Homework 4

Carly Greutert

5 March 2023

Note: If you are working with a partner, please submit only one homework per group with both names and whether you are taking the course for graduate credit or not. Submit your Rmarkdown (.Rmd) and the compiled pdf on Gauchospace.

Problem 1. Frequentist Coverage of The Bayesian Posterior Interval.

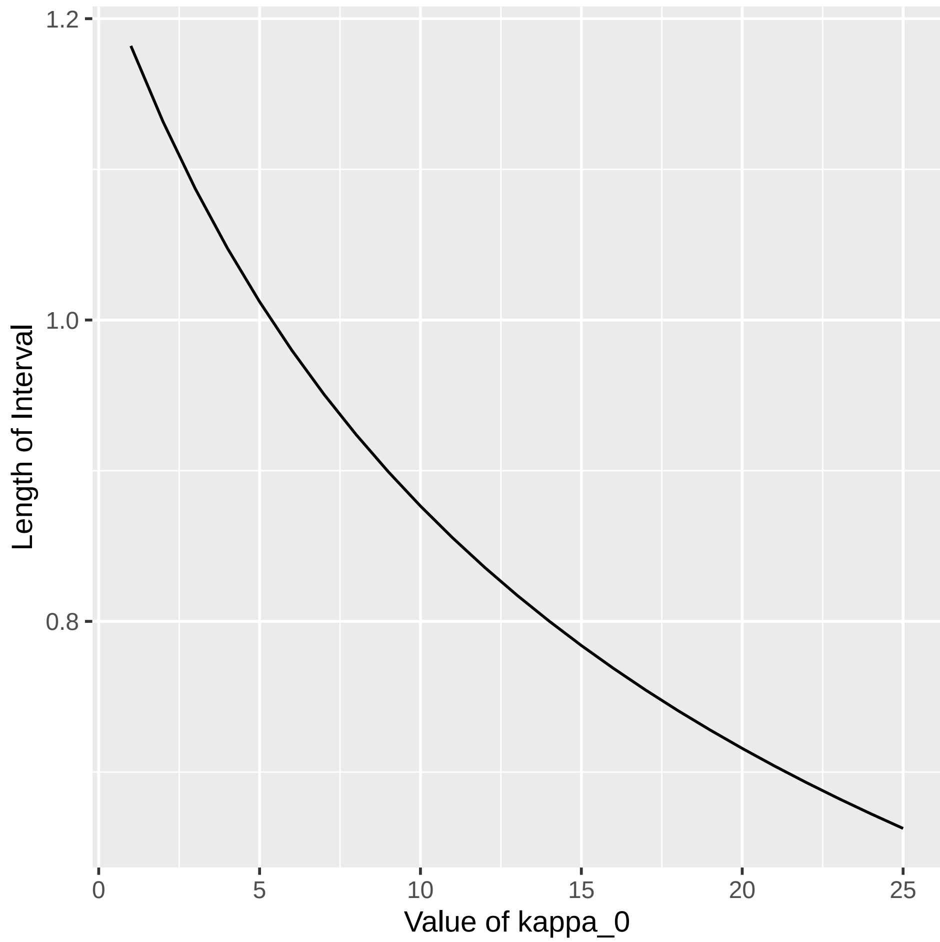
In the "random facts calibration game" we explored the importance and difficulty of well-calibrated prior distributions by examining the calibration of subjective intervals. Suppose that $y_1, .., y_n$ is an IID sample from a $Normal(\mu, 1)$. We wish to estimate μ.

1a. For Bayesian inference, we will assume the prior distribution $\mu \sim Normal(0, \frac{1}{\kappa_0})$ for all parts below. Remember, from lecture that we can interpret κ_0 as the pseudo-number of prior observations with sample mean $\mu_0 = 0$. State the posterior distribution of μ given $y_1, .., y_n$. Report the lower and upper bounds of the 95% quantile-based posterior credible interval for μ, using the fact that for a normal distribution with standard eviation σ, approximately 95% of the mass is between $\pm 1.96\sigma$.

$\mu|Y \sim$ N(μ_n, τ_n^2) where $\mu_n = \frac{\frac{1}{\frac{1}{\kappa_0}}*0+\frac{n}{1}*\bar{y}}{\frac{1}{\frac{1}{\kappa_0}}+\frac{n}{1}} = \frac{\kappa_0*0+n\bar{y}}{\kappa_0+n} = \frac{n\bar{y}}{\kappa_0+n}$
and $\tau_n^2 = \frac{1}{\frac{1}{\frac{1}{\kappa_0}}+\frac{n}{1}} = \frac{1}{\kappa_0+n}$
so $\mu|Y\ N(\frac{n\bar{y}}{\kappa_0+n}, \frac{1}{\kappa_0+n})$
The lower bound of the 95% quantile-based posterior credible interval is $\frac{n\bar{y}}{\kappa_0+n} - 1.96 * \sqrt{\frac{1}{\kappa_0+n}}$
The upper bound is $\frac{n\bar{y}}{\kappa_0+n} + 1.96 * \sqrt{\frac{1}{\kappa_0+n}}$

1b. Plot the length of the posterior credible interval as a function of κ_0, for $\kappa_0 = 1, 2, ..., 25$ assuming $n = 10$. Report how this prior parameter effects the length of the posterior interval and why this makes intuitive sense.

```
# Use 'interval_length' to store lengths of credible intervals
interval_length <- numeric()
for (kappa in 1:25){
  interval_length[kappa] = 2 * 1.96 * sqrt(1/(kappa + 10))
}

ggplot(tibble(interval_length), aes(x= 1:25, y = interval_length)) +
  xlab("Value of kappa_0") + ylab("Length of Interval") +
  geom_line()
```




```
. = ottr::check("tests/q1b.R")
```

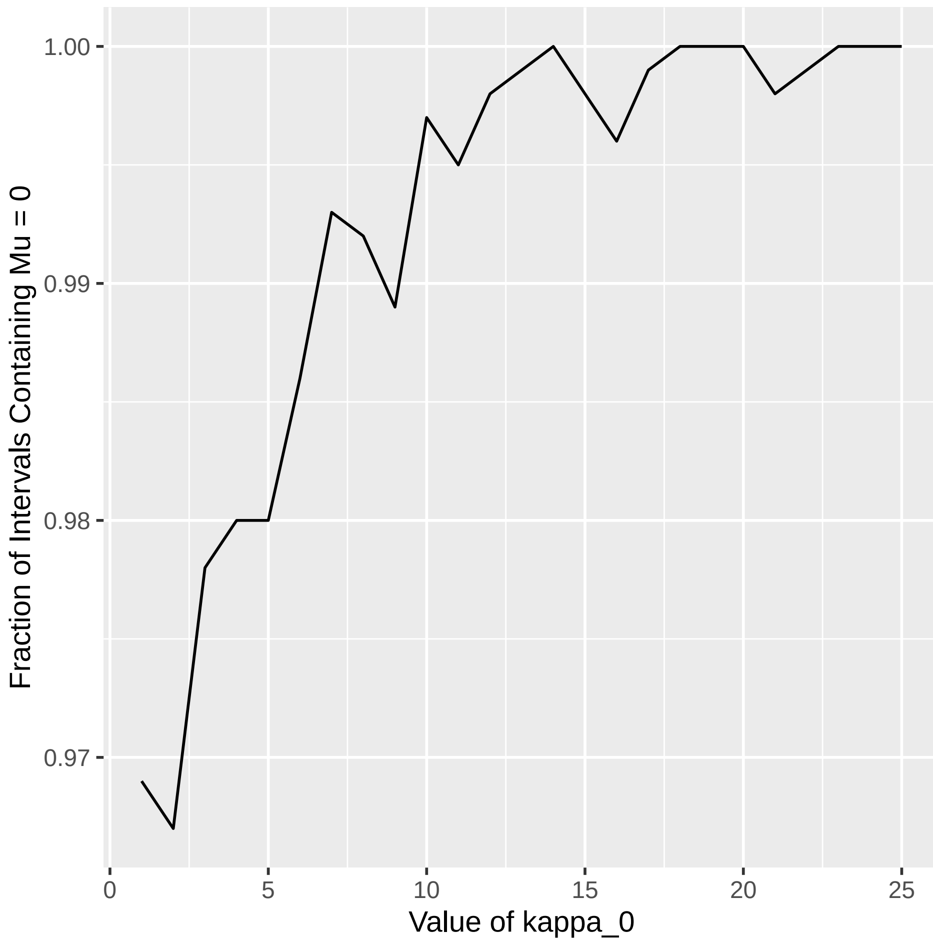
```
##
## All tests passed!
```

As the prior parameter, κ_0 increases, the posterior credible interval decreases. This makes intuitive sense because having a higher value of the prior parameter typically signals confidence/a better prediction about μ and therefore we can shorten the interval to a more accurate guess.

1c. Now we will evaluate the *frequentist coverage* of the posterior credible interval on simulated data. Generate 1000 data sets where the true value of $\mu = 0$ and $n = 10$. For each dataset, compute the posterior 95% interval endpoints (from the previous part) and see if it the interval covers the true value of $\mu = 0$. Compute the frequentist coverage as the fraction of these 1000 posterior 95% credible intervals that contain $\mu = 0$. Do this for each value of $\kappa_0 = 1, 2, ..., 25$. Plot the coverage as a function of κ_0. Store these 25 coverage values in vector called `coverage`.

```
## Fill in the vector called "coverage", which stores the fraction of intervals containing \mu = 0 for e
covers <- c()
coverage <- numeric(25)
for (kappa in 1:25){
for (i in 1:1000){
  yi = rnorm(n = 10, mean = 0, sd = 1)
  lower = (10*mean(yi))/(kappa + 10) - 1.96 * sqrt(1/(kappa + 10))
  upper = (10*mean(yi))/(kappa + 10) + 1.96 * sqrt(1/(kappa + 10))
  covers[i] = between(0, lower, upper)
}
```

```
  coverage[kappa] = sum(covers)/1000
  }
ggplot(tibble(coverage), aes(x= 1:25, y = coverage)) +
  xlab("Value of kappa_0") + ylab("Fraction of Intervals Containing Mu = 0") +
  geom_line()
```



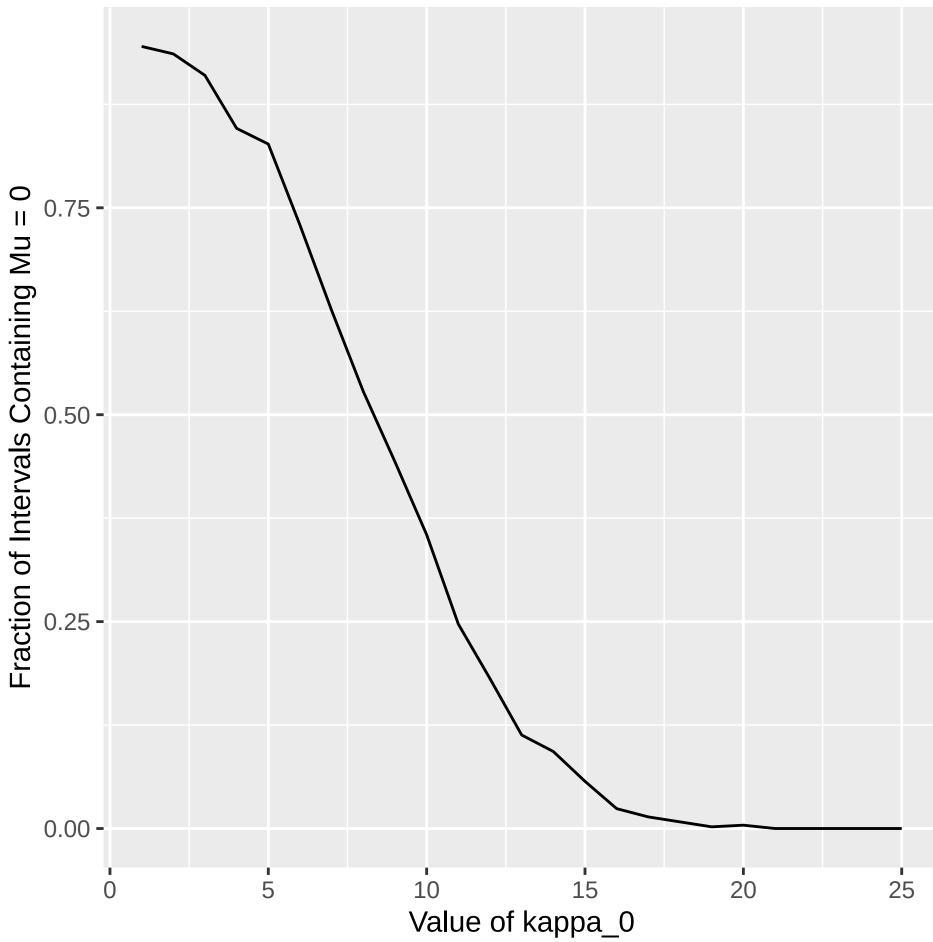

```
. = ottr::check("tests/q1c.R")
```

```
##
## All tests passed!
```

1d. Repeat 1c but now generate data assuming the true $\mu = 1$. Again, store these 25 coverage values in vector called `coverage`.

```
## Fill in the vector called "coverage", which stores the fraction of intervals containing \mu = 1 for e
coverage <- numeric(25)
covers <- c()

for (kappa in 1:25){
for (i in 1:1000){
  yi = rnorm(n = 10, mean = 1, sd = 1)
  lower = (10*mean(yi))/(kappa + 10) - 1.96 * sqrt(1/(kappa + 10))
  upper = (10*mean(yi))/(kappa + 10) + 1.96 * sqrt(1/(kappa + 10))
  covers[i] = between(1, lower, upper)
}
```

```
  coverage[kappa] = sum(covers)/1000
  }
ggplot(tibble(coverage), aes(x= 1:25, y = coverage)) +
  xlab("Value of kappa_0") + ylab("Fraction of Intervals Containing Mu = 0") +
  geom_line()
```




```
. = ottr::check("tests/q1d.R")
```

```
##
## All tests passed!
```

1e. Explain the differences between the coverage plots when the true $\mu = 0$ and the true $\mu = 1$. For what values of κ_0 do you see closer to nominal coverage (i.e. 95%)? For what values does your posterior interval tend to overcover (the interval covers the true value more than 95% of the time)? Undercover (the interval covers the true value less than 95% of the time)? Why does this make sense?

When $\mu = 0$, the higher value of κ_0 leads to a higher coverage value whereas when $\mu = 1$, the reverse is true, the higher value of κ_0 leads to a lower rate of coverage. When $\mu = 0$, all values of κ_0 lead to above 95% coverage, so they all overcover. When $\mu = 1$, $\kappa_0 = 1, 2$ leads to almost 95% coverage, but the rest undercover. This makes sense because the variance is equivalent to $\frac{1}{\kappa_0}$, so the variance shrinks towards zero as we get more confident in our prior, so our interval is less likely to cover 1 as much as it will 0.

Problem 2. Goal Scoring in the Women's World Cup

Let's take another look at scoring in soccer. The Chinese Women's soccer team recently won the AFC Women's Asian Cup. Suppose you are interested in studying the World Cup performance of this soccer team. Let λ be the be the average number of goals scored by the team. We will analyze λ using the Gamma-Poisson model where data Y_i is the observed number of goals scored in the ith World Cup game, ie. we have $Y_i|\lambda \sim Pois(\lambda)$. *A priori*, we expect the rate of goal scoring to be $\lambda \sim Gamma(a, b)$. According to a sports analyst, they believe that λ follows a Gamma distribution with $a = 1$ and $b = 0.25$.

2a. Compute the theoretical posterior parameters a, b, and also the posterior mean.

```
y <- c(4, 7, 3, 2, 3) # Number of goals in each game

post_a <- sum(y) + 1
post_b <- length(y) + 0.25
post_mu <- post_a/post_b
```

```
. = ottr::check("tests/q2a.R")
```

```
##
## All tests passed!
```

2b. Create a new Stan file by selecting "Stan file" and name it `women_cup.stan`. Encode the Poisson-Gamma model in Stan. Use `cmdstanr` to report and estimate the posterior mean of the scoring rate by computing the sample average of all Monte Carlo samples of λ.

```
## Create "women_cup.stan" yourself and fill in the model
soccer_model <- cmdstan_model("women_cup.stan")

## This fits the model to data y
## All parameter samples are stored in a data frame called "samples"
stan_fit <- soccer_model$sample(data=list(N = length(y), Y = y), refresh=0, show_messages = FALSE)
```

```
## Running MCMC with 4 sequential chains...
##
## Chain 1 finished in 0.0 seconds.
## Chain 2 finished in 0.0 seconds.
## Chain 3 finished in 0.0 seconds.
## Chain 4 finished in 0.0 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.0 seconds.
## Total execution time: 0.6 seconds.
```

```
samples <- stan_fit$draws(format="df")

## Compute the posterior mean of the lambda samples

post_mean <- stan_fit$summary()$mean[2]
```

```
. = ottr::check("tests/q2b.R")
```

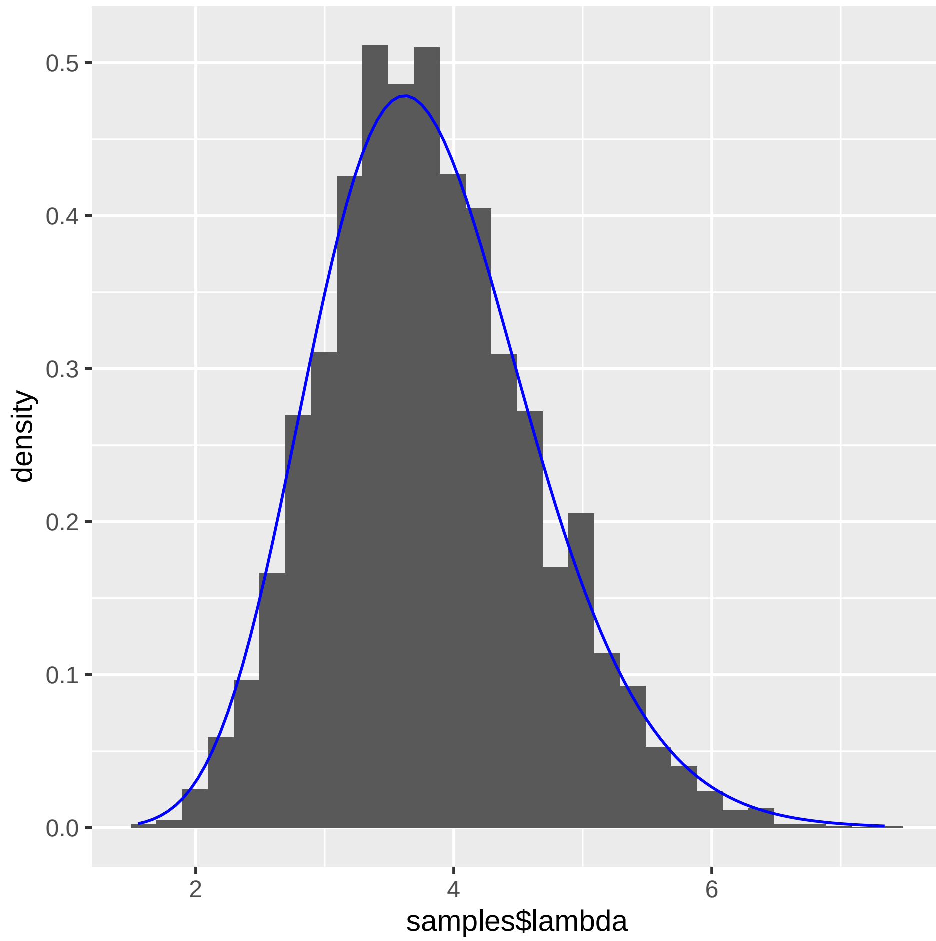
```
##
## All tests passed!
```

2c. Create a histogram of the Monte Carlo samples of λ and add a line showing the theoretical posterior of density of λ. Do the Monte Carlo samples coincide with the theoretical density?

```r
f1 <- function(x){dgamma(x, post_a, post_b)}
ggplot(tibble(samples$lambda), aes(x = samples$lambda)) + geom_histogram(aes(y = ..density..)) +  stat_
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2d. Use the Monte Carlo samples from Stan to compute the mean of predicative posterior distribution to estimate the distribution of expected goals scored for next game played by the Chinese women's soccer team.

```r
pred_mean <- mean(rpois(n = length(samples$lambda), lambda = samples$lambda))
```

```r
. = ottr::check("tests/q2d.R")
```

```
##
## All tests passed!
```

Problem 3. Bayesian inference for the normal distribution in Stan.

Create a new Stan file and name it `IQ_model.stan`. We will make some basic modifications to the template example in the default Stan file for this problem. Consider the IQ example used from class. Scoring on IQ tests is designed to yield a N(100, 15) distribution for the general population. We observe IQ scores for a sample of n individuals from a particular town, $y_1, \ldots y_n \sim N(\mu, \sigma^2)$. Our goal is to estimate the population mean in the town. Assume the $p(\mu, \sigma) = p(\mu \mid \sigma)p(\sigma)$, where $p(\mu \mid \sigma)$ is $N(\mu_0, \sigma/\sqrt{\kappa_0})$ and $p(\sigma)$ is Gamma(a, b). Before you administer the IQ test you believe the town is no different than the rest of the population, so you assume a prior mean for μ of $\mu_0 = 100$, but you aren't to sure about this a priori and so you set $\kappa_0 = 1$ (the effective number of pseudo-observations). Similarly, a priori you assume σ has a mean of 15 (to match the intended standard deviation of the IQ test) and so you decide on setting $a = 15$ and $b = 1$ (remember, the mean of a Gamma is a/b). Assume the following IQ scores are observed:

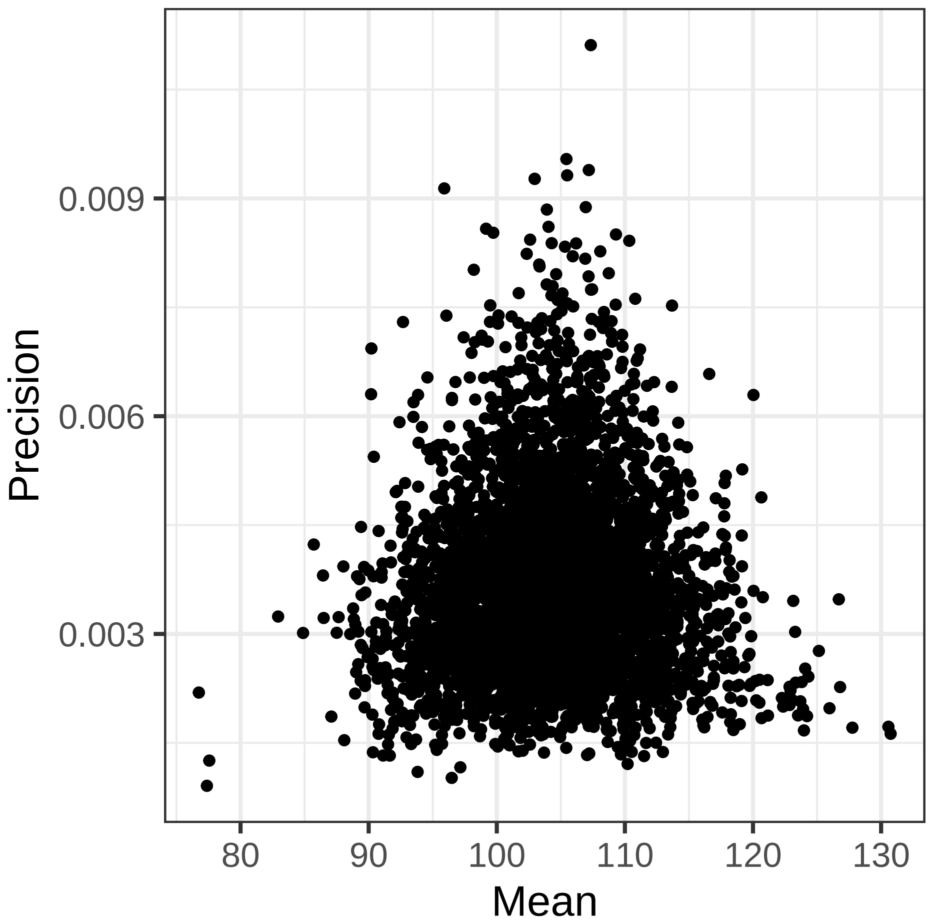
```
y <- c(70, 85, 111, 111, 115, 120, 123)
n <- length(y)
```

3a. Make a scatter plot of the posterior distribution of the mean, μ, and the precision, $1/\sigma^2$. Put μ on the x-axis and $1/\sigma^2$ on the y-axis. What is the posterior relationship between μ and $1/\sigma^2$? Why does this make sense? *Hint:* review the lecture notes.

```
normal_stan_model <- cmdstan_model("IQ_model.stan")

# Run rstan and extract the samples
stan_fit <- normal_stan_model$sample(data=list(N = n, y = y, mu0 = 100, k0 = 1), refresh=0, show_messag
```

```
## Running MCMC with 4 sequential chains...
##
## Chain 1 finished in 0.0 seconds.
## Chain 2 finished in 0.0 seconds.
## Chain 3 finished in 0.0 seconds.
## Chain 4 finished in 0.0 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.0 seconds.
## Total execution time: 0.4 seconds.
```

```
samples <- stan_fit$draws(format="df")

mu_samples <- samples$mu
sigma_samples <- samples$sigma
precision_samples <- 1/((sigma_samples)^2)

## Make the plot
tibble(Mean = mu_samples, Precision=precision_samples) %>%
 ggplot() +
 geom_point(aes(x=Mean, y=Precision)) +
 theme_bw(base_size=16)
```




```r
. = ottr::check("tests/q3a.R")
```

```
##
## All tests passed!
```

I notice that when precision is high, there is a large sampling variance and when it is low, there is low sampling variance. This makes intuitive sense because precision is the reciprocal of the variance, so when variance is high, the precision will be low, they have an inverted relationship. To continue, a lower sampling variance is indicative of higher confidence in μ so their values are spread further apart.

3b. You are interested in whether the mean IQ in the town is greater than the mean IQ in the overall population. Use Stan to find the posterior probability that μ is greater than 100.

```r
mean(mu_samples > 100)
```

```
## [1] 0.76525
```

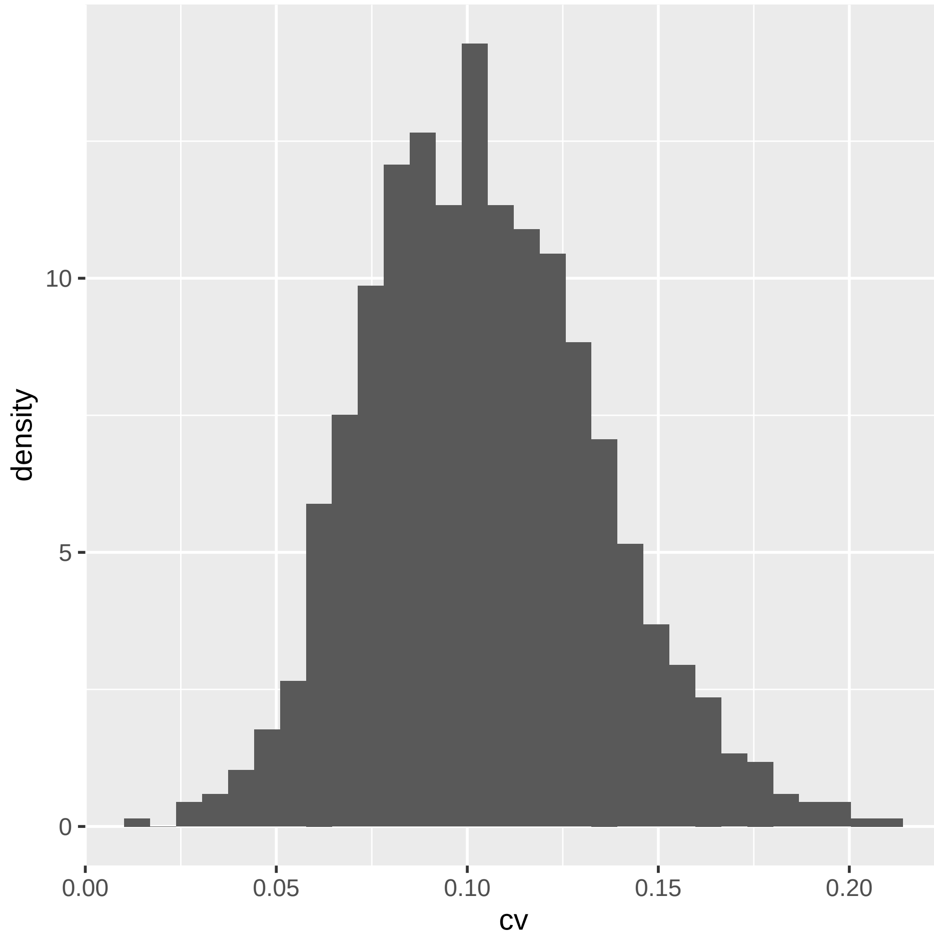
Out of all the generated posterior μ samples, it appears that ~76.9% of the town IQ is greater than the mean of the overall population.

3c. The coefficient of variation, $c_v = \sigma/\mu$ is defined as the standard deviation over the mean. Make a histogram of $p(c_v \mid y)$ from Monte Carlo samples and report the posterior mean and the lower and upper endpoints of the 95% quantile based interval.

```r
cv <- c()
for (s in 1:1000){
  mu <- rgamma(n, 15 + sum(y), 1 + n)
  tilde_Y <- rpois(n, mu)
```

```r
  cv <- append(cv, sd(tilde_Y)/mean(tilde_Y))
}
ggplot(tibble(cv), aes(x = cv)) + geom_histogram(aes(y = ..density..))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
mean(cv)
```

```
## [1] 0.1040499
```

```r
quantile(cv, 0.05)
```

```
##         5%
## 0.05902146
```

```r
quantile(cv, 0.95)
```

```
##       95%
## 0.1587091
```