Introduction to the course and Bayesian data analysis

Frank Edwards

1/31/2020

Sampling from the posterior

- We use Bayesian methods to *approximate* the posterior distribution
- We can then sample parameters from the estimated posterior to learn about parameters or to simulate predictions
- We can plot densities, directly sample *credible intervals*, compare differences in means, all without relying on a theoretical *sampling distribution*, like we do with frequentist methods

Integrals can be hard

Let's say we want to know how much probability mass there is within ± 1.4 standard deviations of 0 for $x \sim N(0, 1)$.

- The Normal probability density function (PDF) is: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$

Integrals can be hard

Let's say we want to know how much probability mass there is within ± 1.4 standard deviations of 0 for $x \sim N(0, 1)$.

- The Normal probability density function (PDF) is: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$
- To answer our question exactly, we can take the integral of the PDF from -1.4 to 1.4 or:

$$\int_{-1.4}^{1.4} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx$$

Integrals can be hard

Let's say we want to know how much probability mass there is within ± 1.4 standard deviations of 0 for $x \sim N(0, 1)$.

- The Normal probability density function (PDF) is: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$
- To answer our question exactly, we can take the integral of the PDF from -1.4 to 1.4 or:

$$\int_{-1.4}^{1.4} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx$$

I don't want to do this, and I suspect neither do you.

Integrals can be hard

Let's say we want to know how much probability mass there is within ± 1.4 standard deviations of 0 for $x \sim N(0, 1)$.

- The Normal probability density function (PDF) is: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$
- To answer our question exactly, we can take the integral of the PDF from -1.4 to 1.4 or:

$$\int_{-1.4}^{1.4} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx$$

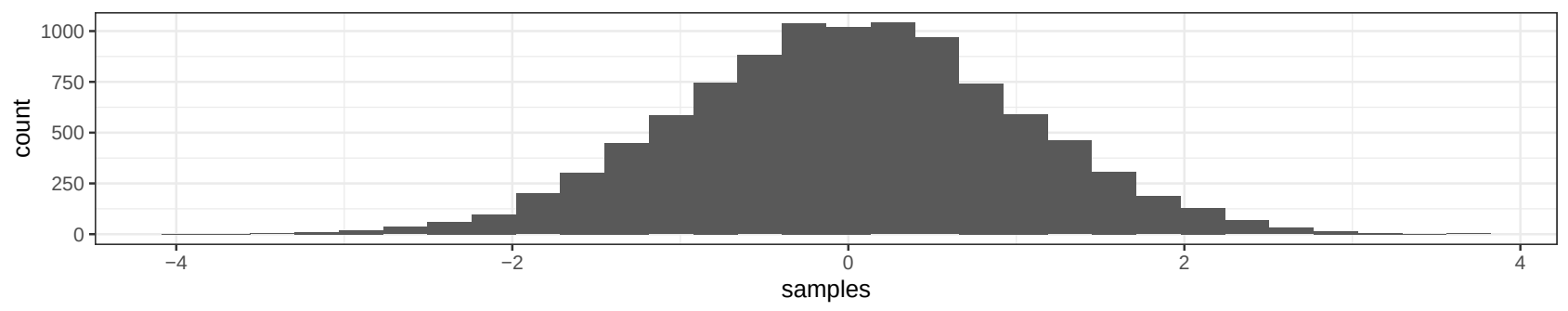
I don't want to do this, and I suspect neither do you.

Note: for defined probability distributions, we can use the, like dnorm(). However, most of our posteriors won't be so simple to work with.

But simulation is easy!

1. Sample from the target distribution
2. Summarize our quantity of interest

```
n_samples<-10000
samples<-rnorm(n = n_samples, mean = 0, sd = 1)
ggplot(data.frame(samples), aes(x=samples)) + geom_histogram()
```



Using samples to calculate quantities of interest

What are the mean and SD of this paramter?

```
mean(samples)
```

```
## [1] 0.006419005
```

```
sd(samples)
```

```
## [1] 1.003313
```

How much mass is within 0 $\pm$ 1.4?

```
samples_df<-data.frame(samples)
samples_df %>%
  filter(samples>=-1.4 & samples<=1.4) %>%
  summarise(mass = n()/n_samples)
```

```
##    mass
## 1 0.836
```

Estimation strategies

Because we are computing the product of probability distributions there sometimes aren't exact solutions. We'll rely on 3 algorithms to *approximate* posterior distributions to condition the prior on the likelihood of the data.

- Grid approximation (check!)
- Quadratic approximation (today!)
- Markov Chain Monte Carlo (MCMC) (week 7 or 8 on)

Grid approximation algorithm

1. Define the grid
2. Compute the prior for each parameter value on the grid
3. Compute the likelihood for each parameter value on the grid
4. Multiply the prior by the likelihood
5. Divide by the sum of all values

Grid approximation in R

```r
length <- 50
### make our grid
grid<-seq(from = 0, to = 1, length.out = length)
prior  <-  rep(1, length)
likelihood  <-  dbinom(6, size = 9, prob = grid)
posterior <- prior * likelihood / sum(prior * likelihood)
```

Sampling from a grid-estimated posterior

- We've estimated the posterior probability density for each value on the grid

Sampling from a grid-estimated posterior

- We've estimated the posterior probability density for each value on the grid
- We can draw random samples from this distribution. With a large enough number of samples, we can closely approximate the distribution

Sampling from a grid-estimated posterior

- We've estimated the posterior probability density for each value on the grid
- We can draw random samples from this distribution. With a large enough number of samples, we can closely approximate the distribution
- Using these samples, we can learn alot about the parameter of interest

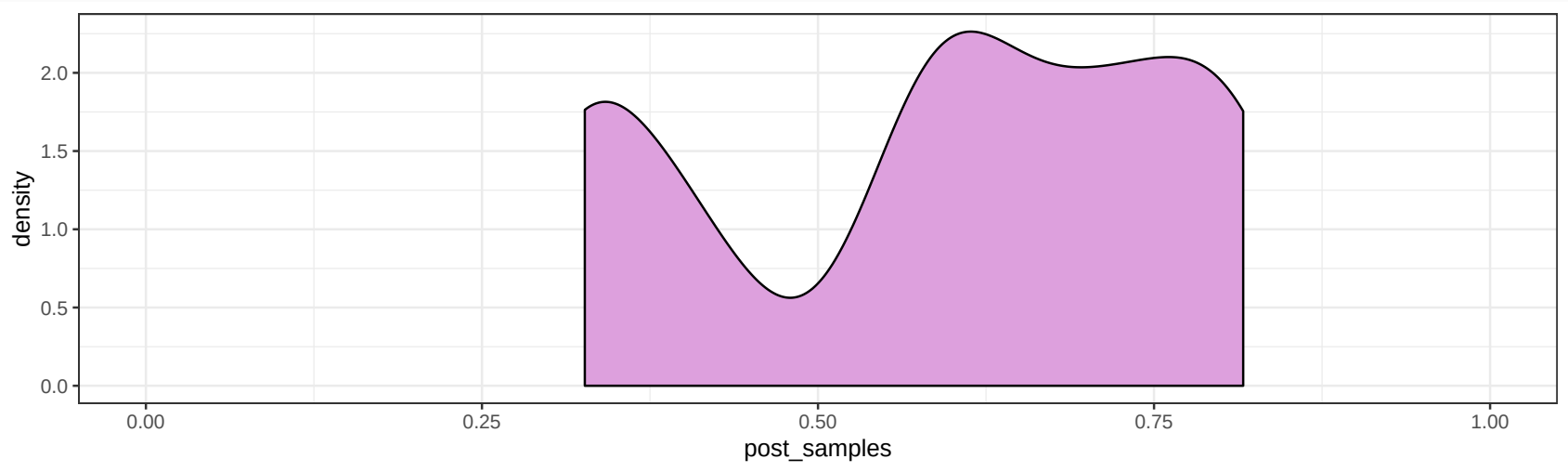
Draw samples of proportion water from grid estimated posterior

```
post_samples<-sample(
  grid,
  prob = posterior,
  size = 10,
  replace = T)

round(post_samples,2)
```

```
##  [1] 0.73 0.82 0.59 0.80 0.41 0.57 0.33 0.33 0.63 0.69
```

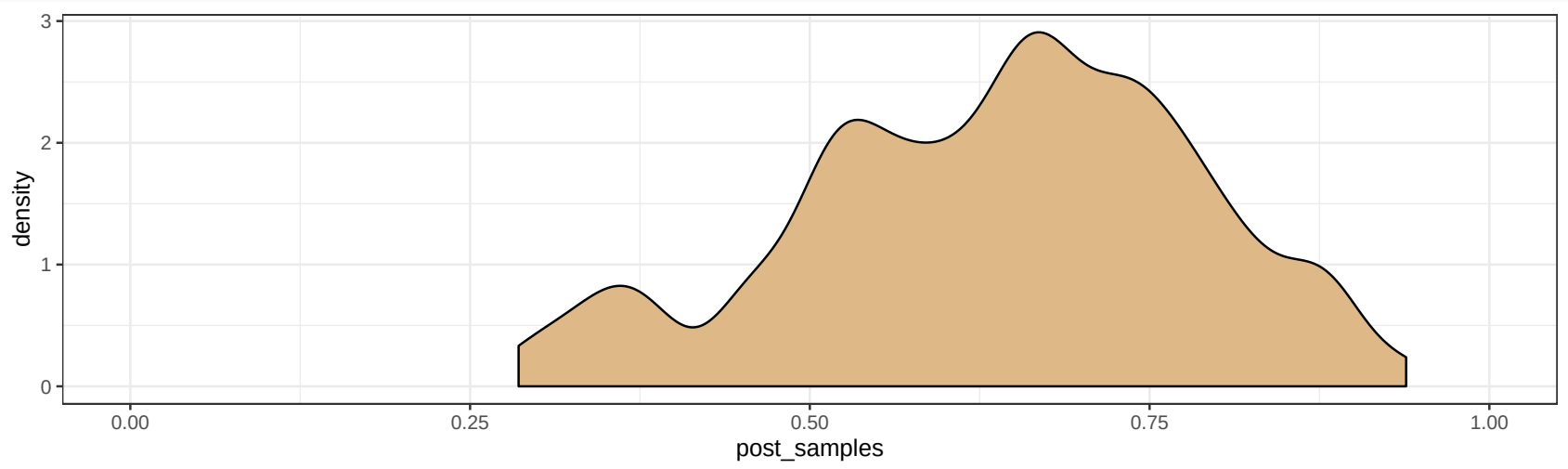
Our estimate of p, 10 samples

```
plot_dat<-tibble(post_samples)
ggplot(plot_dat,
       aes(x = post_samples)) +
  geom_density(adjust = 1/2,
               fill = "plum")+
  coord_cartesian(xlim=c(0,1))
```



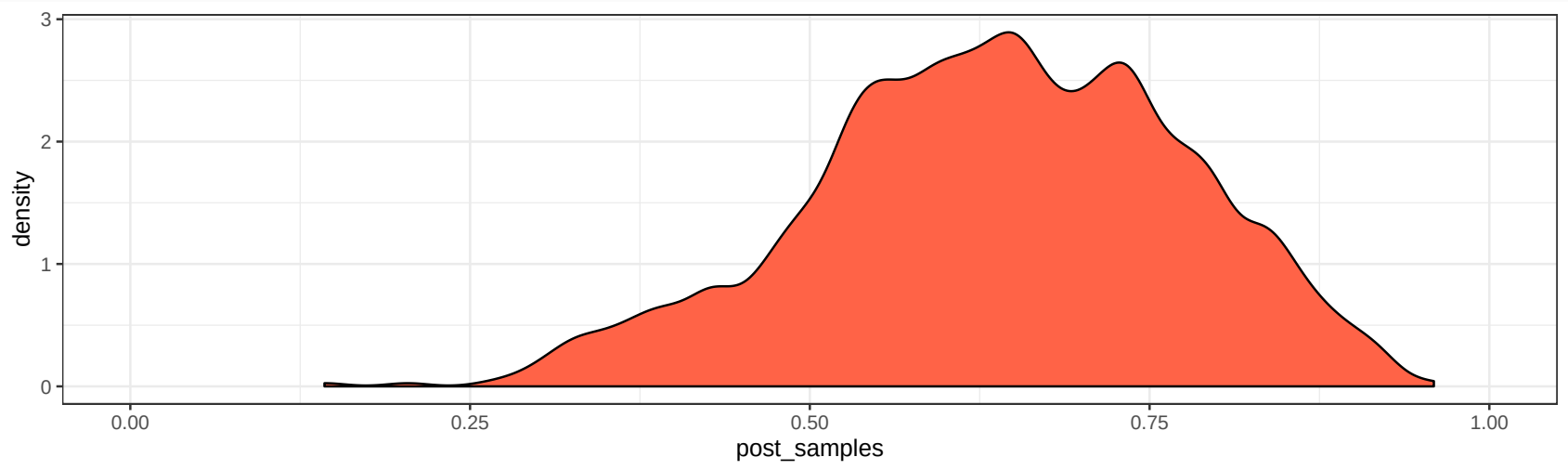
The impact of sample size on the posterior density: 100 samples

```
post_samples<-sample(
  grid,
  prob = posterior,
  size = 100,
  replace = T)

plot_dat<-tibble(post_samples)
ggplot(plot_dat,
       aes(x = post_samples)) +
  geom_density(adjust = 1/2,
               fill = "burlywood") +
  coord_cartesian(xlim=c(0,1))
```



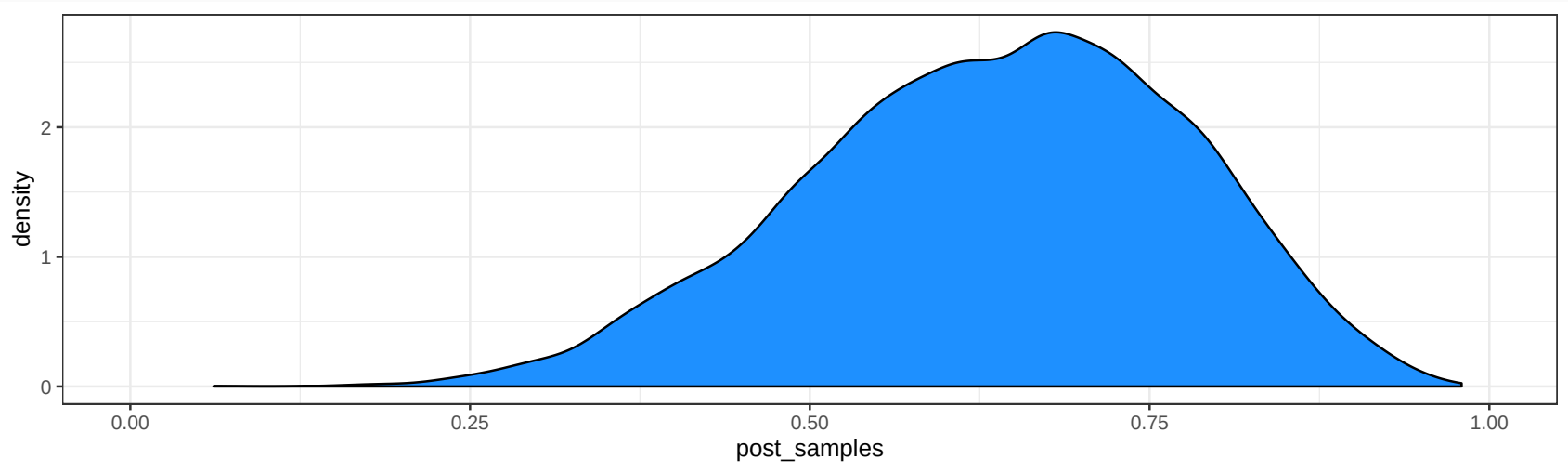
The impact of sample size on the posterior density: 1000 samples

```
post_samples<-sample(
  grid,
  prob = posterior,
  size = 1000,
  replace = T)

plot_dat<-tibble(post_samples)
ggplot(plot_dat,
       aes(x = post_samples)) +
  geom_density(adjust = 1/2,
               fill = "tomato")+
  coord_cartesian(xlim=c(0,1))
```



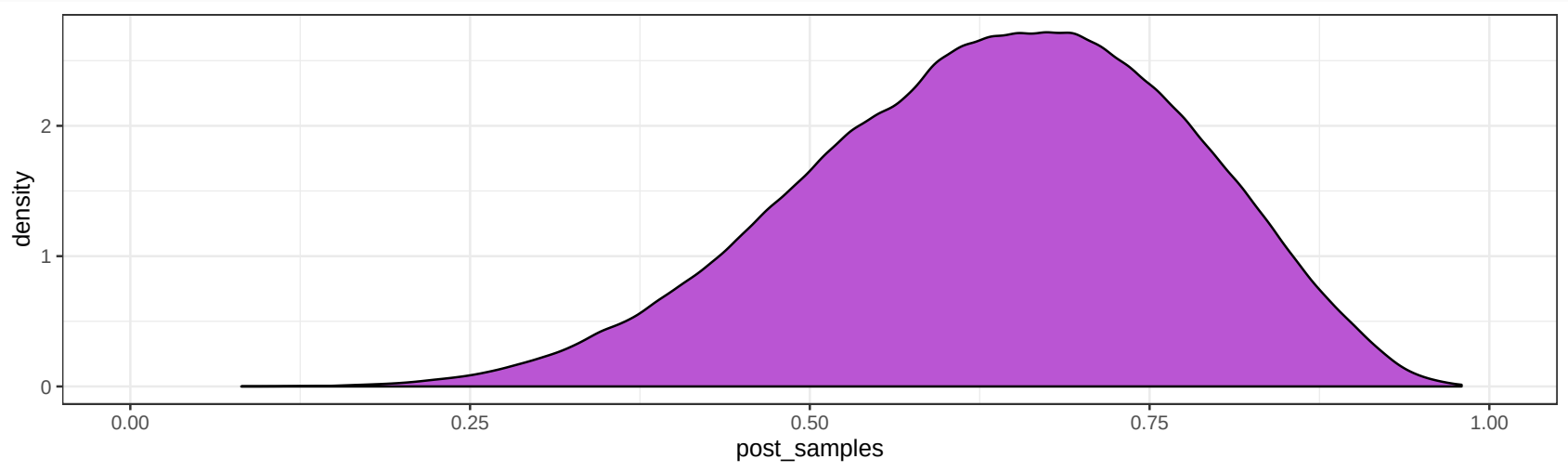
The impact of sample size on the posterior density: 10000 samples

```
post_samples<-sample(
  grid,
  prob = posterior,
  size = 10000,
  replace = T)

plot_dat<-tibble(post_samples)
ggplot(plot_dat,
       aes(x = post_samples)) +
  geom_density(fill = "dodgerblue")+
  coord_cartesian(xlim=c(0,1))
```



The impact of sample size on the posterior density: 100000 samples

```
post_samples<-sample(
  grid,
  prob = posterior,
  size = 100000,
  replace = T)

plot_dat<-tibble(post_samples)
ggplot(plot_dat,
       aes(x = post_samples)) +
  geom_density(fill = "mediumorchid") +
  coord_cartesian(xlim=c(0,1))
```



So what now?

What questions do we typically ask about a parameter?

So what now?

What questions do we typically ask about a parameter?

- What is the mean / expected value of the parameter?
- How certain are we about the location of the parameter?

Using posterior samples

What is $E(p)$

Using posterior samples

What is $E(p)$

```
mean(post_samples)
```

```
## [1] 0.6365078
```

Using posterior samples

What is $E(p)$

```
mean(post_samples)
```

```
## [1] 0.6365078
```

How certain are we about the location of p? Where is 92 percent of the posterior probability?

Using posterior samples

What is $E(p)$

```
mean(post_samples)
```

```
## [1] 0.6365078
```

How certain are we about the location of p? Where is 92 percent of the posterior probability?

```
quantile(post_samples, c(0.04, 0.96))
```

```
##        4%       96%
## 0.3877551 0.8571429
```

Using posterior samples

What is $E(p)$

```
mean(post_samples)
```

```
## [1] 0.6365078
```

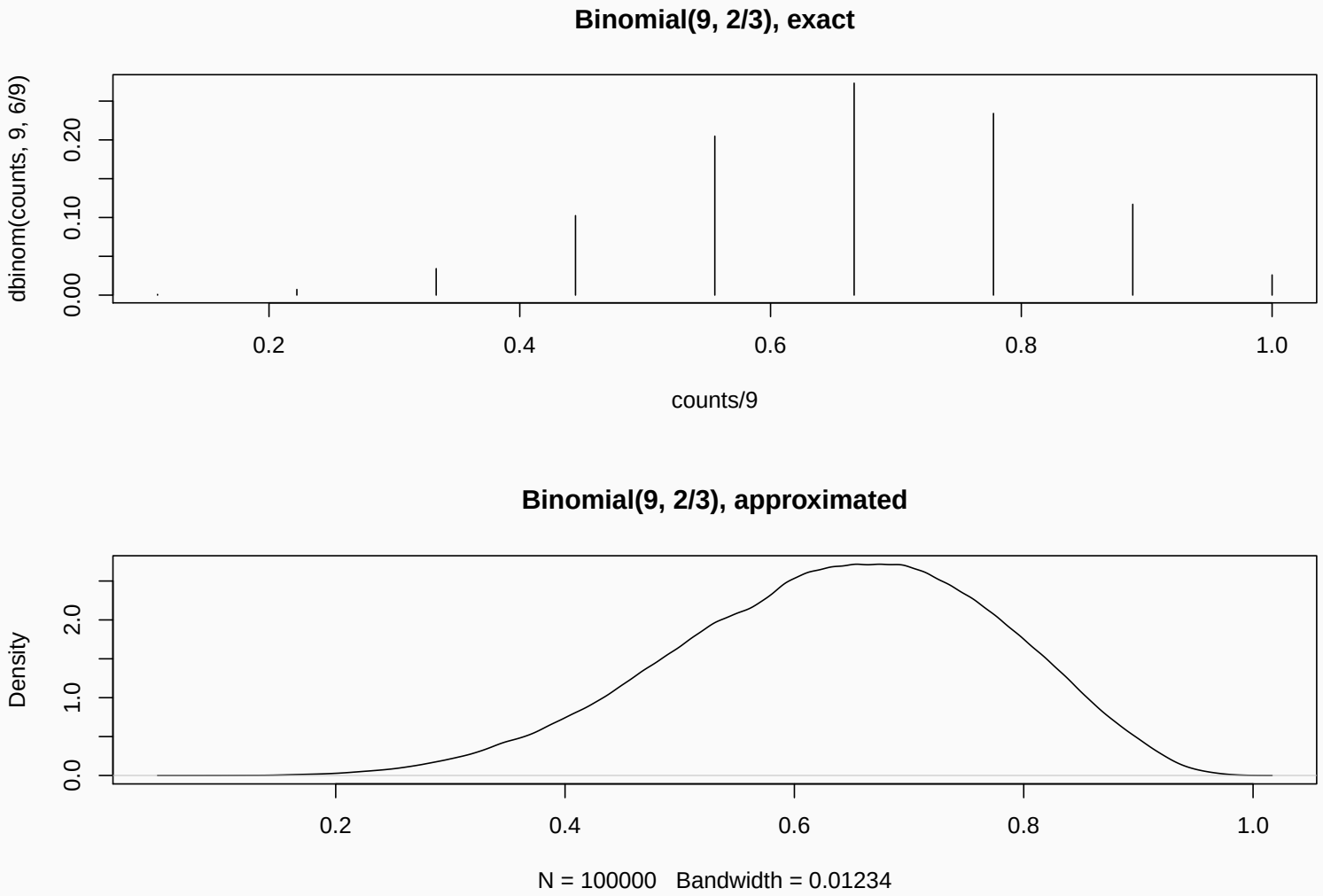
How certain are we about the location of p? Where is 92 percent of the posterior probability?

```
quantile(post_samples, c(0.04, 0.96))
```

```
##        4%       96%
## 0.3877551 0.8571429
```

Conditional on the data and our assumptions (priors, likelihood), we can describe the posterior *without* appealing to theoretical replications (!!!)

Approximation via sampling, brute force when exact solutions are difficult / undefined



Quadratic approximation

1. Assume our posterior is approximately Normal (often reasonable)
2. Find the mode of the psoterior
3. Estimate curvature at the logarithm of the posterior with a parabola
4. Use the resulting Gaussian / Normal distribution for inference
5. With enough data quap $\longrightarrow$ MLE

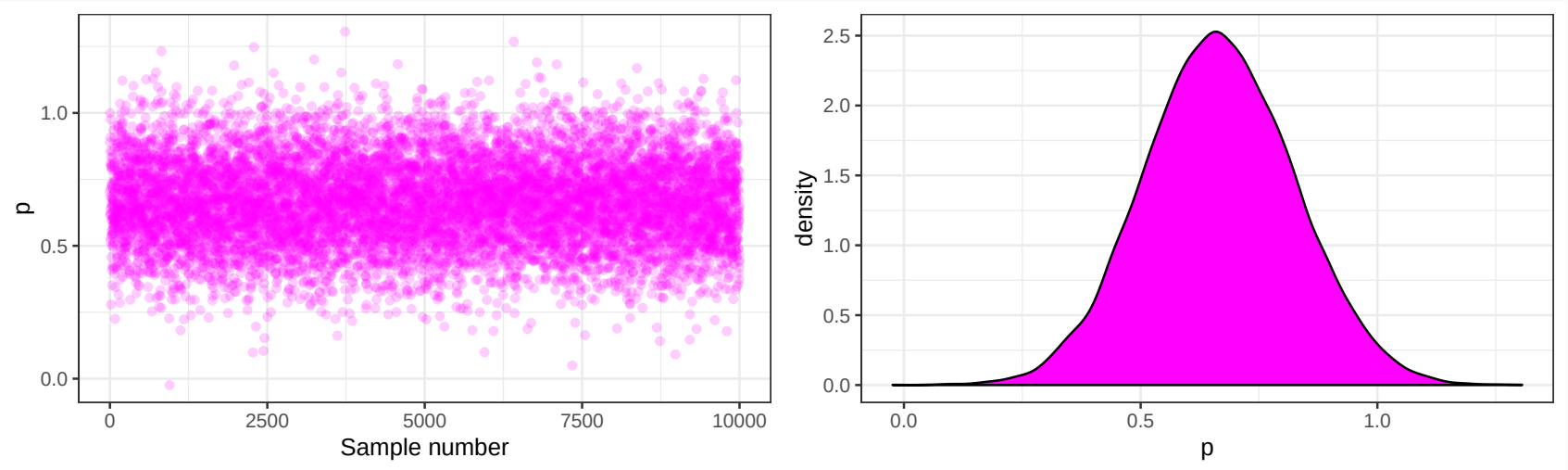
Using quadratic approximation in R

```
### let's treat 1 as water, 0 as land
dat<-list(W = 6, L = 3)

formula_list<-alist(
  # likelihood
  W ~ dbinom(W+L, p),
  # prior
  p ~ dunif(0,1)
)

model1<-quap(
  formula_list,
  data = dat
)

summary(model1)
```

```
##        mean        sd      5.5%     94.5%
## p 0.6666663 0.1571339 0.4155361 0.9177966
```

Extract posterior samples

```
post_samples<-extract.samples(model1)

p1<-ggplot(post_samples,
       aes(y = p, x = 1:nrow(post_samples))) +
  geom_point(alpha = 0.2, color = "magenta") +
  xlab("Sample number")

p2<-ggplot(post_samples,
       aes(x = p)) +
  geom_density(fill = "magenta")

grid.arrange(p1, p2, ncol = 2)
```



Using this object

```
str(post_samples)
```

```
## 'data.frame':    10000 obs. of  1 variable:
##  $ p: num  0.889 0.701 1 0.615 0.685 ...
##  - attr(*, "source")= chr "quap posterior: 10000 samples from model1"
```

```
summary(post_samples)
```

```
##        p
##  Min.   :-0.02396
##  1st Qu.: 0.55943
##  Median : 0.66402
##  Mean   : 0.66611
##  3rd Qu.: 0.77363
##  Max.   : 1.30579
```

Learning from the posterior

Questions we can ask with posterior samples:

- How much posterior probability lies (below/above/between) some parameter value(s)?
- Which parameter values mark the (lower/upper) n% of the posterior probability?
- Which parameter value has the highest posterior probability?

How much posterior probability lies below/above/between some parameter value(s)?

We can simply filter the samples, then divide by the size of the posterior sample.

Q: What is the posterior probability that there is less than 60 percent water on the globe, conditional on our model and the data?

```
post_samples %>%
  filter(p<0.6) %>%
  summarise(p = n()/nrow(post_samples))
```

```
##        p
## 1 0.3408
```

How much posterior probability lies below/above/between some parameter value(s)?

Q: What is the posterior probability that there is between 60 percent and 90 percent water on the globe, conditional on our model and the data?

```
post_samples %>%
  filter(p>=0.6 & p<=0.9) %>%
  summarise(p = n()/nrow(post_samples))
```

```
##        p
## 1 0.5885
```

Which parameter values mark the (lower/upper) n% of the posterior probability?

Q: What is the parameter region in which 89% of the posterior probability mass lies?

Which parameter values mark the (lower/upper) n% of the posterior probability?

Q: What is the parameter region in which 89% of the posterior probability mass lies?

```
quantile(post_samples$p, c(0.05, 0.94))
```

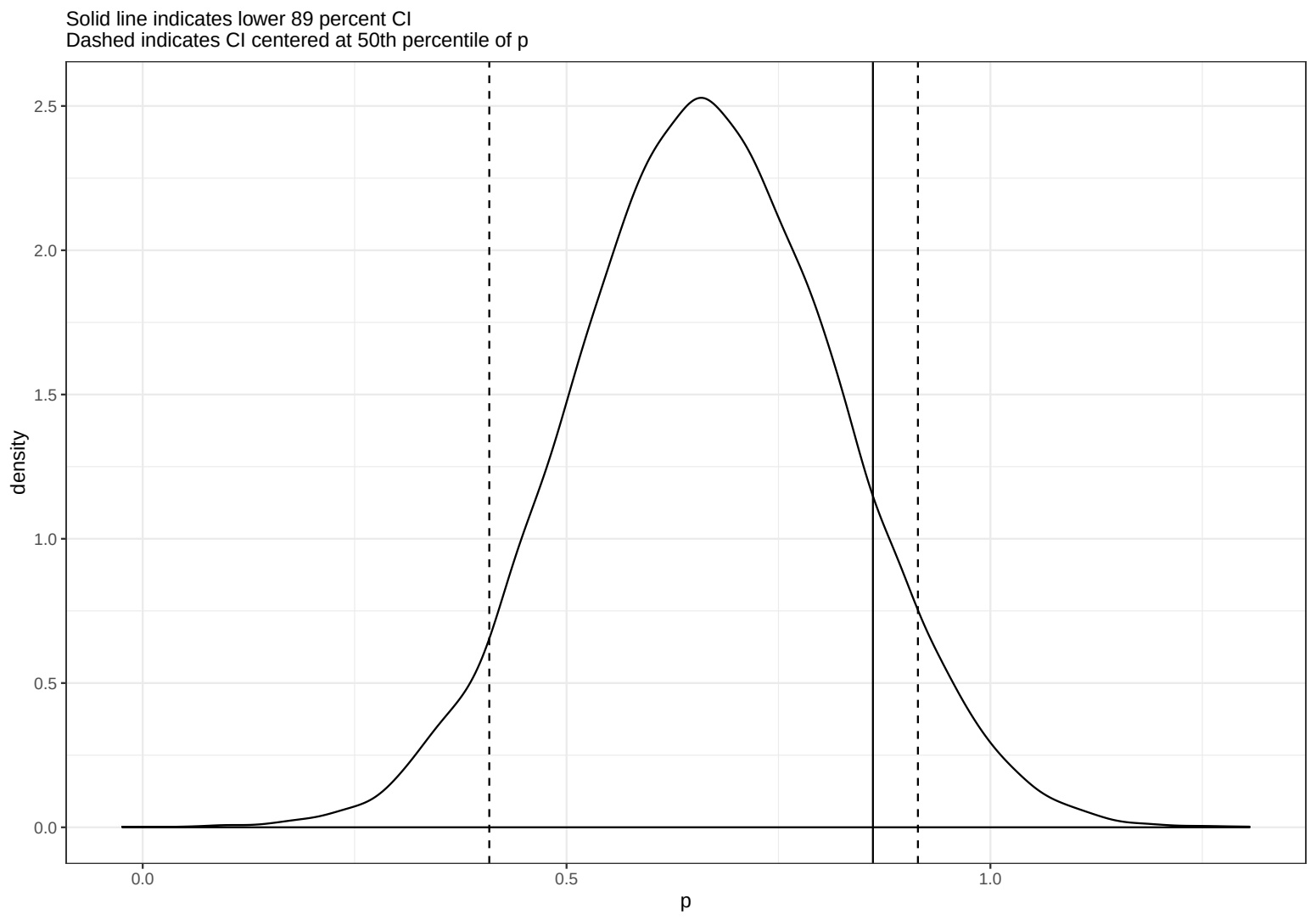
```
##        5%       94%
## 0.4088685 0.9145056
```

```
quantile(post_samples$p, 0.89)
```

```
##       89%
## 0.8614138
```

Thoughts on the differences between these two intervals?

What the intervals look like



Confidence intervals versus Bayesian credible / compatible intervals

- Confidence interval: If we replicated the experiment many times, the (fixed) true value would lie within this interval 89 percent of the time.

Confidence intervals versus Bayesian credible / compatible intervals

- Confidence interval: If we replicated the experiment many times, the (fixed) true value would lie within this interval 89 percent of the time.
- Credible interval: Conditional on our data and model (priors, likelihood), 89 percent of the compatible parameter values lie within this range.

Confidence intervals versus Bayesian credible / compatible intervals

- The frequentist confidence approach assumes the parameter value is fixed, randomness results from sampling the data

Confidence intervals versus Bayesian credible / compatible intervals

- The frequentist confidence approach assumes the parameter value is fixed, randomness results from sampling the data
- The Bayesian approach assumes that information about the parameter is random, not fixed

Confidence intervals versus Bayesian credible / compatible intervals

- The frequentist confidence approach assumes the parameter value is fixed, randomness results from sampling the data
- The Bayesian approach assumes that information about the parameter is random, not fixed
- Neither is correct, but the interpretation of the Bayesian interval has a far more intuitive interpretation

Note: There's nothing inherently scientific about 0.95. It is merely convention. Other intervals are just as valid. McElreath likes .89 because it is prime. I like 0.92 or 0.90 because they are even.

Point estimates

Which parameter value has the highest posterior probability?

```
chainmode(post_samples$p, adj = 0.01)
```

```
## [1] 0.6708742
```

This is the maximum a posteriori (MAP) estimate.

Point estimates

Which parameter value has the highest posterior probability?

```
chainmode(post_samples$p, adj = 0.01)
```

```
## [1] 0.6708742
```

This is the maximum a posteriori (MAP) estimate.

If our posterior is approximately symmetric, the MAP and median will be similar

```
median(post_samples$p)
```

```
## [1] 0.6640196
```

Simulating prediction

We've learned how to describe the parameter p with posterior samples.

Simulating prediction

We've learned how to describe the parameter p with posterior samples.

What would happen if we re-ran the experiment?

Simulating prediction

We've learned how to describe the parameter p with posterior samples.

What would happen if we re-ran the experiment?

- **Posterior predictive simulation** incorporates updated information on the parameter into predictions of new values of our observed outcome variable (W, L)

Posterior prediction algorithm

1. Sample from the posterior distribution
2. Use these sampled posterior parameter values to sample new values from the likelihood

Posterior prediction algorithm

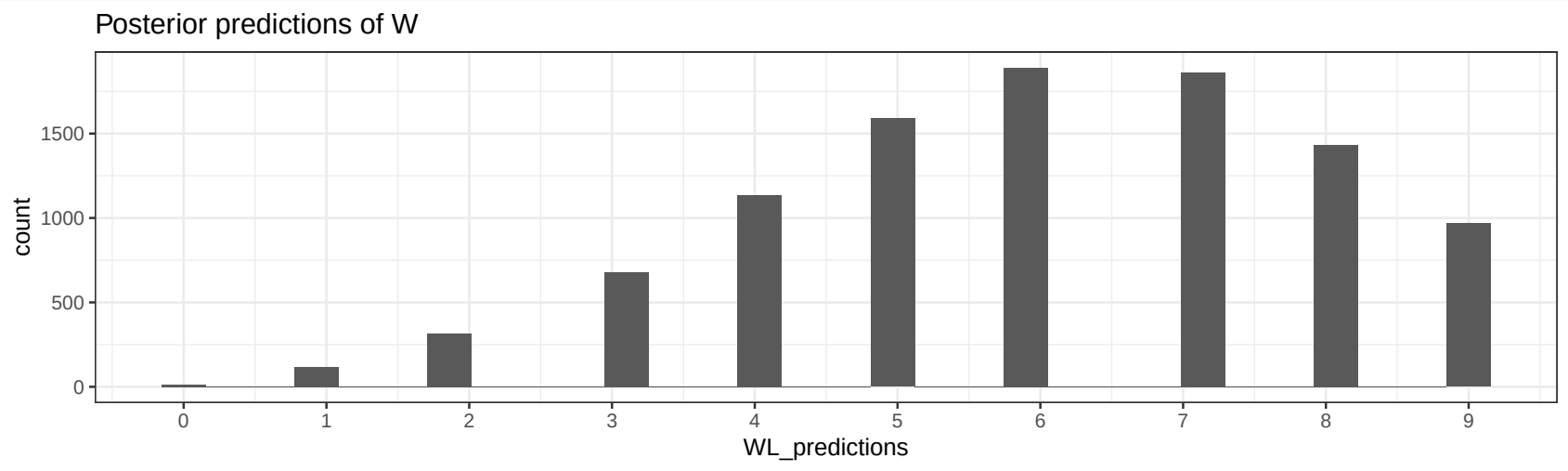
1. Sample from the posterior distribution
2. Use these sampled posterior parameter values to sample new values from the likelihood

```
# Pull samples of p from quap estimated model1
p_samples<-extract.samples(model1)
# because of impossible p >1 (Gaussian)
p_samples<-p_samples %>%
  mutate(p = ifelse(p>1, 1, p))
## sample from likelihood
WL_predictions<-rbinom(1e4, size = 9, prob = p_samples$p)
table(WL_predictions)
```

```
## WL_predictions
##    0    1    2    3    4    5    6    7    8    9
##   13  117  317  678 1136 1589 1890 1861 1431  967
```

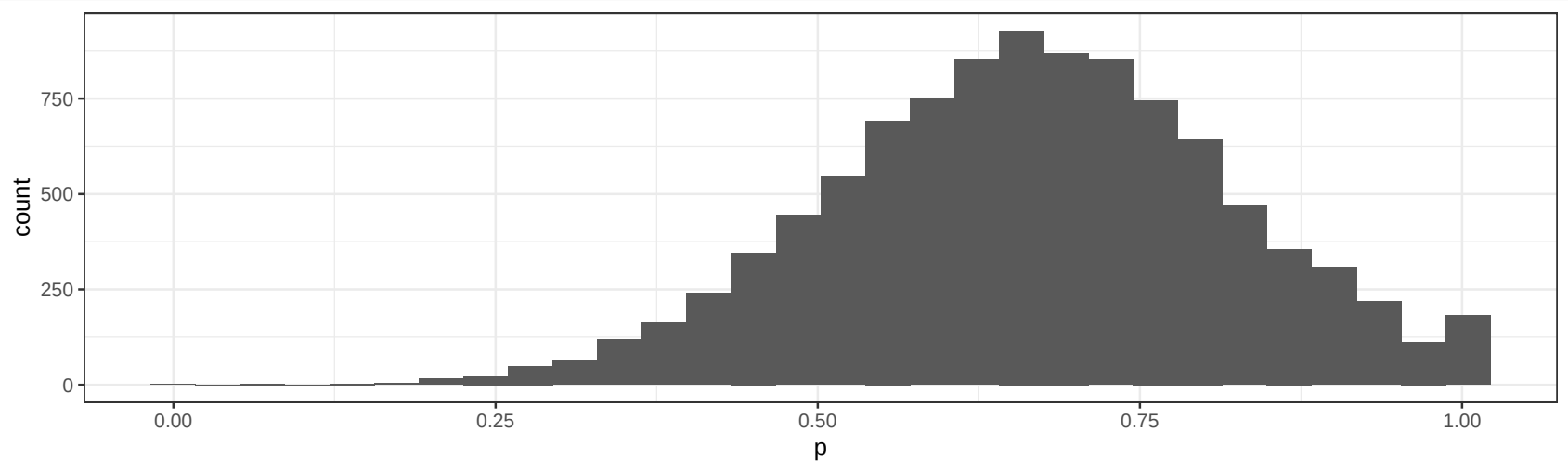
Visualize the predictions

```
ggplot(data.frame(WL_predictions),
       aes(x = WL_predictions)) +
  geom_histogram() +
  scale_x_continuous(breaks=0:9) +
  labs(title = "Posterior predictions of W")
```



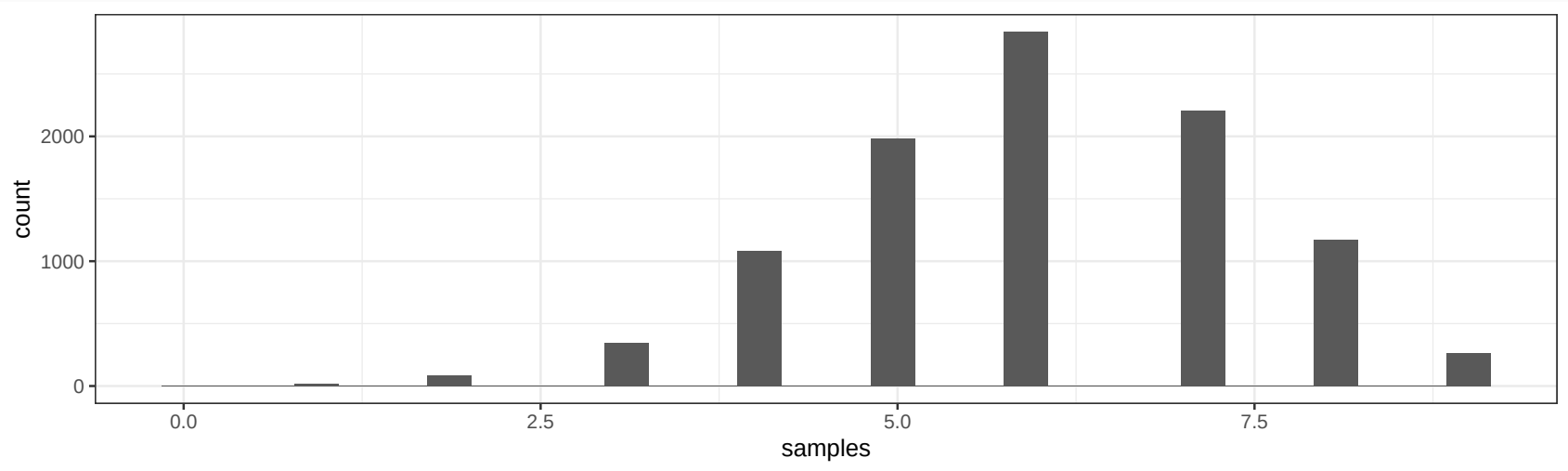
The impact of uncertainty in p on predictions

```
ggplot(p_samples, aes(x = p)) +
  geom_histogram()
```



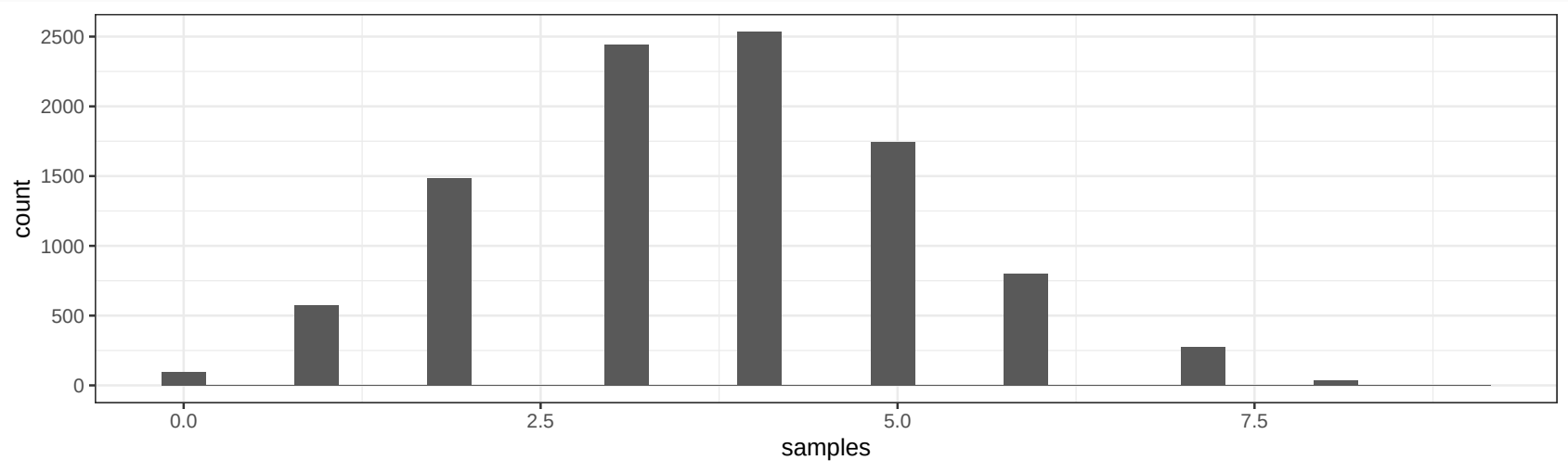
The impact of uncertainty in p on predictions: p = posterior median

```
## set p at the median of the posterior
samples<-rbinom(1e4, 9, median(p_samples$p))
ggplot(data.frame(samples), aes(x = samples)) +
  geom_histogram()
```



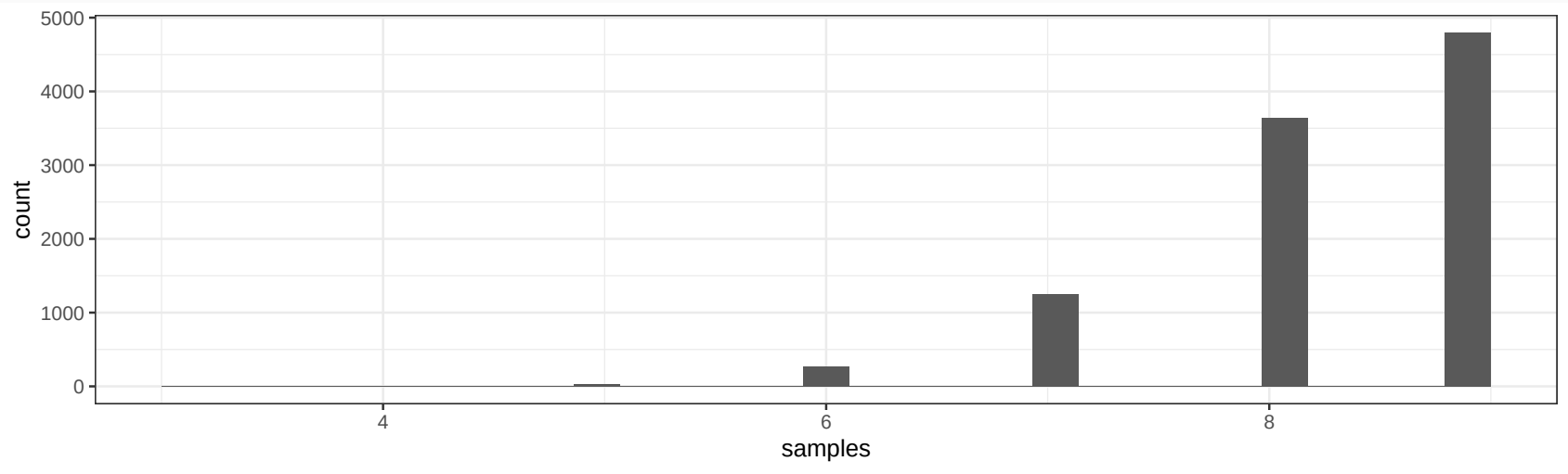
The impact of uncertainty in p on predictions: p = posterior 5th percentile

```
## set p at the 5th percentile of the posterior
samples<-rbinom(1e4, 9, quantile(p_samples$p, 0.05))
ggplot(data.frame(samples), aes(x = samples)) +
  geom_histogram()
```



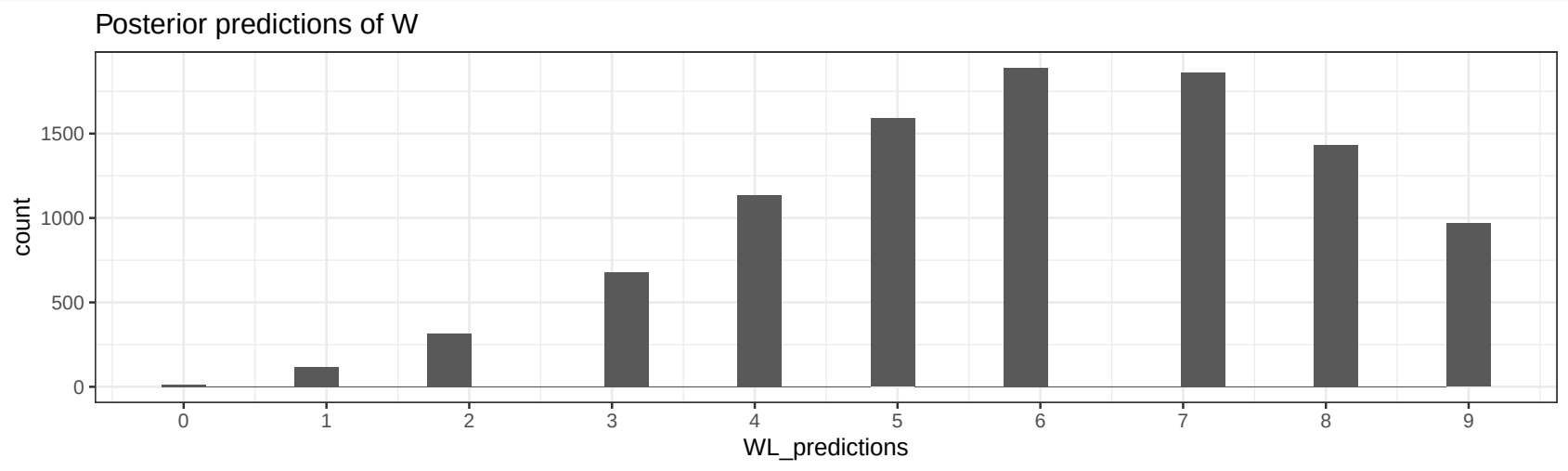
The impact of uncertainty in p on predictions: p = posterior 95th percentile

```
## set p at the 95th percentile of the posterior
samples<-rbinom(1e4, 9, quantile(p_samples$p, 0.95))
ggplot(data.frame(samples), aes(x = samples)) +
  geom_histogram()
```



Posterior predictive distributions propagate uncertainty in p into predictions

```
ggplot(data.frame(WL_predictions),
       aes(x = WL_predictions)) +
  geom_histogram() +
  scale_x_continuous(breaks=0:9) +
  labs(title = "Posterior predictions of W")
```



Prior prediction

We can also use simulation to better understand and select priors

Prior prediction

We can also use simulation to better understand and select priors

```
### our prior
formula_list
```

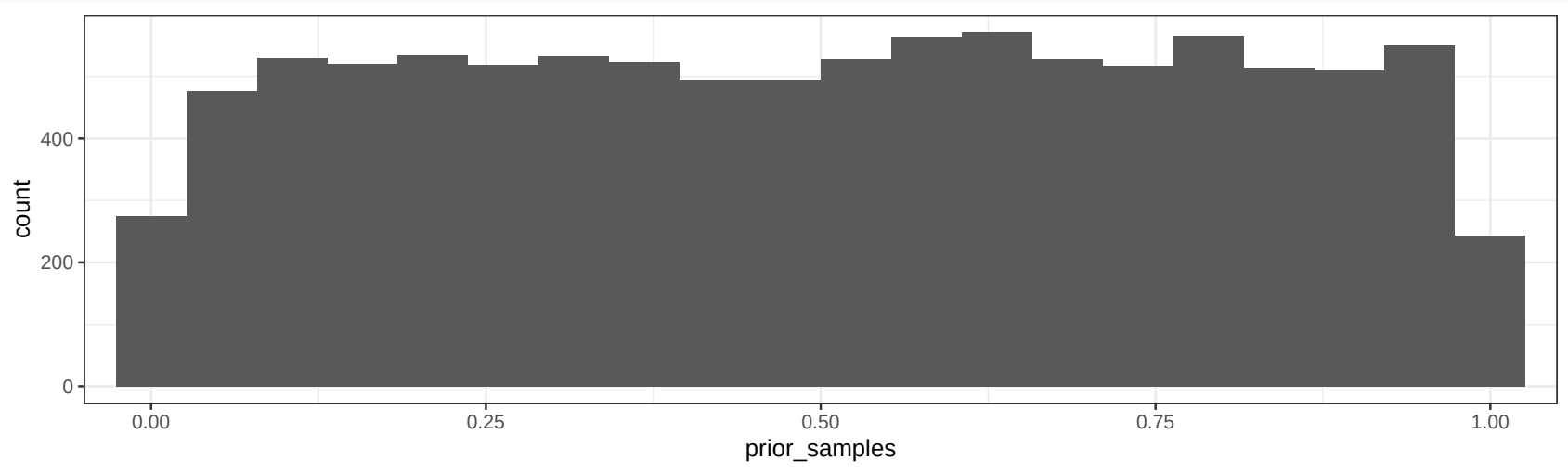
```
## [[1]]
## W ~ dbinom(W + L, p)
##
## [[2]]
## p ~ dunif(0, 1)
```

```
prior_samples<-runif(1e4, 0, 1)

head(prior_samples)
```

```
## [1] 0.2147788 0.3319766 0.2633456 0.9763804 0.7989332 0.7758785
```

Visualize the prior predictions

```
ggplot(data.frame(prior_samples),
                  aes(x = prior_samples)) +
  coord_cartesian(xlim = c(0, 1)) +
  geom_histogram(bins = 20)
```



Summary

- Simulation is a fundamental tool in Bayesian data analysis
- We learn a lot from visualizing samples from our estimated prior, posterior, and posterior predictive distributions
- These visuals help you carefully check assumptions and model performance
- Posterior prediction is a first step to moving past the typical point estimate / standard error presentation of results, and toward visual and interval-driven presentations
- HW 2 is posted in ./hw
- HW 1 solutions will be posted in ./hw