14/12

Applied Bayesian modeling - HW2

Álvaro J. Castro Rivadeneira

September 16, 2022

Score: The maximum number of points in this HW is 12 points, with 3 points extra credit. For calculating a final HW grade, the points will be rescaled to a maximum score of (12+3)/12*100% = 125%.

What to hand in: For any exercises using R, we need an Rmd and a knitted pdf. You can add the answers to other exercises in the same markdown file or create them in a different way (as long as it's legible). if you create them outstide markdown, please combine all answers into one pdf.

Part 1 - based on module 4 $\boxed{3}$

Exercise 1: Derivation of a posterior using Bayes' rule [3 pts]

This exercise is about the material in module 4.

Obtain $p(\mu|\boldsymbol{y})$ when prior and data are defined as follows:

$$\begin{aligned} y_i|\theta_i,\sigma^2 &\sim N(\theta_i,\sigma^2)(\text{independent}), \text{ for } i=1,2,\ldots,n; \\ \theta_i &= \mu + r_i; \\ \mu &\sim N(m_0, s^2_{\mu0}); \end{aligned}$$

where r_i refers to a known/fixed reporting error (that varies across observations) and σ^2 is known.

Note that you can treat the r_i as fixed, not random.
(A more realistic but somewhat more difficult exercise would be to have r_i be random, and assume that they are normally distributed and independent of the y_i's).

Answer:
If every y_i is normally distributed around θ_i, then $\bar{y}$ is also normally distributed around a mean $\bar{\theta}$, with a variance $\frac{\sigma^2}{n}$.

As was explained in class, we can define a variable $z_i = y_i - r_i$ such that:

$$\bar{z} = \frac{1}{i}\cdot\sum_{i=1}^{n}(y_i - r_i) = \frac{1}{i}\cdot\sum_{i=1}^{n}(y_i) - \frac{1}{i}\cdot\sum_{i=1}^{n}(r_i) = \bar{y} - \bar{r}$$

Given this, and that $\mu = \theta_i - r_i$, and that the variance only depends on θ_i since r_i is fixed, then:

$$z_i|\mu,\sigma^2 \sim N(\mu,\sigma^2)$$

So, going back to Bayes' rule, we can derive:

$$p(\mu|z) \propto p(\mu)p(z|\mu)$$

$$p(\mu|z) \propto \exp\left(\frac{-1}{2s^2_{\mu_0}}(\mu - m_0)^2\right)\cdot\exp\left(\frac{-n}{2\sigma^2}(\bar{z}-\mu)^2\right)$$

$$p(\mu|z) \propto \exp\left(-\frac{1}{2}f(\mu)\right)$$

where

$$f(\mu) = 1/s^2_{\mu_0}(\mu^2 - 2\cdot m_0 \cdot \mu) + n/\sigma^2(\mu^2 - 2\cdot \bar{z}\cdot \mu)$$
$$f(\mu) \propto (1/s^2_{\mu_0} + n/\sigma^2)\mu^2 - 2\cdot(1/s^2_{\mu_0}m_0 + n\cdot \bar{z}/\sigma^2)\cdot \mu$$

So that:

$$\mu|z \sim N\left(\frac{m_0/s^2_{\mu_0} + n\cdot \bar{z}/\sigma^2}{1/s^2_{\mu_0} + n/\sigma^2}, \frac{1}{1/s^2_{\mu_0} + n/\sigma^2}\right)$$ ✓

Substituting $\bar{z} = \bar{y} - \bar{r}$ we get:

$$p(\mu|(y-r)) \propto p(\mu)p((y-r)|\mu)$$

Since we assume that r is fixed, then $p(\mu|z) = p(\mu|y,r) = p(\mu|y)$

$$p(\mu|z) = p(\mu|y) \propto \exp\left(\frac{-1}{2s^2_{\mu_0}}(\mu - m_0)^2\right)\cdot \exp\left(\frac{-n}{2\sigma^2}((\bar{y}-\bar{r}) - \mu)^2\right)$$
$$p(\mu|y) \propto \exp\left(-\frac{1}{2}f(\mu)\right)$$

where

$$f(\mu) = 1/s^2_{\mu_0}(\mu^2 - 2\cdot m_0\cdot \mu) + n/\sigma^2(\mu^2 - 2\cdot(\bar{y}-\bar{r})\cdot \mu)$$
$$f(\mu) \propto (1/s^2_{\mu_0} + n/\sigma^2)\mu^2 - 2\cdot(1/s^2_{\mu_0}m_0 + n\cdot(\bar{y}-\bar{r})/\sigma^2)\cdot \mu$$

So that:

$$\mu|y \sim N\left(\frac{m_0/s^2_{\mu_0} + n\cdot(\bar{y}-\bar{r})/\sigma^2}{1/s^2_{\mu_0} + n/\sigma^2}, \frac{1}{1/s^2_{\mu_0} + n/\sigma^2}\right)$$

Exercise 2: Interpretation of the role of prior information for estimating IQ scores

This exercise is about the material in module 4, based on Hoff 5.4 (but using a different prior on μ).

Background: Scoring on IQ tests is designed to produce a normal distribution with a mean of 100 and a standard deviation of 15 (a variance of 225) when applied to the general population.

Suppose we are to sample n individuals from a particular town in the United States and then estimate μ, the town-specific mean IQ score, based on the sample of size n.
Let y_i denote the IQ score for the i-th person in the town of interest, and assume $y_1, y_2, \ldots, y_n|\mu, \sigma^2 \sim N(\mu, \sigma^2)$ (independent), with $\sigma = 15$. Suppose that $\bar{y} = 113$ and $n = 10$.

For Bayesian inference about μ, the following prior will be used:

$$\mu \quad \sim \quad N(m_0, s^2_{\mu 0}),$$

with $m_0 = 100$ and $s_{\mu 0} = \sigma = 15$ (based on the information about IQ scores).

Exercise 2a [5pts] 4.5

(i) Given the information above, obtain the expression for the Bayesian point estimate of μ, $\hat{\mu}_{Bayes} = E(\mu|\boldsymbol{y})$, in terms of m_0, n, $\bar{y}$, and/or σ (don't plug in values yet; note that the expression simplifies relative to the expression we obtained in mdule 4, using only a subset of these).

For the point estimate of μ, $\hat{\mu}_{Bayes} = E(\mu|\boldsymbol{y})$, we can use the mean of the posterior, which is:

$$\hat{\mu}_{Bayes} = \frac{m_0/s^2_{\mu_0} + n \cdot \bar{y}/\sigma^2}{1/s^2_{\mu_0} + n/\sigma^2}$$

Since $s_{\mu_0} = \sigma$, I can substitute the former with the latter to get:

$$\hat{\mu}_{Bayes} = \frac{m_0/\sigma^2 + n \cdot \bar{y}/\sigma^2}{1/\sigma^2 + n/\sigma^2} = \frac{m_0 + n \cdot \bar{y}}{\sigma^2} \cdot \frac{\sigma^2}{(1+n)}$$

$$\hat{\mu}_{Bayes} = \frac{m_0 + n \cdot \bar{y}}{(1+n)}$$ ✓

Which is surprising insofar the variance is removed from the equation.

(ii) Interpret the Bayesian point estimate as the weighted combination of prior information and data.

Since the variance is the same in the prior and the likelihood, then it doesn't contribute to weighting both of these, as it will give equal weight to both. Thus, the Bayesian mean becomes simply an average of all the data points including the prior mean as another data point. This can be seen more clearly if we assumed that $m_0 = \bar{y}$, so that the numerator above becomes $\bar{y} + n \cdot \bar{y} = (1+n) \cdot \bar{y}$. Thus, you would simply end up with $\bar{y}$ as the point estimate. The weight of the prior depends on how many data points there are, and its weight is ultimately just that of another data point.

(iii) Then calculate the value for $\hat{\mu}_{Bayes}$ given the information provided and construct a 95% credible interval for μ.

Plugging in the values, we get

$$\hat{\mu}_{Bayes} = \frac{100 + 10 \cdot 113}{(1+10)} = \frac{1230}{11} = 111.818$$ ✓

```
(mu.hat = (100 + (10*113)) / (1 +10))
```

```
## [1] 111.8182
```

To get the 95% credible interval, I can use:

$$var = \frac{1}{1/s^2_{\mu_0} + n/\sigma^2} = \frac{\sigma^2}{n+1}$$

Need to use posterior variance, -0.5

```
qnorm(c(0.025, 0.975), mean = mu.hat, sd = 15) # 95% quantile-based CI
```

```
## [1]  82.41872 141.21764
```

This means that my 95% credible interval ranges from 82.42 - 141.22, which notably includes this town's mean.

2b Extra credit [3pts] 2.5

We will now compare the sampling properties of the Bayes estimator for μ, $\hat{\mu}_{Bayes} = E(\mu|\boldsymbol{y})$, to the maximum likelihood estimator $\hat{\mu}_{MLE} = \bar{y}$. Sampling properties of estimators refer to their behavior under hypothetically repeatable surveys or experiments, summarized into bias and mean squared error, as explained below with μ^* referring to the (unknown) true value of μ:
- The bias of an estimator is defined as the difference between its expected value (w.r.t. data $\boldsymbol{y}$) and the (unknown) true value:

$$Bias(\hat{\mu}) = E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] - \mu^*$$

(adding the subscript with the expectation here to make it clear that the expectation is wrt $\boldsymbol{y}$, to clarify that the data are what's random here; also adding conditioning on true value μ^*)
- Bias refers to how close the center of mass of the sampling distribution of an estimator is to the true value. An unbiased estimator is an estimator with zero bias, which sounds desirable. However, bias does not tell us how far away an estimate might be from the true value. For example, y_1 is an unbiased estimator of the population mean μ, but will generally be farther away from μ than $\bar{y}$.
- To evaluate how close an estimator $\hat{\mu}$ is likely to be to the true value μ^*, we might use the mean squared error (MSE). Letting $m = E_{\boldsymbol{y}}[\hat{\mu}|\mu^*]$, the MSE of an estimator $\hat{\mu}$ is

$$\begin{aligned}
MSE[\hat{\mu}|\mu^*] &= E_{\boldsymbol{y}}[(\hat{\mu} - \mu^*)^2|\mu^*],\\
&= E_{\boldsymbol{y}}[(\hat{\mu} - m + m - \mu^*)^2|\mu^*],\\
&= E_{\boldsymbol{y}}[(\hat{\mu} - m)^2|\mu^*] + 2E_{\boldsymbol{y}}[(\hat{\mu} - m)(m - \mu^*)] + E[(m - \mu^*)^2|\mu^*],\\
&= E_{\boldsymbol{y}}[(\hat{\mu} - m)^2|\mu^*] + (m - \mu^*)^2,\\
&= Var[\hat{\mu}|\mu^*] + Bias(\hat{\mu})^2.
\end{aligned}$$

This means that, before the data are gathered, the expected distance from the estimator to the true value depends on how close μ^* is to the center of the distribution of $\hat{\mu}$ (the bias), as well as how spread out the distribution is (the variance of $\hat{\mu}$).

Exercise Suppose that (unknown to us) the true mean IQ score μ^* in the town was quite high, $\mu^* = 112$. Calculate the bias, variance and MSE of the Bayes and ML estimators. Which estimator has a larger bias? Which estimator has a larger MSE?
Hint (based on a commonly made mistake): $E_{\boldsymbol{y}}(\hat{\mu}_{Bayes}|\mu^*) \neq \hat{\mu}_{Bayes}$, you need to get the expected value w.r.t. the data $\boldsymbol{y}$.

Answer:
I will use the previously given information that the prior mean is $m_0 = 100$ and $s_{mu0} = \sigma = 15$.

Bayesian Estimator
Bias

$$E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] = \frac{1/s_{\mu 0}^2}{1/s_{\mu 0}^2 + n/\sigma^2} m_0 + \frac{n/\sigma^2}{1/s_{\mu 0}^2 + n/\sigma^2} \bar{y}$$

$$E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] = \frac{1/225}{1/225 + n/225} 100 + \frac{n/225}{1/225 + n/225} \bar{y} = \frac{100}{n+1} + \frac{n \cdot \bar{y}}{n+1}$$

```
bayes_exp_mu <- function(n, ybar){(100/(n+1)) + (n*ybar/(n+1))}
(bayes_exp_mu(10, 113))
```

```
## [1] 111.8182
```

```
(bayes_exp_mu(1000, 113))
```

```
## [1] 112.987
```

```
(bayes_exp_mu(1, 113))
```

```
## [1] 106.5
```

```
(bayes_exp_mu(10, 133))
```

```
## [1] 130
```

```
(bayes_exp_mu(1000, 133))
```

```
## [1] 132.967
```

```
(bayes_exp_mu(1, 133))
```

```
## [1] 116.5
```

```
(bayes_exp_mu(1000, 112))
```

```
## [1] 111.988
```

As n increases, the expected value approximates $\bar{y}$ although it never quite reaches it. When n is small, the expected value is closer to the prior. Since μ^* is a fixed value and not a random variable, the bias becomes:

$$Bias(\hat{\mu}) = E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] - \mu^* = \frac{100 + n \cdot \bar{y}}{n+1} - 112 = \frac{100 + n \cdot \bar{y} - 112n - 112}{n+1}$$

$$Bias(\hat{\mu}) = \frac{n(\bar{y} - 112) - 12}{n+1}$$

```
bayes_bias_mu <- function(n, ybar){((n*(ybar-112))-12)/(n+1)}
(bayes_bias_mu(10, 113))
```

```
## [1] -0.1818182
```

```
(bayes_bias_mu(1000, 113))
```

```
## [1] 0.987013
```

```
(bayes_bias_mu(1, 113))
```

```
## [1] -5.5
```

```r
(bayes_bias_mu(10, 133))
```

```
## [1] 18
```

```r
(bayes_bias_mu(1000, 133))
```

```
## [1] 20.96703
```

```r
(bayes_bias_mu(1, 133))
```

```
## [1] 4.5
```

```r
(bayes_bias_mu(10, 112))
```

```
## [1] -1.090909
```

✓

```r
(bayes_bias_mu(1000, 112))
```

```
## [1] -0.01198801
```

```r
(bayes_bias_mu(1, 112))
```

```
## [1] -6
```

```r
(bayes_bias_mu(10, 224))
```

```
## [1] 100.7273
```

```r
(bayes_bias_mu(1000, 224))
```

```
## [1] 111.8761
```

```r
(bayes_bias_mu(1, 224))
```

```
## [1] 50
```

```r
(bayes_bias_mu(12, 113))
```

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

```r
(bayes_bias_mu(6, 114))
```

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

```
n.bias <- 1
bayes.bias.1 <- cbind(bayes_bias_mu(n.bias, 110), bayes_bias_mu(n.bias, 112), bayes_bias_mu(n.bias, 113
                      bayes_bias_mu(n.bias, 114), bayes_bias_mu(n.bias, 133), bayes_bias_mu(n.bias, 224
n.bias <- 10
bayes.bias.10 <- cbind(bayes_bias_mu(n.bias, 110), bayes_bias_mu(n.bias, 112), bayes_bias_mu(n.bias, 11
                      bayes_bias_mu(n.bias, 114), bayes_bias_mu(n.bias, 133), bayes_bias_mu(n.bias, 224
n.bias <- 1000
bayes.bias.1000 <- cbind(bayes_bias_mu(n.bias, 110), bayes_bias_mu(n.bias, 112), bayes_bias_mu(n.bias,
                      bayes_bias_mu(n.bias, 114), bayes_bias_mu(n.bias, 133), bayes_bias_mu(n.bias, 224
```

This means that the bias is larger the further $\bar{y}$ is from μ^* and that bias often increases as sample size increases, although this doesn't hold when $\bar{y} = \mu^*$. Finally, it's worth noting that bias cannot equal zero when $\bar{y} = \mu^*$, but that only happens only when $\bar{y} - 112 = 12/n$.

Variance
The variance, is given by: Variance of the Bayes estimator instead of posterior variance, -0.5

$$Var[\hat{\mu}|\mu^*] = \underline{E_{\boldsymbol{y}}[(\hat{\mu} - m)^2|\mu^*]} = \frac{1}{1/s^2_{\mu 0} + n/\sigma^2} = \frac{1}{1/225 + n/225} = \frac{225}{n+1}$$

```
bayes_variance <- function(n){225/(n+1)}
(bayes_variance(10))
```

```
## [1] 20.45455
```

```
(bayes_variance(1000))
```

```
## [1] 0.2247752
```

```
(bayes_variance(1))
```

```
## [1] 112.5
```

```
(bayes.variance <- cbind(bayes_variance(1), bayes_variance(10), bayes_variance(1000)))
```

```
##       [,1]     [,2]      [,3]
## [1,] 112.5 20.45455 0.2247752
```

This means that as n increases, the variance decreases, regardless of $\bar{y}$ or the mean (expected or real).

MSE
Finally, for the MSE we use:

$$MSE[\hat{\mu}|\mu^*] = Var[\hat{\mu}|\mu^*] + Bias(\hat{\mu})^2 = \frac{225}{n+1} + (n(\bar{y} - 112) - 12)^2$$
$$MSE[\hat{\mu}|\mu^*] = \frac{225}{n+1} + (\bar{y}n - 112n - 12)^2$$

```
bayes_mse <- function(n = 10, ybar = 113){(225/(n+1)) + (((n*(ybar-112))-12)/(n+1))^2}
(bayes_mse(10, 113))
```

```
## [1] 20.4876
```

```r
(bayes_variance(10) + (bayes_bias_mu(10, 113))^2)
```

```
## [1] 20.4876
```

```r
(bayes_mse(1000, 113))
```

```
## [1] 1.19897
```

```r
(bayes_variance(1000) + (bayes_bias_mu(1000, 113))^2)
```

```
## [1] 1.19897
```

```r
(bayes_mse(1, 113))
```

```
## [1] 142.75
```

```r
(bayes_variance(1) + (bayes_bias_mu(1, 113))^2)
```

```
## [1] 142.75
```

```r
(bayes_mse(10, 133))
```

```
## [1] 344.4545
```

```r
(bayes_variance(10) + (bayes_bias_mu(10, 133))^2)
```

```
## [1] 344.4545
```

```r
(bayes_mse(1000, 133))
```

```
## [1] 439.8412
```

```r
(bayes_variance(1000) + (bayes_bias_mu(1000, 133))^2)
```

```
## [1] 439.8412
```

```r
(bayes_mse(1, 133))
```

```
## [1] 132.75
```

```r
(bayes_variance(1) + (bayes_bias_mu(1, 133))^2)
```

```
## [1] 132.75
```

```r
(bayes_mse(10, 112))
```

```
## [1] 21.64463
```

```r
(bayes_variance(10) + (bayes_bias_mu(10, 112))^2)
```

```
## [1] 21.64463
```

```r
(bayes_mse(1000, 112))
```

```
## [1] 0.2249189
```

```r
(bayes_variance(1000) + (bayes_bias_mu(1000, 112))^2)
```

```
## [1] 0.2249189
```

```r
(bayes_mse(1, 112))
```

```
## [1] 148.5
```

```r
(bayes_variance(1) + (bayes_bias_mu(1, 112))^2)
```

```
## [1] 148.5
```

```r
(bayes_mse(10, 224))
```

```
## [1] 10166.44
```

```r
(bayes_mse(1000, 224))
```

```
## [1] 12516.49
```

```r
(bayes_mse(1, 224))
```

```
## [1] 2612.5
```

```r
(bayes_mse(12, 113))
```

```
## [1] 17.30769
```

```r
(bayes_mse(6, 114))
```

```
## [1] 32.14286
```

```
n.mse <- 1
bayes.mse.1 <- cbind(bayes_mse(n.mse, 110), bayes_mse(n.mse, 112), bayes_mse(n.mse, 113),
                      bayes_mse(n.mse, 114), bayes_mse(n.mse, 133), bayes_mse(n.mse, 224))
n.mse <- 10
bayes.mse.10 <- cbind(bayes_mse(n.mse, 110), bayes_mse(n.mse, 112), bayes_mse(n.mse, 113),
                      bayes_mse(n.mse, 114), bayes_mse(n.mse, 133), bayes_mse(n.mse, 224))
n.mse <- 1000
bayes.mse.1000 <- cbind(bayes_mse(n.mse, 110), bayes_mse(n.mse, 112), bayes_mse(n.mse, 113),
                      bayes_mse(n.mse, 114), bayes_mse(n.mse, 133), bayes_mse(n.mse, 224))
```

It is noteworthy that when $\bar{y}$ is close to μ^*, increasing n decreases the MSE, but when $\bar{y}$ is far from μ^* then increasing n increases the MSE.

Now, for the ML estimators **Maximum Likelihood Estimator**
Bias

$$E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] = \bar{y}$$

As n increases, the expected value approximates $\bar{y}$ although it never quite reaches it. When n is small, the expected value is closer to the prior. Since μ^* is a fixed value and not a random variable, the bias becomes:

$$Bias(\hat{\mu})^{=} E_{\boldsymbol{y}}[\hat{\mu}|\mu^*] - \mu^* = \bar{y} - 112$$ MLE is always unbiased

```
mle_bias_mu <- function(ybar){(ybar-112)}
(mle_bias_mu(113))
```

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

```
(mle_bias_mu(133))
```

```
## [1] 21
```

```
(mle_bias_mu(112))
```

```
## [1] 0
```
✓

```
(mle_bias_mu(224))
```

```
## [1] 112
```

```
mle.bias <- cbind(mle_bias_mu(110), mle_bias_mu(112), mle_bias_mu(113),
                      mle_bias_mu(114), mle_bias_mu(133), mle_bias_mu(224))
```

This means that bias in MLE is independent of n and only depends on $\bar{y}$.

Now, to compare the bias for the Bayes and MLE estimators, I will take the absolute value of the bias, comparing different sample sizes $n = 1, 10, 1,000$ for the Bayes estimator and subtract the MLE estimator bias from it, to see which is larger.

```r
(abs(bayes.bias.1) - abs(mle.bias))
```

```
##      [,1] [,2] [,3] [,4]  [,5] [,6]
## [1,]    5    6  4.5    3 -16.5  -62
```

```r
(abs(bayes.bias.10) - abs(mle.bias))
```

```
##           [,1]     [,2]       [,3]      [,4] [,5]      [,6]
## [1,] 0.9090909 1.090909 -0.8181818 -1.272727   -3 -11.27273
```

```r
(abs(bayes.bias.1000) - abs(mle.bias))
```

```
##            [,1]       [,2]        [,3]        [,4]        [,5]       [,6]
## [1,] 0.00999001 0.01198801 -0.01298701 -0.01398601 -0.03296703 -0.1238761
```

When $\bar{y}$ is close to μ^*, the bias for the Bayes Estimator is larger, but as it gets further from μ^*, then the ML Estimator has a bigger bias. Also, as n increases, the bias decreases importantly.

Variance
The variance, is given by:

$$Var[\hat{\mu}|\mu^*] = \sigma^2/n = 225/n$$

```r
mle_variance <- function(n){225/(n)}
(mle_variance(10))
```

```
## [1] 22.5
```

✓

```r
(mle_variance(1000))
```

```
## [1] 0.225
```

```r
(mle_variance(1))
```

```
## [1] 225
```

```r
(mle.variance <- cbind(mle_variance(1), mle_variance(10), mle_variance(1000)))
```

```
##      [,1] [,2]  [,3]
## [1,]  225 22.5 0.225
```

This means that as n increases, the variance decreases, regardless of $\bar{y}$ or the mean (expected or real).

Now, to compare the variance for the Bayes and MLE estimators, I will subtract the latter from the former, comparing different sample sizes $n = 1, 10, 1,000$, to see which is larger.

```r
(bayes.variance - mle.variance)
```

```
##          [,1]      [,2]          [,3]
## [1,] -112.5 -2.045455 -0.0002247752
```

In all instances the Bayesian estimator has smaller variance, although as n increases, this difference becomes inconsequential.

MSE
Finally, for the MSE we use:

$$MSE[\hat{\mu}|\mu^*] = Var[\hat{\mu}|\mu^*] + Bias(\hat{\mu})^2 = \frac{225}{n} + \underline{\bar{y}^2} \quad (\bar{y}-\mu^*)^2 = 0$$

```r
mle_mse <- function(n = 10, ybar = 113){(225/(n)) + (ybar-112)^2}
(mle_mse(10, 113))
```

```
## [1] 23.5
```

```r
(mle_variance(10) + (mle_bias_mu(113))^2)
```

```
## [1] 23.5
```

```r
(mle_mse(1000, 113))
```

```
## [1] 1.225
```

```r
(mle_variance(1000) + (mle_bias_mu(113))^2)
```

```
## [1] 1.225
```

```r
(mle_mse(1, 113))
```

```
## [1] 226
```

```r
(mle_variance(1) + (mle_bias_mu(113))^2)
```

```
## [1] 226
```

```r
(mle_mse(10, 133))
```

```
## [1] 463.5
```

```r
(mle_variance(10) + (mle_bias_mu(133))^2)
```

```
## [1] 463.5
```

```r
(mle_mse(1000, 133))
```

```
## [1] 441.225
```

```r
(mle_variance(1000) + (mle_bias_mu(133))^2)
```

```
## [1] 441.225
```

```r
(mle_mse(1, 133))
```

```
## [1] 666
```

```r
(mle_variance(1) + (mle_bias_mu(133))^2)
```

```
## [1] 666
```

```r
(mle_mse(10, 112))
```

```
## [1] 22.5
```

```r
(mle_variance(10) + (mle_bias_mu(112))^2)
```

```
## [1] 22.5
```

```r
(mle_mse(1000, 112))
```

```
## [1] 0.225
```

```r
(mle_variance(1000) + (mle_bias_mu(112))^2)
```

```
## [1] 0.225
```

```r
(mle_mse(1, 112))
```

```
## [1] 225
```

```r
(mle_variance(1) + (mle_bias_mu(112))^2)
```

```
## [1] 225
```

```r
(mle_mse(10, 224))
```

```
## [1] 12566.5
```

```r
(mle_mse(1000, 224))
```

```
## [1] 12544.23
```

```r
(mle_mse(1, 224))
```

```
## [1] 12769
```

```r
(mle_mse(12, 113))
```

```
## [1] 19.75
```

```r
(mle_mse(6, 114))
```

```
## [1] 41.5
```

```r
n.mse <- 1
mle.mse.1 <- cbind(mle_mse(n.mse, 110), mle_mse(n.mse, 112), mle_mse(n.mse, 113),
                      mle_mse(n.mse, 114), mle_mse(n.mse, 133), mle_mse(n.mse, 224))
n.mse <- 10
mle.mse.10 <- cbind(mle_mse(n.mse, 110), mle_mse(n.mse, 112), mle_mse(n.mse, 113),
                      mle_mse(n.mse, 114), mle_mse(n.mse, 133), mle_mse(n.mse, 224))
n.mse <- 1000
mle.mse.1000 <- cbind(mle_mse(n.mse, 110), mle_mse(n.mse, 112), mle_mse(n.mse, 113),
                      mle_mse(n.mse, 114), mle_mse(n.mse, 133), mle_mse(n.mse, 224))
```

In the case of the ML, the MSE always decreases as n increases, and always increases the further $\bar{y}$ is from μ^*.

Now, to compare the MSE for the Bayes and MLE estimators, I will compare different sample sizes $n = 1, 10, 1,000$ for both estimators and subtract the MLE estimator from the Bayes estimator, to see which is larger.

```r
(bayes.mse.1 - mle.mse.1)
```

```
##       [,1]  [,2]   [,3]  [,4]    [,5]     [,6]
## [1,] -67.5 -76.5 -83.25 -91.5 -533.25 -10156.5
```

```r
(bayes.mse.10 - mle.mse.10)
```

```
##          [,1]       [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 2.417355 -0.8553719 -3.012397 -5.516529 -119.0455 -2400.062
```

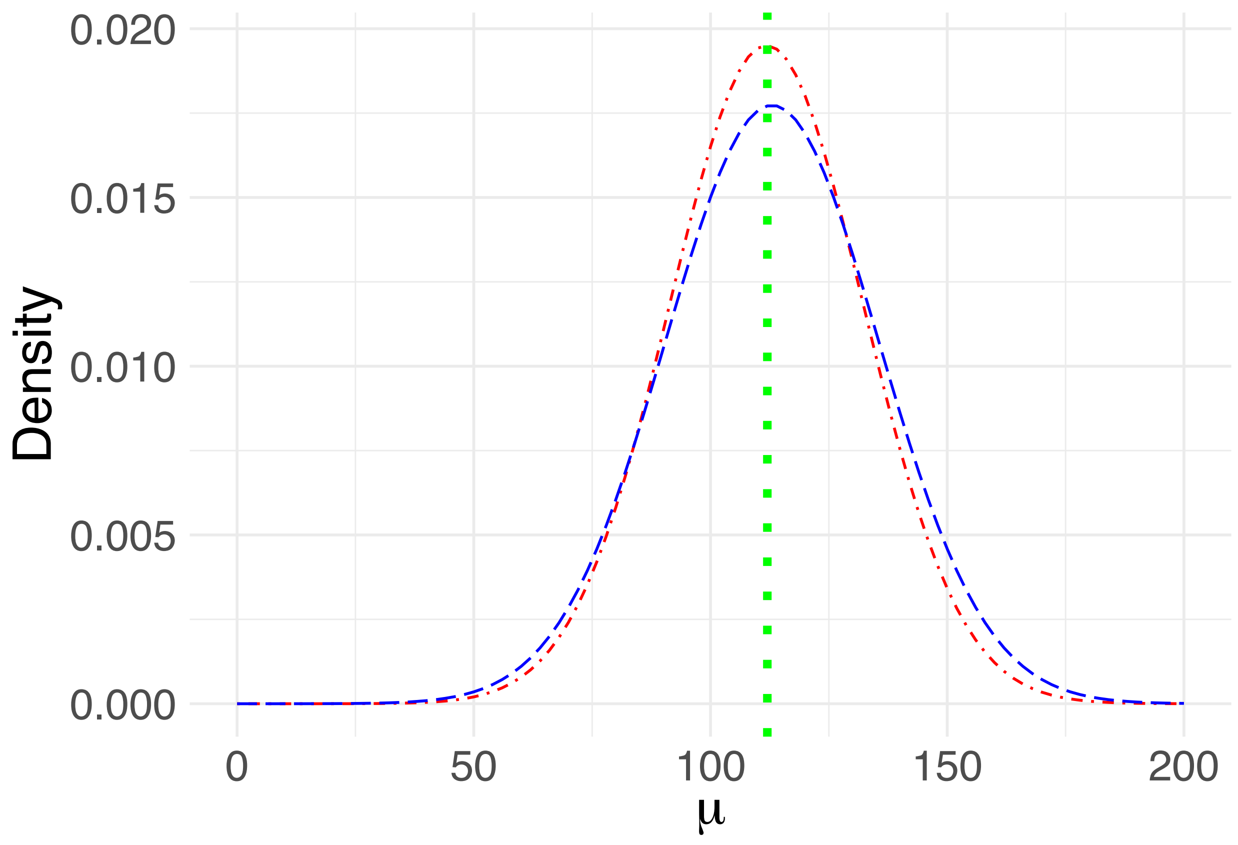
```r
(bayes.mse.1000 - mle.mse.1000)
```

```
##            [,1]          [,2]        [,3]        [,4]      [,5]      [,6]
## [1,] 0.03983507 -8.106279e-05 -0.02603014 -0.05597322 -1.383753 -27.73313
```

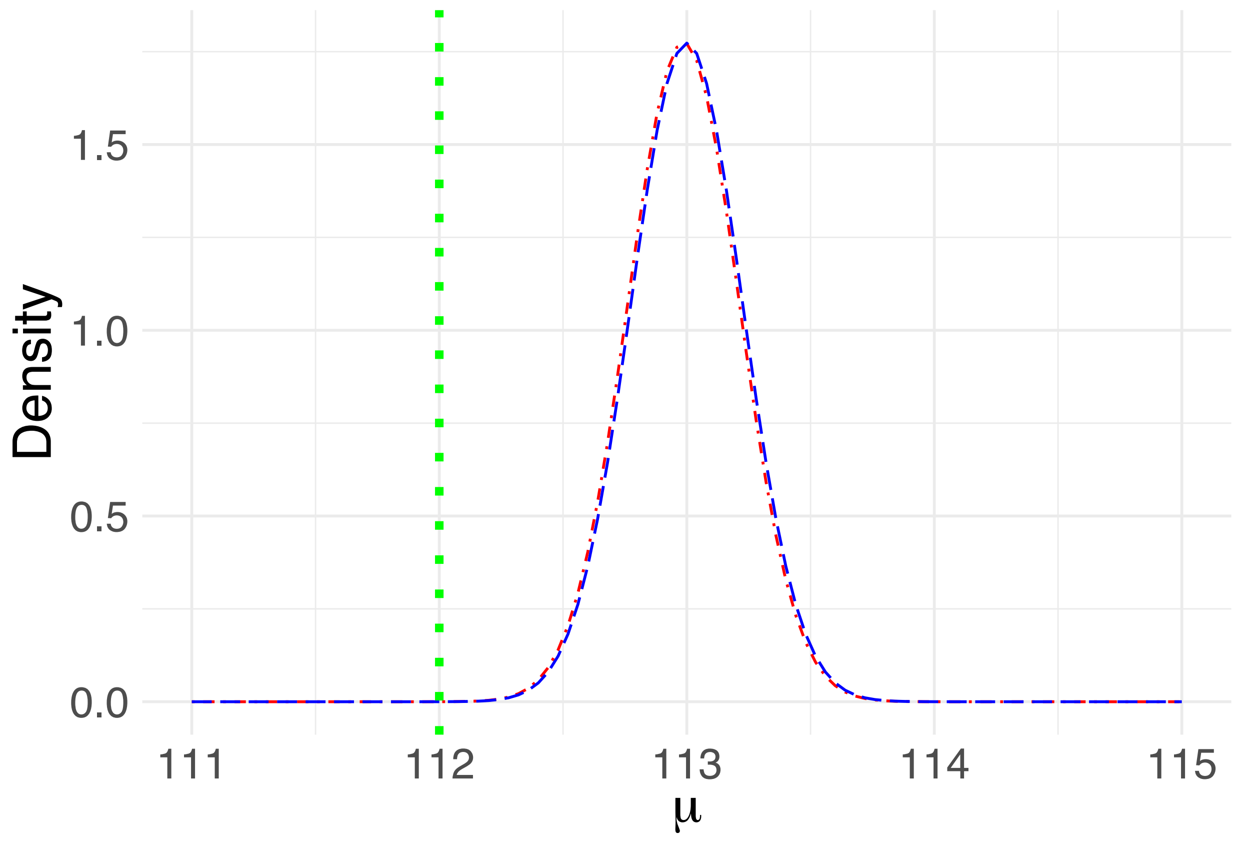
In most instances the Bayesian estimator has a smaller MSE than the ML estimator, especially as $\bar{y}$ gets further from μ^*. Also, as n increases, the bias decreases importantly.

Optional: Consider
(i) plotting the sampling distributions for both the Bayes estimator as well as the MLE.

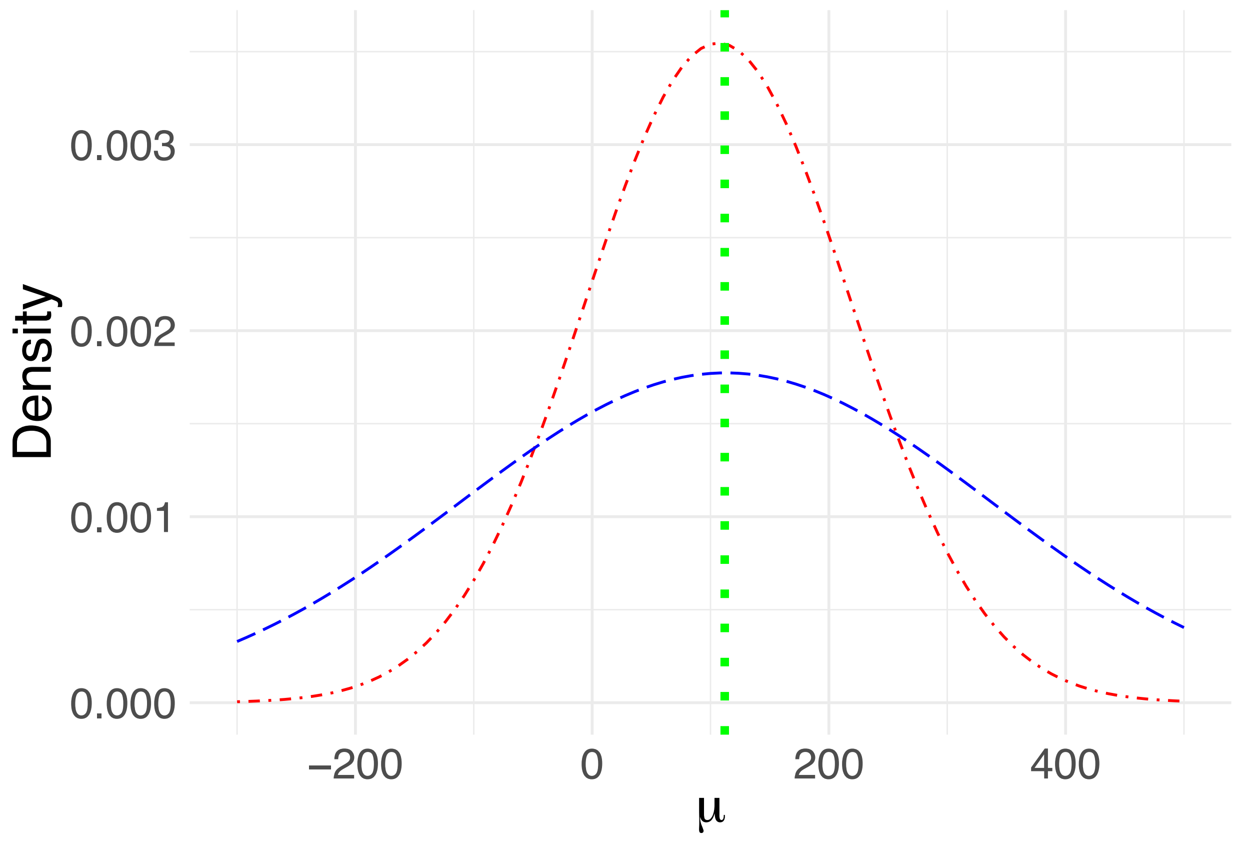
```r
plot_densities <- function(n = 10, ybar = 113, xlow = 0, xhigh = 200){
  bayes.mu <- bayes_exp_mu(n, ybar)
  bayes.var <- bayes_variance(n)
  sd.mle <- mle_variance(n)
  bayes_dens <- function(x) dnorm(x, mean = bayes.mu , sd = bayes.var)
  mle_dens <- function(x) dnorm(x, mean = ybar, sd = sd.mle)

  samp.dist <- ggplot(NULL, aes(c(xlow, xhigh))) +
    geom_line(stat = "function", fun = bayes_dens, color = "red", linetype = "dotdash") +
    geom_line(stat = "function", fun = mle_dens, linetype = "longdash", color = "blue") +
    theme_minimal() +
    ylab("Density") +
    xlab(expression(mu)) +
    theme(
      legend.position = "top",
      legend.title = element_blank(),
      text = element_text(size = 20)
    )
  samp.dist + geom_vline(xintercept = 112, linetype="dotted", color = "green", size=1.5)
}
plot_densities()
```



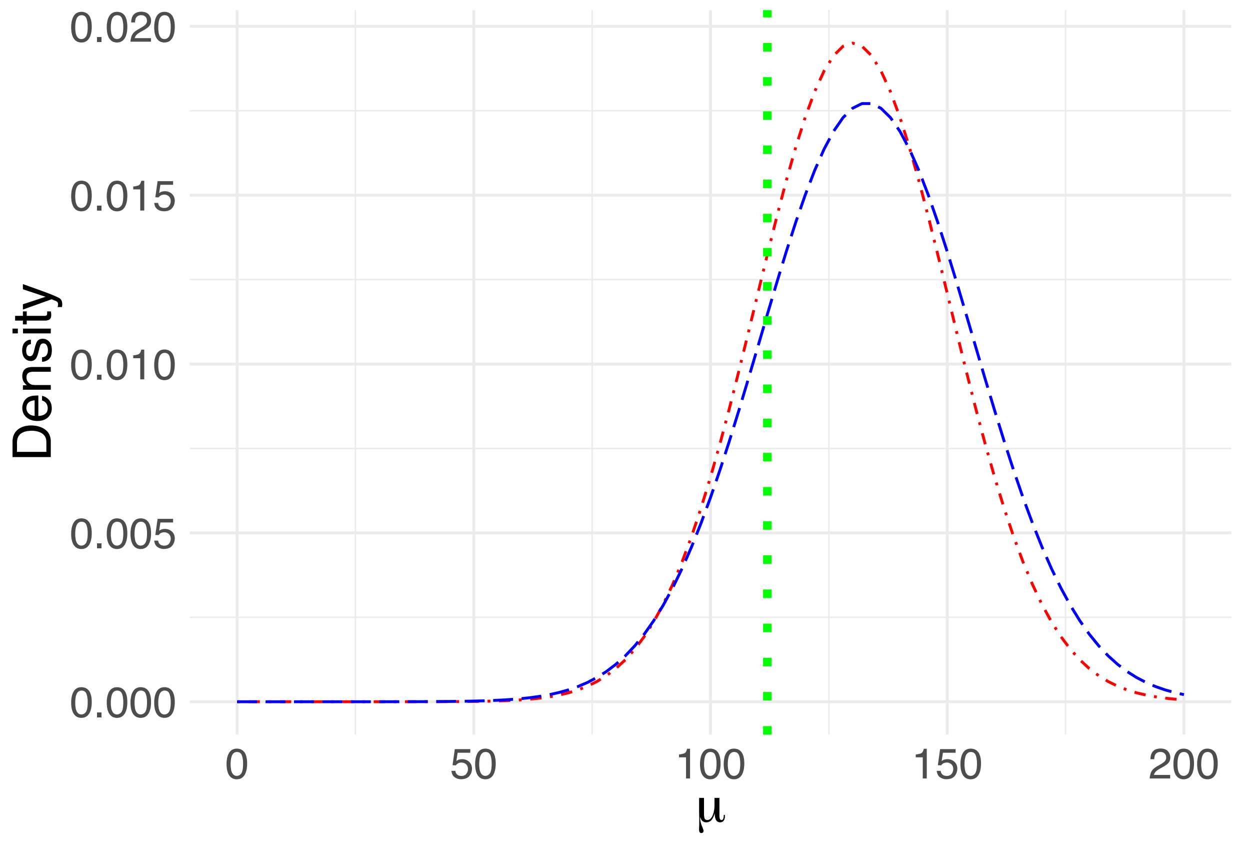
```r
plot_densities(1000, 113, 111, 115)
```



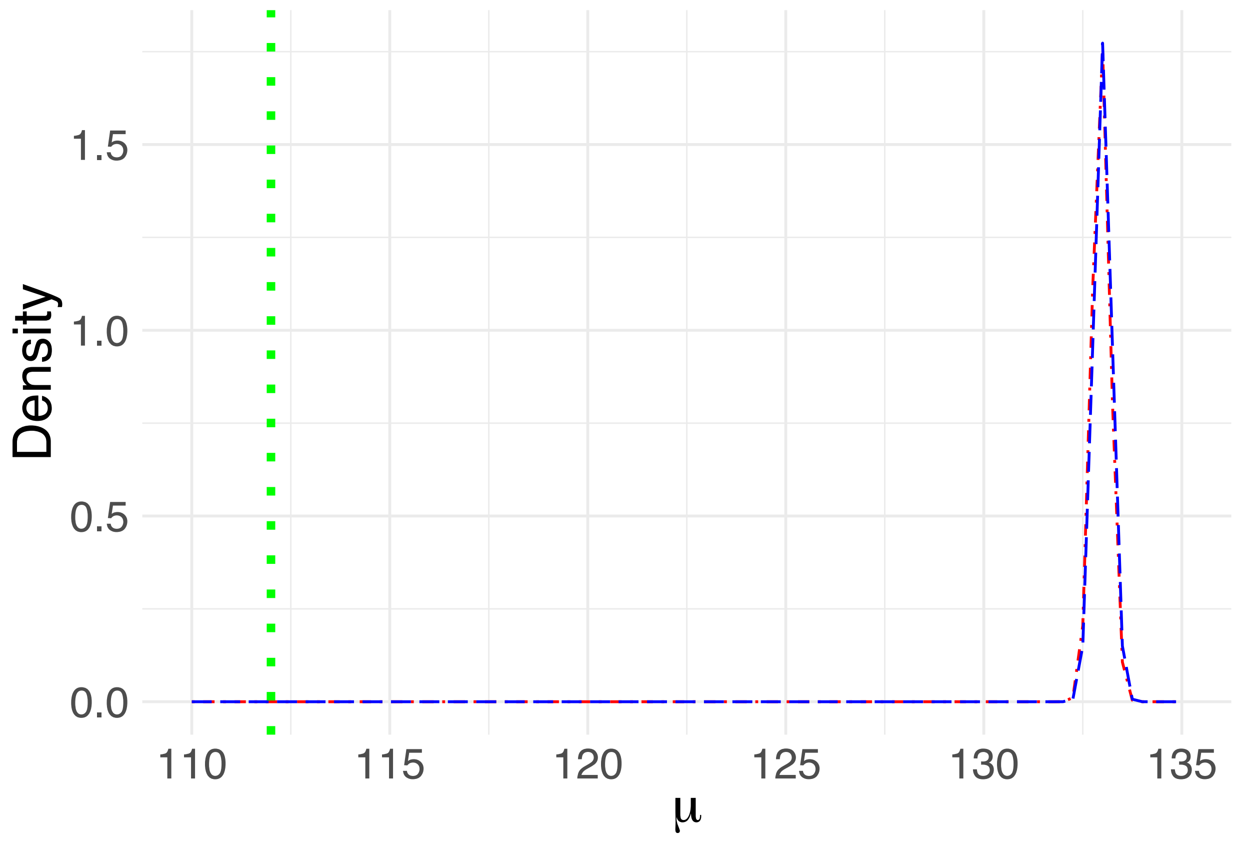
```
plot_densities(1, 113, -300,500)
```



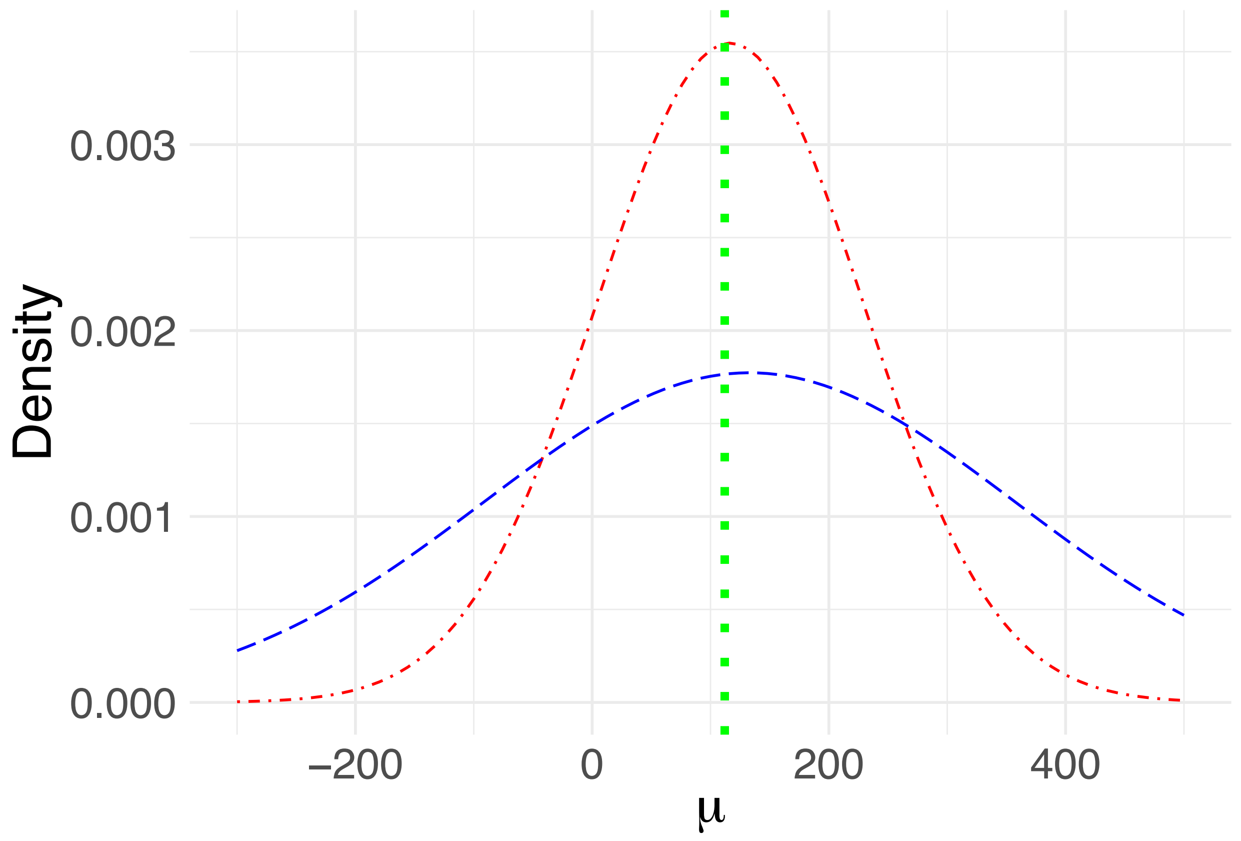
```r
plot_densities(10, 133)
```



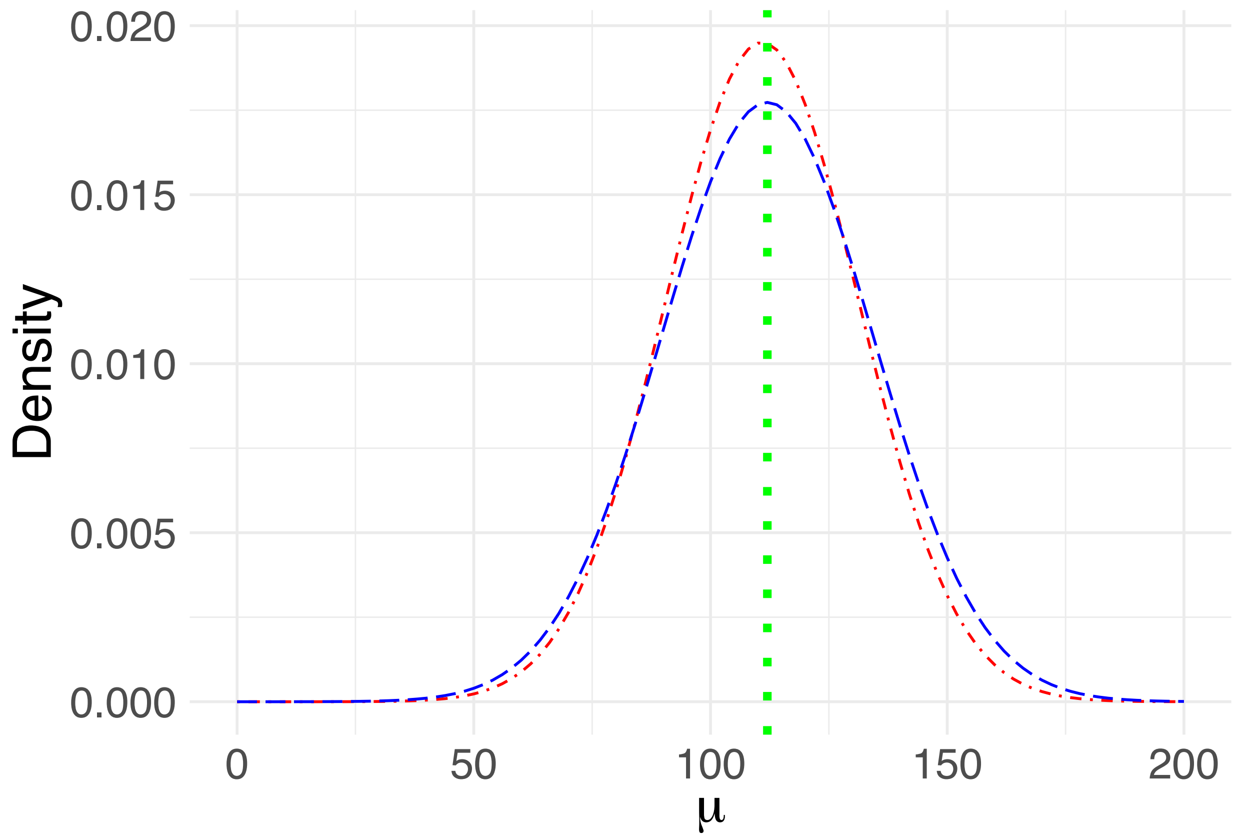
```
plot_densities(1000, 133, 110, 135)
```



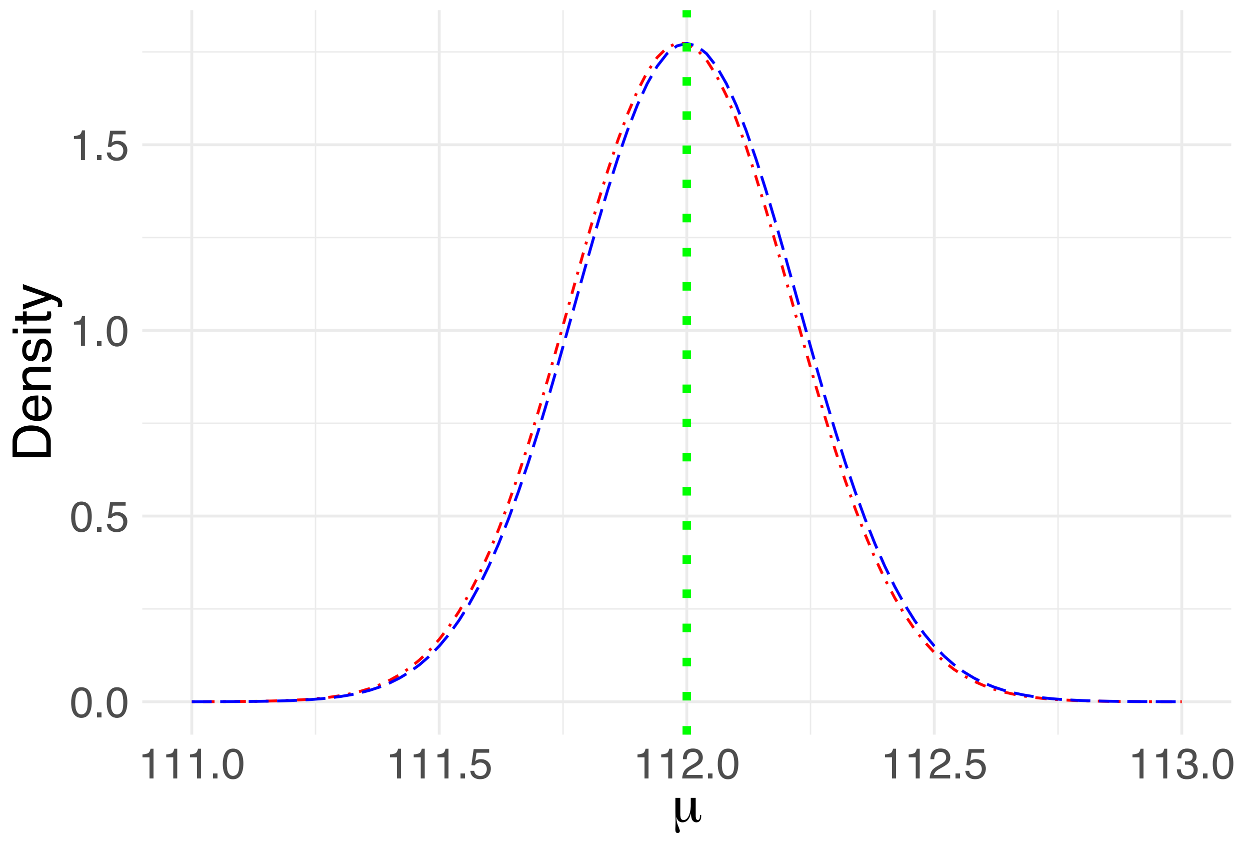
```
plot_densities(1, 133, -300,500)
```



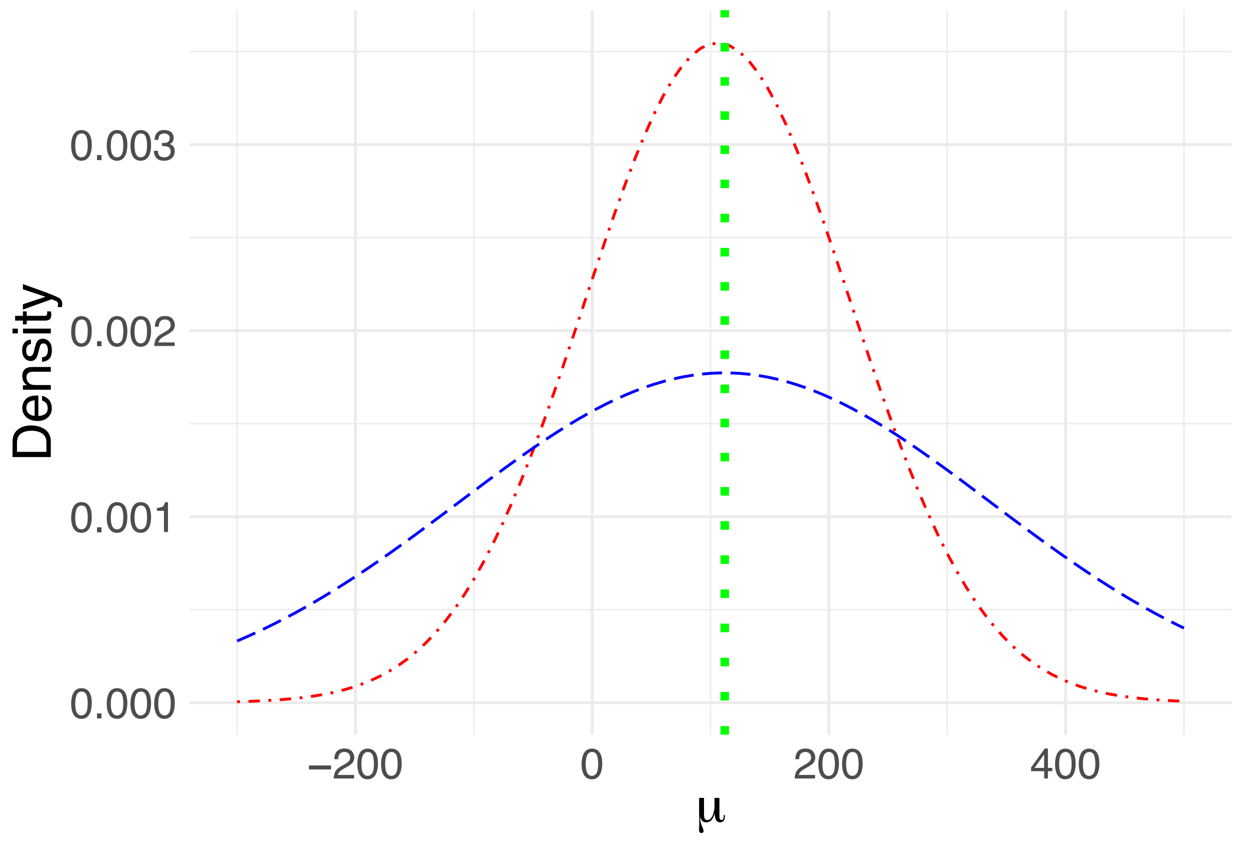
```
plot_densities(10, 112)
```



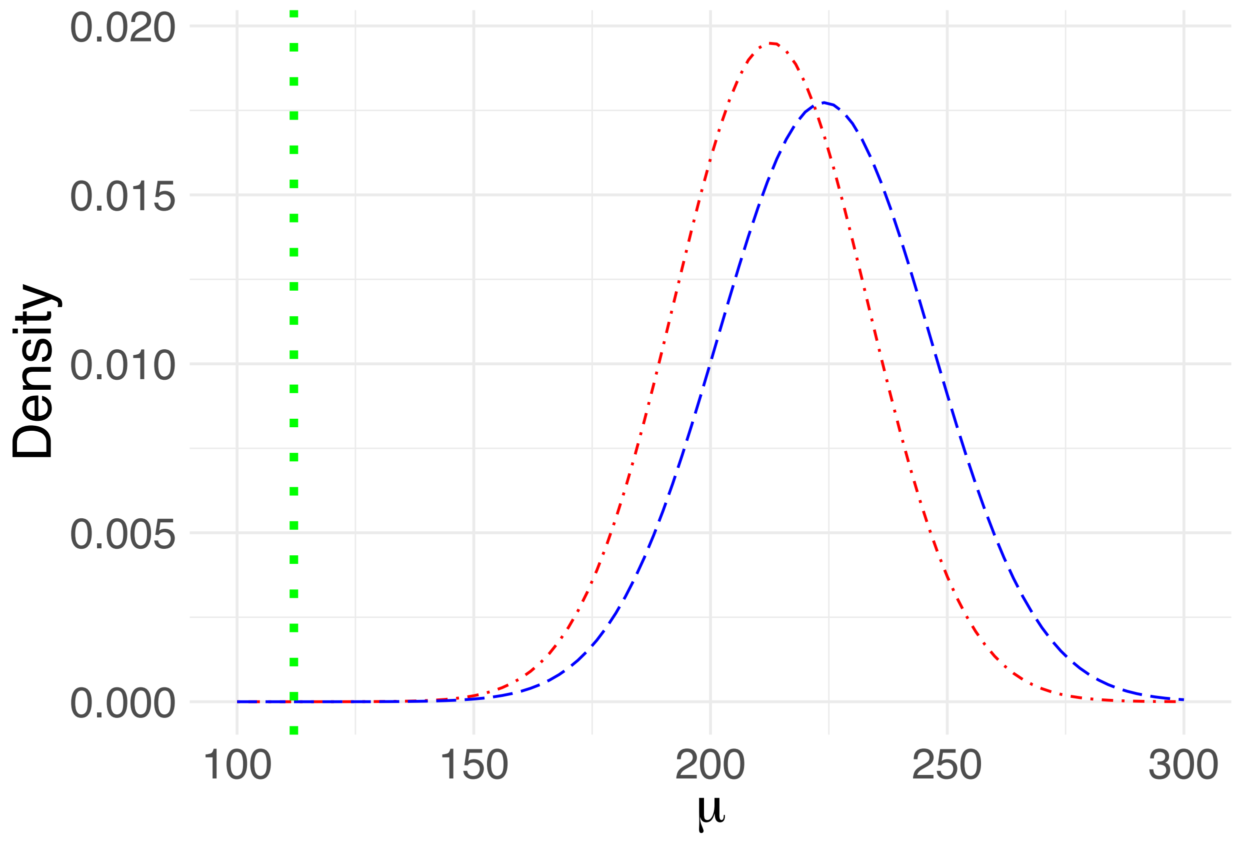
```
plot_densities(1000, 112, 111, 113)
```



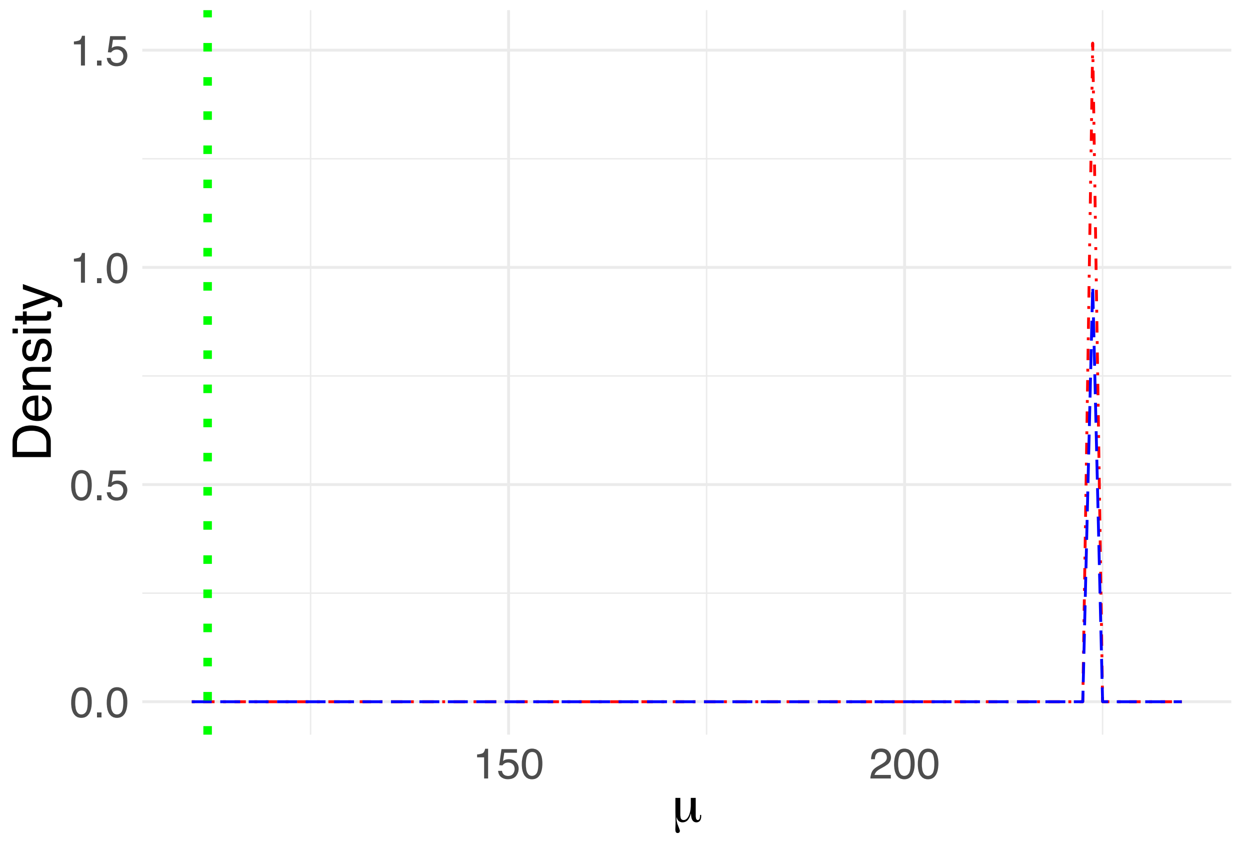
```
plot_densities(1, 112, -300,500)
```



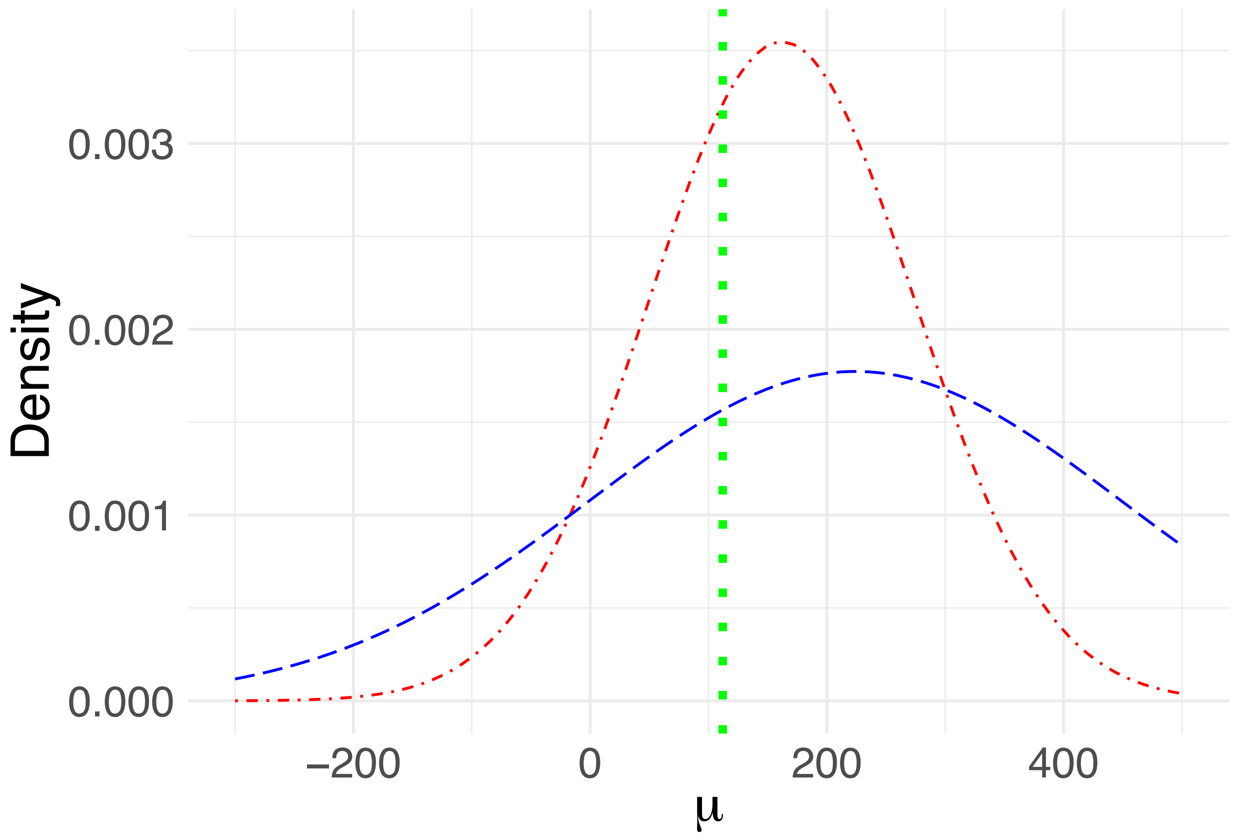
```r
plot_densities(10, 224, 100, 300)
```



```
plot_densities(1000, 224, 110, 235)
```



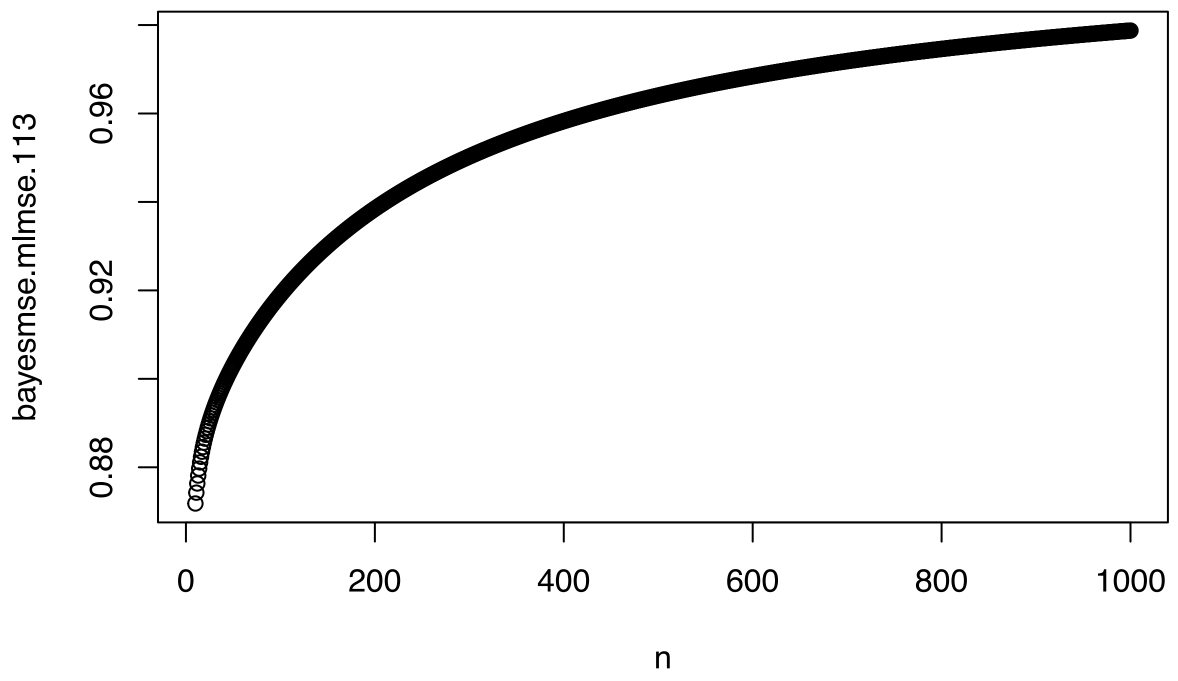
```
plot_densities(1, 224, -300,500)
```



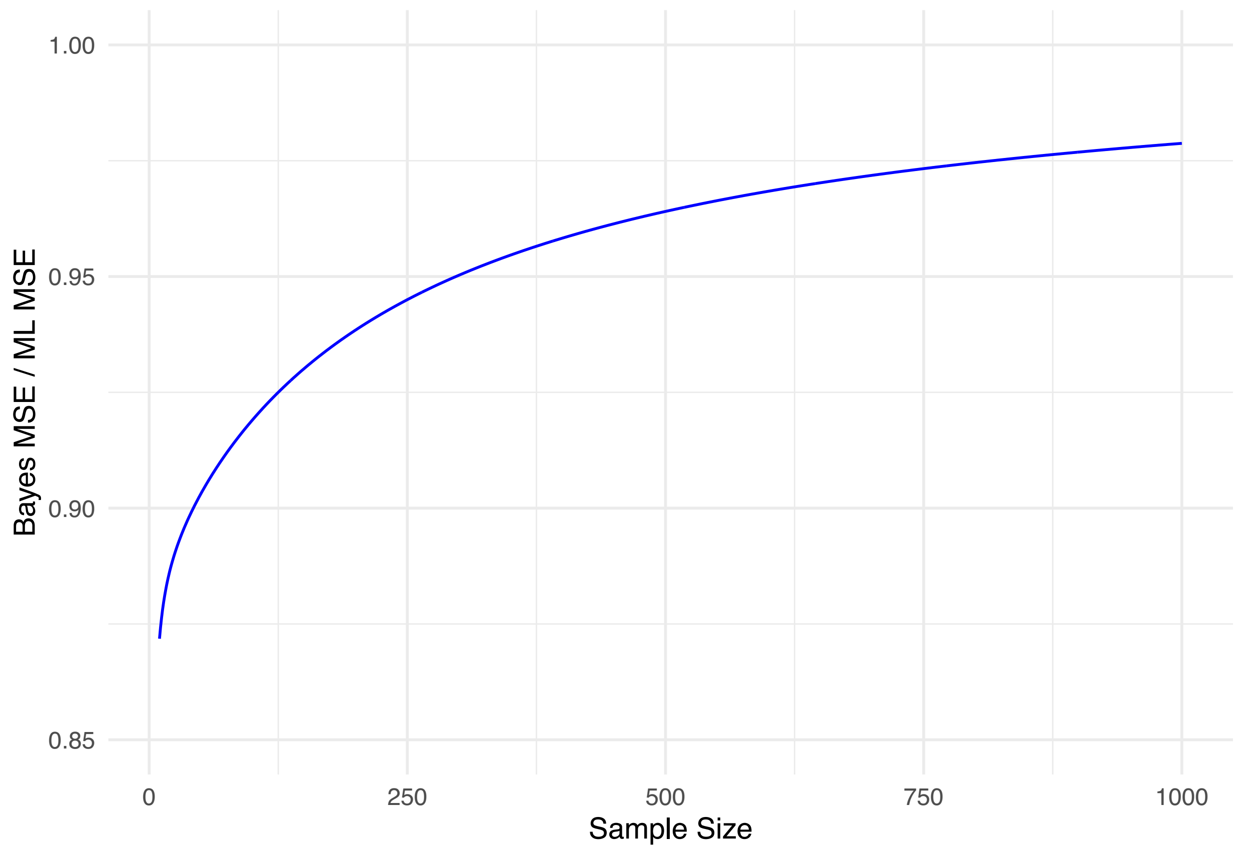
When n is very large (e.g., $n > 1,000$), the Bayesian and ML Estimators have approximately the same distribution. Nonetheless, when n is smaller, the Bayesian Estimator has a higher density near the true mean, even when the bias is larger (e.g., $\hat{\mu} = 224$). When n is especially small, the variance in the Bayesian Estimator is smaller.

(ii) obtaining the Bayes and ML MSEs for sample sizes $n = 10$ to 1,000 and plotting the ratio (Bayes MSE)/(ML MSE) against sample size, to then interpret your findings.

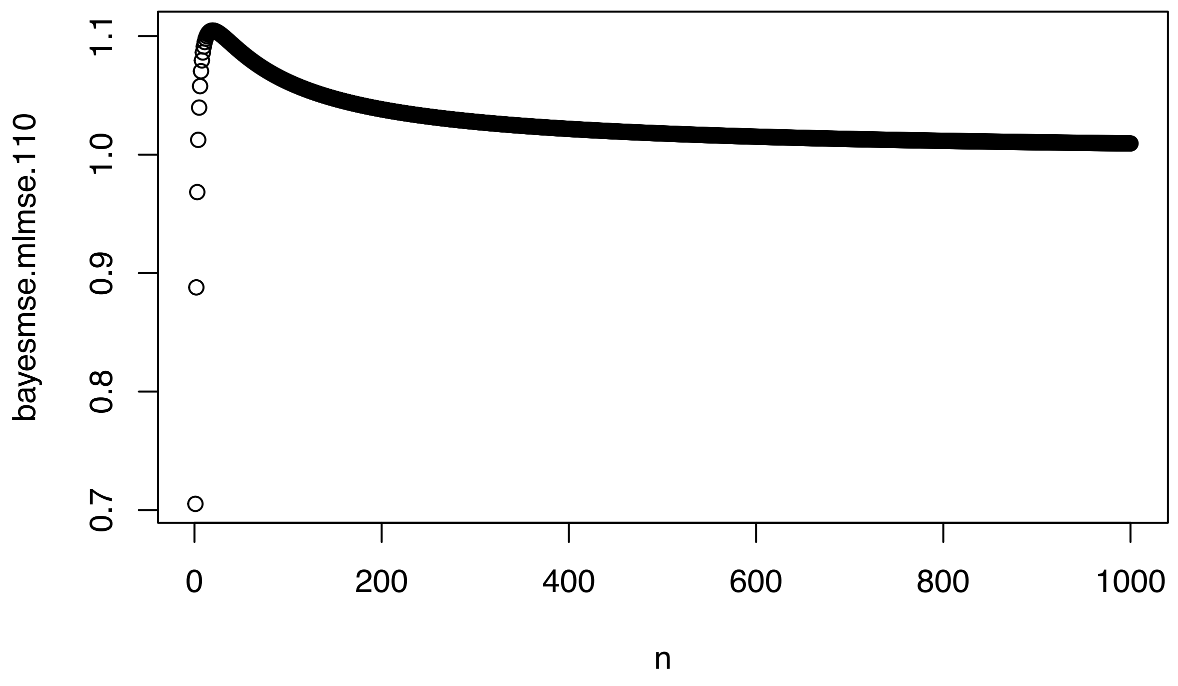
```r
n <- seq(10, 1000)
ybar <- 113
bayes.mse <- c()
for(a in 10:1000){
  temp <- bayes_mse(a, ybar)
  bayes.mse <- append(bayes.mse, temp)
}
mle.mse <- c()
for(a in 10:1000){
  temp <- mle_mse(a, ybar)
  mle.mse <- append(mle.mse, temp)
}
bayesmse.mlmse.113 <- bayes.mse / mle.mse
ratio.plot <- as_tibble(cbind(n, bayesmse.mlmse.113))
plot(n, bayesmse.mlmse.113)
```



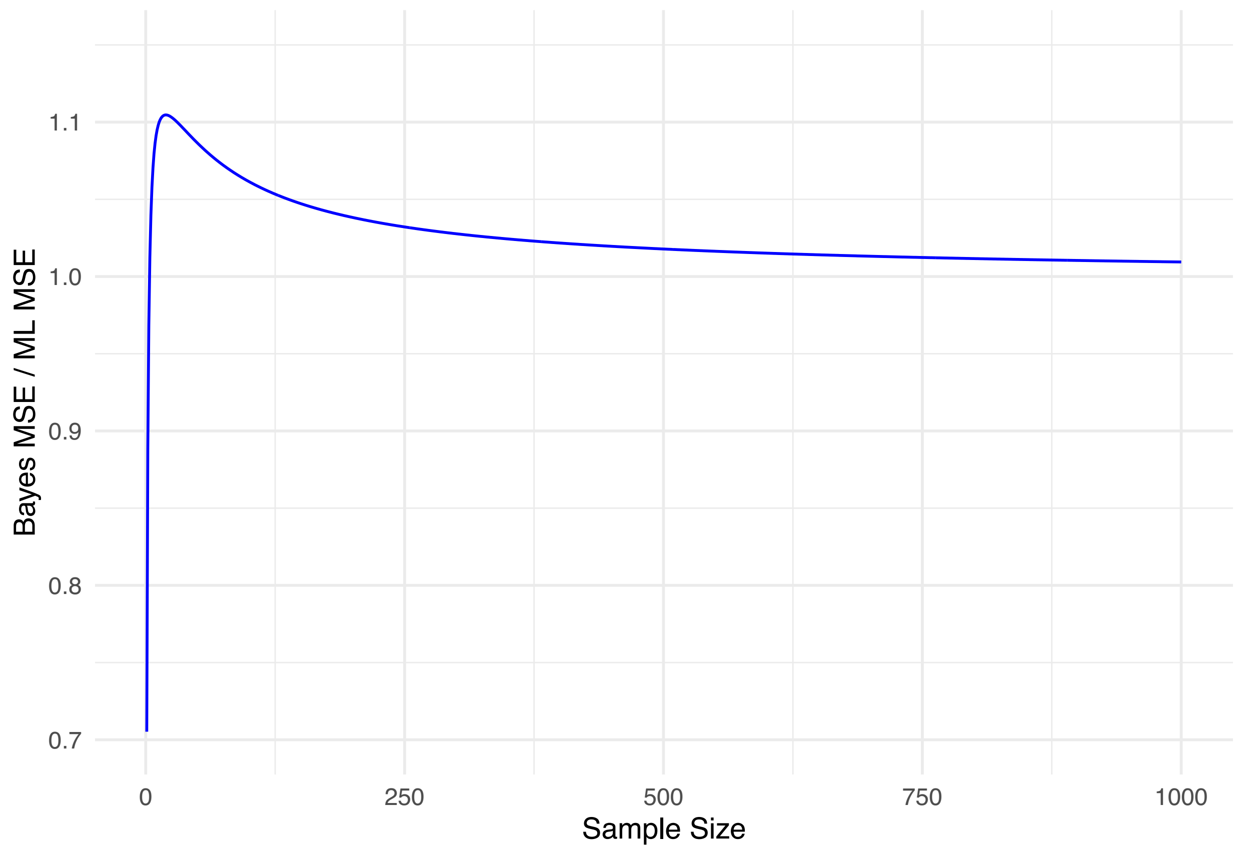
```r
ratio.plot %>%
  ggplot(aes(x=n, y=bayesmse.mlmse.113)) +
  geom_line(size = .5, color = "blue") +
  theme_minimal() +
  ylab("Bayes MSE / ML MSE") +
  ylim(0.85, 1) +
  xlab("Sample Size")
```



```
n <- seq(1, 1000)
ybar <- 110
bayes.mse <- c()
for(a in 1:1000){
  temp <- bayes_mse(a, ybar)
  bayes.mse <- append(bayes.mse, temp)
}
mle.mse <- c()
for(a in 1:1000){
  temp <- mle_mse(a, ybar)
  mle.mse <- append(mle.mse, temp)
  }
bayesmse.mlmse.110 <- bayes.mse / mle.mse
ratio.plot <- as_tibble(cbind(n, bayesmse.mlmse.110))
plot(n, bayesmse.mlmse.110)
```



```
ratio.plot %>%
  ggplot(aes(x=n, y=bayesmse.mlmse.110)) +
  geom_line(size = .5, color = "blue") +
  theme_minimal() +
  ylab("Bayes MSE / ML MSE") +
  ylim(0.7, 1.15) +
  xlab("Sample Size")
```



With the values given in the assignment, where $\bar{y} = 113$, the MSE for the Bayesian estimator is always smaller than the MSE for the ML estimator. Nonetheless, under a different circumstance, when $\bar{y} = 110$, the MSE for the ML estimator is smaller than the Bayesian estimator, except when n is very small. Nonetheless, in all instances, as n increases, the Bayesian and ML estimators end up having roughly the same MSE.

Part 2 - based on module 5

Execise 3: get stan going on your laptop [4pts] 4

The goal of this exercise is just to make sure you have stan (specifically, through rstan and brms) working on your laptop. To do so, please work through the Rmd file with module5, module5_sampling.Rmd.

For this HW Rmd, please add an example model fit using brms. This can be a model fit to radon data, copied from the module5 example, or a model fit from the brms examples. Also let us know if you successfully installed rstan and got it to work. If not, please explain the issue.

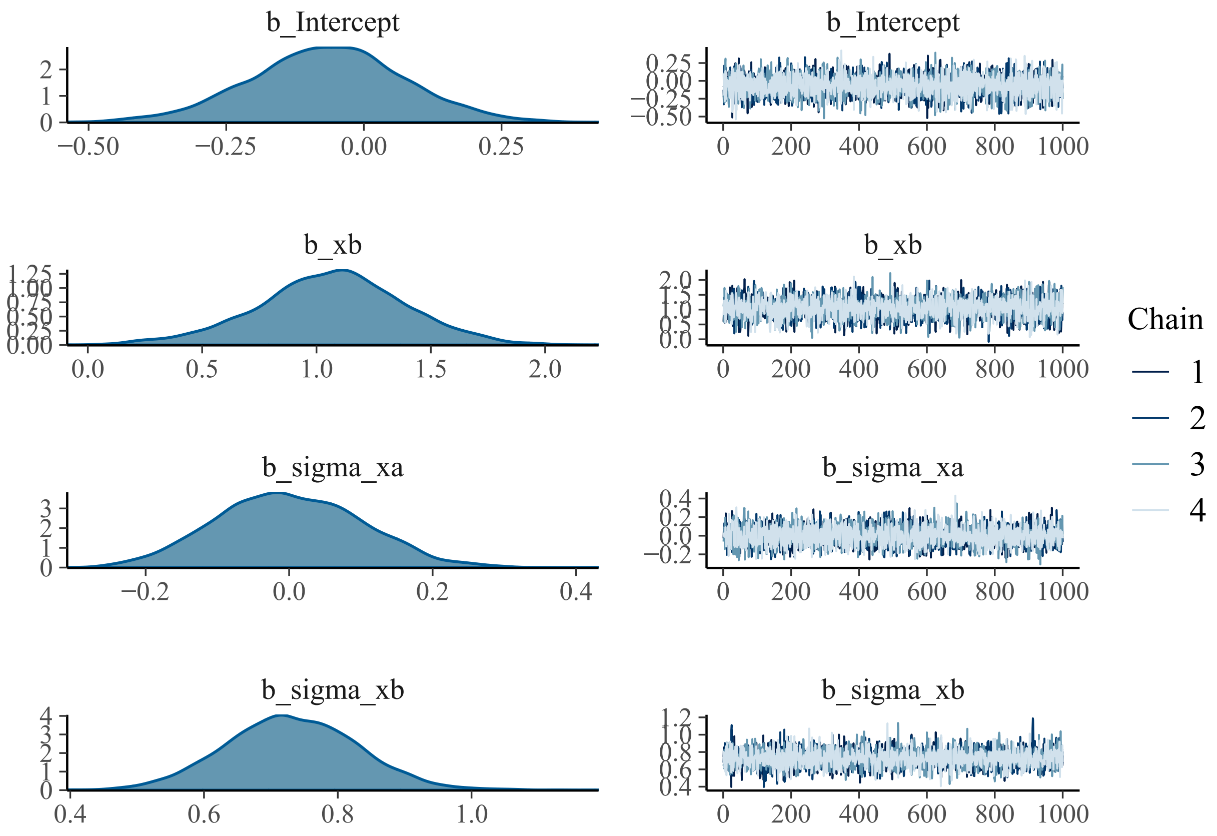
Answer:
I successfully installed `rstan` and got it to work. I worked through module 5, and to demonstrate it, here is a normal model fit from the `brms` examples with some slight modifications:

```r
set.seed(1234)
mu_priorhw2 <- set_prior("normal(0,1)", class = "Intercept")
# Normal model with heterogeneous variances
data_het <- data.frame(
  y = c(rnorm(100), rnorm(100, 1, 2)),
  x = factor(rep(c("a", "b"), each = 100))
)
```

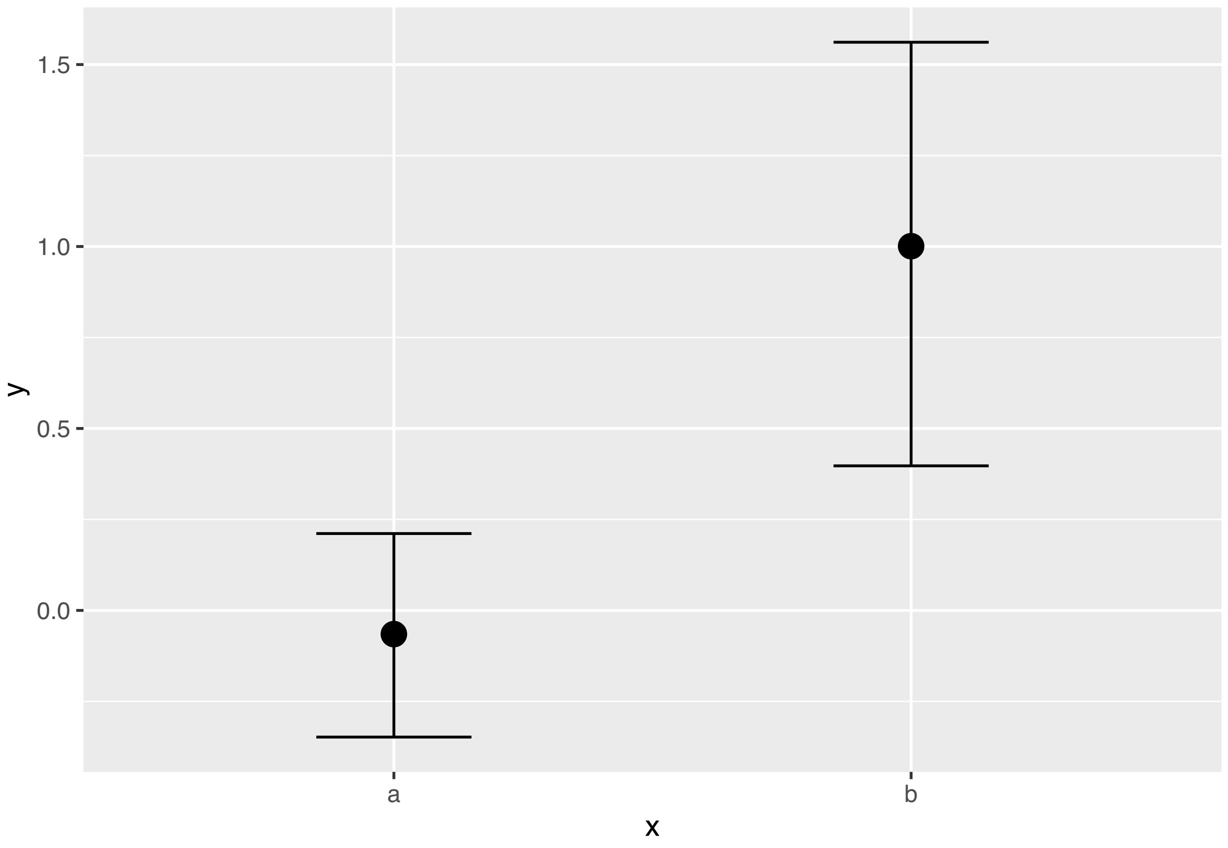
```
fit6 <- brm(bf(y ~ x, sigma ~ 0 + x), family = gaussian(), data = data_het,
            cores = getOption("mc.cores", 2),
            prior = c(mu_priorhw2),
            file = "output/exercise3"
            )
summary(fit6)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = log
## Formula: y ~ x
##          sigma ~ 0 + x
##    Data: data_het (Number of observations: 100)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     -0.07      0.14    -0.35     0.21 1.00     6040     3168
## xb             1.06      0.32     0.39     1.69 1.00     3061     2784
## sigma_xa       0.00      0.10    -0.19     0.20 1.00     3380     2542
## sigma_xb       0.73      0.10     0.54     0.93 1.00     3428     2960
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

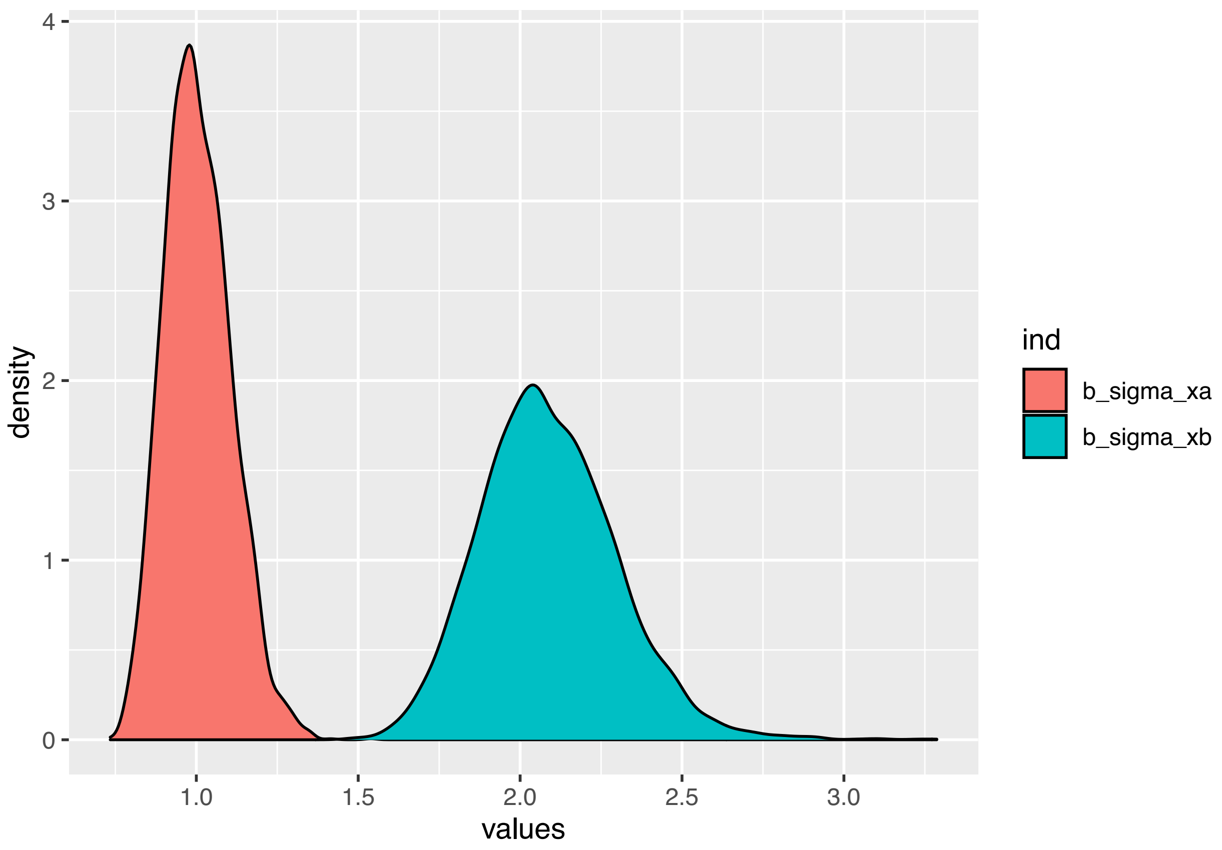
```
plot(fit6)
```



```
conditional_effects(fit6)
```



```
# extract estimated residual SDs of both groups
sigmas <- exp(as.data.frame(fit6, variable = "^b_sigma_", regex = TRUE))
ggplot(stack(sigmas), aes(values)) +
  geom_density(aes(fill = ind))
```


density
values
ind
b_sigma_xa
b_sigma_xb
0
1
2
3
4
1.0
1.5
2.0
2.5
3.0