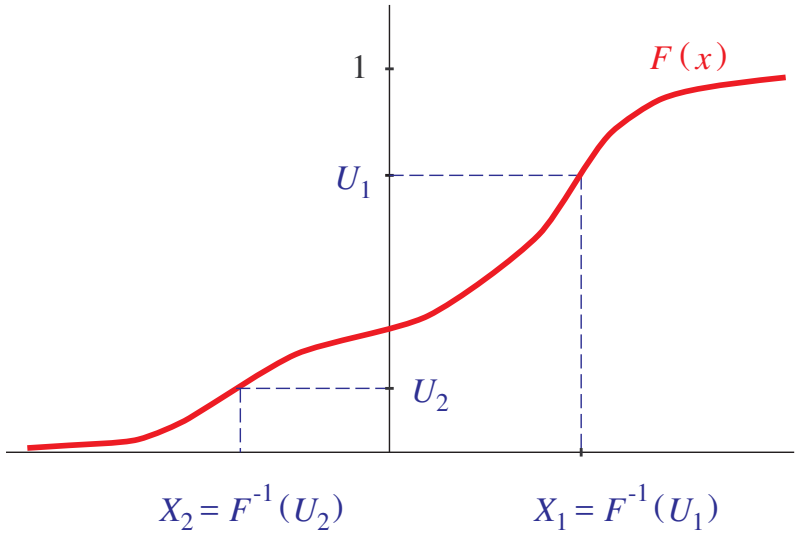
- Methods to generate observation of random variable $X \sim F$
  - exponential, gamma, normal, etc.
  - Generated observation known as **random variate**
- Methods take one or more unif[0, 1] random numbers and transforms them into an observation from F
- Want methods that are exact, fast, and use little storage.
- Why study methods when simulation packages have built-in fcns?
  - Stochastic model may have non-standard distributions.
  - Variance-reduction techniques may require generating observations in specific way.
  - Simulation packages often slow compared to code written in high-level programming language.

Continuous Random Variables

- Generate random variate X having CDF F as follows :

> 1. Generate a uniform[0, 1] random number U
> 2. Set $X = F^{-1}(U)$

- Assumption: Inverse F^{-1} exists
  - F^{-1} satisfies $F^{-1}(F(x)) = x$ and $F(F^{-1}(y)) = y$
  - e.g., if $F(x) = x^2$ for $0 \le x \le 1$, then $F^{-1}(y) = \sqrt{y}$ for $0 \le y \le 1$.
- Using inverse and hence ... **Inverse-Transform Method** or simply **Inversion**



Proof of Correctness of Inversion:

- Recall: if $U \sim$ unif[0, 1], then

$$P(U \leq y) = y, \quad \text{for } \ y \in [0, 1]. \tag{1}$$

- Need to show CDF of random variate $X = F^{-1}(U)$ produced by inversion is precisely $F(x)$. For any $x \in \mathbf{R}$,

$$\begin{aligned} \mathcal{P}(X \leq x) &= \mathcal{P}(F^{-1}(U) \leq x) \\ &= \mathcal{P}(U \leq F(x)) \qquad (2) \\ &= F(x) \qquad (3) \end{aligned}$$

where

- Eq. (2) follows by fact that F is increasing function
- Eq. (3) follows from fact that $0 \leq F(x) \leq 1$ and from Eq. (1)

Algorithm for Inverse-Transform Method:

- Given: CDF $F(x)$ or density $f(x)$.
  - If density f given, then first integrate to get CDF F.
  - Most frequent error: incorrect limits on the integration
- Set $F(X) = U$ and solve for X in terms of U.
  - Make sure that solution X lies in correct range.

Example : Generate $X \sim$ exp(λ) Using Inversion

- Density

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{for } x \geq 0, \\ 0 & \text{for } x < 0. \end{cases}$$

- CDF

$$F(x) = \begin{cases} 1 - e^{-\lambda x} & \text{for } x \geq 0, \\ 0 & \text{for } x < 0. \end{cases}$$

- Set $F(X) = U$ and solve for X :

$$\begin{aligned} 1 - e^{-\lambda X} &= U \\ \iff \quad e^{-\lambda X} &= 1 - U \\ \iff \quad X &= -\frac{1}{\lambda} \ln(1 - U) \end{aligned}$$

- Algorithm to generate $X \sim$ expon(λ):

> 1. Generate unif[0,1] random number U
> 2. Set $X = -\frac{1}{\lambda} \ln(1 - U)$

- In Excel, can use the formula

```
=-LN(1-RAND())/LAMBDA
```

where `LAMBDA` is a user-defined constant.

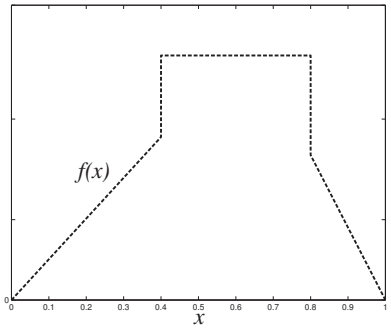
- When U is uniform[0,1], so is $1 - U$, so can instead define

$$X = -\frac{1}{\lambda} \ln(U)$$

Example : Inversion with density f

- Density

$$f(x) = \begin{cases} 45x/16 & \text{if } 0 \le x \le 0.4 \\ 27/16 & \text{if } 0.4 < x \le 0.8 \\ 5 - 5x & \text{if } 0.8 < x \le 1 \\ 0 & \text{otherwise} \end{cases}$$



- Compute CDF F:
  - If $0 \le x \le 0.4$, then

$$F(x) = \int_0^x \frac{45}{16} u \, du = \frac{45}{32} x^2$$

  - If $0.4 < x \le 0.8$, then

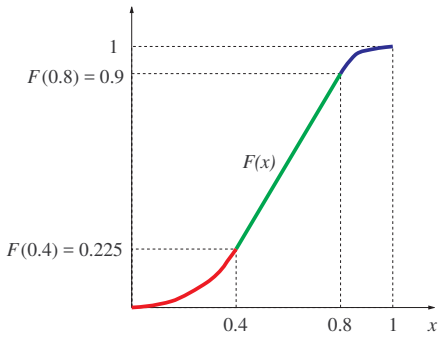
$$F(x) = \int_0^{0.4} \frac{45}{16} u \, du + \int_{0.4}^x \frac{27}{16} du = \frac{27}{16} x - \frac{9}{20}$$

  - If $0.8 < x \le 1$, then

$$\begin{aligned} F(x) &= \int_0^{0.4} \frac{45}{16} u \, du + \int_{0.4}^{0.8} \frac{27}{16} du + \int_{0.8}^x (5 - 5u) \, du \\ &= -\frac{5}{2} x^2 + 5\,x - \frac{3}{2} \end{aligned}$$

- CDF

$$F(x) = \begin{cases} 0 & \text{if } x < 0 \\ 45x^2/32 & \text{if } 0 \le x \le 0.4 \\ 27x/16 - 9/20 & \text{if } 0.4 < x \le 0.8 \\ -5x^2/2 + 5x - 3/2 & \text{if } 0.8 < x \le 1 \\ 1 & \text{if } x > 1 \end{cases}$$



- Solve $X = F^{-1}(U)$:
  - If $0 \le U \le F(0.4) = 0.225$, then

$$F(X) = \frac{45}{32} X^2 = U \quad \Longrightarrow \quad X = \sqrt{\frac{32}{45} U}$$

  - If $F(0.4) = 0.225 < U \le F(0.8) = 0.9$, then

$$F(X) = \frac{27}{16} X - \frac{9}{20} = U \quad \Longrightarrow \quad X = \frac{16}{27} U + \frac{4}{15}$$

  - If $F(0.8) = 0.9 < U \le 1$, then

$$F(X) = -\frac{5}{2} X^2 + 5X - \frac{3}{2} = U \quad \Longrightarrow \quad X = 1 \pm \sqrt{\frac{2(1-U)}{5}}$$

    ▷ We take $X = 1 - \sqrt{\frac{2(1-U)}{5}}$ so that $0.8 < X \le 1$ when $0.9 < U \le 1$.

- Algorithm to generate X with density f using inversion:

> 1. Generate unif[0,1] random number U.
> 2. If $0 \le U \le 0.225$, set $X = \sqrt{\frac{32}{45}U}$ and stop.
> 3. If $0.225 < U \le 0.9$, set $X = \frac{16}{27}U + \frac{4}{15}$ and stop.
> 4. If $0.9 < U \le 1$, set $X = 1 - \sqrt{\frac{2(1-U)}{5}}$ and stop.

- In Excel, can use the following formula:

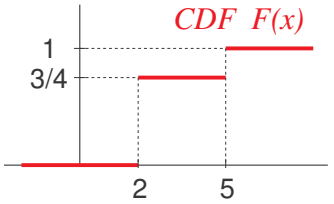
```
=IF(A2<0.225,SQRT(32*A2/45),
  IF(A2<0.9,16*A2/27+4/15,1-SQRT(2*(1-A2)/5)))
```

where cell A2 has formula `=RAND()`

Inverse-transform method : Discrete random variables

- **Goal:** generate random variate X having PMF

$$\mathcal{P}(X = 2) = \frac{3}{4}, \qquad \mathcal{P}(X = 5) = \frac{1}{4}$$



- Algorithm

> 1. Generate $U \sim$ unif$[0, 1]$.
> 2. If $0 \le U \le 3/4$, set $X = 2$ and stop.
> 3. If $3/4 < U \le 1$, set $X = 5$ and stop.

- More generally, to generate *discrete* RV X with PMF

$$\mathcal{P}(X = x_i) = p_i, \quad i = 1, 2, \ldots, m,$$

use following algorithm:

> 1. Generate unif[0,1] random number U
> 2. *Transform* U into X as follows:
>
> $$X = x_j \quad \text{if} \quad \sum_{i=1}^{j-1} p_i < U \le \sum_{i=1}^{j} p_i$$

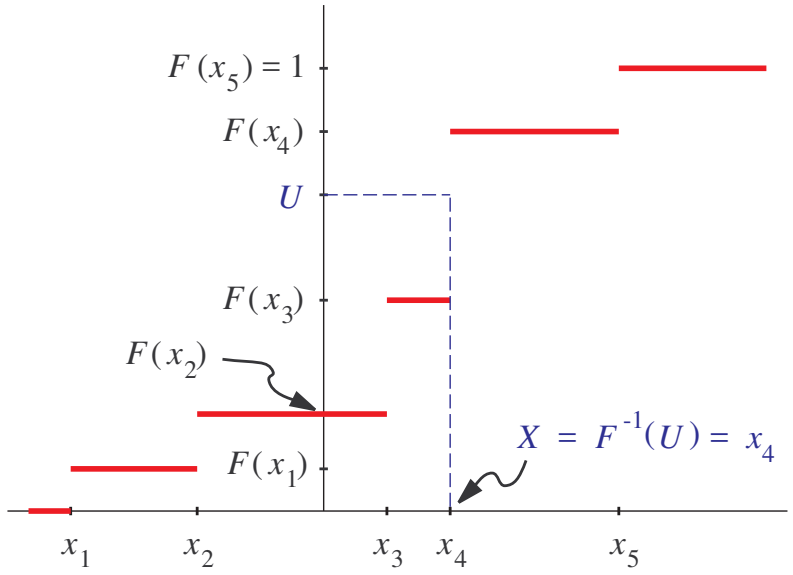
- Proof algorithm works ...

$$\begin{aligned}\mathcal{P}(X = x_j) &= \mathcal{P}\left(\sum_{i=1}^{j-1} p_i < U \le \sum_{i=1}^{j} p_i\right) \\ &= \sum_{i=1}^{j} p_i - \sum_{i=1}^{j-1} p_i = p_j.\end{aligned}$$

- Suppose $x_1 < x_2 < \cdots < x_m$

- Then CDF F of X is

$$F(x) = \begin{cases} 0, & \text{if } x < x_1, \\ \sum_{i=1}^{j} p_i, & \text{if } x_j \le x < x_{j+1}, \quad j \le m-1, \\ 1, & \text{if } x \ge x_m \end{cases}$$



- Thus, $X = x_j$ if $F(x_{j-1}) < U \leq F(x_j)$
- If one defines the *generalized* inverse F^{-1} by

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

then ... $X = F^{-1}(U)$

Example : simple distribution

- **Goal:** generate random variate X with PMF $p_i = \mathcal{P}\{X = x_i\}$, $i = 1, 2, 3, 4$:

$$p_1 = 0.20, \quad p_2 = 0.15, \quad p_3 = 0.25, \quad p_4 = 0.40$$

- Algorithm to generate $X \sim \{p_1, \ldots, p_4\}$:

1. Generate unif[0,1] random number U.
2. If $U \leq p_1 = 0.2$, set $X \leftarrow x_1$ and stop.
3. If $U \leq p_1 + p_2 = 0.35$, set $X \leftarrow x_2$ and stop.
4. If $U \leq p_1 + p_2 + p_3 = 0.60$, set $X \leftarrow x_3$ and stop.
5. Else set $X \leftarrow x_4$ and stop.

- Is this most efficient way of generating the random variate ?
  - Amount of time proportional to *number of intervals* tested.
  - Better to consider values x_j in *decreasing* order of p_j's.
- More efficient algorithm :

1. Generate $U \sim$ unif[0, 1].
2. If $U \leq p_4 = 0.4$, set $X \leftarrow x_4$ and stop.
3. If $U \leq p_4 + p_3 = 0.65$, set $X \leftarrow x_3$ and stop.
4. If $U \leq p_4 + p_3 + p_1 = 0.85$, set $X \leftarrow x_1$ and stop.
5. Else set $X \leftarrow x_2$ and stop.

Example : Geometric distribution

- Geometric distribution with parameter p, $0 < p < 1$.
- PMF of geometric random variable X:

$$\mathcal{P}(X = i) \;=\; pq^{i-1}, \quad i \geq 1, \;\; q = (1-p)$$

- X = number of IID trials needed to get first success.
  - Each trial has probability p of success.
- CDF of geometric

$$\begin{aligned} F(i) &= \mathcal{P}(X \leq i) \\ &= 1 - \mathcal{P}(X > i) \\ &= 1 - \mathcal{P}(\text{first } i \text{ trials are failures}) \\ &= 1 - q^i \end{aligned}$$

- In this case we can obtain simple expression for $F^{-1}(U)$.

- Transformation : $X = j$ iff

$$\begin{aligned} F(j-1) \;<\; U \;\leq\; F(j) \quad &\Longleftrightarrow \quad 1 - q^{j-1} \;<\; U \;\leq\; 1 - q^j \\ &\Longleftrightarrow \quad q^j \;\leq\; 1 - U \;<\; q^{j-1} \end{aligned}$$

- Since $q^j \downarrow$ as $j \uparrow$,

$$\begin{aligned} X &= \min\{j : \; q^j \leq 1 - U\} \\ &= \min\{j : \; j \ln(q) \leq \ln(1-U)\} \\ &= \min\left\{j : \; j \geq \frac{\ln(1-U)}{\ln(q)}\right\} \end{aligned}$$

- Thus,

$$X = \left\lceil \frac{\ln(1-U)}{\ln(1-p)} \right\rceil$$

where $\lceil \cdot \rceil$ is round-up function.

- In Excel, can use the formula `=ROUNDUP((LN(1-RAND())/LN(1-P)))` where `P` is the given parameter (success probability) of geometric distn.

Example : Binomial random variable

- $X \sim$ binomial(n, p)
  - n = integer number of IID trials (fixed)
  - p = probability of success in each trial
  - X = number of successes out of the n trials
  - PMF

$$p_i \;=\; \mathcal{P}(X = i) \;=\; \binom{n}{i} p^i (1-p)^{n-i}, \quad i = 0, 1, \ldots, n$$

- Since $\binom{n}{i} = n!/[i!(n-i)!]$, we get the recursion

$$p_{i+1} \;=\; \left(\frac{n-i}{i+1}\right)\left(\frac{p}{1-p}\right) p_i$$

- No simple closed-form solution for computing F^{-1} ... algorithm

- Algorithm to generate $X \sim$ binomial(n, p) using inversion:

1. Generate $U \sim$ unif[0, 1]
2. Set $i \leftarrow 0$,
   $P \leftarrow (1-p)^n$,
   $F \leftarrow P$
3. If $U \leq F$, set $X \leftarrow i$ and stop
4. Set $P \leftarrow \left(\frac{n-i}{i+1}\right)\left(\frac{p}{1-p}\right) P$,
   $F \leftarrow F + P$,
   $i \leftarrow i + 1$
5. Goto step 3

Disadvantages and Advantages of Inversion

Disadvantages:

1. Need closed-form expression for CDF F so can solve for F^{-1} (or at least we need to be able to numerically evaluate F^{-1})
2. Speed ... often slow because of number of comparisons required

Advantages:

Inversion often preserves *monotonicity* and *correlation*, which help in

1. Variance-reduction techniques
2. Generating truncated distributions
3. Order statistics

Convolution method

- Suppose X is sum of independent random variables $Z_1, Z_2, \ldots, Z_m$, i.e.,

$$X = Z_1 + Z_2 + \cdots + Z_m$$

where $Z_i \sim F_i$ and are all independent.

- Algorithm to generate X :

1. Generate $U_1, U_2, \ldots, U_m \sim$ unif[0, 1], indep
2. Inverse-transform method : $Z_i = F_i^{-1}(U_i)$, $i = 1, 2, \ldots, m$
3. Set $X = \sum_{i=1}^{m} Z_i$

- Can replace step 2 with any method to generate Z_i

- Why is this called the *convolution method* ?
  - Let Z_1, Z_2 be independent with densities f_1 and f_2, resp.
  - Let $X = Z_1 + Z_2$
  - Density f_X of X is

$$f_X(x) = \int f_1(u)\, f_2(x-u)\, du$$

  - Operation on f_1 and f_2 called *convolution*.
  - Convolution of f_1 and f_2 denoted by $f_1 * f_2$.
  - Method *samples* from density $f_1 * f_2 * \cdots * f_m$.

Example: Generate Erlang(λ,m) Using Convolution Method

- **Problem**: Generate random variate with Erlang(λ, m) distn.
  - Sometimes Erlang distn is specified with parameters (β, m), where scale parameter $\beta = 1/\lambda$.
- Density $f_{\lambda,m}$ of Erlang(λ, m) distn is

$$f_{\lambda,m}(x) = \frac{\lambda(\lambda x)^{m-1}}{(m-1)!} e^{-\lambda x}, \quad x \geq 0$$

- **Fact :** If $Z_i \sim \exp(\lambda)$, $i = 1, \ldots, m$, and independent, then

$$X = Z_1 + Z_2 + \cdots + Z_m \text{ is Erlang}(\lambda, m)$$

- All set for convolution method ...

- Algorithm to generate $X \sim \text{Erlang}(\lambda, m)$ using convolution:

> 1. Generate $U_1, U_2, \ldots, U_m \sim \text{unif}[0,1]$, indep
> 2. Set $Z_i = -\frac{1}{\lambda}\ln(U_i), \;\; i = 1, 2, \ldots, m$
> 3. Set $X = \sum_{i=1}^{m} Z_i$

- In Excel for $m = 2$, can use formula

```
=-LN(RAND())/LAMBDA-LN(RAND())/LAMBDA
```

  where `LAMBDA` is a user-defined constant.

- Not very efficient when m is large ...
  requires m random numbers to generate 1 random variate
- More efficient methods exist

Composition method

- **Goal:** Generate observations of RV X with CDF F
- Suppose CDF F can be written in the form:

$$F(x) = \sum_{i=1}^{m} p_i \, F_i(x)$$

  where
  - $p_i \geq 0, \;\; \sum_{i=1}^{m} p_i = 1,$ i.e., $(p_1, \ldots, p_m)$ is PMF
  - each F_i is CDF from which we can generate random variate
- If each F_i has density f_i, then density f of X is

$$f(x) = \sum_{i=1}^{m} p_i \, f_i(x)$$

- Define a new discrete random variable I as follows:

$$\mathcal{P}(I = j) \;=\; p_j, \quad 1 \leq j \leq m$$

- Let Y_j be random variable with CDF F_j
- Define a new random variable Z as follows :

$$Z = Y_j \quad \text{if } I = j$$

- CDF G of Z

$$\begin{aligned} G(x) \;&=\; \mathcal{P}(Z \leq x) \;=\; \sum_{j=1}^{m} \mathcal{P}(\, Z \leq x, \, I = j \,) \\ &= \sum_{j=1}^{m} \mathcal{P}(\, Z \leq x \mid I = j \,) \, \mathcal{P}(I = j) \\ &= \sum_{j=1}^{m} \mathcal{P}(Y_j \leq x) \, p_j \;=\; \sum_{j=1}^{m} F_j(x) \, p_j \;=\; F(x) \end{aligned}$$

- **Random variables Z and X have same distn ...**
- Thus, an algorithm for generating random variate Z will also result in (correct) random variate X ...
- Generic algorithm to generate $X \sim F$ using composition when $F(x) = \sum_{i=1}^{m} p_i \, F_i(x)$:

> 1. Generate I with PMF $(p_1, \ldots, p_m)$
>   (e.g., using inverse-transform)
> 2. If $I = j$, generate $Y_j \sim F_j$, indep of I
>   (inverse-transform or convolution or ...)
> 3. Set $X = Y_j$

$$F(x) = \sum_{i=1}^{m} p_i F_i(x)$$

- Known as **composition method**
- Main problems:
  1. How to "split" the CDF into pieces?
  2. How to decide on the correct pieces?
- There are methods for the first problem.
- But no definitive answer for the second.
  Will return to it after an example.

Example : Hyperexp($\lambda_1, \alpha_1, \lambda_2, \alpha_2$) Using Composition

- **Goal:** generate $X \sim$ Hyperexp$(\lambda_1, \alpha_1, \lambda_2, \alpha_2)$ distn.
- Density f is

$$f(x) = \begin{cases} \alpha_1 \left(\lambda_1 e^{-\lambda_1 x}\right) + \alpha_2 \left(\lambda_2 e^{-\lambda_2 x}\right), & \text{if } x \geq 0, \\ 0, & \text{otherwise,} \end{cases}$$

  where given $\lambda_1, \lambda_2 > 0$ and $\alpha_1, \alpha_2 \geq 0$, $\alpha_1 + \alpha_2 = 1$.
- Density f is already in the form we need (why?)
- Random variable I with

$$I = \begin{cases} 1, \text{ with prob. } \alpha_1, \\ 2, \text{ with prob. } \alpha_2 = 1 - \alpha_1 \end{cases}$$

- Random variables $Y_j \sim \exp(\lambda_j)$, $j = 1, 2$

- Composition method to generate
  $X \sim$ Hyperexp$(\lambda_1, \alpha_1, \lambda_2, \alpha_2)$

> 1. Generate I as follows:
> (a) Generate $U \sim$ unif$[0, 1]$.
> (b) If $U < \alpha_1$, set $I \leftarrow 1$;
> Else set $I \leftarrow 2$.
> 2. Generate $X = Y_I$ as follows:
> (a) Generate $U' \sim$ unif$[0, 1]$, indep of U.
> (b) If $I = 1$, set $X \leftarrow Y_1 = -\frac{1}{\lambda_1} \ln(U')$;
> Else set $X \leftarrow Y_2 = -\frac{1}{\lambda_2} \ln(U')$.

- In Excel, can use formula
  `=IF(RAND()<ALPHA1,-LN(RAND())/LAMBDA1,-LN(RAND())/LAMBDA2)`
  where `ALPHA1`, `LAMBDA1`, `LAMBDA2` are user-defined constants.

Composition Method: Splitting

Problem: often only given original density f, and we need to figure out how to express it as mixture of other densities f_i:

$$f(x) = \sum_{i=1}^{m} p_i f_i(x)$$

One approach: *vertical split* — partition x-axis into intervals

- **Step 1:** define $m - 1$ break points

$$-\infty = a_0 < a_1 < a_2 < \cdots < a_{m-1} < a_m = \infty$$

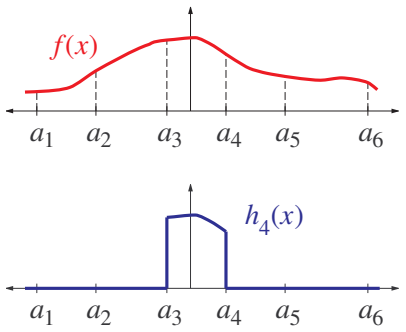
  and partition x-axis into intervals using break points:

  $R_1 = (a_0, a_1], \quad R_2 = (a_1, a_2], \quad \ldots, \quad R_m = (a_{m-1}, a_m)$

  - Useful to define **indicator function** for set $A \subseteq \mathbf{R}$

$$I_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \notin A \end{cases}$$

- **Step 2:** define function $h_i(x) = f(x)\, I_{(a_{i-1},a_i]}(x)$



  - Note that $f(x) = h_1(x) + h_2(x) + \cdots + h_m(x)$

- **Step 3:** compute

$$p_i \equiv \int_{-\infty}^{\infty} h_i(x)\, dx = \int_{a_{i-1}}^{a_i} f(x)\, dx$$

so $0 \le p_i \le 1$ and $p_1 + p_2 + \cdots + p_m = 1$

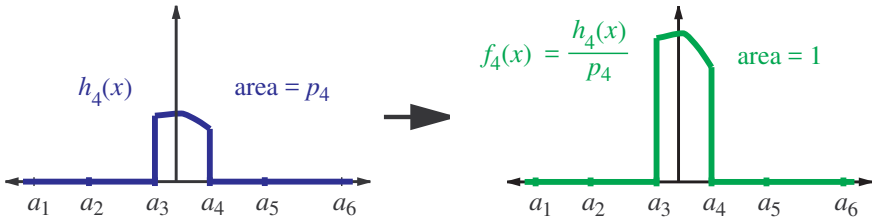
- **Step 4:** write

$$\begin{aligned} f(x) &= h_1(x) + h_2(x) + \cdots + h_m(x) \\ &= p_1\, f_1(x) + p_2\, f_2(x) + \cdots + p_m\, f_m(x), \end{aligned}$$

where

$$f_i(x) = \frac{h_i(x)}{p_i}$$

  - f_i is density: nonnegative and integrates to 1
  - f_i has same shape as f over R_i but elongated since $p_i \le 1$

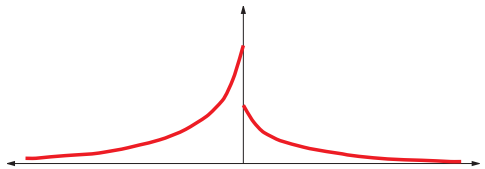


Example : Composition with Vertical Split

- **Goal:** generate random variate having density

$$f(x) = \begin{cases} 0.75\, \lambda_1\, e^{\lambda_1 x}, & \text{if } x \le 0, \\ 0.25\, \lambda_2\, e^{-\lambda_2 x}, & \text{if } x > 0 \end{cases}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are given constants



- **Step 1:** Partition x-axis at origin:

$$R_1 = (-\infty, 0], \qquad R_2 = (0, \infty)$$

- **Step 2:** Split the density f :

$$h_1(x) = 0.75\, \lambda_1\, e^{\lambda_1 x}\, I_{(-\infty,0]}(x),$$

$$h_2(x) = 0.25\, \lambda_2\, e^{-\lambda_2 x}\, I_{(0,\infty)}(x).$$

- **Step 3:** Calculate p_i's to normalize h_i's:

$$p_1 = \int_{-\infty}^{0} f(x)\, dx = \int_{-\infty}^{0} 0.75\, \lambda_1\, e^{\lambda_1 x}\, dx = 0.75,$$

$$p_2 = \int_{0}^{\infty} f(x)\, dx = \int_{0}^{\infty} 0.25\, \lambda_2\, e^{-\lambda_2 x}\, dx = 0.25$$

- **Step 4:** Write $f(x) = p_1\, f_1(x) + p_2\, f_2(x)$, where

$$f_1(x) = \frac{h_1(x)}{p_1} = \lambda_1\, e^{\lambda_1 x}\, I_{(-\infty,0]}(x),$$

$$f_2(x) = \frac{h_2(x)}{p_2} = \lambda_2\, e^{-\lambda_2 x}\, I_{(0,\infty)}(x)$$

- Composition algorithm to generate X with density

$$f(x) \;=\; 0.75\,\lambda_1\,e^{\lambda_1 x}\,I_{(-\infty,0]}(x) + 0.25\,\lambda_2\,e^{-\lambda_2 x}\,I_{(0,\infty)}(x)$$

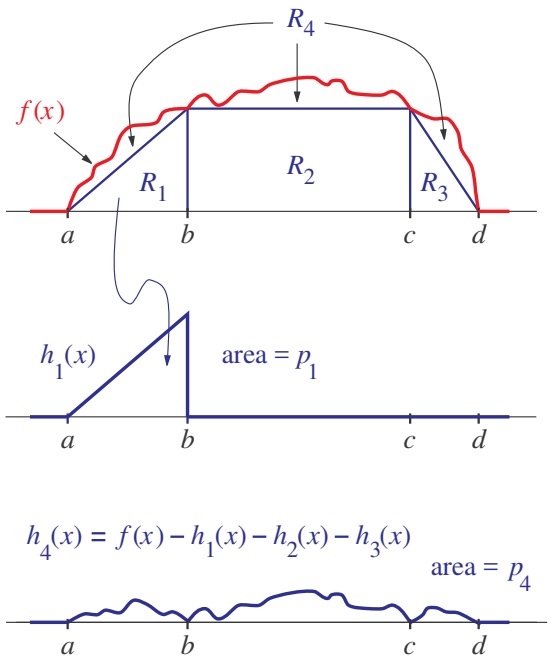
1. Generate I as follows:
   (a) Generate $U \sim$ unif[0, 1].
   (b) If $U < p_1 = 0.75$, set $I \leftarrow 1$;
   Else set $I \leftarrow 2$.
2. Generate $X = Y_I$ as follows:
   (a) Generate $U' \sim$ unif[0, 1], indep of U.
   (b) If $I = 1$, set $X \leftarrow Y_1 = \frac{1}{\lambda_1}\ln(U')$;
   Else set $X \leftarrow Y_2 = -\frac{1}{\lambda_2}\ln(U')$.

- In Excel, can use formula
  `=IF(RAND()<0.75,LN(RAND())/LAMBDA1,-LN(RAND())/LAMBDA2)`
  where `LAMBDA1`, `LAMBDA2` are user-defined constants.

Composition Method: General Splitting

- Can apply splitting in other ways
- **Idea:** partition area under density f into regions R_i, $i = 1, \ldots, m$.
  - Let p_i be area of region R_i
  - $p_1 + p_2 + \cdots + p_m = 1$ since f integrates to 1
  - Define function h_i based on region R_i
  - Normalize h_i by p_i to obtain density f_i
  - Then express

$$f(x) \;=\; p_1 f_1(x) \;+\; p_2 f_2(x) \;+\; \cdots \;+\; p_m f_m(x)$$



Composition Method: General Splitting

- Note that

$$f(x) \;=\; h_1(x) \;+\; h_2(x) \;+\; h_3(x) \;+\; h_4(x)$$

- Each $h_i(x)$ is not a density since

$$p_i \;\equiv\; \int h_i(x)\,dx \;<\; 1$$

  and $p_1 + p_2 + p_3 + p_4 \;=\; 1$

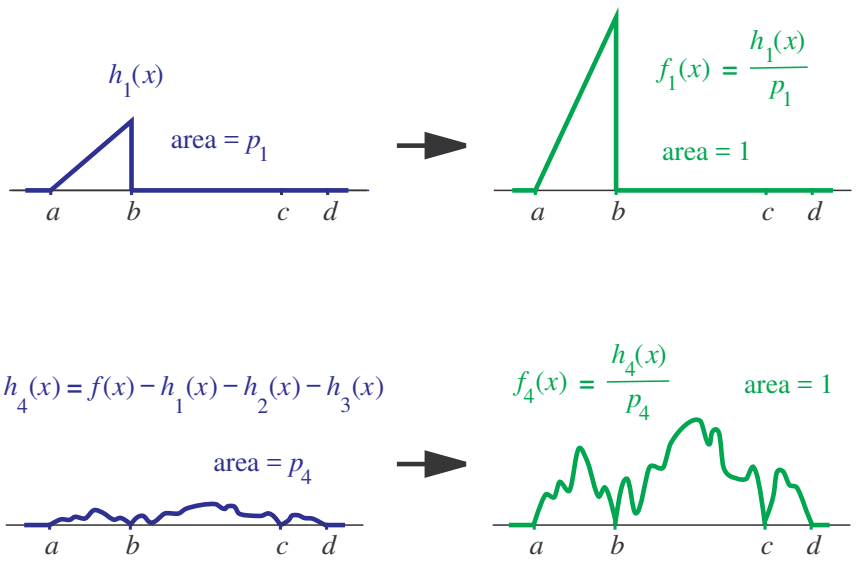
- Thus, normalize h_i to obtain a density f_i:

$$f_i(x) \;=\; \frac{h_i(x)}{p_i}$$

  - f_i has same basic shape as h_i but elongated since $p_i < 1$

- Since $h_i(x) = p_i f_i(x)$, we get

$$f(x) \;=\; p_1\,f_1(x) \;+\; p_2\,f_2(x) \;+\; p_3\,f_3(x) \;+\; p_4\,f_4(x)$$



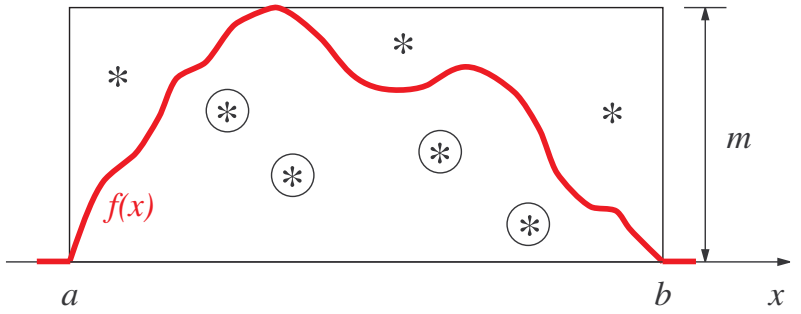
Composition Method: General Splitting

- Choose partition $R_1, \ldots, R_m$ so area under density f is mostly contained in R_i's that are right triangles or rectangles
- Sometimes, most efficient method to generate random variates from a distribution uses this type of composition.
- Can quickly generate observations from densities that are right triangles or rectangles (uniform) densities.
- In our example
  - Can show that if Z_1, Z_2 are indep unif$[a, b]$, then $Y_1 = \max(Z_1, Z_2)$ has density f_1, which is given by
$$f_1(x) = \frac{2(x-a)}{(b-a)^2} I_{[a,b]}(x).$$
  - When p_4 is small, we only rarely generate from density f_4

Acceptance-Rejection Method : Continuous

- Method well-suited to generate random variate when its density function f is easy to calculate.
- Suppose X has density f, which is bounded.
- For now, suppose that support of X is finite interval $[a, b]$.
  - $f(x) > 0$ only when $a \le x \le b$.
- Enclose density in rectangle R with
  - Base having length $(b - a)$
  - Height m, where $m = \max_{a \le x \le b} f(x)$

- Throw down points uniformly in rectangle R.
  - Denote these points by $*$
  - Throw away (or **reject**) any points above the density f
  - **Accept** points below the density f.
    Denote these points by $\circledast$



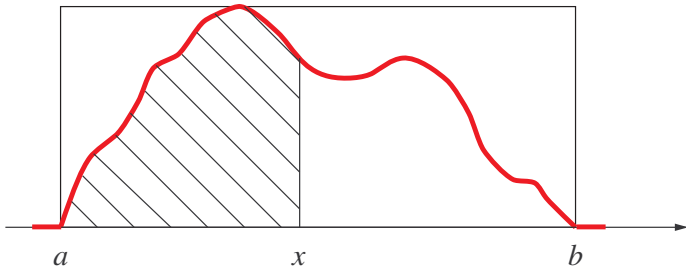
Claim: x-coordinate X of accepted point has density f.

Proof:

- Need to show $P(X \le x) = \int_a^x f(t)\,dt$ for each $x \in [a, b]$.
- Let (Z_1, Z_2) be (x, y)-coordinate of a random point distributed uniformly in rectangle R
- Accept $(Z_1, Z_2) \iff Z_2 \le f(Z_1)$.
- Then, for $a \le x \le b$,

$$\begin{aligned}\mathcal{P}(X \le x) &= \mathcal{P}(Z_1 \le x \mid \text{accept } (Z_1, Z_2)) \\ &= \frac{\mathcal{P}(Z_1 \le x,\ \text{accept } (Z_1, Z_2))}{\mathcal{P}(\text{accept } (Z_1, Z_2))} \\ &= \frac{\mathcal{P}(Z_1 \le x,\ Z_2 \le f(Z_1))}{\mathcal{P}(Z_2 \le f(Z_1))} \qquad (1)\end{aligned}$$

- Numerator of Eq. (1):
  - $\mathcal{P}(Z_1 \le x,\ Z_2 \le f(Z_1))$ is just the probability that (Z_1, Z_2) falls into shaded region below:



  - Because (Z_1, Z_2) has uniform distn over R,

$$\mathcal{P}(Z_1 \le x,\ Z_2 \le f(Z_1)) = \frac{\int_a^x f(t)\,dt}{\text{area of } R} \qquad (2)$$

- Denominator of Eq. (1) is $\mathcal{P}(\text{ accept }) = \mathcal{P}(\ Z_2 \le f(Z_1)\ )$
  - Since $\mathcal{P}(Z_1 \le b) = 1$,

$$\mathcal{P}(\ Z_2 \le f(Z_1)\ ) = \mathcal{P}(\ Z_1 \le b,\ Z_2 \le f(Z_1)\ )$$

  - Putting $x = b$ in Eq. (2), we see that acceptance prob is

$$\mathcal{P}(\ Z_2 \le f(Z_1)\ ) = \frac{\int_a^b f(t)\,dt}{\text{area of } R} = \frac{1}{\text{area of } R}$$

  since $\int_a^b f(t)\,dt = 1$.

- Thus, from Eq. (1),

$$\mathcal{P}(X \le x) = \frac{\mathcal{P}(Z_1 \le x,\ Z_2 \le f(Z_1))}{\mathcal{P}(Z_2 \le f(Z_1))} = \int_a^x f(t)\,dt$$

as desired.

Generic Algorithm to Generate X Using Acceptance-Rejection Method with Uniform Majorizing Function when support is finite interval $[a, b]$

1. Generate $U_1, U_2 \sim$ unif[0, 1], independent.
2. Set $Z_1 = a + (b - a)U_1$;
   $Z_2 = mU_2$.
3. If $Z_2 \le f(Z_1)$, set $X \leftarrow Z_1$ and stop;
   Else, discard (Z_1, Z_2) and return to Step 1.

Not every pair (U_1, U_2) leads to a random variate X.

Remarks on Acceptance-Rejection Method:

- As shown on slide 6-51, each iteration has

$$\mathcal{P}(\text{ accept }) \;=\; \mathcal{P}(\, Z_2 \le f(Z_1)\,) \;=\; (\text{ area of } R\,)^{-1} = \frac{1}{(b-a)m}$$

- Number N of pairs (U_1, U_2) of uniforms required to generate X is geometric$(1/c)$ RV:

$$\mathcal{P}(N = k) \;=\; \frac{1}{c}\left(1 - \frac{1}{c}\right)^{k-1}, \quad k \ge 1,$$

where c = area of R.

- Mean number of pairs required is

$$\mathcal{E}[N] \;=\; c \;=\; (b-a)m \;\ge\; 1$$

- Measure of efficiency of acceptance-rejection method is closeness of $(b-a)m$ to 1.

Example : beta(4,3) distn using acceptance-rejection

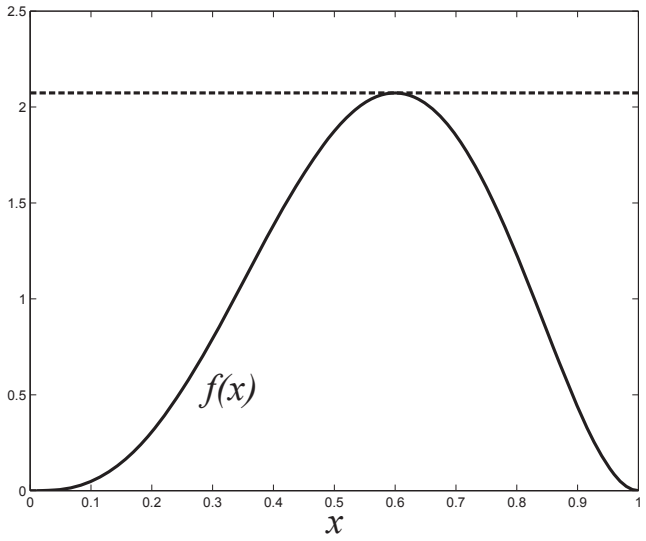
- beta(4,3) has density

$$f(x) \;=\; 60\, x^3 (1-x)^2\, I_{[0,1]}(x)$$

- Difficult to use inversion. Why?

Acceptance-rejection

- **Step 1:** Set interval $[a, b] = [0, 1]$.
- **Step 2:** Calculate $m \;=\; \max\{f(x) : x \in [0, 1]\}$.
  - Using calculus can show
    - ▷ Maximum occurs at $x = 0.6$
    - ▷ $m = 2.0736$



- Algorithm 1 to generate $X \sim$ beta(4,3) using acceptance-rejection:

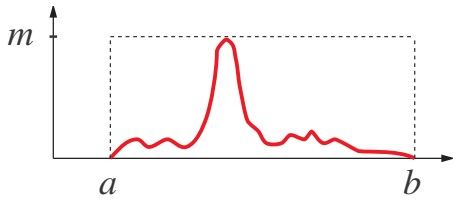
1. Generate $U_1, U_2 \sim$ unif[0, 1], independent.

2. Set $Z_1 \;=\; U_1$;
   $Z_2 \;=\; 2.0736\, U_2$

3. If $Z_2 \;<\; f(Z_1) \;=\; 60\, Z_1^3 (1 - Z_1)^2$, set $X \leftarrow Z_1$ and stop;
   Else, discard (Z_1, Z_2) and return to Step 1.

(Generalized) Acceptance-Rejection

- Previous A-R method not always applicable or efficient:
  - Continuous random variables with infinite support
  - $(b-a)m$ very large

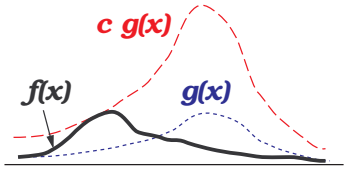


- Want more general acceptance-rejection method.

- **Goal:** Generate random variate having density f
- **Assumption:** $\exists$ RV Y with density g such that
  - Can generate observations of Y
  - $\exists$ constant $c > 0$ such that $f(x) \le c\,g(x) \;\; \forall\, x$

- Choose smallest such constant

$$c \;=\; \max\left\{\frac{f(x)}{g(x)} : x \in \mathbf{R}\right\}$$

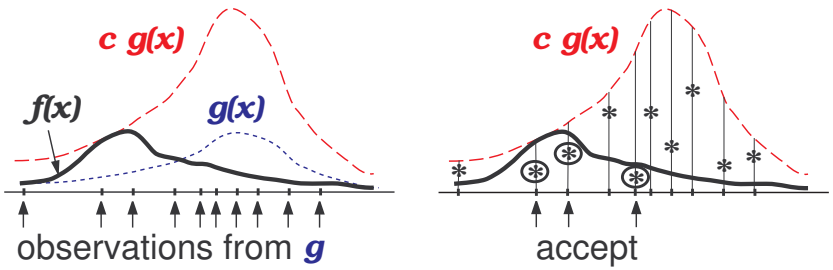


- $g(x)$ is known as *majorizing density*.
- $h(x) = c\,g(x)$ is known as *majorizing function*.
- Given Y, if $U \sim$ unif[0, 1] and independent of Y, then

$$c\,g(Y)\,U \;\sim\; \text{unif}[0,\, c\,g(Y)].$$

- Generic algorithm to generate $X \sim f$ using acceptance-rejection with majorizing density g :

> 1. Generate $Y \sim g$.
> 2. Generate $U \sim$ unif[0, 1], indep of Y.
> 3. If $c\,g(Y)\,U \;<\; f(Y)$, set $X \leftarrow Y$ and stop; Else, discard (Y, U) and return to Step 1.



Proof: Analogous to case with uniform majorizing density

- Need to show Y in accepted (Y, U) has CDF F with density f.
- Define event $A = \{\text{ pair } (Y, U) \text{ accepted }\}$
- Accept: $\quad c\,g(Y)\,U < f(Y) \quad\Longleftrightarrow\quad U < f(Y)/[c\,g(Y)]$
- Law of total probability yields

$$\begin{aligned}
\mathcal{P}(A) &= \int_{-\infty}^{\infty} \mathcal{P}(A \mid Y \in (y, y+dy))\; g(y)\,dy \\
&= \int_{-\infty}^{\infty} \mathcal{P}\left(U < \frac{f(y)}{c\,g(y)}\right) g(y)\,dy \\
&= \int_{-\infty}^{\infty} \frac{f(y)}{c\,g(y)}\; g(y)\,dy \\
&= \int_{-\infty}^{\infty} \frac{f(y)}{c}\,dy \;=\; \frac{1}{c},
\end{aligned}$$

where third steps holds because $P(U < z) = z$ for $z \in [0, 1]$

• Similarly

$$\begin{aligned}
&\mathcal{P}(\{Y \le x\} \cap A) \\
&= \int_{-\infty}^{x} \mathcal{P}(A \mid Y \in (y, y+dy))\; g(y)\, dy \\
&= \int_{-\infty}^{x} \mathcal{P}\left(U \le \frac{f(y)}{c\, g(y)}\right)\; g(y)\, dy \\
&= \int_{-\infty}^{x} \frac{f(y)}{c\, g(y)}\, g(y)\, dy \\
&= \int_{-\infty}^{x} \frac{f(y)}{c}\, dy \;=\; \frac{F(x)}{c}
\end{aligned}$$

• So ...

$$\begin{aligned}
\mathcal{P}(X \le x) &= \mathcal{P}(Y \le x \mid A) \\
&= \frac{\mathcal{P}(\{Y \le x\} \cap A)}{\mathcal{P}(A)} \\
&= \frac{F(x)/c}{1/c} \\
&= F(x)
\end{aligned}$$

QED

Remarks on Acceptance-Rejection Method

• As shown on slide 6-60, each iteration has

$$\mathcal{P}(\text{accept}) \;=\; \frac{1}{c}$$

where $c \;=\; \max\left\{\frac{f(x)}{g(x)} : x \in \mathbf{R}\right\}$

• Number N of iterations required to generate X is geometric RV:

$$\mathcal{P}(N = k) \;=\; \frac{1}{c}\left(1 - \frac{1}{c}\right)^{k-1}, \quad k \ge 1.$$

• Mean number of pairs required is

$$\mathcal{E}[N] \;=\; c \;\ge\; 1$$

• Want c to be close to 1 for algorithm to be efficient.

Example : beta(4,3) distn using acceptance-rejection with non-uniform majorizing function

• beta(4,3) has density

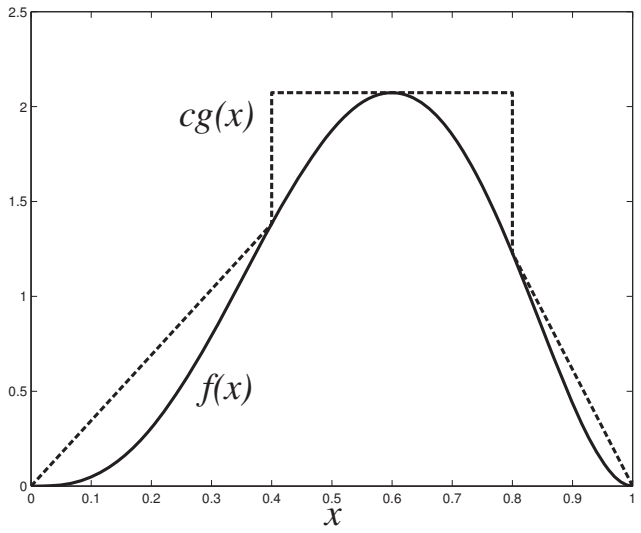
$$f(x) \;=\; 60\, x^3 (1-x)^2\, I_{[0,1]}(x)$$

• Step 1: Specify majorizing density g:

$$g(x) \;=\; \begin{cases} 45x/16 & \text{if } 0 \le x \le 0.4 \\ 27/16 & \text{if } 0.4 < x \le 0.8 \\ 5 - 5x & \text{if } 0.8 < x \le 1 \end{cases}$$

• Step 2: Compute

$$c \;=\; \max\left\{\frac{f(x)}{g(x)} : x \in [0,1]\right\} \;=\; 1.2288$$



- Algorithm 2 to generate $X \sim$ beta(4,3) using acceptance-rejection with non-uniform majorizing density g:

1. Generate $Y \sim g$, e.g., using inversion as on slide 6-13.
2. Generate $U \sim$ unif[0, 1], indep of Y.
3. If $1.2288\, g(Y)\, U < f(Y)$, set $X \leftarrow Y$ and stop; Else, discard (Y, U) and return to Step 1.

More Remarks on Acceptance-Rejection Method

- Tradeoffs with acceptance-rejection...
  - Ease of generation from majorizing density $g(x)$
  - Constant $c = \max \frac{f(x)}{g(x)} = \mathcal{E}[\text{number of iterations}]$
- Compare two previous methods for beta(4,3)
  - **Method 1**:
$$g(x) \;=\; I_{[0,1]}(x)$$
  - **Method 2**:
$$g(x) \;=\; \begin{cases} 45x/16 & \text{if } 0 \le x \le 0.4 \\ 27/16 & \text{if } 0.4 < x \le 0.8 \\ 5 - 5x & \text{if } 0.8 < x \le 1 \end{cases}$$

Method 1:

- Majorizing density:
$$g(x) \;=\; I_{[0,1]}(x)$$
- $c = 2.0736$
- Complexity of generation method :
  1. Expected number of iterations $\mathcal{E}[N] = c = 2.0736$
  2. Number of uniform random numbers per iteration $R = 2$,

$$\text{Total complexity} \;=\; R \cdot \mathcal{E}[N] \;=\; 4.1472$$

Method 2:

- Majorizing density:

$$g(x) = \begin{cases} 45x/16 & \text{if } 0 \le x \le 0.4 \\ 27/16 & \text{if } 0.4 < x \le 0.8 \\ 5 - 5x & \text{if } 0.8 < x \le 1 \end{cases}$$

- $c = 1.2288$
- Complexity of generation method :
  1. Expected number of iterations $\mathcal{E}[N] = c = 1.2288$
  2. Number of uniform random numbers per iteration $R = 2$
  3. Cost of inversion alg (relative to generating one unif) $= C_I$

  Total complexity $= (R+C_I)\cdot\mathcal{E}[N] = 2.4576+1.2288C_I$

Even More Remarks on Acceptance-Rejection Method

- Acceptance-rejection method only requires computing density function f, not CDF F.
  - Inversion requires computing inverse F^{-1}
- Sometimes CDF cannot be calculated in closed form, e.g., normal distn
  - CDF Φ and inverse CDF Φ^{-1} of $\mathcal{N}(0,1)$ have polynomial approximations
  - In Excel, can use `normsdist` or `normsinv` function.
- For acceptance-rejection to generate $X \sim f$ with infinite support, choose majorizing density g with no lighter tails than f.

Example : Truncation

- Suppose can generate observation of RV Y with density g
- Want observation of $X = Y$ given $Y \in A$
  - X is random variable Y truncated to A.
- Thus, density $f(x)$ of X is given by

$$f(x) = \frac{g(x)}{\mathcal{P}(Y \in A)} I_A(x)$$

- Will use acceptance/rejection technique with g as majorizing density

- Constant c is

$$c = \max_x \frac{f(x)}{g(x)} = \max_x \frac{g(x)\, I_A(x)/\mathcal{P}(Y \in A)}{g(x)} = \frac{1}{\mathcal{P}(Y \in A)}$$

- Algorithm for truncation with continuous distn:

> 1. Generate observation of $Y \sim g$.
> 2. Generate random number $U \sim$ unif[0, 1], indep of Y.
> 3. If $U < \frac{f(Y)}{cg(Y)}$, set $X \leftarrow Y$ and stop; Else, discard (Y, U) and return to Step 1.

- Can simplify this algorithm since

$$f(x) = \frac{g(x)\, I_A(x)}{\mathcal{P}(Y \in A)}, \qquad c = \frac{1}{\mathcal{P}(Y \in A)}, \qquad \frac{f(x)}{c\, g(x)} = I_A(x).$$

  - If $Y \in A$, then $\frac{f(Y)}{c\, g(Y)} = I_A(Y) = 1$, i.e., Y is *always* accepted in Step 3.
  - If $Y \notin A$, then $\frac{f(Y)}{c\, g(Y)} = I_A(Y) = 0$, i.e., Y is never accepted in Step 3.

- Simplified algorithm for truncation:

> 1. Generate observation of $Y \sim g$.
> 2. If $Y \in A$, set $X \leftarrow Y$ and stop;
>   Else, discard (Y, U) and return to Step 1.

- Hard part here is recognizing that X has "truncated" distn ...
  - The giveaway pattern that one should be looking for is that the density $f(x)$ of X has the form

$$f(x) = \frac{1}{c} \cdot g(x)$$

    for some "nice" density and constant c
  - Example:

$$f(x) = \frac{e^{-x}}{1 - e^{-1}}\, I_{[0,1]}(x)$$

Acceptance-Rejection Method : Discrete

- Can also apply acceptance-rejection to generate discrete RV
- Suppose can generate observations of $Y \sim \{q_j : j = 1, 2, \ldots\}$
- Want to use this to generate an observation of $X \sim \{p_j : j = 1, 2, \ldots\}$
- Define constant c:

$$c = \max_j \frac{p_j}{q_j}$$

- Generic algorithm to generate discrete RV X using acceptance-rejection method :

> 1. Generate $Y \sim \{q_j : j \geq 1\}$.
> 2. Generate random number $U \sim$ unif[0, 1], indep of Y.
> 3. If $U < \frac{p_Y}{c\, q_Y}$, set $X \leftarrow Y$ and stop;
>   Else, discard (Y, U) and return to Step 1.

Remarks :

- Each iteration is *independently* accepted with probability $\frac{1}{c}$
- Thus, number of iterations needed is *geometric* with mean c
- So the closer c is to 1 the better.

Example : simple PMF using acceptance-rejection

- Want to generate observations of X with PMF

$$p_j = \{0.40,\ 0.25,\ 0.2,\ 0.15\}$$

- Know how to generate Y with PMF

$$q_j = \{0.25,\ 0.25,\ 0.25,\ 0.25\}$$

- Define c as follows:

$$c = \max\left\{\frac{p_j}{q_j} : j = 1,2,3,4\right\}$$
$$= \frac{0.4}{0.25} = 1.6$$

- Algorithm to generate $X \sim$ PMF p_j using acceptance-rejection:

1. Generate random number $U_1 \sim$ unif[0, 1] and set $Y \leftarrow \lceil 4U_1 \rceil$.
2. Generate another random number $U_2 \sim$ unif[0, 1], indep of Y.
3. If $U_2 < p_Y/0.4$, set $X \leftarrow Y$ and stop; Else, discard (U_1, U_2) and return to Step 1.

Example : truncation

- Suppose know how to generate $Y \sim \{q_j : j \geq 1\}$
- Let A be some set of indices $\{j_1, j_2, \ldots, j_n\}$
- Want to generate observations of $X = Y$ given $Y \in A$, i.e., X is random variable restricted or "truncated" to set A.
- PMF $p = \{p_j : j \in A\}$ of X:

$$p_j = \mathcal{P}(\{Y = j\} \mid \{Y \in A\})$$
$$= \begin{cases} \frac{\mathcal{P}(Y=j)}{\mathcal{P}(A)} = \frac{q_j}{\sum_{k\in A} q_k} & \text{if } j \in A, \\ 0 & \text{otherwise} \end{cases}$$

- Define c

$$c = \max\left\{\frac{p_j}{q_j} : j \geq 1\right\} = \frac{1}{\sum_{k\in A} q_k}$$

- Algorithm to generate X using truncation:

1. Generate $Y \sim \{q_j : j \geq 1\}$.
2. Generate random number $U \sim$ unif[0, 1], independent of Y.
3. If $U < \frac{p_Y}{c\, q_Y}$, set $X \leftarrow Y$ and stop; Else, discard (Y, U) and return to Step 1.

$$p_j = \begin{cases} \frac{\mathcal{P}(Y=j)}{\mathcal{P}(A)} = \frac{q_j}{\sum_{k\in A} q_k} & \text{if } j \in A, \\ 0 & \text{otherwise} \end{cases}, \qquad c = \frac{1}{\sum_{k\in A} q_k}$$

- Analyze Step 3 a little more closely ...

If $j \in A$ then

$$U < \frac{p_j}{c\, q_j} \iff U < 1 \iff \text{always accept}$$

If $j \notin A$ then

$$U < \frac{p_j}{c\, q_j} \iff U < 0 \iff \text{never accept}$$

- Simplified algorithm for truncation:

> 1. Generate $Y \sim \{q_j : j \geq 1\}$.
>
> 2. If $Y \in A$, set $X \leftarrow Y$ and stop;
> Else, discard Y and return to Step 1.

- Expected number of iterations $c = \frac{1}{q(A)}$

Generating $\mathcal{N}(0,1)$ variates

Facts :

- If $Z \sim \mathcal{N}(0,1)$, then $Y = \sigma Z + \mu \sim \mathcal{N}(\mu, \sigma^2)$.
  - Therefore will focus on $\mathcal{N}(0,1)$ random variates only.
- CDF Φ of $\mathcal{N}(0,1)$ not available in closed form
  - $\exists$ polynomial approximations to Φ and Φ^{-1} (in Excel, `normsdist` and `normsinv`), so could use approx inversion.
    - ▷ In Excel, can generate standard normal using `=normsinv(rand())`

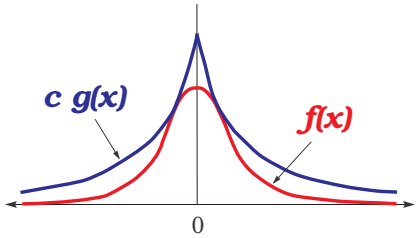
Algorithm N1: Acceptance-Rejection technique

- Want to generate $X \sim f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$
- Slight modification of algorithm on slide 6-41 can generate

$$Y \sim g(x) = \frac{1}{2} e^x I_{(-\infty,0)}(x) + \frac{1}{2} e^{-x} I_{[0,\infty)}(x)$$

- Calculate constant c using calculus:

$$c = \max\left\{ \frac{f(x)}{g(x)} : x \in \mathbf{R} \right\} = \sqrt{\frac{2e}{\pi}} \approx 1.3$$



- Algorithm for generating normal using acceptance-rejection :

> 1. Generate random variate $Y \sim g(x)$ [e.g., using composition, slide 6-41]
> 2. Generate $U \sim$ unif[0, 1], indep of Y
> 3. If $c\,g(Y)\,U \;\leq\; f(Y)$, where $c = \sqrt{2e/\pi}$, then set $X \leftarrow Y$ and stop; Else, discard (Y, U) and return to step 1

- Expected number of uniforms per observation :

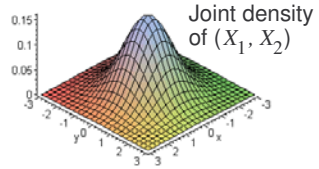
$$3c \;=\; 3\sqrt{\frac{2e}{\pi}} \;\approx\; 4$$

Algorithm N2 : Box-Muller Method to generate normals

> 1. Generate two *independent* uniform random numbers $U_1, U_2 \sim$ unif[0, 1].
> 2. Set $X_1 \leftarrow \sqrt{-2\ln(U_1)}\,\cos(2\pi U_2)$.
> 3. Set $X_2 \leftarrow \sqrt{-2\ln(U_1)}\,\sin(2\pi U_2)$.
> 4. Return X_1 and X_2 as IID $\mathcal{N}(0, 1)$.

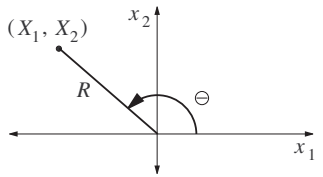
Claim : X_1 and X_2 are independent $\mathcal{N}(0, 1)$ random variables.

Intuition for Box-Muller Method



- Suppose $X_1, X_2 \sim \mathcal{N}(0, 1)$ IID.
- Plot (X_1, X_2) in 2-D space, and use polar coordinates (R, Θ):

$$X_1 \;=\; R\cos(\Theta)$$
$$X_2 \;=\; R\sin(\Theta)$$



- Can show radius R and angle Θ indep.
- Note that $R^2 = X_1^2 + X_2^2$ has chi-square distn with 2 d.f., which is same as exponential distn with mean 2:

$$R \;=\; \sqrt{-2\ln(U_1)}$$

- By symmetry of (X_1, X_2) density, can show $\Theta \sim$ unif$[0, 2\pi]$:

$$\Theta \;=\; 2\pi U_2$$

Proof of Box-Muller Method

Claim : X_1 and X_2 are independent $\mathcal{N}(0, 1)$ random variables.

Proof :

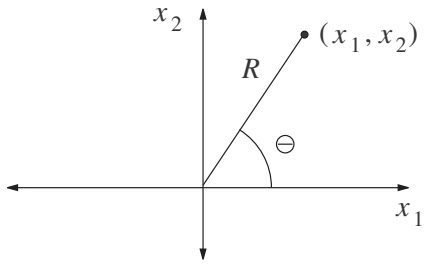
- Have to prove that

$$\mathcal{P}(X_1 \leq x_1,\, X_2 \leq x_2)$$
$$= \int_{-\infty}^{x_1} \frac{1}{\sqrt{2\pi}} e^{-u^2/2}\,du \;\cdot\; \int_{-\infty}^{x_2} \frac{1}{\sqrt{2\pi}} e^{-v^2/2}\,dv$$

- *Polar coordinates*: (r, θ), $r \geq 0$, $\theta \in [0, 2\pi)$.
- Point in 2-D $\mathbf{x} = (x_1, x_2)$ can be represented in polar coordinates (R, Θ) satisfying following relations:

$$\begin{aligned} x_1 &= R\cos(\Theta) \\ x_2 &= R\sin(\Theta) \end{aligned}$$

$$\begin{aligned} R &= \sqrt{x_1^2 + x_2^2} \\ \Theta &= \tan^{-1}\left(\frac{x_2}{x_1}\right) \end{aligned}$$



- In Box-Muller algorithm, U_1 and U_2 IID unif[0, 1]
  - $X_1 = \sqrt{-2\ln(U_1)}\cos(2\pi U_2)$
  - $X_2 = \sqrt{-2\ln(U_1)}\sin(2\pi U_2)$
- Express (X_1, X_2) in polar coordinates:

$$\begin{aligned} R &= \sqrt{X_1^2 + X_2^2} = \sqrt{-2\ln(U_1)}, \\ \Theta &= \tan^{-1}\left(\frac{X_2}{X_1}\right) = 2\pi U_2 \end{aligned}$$

- Compute CDF of random variable Θ: for $\theta \in [0, 2\pi)$,

$$\begin{aligned} F_\Theta(\theta) &= \mathcal{P}(\Theta \leq \theta) \\ &= \mathcal{P}(2\pi U_2 \leq \theta) \\ &= \theta/(2\pi), \end{aligned}$$

so density $f_\Theta(\theta) = 1/(2\pi)$ and $\Theta \sim$ unif$[0, 2\pi)$.

- Compute CDF of random variable R:

$$\begin{aligned} F_R(r) &= \mathcal{P}(R \leq r) \\ &= \mathcal{P}(\sqrt{-2\ln(U_1)} \leq r) \\ &= \mathcal{P}(U_1 \geq e^{-r^2/2}) \\ &= 1 - e^{-r^2/2}, \end{aligned}$$

so density $f_R(r) = \frac{d}{dr}F_R(r) = re^{-r^2/2}$.

- Since U_1 and U_2 are independent, so are R and Θ.
- Therefore, joint density of (R, Θ):

$$f_{R,\Theta}(r, \theta) = f_R(r)\, f_\Theta(\theta) = \frac{1}{2\pi} r e^{-r^2/2}$$

- Now we can work out joint distn of (X_1, X_2):

$$\begin{aligned} &\mathcal{P}(X_1 \leq x_1,\ X_2 \leq x_2) \\ &= \mathcal{P}\left(R\cos(\Theta) \leq x_1,\ R\sin(\Theta) \leq x_2\right) \\ &= \int\int_{\{(r,\theta):\ r\cos(\theta) \leq x_1,\ r\sin(\theta) \leq x_2\}} f_{R,\Theta}(r, \theta)\, dr\, d\theta \\ &= \int\int_{\{(r,\theta):\ r\cos(\theta) \leq x_1,\ r\sin(\theta) \leq x_2\}} \frac{1}{2\pi} e^{-r^2/2}\, r\, dr\, d\theta \end{aligned}$$

- Change of variables :

$$u = r\cos(\theta), \quad v = r\sin(\theta), \quad du\, dv = r\, dr\, d\theta$$

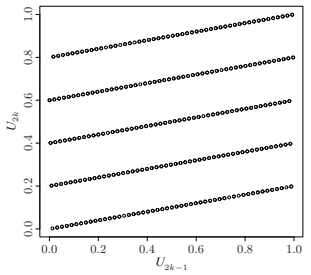
- Thus,

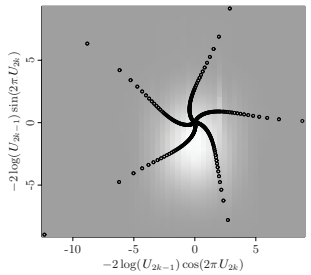
$$\begin{aligned}
&\mathcal{P}(X_1 \le x_1,\ X_2 \le x_2) \\
&= \int\int_{\{(u,v):\, u\le x_1,\ v\le x_2\}} \frac{1}{2\pi} e^{-\frac{u^2+v^2}{2}}\, du\, dv \\
&= \int_{-\infty}^{x_1} \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du \ \cdot\ \int_{-\infty}^{x_2} \frac{1}{\sqrt{2\pi}} e^{-v^2/2} dv
\end{aligned}$$

QED

Remarks on Box-Muller Algorithm:

- Generates two independent random variates from $\mathcal{N}(0,1)$ using two independent unif[0, 1] random numbers U_1 and U_2
  - One uniform per generated random variate.
- However, sin and cos are expensive to compute.
- If U_1, U_2 are successive unif[0, 1]'s from LCG with modulus m, then (U_1, U_2) lies on one of $\le \sqrt{2m}$ lines in unit square.





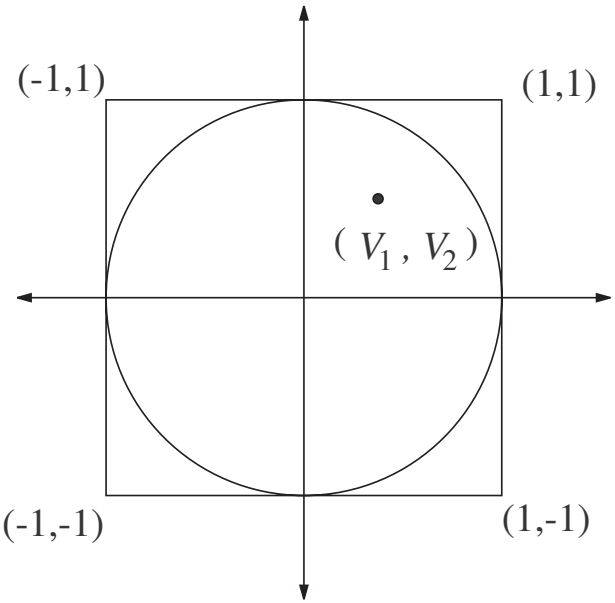
Algorithm N3: Marsaglia-Bray method to generate normals

1. Generate $U_1, U_2 \sim$ unif[0, 1] independent.

2. Let $V_1 = 2U_1 - 1$,
   $V_2 = 2U_2 - 1$,
   $W = V_1^2 + V_2^2$.

3. If $W \le 1$,
   - Set $X_i \leftarrow \sqrt{-2\ln(W)} \cdot \frac{V_i}{\sqrt{W}}$, $i = 1, 2$.
   - Stop and return X_1 and X_2 as IID $\mathcal{N}(0,1)$.

   Else
   - Discard (U_1, U_2) and return to step 1.



Claim : X_1 and X_2 are independent $\mathcal{N}(0,1)$ variables

Proof : Outline is as follows:

- Let (V_1, V_2) be a pair that is *accepted* by Algorithm N3, i.e., $W = V_1^2 + V_2^2 \leq 1$
- Then (V_1, V_2) is *uniformly* distributed on unit disk.
- Let (L, Ψ) be the *polar representation* of (V_1, V_2), i.e.,

$$\begin{aligned} V_1 &= L\cos(\Psi) \\ V_2 &= L\sin(\Psi) \end{aligned}$$

Notice ... $L = \sqrt{V_1^2 + V_2^2} = \sqrt{W}$

- Since (V_1, V_2) uniform on unit disk
  - L and Ψ are independent
  - $L^2 \sim$ unif[0, 1], so L has density $f_L(l) = 2l$ for $l \in [0,1]$
  - $\Psi \sim$ unif$[0, 2\pi]$, i.e., density $f_\Psi(\psi) = 1/(2\pi)$
- Define $R = \sqrt{-4\ln(L)}$ and $\Theta = \Psi$.
  - R and Θ are independent
  - So joint density $f_{R,\Theta}(r,\theta) = f_R(r)\, f_\Theta(\theta)$.
- Since L has CDF $F_L(l) = l^2$, the CDF of R is

$$\begin{aligned} F_R(r) &= \mathcal{P}(\sqrt{-4\ln(L)} \leq r) \\ &= \mathcal{P}(L \geq e^{-r^2/4}) \\ &= 1 - e^{-r^2/2}, \end{aligned}$$

so R has density $f_R(r) = re^{-r^2}$.

- $\Theta = \Psi \quad \Rightarrow \quad f_\Theta(\theta) = 1/(2\pi)$
- Therefore, joint density $f_{R,\Theta}(r,\theta) = \frac{1}{2\pi} r e^{-r^2/2}$
- Algorithm N3 sets

$$X_1 = \sqrt{-2\ln(W)}\frac{V_1}{\sqrt{W}} = R\cos(\Theta)$$

$$X_2 = \sqrt{-2\ln(W)}\frac{V_2}{\sqrt{W}} = R\sin(\Theta)$$

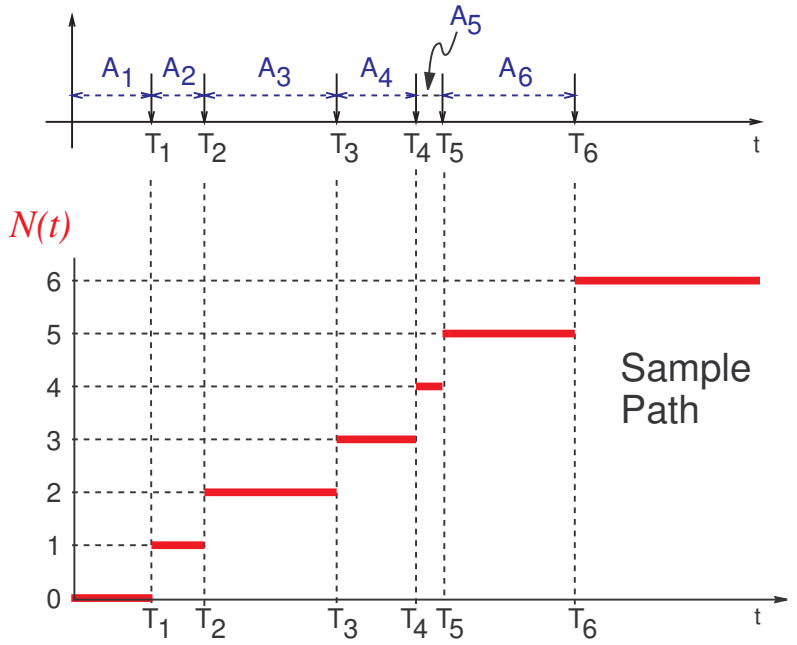
- Rest of proof like that for Algorithm N2.
- Efficiency of Algorithm N3
  - In Step 3, accept (U_1, U_2) with probability $\pi/4$.
  - $E[\#\text{ iterations}] = 4/\pi \approx 1.27$

Renewal Processes

- Model times of events
  - Customer arrivals at car wash, post office, etc.
  - Jumps in price of risky asset
- *Event times*: ith event occurs at time T_i, $i = 1, 2, \ldots$
- *Inter-event times*: A_i = time elapsed between $(i-1)$-th event and i-th event, i.e.,

$$A_1 = T_1$$

$$A_i = T_i - T_{i-1}, \quad i = 2, 3, \ldots$$



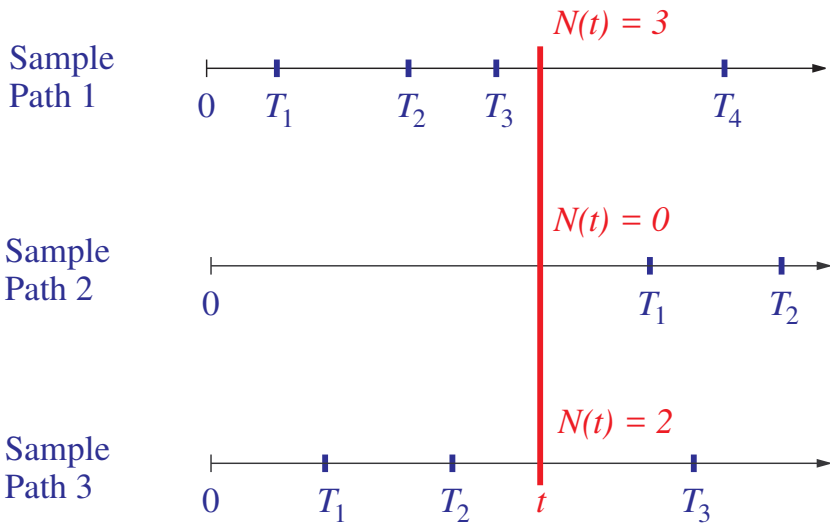
- Equivalent representation : **Counting Process** $[N(t) : t \geq 0]$
  - $N(t) = \#$ events in $[0, t]$
  - $[N(t) : t \geq 0]$ is continuous-time stochastic process taking values in $\mathbf{Z}_+ = \{0, 1, 2, \ldots\}$.
- For $0 \leq s \leq t$, the **increment**

  $$N(t) - N(s) \;=\; \#\text{ events in the interval } (s, t].$$

- **Defn:** A *renewal process* is a counting process $[N(t) : t \geq 0]$ in which inter-event times $A_1, A_2, \ldots$ are IID.
- Simulating renewal processes is easy !!

- Algorithm for generating sample path of renewal process with inter-event times $A_1, A_2, A_3, \ldots$ IID with CDF F:

1. Initialization: Set $i = 1$, $T_0 = 0$.
2. Generate $A_i \sim F$, independent of all A_j, $j < i$.
3. Set $T_i \leftarrow T_{i-1} + A_i$.
4. Set $N(t) = i - 1$ for $t \in [T_{i-1}, T_i)$.
5. Set $i \leftarrow i + 1$, and goto Step 2.



- Suppose $N = [N(t) : t \geq 0]$ is renewal process.
- **Definition:** N has **independent increments** if # of events occurring in *disjoint* intervals are *independent*.
  - if $s \leq t \leq u \leq v$, then
    $N(t) - N(s)$ is independent of $N(v) - N(u)$.
- **Definition:** N has **stationary increments** if distn of # of events that occur in interval depends only on interval's length and not its location.
  - if $(s, t]$ and $(u, v]$ are two intervals both of length τ, then

$$\begin{aligned} \mathcal{P}\{N(t) - N(s) = k\} &= \mathcal{P}\{N(v) - N(u) = k\} \\ &= \mathcal{P}\{N(\tau) - N(0) = k\} \end{aligned}$$

- A Poisson process is a special renewal process.

Poisson process

- **Definition:** Counting process $N = [N(t) : t \geq 0]$ is called **Poisson process with rate** λ if

(a) $N(0) = 0$.

(b) N has independent increments.

(c) N has stationary increments.

(d) For small δt,

$$\mathcal{P}\{N(\delta t) = 1\} \approx \lambda\, \delta t,$$

i.e., $\lim_{\delta t \to 0} \frac{\mathcal{P}\{N(\delta t)=1\}}{\delta t} = \lambda$.

(e) For small δt,

$$\mathcal{P}\{N(\delta t) \geq 2\} = o(\delta t),$$

i.e., $\lim_{\delta t \to 0} \frac{\mathcal{P}\{N(\delta t)\geq 2\}}{\delta t} = 0$

Facts about Poisson process:

1. Number $N(t)$ of events occurring in time interval $[0, t]$ is Poisson random variable with mean λt, i.e.,
$$\mathcal{P}\{N(t) = k\} = e^{-\lambda t}(\lambda t)^k/k!, \qquad k = 0, 1, 2, \ldots.$$
2. Expected event rate at time t is λ, independent of t :
$$\lim_{\delta t \to 0} \frac{\mathcal{E}[N(t + \delta t) - N(t)]}{\delta t} = \lim_{\delta t \to 0} \frac{1(\lambda \delta t) + 0(1 - \lambda \delta t)}{\delta t} = \lambda,$$
where we have used properties (d) and (e).
3. Renewal process $[N(t) : t \geq 0]$ is a *Poisson* process with rate $\lambda > 0$ iff inter-event times IID with $A_i \sim \exp(\lambda)$ for all $i \geq 1$.
4. Conditioned on event $\{N(t) = k\}$, first k event times
$$(T_1, T_2, \ldots, T_k) \stackrel{\mathcal{D}}{=} (U_{(1)}, U_{(2)}, \ldots, U_{(k)}),$$
where $U_1, U_2, \ldots, U_k \sim \text{unif}[0, t]$ IID with sorted values $U_{(1)} \leq U_{(2)} \leq \cdots \leq U_{(k)}$.

Proof of Fact 4: Conditioned on event $\{N(t) = k\}$,
first k event times have same joint distn as k ordered IID unif$[0, t]$.

- Distn of first event time T_1 given $N(t) = 1$: for $0 \leq s \leq t$,

$$\begin{aligned} &\mathcal{P}\{\, T_1 \leq s \mid N(t) = 1 \,\} \\ &\quad = \frac{\mathcal{P}\{\, T_1 \leq s,\ N(t) = 1 \,\}}{\mathcal{P}\{\, N(t) = 1 \,\}} \\ &\quad = \frac{\mathcal{P}\{\, 1 \text{ event in } [0, s],\ 0 \text{ events in } (s, t] \,\}}{\mathcal{P}\{\, N(t) = 1 \,\}} \\ &\quad = \frac{\mathcal{P}\{\, 1 \text{ event in } [0, s] \,\}\, \mathcal{P}\{\, 0 \text{ events in } (s, t] \,\}}{\mathcal{P}\{\, N(t) = 1 \,\}} \\ &\quad = \frac{[(\lambda s)^1 e^{-\lambda s}/1!]\ [(\lambda(t - s))^0 e^{-\lambda(t-s)}/0!]}{(\lambda t)^1 e^{-\lambda t}/1!} \\ &\quad = \frac{s}{t}, \qquad \text{which is CDF of unif}[0, t]. \end{aligned}$$

- Can generalize argument to condition on $\{N(t) = k\}$.

Algorithms for generating Poisson process

Algorithm P1: *approximation* based on properties (d) and (e):
$\mathcal{P}\{N(\delta t) = 1\} \approx \lambda\,\delta t, \quad \mathcal{P}\{N(\delta t) \geq 2\} = o(\delta t)$

1. Initialization:
   Set $t = 0$ and $N(0) = 0$.
   Select *small* interval length δt.
2. Set $t \leftarrow t + \delta t$.
3. Generate $U \sim$ unif$[0, 1]$, indep of all previous U's.
4. If $U \leq \lambda\,\delta t$, set $N(t) \leftarrow N(t - \delta t) + 1$ (1 event);
   Else, set $N(t) \leftarrow N(t - \delta t)$ (no events).
5. Goto step 2.

Approximation for small δt; better (exact) algorithms exist.

Algorithm P2: for generating Poisson process with rate λ:
Based on Fact 3 (IID exponential(λ) inter-event times A_i)

1. Initialization: Set $i = 1$, $T_0 = 0$.
2. Generate $A_i \sim \exp(\lambda)$,
   independent of all A_j, $j < i$.
3. Set $T_i \leftarrow T_{i-1} + A_i$.
4. Set $N(t) = i - 1$ for $t \in [T_{i-1}, T_i)$.
5. Set $i \leftarrow i + 1$, and goto Step 2.

Algorithm P3: for generating Poisson process in $[0, t]$ for *fixed* t:
Based on Fact 4 ($T_i \sim$ unif$[0, t]$ given $\{N(t) = k\}$)

1. Generate $N(t) \sim$ Poisson(λt).
2. If $N(t) = k$, generate independent
   $U_1, U_2, \ldots, U_k \sim$ unif$[0, t]$.
3. Set $T_0 = 0$, and for $i = 1, 2, \ldots, k$, define the ith event time $T_i = U_{(i)}$, i.e., the i-th smallest of the U_j's.
4. For $i = 1, 2, \ldots, k$, let $N(s) = i-1$ for $s \in [T_{i-1}, T_i)$.
   Also, let $N(s) = k$ for $s \in [T_k, t]$.

Explanation of Algorithm P3 for Poisson process:

- In step 2, we generate $N(t) = k$ events
  - $N(t)$ has Poisson distribution with parameter λt by Fact 1.
- Fact 4: given $N(t) = k$, first k event times $(T_1, T_2, \ldots, T_k)$ have same joint distn as k sorted IID unif$[0, t]$.
- Step 3 rearranges $U_1, U_2, \ldots, U_k \sim$ unif$[0, t]$ IID in increasing order to get sequential event times.

Nonhomogeneous Poisson process (NHPP)

- For (homogeneous) Poisson process (HPP), expected rate of events is **constant** and equal to λ
- What if we want to model events where rate is time dependent?
- **Example** : Rate at which cars arrive to toll booth on George Washington Bridge is time-dependent
  - High during commuting hours
  - Lower for non-commuting hours
  - Moreover, it is expected to be periodic.
- Modify HPP to be **time-dependent** by defining rate **function** $\lambda(t) \geq 0$
  - $\lambda(t)$ is event rate at time t

- **Defn:** Counting process $[N(t) : t \geq 0]$ is **Nonhomogeneous Poisson Process with rate function** $\lambda(t)$ (NHPP($\lambda(t)$) if

(a) $N(0) = 0$

(b) Independent increments: if $s \leq t \leq u \leq v$, then $N(t) - N(s)$ indep of $N(v) - N(u)$.

(c) $\lim_{\delta t \to 0} \frac{\mathcal{P}\{N(t+\delta t) - N(t) = 1\}}{\delta t} = \lambda(t)$ (time-dependence)

(d) $\lim_{\delta t \to 0} \frac{\mathcal{P}\{N(t+\delta t) - N(t) \geq 2\}}{\delta t} = 0$

- Definition implies expected event rate at time t is

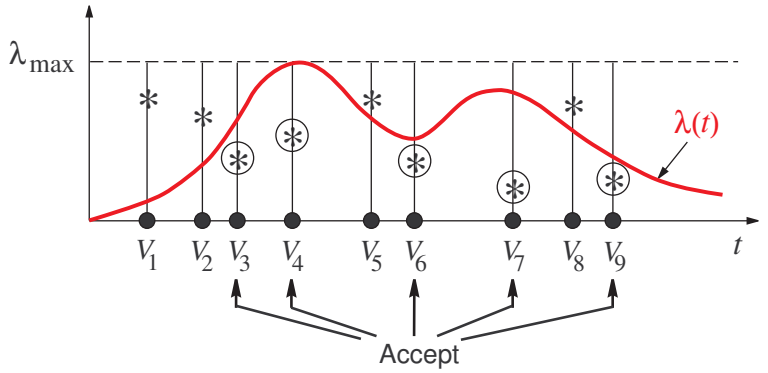
$$\lim_{\delta t \to 0} \frac{\mathcal{E}[N(t+\delta t) - N(t)]}{\delta t} = \lambda(t).$$

- NHPP also called **non-stationary** or **inhomogeneous** Poisson process.

Thinning algorithm for generating NHPP

- Choose $\lambda_{\max} \geq \lambda(t)$ for all $t \geq 0$
- **Algorithm** : Acceptance/rejection for NHPP($\lambda(t)$)

1. Generate HPP with (constant) rate $\lambda_{\max}$.
2. Let $V_1, V_2, \ldots$ be sequence of event times of HPP.
3. Accept event time V_i with probability $\frac{\lambda(V_i)}{\lambda_{\max}}$, indep of all other events;
   Otherwise, reject event time V_i.
4. Let $T_0 = 0$, and let T_i, $i = 1, 2, \ldots,$ be the ith event time of NHPP.
5. For $i = 1, 2, \ldots,$ set $N(t) = i - 1$ for $t \in [T_{i-1}, T_i)$.



- In step 3,

accept event time V_i with probability $\frac{\lambda(V_i)}{\lambda_{\max}}$

$\iff$ $\lambda_{\max} U_i \leq \lambda(V_i)$ for IID $U_i \sim$ Unif[0, 1]

- **Reference:** Lewis, P. A. W. and Shedler, G. S., "Simulation of Non-homogeneous Poisson Processes by Thinning," *Naval Research Logistics Quarterly*, Vol. 26, pp. 403-413, 1979
- Denote sequence of accepted event times by $\{T_k : k \geq 1\}$
- **Claim**: Accepted sequence of events forms an NHPP($\lambda(t)$)
- **Proof sketch**:
  - Properties (a), (b) and (d) are true for (homogeneous) Poisson process with rate $\lambda_{\max}$
  - So corresponding properties will remain true even if some HPP events are dropped.

- To show property (c), let T_k be time of kth event of NHPP, so

$$\begin{aligned}
&\mathcal{P}\{T_k \in (t, t+\delta t)\} \\
&= \mathcal{P}(\{V_i \in (t, t+\delta t)\} \cap \{V_i \text{ accepted}\}) \\
&= \mathcal{P}\{V_i \in (t, t+\delta t)\}\ \mathcal{P}\{V_i \text{ accepted} \mid V_i \in (t, t+\delta t)\} \\
&= [\lambda_{\max}\delta t]\left[\frac{\lambda(t)}{\lambda_{\max}}\right] \\
&= \lambda(t)\,\delta t
\end{aligned}$$

Estimating the Rate Function of NHPP

- Difficulty: need to estimate entire function $\lambda(t)$, $t \geq 0$, and not just a few parameters.
- One approach: assume function $\lambda(t) = a + b\sin(ct)$
  - Estimate parameters a, b, c using least squares.
  - Reference: Kuhl, M.E., and J.R. Wilson. (2000). Least squares estimation of nonhomogeneous Poisson processes. *Journal of Statistical Computation and Simulation*, vol. 67, pp. 75–108.
- Other approaches
  - Section 6.12.2 of Law (2015).

Autoregressive Time-Series Model

- Consider sequence $(Y_1, Y_2, Y_3, \ldots)$ with

$$Y_1 \sim \mathcal{N}\left(0, \frac{1}{1-\theta^2}\right),$$

$$Y_t = \theta Y_{t-1} + \epsilon_t, \qquad t = 2, 3, \ldots$$

  - θ is given constant with $-1 < \theta < 1$.
  - $\epsilon_2, \epsilon_3, \ldots$ are IID $\mathcal{N}(0,1)$
- Then can show $Y_t \sim \mathcal{N}\left(0, \frac{1}{1-\theta^2}\right)$ for each $t = 1, 2, 3, \ldots$
- $(Y_1, Y_2, Y_3, \ldots)$ is **stationary process** with lag-j correlation

$$\rho_j = \text{corr}(Y_t, Y_{t+j}) = \theta^j$$

- Next generalize to allow for arbitrary mean and variance.

- AR(1): Autoregressive process of order 1: $(Y_1, Y_2, Y_3, \ldots)$, with

$$Y_1 \sim \mathcal{N}\left(\mu, \frac{\sigma_\epsilon^2}{1-\theta^2}\right),$$

$$Y_t = \mu + \theta(Y_{t-1} - \mu) + \epsilon_t, \qquad t = 2, 3, \ldots$$

  - μ, σ_ϵ and θ are constants, $-1 < \theta < 1$
  - $\epsilon_2, \epsilon_3, \ldots$ are IID $\mathcal{N}(0, \sigma_\epsilon^2)$

- Then $Y_t \sim \mathcal{N}\left(\mu, \frac{\sigma_\epsilon^2}{1-\theta^2}\right)$ for each $t = 1, 2, 3, \ldots$
- $(Y_1, Y_2, Y_3, \ldots)$ is stationary process with lag-j correlation

$$\rho_j = \text{corr}(Y_t, Y_{t+j}) = \theta^j$$

Algorithm to generate AR(1) time series $(Y_1, Y_2, Y_3, \ldots)$

1. Set $t = 1$, and generate $Y_1 \sim \mathcal{N}(\mu, \sigma_\epsilon^2/(1-\theta^2))$.
2. Set $t \leftarrow t + 1$.
3. Generate $\epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2)$, independent of Y_1 and all previous ϵ_i.
4. Set $Y_t = \mu + \theta(Y_{t-1} - \mu) + \epsilon_t$, and go to Step 2.

Can generalize to AR(p) time series:

$$Y_t = \mu + \theta_1(Y_{t-1} - \mu) + \theta_2(Y_{t-2} - \mu) + \cdots + \theta_p(Y_{t-p} - \mu) + \epsilon_t$$

- Often used in economic and finance models.

Estimating Parameters μ, θ, σ_ϵ of AR(1) Time Series

- Rewrite $Y_t = \mu + \theta(Y_{t-1} - \mu) + \epsilon_t$ as

$$\begin{aligned} Y_t &= (1-\theta)\mu + \theta Y_{t-1} + \epsilon_t \\ &= \beta_0 + \beta_1 Y_{t-1} + \epsilon_t \end{aligned}$$

- Linear regression model with $\beta_0 = (1-\theta)\mu$ and $\beta_1 = \theta$, so use least squares to estimate parameters of AR(1)
- Let $Y_1, Y_2, \ldots, Y_{n+1}$ be data from hypothesized AR(1).
- Least squares: choose $\widehat{\theta}$ and $\widehat{\mu}$ to minimize sum of squared errors

$$S = \sum_{t=2}^{n+1} \left[Y_t - (1-\theta)\mu - \theta Y_{t-1}\right]^2$$

- Estimate σ_ϵ^2 using residuals $\widehat{\epsilon}_t = Y_t - (1-\widehat{\theta})\widehat{\mu} - \widehat{\theta} Y_{t-1}$:

$$\widehat{\sigma}_\epsilon^2 = \frac{1}{n-2} \sum_{t=2}^{n+1} \widehat{\epsilon}_t^2$$

Multivariate Distributions

Example 1:

- Suppose a job consists of 2 subtasks to be completed sequentially.
- So total time to complete job is

$$T = X_1 + X_2$$

  where X_1 and X_2 are subtask times.

- X_1 and X_2 may be **dependent**.
  - If first subtask takes an unusually long time, then same is likely true for second subtask.

Example 2:

- 3 stocks, e.g., GOOG, INTC, MSFT
- Let X_i denote the return of the ith stock tomorrow.
- Often X_1, X_2, X_3 are **dependent**.
  - If GOOG has an unusually large return tomorrow, then INTC and MSFT are also likely to.

Goal: Generate correlated random vector

$$\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_d \end{pmatrix}$$

- Suppose $\mathbf{X} \in \mathbf{R}^d$ is random vector:

$$\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_d \end{pmatrix}$$

where

- X_i and X_j not necessarily independent for $i \neq j$
- $\mathbf{X}$ not necessarily multivariate normal

- Expectation of $\mathbf{X}$:

$$\mathcal{E}[\mathbf{X}] = \begin{pmatrix} \mathcal{E}[X_1] \\ \mathcal{E}[X_2] \\ \vdots \\ \mathcal{E}[X_d] \end{pmatrix} \equiv \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_d \end{pmatrix} \equiv \boldsymbol{\mu}_{\mathbf{x}}$$

- Covariance matrix $\boldsymbol{\Psi}_{\mathbf{x}}$ of $\mathbf{X}$:

$$\boldsymbol{\Psi}_{\mathbf{x}} \equiv \mathcal{E}[(\mathbf{X} - \boldsymbol{\mu}_{\mathbf{x}})(\mathbf{X} - \boldsymbol{\mu}_{\mathbf{x}})^\top]$$

$$= \mathcal{E}\left[\begin{pmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \\ \vdots \\ X_d - \mu_d \end{pmatrix} \begin{pmatrix} X_1 - \mu_1, & X_2 - \mu_2, & \cdots, & X_d - \mu_d \end{pmatrix}\right]$$

$$= \mathcal{E}\begin{bmatrix} (X_1-\mu_1)(X_1-\mu_1) & (X_1-\mu_1)(X_2-\mu_2) & \cdots & (X_1-\mu_1)(X_d-\mu_d) \\ (X_2-\mu_2)(X_1-\mu_1) & (X_2-\mu_2)(X_2-\mu_2) & \cdots & (X_2-\mu_2)(X_d-\mu_d) \\ \vdots & \vdots & \ddots & \vdots \\ (X_d-\mu_d)(X_1-\mu_1) & (X_d-\mu_d)(X_2-\mu_2) & \cdots & (X_d-\mu_d)(X_d-\mu_d) \end{bmatrix}$$

$$= \begin{bmatrix} \mathsf{Cov}(X_1, X_1) & \mathsf{Cov}(X_1, X_2) & \cdots & \mathsf{Cov}(X_1, X_d) \\ \mathsf{Cov}(X_2, X_1) & \mathsf{Cov}(X_2, X_2) & \cdots & \mathsf{Cov}(X_2, X_d) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{Cov}(X_d, X_1) & \mathsf{Cov}(X_d, X_2) & \cdots & \mathsf{Cov}(X_d, X_d) \end{bmatrix}$$

- Suppose
  - $\mathbf{A} = (a_{i,j} : 1 \leq i \leq m,\ 1 \leq j \leq d)$ is matrix of constants
  - $\mathbf{b} = (b_i : 1 \leq i \leq m) \in \mathbf{R}^m$ is column vector of constants
  - $\mathbf{X} = (X_i : 1 \leq i \leq d) \in \mathbf{R}^d$ is random vector with mean $\boldsymbol{\mu}_{\mathbf{x}}$ and covariance matrix $\boldsymbol{\Psi}_{\mathbf{x}}$

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1d} \\ a_{21} & a_{22} & \cdots & a_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{md} \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_d \end{pmatrix}$$

- Consider linear transformation $\mathbf{AX} + \mathbf{b} \in \mathbf{R}^m$

$$\mathbf{AX}+\mathbf{b} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1d} \\ a_{21} & a_{22} & \cdots & a_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{md} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_d \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

$$= \begin{pmatrix} a_{11}X_1 + a_{12}X_2 + \cdots + a_{1d}X_d + b_1 \\ a_{21}X_1 + a_{22}X_2 + \cdots + a_{2d}X_d + b_2 \\ \vdots \\ a_{m1}X_1 + a_{m2}X_2 + \cdots + a_{md}X_d + b_m \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^d a_{1i} X_i + b_1 \\ \sum_{i=1}^d a_{2i} X_i + b_2 \\ \vdots \\ \sum_{i=1}^d a_{mi} X_i + b_m \end{pmatrix}$$

- Expectation of $\mathbf{AX}+\mathbf{b}$:

$$\mathcal{E}[\mathbf{AX}+\mathbf{b}] = \begin{pmatrix} \mathcal{E}[\sum_{i=1}^d a_{1i} X_i + b_1] \\ \mathcal{E}[\sum_{i=1}^d a_{2i} X_i + b_2] \\ \vdots \\ \mathcal{E}[\sum_{i=1}^d a_{mi} X_i + b_m] \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^d a_{1i} \mathcal{E}[X_i] + b_1 \\ \sum_{i=1}^d a_{2i} \mathcal{E}[X_i] + b_2 \\ \vdots \\ \sum_{i=1}^d a_{mi} \mathcal{E}[X_i] + b_m \end{pmatrix}$$

$$= \mathbf{A}\,\boldsymbol{\mu}_{\mathbf{x}} + \mathbf{b}$$

since $\mathcal{E}[aY+bZ] = a\,\mathcal{E}[Y] + b\,\mathcal{E}[Z]$.

- Covariance matrix of $\mathbf{AX}+\mathbf{b}$ is

$$\begin{aligned} \boldsymbol{\Psi}_{\mathbf{AX}+\mathbf{b}} &\equiv \mathcal{E}\left[(\mathbf{AX}+\mathbf{b}-\mathcal{E}[\mathbf{AX}+\mathbf{b}])\,(\mathbf{AX}+\mathbf{b}-\mathcal{E}[\mathbf{AX}+\mathbf{b}])^\top \right] \\ &= \mathcal{E}\left[(\mathbf{AX}-\mathbf{A}\boldsymbol{\mu}_{\mathbf{x}})\,(\mathbf{AX}-\mathbf{A}\boldsymbol{\mu}_{\mathbf{x}})^\top \right] \\ &= \mathcal{E}\left[\mathbf{A}\,(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})\,(\mathbf{A}\,(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})\,)^\top \right] \\ &= \mathcal{E}\left[\mathbf{A}\,(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})\,(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})^\top\,\mathbf{A}^\top \right] \\ &= \mathbf{A}\,\mathcal{E}\left[(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})\,(\mathbf{X}-\boldsymbol{\mu}_{\mathbf{x}})^\top \right]\,\mathbf{A}^\top \\ &= \mathbf{A}\,\boldsymbol{\Psi}_{\mathbf{x}}\,\mathbf{A}^\top \end{aligned}$$

- So $\mathbf{AX}+\mathbf{b}$ has
  - mean $\mathbf{A}\boldsymbol{\mu}_{\mathbf{x}}+\mathbf{b}$
  - covariance matrix $\mathbf{A}\boldsymbol{\Psi}_{\mathbf{x}}\mathbf{A}^\top$

Multivariate Normal Distribution

- Suppose $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)^\top \in \mathbf{R}^d$ with $\mathbf{Z} \sim \mathcal{N}_d(\mathbf{0}, \mathbf{I})$, i.e., $\mathbf{Z}$ is normal random vector with
  - mean vector $\mathbf{0}$
  - covariance matrix $\mathbf{I}$ is $d \times d$ identity matrix:

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

- Then $\mathbf{Z}$ is a vector of IID standard normals
  - Each $Z_i \sim \mathcal{N}(0,1)$
  - $Z_1, Z_2, \ldots, Z_d$ independent
    - ▷ Z_i, Z_j normal and $\mathrm{Cov}(Z_i, Z_j) = 0$ imply Z_i, Z_j indep.

- Define vector $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_d)^\top$ as linear transformation of $\mathbf{Z} \sim \mathcal{N}_d(\mathbf{0}, \mathbf{I})$:

$$\mathbf{Y} = \mathbf{A}\mathbf{Z} + \boldsymbol{\mu}$$

for some constant matrix $\mathbf{A}$ and constant vector $\boldsymbol{\mu}$

  - $\mathbf{Y}$ has mean $\mathbf{A}\,\mathcal{E}[\mathbf{Z}] + \boldsymbol{\mu} = \boldsymbol{\mu}$ since $\mathcal{E}[\mathbf{Z}] = \mathbf{0}$.
  - $\mathbf{Y}$ has covariance matrix $\mathbf{A}\mathbf{I}\mathbf{A}^\top = \mathbf{A}\mathbf{A}^\top \equiv \boldsymbol{\Sigma}$.
  - Since weighted sums of indep normals are normal, can show

$$\mathbf{Y} \;=\; \mathbf{A}\mathbf{Z} + \boldsymbol{\mu} \;=\; \begin{pmatrix} \sum_{i=1}^d a_{1i} Z_i \;+\; \mu_1 \\ \sum_{i=1}^d a_{2i} Z_i \;+\; \mu_2 \\ \vdots \\ \sum_{i=1}^d a_{di} Z_i \;+\; \mu_d \end{pmatrix} \;\sim\; \mathcal{N}(\boldsymbol{\mu},\; \boldsymbol{\Sigma})$$

  - Each $Y_i \sim \mathcal{N}(\mu_i,\, \boldsymbol{\Sigma}_{ii}) \in \mathbf{R}$.

Algorithm: to generate multivariate normal random vector $\mathbf{Y} \sim \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where

- $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_d)^\top \in \mathbf{R}^d$
- mean vector $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_d)^\top \in \mathbf{R}^d$
- covariance matrix $\boldsymbol{\Sigma} = (\Sigma_{ij} : i, j = 1, 2, \ldots, d) \in \mathbf{R}^{d \times d}$

1. Find matrix $\mathbf{A} \in \mathbf{R}^{d \times d}$ such that $\boldsymbol{\Sigma} = \mathbf{A}\mathbf{A}^\top$, e.g., using Cholesky decomposition or singular value decomposition.
2. Generate d IID $\mathcal{N}(0, 1)$ random variates $Z_1, Z_2, \ldots, Z_d$, and set $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)^\top$.
3. Return $\mathbf{Y} = \mathbf{A}\mathbf{Z} + \boldsymbol{\mu}$.

Correlated Variates: NORTA (Normal-To-Anything)

Goal: generate correlated random vector (X_1, X_2) with marginal distributions F_1 and F_2.

- First, some background ...
- Suppose we want to generate X from univariate CDF F.
- Let Φ be CDF of standard normal RV $Z \sim \mathcal{N}(0, 1)$
- Can express $Z = \Phi^{-1}(U)$, where $U \sim$ unif[0, 1]
- Hence, $W \equiv \Phi(Z) \sim$ unif[0, 1] since

$$W \;=\; \Phi(Z) \;=\; \Phi[\Phi^{-1}(U)] \;=\; U$$

- **NORTA transformation** to generate $X \sim F$:

$$X \;=\; F^{-1}[\Phi(Z)] \;\sim\; F$$

since $\Phi(Z) \sim$ unif[0, 1].

- **Goal:** generate bivariate random vector $(X_1, X_2)^\top$ with
  - Marginals $X_1 \sim F_1$ and $X_2 \sim F_2$
  - X_1 and X_2 have correlation ρ_x
  - F_1 and F_2 need not be from same distn family
    - ▷ e.g., F_1 could be exponential, and F_2 could be beta.
- Suppose $(Z_1, Z_2)^\top$ is bivariate normal with $\mathcal{E}[Z_i] = 0$, $\mathcal{V}[Z_i] = 1$, and correlation ρ_z.
  - Marginal distn of Z_i is $\mathcal{N}(0, 1)$ for each $i = 1, 2$.
  - Can generate $(Z_1, Z_2)^\top$ using algorithm on slide 6-135.
- Then $\Phi(Z_1)$ and $\Phi(Z_2)$ are **correlated** unif[0, 1] RVs.
- Use NORTA to generate $(X_1, X_2)^\top$ as

$$X_1 \;=\; F_1^{-1}[\Phi(Z_1)], \qquad X_2 \;=\; F_2^{-1}[\Phi(Z_2)].$$

NORTA Algorithm: to generate multivariate random vector $\mathbf{X} = (X_1, X_2, \ldots, X_d)^\top$ with given

- Marginal CDF F_i for X_i for each $i = 1, 2, \ldots, d$
- Correlation $\rho_{ij} = \text{Corr}[X_i, X_j]$ for each $i \neq j$.

1. Find "appropriate" (see next slide) covariance matrix $\boldsymbol{\Sigma} = (\Sigma_{ij} : i, j = 1, 2, \ldots, d)$ of multivar normal $\mathcal{N}_d(\mathbf{0}, \boldsymbol{\Sigma})$, where $\Sigma_{ii} = 1$ for all $i = 1, 2, \ldots, d$.

2. Generate $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)^\top \sim \mathcal{N}_d(\mathbf{0}, \boldsymbol{\Sigma})$, e.g., using algorithm on p. 6-135.

3. Set $X_i = F_i^{-1}(\Phi(Z_i))$ for each $i = 1, 2, \ldots, d$, and return $\mathbf{X} = (X_1, X_2, \ldots, X_d)^\top$.

Comments on NORTA

- Requires evaluation of normal CDF Φ and inverse F_i^{-1} of desired marginal CDF F_i
  - Φ has no closed-form expression, and F_i^{-1} may not either.
  - Use numerical methods.
- Correlation ρ_{ij} of pair $(X_i, X_j)^\top$ generated by NORTA may not equal correlation Σ_{ij} of bivariate normal $(Z_i, Z_j)^\top$.
  - $|\rho_{ij} - \Sigma_{ij}|$ often small, but not always.
  - ρ_{ij} is nondecreasing function of Σ_{ij}.
  - ρ_{ij} and Σ_{ij} always have same sign.
  - Search algorithm to find Σ_{ij} that approx gives desired ρ_{ij}.
- Correlations ρ_{ij} do not completely specify joint dist of $\mathbf{X}$.
- For $(Z_1, Z_2, \ldots, Z_d)^\top \sim \mathcal{N}_d(\mathbf{0}, \boldsymbol{\Sigma})$ with each $\Sigma_{ii} = 1$, CDF of $(\Phi(Z_1), \Phi(Z_2), \ldots, \Phi(Z_d))^\top$ is **Gaussian copula**.

Other Copulas

- Let $\mathbf{W} = (W_1, W_2, \ldots, W_d)^\top$ be random vector with each $W_i \sim$ unif[0, 1], where $W_1, W_2, \ldots, W_d$ may be dependent.
- A **copula** is the joint CDF C of $\mathbf{W}$:
  $C(w_1, w_2, \ldots, w_d) = P(W_1 \leq w_1, W_2 \leq w_2, \ldots, W_d \leq w_d)$.
- Each copula defines a dependence structure.
- One way of constructing a copula:
  - Start with random vector $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)^\top$ having some joint CDF G with continuous marginal CDFs G_i.
  - Define vector $\mathbf{W}$ with each $W_i = G_i(Z_i) \sim$ unif[0, 1].
  - Joint CDF C_G of $\mathbf{W}$ is then the copula induced by G.
- To construct random vector $\mathbf{X} = (X_1, X_2, \ldots X_d)^\top$ with copula C_G and each X_i having marginal CDF F_i
  - $X_i = F_i^{-1}(G_i(Z_i))$ for each $i = 1, 2, \ldots, d$.

AutoRegressive-to-Anything (ARTA)

- Recall AR(1) from slide 6-123: $(Y_1, Y_2, Y_3, \ldots)$, with

$$Y_1 \sim \mathcal{N}\left(\mu, \frac{\sigma_\epsilon^2}{1 - \theta^2}\right),$$
$$Y_t = \mu + \theta(Y_{t-1} - \mu) + \epsilon_t, \qquad t = 2, 3, \ldots$$

- μ, σ_ϵ and θ are constants, $-1 < \theta < 1$
- $\epsilon_2, \epsilon_3, \ldots$ are IID $\mathcal{N}(0, \sigma_\epsilon^2)$
- Then $Y_t \sim \mathcal{N}(\mu, \sigma_\epsilon^2/(1 - \theta^2))$ for each $t = 1, 2, 3, \ldots$
- $\text{Corr}(Y_t, Y_{t+j}) = \theta^j$
- Choose $\mu = 0$ and $\sigma_\epsilon^2 = 1 - \theta^2$, so $Y_t \sim \mathcal{N}(0, 1)$.
- Apply NORTA to $(Y_1, Y_2, Y_3, \ldots)$ to get time series $(X_1, X_2, X_3, \ldots)$ with marginal CDF F:

$$X_t = F^{-1}[\Phi(Y_t)]$$

References on NORTA, ARTA, and Copulas:

- Cario, M. C. and B. L. Nelson (1998), "Numerical Methods for Fitting and Simulating Autoregressive-to-Anything Processes," *INFORMS Journal on Computing*, 10, 72–81.
- Biller, B. and S. Ghosh (2006), "Multivariate Input Processes," Chapter 5 in *Simulation: Handbooks in Operations Research and Management Science, Vol. 13*, S. G. Henderson and B. L. Nelson (Eds.), Elsevier.
- Nelsen, R. B. (1999), *Introduction to Copulas*, Springer.
- McNeil A. J., R. Frey, and P. Embrechts (2005), *Quantitative Risk Management: Concepts, Techniques, Tools*. Princeton, NJ: Princeton University Press.