Math 490 Exam #2

Maxwell Levin

March 9, 2018

Problem 1. (10 Pts)

Use the rejection sampling method to sample from the distribution of random variable X that has the probability density function $f(x)$ given by

$$f(x) = \begin{cases} \frac{2x}{5} & 0 \leq x < 1; \\ \frac{2}{5} & 1 \leq x < 2; \\ \frac{4}{5} - \frac{x}{5} & 2 \leq x \leq 4; \\ 0 & \text{otherwise.} \end{cases}$$

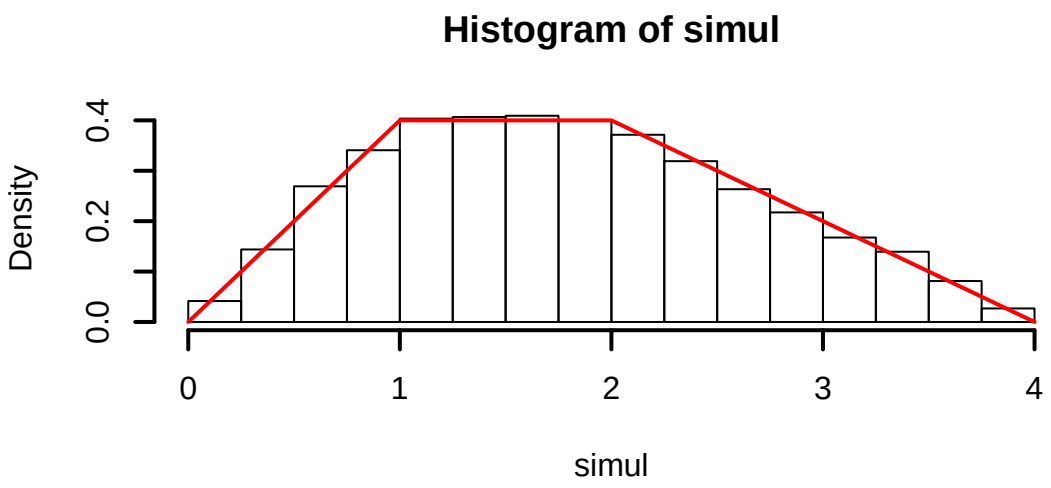
Note that the random variable X has mean $E(X) = \frac{9}{5}$ and variance $Var(X) = \frac{109}{150}$. Answer the following questions:

a. (8 Pts) Use the Uniform(0,4) distribution as the proposal distribution to sample from $X \sim f(x)$. Apply the rejection sampling method with 10000 trials. Make a histogram (with density on the vertical axis), add the density curve to the histogram, and report the mean and the standard deviation of the simulated values.

We can run the following code in R:

```
target = function(x) {
  ifelse(0 <= x & x < 1, 2*x/5,
         ifelse(1 <= x & x < 2, 2/5,
                ifelse(2 <= x & x < 4, 4/5 - x/5, 0)
                )
         )
}

proposal = function(x) { 0.25; }

rejCont = function(trials) {
  supremum = 8/5
    results <- c()
  for (i in 1:trials){
      u<-runif(1)
      y<-runif(1, 0, 4)
      if (u <= target(y) / (supremum * proposal(y)) ) {
          results = c(results, y)
      }
    }
  results;
}

trials = 10000
simul = rejCont(trials)
```

```r
hist(simul, breaks=seq(0, 4, .25), prob=T, lwd=2)

x = seq(0, 4, 0.01)
y = target(x)
lines(x, y, col="red", lwd=2)
```



The mean of our simulation is

```
[1] 1.802071
```

The variance is

```
[1] 0.7311974
```

The standard deviation is

```
[1] 0.8551008
```

Notice that our results are in agreement with what we've calculated.

b. (2 Pts) For the rejection sampling method used in part (a), report the probability of acceptance in your simulation. Also find the theoretic probability of acceptance.

We can get the probability of acceptance of our simulation by running the following code:

```r
length(simul)/10000
```

```
[1] 0.6256
```

We know that the theoretical probability of acceptance is given by

$$Pr\{\text{Accept}\} = \frac{1}{M} = \frac{5}{8} = 0.625.$$

The probability of acceptance in our simulation is near the theoretical probability of acceptance.

In Problems 2, 3, and 4, we will apply some Monte Carlo methods to estimate the definite integral

$$I = \int_0^4 e^{-(x-2)^2} dx \approx 1.76416.$$

Problem 2. (18 Pts)

First, let us rewrite $I = \int_0^4 4e^{-(x-2)^2} \frac{1}{4} dx$, and consider the random variable $\phi(X) = 4e^{-(X-2)^2}$ with $X \overset{d}{\sim}$ Uniform(0, 4).

a. (2 Pts) Use $\phi(X)$ to construct a Monte Carlo estimator and label this estimator by Z_N^{MC}, where N is the sample size. Clearly explain the Z_N^{MC} that you construct.

First, notice that our integral has become

$$I = \int_0^4 \frac{1}{4} \phi(x) dx = E[\phi(X)],$$

where $\phi(\cdot)$ and X are given in the problem description. Because our goal is to use Monte Carlo methods to estimate this integral, we must approximate the expectation value for $\phi(X)$ by taking the average value of X for some $X \overset{d}{\sim}$ Uniform(0, 1). Thus we arrive at the Monte Carlo estimator,

$$I \approx Z_N^{MC} = \frac{\phi(X_1) + \phi(X_2) + \cdots + \phi(X_N)}{N}.$$

b. (3 Pts) If sample size $N = 400$, what are the mean and standard deviation of the Monte Carlo estimator Z_{400}^{MC}?

We know that the mean of our Monte Carlo estimator does not change with sample size. That is,

$$E[Z_{400}^{MC}] = E[\phi(X)] = I \approx 1.76416.$$

Calculating the standard deviation requires some more work. First, we must compute the variance for when $N = 1$:

$$Var(\phi(X)) = E[\phi(X)^2] - I^2 = \int_0^4 e^{-2(x-2)^2} - I^2.$$

Since this integral is difficult to do analytically, we can just ask Mathematica to compute it using numerical methods of integration:

```
I = N[Integrate[e^(-(x-2)^2), {x, 0, 4}], 6]
N[Integrate[4*e^(-2*(x-2)^2), {x, 0, 4} ] - I^2, 6]
1.90068
```

We take the square root of our variance to get

$$SD(\phi(X)) = \sqrt{Var(\phi(X))} \approx 1.37865.$$

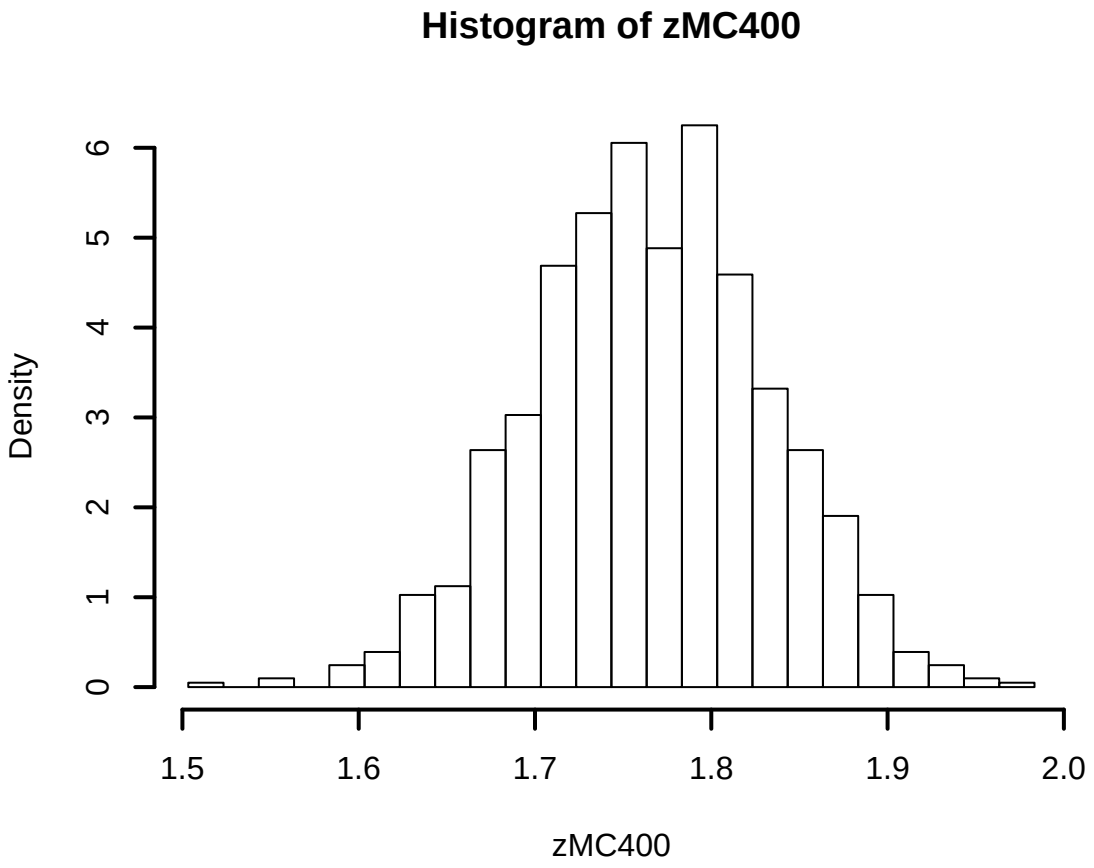
To get the standard deviation when our sample size is 400, we simply take

$$SD(Z_{400}^{MC}) = \frac{SD(\phi(X))}{\sqrt{400}} \approx 0.068932.$$

c. (3 Pts) With sample size $N = 400$, use R to simulate 1024 Monte Carlo estimates of I. Make a histogram of those 1024 estimates and report the mean and the standard deviation of those 1024 estimates.

We can run the following code in R:

```
phi = function(x) {
  4*exp(-(x-2)^2)
}

zMC = function(sam, rep) {
  obs = numeric(rep)
  for (i in 0:rep) {
    obs[i] = sum(phi(runif(sam, 0, 4)))/sam
  }
  obs
}

zMC400 = zMC(400, 1024)
hist(zMC400, breaks=seq(min(zMC400), max(zMC400+0.02), .02), prob=T, lwd=2)
```



The mean is

```
[1] 1.765893
```

The standard deviation is

```
[1] 0.06683839
```

These are pretty close to our calculations!

d. (2 Pts) If sample size $N = 400$, how likely is the Monte Carlo estimator Z_{400}^{MC} to yield an estimate of I within error ± 0.01? Use R to simulate this probability with 10000 runs.

We can run the following code in R to experimentally find this probability:

```
I = 1.76416
zMC400 = zMC(400, 10000)
length(zMC400[0.01 >= abs(I - zMC400)])/10000
```

```
[1] 0.1121
```

We see that we have about a 12% chance of being accurate to two decimal places. We can do better.

e. (2 Pts) How large should N be for the Monte Carlo estimator Z_N^{MC} to yield an estimate of I within error ± 0.01 with probability 95%?

We know from class that

$$N \geq \left(\frac{1.96\sigma}{\epsilon}\right)^2,$$

$$N \geq \left(\frac{1.96(1.37865)}{0.01}\right)^2 \approx 73,017.$$

Let's try this with $N = 80,000$ to see if we are accurate to two decimal places

```
zMC_better = zMC(80000, 1024)
mean(zMC_better)
```

```
[1] 1.763987
```

This looks good! Just to confirm, let's test it with some code similar to what we wrote in part (d),

```
length(zMC_better[0.01 >= abs(I - zMC_better)])/1024
```

```
[1] 0.9648438
```

We see that about 96% of our estimates for I are accurate to two decimal places. This is more acceptable.

f. (3 Pts) Alice simulates $\phi(X_1), \phi(X_2), \ldots, \phi(X_{400})$ and reports that the 400 values have a sample mean of 1.8413 and a sample standard deviation of 1.4078. Construct a 95% confidence interval for the true value I based on Alice's sample.

Recall from class 18 that a 95% confidence interval for some sample population is given by

$$\overline{Y} \pm \frac{1.96(S)}{\sqrt{n}},$$

where $\overline{Y} = 1.8413$, $S = 1.4078$, and $n = 400$. Thus we have

$$1.8413 \pm \frac{1.96(1.4078)}{\sqrt{400}}$$

Thus a 95% confidence interval for I based on Alice's sample is $[1.70334, 1.97926]$.

g. (3 Pts) Bob simulates $\phi(X_1), \phi(X_2), \ldots, \phi(X_{400})$ and reports that the 400 values have a sample mean of 1.8742 and a sample standard deviation of 1.3485. Test whether Bob overestimates the true value of I at a significance level $\alpha = 5\%$. Report your test statistic and the associated P-value.

We calculate the test statistic by

$$t - stat = \frac{\overline{Y} - \mu_0}{\frac{S}{\sqrt{n}}},$$

$$t - stat \approx \frac{1.8742 - 1.76416}{\frac{1.3485}{\sqrt{400}}} \approx 1.632.$$

We can get our P-value by using our test-statistic in R:

```
pt(-1.632, 399)
```

```
[1] 0.05173421
```

Thus we see that our P-value is larger than our specified α, so we do not have enough evidence to reject our null hypothesis that the mean Bob found is the true mean. Thus we cannot say that Bob is overestimating the true value of I.

Problem 3. (10 Pts)

Next, we estimate I by throwing darts into the rectangle $\{(u, v) \mid 0 \leq u \leq 4; 0 \leq v \leq 1\}$. That is, a throw consists of generating two random numbers $U \overset{d}{\sim}$ Uniform(0, 4) and $V \overset{d}{\sim}$ Uniform(0, 1) independently. The ith throw T_i is called a hit if $V \leq e^{-(U-2)^2}$; otherwise it is called a miss.

a. (2 Pts) Use this throwing darts idea to construct a Monte Carlo estimator, labeled W_N^{MC} with N as the sample size. Clearly explain how you construct W_N^{MC}.

Our Monte Carlo estimator W_N^{MC} sums the number of hits we get and divides by the number of throws to get the ratio between the area underneath the curve $e^{-(x-2)^2}$ and the rectangle specified in our problem. We then multiply this ratio by the area of the rectangle to get the area underneath the curve $e^{-(x-2)^2}$. Thus, our Monte Carlo estimator is given by

$$I \approx W_N^{MC} = 4\left(\frac{\phi(X_1, Y_1) + \phi(X_2, Y_2) + \cdots + \phi(X_N, Y_N)}{N}\right),$$

where $\phi(X, Y)$ is given by

$$\phi(X, Y) = \begin{cases} 1 & Y \leq e^{-(X-2)^2}; \\ 0 & \text{otherwise.} \end{cases}$$

We also have $X \overset{d}{\sim}$ Uniform(0, 4) and $Y \overset{d}{\sim}$ Uniform(0, 1).

b. (3 Pts) If sample size $N = 400$, what is the mean and the standard deviation of the Monte Carlo estimator W_N^{MC}?

We know that our mean is I because of how we have constructed W_N^{MC}. We know

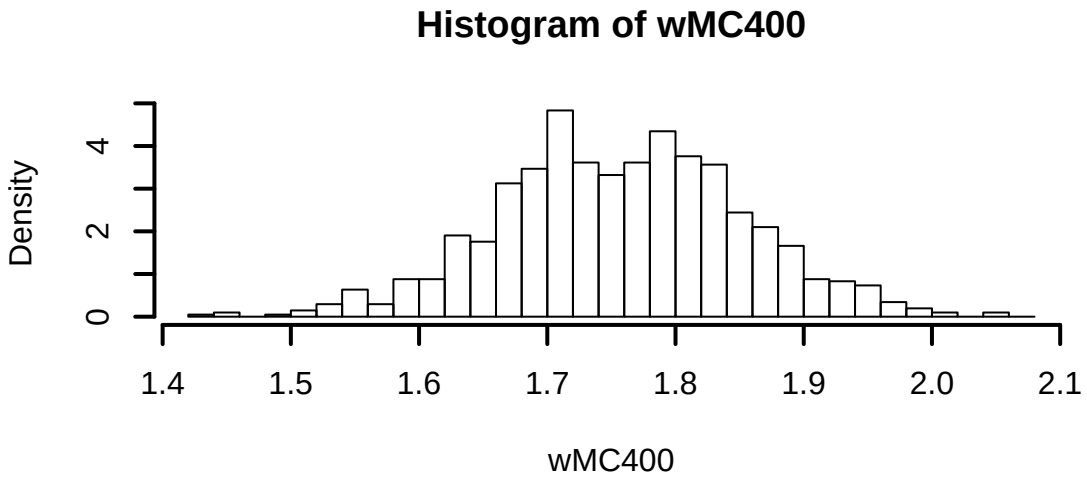
$$Var[W_1^{MC}] = (1 - I)^2\left(\frac{I}{4}\right) + (0 - I)^2\left(1 - \frac{I}{4}\right),$$

$$SD[W_1^{MC}] = \sqrt{(1-I)^2\left(\frac{I}{4}\right) + (0-I)^2\left(1-\frac{I}{4}\right)} \approx 1.99717,$$

$$SD[W_{400}^{MC}] = \frac{1}{\sqrt{400}} SD[W_1^{MC}] \approx 0.09986.$$

c. (3 Pts) With sample size $N = 400$, use R to simulate 1024 Monte Carlo estimates of I. Make a histogram of those 1024 Monte Carlo estimates and report the mean and the standard deviation of those 1024 estimates.

```
wMC = function(sam, rep) {
  obs = numeric(rep)
  for (i in 0:rep) {
    u = runif(sam, 0, 4)
    v = runif(sam, 0, 1)
    obs[i] = 4*length(v[v <= exp(-(u-2)^2)])/sam
  }
  obs
}

wMC400 = wMC(400, 1024)
hist(wMC400, breaks=seq(min(wMC400), max(wMC400+0.02), .02), prob=T, lwd=2)
```



The mean of our simulation is

```
[1] 1.76252
```

The standard deviation of our simulation is

```
[1] 0.09514767
```

These both agree with our previous calculations!

d. (2 Pts) If sample size $N = 400$, how likely is the Monte Carlo estimate W_{400}^{MC} to yield an estimate of I within error ± 0.01? Use R to simulate this probability.

We run the following code in R to simulate this probability

```
length(wMC400[0.01 >= abs(I - wMC400)])/1024
```

```
[1] 0.0703125
```

Thus we see we have about a 7% chance of estimating I with two decimal places of accuracy. This is not great.

Problem 4. (12 Pts)

Now we consider estimating I by using importance sampling techniques. We rewrite

$$I = \int_0^4 \frac{e^{-(x-2)^2}}{g(x)} g(x)dx,$$

where $g(x)$ is a probability density function given by

$$g(x) = \begin{cases} \frac{1}{5} & 0 \leq x < 1; \\ \frac{3}{10} & 1 \leq x < 3; \\ \frac{1}{5} & 3 \leq x < 4; \\ 0 & \text{otherwise.} \end{cases}$$

a. (2 Pts) Based on the choice of the p.d.f. $g(\cdot)$, construct an importance sampling estimator and label this estimator by Z_N^{IS}, where N is the sample size. Explain how you construct Z_N^{IS}.

First we rewrite

$$I = \int_0^4 g(x)\Psi(x)dx,$$

where $\Psi(x) = \frac{e^{-(x-2)^2}}{g(x)}$ for convenience. We construct our Monte Carlo estimator for I by

$$I \approx Z_N^{IS} = \frac{\Phi(X_1) + \Phi(X_2) + \cdots + \Phi(X_N)}{N},$$

where $X_1, X_2, \ldots, X_N \overset{iid}{\sim} g(\cdot)$.

b. (3 Pts) If sample size $N = 400$, what are the mean and the standard deviation of the importance sampling estimator Z_{400}^{IS}?

We know that our mean is given by I as long as our estimator is correct. To calculate the standard deviation we must first calculate our variance by

$$\sigma_\Phi^2 = Var[\Phi(X)] = E[\Phi(X)^2] - E[\Phi(X)]^2,$$

$$\sigma_\Phi^2 = E[\Phi(X)^2] - I^2.$$

We have to do some work before we can ask Mathematica to compute a numerical value for us:

$$E[\Phi(X)^2] = \int_0^4 \frac{1}{g(x)} e^{-2(x-2)^2} dx,$$

$$E[\Phi(X)^2] = \int_0^1 5e^{-2(x-2)^2}dx + \int_1^3 \frac{10}{3}e^{-2(x-2)^2}dx + \int_3^4 5e^{-2(x-2)^2}dx.$$

We can now ask Mathematica to compute this last line.

```
I_2  = Integrate[5e^(-2*(x - 2)^2), {x, 0, 1}] + Integrate[(10/3)*e^(-2*(x - 2)^2), {x, 1, 3}]
I_2 += Integrate[5e^(-2*(x - 2)^2), {x, 3, 4}]
I_2 = N[I_2, 6]
4.27236
```

We then see that

$$Var[Z_1^{IS}] \approx 4.27236 - (1.76416)^2 \approx 1.16010,$$

$$SD[Z_1^{IS}] = \sqrt{Var[Z_1^{IS}]} \approx 1.07708,$$

$$SD[Z_{400}^{IS}] = \frac{1}{\sqrt{400}} SD[Z_1^{IS}] \approx 0.05385.$$

c. (4 Pts) If we use the CDF inversion sampling method to sample values from the p.d.f. $g(x)$, find the cumulative distribution function $G(\cdot)$ and the inverse $G^{-1}(\cdot)$.

Our cumulative distribution function $G(\cdot)$ can be found by taking the integral of $g(\cdot)$ from negative infinity to x. Doing this we see that

$$G(x) = \begin{cases} 0 & x < 0; \\ \frac{x}{5} & 0 \leq x < 1; \\ \frac{1}{5} + \frac{3}{10}(x-1) & 1 \leq x < 3; \\ \frac{4}{5} + \frac{1}{5}(x-3) & 3 \leq x \leq 4; \\ 1 & x > 4. \end{cases}$$

Now we can calculate $G^{-1}(\cdot)$ by setting $y = G(x)$ and solving for x. Doing this we see that

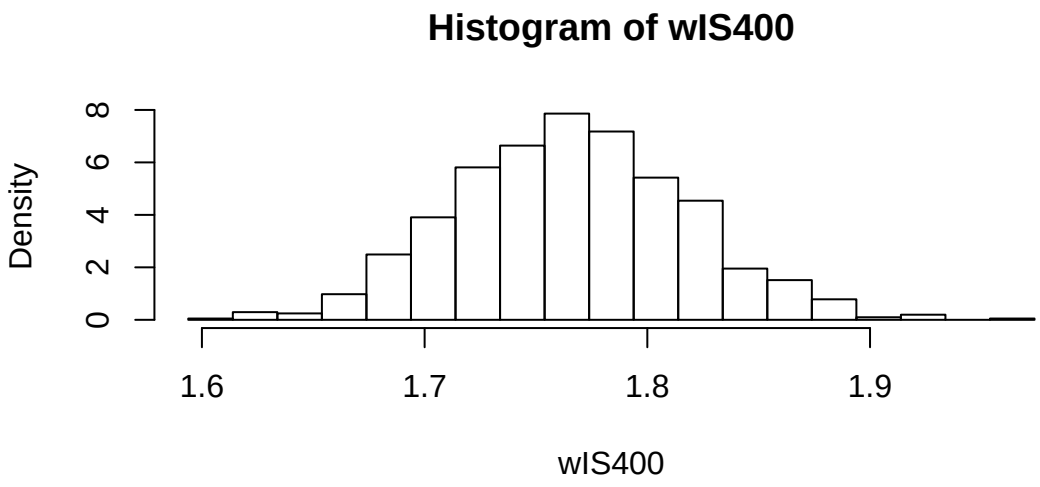
$$G^{-1}(y) = \begin{cases} 5y & 0 \leq y < \frac{1}{5}; \\ \frac{10}{3}\left(y - \frac{1}{5}\right) + 1 & \frac{1}{5} \leq y < \frac{4}{5}; \\ 5\left(y - \frac{4}{5}\right) + 3 & \frac{4}{5} \leq y < 1. \end{cases}$$

d. (3 Pts) With sample size $N = 400$, use the CDF inversion sampling method to simulate 1024 importance sampling estimates of I. Make a histogram of those 1024 importance sampling estimates, and report the mean and the standard deviation of those 1024 estimates.

We can run the following code in R:

```
gpdf <- function(x) { # define p.d.f. g(.) #
  ifelse((x >= 0 & x < 1), 1/5,
    ifelse((x >= 1 & x < 3), 3/10,
      ifelse((x >= 3 & x <= 4), 1/5, 0
      )
    )
  )
}

Ginv <- function(y) { # define c.d.f. inverse G^(-1)(.) #
    ifelse((y >= 0 & y < 1/5), 5*y,
      ifelse((y >= 1/5 & y < 4/5), 10*(y - 1/5)/3 + 1,
        ifelse((y >= 4/5 & y <= 1), 5*(y - 4/5) + 3, 0
```

```
        )
      )
  )
}

cdfInv <- function(n) { # CDF inversion sampling #
  U <- runif(n);
  Ginv(U);
}

clt <- function(sam, rep){
  obs <- NULL;
  for (i in 1:rep){
    y <- cdfInv(sam);
    psi <- exp(-(y-2)^2) / gpdf(y);
    psibar <- mean(psi);
    obs <- c(obs, psibar);
  }
  obs;
}

wIS400 = clt(400, 1024)
hist(wIS400, breaks=seq(min(wIS400), max(wIS400)+0.02, 0.02), prob=T)
```



The mean is

```
[1] 1.765919
```

The Standard Deviation is

```
[1] 0.05210682
```

This is what we expect! Wonderful!

Conclusion

To conclude this exam, I'd like to take a moment to compare the effectiveness of the Monte Carlo methods we've used to approximate

$$I = \int_0^4 e^{-(x-2)^2} dx \approx 1.76416.$$

In problem 2 we used a relatively standard Monte Carlo estimation method which involved multiplying our integrand by $\frac{4}{4}$ and taking the average value of 400 random values of $4e^{-(x-2)^2}$, where $X \overset{d}{\sim}$ Uniform0, 1. This resulted in a theoretical standard deviation of about 0.07 for our sample size of 400.

In problem 3 we examined the more conceptually-clear approach of randomly tossing darts, and assigning each throw as either a hit or a miss. We then tallied the number of hits and misses and used the ratio of hits to tosses to get an approximation for the percentage of the area of our integral to the area of the rectangle we tossed darts into. We then multiplied this ratio by the area of our rectangle to get an approximation for the area of our integral. Doing this, we found that our theoretical standard deviation was around 0.1 for our sample size of 400 throws.

Lastly, we looked at a Monte Carlo integration technique called importance sampling. This technique utilizes some knowledge that we have about our integrand to weight certain areas of our Monte Carlo simulation more than other areas. We chose the p.d.f. $g(x)$ as our "weight" function and applied the c.d.f inversion method to approximate our integral. We found that with this method we achieved a theoretical standard deviation of about 0.05 for our sample size of 400.

Thus we see that our importance sampling method is clearly the most accurate & precise Monte Carlo method that we employed to tackle our integral. One downside of this method is that it required some knowledge on our part about the function we were integrating, because we had to define a weight function to help us increase our accuracy. The other two methods did not require us to know the shape of our function we were integrating, which is nice because these other methods are easy to generalize to much more complicated integrals. In particular, I believe that the Monte Carlo method we used in problem 3 is particularly effective in higher-dimensional spaces, as all that is required is the area that we are tossing darts into. This method has the downside of being slightly less accurate, but for large sample sizes I think this is not significant.

This concludes my Exam #2.