Stan: Mean and Standard Deviation of Univariate Normal

Carolyn J. Anderson

Edpsy 590BAY

This is a continuation of the Anorexia data example and to make this set of notes self contained, we’ll repeat the data set up steps

### Anorexia: estimate the mean  
library(rstan)

## Loading required package: StanHeaders

## Loading required package: ggplot2

## rstan (Version 2.19.2, 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)

## For improved execution time, we recommend calling  
## Sys.setenv(LOCAL\_CPPFLAGS = '-march=native')  
## although this causes Stan to throw an error on a few processors.

stan\_version()

## [1] "2.19.1"

setwd("D:\\Dropbox\\edps 590BAY\\Lectures\\9 Ham\_Stan\_brms")  
anorex <- read.table("anorexia\_data.txt",header=TRUE)  
# create change   
anorex$change <- anorex$weight2 - anorex$weight1  
  
#############################################  
# Sample statistics  
#############################################  
n=nrow(anorex)  
ybar = mean(anorex$change)  
s2 = var(anorex$change)  
stderr <- sqrt(s2/n)  
sample.stat <- c(n,ybar,s2,stderr)  
names(sample.stat) <- c("n","ybar","s2","stderr")  
sample.stat

## n ybar s2 stderr   
## 72.000000 2.763889 63.737833 0.940876

We now add in sigma in the paramter block and model block

#############################################  
# stan model definition  
#############################################  
stanmodel2 <- "  
data {  
 int<lower=0> N;  
 real y[N];  
}   
parameters {  
 real mu;  
 real<lower=0> sigma;  
}   
model {  
 target += cauchy\_lpdf(sigma | 0,1);  
 target += normal\_lpdf(mu | 0, 10);  
 target += normal\_lpdf(y | mu, sigma);  
}   
"

We also need out data list

########################################  
# Make data list  
########################################  
y <- anorex$change  
N <- nrow(anorex)   
dat <- list(N = N, y = y)

And again to improve exectution

# To improve execution  
options(mc.cores = parallel::detectCores()) #  
rstan\_options(auto\_write = TRUE)  
Sys.setenv(LOCAL\_CPPFLAGS = '-march=native')

Now for the part that takes a bit of time

fit.2 <- stan(  
 model\_code = stanmodel2,   
 model\_name = "example 2",  
 data = dat,   
 iter = 2000,   
 chains = 4,  
 cores = 4,   
 warmup = floor(2000/2),  
 verbose = FALSE)

At this point (barring any unforseen errors), we can look at results and some graphics

########################################  
# Output and various graphics  
########################################  
print(fit.2)  
stan\_plot(fit.2)  
stan\_trace(fit.2)  
stan\_dens(fit.2)  
stan\_hist(fit.2)  
stan\_ac(fit.2)  
stan\_scat(fit.2,pars=c("mu","sigma"))

Next step, try some linear regression with Nels data.