Chapter 1

Expectation

Goals:

1. approximating expectations
2. some basics on random number generators
3. some simulation strategies

1.1 Approximating expectations

Many statistical calculations revolve around the computation of an expectation. Suppose $\underset{\sim}{X} = \{X_1, X_2, \ldots, X_n\}$ is a random vector with probability density (or mass function) $f(\underset{\sim}{x}|\theta) = f(x_1, x_2, \ldots, x_n|\theta)$ where θ is a p-dimensional parameter. Then, if $g(X)$ is a function of X, the expected value of $g(X)$ is defined to be

$$\begin{aligned} \mu \equiv \mathrm{E}_\theta[g(X)] &= \int_{\mathbb{R}^k} g(x)f(x)\,\mathrm{d}x \qquad x \text{ continuous} \\ &= \sum_{x:f(x)>0} g(x)f(x) \qquad x \text{ discrete.} \end{aligned}$$

Note that the integral and sum may be over a high dimensional space (i.e., k could be large). Note that any probability can be expressed as an expectation. For example, if we wish to know for some set C

$$\begin{aligned}\Pr_\theta[t(X) \in C] &= \int_{\{x:f(x)\in C\}} f(x)\,\mathrm{d}x \\ &= \int_{\mathbb{R}^k} 1_{\{t(X)\in C\}} f(x)\,\mathrm{d}x \\ &= \mathrm{E}_\theta[1_{\{t(X)\in C\}}] \\ &= \mathrm{E}_\theta[g(X)]\end{aligned}$$

where

$$g(x) \;=\; 1_{\{t(x)\in C\}} \;=\; \begin{cases} 1 & t(x) \in C \\ 0 & \text{else} \end{cases} .$$

The same representation holds when X is discrete, i.e., in general,

$$\Pr_\theta[t(X) \in C] \;=\; \mathrm{E}_\theta[g(X)].$$

More generally, $g(X)$ may depend on θ or on θ_0, a specific value of θ. It is important to recognize that the form of $g(X)$ could be exceedingly complex.

Example: Multinomial Suppose

$$\begin{aligned}\underset{\sim}{X} = \{X_1, X_2, \ldots, X_k\} &\sim \text{Multinomial}(N, \underset{\sim}{\theta} = \{\theta_1, \theta_2, \ldots, \theta_k\}) \\ \Pr_{\underset{\sim}{\theta}}(X_1 = x_1, X_2 = x_2, \ldots, X_k = x_k) &= \frac{N!}{x_1!x_2!\cdots x_k!}\theta_1^{x_1}\theta_2^{x_2}\cdots\theta_k^{x_k},\end{aligned}$$

where $x_i \geq 0$ is integer valued with $x_1 + x_2 + \cdots + x_k = N$.

The Multinomial is used as a model for a situation where every unit in a population falls into exactly one of k mutually exclusive and exhaustive

categories. The population proportion in category i is θ_i. If we select N at random with replacement from the population and let X_i be the number sampled from group i, then $\underset{\sim}{X} = \{X_1, X_2, \ldots, X_k\}$ is Multinomial$(N, \underset{\sim}{\theta})$.

Remarks This is a generalization of the Binomial distribution, and in particular the **marginal** distribution of X_i is

$$\begin{aligned} X_i &\sim \text{Binomial}(N, \theta_i) \\ \Pr_{\theta_i}(X_i = x_i) &= \binom{N}{x} \theta_i^x (1-\theta_i)^{N-x}, \quad x = 0, 1, 2, \ldots, N. \end{aligned}$$

Note that the sample space for the Multinomial is $S = \{x_1, x_2, \ldots, x_k\}$ where each $x_i \geq 0$ is an integer and $x_1 + x_2 + \cdots + x_k = N$. For large k or N, this set S is "large".

In the so-called "χ^2 goodness-of-fit problem", we are interested in testing the hypothesis

$$\begin{aligned} H_0 : &\quad \theta_1 = \theta_{01}; \theta_2 = \theta_{02}; \ldots; \theta_k = \theta_{0k}; \\ H_A : &\quad \text{at least one } \theta_i \neq \theta_{0i}, \end{aligned}$$

where $\theta_{01}, \theta_{02}, \ldots, \theta_{0k}$ are specified constants. A standard approach is to consider the statistic

$$t(\underset{\sim}{x}|\underset{\sim}{\theta}_0) \equiv \sum_{i=1}^{k} \frac{(x_i - N\theta_{0i})^2}{N\theta_{0i}},$$

which is the usual Pearson χ^2-statistic, $\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$ where O_i is the observed frequency and E_i is the expected frequency. If H_0 is true, and N is large, then

$$t(\underset{\sim}{x}|\underset{\sim}{\theta}_0) \dot{\sim} \chi^2_{k-1}.$$

So for a test with size approximately equal to, say $\alpha = 0.05$, we reject H_0 if

$$t(\underset{\sim}{x}|\underset{\sim}{\theta}_0) \;\geq\; \chi^2_{k-1,0.95}.$$

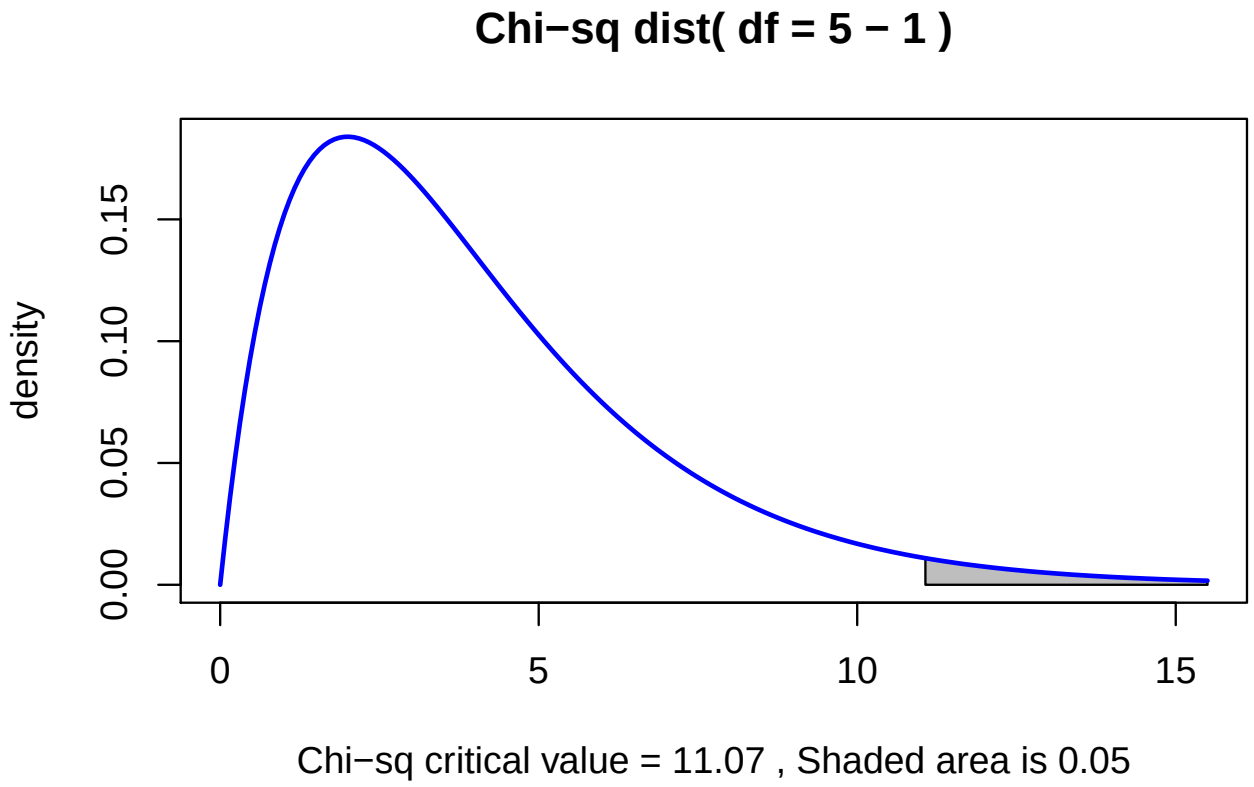
If H_0 is true,

$$\mathrm{E}_{\underset{\sim}{\theta}_0}[g(\underset{\sim}{X}|\underset{\sim}{\theta}_0)] \;=\; \Pr_{\underset{\sim}{\theta}_0}[t(\underset{\sim}{x}|\underset{\sim}{\theta}_0) \geq \chi^2_{k-1,0.95}] \;\doteq\; \alpha \;=\; 0.05.$$

```
# Plot Chi-sq distribution with shaded 0.05 right tail
par(mfrow=c(1,1))
k <- 5
chi2.95 <- qchisq(0.95, k)
lim.lower <- 0;
lim.upper <- chi2.95 * 1.4;
x.curve <- seq(lim.lower, lim.upper, length=200)
y.curve <- dchisq(x.curve, df = k - 1)
# set up plotting area
plot(x.curve, y.curve, type = "n"
    , ylab = "density"
    , xlab = paste("Chi-sq critical value =", signif(chi2.95, 5)
                , ", Shaded area is 0.05")
    , main = paste("Chi-sq dist( df =", k, "- 1 )")
    )

# plot shaded region
x.pval.u <- seq(chi2.95, lim.upper, length=200)
y.pval.u <- dchisq(x.pval.u, df = k-1)
polygon(c(chi2.95, x.pval.u, lim.upper)
      , c(0, y.pval.u, 0), col="gray")

# plot curve last so it covers shaded region
points(x.curve, y.curve, type = "l", lwd = 2, col = "blue")
```



Question: How close is $\mathrm{E}_{\underset{\sim}{\theta}_0}[g(\underset{\sim}{X}|\underset{\sim}{\theta}_0)]$ to $\alpha = 0.05$?
Calculating this expectation exactly is challenging.

Example: Simple linear regression Suppose we have a simple linear regression model

$$Y_i \;=\; \beta_0 + \beta_1 x_i + \varepsilon_i, \qquad i = 1, \ldots, n.$$

Let $\hat{\underset{\sim}{\beta}} = [\hat{\beta}_0, \hat{\beta}_1]^\top$ be the LS estimate of $\underset{\sim}{\beta} = [\beta_0, \beta_1]^\top$ and recall that the estimated slope is given by

$$\hat{\beta}_1 \;=\; \frac{\sum_{i=1}^n (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^n (x_i - \bar{x})^2}.$$

Suppose we are interested in testing $H_0 : \beta_1 = 0$. A usual approach is to establish the "t-statistic"

$$T \;=\; \frac{\hat{\beta}_1 - 0}{\mathrm{SE}(\hat{\beta}_1)} \;=\; \frac{\hat{\beta}_1}{\mathrm{SE}(\hat{\beta}_1)}$$

where

$$\mathrm{SE}(\hat{\beta}_1) \;=\; \frac{\hat{\sigma}^2}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

and

$$\hat{\sigma}^2 \;=\; \frac{1}{n-2} \sum_{i=1}^n \Big\{ Y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i) \Big\}^2 .$$

It is well known that when H_0 is true, that $T \sim t_{n-2}$ provided $\varepsilon_i \overset{\text{ind}}{\sim} \mathrm{Normal}(0, \sigma^2)$. What is the distribution of T if the errors follow a different distribution?

Noting that we typically assume x_is are fixed and Y_is are random, the distribution of $\underset{\sim}{Y} = [Y_1, \ldots, Y_n]^\top$ is obtained as a simple linear transformation of the distribution of $\underset{\sim}{\varepsilon} = [\varepsilon_1, \ldots, \varepsilon_n]^\top$. If the density of $\underset{\sim}{Y}$ is $f(y)$, then the cdf of $T \equiv f(Y)$ is

$$\Pr_{\underset{\sim}{\beta}, \sigma^2}(t(Y) \le c) \;=\; \mathrm{E}_{\underset{\sim}{\beta}, \sigma^2} g(Y) \;=\; \int_{\mathbb{R}^n} g(y) f(y) \, \mathrm{d}y$$

where

$$g(y) \;=\; 1_{t(Y) \le c}.$$

This probability needs to be evaluated for all possible c to give the CDF of T. Note that the density of the εs can be arbitrarily complex — it

could involve dependence among the ε_is, the ε_is could be nonnormal and heavy-tailed, etc. If n is large there is probably no hope of computing this expression exactly, except for very special cases.

These examples were meant to convey the complexity of many practical issues or questions concerning the behavior of statistical procedures, and that many problems revolve around the evaluation of expectations.

1.2 Approaches to evaluate expectations

- analytical
- numerical (approximations)
- stochastic (Monte Carlo methods)

An exact **analytical** answer is almost always best. For low-dimensional problems where x is continuous, **numerical integration** is a natural approach in many problems. A **stochastic**, or simulation-based, approach is often needed when other methods fail, for example high-dimensional intervals are difficult to accurately approximate numerically, so probabilistic methods are used instead.

Simple stochastic or Monte Carlo methods are based on the SLLN (strong law of large numbers). Suppose $X_1, \ldots, X_n$ are iid with the same distribution as $\underset{\sim}{X} = [X_1, \ldots, X_n]^\top$, that is, $X_i \sim f(x_i, \theta)$, using notation from page 1. Then, with

$$\mu \equiv \mathrm{E}_\theta[g(X)]$$

we can use the approximation

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} g(X_i).$$

By the SLLN,

$$\hat{\mu} \;\rightarrow\; \mu \;\equiv\; \mathrm{E}_\theta[g(X)] \quad \text{as} \quad n \rightarrow \infty.$$

Assuming $\mathrm{E}_\theta[g^2(X)]$ is finite, the uncertainty of error in $\hat{\mu}$ can be quantified via

$$\mathrm{Var}(\hat{\mu}) \;=\; \frac{1}{n}\mathrm{Var}_\theta g(X)$$

which can be estimated with

$$\widehat{\mathrm{Var}}(\hat{\mu}) \;=\; \frac{1}{n}\left\{\frac{1}{n-1}\sum_{i=1}^{n}(g(X_i)-\hat{\mu})^2\right\},$$

where the term inside the braces is the sample variance of $g(X_i)$s. Typically,

$$\hat{\mu} \;\dot{\sim}\; \text{Normal}\left(\mu, \widehat{\mathrm{Var}}(\hat{\mu})\right),$$

that is, the sampling distribution of our Monte Carlo estimate of μ is approximately normal.

The success of this method revolves around

1. being able to sample (generate) vectors $\underset{\sim}{X} = [X_1, \ldots, X_n]^\top$ from distribution $f(\underset{\sim}{x}|\underset{\sim}{\theta}_0)$ easily and accurately, and

2. the uncertainty in $\hat{\mu}$ being minimized.

I will present some discussion of each issue.

1.2.1 Random number generation

Most algorithms for generating random samples from a probability distribution originate from and era where the focus was on transforming

uniform random numbers to give the desired distribution. The reasons for this will be outlined below. Because any (most?) method for generating random uniforms is algorithmic, such observations cannot be truly random, but rather "pseudo-random numbers". A good algorithm for generating random samples from a specific distribution should be able to pass any statistical test that the samples being generated are iid from the distribution (well, any test that is not specifically designed knowing the actual algorithm being used).

Generating uniform random variables

We should be familiar with the uniform density on the interval $[0, 1]$. The density Uniform$(0, 1)$ is $f(x) = 1$ where $0 \leq x \leq 1$, and 0 otherwise.

A standard way to generate pseudo-random uniform rvs is to start with an initial value x_0, called the "seed", and then recursively compute

$$x_n \;=\; \{ax_{n-1} + b\} \mod m$$

where a, b, and m are integers. That is, x_n is the remainder from dividing $(ax_{n-1} + b)$ by m. The pseudo-random numbers correspond to x_n/m. This is called a **mixed-congruential** generator (it has additive and multiplicative components).

One often chooses m equal to the computer's word length, because that makes modular arithmetic efficient. For a 32-bit word machine (where the first bit is a sign bit), it has been shown that the linear (that is, $b = 0$) generator with $m = 2^{31} - 1$ and $a = 7^5 = 16807$ "works well".

In R , searching for help on `.Random.seed` will provide information on the algorithms available for random number generation. By default, R sets the seed based on the clock time. Alternatively, you can specify your own seed, which is useful in debugging code (because the same samples are generated, you can focus debugging efforts on the remaining code).

Direct methods

Any discrete rv X can be generated from a uniform distribution. Suppose X is a scalar rv with

$$\Pr(X = x_j) \;=\; p_j, \qquad j = 0, 1, 2, \ldots,$$

where $\sum_j p_j = 1$, and let $U \sim \text{Uniform}(0, 1)$. If we set

$$X \;=\; \begin{cases} x_0 & \text{if } \; U \le p_0 \\ x_1 & \text{if } \; p_0 < U \le p_0 + p_1 \\ \vdots & \\ x_i & \text{if } \; \sum_{j=0}^{i-1} p_j < U \le \sum_{j=0}^{i} p_j \\ \vdots & \end{cases}$$

then

$$\begin{aligned} \Pr(X = x_0) \;&=\; \Pr(U \le p_0) \;=\; p_0 \\ \Pr(X = x_1) \;&=\; \Pr(p_0 < U \le p_0 + p_1) \;=\; (p_0 + p_1) - p_0 \;=\; p_1 \\ &\;\;\vdots \\ \Pr(X = x_i) \;&=\; \Pr\left(\sum_{j=0}^{i-1} p_j < U \le \sum_{j=0}^{i} p_j \right) \;=\; p_i \\ &\;\;\vdots \end{aligned}$$

That is, X has the desired distribution.

All we're doing here is partitioning $[0, 1]$ into intervals of length p_j, $j = 0, 1, 2, \ldots$, and seeing into which interval U falls.

In the example below $\Pr(X = 0.1i)$, $i = 1, 2, 3, 4$, and the idea is easily programmed using a loop.

```
x <- 1:4          # define x values taking probabilities
p <- 0.1 * 1:4    # define probabilities
cp <- cumsum(p)  # cumulative sum of probabilities
cp

## [1] 0.1 0.3 0.6 1.0

U <- runif(1)  # draw uniform random number
ii <- 1;
while (U > cp[ii]) {
  ii <- ii + 1
}
xi <- x[ii]
c(U, xi)  # generated u and xi

## [1] 0.1715 2.0000
```

To improve efficiency, you need to minimize the number of steps in the `while()` loop. This can be done by ordering p_is from largest to smallest ("carrying along the xs") before looping. You need to be more careful if the number of values x assumes is not finite.

Example: Binomial samples Suppose you wish to generate $X_i \overset{\text{iid}}{\sim}$ Binomial(m, p), that is

$$\Pr(X_i = x) \;=\; \binom{m}{x} p^x (1-p)^{m-x}, \quad x = 0, 1, 2, \ldots, m.$$

This can also be done using the above approach. However, it is also possible to use the following characterization.

If $Y_1, Y_2, \ldots, Y_m$ are iid Bernoulli(p), then

$$X \;=\; Y_1 + Y_2 + \cdots + Y_m \;\sim\; \text{Binomial}(m, p).$$

Recall that

$$Y_i = \text{Bernoulli}(p) \Leftrightarrow \begin{array}{l} \Pr(Y_i = 1) = p \\ \Pr(Y_i = 0) = 1 - p. \end{array}$$

You can easily generate a Bernoulli(p) rv from a Uniform$(0, 1)$, that is

$$\begin{array}{rcll} Y_i & = & 1 & \text{if } U < p \\ & = & 0 & \text{else.} \end{array}$$

So generating X only requires generating m iid Uniform$(0, 1)$ rvs and a simple comparison. For example, given m and p,

```
m <- 10
p <- 0.8
U <- runif(m)  # draw m uniform random number
X <- sum( (U < p) )
X
```

```
## [1] 9
```

```
# or in one step
X <- sum(runif(m) < p)
X
```

```
## [1] 7
```

You can generate $X_1, X_2, \ldots, X_n \overset{\text{iid}}{\sim} \text{Binomial}(m, p)$ via

```
n <- 20
m <- 10
p <- 0.8
U <- matrix(runif(m*n), nrow=n)  # draw uniform random numbers
X <- apply(U < p, MARGIN=1, sum)
X
```

```
##  [1] 7 9 9 7 8 9 9 9 5 7 8 9 9 9 5 8 7 9 5 7
```

1.2.2 Inverse cdf method

The uniform distribution plays a central role when generating continuous random variables. If we have a scalar rv X with cumulative distribution function (cdf)

$$F(t) \;=\; \Pr(X \leq t), \quad -\infty < t < \infty,$$

then X has the same distribution as $F^{-1}(U)$, where $U \sim \text{Uniform}(0,1)$. We have to be a bit careful in the definition of $F^{-1}(\cdot)$, but the basic idea is that

$$\Pr(F^{-1}(U) \leq t) \;=\; \Pr(U \leq F(t)) \;=\; F(t),$$

thus, $F^{-1}(U)$ has cdf $F(t)$. That is, $X \sim F^{-1}(U)$.

This idea can be directly exploited in only a few selected distributions where $F^{-1}(\cdot)$ is available. For example, if $U \sim \text{Uniform}(0,1)$, then

$$X \;=\; -\log(U)/\lambda \;\sim\; \text{Exponential}(\lambda)$$

with density

$$f(x|\lambda) \;=\; \lambda e^{-\lambda x}, \quad x > 0.$$

Also, if $U \sim \text{Uniform}(0,1)$, then

$$X \;=\; \alpha + \beta \tan\{\pi(U - 0.5)\} \;\sim\; \text{Cauchy}(\alpha, \beta)$$

with density

$$f(x|\alpha,\beta) \;=\; \frac{1}{\pi\beta\left\{1 + \left(\frac{x-\alpha}{\beta}\right)^2\right\}}.$$

Direct methods (continuous rv)

Simple transformations of uniforms can often be used to generate random variables with specific distributions. Here are two well-known approaches to generate Normal$(0, 1)$ rvs.

Box-Muller (1958) Given $U_1, U_2 \sim$ Uniform$(0, 1)$, set

$$\begin{aligned} z_1 &= \sqrt{-2\ln(U_1)}\cos(2\pi U_2) \\ z_2 &= \sqrt{-2\ln(U_1)}\sin(2\pi U_2). \end{aligned}$$

Then z_1 and z_2 are iid Normal$(0, 1)$. This comes from the polar coordinate representation of (z_1, z_2). That is, what distribution on polar coordinates leads to normals, and how is this distribution generated from uniforms? The proof of this is by the change-of-variable formula.

The Box-Muller method is not very efficient, because of the need to evaluate the trigonometric functions.

Polar method This is more efficient than Box-Muller, and is a special case of a rejection method.

1. Generate $U_1, U_2 \sim$ Uniform$(0, 1)$
2. Set $V_i = 2U_i - 1$ and $s^2 = V_1^2 + V_2^2$
3. If $s > 1$, return to step 1.
4. Else, set $T = \sqrt{-2\ln(s)/s}$ and return $z_1 = TV_1$ and $z_2 = TV_2$

One can show that $z_1, z_2 \overset{\text{iid}}{\sim}$ Normal$(0, 1)$.

Geometric motivations for both methods can be found.

Remark If $z_i \sim \text{Normal}(0,1)$ then $az_i + b \sim \text{Normal}(b, a^2)$, so it is easy to generate arbitrary normals from Normal$(0,1)$.

Functions of normal rvs

χ^2-distribution A **chi-squared** rv with df = degrees-of-freedom = k, an integer, can be generated via

$$X \;=\; Z_1^2 + Z_2^2 + \cdots + Z_k^2 \;\sim\; \chi_k^2,$$

where $Z_1, Z_2, \ldots, Z_k$ are iid Normal$(0,1)$.

***t*-distribution** A Student's t-distribution with df=k (integer) can be generated via

$$X \;=\; \frac{Y}{\sqrt{Z/k}} \;\sim\; t(k),$$

where $Y \sim \text{Normal}(0,1)$ independent of $Z \sim \chi_k^2$.

***F*-distribution** An $F_{k,m}$ distribution can be generated via

$$X \;=\; \frac{Y/k}{Z/m} \;\sim\; F(k,m),$$

where $Y \sim \chi_k^2$ independent of $Z \sim \chi_m^2$.

1.2.3 Rejection sampling

This method is sometimes used when it is difficult to explicitly generate an $X \sim f(x)$. Suppose $h(x)$ is another density from which we know how

to sample easily. Let $e(x)$ be an envelope function, such that for a given user-specified constant α we have

$$e(x) \equiv \frac{h(x)}{\alpha} \geq f(x)$$

for all x where $f(x) > 0$. To generate $X \sim f(x)$

1. Sample $Y \sim h$

2. Sample $U \sim \text{Uniform}(0, 1)$ independently of Y

3. If $U > f(Y)/e(Y)$, then reject Y and return to step 1.

4. Else, define $X = Y$ as our sampled observation from $f(x)$.

To see that X has density $f(x)$, note

$$\begin{aligned}
\Pr(X \le x) &= \Pr\left(Y \le x \middle| U \le \frac{f(Y)}{e(Y)}\right) \\
&= \frac{\Pr\left(Y \le x \text{ and } U \le \frac{\alpha f(Y)}{h(Y)}\right)}{\Pr\left(Y \in \mathbb{R}^1 \text{ and } U \le \frac{\alpha f(Y)}{h(Y)}\right)} \\
&= \frac{\Pr\left(Y \le x \text{ and } U \le \frac{\alpha f(Y)}{h(Y)}\right)}{\Pr\left(U \le \frac{\alpha f(Y)}{h(Y)}\right)} \\
&= \frac{\left\{\int_{-\infty}^{x}\left[\int_0^{\alpha f(z)/h(z)} 1\,\mathrm{d}u\right] h(z)\,\mathrm{d}z\right\}}{\left\{\int_{-\infty}^{\infty}\left[\int_0^{\alpha f(z)/h(z)} 1\,\mathrm{d}u\right] h(z)\,\mathrm{d}z\right\}} \\
&= \frac{\int_{-\infty}^{x} \frac{\alpha f(z)}{h(z)} h(z)\,\mathrm{d}z}{\int_{-\infty}^{\infty} \frac{\alpha f(z)}{h(z)} h(z)\,\mathrm{d}z} \\
&= \frac{\int_{-\infty}^{x} f(z)\,\mathrm{d}z}{\int_{-\infty}^{\infty} f(z)\,\mathrm{d}z} \\
&= \int_{-\infty}^{x} f(z)\,\mathrm{d}z.
\end{aligned}$$

If the cdf of X is $\int_{-\infty}^{x} f(z)\,\mathrm{d}z$, then the density is

$$\frac{\mathrm{d}\Pr(X \le x)}{\mathrm{d}x} = f(x),$$

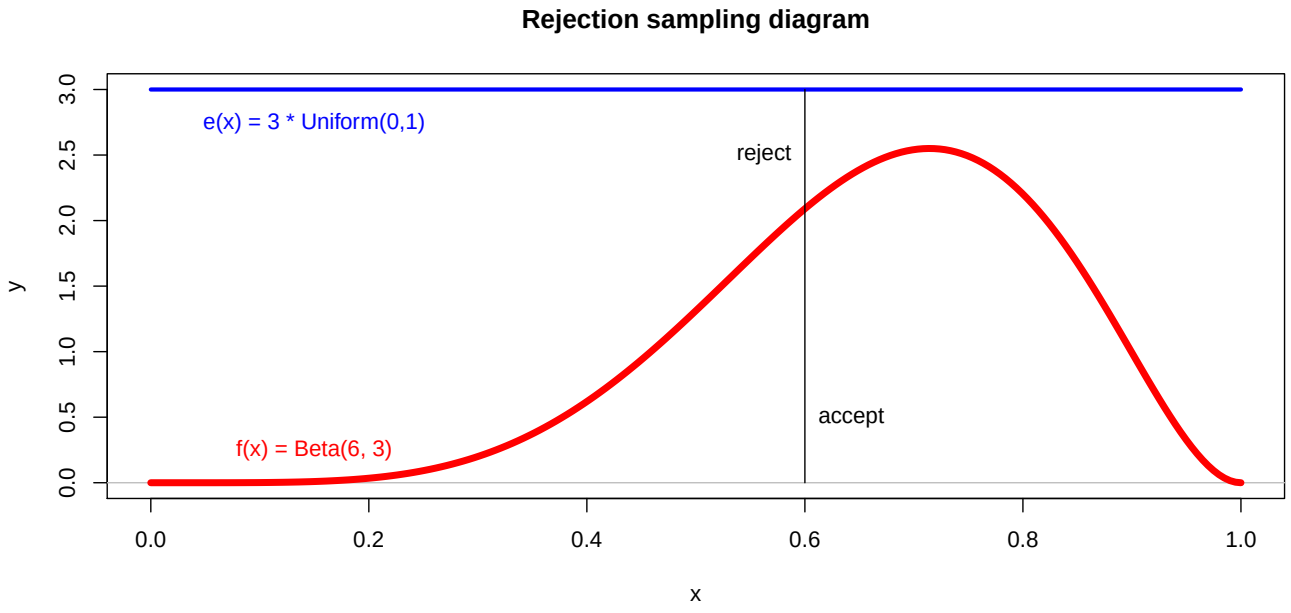
that is, $X \sim f(x)$.

Beta distribution with Uniform envelope The following image and example[1] illustrates the idea using $e(x) = 3 \times \text{Uniform}(0, 1)$ as the

[1] `http://playingwithr.blogspot.com/2011/06/rejection-sampling.html`

envelope function and $f(x) = \text{Beta}(6, 3)$ as the function of interest.

```
# Rejection sampling diagram
x <- seq(0, 1, length=2e2)
y.u <- dunif(x, 0, 1)*3
y.b <- dbeta(x, 6, 3)
plot(x, y.u, type="l", col = "blue", lwd=3, xlim=range(x), ylim=c(0,3),
      main="Rejection sampling diagram", xlab = "x", ylab = "y")
abline(h = 0, col = "gray75")
points(x, y.b, type="l", col = "red", lwd=5)
lines(x=c(0.6,0.6), y=c(0,3))
text(x=0.6, y=2.5, labels="reject", pos=2)
text(x=0.6, y=0.5, labels="accept", pos=4)
text(x=0.15, y=2.9, labels="e(x) = 3 * Uniform(0,1)", pos=1, col="blue")
text(x=0.15, y=0.1, labels="f(x) = Beta(6, 3)", pos=3, col = "red")
```



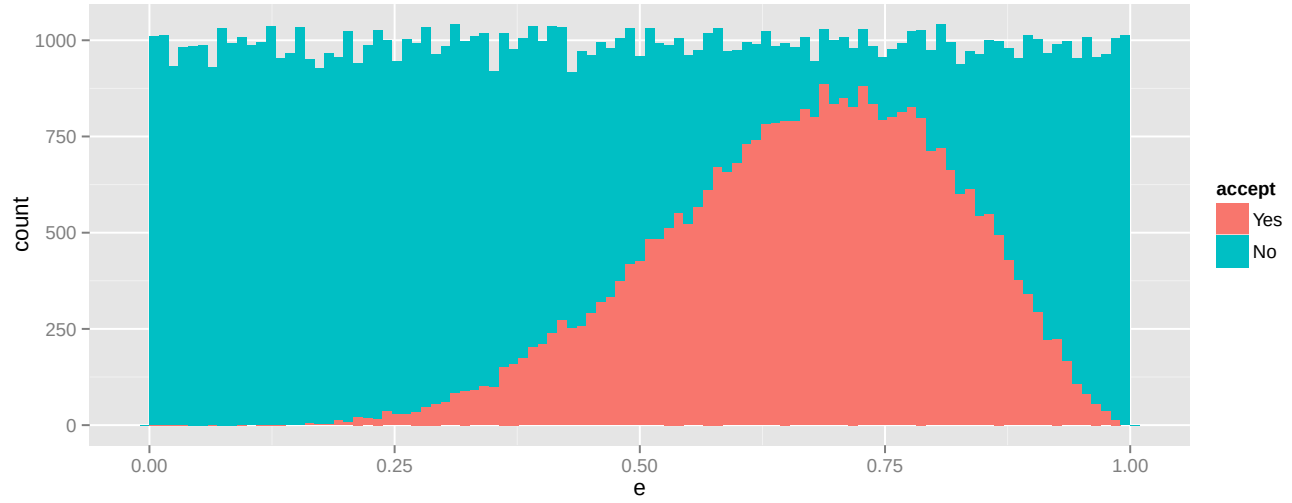

```
# data example for rejection sampling
R <- 1e5                # number of rejection samples
e <- runif(R,0,1)       # sample from enveloping function
accept <- rep("No", R)  # initialize samples as "No" accept
U <- runif(R, 0, 1)     # sample from uniform distribution

# accept if the height of the envelope is less than the height of the function
#   e(Y) * U * scale <= f(Y)
accept[ ( dunif(e, 0, 1) * U * 3 <= dbeta(e, 6, 3) ) ] <- "Yes"

# put into a data.frame for plotting
sam <- data.frame(e, accept = factor(accept, levels = c("Yes","No")))

# plot a stacked histogram
```

```
library(ggplot2)
p <- ggplot(sam, aes(x = e))
p <- p + geom_histogram(aes(fill = accept), binwidth = 1/101)
print(p)
```



The efficiency of the algorithm relates to how closely $e(x)$ envelopes $f(x)$. If $e(x) \gg f(x)$ over the entire range, then many samples $Y \sim g$ will be rejected. The scalar α should be made sufficiently large (that is, try to get $e(x)$ to "touch" $f(x)$ at some x).

Gamma distribution with Normal envelope X is a Gamma(r, λ) rv if it has density

$$f(x|r,\lambda) = \frac{\lambda^r x^{r-1} \exp\{-\lambda x\}}{\Gamma(r)}, \quad x \geq 0,$$

and 0 elsewhere.

Note that if $X^* \sim$ Gamma$(r, 1)$ then $X^*/\lambda \sim$ Gamma(r, λ). The **rejection method** is commonly used to generate Gamma$(r, 1)$ rvs from which arbitrary Gamma rvs may be obtained. The idea is the following.

Suppose Y has density

$$f(y) = \frac{t'(y)t(y)^{r-1} \exp\{-t(y)\}}{\Gamma(r)}$$

for $t(y) = a(1+by)^3$ for $-1/b < y < \infty$, $a = r - 1/3$, and $b = 1/\sqrt{9a}$. Then, $X = t(Y) \sim \text{Gamma}(r, 1)$. So, if we can generate $Y \sim f(y)$ we can then transform $X = t(Y)$ to get the desired distribution. One can show with a bit of work that for these choices of a and b,

$$\begin{aligned} f(y) &= \text{const } \exp\{a \log(t(y)/a) - t(y) + a\} \\ &= \text{const } q(y), \end{aligned}$$

and that the function

$$q(y) \le \exp\{-y^2/2\},$$

which is proportional to a Normal$(0, 1)$ density.

Thus, if we define

$$\begin{aligned} e(y) &= \text{envelope function} \\ &= \frac{1}{\sqrt{2\pi}} \exp\{-y^2/2\} \times \sqrt{2\pi}\,\text{const} \ge \text{const } q(y) \\ &= h(y)\frac{1}{\alpha} \ge f(y), \end{aligned}$$

then we can use the rejection method as follows.

1. Sample $Y \sim \text{Normal}(0, 1)$
2. Sample $U \sim \text{Uniform}(0, 1)$ independently of Y
3. If $U > f(Y)/e(Y) = q(Y)/h(Y)$, then reject Y and return to step 1.
4. Else, define $X = t(Y)$ as our sampled observation from $f(x)$.

Can you implement this as I did the Beta/Uniform example before?

Remark It is important to note that we did not need to know the value of the constant α here. We only needed to know the kernel of the density $f(y)$. This suggests that this method is useful for situations where the proportionality constant for a density is unknown. This is especially important for Bayesian applications where the posterior density is typically known only up to the constant of proportionality.

Beta samples X is a Beta(α, β) rv if it has density

$$f(x|\alpha, \beta) \;=\; \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}, \quad 0 \leq x \leq 1,$$

and 0 elsewhere, where $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$. Note that if z_1 and z_2 are independent Gamma$(\alpha, 1)$ and Gamma$(\beta, 1)$ rvs, then

$$X = \frac{z_1}{z_1 + z_2} \;\sim\; \text{Beta}(\alpha, \beta).$$

This provides a straightforward means to generate Beta rvs from Gamma rvs.