Exercise 6: Iris data, Linear Regression

Jason Doll

June 8, 2018

Objective

This exercise will use skills obtained in previous exercises to create the linear regression model.

Background

In this example, we will use the Iris data set that is included in base R. You will write a stan program to estimate parameters of a linear regression model to predict sepal width from sepal length. The R code to initialize workspace, read data, and package and send the data with initial values to Stan are provided in the "Ex6_ext_lr.R" file located in the "Ex6_Iris" folder. This is a self driven exercise, review the R code then create the .stan file. The model you will fit to individual observations i is:

$$
\begin{aligned}
\text{Model:} \quad & y_i = \alpha + \beta X_i + \epsilon_i \\
& \epsilon_i \sim \text{Normal}(0, \sigma) \\
\text{Priors:} \quad & \alpha \sim \text{Normal}(0, 100) \\
& \beta \sim \text{Normal}(0, 100) \\
& \sigma \sim \text{half-cauchy}(0, 5)
\end{aligned}
$$

Where Y = sepal width and X = sepal length

R packages required for this exercise

1. rstan

Directions

Create a new text file and save it as "Ex6_est_lr.stan" in the "Ex6_Iris" folder. Using the code from previous exercises, create the neccessary stan model. The R code and final figures are below.

R Code

Load library, clear workspace, and load data

```
#####CSTAT Workshop###############
#####Exercise 5###################
#####Analysis of Iris data########
#####File provided by instructor##
###################################

#install/load the rstan package
require(rstan)
```

```
## Loading required package: rstan
```

```
## Loading required package: ggplot2

## Loading required package: StanHeaders

## rstan (Version 2.17.3, GitRev: 2e1f913d3ca3)

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

```
#Set working directory to source file locations
#In Rstudio
#This directory must have all data files and Stan model code needed.
#If you receive an error, manually set the working directory
#setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

#clear workspace
rm(list=ls())

#Generate simulated data
#Load Iris data

iris = iris

#specify number of chains, used to initialize values and specify chains
nchains = 3

# Specify data:
dataList = list(
  'n'=nrow(iris),
  'x'=iris$Sepal.Length,
  'y'=iris$Sepal.Width
)
```

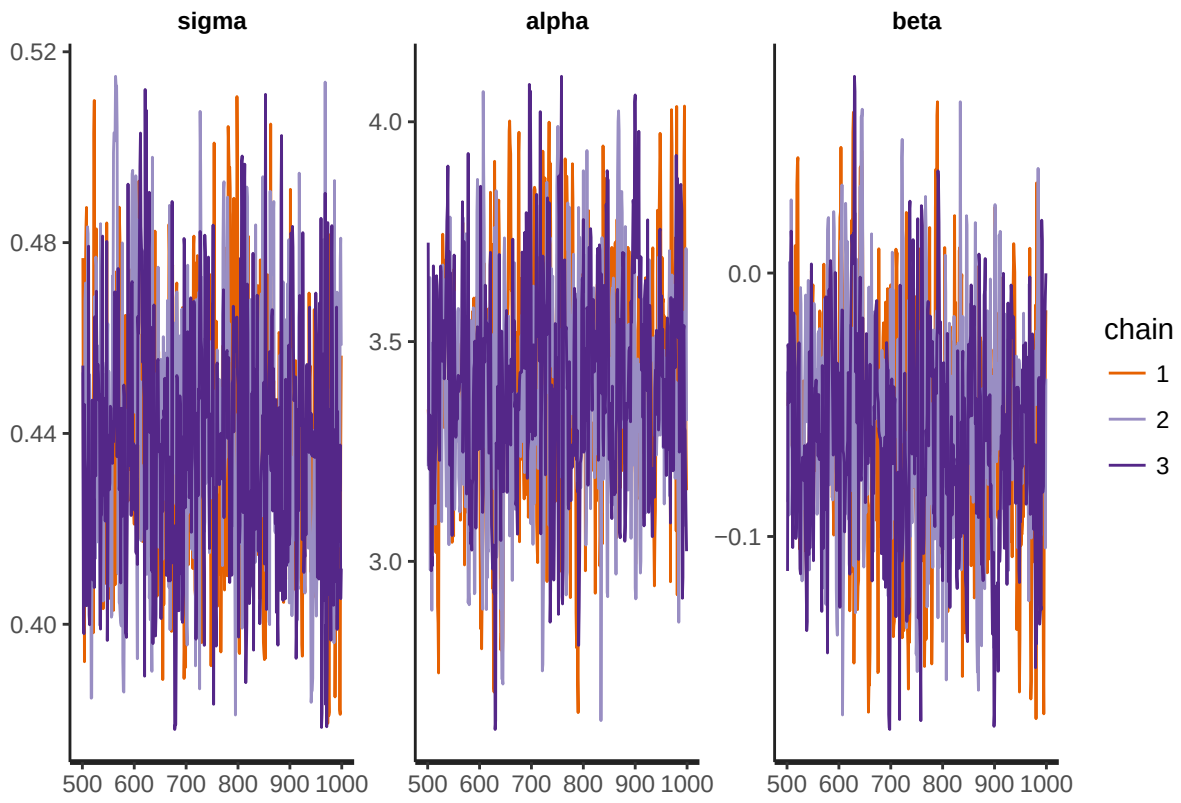
Initialize paramter values and send everything to stan.

```
#Initialize values
#convergence can be improved by setting reasonable starting values
#i.e, range of observations from 1-20, don't intialize mean at 100000
#Use different starting values for each chain
initslst <- lapply(1:nchains,function(i) {
  list(
    alpha = rnorm(1,0,1),
    beta = rnorm(1,0,1),
    sigma=runif(1,1,10)
  )
})

#send everything to Stan
fit2 <- stan(file = 'Ex6_est_lr.stan',
             data = dataList ,
             init = initslst,
             chains = nchains,
             iter = 1000 ,
             warmup = 500 ,
             thin = 1 )
```

View traceplots and pararmeter estimates

```
#View traceplots
traceplot(fit2)
```



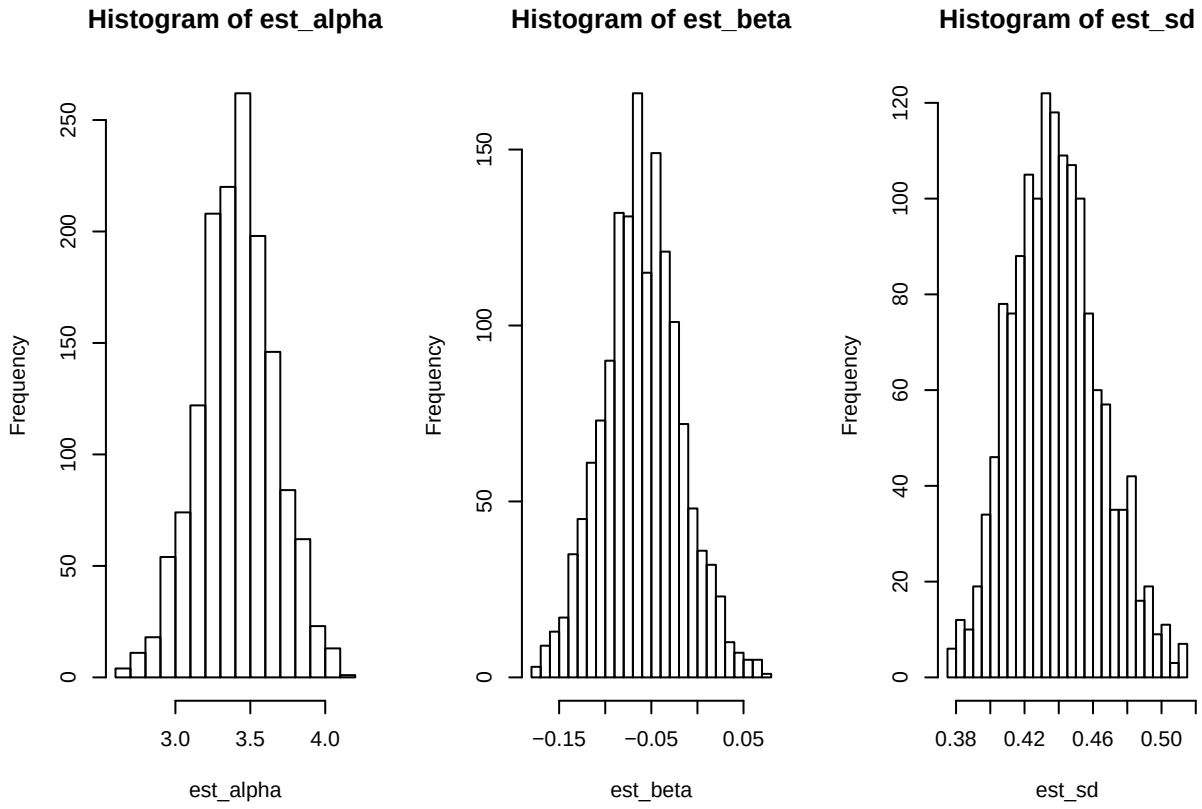

```
#view results
fit2
```

```
## Inference for Stan model: Ex6_est_lr.
## 3 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=1500.
##
##        mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## sigma  0.44    0.00 0.03  0.39  0.42  0.44  0.45  0.49   553    1
## alpha  3.41    0.01 0.25  2.92  3.25  3.42  3.57  3.90   446    1
## beta  -0.06    0.00 0.04 -0.14 -0.09 -0.06 -0.03  0.02   441    1
## lp__  48.75    0.05 1.21 45.77 48.12 49.09 49.66 50.16   514    1
##
## Samples were drawn using NUTS(diag_e) at Fri Jun 08 08:25:59 2018.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Extract paramters and plot histograms

```
#extract results
est_alpha=rstan::extract(fit2,"alpha")$alpha
est_beta=rstan::extract(fit2,"beta")$beta
est_sd=rstan::extract(fit2,"sigma")$sigma
```

```r
#plot results
par(mfrow=c(1,3))
hist(est_alpha,breaks=20);
hist(est_beta,breaks=20);
hist(est_sd,breaks=20);
```




```r
par(mfrow=c(1,1))
```