Lecture 6: The Normal Model

Professor Alexander Franks

2020-11-09

Announcements

- Reading: Chapter 5, Hoff

The Normal Distribution

- One of the most utilized probability models in data analysis
- Central Limit Theorem
- Separate parameters for the mean and the variance (intuitive)

Normal Distribution

- Symmetric with mode = median = mean = μ
- Approximately 95% of the population lies within two standard deviations of the mean
- Density:

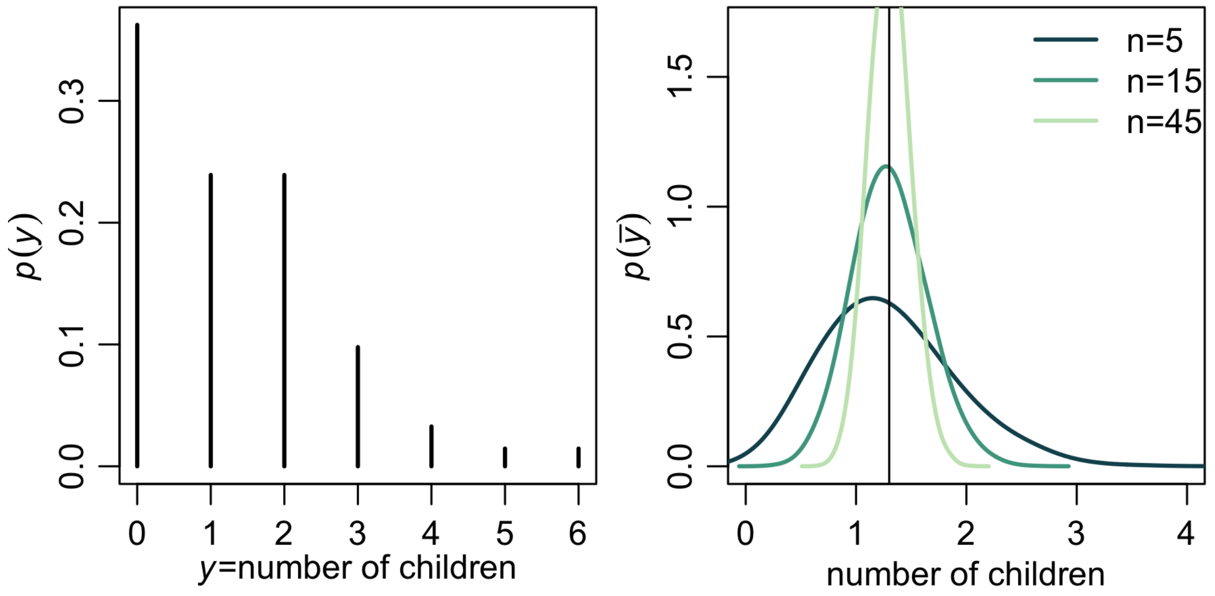
$$p(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2}, \quad -\infty < y < \infty$$

- $X \sim N(\mu_x, \sigma_x^2)$ and $Y \sim N(\mu_y, \sigma_y^2)$ with X and Y independent then

$$aX + bY \sim N(a\mu_x + b\mu_y, a^2\sigma_x^2 + b^2\sigma_y^2)$$

- In R: `dnorm, rnorm, pnorm, qnorm`.
    - Warning: the argument to the `norm` functions `R` is σ not σ^2!

The Central Limit Theorem



CLT: $\bar{y} \approx N(E[Y], \mathrm{Var}[Y]/n)$

Bayesian inference in the normal model

- Assume $y_1, \dots y_n \sim N(\mu, \sigma^2)$ with σ^2 a known constant
- Lets start with a non-informative, improper prior: $p(\mu) \propto \text{const}$
- What is the posterior distribution $p(\mu \mid y_1, \dots y_n, \sigma^2)$?

Bayesian inference in the normal model

- Assume $y_1, \dots y_n \sim N(\mu, \sigma^2)$ with σ^2 a known constant
- The normal prior distribution is conjugate for μ in the normal sampling model
- Sampling distribution, prior distribution and posterior distribution are all normal.
- Assume the prior is $p(\mu) \sim N(\mu_0, \tau^2)$
- What are the parameters of the posterior $p(\mu \mid y_1, \dots y_n, \sigma^2)$?

A conjugate prior for the normal likelihood

- The normal distribution is conjugate for the normal likelihood
    - Often called the "normal-normal model"
- $Y_i \sim N(\mu, \sigma^2)$ and $\mu \sim N(\mu_0, \tau^2)$ implies that the posterior distribution $p(\mu \mid y)$ is also normally distributed:

$$\mu \mid Y \sim N(\mu_n, \tau_n^2)$$

where $\mu_n = \frac{\frac{1}{\tau^2}\mu_0 + \frac{n}{\sigma^2}\bar{y}}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}$ and $\tau_n^2 = \frac{1}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}$

The posterior mean and pseudo-counts

$$\mu_n = \frac{\frac{1}{\tau^2}}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}\mu_0 + \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}\bar{y}$$
$$= (1 - w)\mu_0 + w\bar{y}$$

where $w = \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau^2} + \frac{n}{\sigma^2}}$

Can we think about the normal prior parameters in terms of pseudo-counts?

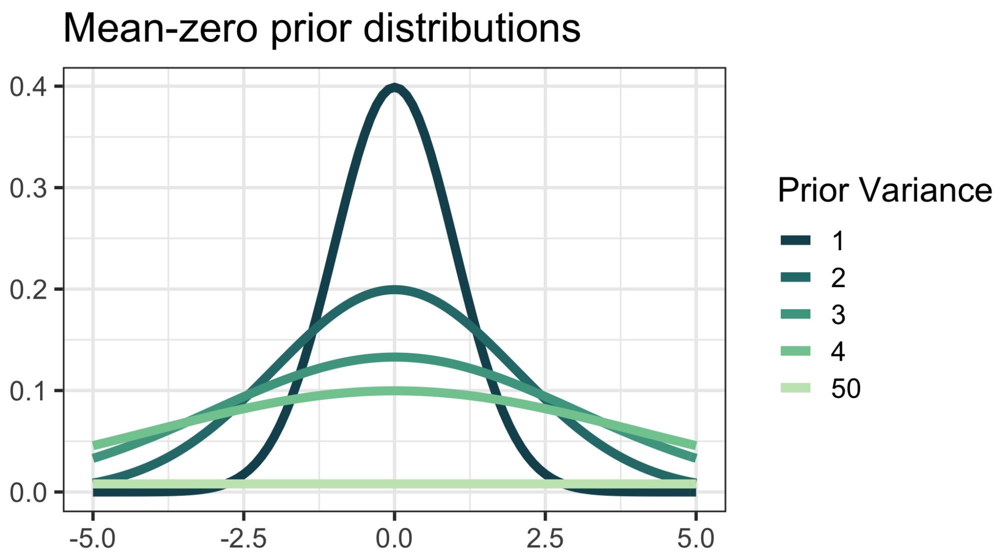
The posterior mean and pseudo-counts

$$\begin{aligned}\mu_n &= \frac{\frac{1}{\tau^2}}{\frac{1}{\tau^2}+\frac{n}{\sigma^2}}\mu_0 + \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau^2}+\frac{n}{\sigma^2}}\bar{y} \\ &= (1-w)\mu_0 + w\bar{y}\end{aligned}$$

where $w = \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau^2}+\frac{n}{\sigma^2}}$

- Let's reparameterize: $\tau^2 = \frac{\sigma^2}{\kappa_0}$
- Then: the posterior variance is $\frac{\sigma^2}{\kappa_0+n}$
- And: $(1-w) = \frac{\kappa_0}{\kappa_0+n}$
- κ_0 are the prior counts and μ_0 is the prior sample average.

Conjugate prior with increasing variance



Bayes Estimators

Estimators: Bayes / Frequentist Unification

- Bayesian inference provides a straightforward procedure for producing estimators given your prior beliefs.

    1. Compute posterior distribution

    2. Summarize the posterior distribution with a point estimator (e.g. posterior mean or posterior mode) and a probability interval

- Frequentists provide tools for evaluating the sampling properties of an estimator.

    - Bias, variance and MSE of an estimator
    - Well-calibrated probability intervals

- Both are useful!

The Bias-Variance Tradeoff

Reminder: an estimator is a random variable, an estimate is a constant

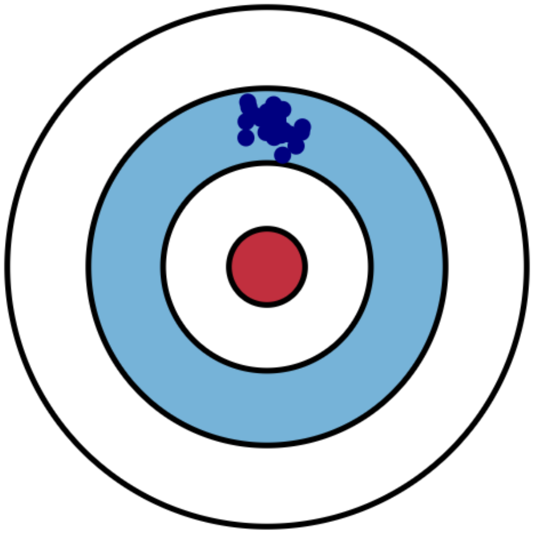
- *Bias*: systematic sampling error of the estimator
- *Variance*: variance of the estimator (from sampling & measurement error)
- Often we evaluate an estimator in terms of mean square error: $\mathrm{MSE}(\hat{\theta}) = E_Y(\hat{\theta} - \theta)^2$
- The Bias-Variance tradeoff: $\mathrm{MSE}(\hat{\theta}) = \mathrm{Var}(\hat{\theta}) + \mathrm{Bias}(\hat{\theta})^2$

The Bias-Variance Tradeoff

- Variance of an estimator comes sampling from a population
  - If you were to repeatedly draw new samples of the same size how much would your estimates vary?
  - e.g. if $y_i \sim N(\mu, \sigma^2)$ then $\mathrm{Var}(\bar{Y}) = \sigma^2/n$

Bias

The expected difference between the estimate and the response

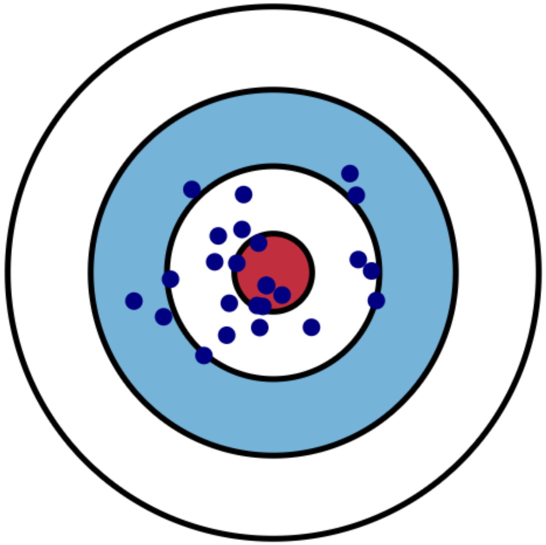


Statistical definition of bias:

$$E_Y[\hat{\theta} - \theta]$$

Variance

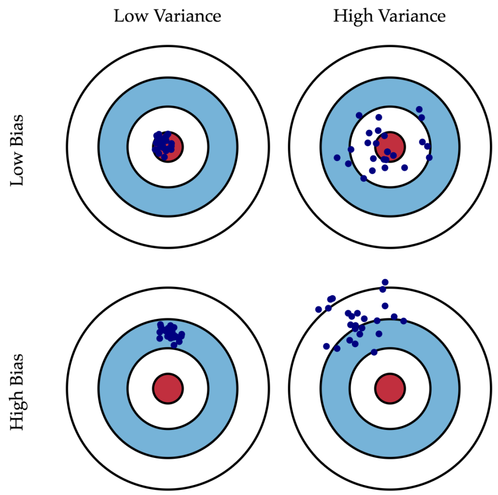
How variable is the prediction about its mean?



Statistical definition of variance:

$$E_Y[\hat{\theta} - E_Y[\hat{\theta}]]^2$$

Bias and Variance



$$\underbrace{\mathrm{MSE}(\hat{\theta})}_{\text{accuracy}} = \underbrace{\mathrm{Var}(\hat{\theta})}_{\text{variance}} + \underbrace{\Big(E[\hat{\theta} - \theta]\Big)^2}_{\text{bias squared}}$$

The Bias-Variance Tradeoff

- The prior distribution (usually) makes your estimator biased...
- But the prior distribution also (usually) reduces the variance!
- Example: compute the frequentist mean and variance of the posterior mean.

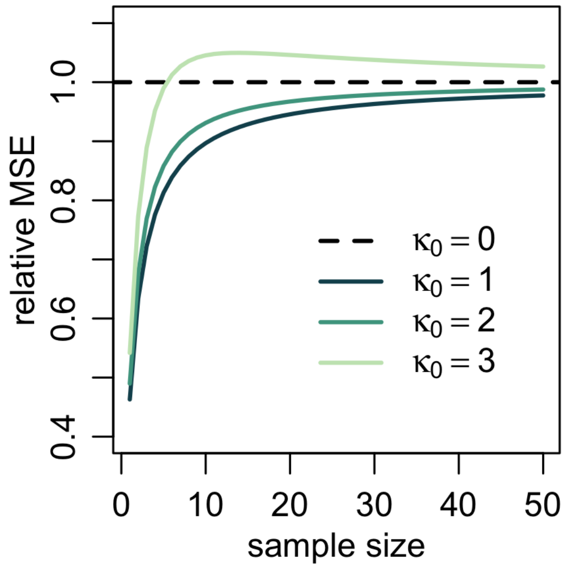
Example: IQ scores

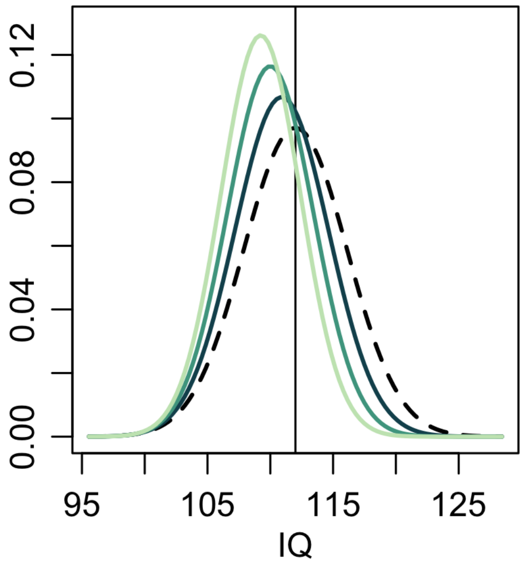
- Scoring on IQ tests is designed to yield a N(100, 15) distribution for the general population
- We observe IQ scores for a sample of n individuals from a particular town and estimate μ, the town-specific IQ score
- If we lacked knowledge about the town, a natural choice would be $\mu_0 = 100$
- Suppose the true parameters for this town are $\mu = 112$ and $\sigma = 13$
    - The town is smarter on average than the general population

Example: IQ scores

- What is the mean squared error of the MLE? MSE of the posterior mean?
- $\text{MSE}[\hat{\mu}_{MLE}] = \text{Var}[\hat{\mu}_{MLE}] = \frac{\sigma^2}{n} = \frac{169}{n}$
- $\text{MSE}[\hat{\mu}_{PM}|\theta_0] = w^2 \frac{169}{n} + (1-w)^2 144$
- Reminder: $w = \frac{n}{\kappa_0 + n}$. For what values of n and κ_0 is the MSE smaller for the posterior mean estimator than the maximum likelihood?

Example: IQ scores





Decision Theory

Why the posterior mean?

- Often times we need to make a "decision" by providing a single estimate
- The posterior provides a full distribution over θ, which can be summarized in infinitely many ways
- Specify a *loss function* which describes the cost of estimating $\hat{\theta}$ when the truth is θ

Bayes Estimators

- The *loss function*: $L(\hat{\theta}, \theta)$
  - Squared error: $L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2$
  - Absolute error: $L(\hat{\theta}, \theta) = |\hat{\theta} - \theta|$
- The **Bayes risk** is the posterior expected loss:

$$E_{\theta|y}[L(\hat{\theta}, \theta)] = \int L(\hat{\theta}, \theta) p(\theta \mid y) d\theta$$

- Choose an estimator of θ based on minimizing the Bayes risk.
- An estimator $\hat{\theta}$ is said to be a **Bayes estimator** if it minimizes the Bayes risk among all estimators.

Squared error loss

$$\min_{\hat{\theta}} E_{\theta|y}(\hat{\theta} - \theta)^2 = \min_{\hat{\theta}} \int (\hat{\theta} - \theta)^2 p(\theta \mid y) d\theta$$

Differentiate with respect to $\hat{\theta}$ and set equal to zero:

Absolute loss

$$\min_{\hat{\theta}} E_{\theta|y}|\hat{\theta} - \theta| = \min_{\hat{\theta}} \int |\hat{\theta} - \theta| p(\theta \mid y) d\theta$$

Differentiate with respect to $\hat{\theta}$ and set equal to zero:

Loss functions in practice

- Squared error and absolute error are good default loss functions
  - Motivated largely by mathematical considerations
- In practice we should define a loss function specific to our problem
- Loss in dollars? Loss in "quality of life"?

Decision making: flu example

- The CDC produces estimates of the expected prevalance and severity of flu during flu season
- Assume θ represents severity of the flu
- $p(\theta \mid y)$ is CDC posterior distribution based on initial data about the upcoming flu season
- $\hat{\theta}$ determines how much flu vaccine to make. How do we determine $L(\hat{\theta}, \theta)$?

Normal distribution, unknown variance

Known mean, unknown variance

- Assume we have n mean-zero normal observations with variance σ^2
- Define $d_i = (y_i - \mu)$ for notation convenience
- What is $p(\sigma^2 \mid \mu, d_1, \ldots d_n)$?

Joint inference for normal parameters

- In practice, both μ and σ are unknown in the normal model.
- We've considered models when either μ is unknown but σ is known.
- In practice neither is none!
- Need to specify a joint prior distribution: $p(\mu, \sigma)$
- This is our first look at *multi-parameter* models.

Joint inference for the mean and variance

In the normal model we typically factorize the prior distribution $p(\mu, \sigma) = p(\mu \mid \sigma)p(\sigma)$.

Specifically:

$$\begin{aligned} \sigma &\propto \frac{1}{\sigma^2} \\ \mu|\sigma &\sim \text{ normal } \left(\mu_0, \sigma^2/\kappa_0\right) \\ Y_1, \ldots, Y_n|\mu, \sigma &\sim \text{ i.i.d. normal} \left(\mu, \sigma^2\right) \end{aligned}$$

- μ_0 is interpreted as the prior sample mean
- κ_0 is a prior sample size

Example: midge wing length

- Modeling wing length of different specifies of midge (small, two-winged flies)
- From prior studies: mean wing length close to 1.9mm.
- Prior mean for μ is $\mu_0 = 1.9$ and non-informative prior for σ.
- Prior sample sizes: choose $\kappa_0 = 1$
- $(\bar{y}, s^2) = (1.804, 0.0169)$ are the sufficient statistics

Working with the log posterior

- As always, we will write down $p(\theta \mid y) \propto p(y \mid \theta)p(\theta)$
- In code, we always work with the **log-posterior** for numerical reasons
  - Mathematically it makes no difference, but computationally it is important
  - $L(\theta) \propto \prod p(y_i \mid \theta)$ is very small for moderate sample size (underflow)
  - $\ell(\theta) = \sum log(p(y_i \mid \theta))$ is numerically stable
- Monte Carlo methods only require that we can evaluate the log posterior

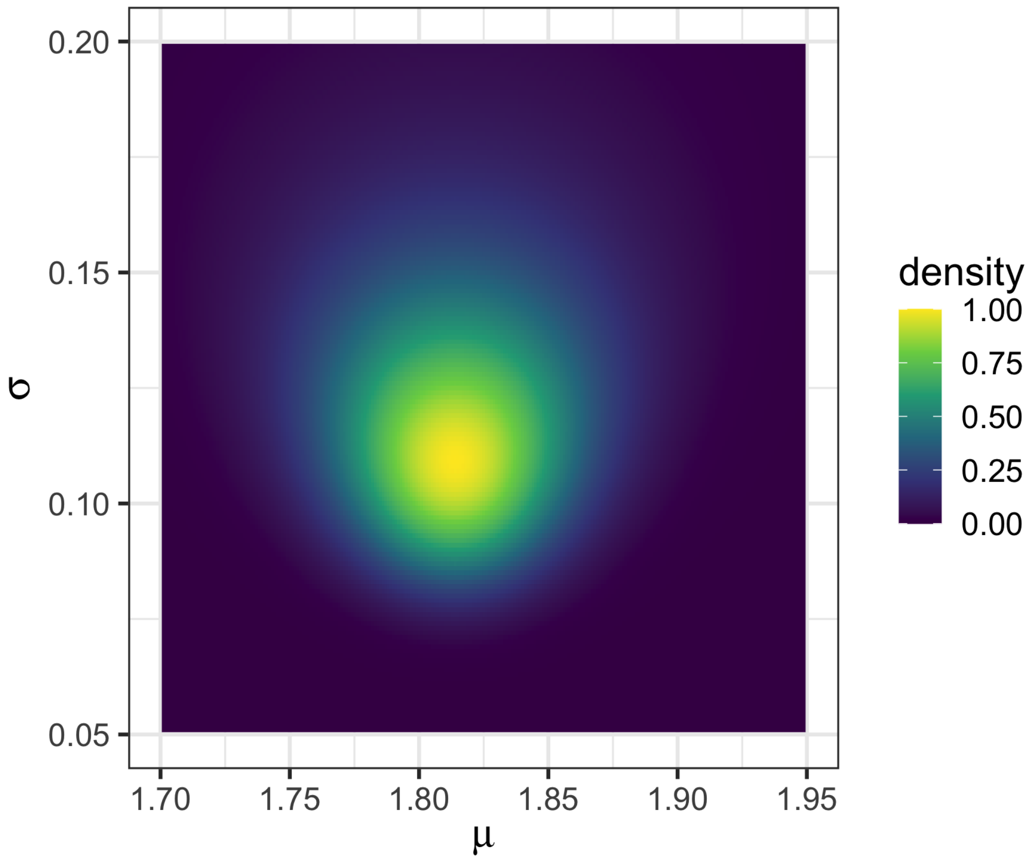
Grid approximation to the posterior distribution

```r
log_normal_posterior <- Vectorize(function(mu, sigma) {

  ### log likelihood
  sum(dnorm(y, mu, sigma, log=TRUE)) +
  ## plus log prior
  dnorm(mu, mu0, sigma/sqrt(k0), log=TRUE) +
  -2*log(sigma)

})


post_grid <- as_tibble(
  expand.grid(seq(1.6, 2.0, by=0.001),
              seq(0.01, 0.25, by=0.001)))
colnames(post_grid) <- c("mu", "s")
```

Grid approximation to the posterior distribution

```
post_grid %>%
  mutate(log_density = log_normal_posterior(mu, s)) %>%
  mutate(density = exp(log_density - max(log_density)) %>%
  ggplot() +
  geom_raster(aes(mu, s, fill=exp(log_density))) +
  xlim(c(1.7, 1.95)) + ylim(c(0.05, 0.2)) +
  xlab(expression(mu)) +
  ylab(expression(sigma)) +
  theme_bw() +
  scale_fill_continuous(type="viridis")
```

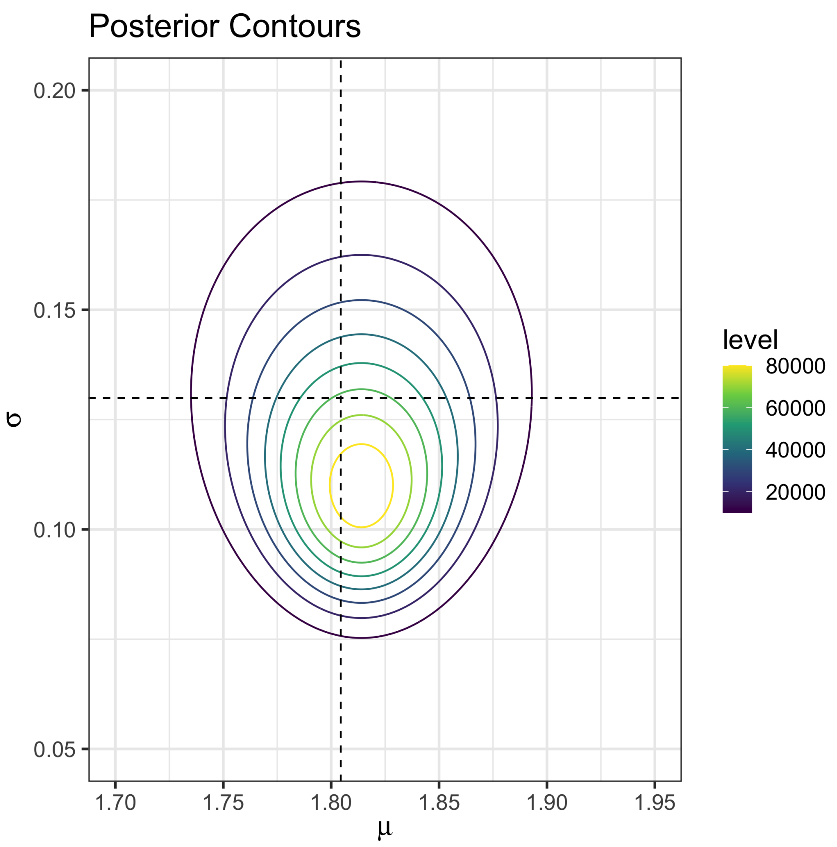
Grid approximation



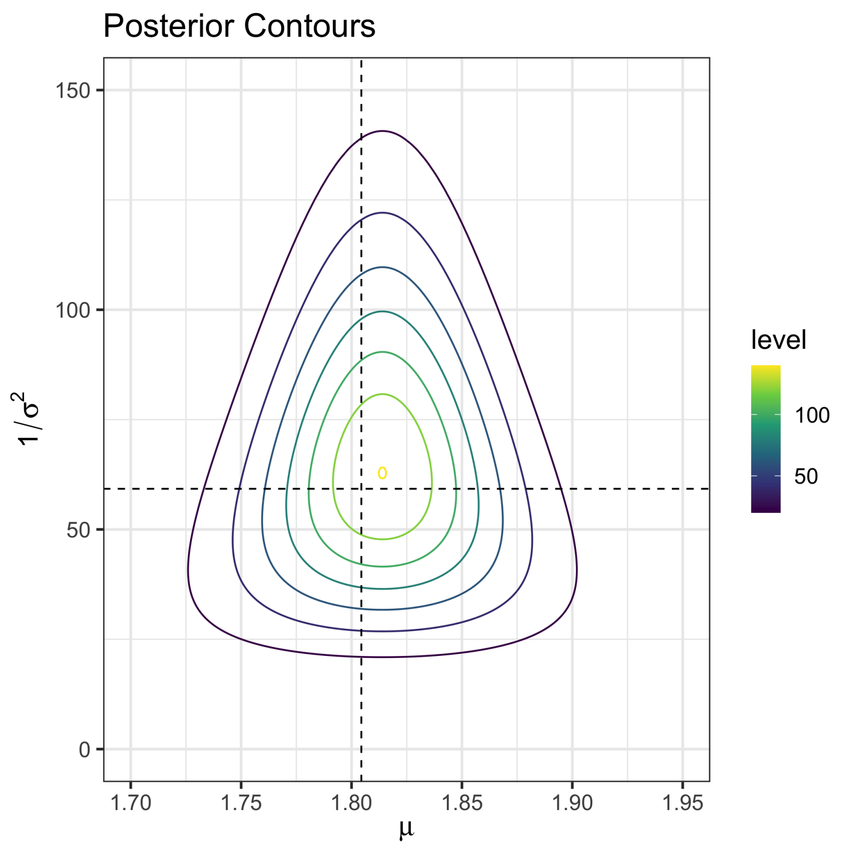
Contour Plot

```
post_grid %>%
  mutate(density = exp(log_density - max(log_density))) %>%
  ggplot() +
  geom_contour(aes(mu, s, z=density, colour=stat(level)), size=2)
  xlim(c(1.7, 1.95)) + ylim(c(0.05, 0.2)) +
  xlab(expression(mu)) + ylab(expression(sigma)) +
  ggtitle("Posterior Contours") +
  theme_bw(base_size=16) +
  scale_color_continuous(type="viridis")
```

Contour Plot (Standard Deviation)



Contour Plot (Precision)



Sampling from the joint posterior

- Contour and raster plots allow us to visualize the posterior (in two dimensions)
  - Need to know approximately where the high posterior density is (not easy)
- When we have more than 2 parameters visualization isn't feasible
- How do we summarize the posterior?
  - e.g. posterior means, posterior probabilities, intervals, etc..

Sampling from the joint posterior

- One approach: convert multi-parameter distribution into the product of many one parameter distributions.

Markov Chain Monte Carlo

- Markov Chain Monte Carlo (MCMC)
- More effective approach to sampling from multi-parameter distributions
- Samples in MCMC are **not** independent samples

MCMC Sampling with Stan

Stan

"A state-of-the-art platform for statistical modeling and high-performance statistical computation."

http://mc-stan.org/

Monte Carlo Sampling with Stan

- Stan is a "probabilistic programming language" with its own simple syntax
- R interface to Stan via `rstan` package
- Define parameters θ and datatypes y
- Provide sampling model $p(y \mid \theta)$ and prior $p(\theta)$
- Stan translates model syntax into the log posterior density $\log(p(\theta \mid y))$ and automatically generates samples from this density

Example Stan File

```
// The input data is a vector 'y' of length 'N'.
data {
  int<lower=0> N;
  real<lower=0> k0;
  vector[N] y;
}

// The parameters accepted by the model. Our model
// accepts two parameters 'mu' and 'sigma'.
parameters {
  real mu;
  real<lower=0> sigma;
}

// The model to be estimated. We model the output
// 'y' to be normally distributed with mean 'mu'
// and standard deviation 'sigma'.
model {
  target += -2*log(sigma); //sigma prior
  mu ~ normal(1.9, sigma^2/k0); // mu prior
  y ~ normal(mu, sigma);
}
```

The Stan File

- A stan file ends in `.stan`
- Three important program blocks in a stan file:
    - Data block
    - Parameter block
    - Model block
    - Each blco kencapsulated in brackets, { ... }.
- Stan needs data types:
    - `int`: integer valued data
    - `real`: continuous data or parameters
- Need to end every line with a semi-colon!
- Need to compile the Stan program before running.

Defining the input data in Stan

```
// The input data is a vector 'y' of length 'N'.
data {
  int<lower=0> N;
  real<lower=0> k0;
  vector[N] y;
}
```

Defining the model parameters in Stan

```
// The parameters accepted by the model. Our model
// accepts two parameters 'mu' and 'sigma'.
parameters {
  real mu;
  real<lower=0> sigma;
}
```

Defining the model in Stan

```
// The model to be estimated. We model the output
// 'y' to be normally distributed with mean 'mu'
// and standard deviation 'sigma'.
model {
  target += -2*log(sigma); //sigma prior
  mu ~ normal(1.9, sigma^2/k0); // mu prior
  y ~ normal(mu, sigma); // data model
}
```

Running Stan in R

```
## Load the rstan library
library(rstan)
```

```
## Loading required package: StanHeaders

## rstan (Version 2.21.2, GitRev: 2e1f913d3ca3)

## For execution on a local, multicore CPU with excess RAM we recommend cal
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)

##
## Attaching package: 'rstan'

## The following object is masked from 'package:tidyr':
##
##     extract
```

```
# compile stan model.  This may take a minute.
stan_model <- stan_model(file="normal_model.stan")
```

```
## Trying to compile a simple C file

## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
```

Running Stan in R

```
## Load the rstan library
library(rstan)

# compile stan model.  This may take a minute.
stan_model <- rstan::stan_model(file="normal_model.stan")

## data is a list, arguments must match the
## arguments in datablock in the stan file
stan_fit <- rstan::sampling(stan_model,
                            data=list(N=n, y=y, k0=k0),
                            refresh=0)
```

Stan: The Basics

```r
library(rstan)

# compile stan model.  This may take a minute.
stan_model <- rstan::stan_model(file="normal_model.stan")

## data is a list, arguments must match the datablock in the stan
stan_fit <- rstan::sampling(stan_model,
                            data=list(N=n, y=y, k0=k0),
                            refresh=0)
```

Stan: The Basics

```
## Convert Stan output to a list of MC samples
samples <- rstan::extract(stan_fit)

## names in list are the defined parameters
names(samples)
```

```
## [1] "mu"    "sigma" "lp__"
```

```
mu_samples <- samples$mu
sigma_samples <- samples$sigma

## By default there will be 4000 samples
length(mu_samples)
```

```
## [1] 4000
```

Stan: The Basics

```r
## Information about MC samples
summary(mu_samples)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.642   1.789   1.815   1.815   1.841   2.028
```

```r
summary(sigma_samples)
```

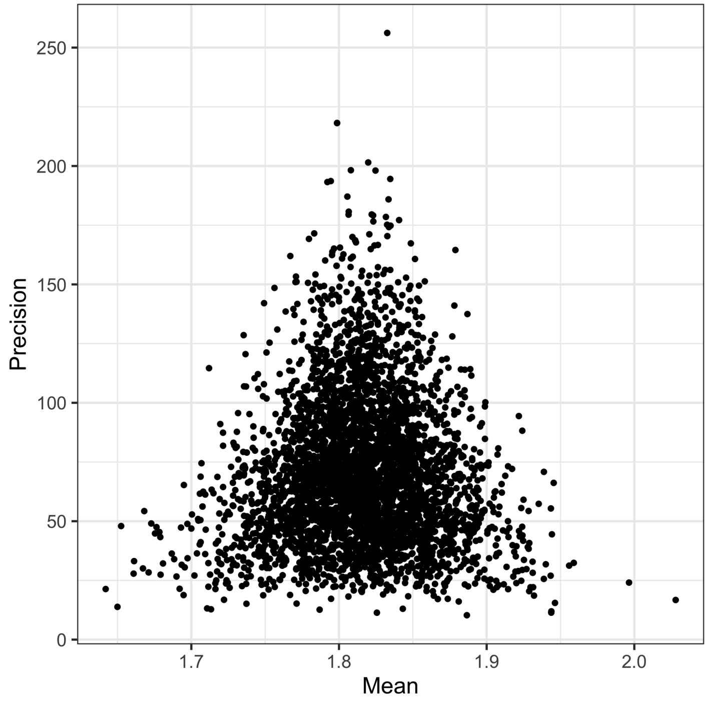
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.06248 0.10583 0.12218 0.12738 0.14352 0.31161
```

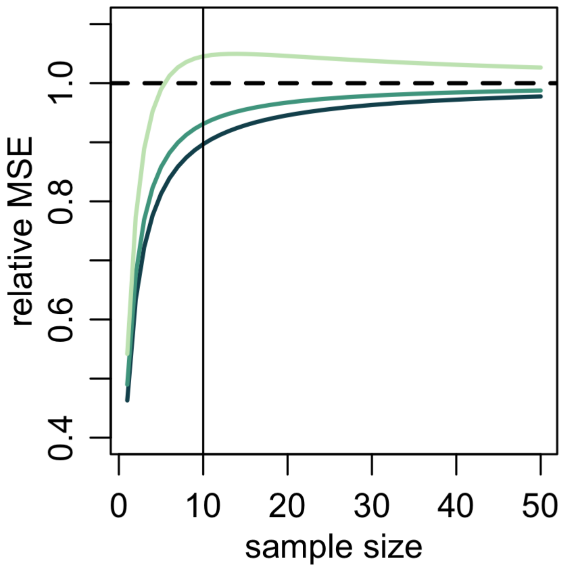
```r
summary(1/sigma_samples^2)
```

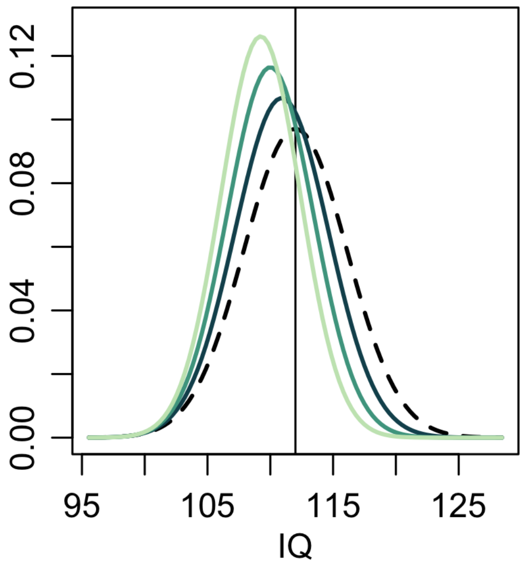
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.30   48.55   66.98   71.51   89.29  256.19
```

Visualizing Posterior Samples from Stan

```
tibble(Mean = mu_samples, Precision=1/sigma_samples^2) %>%
  ggplot() +
  geom_point(aes(x=Mean, y=Precision)) +
  theme_bw(base_size=16)
```




relative MSE
0.4
0.6
0.8
1.0
0
10
20
30
40
50
sample size


0.00
0.04
0.08
0.12
95
105
115
125
IQ