Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.
- A sample x of X is generated as follows:

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.
- A sample x of X is generated as follows:
  1. Generate k exponential random variates, say, $x_1, x_2, \ldots, x_k$:

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.
- A sample x of X is generated as follows:
  1. Generate k exponential random variates, say, $x_1, x_2, \ldots, x_k$:
     1. Generate N uniform random numbers $r_1, r_2, \ldots, r_N$ in $[0, 1]$.

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.
- A sample x of X is generated as follows:
  1. Generate k exponential random variates, say, $x_1, x_2, \ldots, x_k$:
     1. Generate N uniform random numbers $r_1, r_2, \ldots, r_N$ in $[0, 1]$.
     2. For $i = 1, 2, \ldots, k$, $x_i = -\frac{1}{\lambda} \ln(r_i)$.

Erlang random variate

- Let X be an Erlang random variable with parameters k, λ.
- Then $X = X_1 + X_2 + \cdots + X_k$, where X_i's are exponential random variable with parameter λ.
- A sample x of X is generated as follows:
  1. Generate k exponential random variates, say, $x_1, x_2, \ldots, x_k$:
     1. Generate N uniform random numbers $r_1, r_2, \ldots, r_N$ in $[0, 1]$.
     2. For $i = 1, 2, \ldots, k$, $x_i = -\frac{1}{\lambda} \ln(r_i)$.
  2. $x = x_1 + x_2 + \cdots + x_k$.

Poisson random variate

- Let X be a Poisson random variable with parameter λ.

Poisson random variate

- Let X be a Poisson random variable with parameter λ.
- The event $X = n$ can be interpreted as the n arrivals in one unit time.

Poisson random variate

- Let X be a Poisson random variable with parameter λ.
- The event $X = n$ can be interpreted as the n arrivals in one unit time.
- From the definition of Poisson process, we know that the interarrival times $T_1, T_2, \ldots$ are exponentially distributed with parameter λ.

Poisson random variate

- Let X be a Poisson random variable with parameter λ.
- The event $X = n$ can be interpreted as the n arrivals in one unit time.
- From the definition of Poisson process, we know that the interarrival times $T_1, T_2, \ldots$ are exponentially distributed with parameter λ.
- Hence the event $X = n$ is equivalent to

$$T_1 + T_2 + \cdots + T_n \leq 1 < T_1 + T_2 + \cdots + T_{n+1}$$

Poisson random variate · · ·

- That is, if there are n arrivals in one unit time, the sum of interarrival times of these n arrivals has to be less than or equal to one, but if one more interarrival time is added, it must be greater than one (unit time).

Poisson random variate ···

- That is, if there are n arrivals in one unit time, the sum of interarrival times of these n arrivals has to be less than or equal to one, but if one more interarrival time is added, it must be greater than one (unit time).
- A sample t_i of T_i is generated as follows:

$$t_i = -\frac{1}{\lambda}\ln(r_i), \qquad i = 1, 2, \ldots, n+1,$$

where r_i is a uniform random number in $[0, 1]$.

Poisson random variate ···

$$\therefore \sum_{k=1}^{n} -\frac{1}{\lambda}\ln(r_k) \leq 1 < \sum_{k=1}^{n+1} -\frac{1}{\lambda}\ln(r_k)$$

$$\ln\left(\prod_{k=1}^{n} r_k\right) \geq \lambda > \ln\left(\prod_{k=1}^{n+1} r_k\right)$$

$$\prod_{k=1}^{n} r_k \geq e^{-\lambda} > \prod_{k=1}^{n+1} r_k$$

Therefore, the sample n of X is nothing but one which satisfies the above inequality.

Algorithm to generate Poisson random variate

1. Set $n = 0,\ prod = 1$.

Algorithm to generate Poisson random variate

1. Set $n = 0,\ prod = 1$.
2. Generate a uniform random number r in $[0, 1]$. Set $prod = r$.

Algorithm to generate Poisson random variate

1. Set $n = 0,\ prod = 1$.
2. Generate a uniform random number r in $[0, 1]$. Set $prod = r$.
3. while $(prod \geq e^{-\lambda})$
   $n = n + 1$
   generate a random number r
   $prod = r * prod$

   end

Algorithm to generate Poisson random variate

1. Set $n = 0,\ prod = 1$.
2. Generate a uniform random number r in $[0, 1]$. Set $prod = r$.
3. while $(prod \geq e^{-\lambda})$
   $n = n + 1$
   generate a random number r
   $prod = r * prod$
   end
4. The sample of X is n.

Alternate algorithm to generate Poisson random variate

- The random variable X is Poisson with parameter λ if

$$p_i = Pr\{X = i\} = e^{-\lambda}\frac{\lambda^i}{i!}, \qquad i = 0, 1, 2 \ldots$$

Alternate algorithm to generate Poisson random variate

- The random variable X is Poisson with parameter λ if

$$p_i = Pr\{X = i\} = e^{-\lambda}\frac{\lambda^i}{i!}, \qquad i = 0, 1, 2 \ldots$$

- It can be written as

$$p_{i+1} = \frac{\lambda}{i+1}p_i, \qquad i = 0, 1, \ldots$$

with the initial condition

$$p_0 = e^{-\lambda}.$$

Alternate algorithm to generate Poisson random variate · · ·

1. Generate a uniform random number r in $[0, 1]$.

Alternate algorithm to generate Poisson random variate · · ·

1. Generate a uniform random number r in $[0, 1]$.
2. Set $i = 0,\ p = e^{-\lambda},\ F = p$.

Alternate algorithm to generate Poisson random variate $\cdots$

1. Generate a uniform random number r in $[0, 1]$.
2. Set $i = 0,\ p = e^{-\lambda},\ F = p$.
3. If $r < F$, set $X = i$ and stop.

Alternate algorithm to generate Poisson random variate $\cdots$

1. Generate a uniform random number r in $[0, 1]$.
2. Set $i = 0,\ p = e^{-\lambda},\ F = p$.
3. If $r < F$, set $X = i$ and stop.
4. Else $p = \dfrac{\lambda}{i+1} p,\ F = F + p,\ i = i + 1$.

Alternate algorithm to generate Poisson random variate $\cdots$

1. Generate a uniform random number r in $[0, 1]$.
2. Set $i = 0,\ p = e^{-\lambda},\ F = p$.
3. If $r < F$, set $X = i$ and stop.
4. Else $p = \dfrac{\lambda}{i+1} p,\ F = F + p,\ i = i + 1$.
5. Go to Step 3.

Composition method

- Suppose the CDF $F_X(x)$ of a random variable X can be written as

$$F_X(x) = \sum_{i=1}^{m} \alpha_i F_i(x)$$

where $\alpha_i \geq 0$, $\sum_{i=0}^{m} \alpha_i = 1$, that is, α_i is a PMF, and each $F_i(x)$ is a CDF.

Composition method

- Suppose the CDF $F_X(x)$ of a random variable X can be written as

$$F_X(x) = \sum_{i=1}^{m} \alpha_i F_i(x)$$

where $\alpha_i \geq 0$, $\sum_{i=0}^{m} \alpha_i = 1$, that is, α_i is a PMF, and each $F_i(x)$ is a CDF.

- Clearly,

$$f_X(x) = \sum_{i=1}^{m} \alpha_i f_i(x).$$

Composition method $\cdots$

1. Define a new discrete random variable I as follows:

$$p_j = Pr\{I = j\} = \alpha_j, \qquad j = 1, 2, \ldots, m$$

Composition method ···

1. Define a new discrete random variable I as follows:

$$p_j = Pr\{I = j\} = \alpha_j, \qquad j = 1, 2, \ldots, m$$

2. Let X_j be the random variable with CDF $F_j(x)$.

Composition method ···

1. Define a new discrete random variable I as follows:

$$p_j = Pr\{I = j\} = \alpha_j, \qquad j = 1, 2, \ldots, m$$

2. Let X_j be the random variable with CDF $F_j(x)$.
3. If I takes the value j, then simulate a sample of the random variable X_j according to the CDF $F_j(x)$.

Composition method $\cdots$

Algorithm:

1. Generate a sample of I.

Composition method $\cdots$

Algorithm:

1. Generate a sample of I.
2. If $I = j$, generate a sample x_j of X_j.

Composition method $\cdots$

Algorithm:

1. Generate a sample of I.
2. If $I = j$, generate a sample x_j of X_j.
3. Then the sample of X is x_j.

Hyperexponential random variate

- Let X be an hyperexponential random variable with parameters $K, \alpha_i, \lambda_i,\ i = 1, 2, \ldots, K$.

Hyperexponential random variate

- Let X be an hyperexponential random variable with parameters $K, \alpha_i, \lambda_i,\ i = 1, 2, \ldots, K$.
- The CDF is given by $F_X(x) = \sum_{i=1}^{K} \alpha_i F_i(x)$ where $F_i(x) = 1 - e^{-\lambda_i x}$.

Hyperexponential random variate

- Let X be an hyperexponential random variable with parameters $K, \alpha_i, \lambda_i,\ i = 1, 2, \ldots, K$.
- The CDF is given by $F_X(x) = \sum_{i=1}^{K} \alpha_i F_i(x)$ where $F_i(x) = 1 - e^{-\lambda_i x}$.

1. Generate the discrete random variable I with pmf $p_i = \alpha_i,\ i = 1, 2, \ldots, K$

Hyperexponential random variate

- Let X be an hyperexponential random variable with parameters $K, \alpha_i, \lambda_i,\ i = 1, 2, \ldots, K$.
- The CDF is given by $F_X(x) = \sum_{i=1}^{K} \alpha_i F_i(x)$ where $F_i(x) = 1 - e^{-\lambda_i x}$.

1. Generate the discrete random variable I with pmf $p_i = \alpha_i,\ i = 1, 2, \ldots, K$
2. Let $X_j \sim exp(\lambda_j),\ j = 1, 2, \ldots, K$. If $I = j$, then generate a sample x_j of X_j.

Hyperexponential random variate

- Let X be an hyperexponential random variable with parameters $K, \alpha_i, \lambda_i,\ i = 1, 2, \ldots, K$.
- The CDF is given by $F_X(x) = \sum_{i=1}^{K} \alpha_i F_i(x)$ where $F_i(x) = 1 - e^{-\lambda_i x}$.

1. Generate the discrete random variable I with pmf $p_i = \alpha_i,\ i = 1, 2, \ldots, K$
2. Let $X_j \sim exp(\lambda_j),\ j = 1, 2, \ldots, K$. If $I = j$, then generate a sample x_j of X_j.
3. Then a sample value of x of X is given by

$$x = -\frac{1}{\lambda_j} \ln x.$$

Acceptance-Rejection method

The acceptance-rejection method is usually used when the inverse transform method is not directly applicable or is inefficient.

Aim: To simulate a sample x of X with pdf f_X.

1. Consider a random variable Y with pdf f_Y whose sample can be simulated.

Acceptance-Rejection method

The acceptance-rejection method is usually used when the inverse transform method is not directly applicable or is inefficient.

Aim: To simulate a sample x of X with pdf f_X.

1. Consider a random variable Y with pdf f_Y whose sample can be simulated.
2. Choose a constant c such that

$$f_X(t) \leq c f_Y(t), \qquad \forall t.$$

The random variable Y is said to **majorize** X.

Acceptance-Rejection method

The acceptance-rejection method is usually used when the inverse transform method is not directly applicable or is inefficient.

Aim: To simulate a sample x of X with pdf f_X.

1. Consider a random variable Y with pdf f_Y whose sample can be simulated.
2. Choose a constant c such that

$$f_X(t) \leq c f_Y(t), \qquad \forall t.$$

The random variable Y is said to **majorize** X.

1. Suppose X and Y are discrete random variables with pmfs $p_i = Pr\{X = i\}$, $q_i = Pr\{Y = i\}$. Then c can be chosen as follows:

$$c = \max\left\{\frac{p_j}{q_j} : j \geq 1\right\}$$

Acceptance-Rejection method

The acceptance-rejection method is usually used when the inverse transform method is not directly applicable or is inefficient.

Aim: To simulate a sample x of X with pdf f_X.

1. Consider a random variable Y with pdf f_Y whose sample can be simulated.
2. Choose a constant c such that

$$f_X(t) \leq c f_Y(t), \qquad \forall t.$$

The random variable Y is said to **majorize** X.

1. Suppose X and Y are discrete random variables with pmfs $p_i = Pr\{X = i\}$, $q_i = Pr\{Y = i\}$. Then c can be chosen as follows:

$$c = \max\left\{\frac{p_j}{q_j} : j \geq 1\right\}$$

2. Suppose X and Y are continuous random variables with PDFs f_X and f_Y. Then c can be chosen as

$$c = \max\left\{\frac{f_X(x)}{f_Y(x)} : \forall x\right\}.$$

Acceptance-Rejection method ···

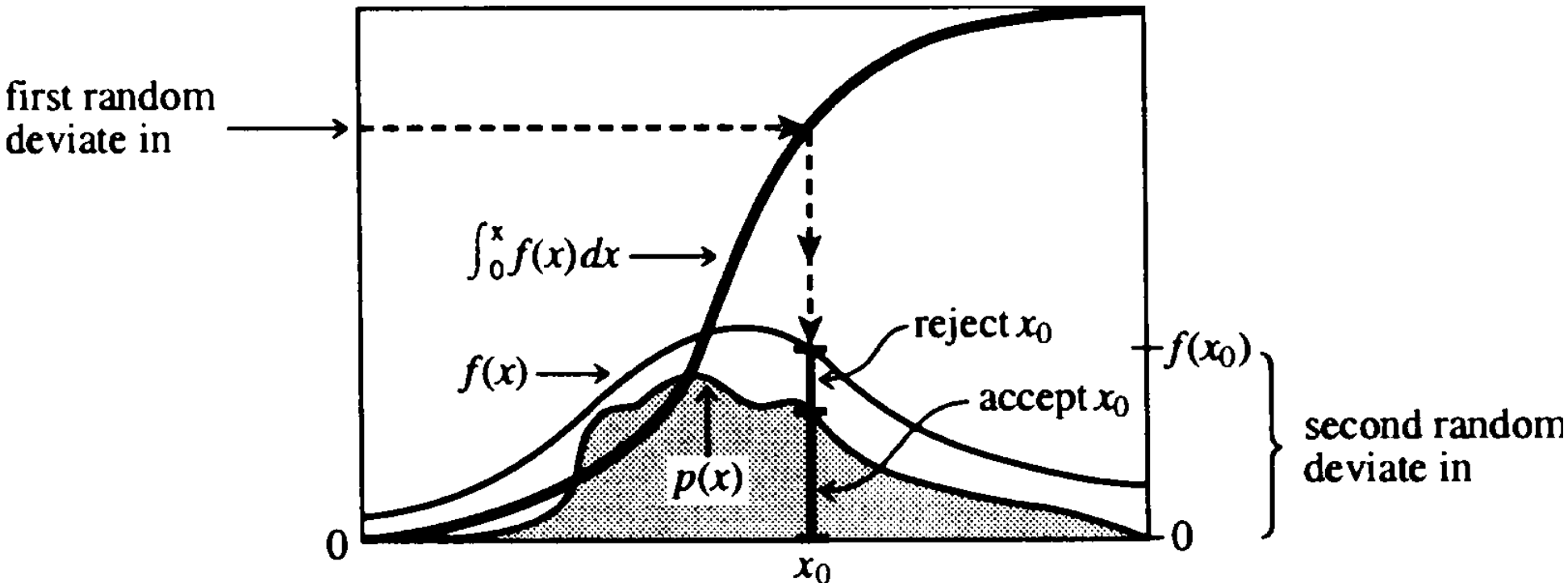

Figure: Acceptance-Rejection method

Acceptance-Rejection method $\cdots$

The algorithm for this method is as follows:

1. Generate a random variate y_1 from the distribution of Y.

Acceptance-Rejection method ···

The algorithm for this method is as follows:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number variate y_2 between 0 and $cf_Y(y_1)$ as follows:

Acceptance-Rejection method ···

The algorithm for this method is as follows:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number variate y_2 between 0 and $cf_Y(y_1)$ as follows:
    1. Generate a uniform random number r.

Acceptance-Rejection method ···

The algorithm for this method is as follows:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number variate y_2 between 0 and $cf_Y(y_1)$ as follows:
   1. Generate a uniform random number r.
   2. Compute $y_2 = 0 + (cf_Y(y_1) - 0)r = cf_Y(y_1)r$.

Acceptance-Rejection method ···

The algorithm for this method is as follows:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number variate y_2 between 0 and $cf_Y(y_1)$ as follows:
   1. Generate a uniform random number r.
   2. Compute $y_2 = 0 + (cf_Y(y_1) - 0)r = cf_Y(y_1)r$.
3. If $y_2 < f_X(y_1)$, then $x = y_1$, else go to Step 1.

Acceptance-Rejection method $\cdots$

The above algorithm is equivalent to the following: steps:

Acceptance-Rejection method $\cdots$

The above algorithm is equivalent to the following: steps:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number r in $[0, 1]$.

Acceptance-Rejection method $\cdots$

The above algorithm is equivalent to the following: steps:

1. Generate a random variate y_1 from the distribution of Y.
2. Generate a uniform random number r in $[0, 1]$.
3. If $r \leq \dfrac{f_X(y_1)}{cf_Y(y_1)}$, then $x = y_1$, else go back to step 1.