Module 6: Introduction to Monte Carlo

Rebecca C. Steorts

Agenda

- Motivation
- Numerical Integration
- Monte Carlo
- The Naive Method
- Importance Sampling
- Rejection Sampling

Intractable Integrals

Goal: approximate

$$\int_X h(x) f(x)\, dx$$

that is intractable, where $f(x)$ is a probability density.

- What's the problem? Typically $h(x)$ is messy!
- Why not use numerical integration techniques?
- In dimension $d = 3$ or higher, Monte carlo really improves upon numerical integration.

Numerical Integration

- The most serious problem is the so-called "curse of dimensionality."
- Suppose we have a d-dimensional integral.
- Numerical integration typically entails evaluating the integrand over some grid of points.
- However, if d is even moderately large, then any reasonably fine grid will contain an impractically large number of points.

Numerical integration

- Let $d = 6$. Then a grid with just ten points in each dimension will consist of 10^6 points.
- If $d = 50$, then even an absurdly coarse grid with just *two* points in each dimension will consist of 2^{50} points (note that $2^{50} > 10^{15}$).

What's happening here?

Numerical integration error rates (big Ohh concepts)

- If $d = 1$ and we assume crude numerical integration based on a grid size n, then we typically get an error of order n^{-1}.
- For most dimensions d, estimates based on numerical integrations required m^d evaluations to achieve an error of m^{-1}.
- Said differently, with n evaluations, you get an error of order $n^{-1/d}$.
- But, the Monte Carlo estimate retains an error rate of $n^{-1/2}$. (The constant in this error rate may be quite large).

Classical Monte Carlo Integration

The generic problem here is to evaluate

$$E_f[h(x)] = \int_X h(x) f(x)\, dx.$$

The classical way to solve this is generate a sample $(X_1, \ldots, X_n)$ from f.

Now propose as an approximation the empirical average:

$$\bar{h}_n = \frac{1}{n} \sum_{j=n}^{n} h(x_j).$$

Why? $\bar{h}_n$ converges a.s. (i.e. for almost every generated sequence) to $E_f[h(X)]$ by the Strong Law of Large Numbers.

Monte Carlo (continued)

Also, under certain assumptions[1], the asymptotic variance can be approximated and then can be estimated from the sample $(X_1, \ldots, X_n)$ by

$$v_n = 1/n^2 \sum_{j=1}^{n} [h(x_j) - \bar{h}_n]^2.$$

Finally, by the CLT (for large n),

$$\frac{\bar{h}_n - E_f[h(X)]}{\sqrt{v_n}} \overset{\text{approx.}}{\sim} N(0,1).$$

(Technically, it converges in distribution).

[1] see Casella and Robert, page 65, for details

Root Mean Squared Error (RMSE)

Assume that X_i are iid observations.

Due to unbiasedness, the root-mean-squared-error (RMSE) equals the standard deviation (square root of the variance) of $\frac{1}{N}\sum X_i$,

$$\begin{aligned}
\text{RMSE} &= \left[\mathbb{E}(|\tfrac{1}{N}\textstyle\sum X_i - \mathbb{E}X|^2)\right]^{1/2} \\
&= \left[\mathbb{V}(\tfrac{1}{N}\textstyle\sum X_i)\right]^{1/2} \\
&= \frac{1}{\sqrt{N}}\mathbb{V}(X)^{1/2} = \sigma(X)/\sqrt{N}. \qquad (1)
\end{aligned}$$

The RMSE tells us how far the approximation will be from the true value, on average.

As a practical matter, we need to be able to draw the samples X_i in a computationally-efficient way.

Return of IQ Scores

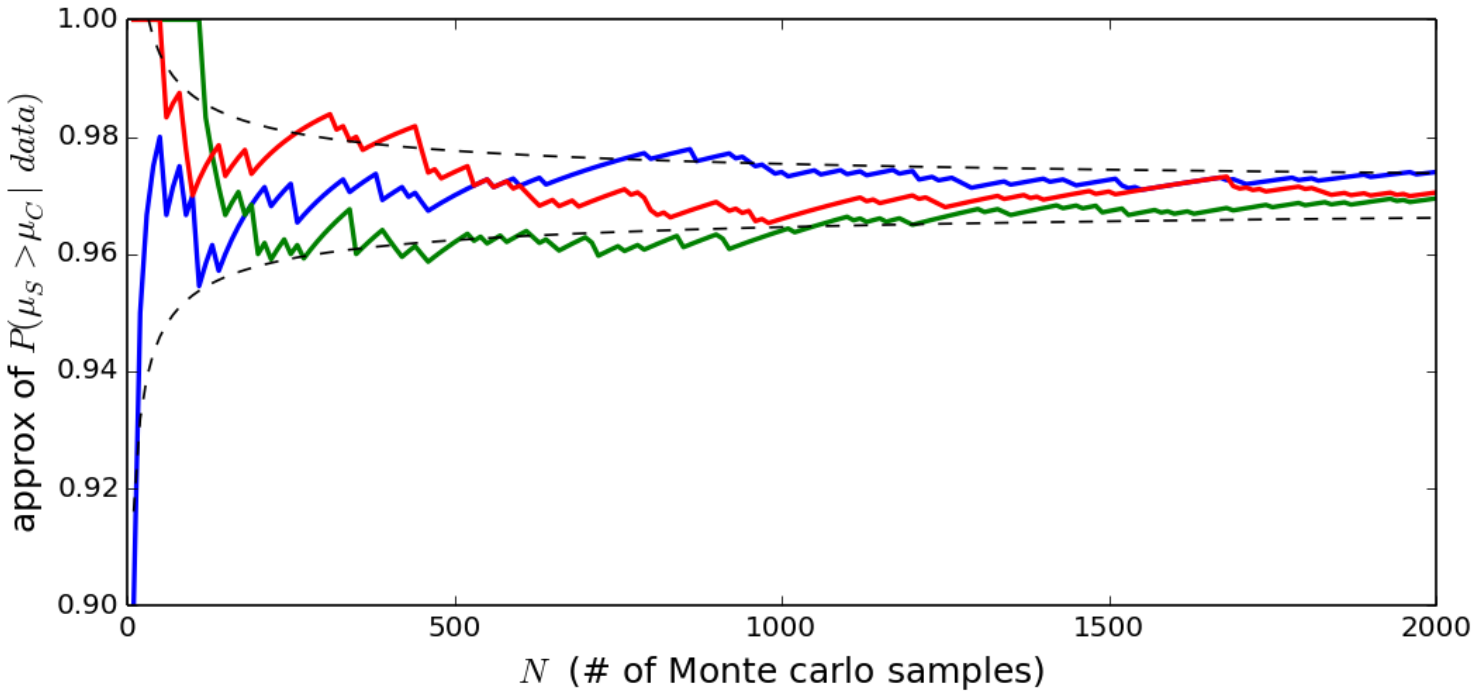


Figure 1: Monte Carlo approximations for an increasing number of samples, N. The red, blue, and green lines indicate three repetitions of the procedure, using different sequences of samples. The dotted lines indicate the true value $\pm$ the RMSE of the Monte Carlo estimator.

Return of IQ Scores

In Module 4, we saw an example involving the mean change in IQ score μ_S and μ_C of two groups of students (spurters and controls). To compute the posterior probability that the spurters had a larger mean change in IQ score, we drew $N = 10^6$ samples from each posterior:

$$(\boldsymbol{\mu}_S^{(1)}, \boldsymbol{\lambda}_S^{(1)}), \ldots, (\boldsymbol{\mu}_S^{(N)}, \boldsymbol{\lambda}_S^{(N)}) \sim \text{NormalGamma}(24.0, 8, 4, 855.0)$$
$$(\boldsymbol{\mu}_C^{(1)}, \boldsymbol{\lambda}_C^{(1)}), \ldots, (\boldsymbol{\mu}_C^{(N)}, \boldsymbol{\lambda}_C^{(N)}) \sim \text{NormalGamma}(11.8, 49, 24.5, 6344.0)$$

and used the Monte Carlo approximation

$$\mathbb{P}(\boldsymbol{\mu}_S > \boldsymbol{\mu}_C \mid \text{data}) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(\boldsymbol{\mu}_S^{(i)} > \boldsymbol{\mu}_C^{(i)}).$$

Return of IQ Scores

- To visualize this, consider the sequence of approximations $\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(\boldsymbol{\mu}_S^{(i)} > \boldsymbol{\mu}_C^{(i)})$ for $N = 1, 2, \ldots$.
- Figure 1 shows this sequence of approximations for three different sets of random samples from the posterior.
- We can see that as the number of samples used in the approximation grows, it appears to be converging to around 0.97.

To visualize the theoretical rate of convergence, the figure also shows bands indicating the true value $\alpha = \mathbb{P}(\boldsymbol{\mu}_S > \boldsymbol{\mu}_C \mid \text{data}) =??$ plus or minus the RMSE of the Monte Carlo estimator, that is, from Equation 1:

$$\alpha \pm \sigma(X)/\sqrt{N} =??$$

Simpify this as much as possible for an ungraded exercise (exam II).

Solution to the ungraded exercise

$$\begin{aligned}\alpha \pm \sigma(X)/\sqrt{N} &= \alpha \pm \sqrt{\alpha(1-\alpha)/N} \\ &= 0.97 \pm \sqrt{0.97(1-0.97)/N}\end{aligned}$$

where X has the posterior distribution of $\mathbb{1}(\mu_S > \mu_C)$ given the data, in other words, X is a Bernoulli(α) random variable. Recall that the variance of a Bernoulli(α) random variable is $\alpha(1-\alpha)$.

Return of IQ Scores

Using the same approach, we could easily approximate any number of other posterior quantities as well, for example,

$$\mathbb{P}(\boldsymbol{\lambda}_S > \boldsymbol{\lambda}_C \,|\, \text{data}) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(\boldsymbol{\lambda}_S^{(i)} > \boldsymbol{\lambda}_C^{(i)})$$

$$\mathbb{E}(|\boldsymbol{\mu}_S - \boldsymbol{\mu}_C| \,|\, \text{data}) \approx \frac{1}{N} \sum_{i=1}^{N} |\boldsymbol{\mu}_S^{(i)} - \boldsymbol{\mu}_C^{(i)}|$$

$$\mathbb{E}(\boldsymbol{\mu}_S / \boldsymbol{\mu}_C \,|\, \text{data}) \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{\mu}_S^{(i)} / \boldsymbol{\mu}_C^{(i)}.$$

Importance Sampling

Recall that we have a difficult, problem child of a function $h(x)$!

- Generate samples from a distribution $g(x)$.
- We then "re-weight" the output.

Note: g is chosen to give greater mass to regions where h is large (the important part of the space).

This is called *importance sampling.*

Importance Sampling

Let g be an arbitrary density function and then we can write

$$I = E_f[h(x)] = \int_X h(x)\frac{f(x)}{g(x)}g(x)\,dx = E_g\left[\frac{h(x)f(x)}{g(x)}\right]. \qquad (2)$$

This is estimated by

$$\hat{I} = \frac{1}{n}\sum_{j=1}^{n}\frac{f(X_j)}{g(X_j)}h(X_j) \longrightarrow E_f[h(X)] \qquad (3)$$

based on a sample generated from g (not f). Since (2) can be written as an expectation under g, (3) converges to (2) for the same reason the Monte carlo estimator $\bar{h}_n$ converges.

The Variance

$$Var(\hat{I}) = \frac{1}{n^2} \sum_i Var\left(\frac{h(X_i)f(X_i)}{g(X_i)}\right) \tag{4}$$

$$= \frac{1}{n} Var\left(\frac{h(X_i)f(X_i)}{g(X_i)}\right) \implies \tag{5}$$

$$\widehat{Var}(\hat{I}) = \frac{1}{n} \widehat{Var}\left(\frac{h(X_i)f(X_i)}{g(X_i)}\right). \tag{6}$$

Simple Example

Suppose we want to estimate $P(X > 5)$, where $X \sim N(0, 1)$.

Naive method (Monte carlo):

- Generate $X_1 \dots X_n \overset{iid}{\sim} N(0, 1)$
- Take the proportion $\hat{p} = \bar{X} > 5$ as your estimate

Importance sampling method:

- Sample from a distribution that gives high probability to the "important region" (the set $(5, \infty)$).
- Do re-weighting.

Importance Sampling Solution

Let $f = \phi_o$ and $g = \phi_\theta$ be the densities of the $N(0,1)$ and $N(\theta,1)$ distributions (θ taken around 5 will work). Then

$$p = \int I(u > 5)\phi_o(u)\, du \tag{7}$$

$$= \int \left[I(u > 5)\frac{\phi_o(u)}{\phi_\theta(u)} \right] \phi_\theta(u)\, du. \tag{8}$$

In other words, if

$$h(u) = I(u > 5)\frac{\phi_o(u)}{\phi_\theta(u)}$$

then $p = E_{\phi_\theta}[h(X)]$.

If $X_1, \ldots, X_n \sim N(\theta, 1)$, then an unbiased estimate is $\hat{p} = \frac{1}{n}\sum_i h(X_i)$.

Naive Method Solution

```
1 - pnorm(5)                          # gives 2.866516e-07
```

```
## [1] 2.866516e-07
```

```
## Naive method (Monte Carlo)
set.seed(1)
mySample <- 100000
x <- rnorm(n=mySample)
# pHat is a bernoulli proportion
pHat <- sum(x>5)/length(x)
# to calculate the variance,
# use p(1-p)/number of trials
sdPHat <- sqrt(pHat*(1-pHat)/length(x)) # gives 0
```

IS Solution

```
set.seed(1)
y <- rnorm(n=mySample, mean=5)
h <- dnorm(y, mean=0)/dnorm(y, mean=5) * I(y>5)
mean(h)                                # gives 2.865596e-07
```

```
## [1] 2.865596e-07
```

```
sd(h)/sqrt(length(h))                  # gives 2.157211e-09
```

```
## [1] 2.157211e-09
```

What is the difference between the naive (monte carlo) and the IS solution?

Question

Is there some special choice regarding our importance region?

What happens when we look at the region $u > 100$?

```
set.seed(1)
y <- rnorm(n=mySample, mean=100)
h <- dnorm(y, mean=0)/dnorm(y, mean=100) * I(y>100)
(mean(h))
```

```
## [1] 0
```

```
# versus 2.865596e-07 (u>5)
(sd(h)/sqrt(length(h)))
```

```
## [1] 0
```

```
# versus 2.157211e-09 (u>5)
```

Harder example

Let $f(x)$ be the pdf of a $N(0,1)$. Assume we want to compute

$$a = \int_{-1}^{1} f(x)dx = \int_{-1}^{1} N(0,1)dx$$

Let $g(X)$ be an arbitrary pdf,

$$a(x) = \int_{-1}^{1} \frac{f(x)}{g(x)} g(x)\ dx.$$

We want to be able to draw $g(x) \sim Y$ easily. But how should we go about choosing $g(x)$?

Harder example (continued)

- Note that if $g \sim Y$, then $a = E[I_{[-1,1]}(Y)\frac{f(Y)}{g(Y)}]$.
- The variance of $I_{[-1,1]}(Y)\dfrac{f(Y)}{g(Y)}$ is minimized picking $g \propto I_{[-1,1]}(x)f(x)$. Nevertheless simulating from this g is usually expensive.
- Some g's which are easy to simulate from are the pdf's of:
  - the Uniform$(-1, 1)$,
  - the Normal$(0, 1)$,
  - and a Cauchy with location parameter 0 (Student t with 1 degree of freedom).
- Below, there is code of how to get a sample from

$$I_{[-1,1]}(Y)\frac{f(Y)}{g(Y)}$$

for the three choices of g.

Solution to Harder example

```
uniformIS <- function(sampleSize=10) {
  sapply(runif(sampleSize,-1,1),
    function(xx) dnorm(xx,0,1)/dunif(xx,-1,1)) }

cauchyIS <- function(sampleSize=10) {
  sapply(rt(sampleSize,1),
    function(xx)
    (xx <= 1)*(xx >= -1)*dnorm(xx,0,1)/dt(xx,2)) }

gaussianIS <- function(sampleSize=10) {
  sapply(rnorm(sampleSize,0,1),
    function(xx) (xx <= 1)*(xx >= -1)) }
```

Harder example (continued)

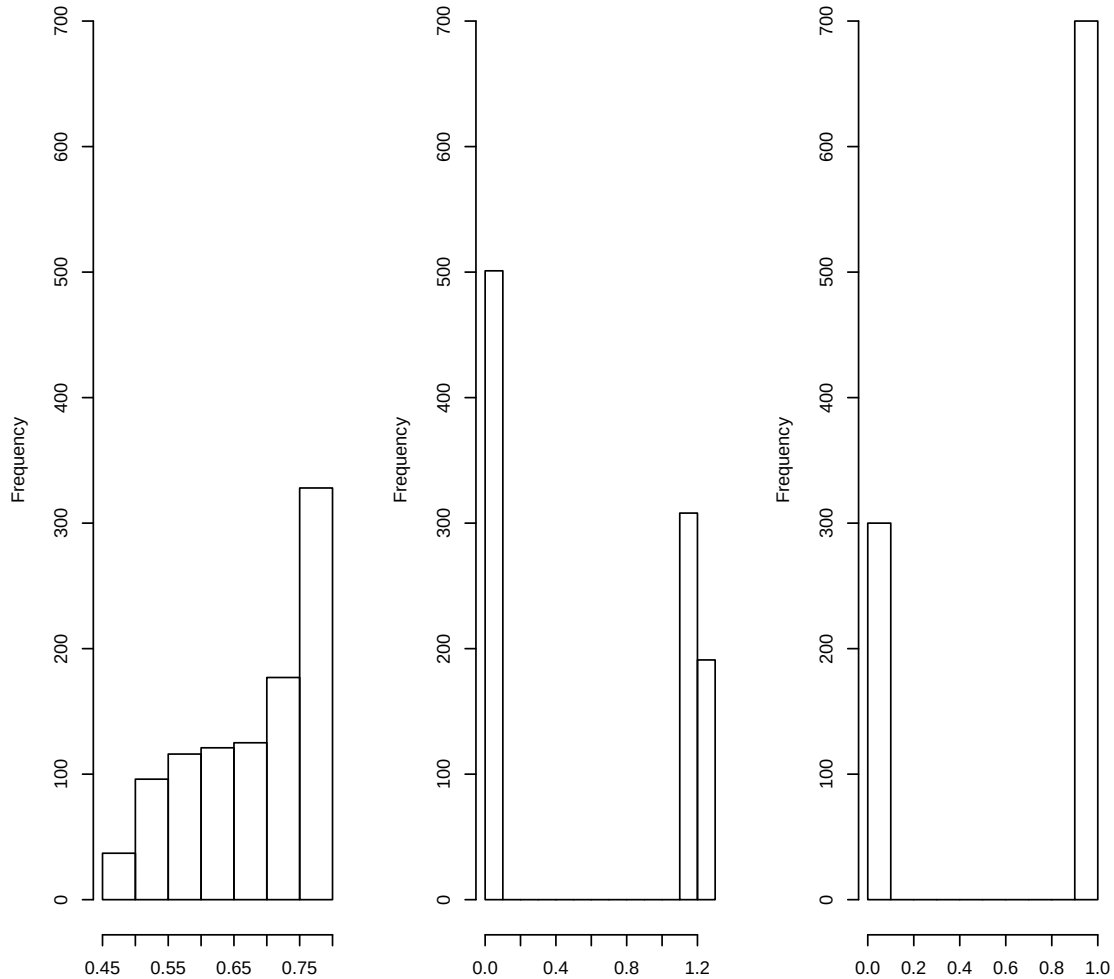


Figure 2: Histograms for samples from $I_{[-1,1]}(Y)\frac{f(Y)}{g(Y)}$ when g is, respectivelly, a uniform, a Cauchy and a Normal pdf.

Rejection Sampling

Rejection sampling is a method for drawing random samples from a distribution whose p.d.f. can be evaluated up to a constant of proportionality.

Compared with the inverse c.d.f. method, rejection sampling has the advantage of working on complicated multivariate distributions. (see homework)

Difficulties? You must design a good proposal distribution (which can be difficult, especially in high-dimensional settings).

Uniform Sampler

Goal: Generate samples from Uniform(A), where A is complicated.

- $X \sim Uniform(Mandelbrot)$.
- Consider $I_{X(A)}$.

The Mandelbrot

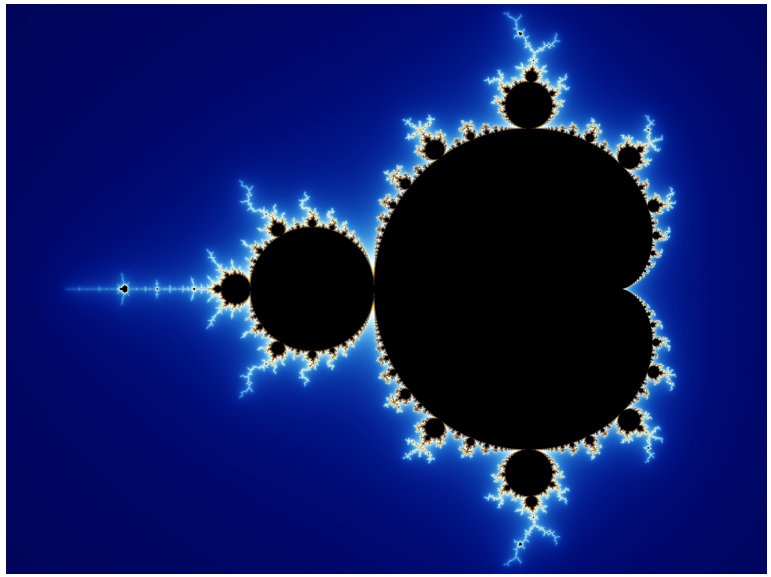


Figure 3: A complicated function A, called the Mandelbrot!

Exercise

- Suppose $A \subset B$.
- Let $Y_1, Y_2, \ldots \sim \text{Uniform(B)}$ iid and
- $X = Y_k$ where $k = \min\{k : Y_k \in A\}$,

Then it follows that

$$X \sim \text{Uniform}(A).$$

Proof: Exercise. Hint: Try the discrete case first and use a geometric series.

Drawing Uniform Samples

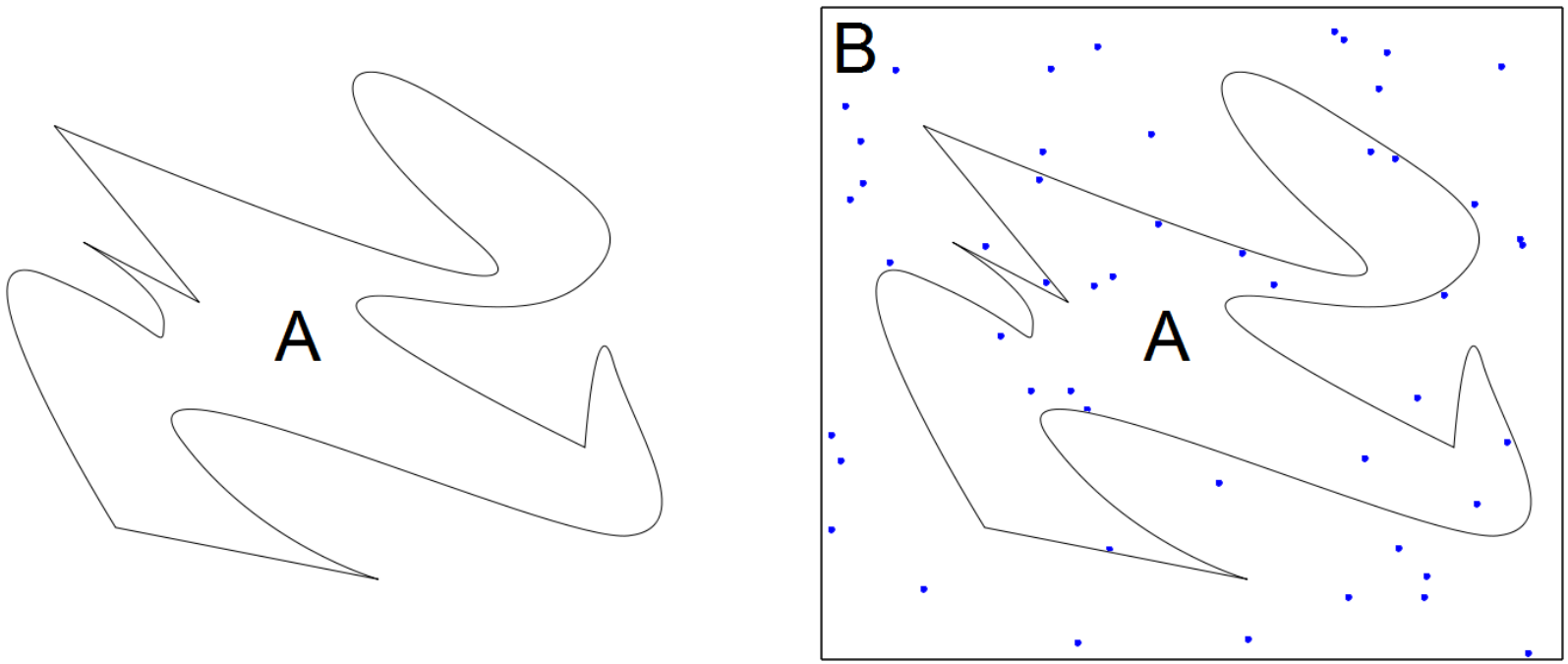


Figure 4: (Left) How to draw uniform samples from region A? (Right) Draw uniform samples from B and keep only those that are in A.

General Rejection Sampling Algorithm

Goal: Sample from a complicated pdf $f(x)$.

Suppose that

$$f(x) = \tilde{f}(x)/\alpha, \alpha > 0$$

.

Assumption: f is difficult to evaluate, $\tilde{f}$ is easy!

Suppose the density g is such that for some known constant M,

$$Mg(x) \geq \ell(x)$$

for all x.

Procedure:

1. Choose a proposal distribution q such that $c > 0$ with
$$cq(x) \geq \tilde{f}(x).$$
2. Sample $X \sim q$, sample $Y \sim \text{Unif}(0, c\, q(X))$ (given X)
3. If $Y \leq \tilde{f}(X), Z = X$, otherwise we reject and return to step (2)

Visualizing just f

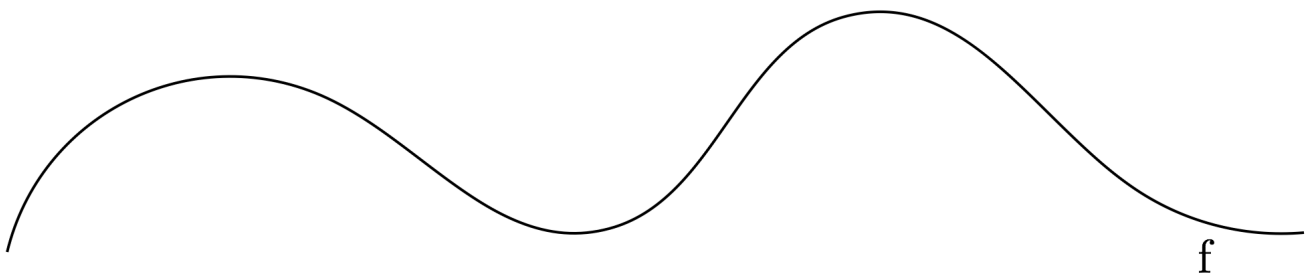


Figure 5: Visualizing just f.

Visualizing just f and $\tilde{f}$

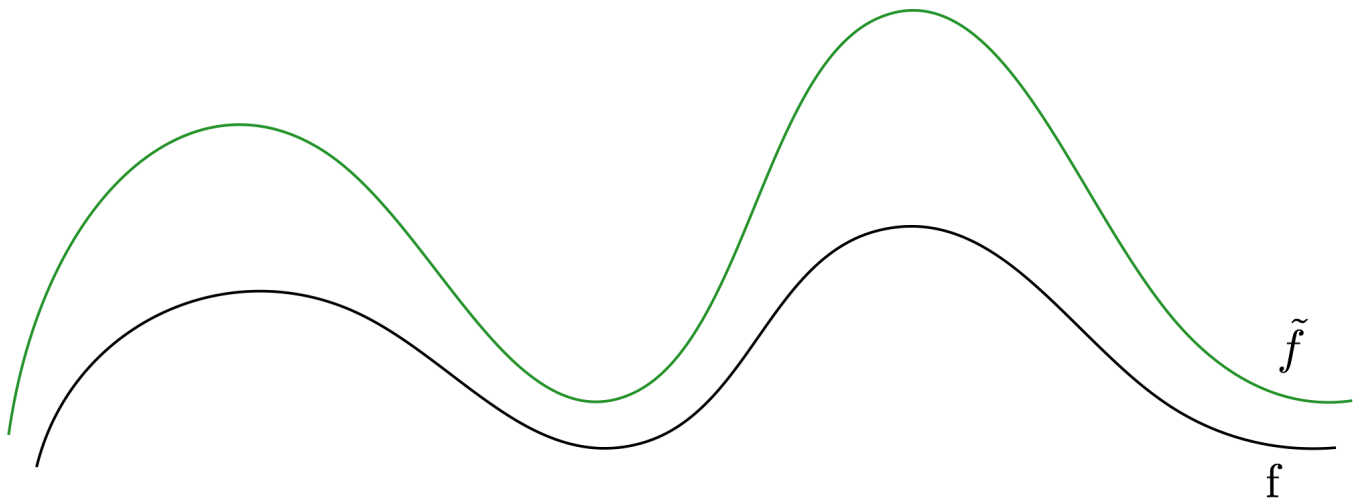


Figure 6: Visualizing just f and $\tilde{f}$.

Enveloping q over f

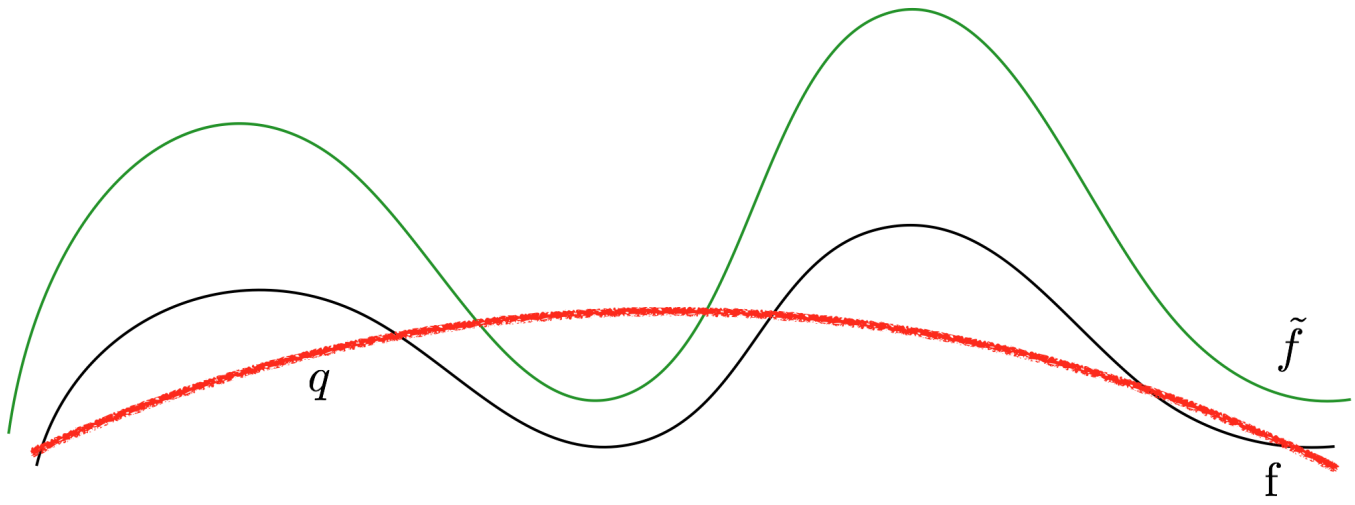


Figure 7: Visualizing f and $\tilde{f}$. Now we look at enveloping q over f.

Enveloping cq over $\tilde{f}$

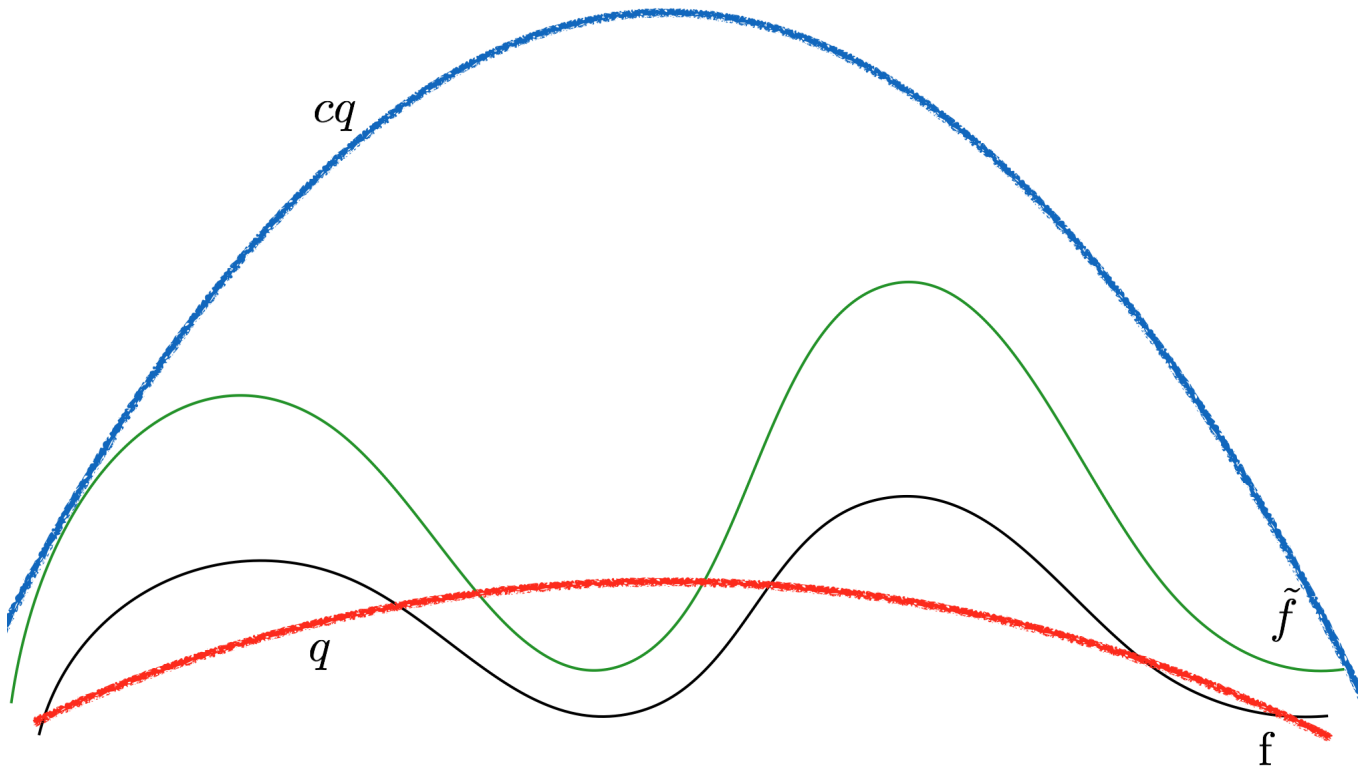


Figure 8: Visualizing f and $\tilde{f}$. Now we look at enveloping cq over $\tilde{f}$.

Recalling the sampling method and accept/reject step

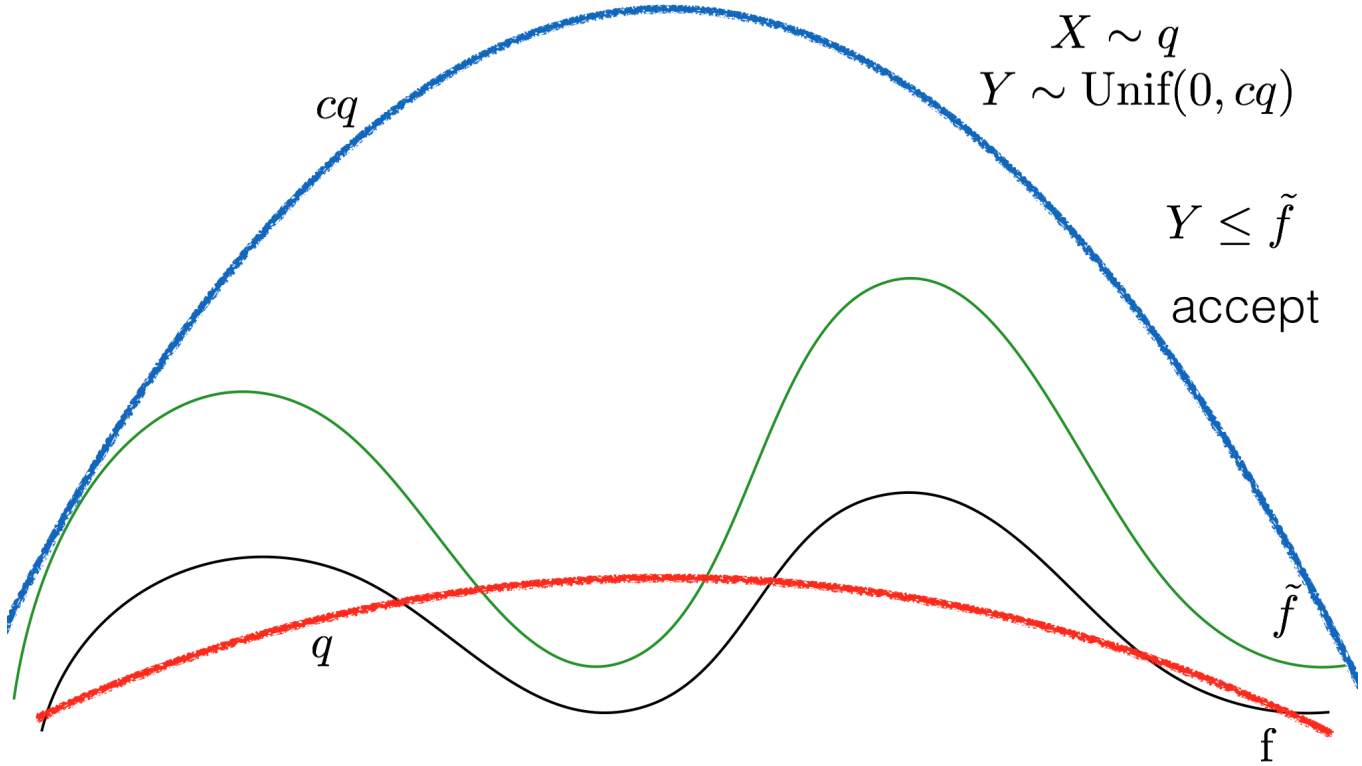


Figure 9: Recalling the sampling method and accept/reject step.

Entire picture and an example point X and Y

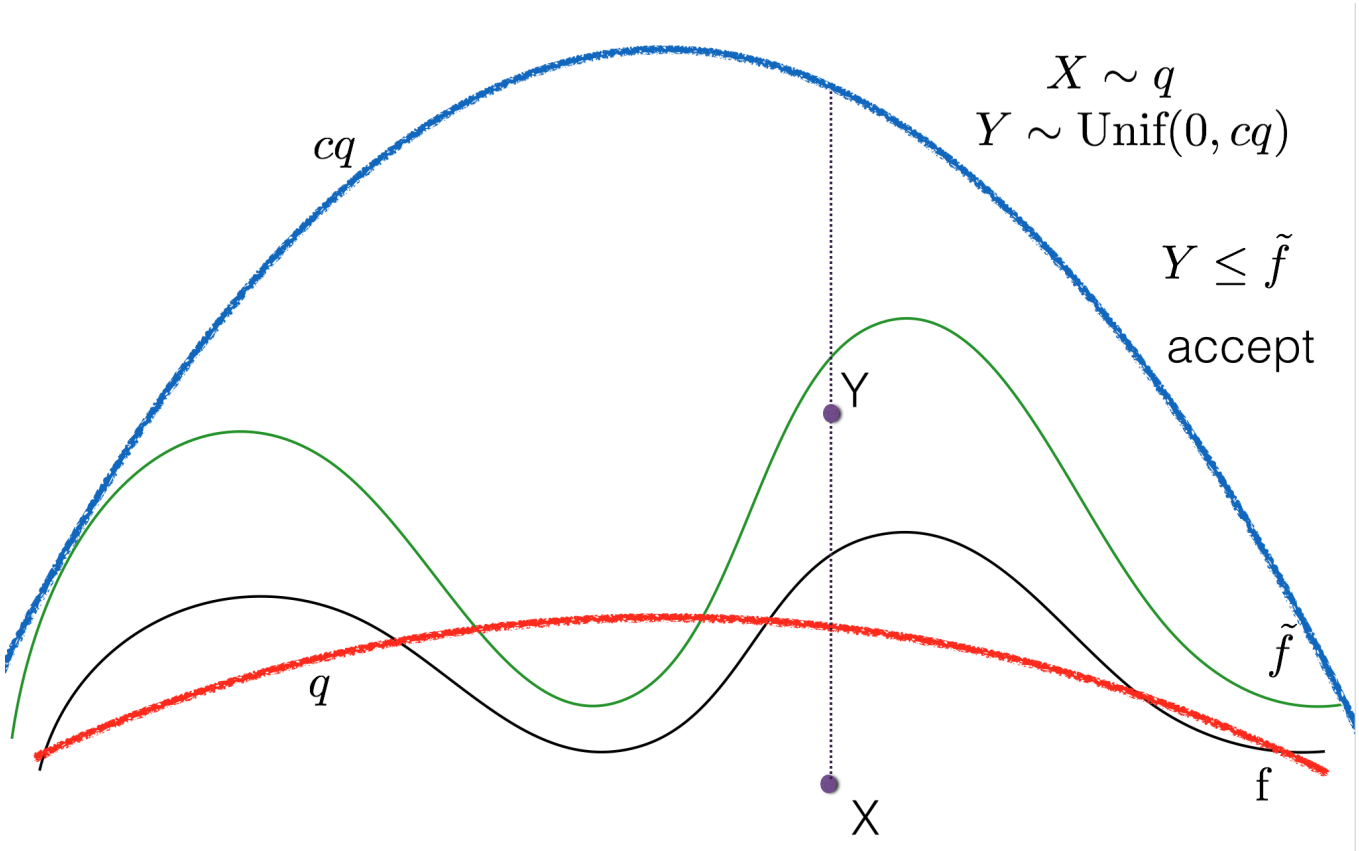


Figure 10: Entire picture and an example point X and Y.

Exercise

Consider the function

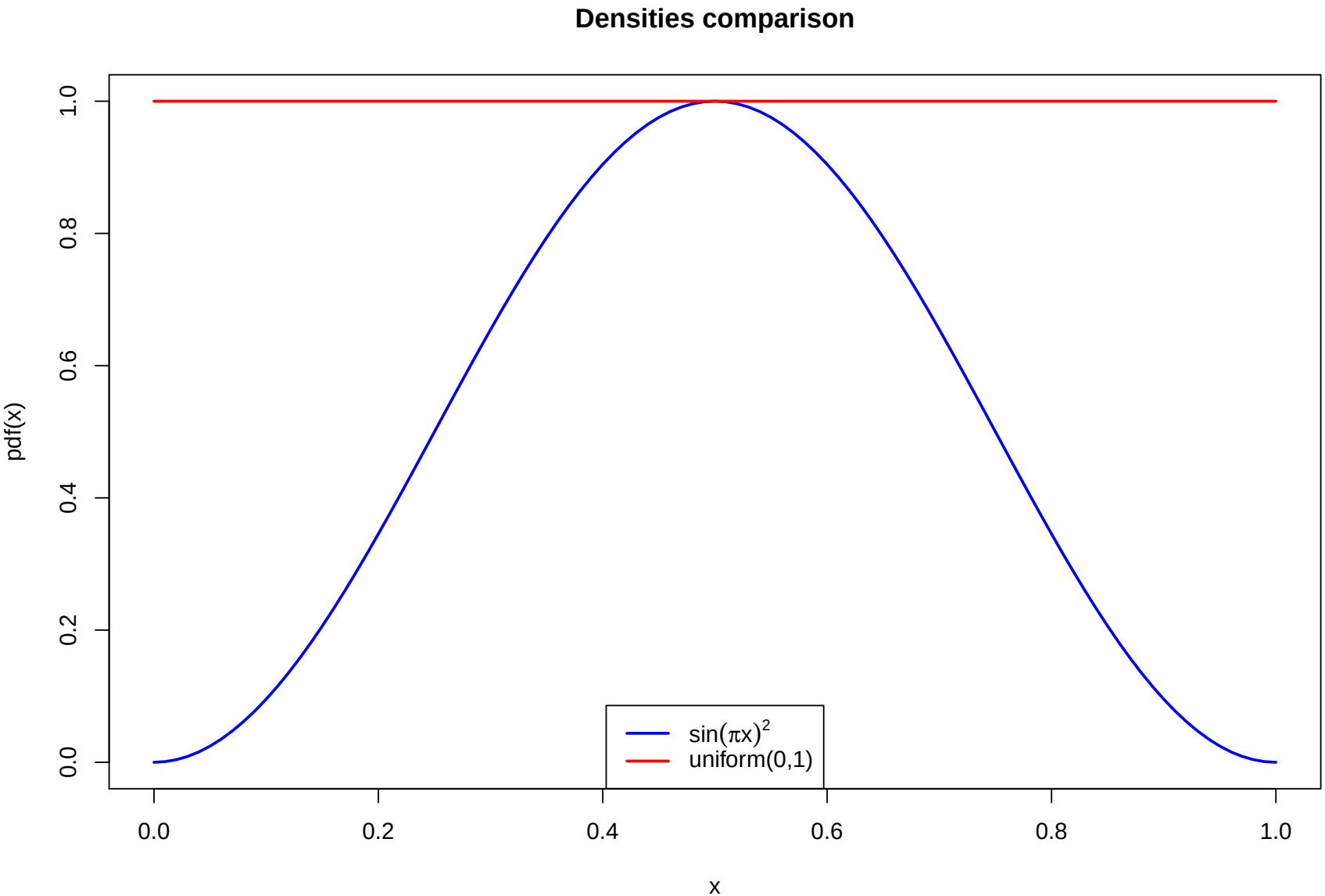
$$f(x) \propto \sin^2(\pi x), x \in [0, 1]$$

1. Plot the densities of $f(x)$ and the Unif(0,1) on the same plot.
2. According to the rejection sampling approach sample from $f(x)$ using the Unif(0,1) pdf as an enveloping function.
3. 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.
4. Repeat Tasks 1 - 3 for Beta(2,2) as an enveloping function. Compare your results with results in Task 3.
5. Using importance sampling calculate the $\mathbb{E}(x)$, where $x \sim f(x)$ from previous task and the proposal distribution is transformed Beta(2,2). (This will be covered in more detail in lab.)

Task 1

```r
# density function for f(x)
densFun <- function(x) {
  return(sin(pi*x)^2)
}
x <- seq(0, 1, 10^-2)
```

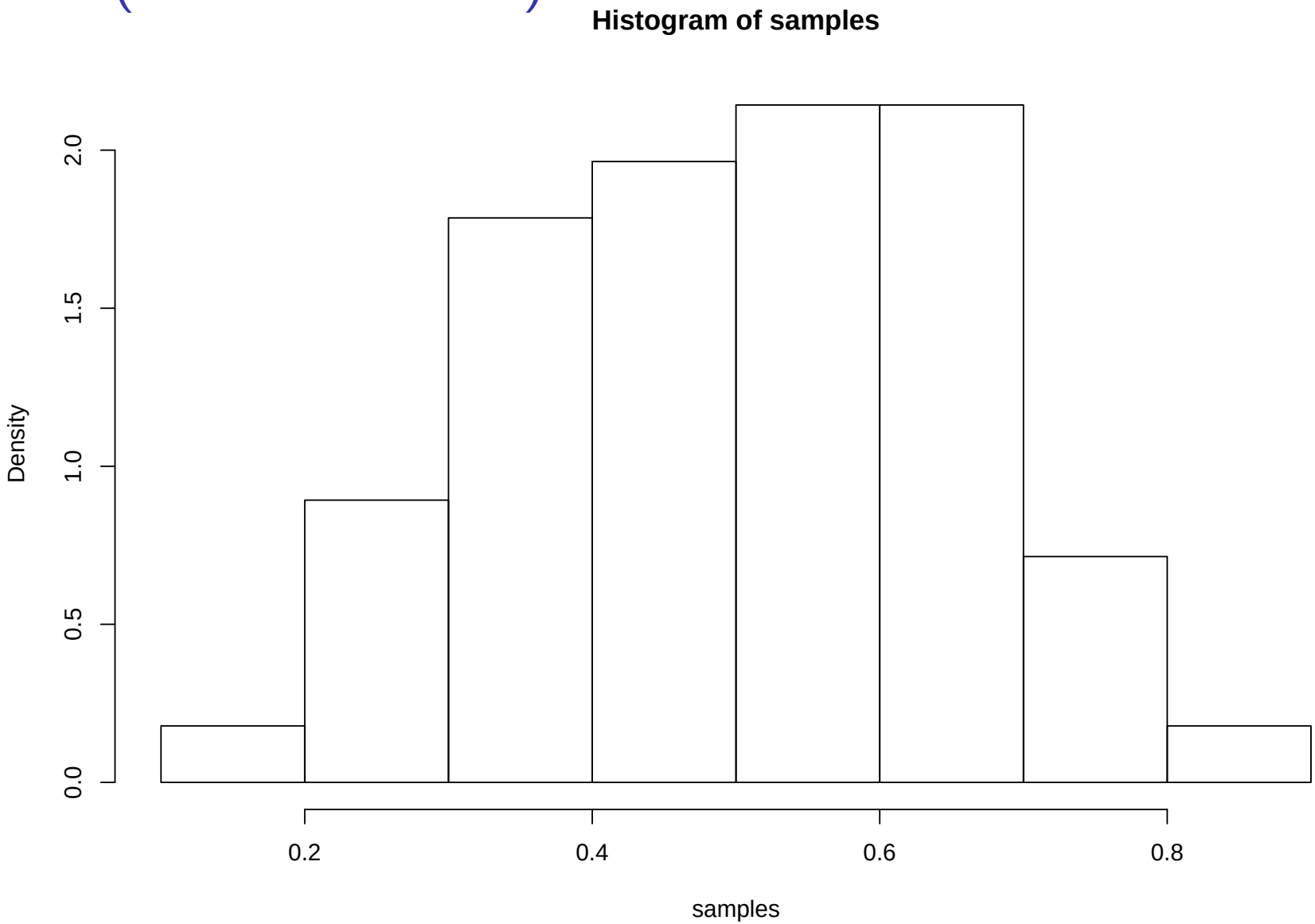
Task 1


Densities comparison
pdf(x)
0.0
0.2
0.4
0.6
0.8
1.0
x
0.0
0.2
0.4
0.6
0.8
1.0
sin(πx)^2
uniform(0,1)

Task 2

```
numSim=10^2
samples = NULL
for (i in 1:numSim) {
  # get a uniform proposal
  proposal <- runif(1)
  # calculate the ratio
  densRat <- densFun(proposal)/dunif(proposal)
  #accept the sample with p=densRat
  if ( runif(1) < densRat ){
    #fill our vector with accepted samples
    samples <- c(samples, proposal)
  }
}
```

Task 3 (Partial solution)




```
## [1] "Acceptance Ratio: 0.56"
```

Task 4 and 5

You will finish these for your next homework and see these in lab 5.