Lab 8

*Jun Rao*

*10/21/2019*

#--------------------------------------  
#--------------------------------------  
# Import/Load the rstan library:  
library(tidyverse)  
library(rstan)  
library(loo)  
rstan\_options(auto\_write = TRUE)  
options(mc.cores = parallel::detectCores())  
#--------------------------------------  
#--------------------------------------

# Random effects & Simpson’s paradox

*In this assignment, we will learn an important lesson about why accounting for group-level differences is essential in your analyses. I have provided you with a data set () that has one outcome variable, one input variable, and one group-level identifier. Note that the input variable has been centered and scaled for you.*

## Your tasks (35 points)

*1.Import your data into R.*

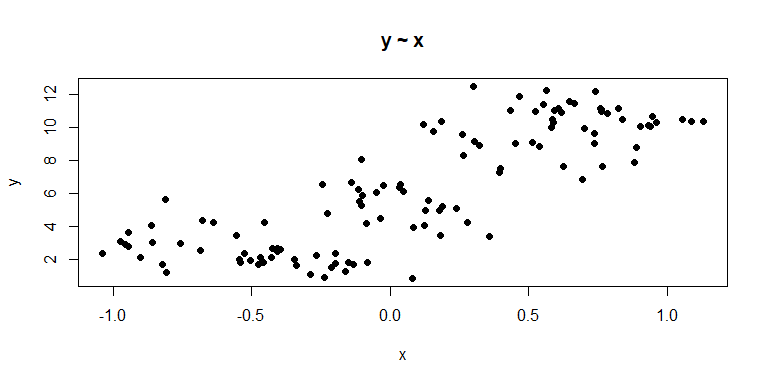
# Read in the .CSV file:  
df = read\_csv("lab8\_data.csv")  
  
#observations  
n\_sample = nrow(df)  
  
#number of groups  
n\_group = length(unique(df$group\_idx))  
  
#y  
y = df$y\_obs  
  
#x  
x = df$x\_scaled  
  
#group\_idx  
group\_idx = df$group\_idx  
  
head(df)

## # A tibble: 6 x 3  
## y\_obs x\_scaled group\_idx  
## <dbl> <dbl> <dbl>  
## 1 1.78 -0.540 1  
## 2 2.09 -0.901 1  
## 3 2.33 -1.04 1  
## 4 1.21 -0.807 1  
## 5 2.88 -0.957 1  
## 6 2.96 -0.756 1

*2.Create two scatterplots (5 points):*

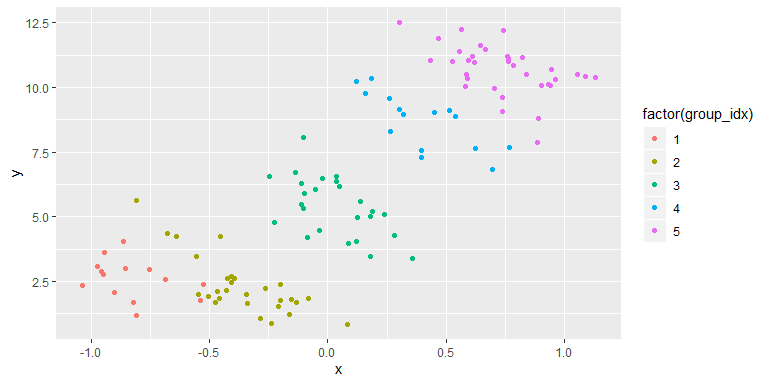
*(a) A scatterplot with distinction between groups*

plot(x, y, main="y ~ x",xlab="x", ylab="y", pch=19)



*(b) A scatterplot with data from specific groups that are clearly distinct*

ggplot(df, aes(x , y , color = factor(group\_idx))) +  
 geom\_point(shape = 19)



*3.Fit a linear regression with complete pooling (i.e., no distinction between groups). Write a brief interpretation about the linear relationship between the input and the outcome variable. In other words, interpret the slope parameter. (10 points)*

stan\_data\_1 = list(n\_sample = n\_sample,  
 x\_vec = x,  
 y\_vec = y)  
  
params\_monitor = c("beta", "alpha", "sigma\_resid", "log\_lik")  
  
test\_fit\_1 = stan(file = "LinReg.stan",  
 data = stan\_data\_1,  
 pars = params\_monitor,  
 chains = 1, # How many chains to run  
 iter = 10, # How many iterations per chain  
 algorithm="NUTS")

##   
## SAMPLING FOR MODEL 'LinReg' NOW (CHAIN 1).  
## Chain 1:   
## Chain 1: Gradient evaluation took 0 seconds  
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 1: Adjust your expectations accordingly!  
## Chain 1:   
## Chain 1:   
## Chain 1: WARNING: No variance estimation is  
## Chain 1: performed for num\_warmup < 20  
## Chain 1:   
## Chain 1: Iteration: 1 / 10 [ 10%] (Warmup)  
## Chain 1: Iteration: 2 / 10 [ 20%] (Warmup)  
## Chain 1: Iteration: 3 / 10 [ 30%] (Warmup)  
## Chain 1: Iteration: 4 / 10 [ 40%] (Warmup)  
## Chain 1: Iteration: 5 / 10 [ 50%] (Warmup)  
## Chain 1: Iteration: 6 / 10 [ 60%] (Sampling)  
## Chain 1: Iteration: 7 / 10 [ 70%] (Sampling)  
## Chain 1: Iteration: 8 / 10 [ 80%] (Sampling)  
## Chain 1: Iteration: 9 / 10 [ 90%] (Sampling)  
## Chain 1: Iteration: 10 / 10 [100%] (Sampling)  
## Chain 1:   
## Chain 1: Elapsed Time: 0.001 seconds (Warm-up)  
## Chain 1: 0 seconds (Sampling)  
## Chain 1: 0.001 seconds (Total)  
## Chain 1:

## Now we will run our full model:  
# How many samples do we want of each parameter, from each chain?  
n\_mc\_samples = 1000  
# How much burn-in?  
n\_burn = 500  
# How much thinning? (take the ith value of the chain)  
n\_thin = 3  
# Total iterations needed:  
n\_iter\_total = (n\_mc\_samples \* n\_thin) + n\_burn  
model\_fit\_1 =  
 stan(fit = test\_fit\_1, # So it knows we're already compiled  
 file = "LinReg.stan",  
 data = stan\_data\_1,  
 pars = params\_monitor,  
 chains = 3,  
 warmup = n\_burn,  
 thin = n\_thin,  
 iter = n\_iter\_total,  
 algorithm="NUTS")  
model\_out\_1 = rstan::extract(model\_fit\_1)  
str(model\_out\_1)

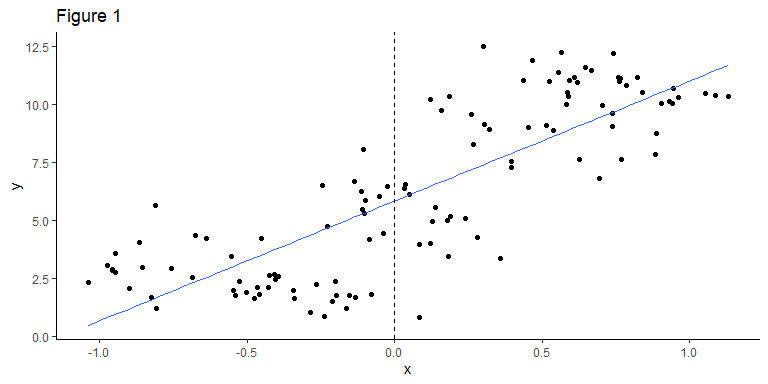
## List of 5  
## $ beta : num [1:3000(1d)] 5.16 4.36 5.25 5.28 5.21 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ alpha : num [1:3000(1d)] 6.04 5.79 5.87 5.62 5.39 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ sigma\_resid: num [1:3000(1d)] 2.34 2.33 2.12 1.92 2.09 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ log\_lik : num [1:3000, 1:117] -1.97 -2.02 -1.85 -1.7 -1.73 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ iterations: NULL  
## .. ..$ : NULL  
## $ lp\_\_ : num [1:3000(1d)] -265 -267 -263 -265 -266 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL

#the 95% credible intervals  
beta\_CI = c(summary(model\_fit\_1)$summary[1:1, "2.5%"],summary(model\_fit\_1)$summary[1:1, "97.5%"])  
beta\_CI

## [1] 4.498770 5.827031

***The 95% credible intervals of beta which is the slope is between 4.527 and 5.857. It means 0 is not included in the 95% credible intervals. Hence, there is a linear relatiionship between the input and the outcome variable. More details can see next graphic (Figuure1).***

#draw the graphics for each group  
ggplot(df, aes(x, y)) +  
 ggtitle("Figure 1")+  
 geom\_point(shape = 19) +  
 geom\_smooth(method = "lm", se = FALSE, size = 0.2) +  
 scale\_color\_brewer(palette = "Set1") +  
 geom\_vline(xintercept = 0, linetype = 2) +  
 theme\_classic()



*4.Fit a linear regression with partial pooling. Specifically, allow a random intercept per group. Again, write a brief interpretation about this linear regression. How has your conclusion changed compared to the outcome in Task 3? (10 points)*

stan\_data\_2 = list(n\_sample = n\_sample,  
 n\_group = n\_group,  
 y\_vec = y,  
 x\_vec = x,  
 group\_idx = group\_idx)  
  
params\_monitor = c("beta", "alpha\_mean", "eta\_alpha", "alpha\_sigma", "sigma\_resid", "log\_lik")  
  
test\_fit\_2 = stan( file = "Intercepts\_LinReg.stan",  
 data = stan\_data\_2,  
 pars = params\_monitor,  
 chains = 1,  
 iter = 10,  
 algorithm="NUTS")

##   
## SAMPLING FOR MODEL 'Intercepts\_LinReg' NOW (CHAIN 1).  
## Chain 1:   
## Chain 1: Gradient evaluation took 0 seconds  
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 1: Adjust your expectations accordingly!  
## Chain 1:   
## Chain 1:   
## Chain 1: WARNING: No variance estimation is  
## Chain 1: performed for num\_warmup < 20  
## Chain 1:   
## Chain 1: Iteration: 1 / 10 [ 10%] (Warmup)  
## Chain 1: Iteration: 2 / 10 [ 20%] (Warmup)  
## Chain 1: Iteration: 3 / 10 [ 30%] (Warmup)  
## Chain 1: Iteration: 4 / 10 [ 40%] (Warmup)  
## Chain 1: Iteration: 5 / 10 [ 50%] (Warmup)  
## Chain 1: Iteration: 6 / 10 [ 60%] (Sampling)  
## Chain 1: Iteration: 7 / 10 [ 70%] (Sampling)  
## Chain 1: Iteration: 8 / 10 [ 80%] (Sampling)  
## Chain 1: Iteration: 9 / 10 [ 90%] (Sampling)  
## Chain 1: Iteration: 10 / 10 [100%] (Sampling)  
## Chain 1:   
## Chain 1: Elapsed Time: 0 seconds (Warm-up)  
## Chain 1: 0.001 seconds (Sampling)  
## Chain 1: 0.001 seconds (Total)  
## Chain 1:

## Now we will run our full model:  
# How many samples do we want of each parameter, from each chain?  
n\_mc\_samples = 1000  
# How much burn-in?  
n\_burn = 500  
# How much thinning? (take the ith value of the chain)  
n\_thin = 3  
# Total iterations needed:  
n\_iter\_total = (n\_mc\_samples \* n\_thin) + n\_burn  
model\_fit\_2 =  
stan(fit = test\_fit\_2, # So it knows we're already compiled  
 file = "Intercepts\_LinReg.stan",  
 data = stan\_data\_2,  
 pars = params\_monitor,  
 chains = 3,  
 warmup = n\_burn,  
 thin = n\_thin,  
 iter = n\_iter\_total,  
 algorithm="NUTS")  
model\_out\_2 = rstan::extract(model\_fit\_2)  
str(model\_out\_2)

## List of 7  
## $ beta : num [1:3000(1d)] -3.29 -3.05 -2.84 -3.48 -3.44 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ alpha\_mean : num [1:3000(1d)] 10.39 5.76 5.56 6.15 9.47 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ eta\_alpha : num [1:3000, 1:5] -10.71 -5.9 -5.52 -6.45 -9.63 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ iterations: NULL  
## .. ..$ : NULL  
## $ alpha\_sigma: num [1:3000(1d)] 7.08 5.59 5.37 4.63 5.5 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ sigma\_resid: num [1:3000(1d)] 0.83 0.843 0.921 0.798 0.847 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL  
## $ log\_lik : num [1:3000, 1:117] -0.812 -0.803 -0.861 -0.73 -0.758 ...  
## ..- attr(\*, "dimnames")=List of 2  
## .. ..$ iterations: NULL  
## .. ..$ : NULL  
## $ lp\_\_ : num [1:3000(1d)] -48.9 -46.4 -48.6 -45.7 -48.6 ...  
## ..- attr(\*, "dimnames")=List of 1  
## .. ..$ iterations: NULL

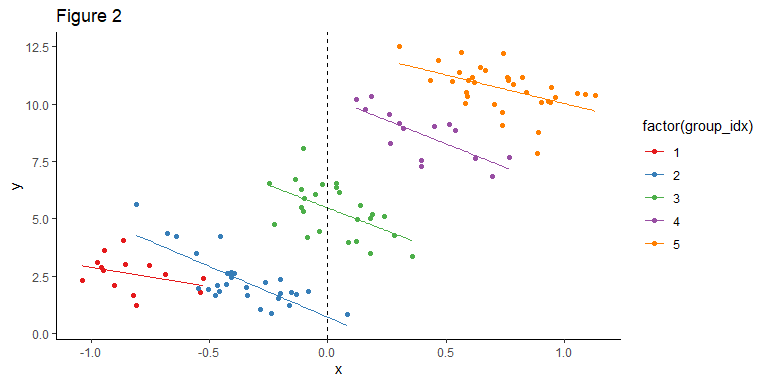
#summary(model\_fit\_2)$summary  
#first we visualize tabularly, the 95% CI for estimated parameters and compare these the the true values  
model\_sum = summary(model\_fit\_2)$summary  
model\_sum[c(1:9),c(4,6,8)]

## 2.5% 50% 97.5%  
## beta -4.2255768 -3.3612860 -2.4824869  
## alpha\_mean -0.8538509 5.9136786 12.4329932  
## eta\_alpha[1] -12.5813342 -6.1110159 0.6856631  
## eta\_alpha[2] -11.3661183 -4.8081377 1.8366116  
## eta\_alpha[3] -7.0311699 -0.4578394 6.3970286  
## eta\_alpha[4] -2.3131249 4.0601165 10.8479128  
## eta\_alpha[5] 0.8007318 7.1830648 13.9409152  
## alpha\_sigma 3.2751447 5.7082367 13.1918307  
## sigma\_resid 0.7391743 0.8394490 0.9660167

# we can see that few parameters were adequately recoverd. This included alpha\_mean and all of the eta\_alphas (9 of 12 params)

***The 95% credible intervals of beta which is the slope is between -4.171 and -2.473. For each group, it has differnent interception. In task 3, we can find that the slope for all the data is a positive number, however, when we analysis each single group, the slope is a negative number. From Figure 2, we can find it intuitively.***

#draw the graphics for each group  
ggplot(df, aes(x, y, color = factor(group\_idx))) +  
 ggtitle("Figure 2")+  
 geom\_point(shape = 19) +  
 geom\_smooth(method = "lm", se = FALSE, size = 0.2) +  
 scale\_color\_brewer(palette = "Set1") +  
 geom\_vline(xintercept = 0, linetype = 2) +  
 theme\_classic()



*5.Compare your two models using the LOO-IC. Which model provides a more parsimonious explanation of the data? Interpret what this means. (5 points)*

# Calculate the LOO and WAIC for a single model:  
# Full model  
log\_lik\_1 = extract\_log\_lik(model\_fit\_1)   
loo\_1 = loo(log\_lik\_1)  
loo\_1$estimates

## Estimate SE  
## elpd\_loo -254.306013 6.5730454  
## p\_loo 2.490803 0.3251701  
## looic 508.612026 13.1460909

# reduced, intercept RE model  
log\_lik\_2 = extract\_log\_lik(model\_fit\_2)   
loo\_2 = loo(log\_lik\_2)  
loo\_2$estimates

## Estimate SE  
## elpd\_loo -148.902259 7.4904658  
## p\_loo 6.613833 0.9253659  
## looic 297.804519 14.9809316

# model comparison of loo  
loo::compare(loo\_1, loo\_2)

## elpd\_diff se   
## 105.4 9.8

***Based on the p\_loo values for both models, the estimate 2.46 is lower than the intercept RE model. Additionally, the elpd\_loo is lower but not significant because SE ranges overlap for both model estimate of elpd\_loo. Therefore, we can conclude that the reduced model is more parsimonious because p\_loo is still greater than the number of parameters.***

*6.This assignment has shown an example of Simpson’s Paradox. Look up this term and write a brief interpretation of how it applies to this assignment. What have you learned? (5 points)*

***Such as we described in the previous questions, we find that if we analysis all the data, the slope in the Figure 1 is postive, however, when we analysis by the group, the slope for each group is negative see Figure 2.***