Lab 5: Rejection sampling

Olivier Binette

28/08/2020

Agenda

1. Rejection sampling
2. Lab 5 Tasks 1-3
3. Questions / Office Hours

Rejection sampling

Rejection sampling

The problem: You want to sample from a distribution on $\mathbb{R}$ given either:

1. its probability density function $p(\theta)$; or
2. some function $f(\theta) \propto p(\theta)$.

Example:

- In a Bayesian framework, we want to sample from the posterior distribution $p(\theta \mid x) \propto p(x \mid \theta)p(\theta) = f(\theta)$.
- You might want call `rnorm` and `rgamma`, use a parametric bootstrap, approximate the p-value corresponding to a complex null hypothesis, etc.

Rejection sampling

The problem: You want to sample from a distribution on $\mathbb{R}$ given either:

1. its probability density function $p(\theta)$; or
2. some function $f(\theta) \propto p(\theta)$.

Example:

- In a Bayesian framework, we want to sample from the posterior distribution $p(\theta \mid x) \propto p(x \mid \theta)p(\theta) = f(\theta)$.
- You might want call `rnorm` and `rgamma`, use a parametric bootstrap, approximate the p-value corresponding to a complex null hypothesis, etc.

Rejection sampling

The problem: You want to sample from a distribution on $\mathbb{R}$ given either:

1. its probability density function $p(\theta)$; or
2. some function $f(\theta) \propto p(\theta)$.

Example:

- In a Bayesian framework, we want to sample from the posterior distribution $p(\theta \mid x) \propto p(x \mid \theta)p(\theta) = f(\theta)$.
- You might want call `rnorm` and `rgamma`, use a parametric bootstrap, approximate the p-value corresponding to a complex null hypothesis, etc.

Rejection sampling

The problem: You want to sample from a distribution on $\mathbb{R}$ given either:

1. its probability density function $p(\theta)$; or
2. some function $f(\theta) \propto p(\theta)$.

Example:

- In a Bayesian framework, we want to sample from the posterior distribution $p(\theta \mid x) \propto p(x \mid \theta)p(\theta) = f(\theta)$.
- You might want call `rnorm` and `rgamma`, use a parametric bootstrap, approximate the p-value corresponding to a complex null hypothesis, etc.

Rejection sampling

The problem: You want to sample from a distribution on $\mathbb{R}$ given either:

1. its probability density function $p(\theta)$; or
2. some function $f(\theta) \propto p(\theta)$.

Example:

- In a Bayesian framework, we want to sample from the posterior distribution $p(\theta \mid x) \propto p(x \mid \theta)p(\theta) = f(\theta)$.
- You might want call `rnorm` and `rgamma`, use a parametric bootstrap, approximate the p-value corresponding to a complex null hypothesis, etc.

Rejection sampling

Summary:

- Sampling from probability distributions is a fundamental problem in statistics and computer science.
- Often you only have access to the probability density function or something proportional to it.
- Rejection sampling is *one* way to address this.

Rejection sampling

Summary:

- Sampling from probability distributions is a fundamental problem in statistics and computer science.
- Often you only have access to the probability density function or something proportional to it.
- Rejection sampling is *one* way to address this.

Rejection sampling

Summary:

- Sampling from probability distributions is a fundamental problem in statistics and computer science.
- Often you only have access to the probability density function or something proportional to it.
- Rejection sampling is *one* way to address this.

Rejection sampling

The area under the graph of a function f is the set of points (x, y) such that $0 \leq y \leq f(x)$.

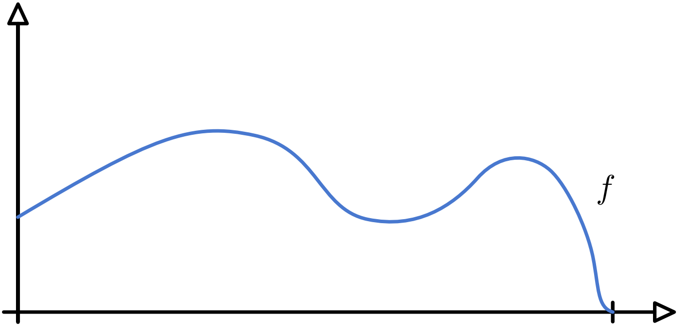
Fundamental lemma of rejection sampling: Let f be a positive and integrable function. If (X, Y) is uniformly distributed under the graph of f, then the marginal probability density function of X is proportional to f.

Rejection sampling

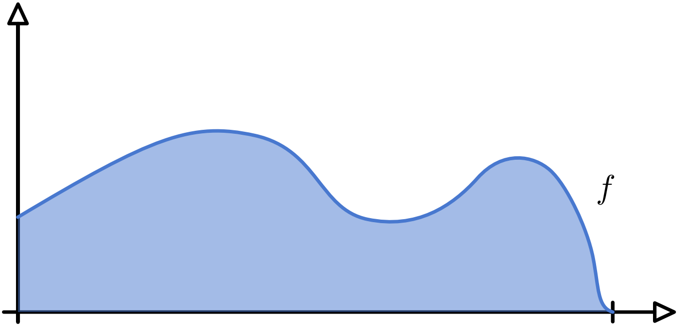
The area under the graph of a function f is the set of points (x, y) such that $0 \leq y \leq f(x)$.

Fundamental lemma of rejection sampling: Let f be a positive and integrable function. If (X, Y) is uniformly distributed under the graph of f, then the marginal probability density function of X is proportional to f.

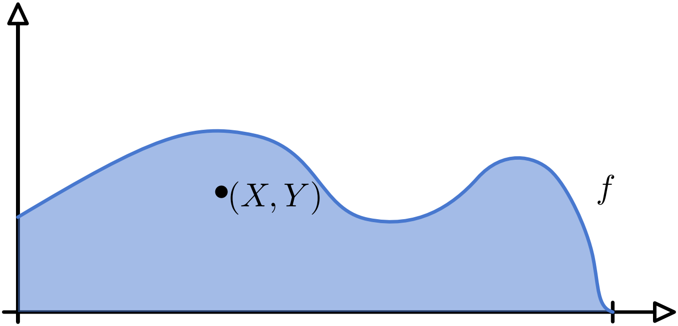
Rejection sampling



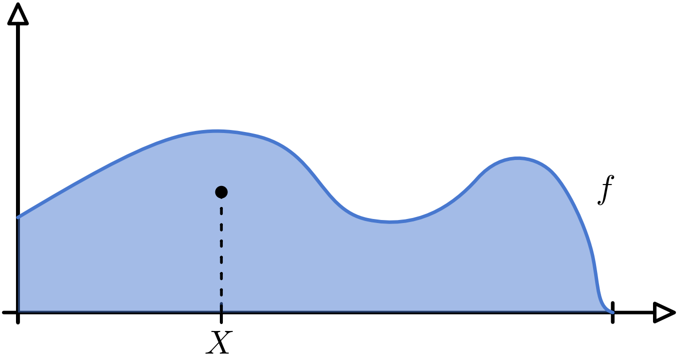
Rejection sampling



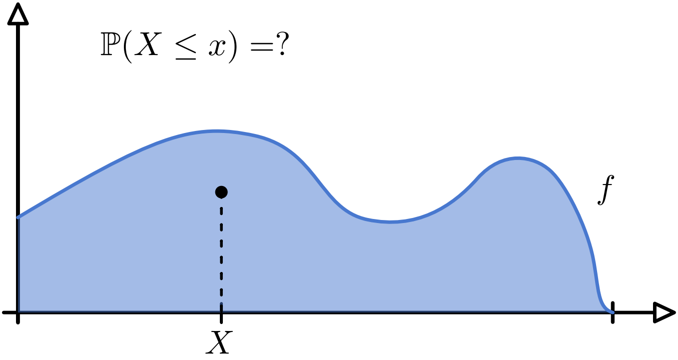
Rejection sampling



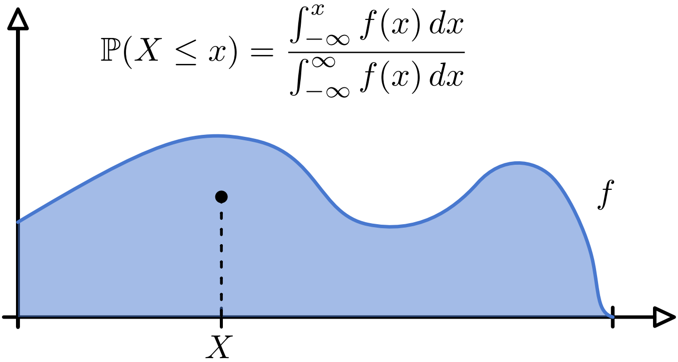
Rejection sampling



Rejection sampling

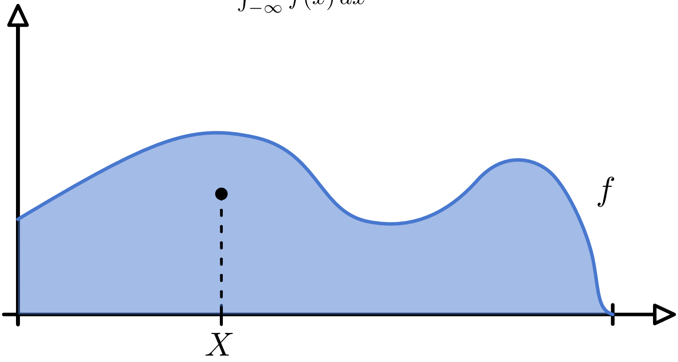


Rejection sampling



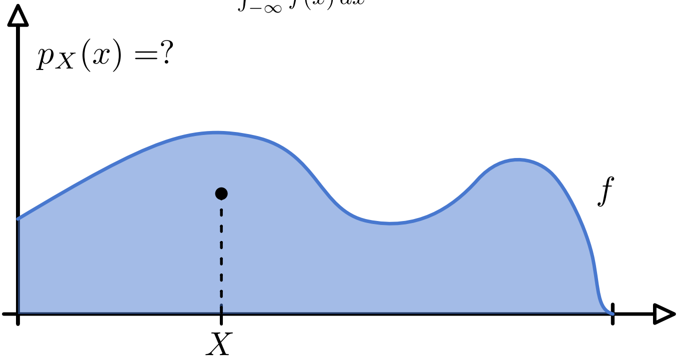
Rejection sampling

$$\mathbb{P}(X \le x) = \frac{\int_{-\infty}^{x} f(x)\,dx}{\int_{-\infty}^{\infty} f(x)\,dx}$$



Rejection sampling

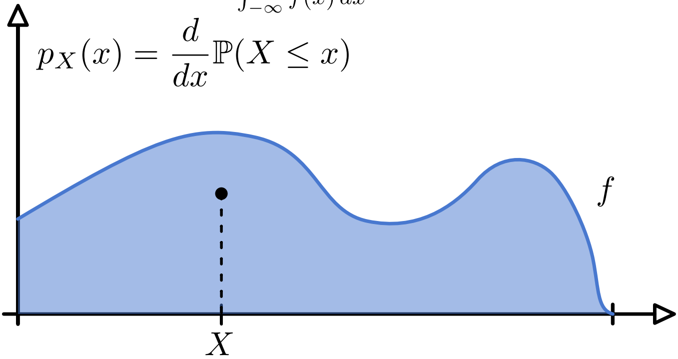
$$\mathbb{P}(X \leq x) = \frac{\int_{-\infty}^{x} f(x)\,dx}{\int_{-\infty}^{\infty} f(x)\,dx}$$



Rejection sampling

$$\mathbb{P}(X \leq x) = \frac{\int_{-\infty}^{x} f(x)\,dx}{\int_{-\infty}^{\infty} f(x)\,dx}$$

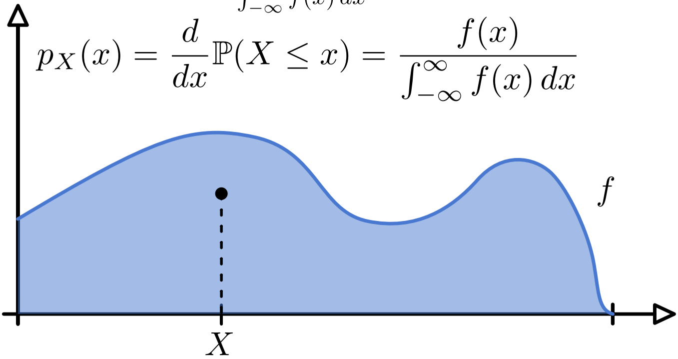
$$p_X(x) = \frac{d}{dx}\mathbb{P}(X \leq x)$$



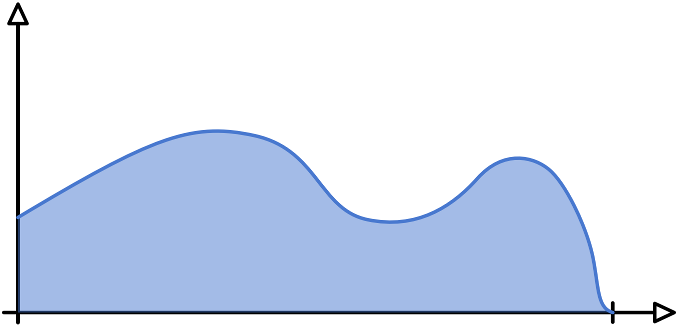
Rejection sampling

$$\mathbb{P}(X \leq x) = \frac{\int_{-\infty}^{x} f(x)\,dx}{\int_{-\infty}^{\infty} f(x)\,dx}$$

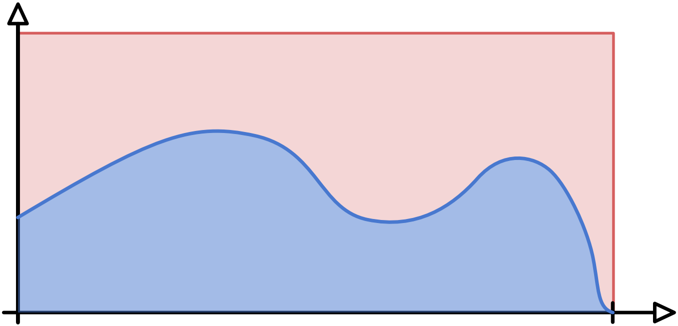
$$p_X(x) = \frac{d}{dx}\mathbb{P}(X \leq x) = \frac{f(x)}{\int_{-\infty}^{\infty} f(x)\,dx}$$



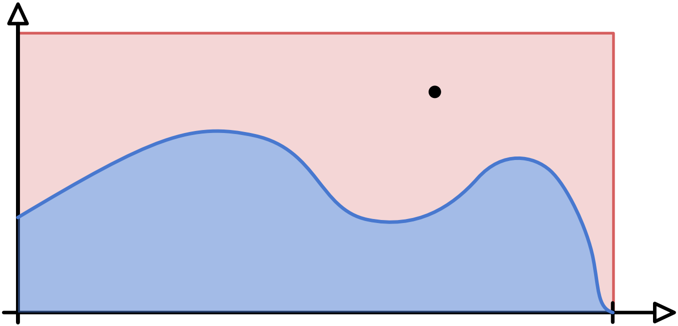
Rejection sampling



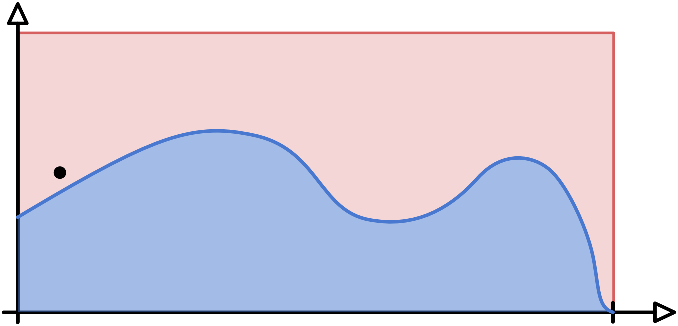
Rejection sampling



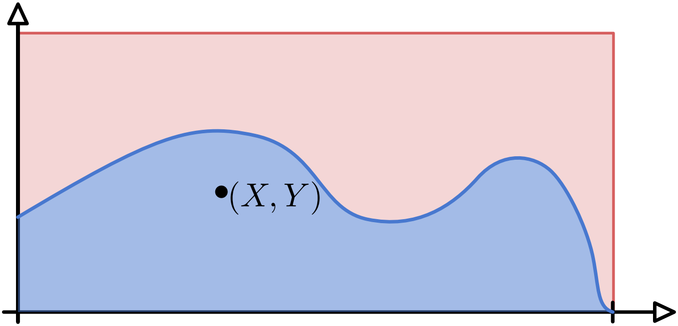
Rejection sampling



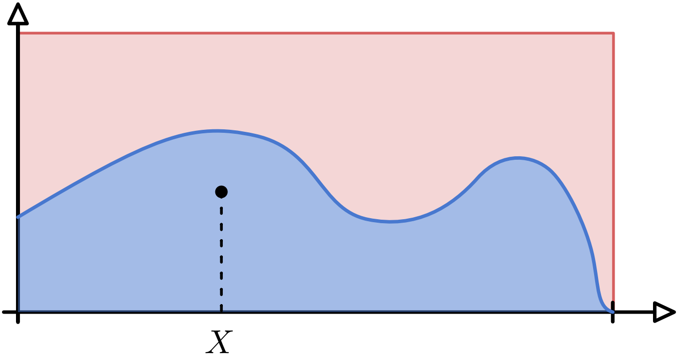
Rejection sampling



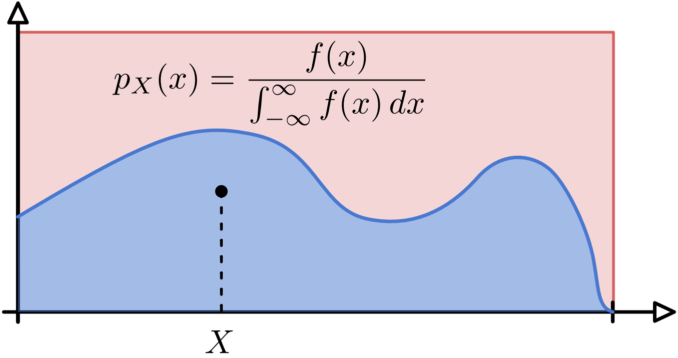
Rejection sampling



Rejection sampling



Rejection sampling



Rejection sampling

Rejection sampling Algorithm

Input:

- An integrable function $f \geq 0$ and an enveloppe $g \geq f$ which you can sample from.[1]

Output:

- A sample X distributed following the density proportional to f.

Algorithm:

1. Sample $X \sim g$ and $Y \mid X \sim \text{unif}(0, g(X))$.
2. If $Y < f(X)$, then return X. Otherwise go back to step 1.

[1] You can sample from the density proportional to g.

Rejection sampling

Rejection sampling Algorithm

Input:

- An integrable function $f \geq 0$ and an enveloppe $g \geq f$ which you can sample from.[1]

Output:

- A sample X distributed following the density proportional to f.

Algorithm:

1. Sample $X \sim g$ and $Y \mid X \sim \text{unif}(0, g(X))$.
2. If $Y < f(X)$, then return X. Otherwise go back to step 1.

[1] You can sample from the density proportional to g.

Rejection sampling

Rejection sampling Algorithm

Input:

- An integrable function $f \geq 0$ and an enveloppe $g \geq f$ which you can sample from.[1]

Output:

- A sample X distributed following the density proportional to f.

Algorithm:

1. Sample $X \sim g$ and $Y \mid X \sim \mathrm{unif}(0, g(X))$.
2. If $Y < f(X)$, then return X. Otherwise go back to step 1.

[1] You can sample from the density proportional to g.

Rejection sampling

Rejection sampling Algorithm

Input:

- An integrable function $f \geq 0$ and an enveloppe $g \geq f$ which you can sample from.[1]

Output:

- A sample X distributed following the density proportional to f.

Algorithm:

1. Sample $X \sim g$ and $Y \mid X \sim \text{unif}(0, g(X))$.
2. If $Y < f(X)$, then return X. Otherwise go back to step 1.

[1] You can sample from the density proportional to g.

Rejection sampling

Note:

- The function g is called the *enveloppe* function, and the corresponding distribution is the *proposal* distribution, or the *instrumental* distribution.
- The function f is called the *target*.

Lab 5

Lab 5

We want to sample from the density proportional to

$$f(x) = \sin^2(\pi x), \quad x \in [0, 1],$$

using rejection sampling.

We'll consider two proposal distributions:

- $Unif(0, 1)$
- $\text{Beta}(2, 2)$

Lab 5

We want to sample from the density proportional to

$$f(x) = \sin^2(\pi x), \quad x \in [0, 1],$$

using rejection sampling.

We'll consider two proposal distributions:

- $Unif(0, 1)$
- $\text{Beta}(2, 2)$

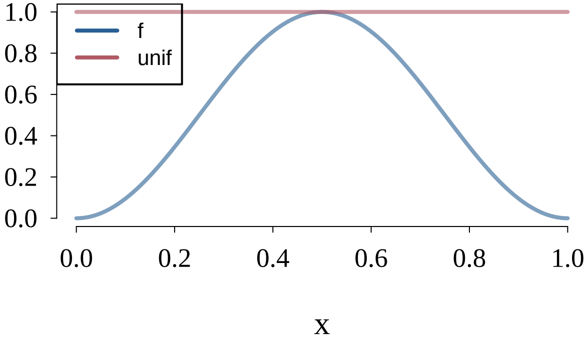
Lab 5

Task 1: Plot $f(x)$ and the Unif(0,1) density. Sample from $f(x)$ using the Unif(0,1) pdf as an enveloping function.

```
f <- function(x) sin(pi*x)^2
unif <- Vectorize(function(x) 1)

x = seq(0,1, length.out = 200)
plot(x, f(x), type="l", col=1, lwd=2, ylab="")
lines(x, unif(x), col=2, lwd=2)

legend("topleft", legend=c("f", "unif"),
       col=cmap.knitr(c(1,2)), lwd=2, lty=1, cex=0.7)
```

Lab 5

Task 1: Plot $f(x)$ and the Unif(0,1) density. Sample from $f(x)$ using the Unif(0,1) pdf as an enveloping function.



Lab 5

Task 1: Plot $f(x)$ and the Unif(0,1) density. Sample from $f(x)$ using the Unif(0,1) pdf as an enveloping function.

Let's implement rejection sampling for a single data point:

```
sample = NULL
while (is.null(sample)) {
  # Step 1
  x = runif(1, min=0, max=1)
  y = runif(1, min=0, max=unif(x))

  # Step 2
  if (y < f(x)) sample = x
}

x
```

```
## [1] 0.8648383
```

Lab 5

Now let's code a more general rejection sampling method.

```
rejection_sampling <- function(f, g, rg) {
  while (TRUE) { # Bad practice; doing this for brevity here.
    # Step 1
    x = rg(1)
    y = runif(1, min=0, max=g(x))

    # Step 2
    if (y < f(x)) return(sample)
  }
}

rejection_sampling(f, unif, runif)
```

```
## [1] 0.8648383
```

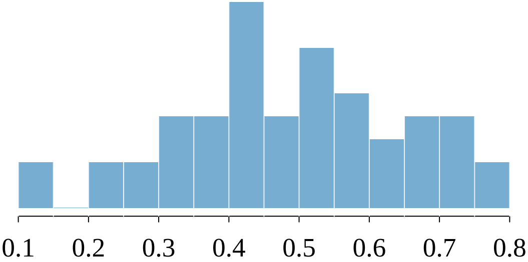
Lab 5

Task 2 Plot a histogram of the points that fall in the acceptance region. Do this for a simulation size of 10^2 and 10^5 and report your acceptance ratio. Compare the ratios and histograms.

```
k = 10^2

x = runif(k)
y = runif(k)

samples = x[y < f(x)]

hist(samples)
mean(y < f(x))
```

Lab 5

Task 2 Plot a histogram of the points that fall in the acceptance region. Do this for a simulation size of 10^2 and 10^5 and report your acceptance ratio. Compare the ratios and histograms.




```
## [1] 0.52
```

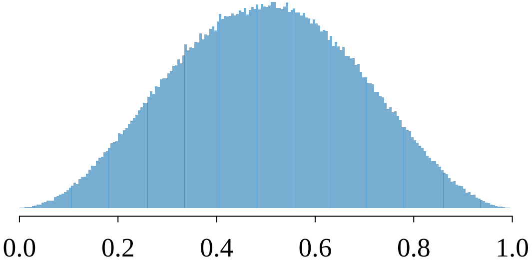
Lab 5

Task 2 Plot a histogram of the points that fall in the acceptance region. Do this for a simulation size of 10^2 and 10^5 and report your acceptance ratio. Compare the ratios and histograms.

```
k = 10^6

x = runif(k)
y = runif(k)

samples = x[y < f(x)]

hist(samples)
mean(y < f(x))
```

Lab 5

Task 2 Plot a histogram of the points that fall in the acceptance region. Do this for a simulation size of 10^2 and 10^5 and report your acceptance ratio. Compare the ratios and histograms.




```
## [1] 0.499041
```

Lab 5

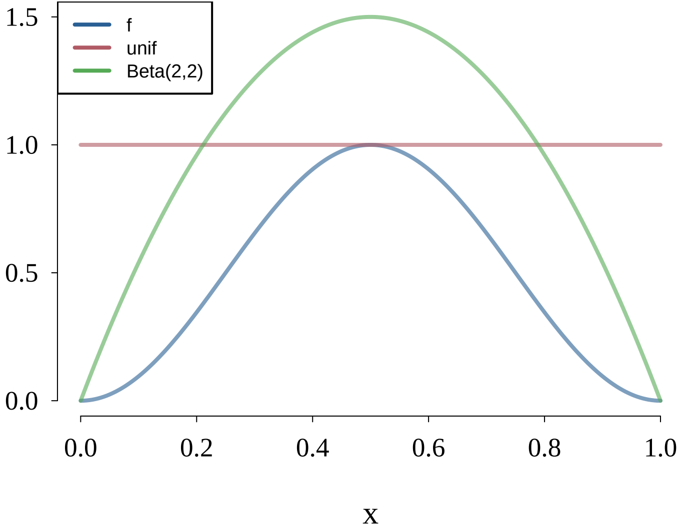
Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.

```
g = function(x) dbeta(x, 2,2)
rg = function(n) rbeta(n, 2,2)
```

```
f <- function(x) sin(pi*x)^2
unif <- Vectorize(function(x) 1)

x = seq(0,1, length.out = 200)
plot(x, f(x), type="l", col=1, lwd=2, ylab="", ylim=c(0,1.5))
lines(x, unif(x), col=2, lwd=2)
lines(x, g(x), col=3, lwd=2)

legend("topleft", legend=c("f", "unif", "Beta(2,2)"),
       col=cmap.knitr(c(1,2,3)), lwd=2, lty=1, cex=0.7)
```

Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.



Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.

```
g = function(x) dbeta(x, 2,2)
rg = function(n) rbeta(n, 2,2)
```

```
rejection_sampling(f, g, rg)
```

```
## [1] 0.8648383
```

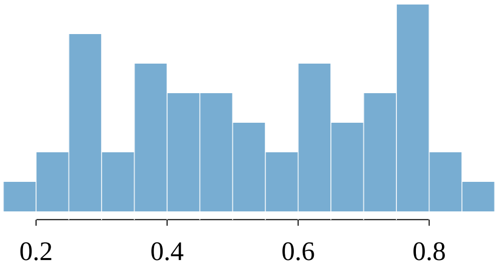
Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.

```
k = 10^2

x = rg(k)
y = runif(k, min=0, max=g(x))

samples = x[y < f(x)]

hist(samples)
mean(y < f(x))
```

Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.




```
## [1] 0.51
```

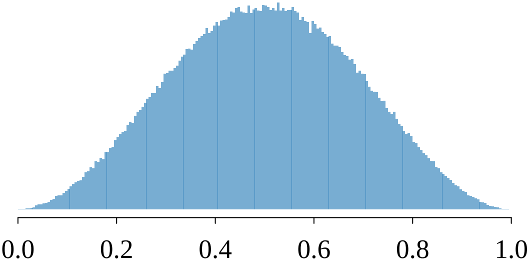
Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.

```
k = 10^6

x = rg(k)
y = runif(k, min=0, max=g(x))

samples = x[y < f(x)]

hist(samples)
mean(y < f(x))
```

Lab 5

Task 3 Repeat Tasks 1 - 2 for Beta(2,2) as an enveloping function.




```
## [1] 0.499393
```