Generating Random Variates

- **Overview**
  - We will discuss algorithms for generating observations ("variates") from non-uniform distributions (e.g. Exponential, Weibull, etc.)
  - Generating random variates is also known as sampling.
  - The algorithms depend on the form of the desired distribution for random variable \( X \). But they have the same general form:

    ![Diagram]

    Generate IID \( U(0,1) \) random number(s) → Transformation → Return \( X \)

  - So generating "good" \( U(0,1) \) is critical.
  - Some desired properties of variates generating algorithm are as follows:
    - Efficiency (low storage);
    - Fast (setup, marginal);
    - Robustness (working well for all parameter values);
    - Simplicity (understand and implement with ease);
    - Requiring only \( U(0,1) \) (and preferably \( 1 U \rightarrow 1 X \)).
• Inverse Transform Method

➢ This is the “simplest” and probably “best” method there is. (Its quality varies among distributions though.)

➢ This method is based on the following theorem.

**Theorem** If $F(x)$ is the distribution function of a continuous random variable (rv) $X$ (i.e., $F(x) = P\{X < x\}$), then the rv $U = F(X)$ is uniformly distributed on $(0,1)$, i.e. $U \sim U(0,1)$.

**Proof.**

\[
P\{U < F(x)\} = P\{F(X) < F(x)\} = P\{F^{-1}(F(X)) < F^{-1}(F(x))\} = P\{X < x\} = F(x),
\]

where $F^{-1}$ is the inverse of $F$ (i.e., if $u = F(x) \Rightarrow x = F^{-1}(u)$).

Noting that if $V \sim U(0,1)$, $P\{V < v\} = v$ completes the proof. ■

➢ It follows from the theorem that we can write $X = F^{-1}(U)$, where $U \sim U(0,1)$.

➢ This leads to the inverse transform method for generating $X$.

1. Generate $u \sim U(0,1)$

2. Set $X = F^{-1}(u)$
The main difficulty in using the inverse transform method is inverting \( F \).

This can be done easily using simple analytical arguments in some cases (e.g. exponential distribution).

In other cases, inverting \( F \) requires approximate numerical methods (e.g. the normal distribution).

- **Example 1**

  Develop an algorithm to generate random variates from an exponential rv \( X \), which is exponentially distributed with mean \( 1/\lambda \).

  First, derive \( F^{-1} \). Note that \( F(x) = 1 - e^{-\lambda x} \). Then, setting \( u = F(x) \) implies

  \[
  u = 1 - e^{-\lambda x} \Rightarrow e^{-\lambda x} = 1 - u \Rightarrow -\lambda x = \ln(1 - u) \Rightarrow x = -\frac{\ln(1 - u)}{\lambda}.
  \]

  Therefore, \( F^{-1}(u) = -\frac{\ln(1 - u)}{\lambda} \).
Note that if \( U \sim U(0,1) \Rightarrow 1 - U \sim U(0,1) \).

The algorithm for generating \( X \) is as follows:

1. Generate \( U \sim U(0,1) \)
2. Set \( X = -\frac{\ln U}{\lambda} \).

Applying this algorithm in Excel to produce 100 exponential random variables with \( \lambda = 1 \) yielded the following graph, plotted by the ExperFit software.

The generated histogram closely fits the density function of the exponential distribution, \( f(x) = \lambda e^{-\lambda x} \).
• Example 2
   ➢ Develop an algorithm to generate random variates from the standard normal r.v., $Z$.
   ➢ The distribution function of $Z$ is
     \[ F(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{t^2}{2}} dt , \]
     which is not invertible analytically, but maybe, numerically.

• Intuition behind the inverse transform method
   ➢ Transform the $u$’s which are uniformly distributed on the vertical axis into $x$’s which are dense in areas where the density of $X, f_X(x)$ is high.
- **Inverse transform for discrete distributions**

  - This is similar to the continuous case, but now the distribution function is not smooth,

  \[ F(x_i) = \sum_{x_j \leq x_i} p(x_j), \]

  Where \( p(x_j) = P\{X = x_j\} \).

  - The inverse transform method is then applied as follows.

    1. Generate \( U \sim U(0,1) \)
    2. Find the smallest \( x_i \) such that \( U \leq F(x_i) \)
    3. Set \( X = x_i \)

  - The inverse transform method can be applied exactly for any discrete distribution.

  - However, it may not be too efficient due to the search in step 2.
• **Example 3**
  - The daily demand for a commodity, $X$, takes on values 1, 2, and 3 with probabilities 0.3, 0.5, and 0.2.
  - In this case, $F(1) = 0.3$, $F(2) = 0.8$, and $F(3) = 1$.
  - Then, $X$ is generated as follows.
    1. Generate $U \sim U(0,1)$
    2. If $U \leq 0.3$, set $X = 1$.
       If $0.3 < U \leq 0.8$, set $X = 2$.
       Otherwise, set $X = 3$.

• **Composition Method**
  - Consider a rv $X$ that takes on two, or more, other rv’s at random.
  - E.g., suppose you’re equally likely to choose roads 1 and 2 every day. Suppose travel times on roads 1 and 2, $X_1$ and $X_2$, are exponentially distributed with rates $\lambda_1$ and $\lambda_2$.
  - Then, your travel time is a composition or a “mixture” of $X_1$ and $X_2$. (This is called a hyperexponential distribution.)
  - The distribution function of $X$ can be written as
    $$F_X(x) = 0.5F_{X_1}(x) + 0.5F_{X_2}(x).$$
  - In general, there are many such mixed distributions with distribution functions of the form
    $$F_X(x) = \sum_i p_i F_{X_i}(x).$$
Generating from such distribution is done using the composition method as follows:

1. Generate an integer $J$ such that $P\{J = j\} = p_j$
2. Generate $X_j$ and set $X = X_j$.

These two steps will require generating two or more $U(0,1)$.

- **Example 4**
  - Develop an algorithm to generate random variates from a hyperexponential distribution, with distribution function
    \[
    F_X(x) = 0.5(1 - e^{-\lambda_1 x}) + 0.5(1 - e^{-\lambda_2 x})
    \]
  - The algorithm is as follows.
    1. Generate $U_1 \sim U(0,1)$
    2. Generate $U_2 \sim U(0,1)$. If $U_1 < 0.5$, set $X = -(1/\lambda_1)\ln(U_2)$. Otherwise, set $X = -(1/\lambda_2)\ln(U_2)$.

- **Convolution Method**
  - Some rvs can be written as a sum (convolution) of other independent rvs. That is,
    \[
    X = \sum_{i=1}^{m} X_i.
    \]
  - In this case, $X$ can be generated as follows:
    1. Generate $X_1, X_2, \ldots, X_m$
    2. Set $X = \sum_{i=1}^{m} X_i$. 
• **Example 5**
  - An Erlang random variable, \( X \), is the sum of \( m \) iid exponential rvs with rate \( \lambda \).
  - The following algorithm could be used to generate from \( X \).
    1. Generate \( U_1, U_2, \ldots, U_m \sim U(0,1) \)
    2. Set \( X_1 = -\ln(U_1), X_2 = -\ln(U_2), \ldots, X_m = -\ln(U_m) \)
    2. Set \( X = \sum_{i=1}^{m} X_i \).

• **Composition versus Convolution**
  - Composition: Expresses the distribution function as a (weighted) sum of other distribution functions.
  - Convolution: Expresses the random variable itself as the sum of other random variables.
  - Not the same thing.

• **Acceptance-Rejection Method**
  - Generally used when inverting \( F \) is difficult and no efficient convolution or composition method exist.
  - Acceptance-rejection works by specifying a function \( t(x) \) that majorizes the density of \( X \), i.e., \( t(x) \geq f(x) \) for all \( x \).
Then, a density function is specified as

\[ r(x) = \frac{t(x)}{c}, \text{ where } c = \int_{-\infty}^{\infty} t(x) dx. \]

The acceptance rejection algorithm is as follows.

1. Generate \( Y \sim r(x) \).
2. Generate \( U \sim U(0,1) \).
3. If \( U \leq f(Y) / t(Y) \), (accept) set \( X = Y \). Otherwise, (reject) go back to Step 2.

It can be shown formally that this method is valid.\(^1\)

It can be shown that the acceptance probability is \( 1/c \), i.e.,

\[ P\{U \leq f(Y) / t(Y)\} = 1/c, \]

which is a measure of efficiency.

- **Example 6**

Consider generating variates from a Beta distribution having the following density function,

\[ f(x) = \begin{cases} 
60x^3(1-x)^2, & 0 \leq x \leq 1, \\
0, & \text{otherwise}
\end{cases} \]

It can be shown analytically that \( f(x) < t(x) \), where

\[ t(x) = \begin{cases} 
2.0736, & 0 \leq x \leq 1, \\
0, & \text{otherwise}
\end{cases} \]

---

\(^1\) The idea of the proof is to show that \( P\{X \leq x \} = P\{Y \leq x \mid A \} \), where \( A \) is event of “accepting” \( X \) and setting \( X = Y \) in Step 3 of the algorithm. The proof follows by conditioning on \( Y \). E.g.,

\[ P\{A\} = P\{U \leq f(Y) / t(Y)\} = \int_{-\infty}^{\infty} P\{U \leq f(y) / t(y)\} r(y) dy = \int_{-\infty}^{\infty} \left( \int_{-\infty}^{\infty} f(y) / t(y) dy \right) r(y) dy = (1/c) \int_{-\infty}^{\infty} f(y) dy = 1/c. \]
In this case, \( c = \int_0^1 t(x)\,dx = \int_0^1 2.0736\,dx = 2.0736 \).

Then,
\[
r(x) = \begin{cases} 
1, & 0 \leq x \leq 1, \\
0, & \text{otherwise}
\end{cases}
\]

Therefore, \( Y \sim r(x) \) is \( U(0,1) \).

The acceptance-rejection algorithm is as follows:

1. Generate \( Y \sim U(0,1) \).
2. Generate \( U \sim U(0,1) \).
3. If \( U \leq 60Y^3(1-Y)^2/2.0736 \), (accept) set \( X = Y \). Otherwise, (reject) go back to Step 2.

In this case the acceptance probability is \( 1/c = 0.48 \). The algorithm could be a bit slow.
- Intuition behind Acceptance-Rejection

- Think in term of the Beta distribution in Example 6.
- The AR algorithm is transforming the $Y \sim U(0,1)$ into $X \sim \text{Beta}$, by admitting $Y$ variates with large $X$ density (with $f(Y) \geq Ut(Y)$) as $X$ variates.