Stats 102C - Lecture 3-3: Rejection Sampling

Miles Chen, acknowledgements Michael Tsiang

Week 3 Friday

Section 1

Rejection Sampling

Rejection Sampling

Rejection sampling is another method for generating random values from a distribution.

We might use rejection sampling when R does not have a built in function and when we can't use inverse CDF method because $F(x)$ or $F^{-1}(x)$ can't be derived.

Rejection Sampling

Goal: generate a random sample from f. We know the PDF of f.

Problem: We don't know $F^{-1}(x)$. We can't generate directly from f.

Requirements:

- Find a candidate distribution $g(x)$ that we can sample from.
- The support $(\mathcal{X})$ of f must be fully contained inside the support of g.
- There is a constant M such that $Mg(x) \geq f(x)$ for all $x \in \mathcal{X}$.

Because we require $Mg(x) \geq f(x)$, the most efficient choice of M is max of $\frac{f(x)}{g(x)}$ (and nothing larger). Anything less than the max of $\frac{f(x)}{g(x)}$ will not satisfy the condition $Mg(x) \geq f(x)$

Rejection Sampling

Algorithm:

1. Generate $X \sim g(x)$
2. Calculate

$$r(X) = \frac{f(X)}{Mg(X)}$$

3. Generate $U \sim \text{Unif}(0, 1)$
4. If $U \leq r(X)$ accept X as a sample from $f(X)$. Otherwise, reject X and repeat.

Example: Beta(2,2)

The Beta(2,2) distribution has PDF:

$$f(x) = 6x(1-x), \text{ for } x \in [0,1]$$

We can sample directly using `rbeta` and we can also use inverse CDF. But we will use rejection sampling.

For our candidate distribution, we will use Unif(0,1), which has PDF

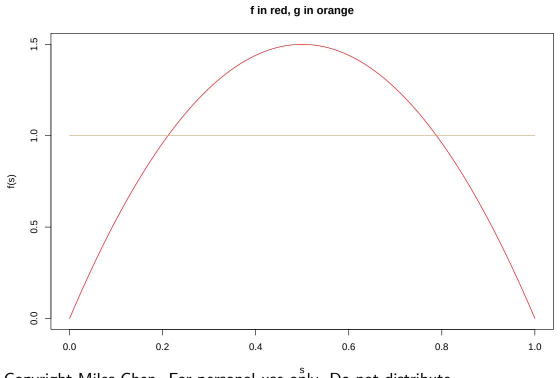
$$g(x) = 1, \text{ for } x \in [0,1]$$

and support compatible wtih f.

Example: Beta(2,2)

We must find a constant M such that $Mg(x) \geq f(x)$ for all $x \in \mathcal{X}$.

```
f <- function(x){6 * x * (1 - x)}
g <- function(x){rep(1, length(x))} # rep creates a vector of constants so I can use it in plot()
s <- seq(0, 1, by = 0.001)
plot(s, f(s), type = "l", col = "red", main = "f in red, g in orange")
lines(s, g(s), col = "orange")
```



Example: Beta(2,2)

The most efficient choice of M is max of $\frac{f(x)}{g(x)}$

$$f(x) = 6x(1-x), \text{ for } x \in [0,1]$$

$$g(x) = 1, \text{ for } x \in [0,1]$$

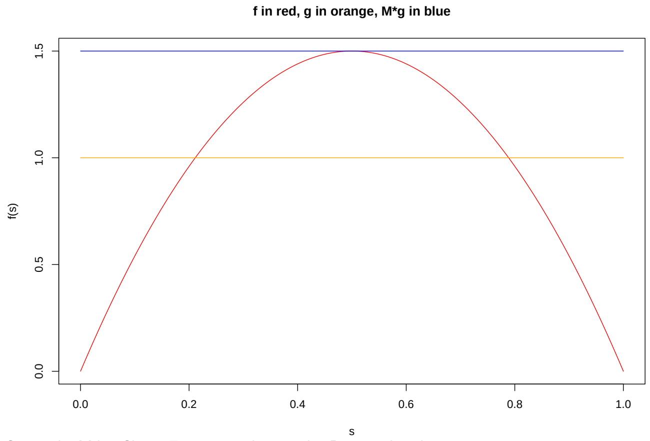
We search for the max of $f(x)/g(x)$

$$\begin{aligned}\frac{f(x)}{g(x)} &= 6x(1-x)/1 \\ 0 &= \frac{d}{dx}6x - 6x^2 \\ &= 6 - 12x \\ x &= 0.5\end{aligned}$$

Maximum occurs at $x = 0.5$. Maximum $= 6(0.5)(1-0.5) = 1.5$. Most efficient choice: $M = 1.5$

Example: Beta(2,2)

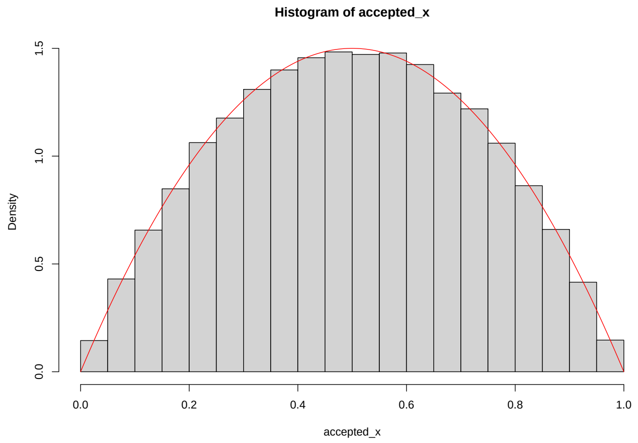
```r
M <- 1.5
plot(s, f(s), type = "l", col = "red", main = "f in red, g in orange, M*g in blue")
lines(s, g(s), col = "orange")
lines(s, M*g(s), col = "blue")
```



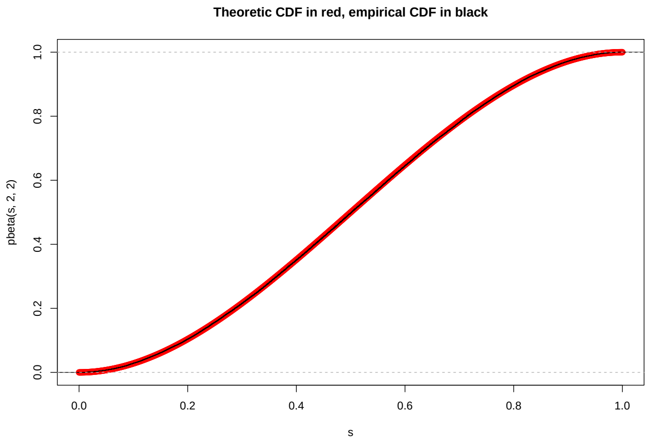
Run the algorithm

```r
# f <- function(x){6 * x * (1 - x)}
# g <- function(x){rep(1, length(x))}
set.seed(1)
N <- 10 ^ 5
proposed_x <- runif(N) # proposal distribution is Unif(0,1)
r_x <- f(proposed_x) / (M * g(proposed_x)) # equivalent to f(x) / 1.5
U <- runif(N)
accepted <- U < r_x
accepted_x <- proposed_x[accepted]
```

```r
hist(accepted_x, breaks = 30, freq = FALSE)
lines(s, dbeta(s, 2, 2), col = "red")
```

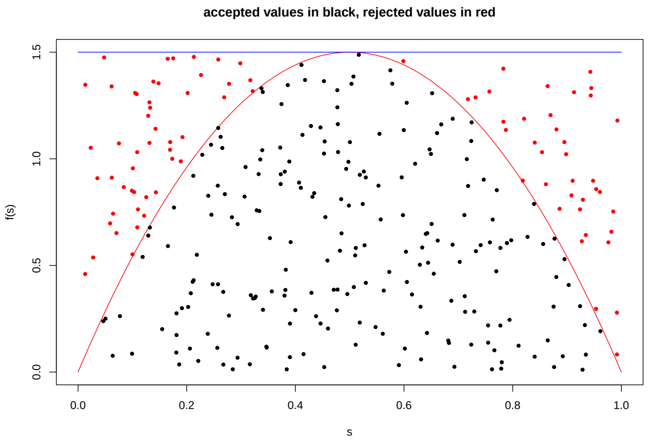


```r
plot(s, pbeta(s, 2, 2), col = "red", lwd = 2, main = "Theoretic CDF in red, empirical CDF in black")
plot(ecdf(accepted_x), add = TRUE)
```



The empirical CDF matches the theoretic CDF.

```r
M <- 1.5
plot(s, f(s), type = "l", col = "red", main = "accepted values in black, rejected values in red")
lines(s, M * g(s), col = "blue")
points(proposed_x[1:300], U[1:300] * M, pch = 20, col = c("red","black")[accepted + 1])
```



Theoretic and Empirical Acceptance rate

The empirical acceptance rate is how many values we accepted divided by how many we proposed.

```r
length(accepted_x)/length(proposed_x)
```

```
## [1] 0.66658
```

The theoretic acceptance rate is $1/M$. The f and g are both PDFs. The integral of $f(x)$ and $g(x) = 1$. The integral of $Mg(x) = M$. The proportion of $Mg(x)$ that is covered by $f(x)$ is $1/M$.

```r
1/M
```

```
## [1] 0.6666667
```

Another Example

Suppose we have the following PDF:

$$f(x) = \sin(x), \text{ for } x \in [0, \pi/2]$$

For our candidate distribution, we will use Unif$(0, \pi/2)$, which has PDF

$$g(x) = 2/\pi, \text{ for } x \in [0, \pi/2]$$

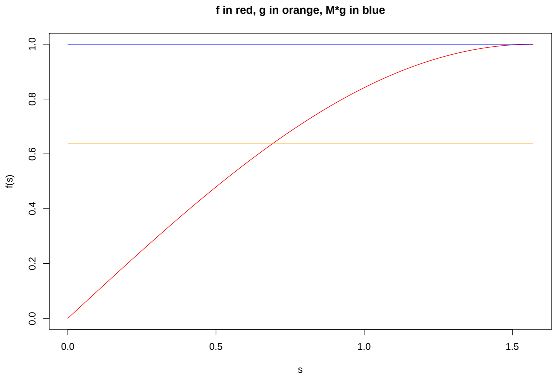
and support compatible with f.

We must find a constant M such that $Mg(x) \geq f(x)$ for all $x \in \mathcal{X}$. The most efficient choice of M is max of $\frac{f(x)}{g(x)}$

$M = \pi/2$

Example: Sin(x)

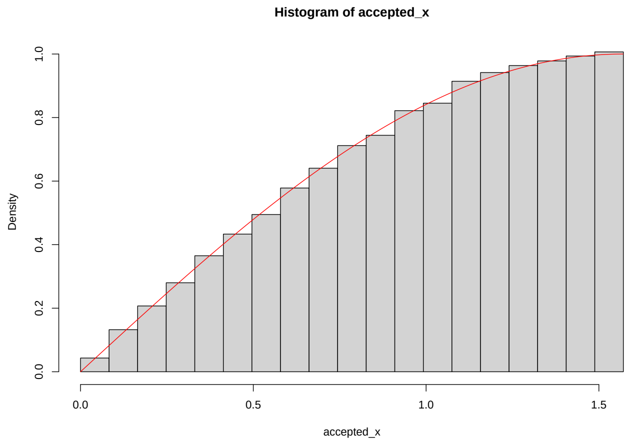
```r
f <- function(x){ sin(x) }
g <- function(x){ rep(2/pi, length(x)) }
s <- seq(0, pi/2, by = 0.001)
M <- pi/2
plot(s, f(s), type = "l", col = "red", main = "f in red, g in orange, M*g in blue")
lines(s, g(s), col = "orange")
lines(s, M*g(s), col = "blue")
```



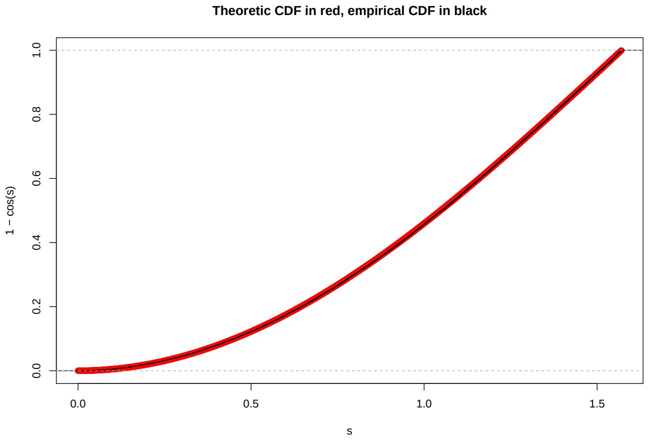
Run the algorithm

```r
# f <- function(x){ sin(x) }
# g <- function(x){ rep(2/pi, length(x)) }
set.seed(1)
N <- 10^5
proposed_x <- runif(N, 0, pi/2) # proposal distribution is Unif(0,pi/2)
r_x <- f(proposed_x)/(M*g(proposed_x)) # equivalent to f(x) / 1
U <- runif(N)
accepted <- U < r_x
accepted_x <- proposed_x[accepted]
```

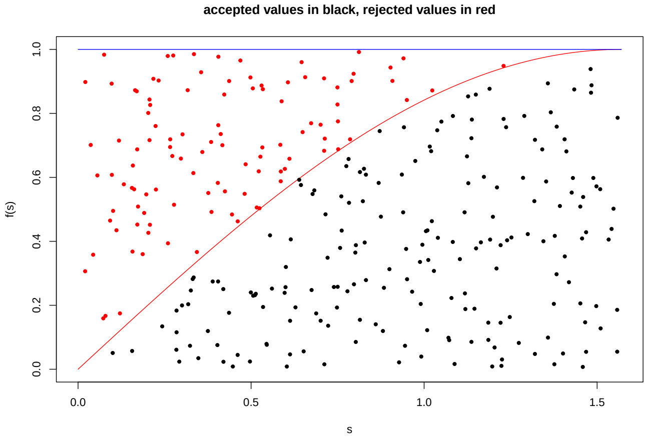
```r
hist(accepted_x, breaks = seq(0, pi/2, length.out = 20), freq = FALSE)
lines(s, f(s), col = "red")
```



```r
plot(s, 1-cos(s), col = "red", lwd = 2, main = "Theoretic CDF in red, empirical CDF in black")
plot(ecdf(accepted_x), add = TRUE)
```



```r
M <- pi/2
plot(s, f(s), type = "l", col = "red", main = "accepted values in black, rejected values in red")
lines(s, M*g(s), col = "blue")
points(proposed_x[1:300], U[1:300], pch = 20,
       col = c("red","black")[accepted + 1])
```



Example: Folded Normal

Suppose we have the PDF of the folded normal:

$$f(x) = 2 \cdot \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \text{ for } x \geq 0$$

For our candidate distribution, we will use Exponential(1) distribution, which has PDF

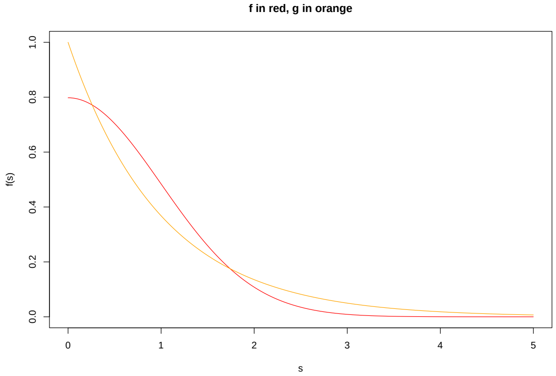
$$g(x) = e^{-x}, \text{ for } x \geq 0$$

The support is compatible with f.

Folded Normal

We must find a constant M such that $Mg(x) \geq f(x)$ for all $x \in \mathcal{X}$. The most efficient choice of M is max of $\frac{f(x)}{g(x)}$

```
f <- function(x){2 * dnorm(x)}
g <- function(x){dexp(x)}
s <- seq(0, 5, by = 0.005)
plot(s, f(s), type = "l", col = "red", main = "f in red, g in orange", ylim = c(0, 1))
lines(s, g(s), col = "orange")
```



Folded Normal

We must find a constant M such that $Mg(x) \geq f(x)$ for all $x \in \mathcal{X}$. The most efficient choice of M is max of $\frac{f(x)}{g(x)}$

$$
\begin{aligned}
M &= \max \frac{f(x)}{g(x)} \\
&= \max \frac{\frac{2}{\sqrt{2\pi}} e^{-x^2/2}}{e^{-x}} \\
&= \max \frac{2}{\sqrt{2\pi}} e^{-(x^2/2 - x)}
\end{aligned}
$$

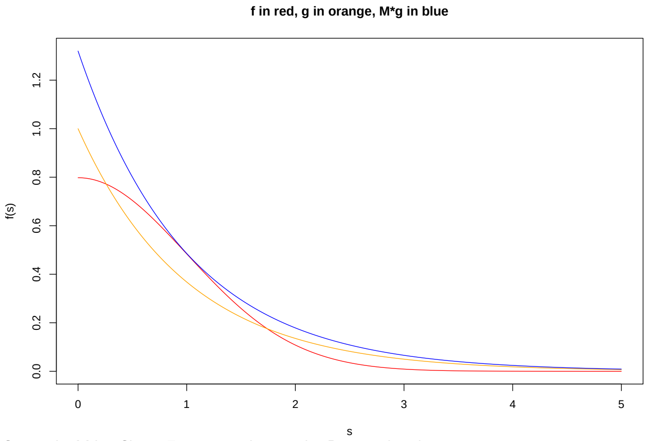
The max of $e^{-(x^2/2-x)}$ occurs at the minimum of $(x^2/2 - x)$. With a little calculus, we can see the minimum occurs at $x = 1$.

We choose M to be equal to the function at $x = 1$:

$$M = \frac{2}{\sqrt{2\pi}} e^{-(1/2-1)} = \frac{2}{\sqrt{2\pi}} e^{1/2} \approx 1.32$$

Example

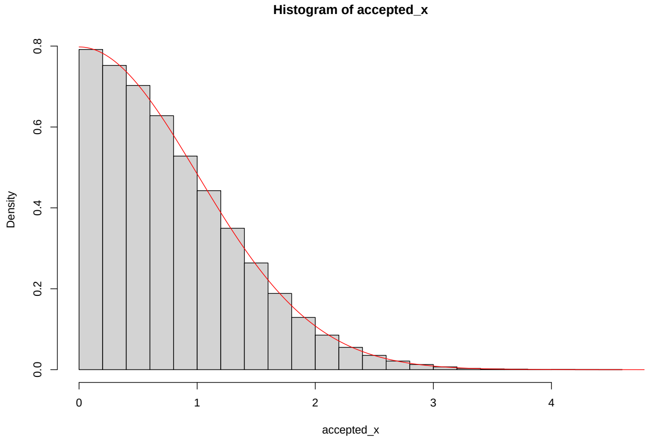
```r
M <- 1.32
plot(s, f(s), type = "l", col = "red", main = "f in red, g in orange, M*g in blue", ylim = c(0, 1.32))
lines(s, g(s), col = "orange")
lines(s, M*g(s), col = "blue")
```



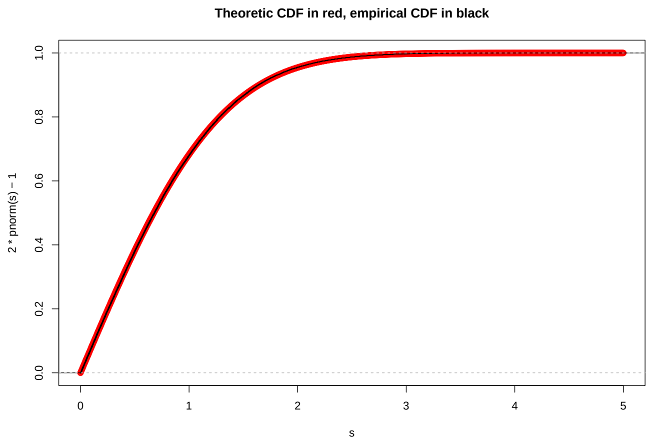
Run the algorithm

```
# f <- function(x){2 * dnorm(x)} # 2 * normal PDF
# g <- function(x){dexp(x)} # exponential PDF
set.seed(1)
N <- 10^5
proposed_x <- rexp(N) # proposal is the exponential dist
r_x <- f(proposed_x)/(M*g(proposed_x))
U <- runif(N)
accepted <- U < r_x
accepted_x <- proposed_x[accepted]
```

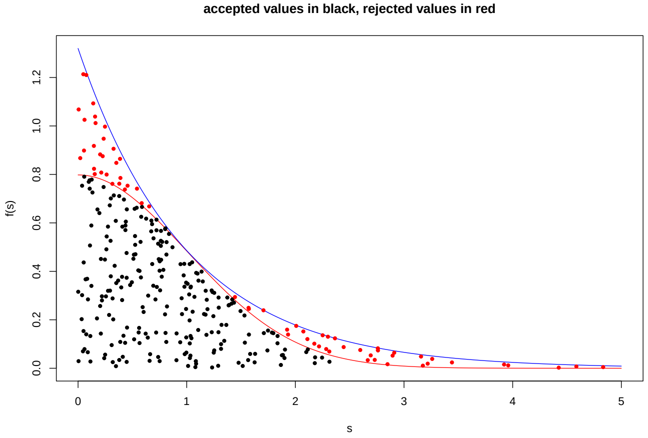
```r
hist(accepted_x, breaks = 30, freq = FALSE)
lines(s, f(s), col = "red")
```



```r
plot(s, 2 * pnorm(s) - 1, col = "red", lwd = 2, main = "Theoretic CDF in red, empirical CDF in black")
plot(ecdf(accepted_x), add = TRUE)
```



```r
plot(s, f(s), type = "l", col = "red", main = "accepted values in black, rejected values in red",
     ylim = c(0, M))
lines(s, M*g(s), col = "blue")
points(proposed_x[1:300], U[1:300] * M * g(proposed_x[1:300]), pch = 20,
       col = c("red","black")[accepted + 1])
```


accepted values in black, rejected values in red
f(s)
s