Assignment7

Yung-Tang Chou

2019年6月5日

Here is the rmarkdown file for the Multivariate Analysis Assignment 7.

The data is regarding the disastrousness of Hurricanes, modeling with the feminity of names, storm damage and the minimun pressure. Before doing the modeling first let’s import needed libraries.

library(rstan)

## Loading required package: ggplot2

## Loading required package: StanHeaders

## rstan (Version 2.18.2, 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.

library(rethinking)

## Loading required package: parallel

## rethinking (Version 1.88)

library(loo)

## This is loo version 2.1.0.  
## \*\*NOTE: As of version 2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use the 'cores' argument or set options(mc.cores = NUM\_CORES) for an entire session. Visit mc-stan.org/loo/news for details on other changes.

## \*\*NOTE for Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see https://github.com/stan-dev/loo/issues/94).

##   
## Attaching package: 'loo'

## The following object is masked from 'package:rethinking':  
##   
## compare

## The following object is masked from 'package:rstan':  
##   
## loo

library(skimr)

##   
## Attaching package: 'skimr'

## The following object is masked from 'package:stats':  
##   
## filter

Let’s import the data. To improve the effectiveness of modeling, let’s do nomalization for dependent variables, femininity, damage\_norm, and min\_pressure. Code is as below.

data("Hurricanes")  
Hurricanes$damage\_norm\_std <-   
 (Hurricanes$damage\_norm - mean(Hurricanes$damage\_norm)) / sd(Hurricanes$damage\_norm)  
Hurricanes$femininity\_std <-   
 (Hurricanes$femininity - mean(Hurricanes$femininity)) / sd(Hurricanes$femininity)  
Hurricanes$min\_pressure\_std <-   
 (Hurricanes$min\_pressure - mean(Hurricanes$min\_pressure)) / sd(Hurricanes$min\_pressure)

A basic summary statistics is shown below.

## Skim summary statistics  
## n obs: 92   
## n variables: 11   
##   
## -- Variable type:factor ----------------------------------------------------------------------------  
## variable missing complete n n\_unique top\_counts  
## name 0 92 92 83 Bob: 3, Bon: 2, Cha: 2, Cin: 2  
## ordered  
## FALSE  
##   
## -- Variable type:integer ---------------------------------------------------------------------------  
## variable missing complete n mean sd p0 p25 p50  
## category 0 92 92 2.09 1.06 1 1 2  
## damage\_norm 0 92 92 7269.78 12934.09 1 245 1650  
## deaths 0 92 92 20.65 40.9 0 2 5  
## female 0 92 92 0.67 0.47 0 0 1  
## min\_pressure 0 92 92 964.91 19.07 909 950 964  
## year 0 92 92 1982.09 18.77 1950 1964.75 1985  
## p75 p100 hist  
## 3 5 ▇▅▁▆▁▁▁▁  
## 8162.5 75000 ▇▁▁▁▁▁▁▁  
## 20.25 256 ▇▁▁▁▁▁▁▁  
## 1 1 ▃▁▁▁▁▁▁▇  
## 982.25 1003 ▁▁▂▇▆▅▇▂  
## 1999 2012 ▆▆▅▃▆▅▇▆  
##   
## -- Variable type:numeric ---------------------------------------------------------------------------  
## variable missing complete n mean sd p0 p25 p50  
## damage\_norm\_std 0 92 92 1.2e-17 1 -0.56 -0.54 -0.43   
## femininity 0 92 92 6.78 3.23 1.06 2.67 8.5   
## femininity\_std 0 92 92 1.3e-16 1 -1.77 -1.27 0.53   
## min\_pressure\_std 0 92 92 -2.6e-16 1 -2.93 -0.78 -0.048  
## p75 p100 hist  
## 0.069 5.24 ▇▁▁▁▁▁▁▁  
## 9.39 10.44 ▅▃▁▁▁▂▆▇  
## 0.81 1.14 ▅▃▁▁▁▂▆▇  
## 0.91 2 ▁▁▂▇▆▅▇▂

< Question 1 > For question 1, we want to model do a Poisson model using femininity as a predictor.

The model is shown below.

model.01.code <- "  
data{  
 int N;  
int d[92];  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
}  
model{  
vector[92] lambda;  
bf ~ normal( 0 , 3 );  
a ~ normal( 0 , 3 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ poisson( lambda );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = poisson\_lpmf( d[i] | lambda[i] );  
}  
"

Note that within the generated quantities block, the log\_lik is also calculated.

Now we are going to run the stan model. Before doing so, to prevent stan model from crashing, we form a specific dataset for the stan model. Code is shown below.

d.01 <- list(  
 N = NROW(Hurricanes),  
 fem = Hurricanes$femininity\_std,  
 d = Hurricanes$deaths  
)

For the stan model, we will use “d.01” generated above as the input data, and use four markov chains to do the modeling. In this cases, we do four chains with four cores.

The model code is shown below.

model.01 <- stan(model\_code = model.01.code, data = d.01,  
 cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

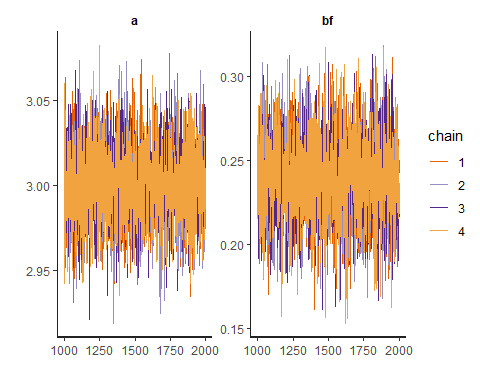
print(model.01, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

## Inference for Stan model: 9f95669c1b0db8e352204dfbbb29337c.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 3.00 0.00 0.02 2.97 3.00 3.03 2716 1  
## bf 0.24 0.00 0.03 0.21 0.24 0.27 2693 1  
## lp\_\_ 3898.57 0.02 0.98 3897.30 3898.87 3899.46 1783 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:13:44 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 2000, which implys a pretty solid model. From the description we can know that the femininity of the Hurricanes’ name is positive related to the death tolls, with the intercept being around 3.00.

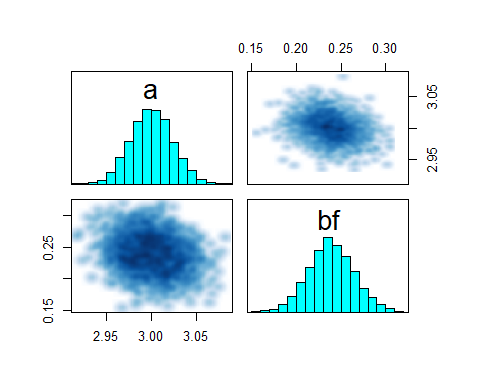
It would be a good idea to check whether the stan model is decent graphically. In order to do that we use traceplot to see if all the params lie in a specific range. The code is as below.

traceplot(model.01, pars = c('a','bf'))

 From the traceplot above we can easily tell that the model converges.

Besides, let’s see the distribution of the model params using pairplot. The code is as below.

pairs(model.01, pars = c('a','bf'))

 From the plots above, we can tell the following things. 1. The stan model converges properly. The traceplot shows that all the values lie in a specific range, which is the desired outcome. 2. The pairplot shows that the distribution of param bf is a little bit right-skewed.

Now it’s time to evaluate the model. In this question we use try to use waic to examine the model’s effectiveness. Code is shown below.

log\_lik\_01 <- extract\_log\_lik(model.01, merge\_chains = FALSE)  
waic\_01 <- waic(log\_lik\_01)

## Warning: 30 (32.6%) p\_waic estimates greater than 0.4. We recommend trying  
## loo instead.

We will do the comparison with another model later.

# ——-

Let’s conduct another model using merely intercept. The code is shown below.

model.02.code <- "  
data{  
 int N;  
int d[92];  
}  
parameters{  
real a;  
}  
model{  
real lambda;  
a ~ normal( 0 , 3 );  
lambda = a;  
lambda = exp(lambda);  
d ~ poisson( lambda );  
}  
generated quantities{  
vector[92] log\_lik;  
real lambda;  
lambda = a;  
lambda = exp(lambda);  
for ( i in 1:92 ) log\_lik[i] = poisson\_lpmf( d[i] | lambda );  
}  
"

Again to avoid the stan model from crashing, we will form a specific dataframe serving as the input dataframe. Code is shown below.

d.02 <- list(  
 N = NROW(Hurricanes),  
 d = Hurricanes$deaths)

For the stan model, we will use “d.02” generated above as the input data, and use four markov chains to do the modeling. In this cases, we do four chains with four cores.

The model code is shown below.

model.02 <- stan(model\_code = model.02.code, data = d.02,  
 cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

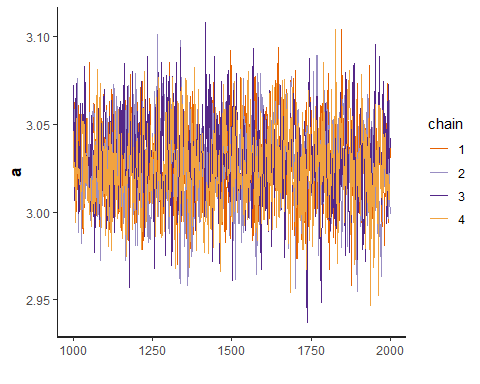
print(model.02, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

## Inference for Stan model: de2313c2b6fd86bb331dd720cfaa0939.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 3.03 0.00 0.02 3.00 3.03 3.06 1639 1  
## lp\_\_ 3851.82 0.02 0.73 3850.96 3852.09 3852.34 1872 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:14:46 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 1000, which implys a pretty solid model. From the description we can know that the with the model using intercept as the only predictor, the intercept wil be around 3.03.

Similarly let’s take a look at the traceplot to evaluate the decentness of the model.

traceplot(model.02, pars = c('a'))

 From the traceplot above we can easily tell that the model converges. With only intercept, we cannot use pairplot to evaluate the effectiveness.

From the plots above, we can tell that the stan model converges properly. The traceplot shows that all the values lie in a specific range, which is the desired outcome..

Again we use waic to evaluate model.02 and then compare model.01 and model.02.

log\_lik\_02 <- extract\_log\_lik(model.02, merge\_chains = FALSE)  
waic\_02 <- waic(log\_lik\_02)

## Warning: 13 (14.1%) p\_waic estimates greater than 0.4. We recommend trying  
## loo instead.

Let’s compare the two models by using the compare function in the rethinking package. Code is shown below.

rethinking::compare(model.01, model.02)

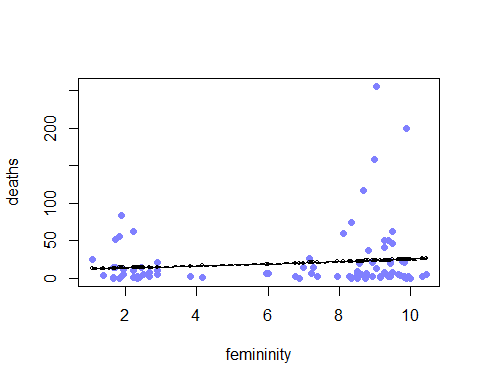
## WAIC pWAIC dWAIC weight SE dSE  
## model.01 4410.063 130.47433 0.0000 1.000000e+00 996.8524 NA  
## model.02 4456.173 85.16015 46.1096 9.714639e-11 1078.8294 145.9707

From the output we can see the model using femininity as predictor is performing way better than that using only intercept. Let’s fit the model with the data to have a clearer view of the model effectiveness. First is to extract samples from the stan model and calculate the mean and PI. Code is shown below.

post <- extract.samples(model.01)  
post.mean <- apply(post$lambda, 2, mean)  
post.PI <- apply(post$lambda, 2, PI, prob=.89)

Then we do the plotting accordingly. Code is shown below.

plot(y=Hurricanes$deaths, x=Hurricanes$femininity, col=rangi2, ylab="deaths", xlab="femininity", pch=16)  
points(y=post.mean, x = Hurricanes$femininity, col = 'black', cex = 0.5)  
segments(x0=Hurricanes$femininity, x1= Hurricanes$femininity, y0=post.PI[1,], y1=post.PI[2,])  
lines(y= post.mean[order(Hurricanes$femininity)], x=sort(Hurricanes$femininity))  
lines( post.PI[1,order(Hurricanes$femininity)], x=sort(Hurricanes$femininity), lty=2 )  
lines( post.PI[2,order(Hurricanes$femininity)], x=sort(Hurricanes$femininity), lty=2 )

 From the plotting you can see there are some outliers on the top right corner, which do not fit the model well. The relationship isn’t very strong, but still slightly positive. The outliers in the meantime shows the same trend.

# ——–

< Question 2 > As the problem description says, counts are nearly always over-dispersed relative to Poisson. Therefore, we are going to fit a gamma-Poisson model to predict deaths. In this model we again use femininity as the predictor.

The stan model for the gamma-Poisson model is shown below.

model.03.code <- "  
data{  
 int N;  
int d[92];  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
real phi;  
}  
model{  
vector[92] lambda;  
phi ~ cauchy( 0 , 3 );  
bf ~ normal( 0 , 5 );  
a ~ normal( 0 , 10 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ neg\_binomial\_2( lambda , phi );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = neg\_binomial\_2\_lpmf( d[i] | lambda[i] , phi );  
}  
"

Note that within the generated quantities block, the log\_lik is also calculated.

To prevent stan model from crashing, we form a specific dataset for the stan model. Code is shown below.

d.03 <- list(  
 N = NROW(Hurricanes),  
 fem = Hurricanes$femininity\_std,  
 d = Hurricanes$deaths  
)

For the stan model, we will use “d.03” generated above as the input data, and use four markov chains to do the modeling. In this cases, we do four chains with four cores.

The model code is shown below.

model.03 <- stan(model\_code = model.03.code, cores = 4, chains = 4, data = d.03)

The descriptions of the model is shown below, using print statement.

print(model.03, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.05, .5, .95))

## Inference for Stan model: bab1658e45901785963609451893d09b.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 5% 50% 95% n\_eff Rhat  
## a 3.03 0.00 0.16 2.78 3.03 3.30 3786 1  
## bf 0.22 0.00 0.16 -0.04 0.22 0.48 3011 1  
## phi 0.45 0.00 0.06 0.36 0.45 0.56 3089 1  
## lp\_\_ 5678.41 0.03 1.21 5676.10 5678.73 5679.73 2183 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:15:52 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 3000, which implys an extremely solid model. We can see that the confidence interval for bf overlaps zero, which implys that there is no significant possitive association between femininity and death.

# ——-

< Question 3 > In this question we add two potential candidate predictors - minimum pressure (min\_pressure) and nomalized estimate of damage (damage\_norm). In this question we want to see the interaction effects between the two and femininity.

Before formulating the model, again let’s create a dataframe solely for the stan model to prevent it from crashing. Code is shown below.

d.04 <- list(  
 N = NROW(Hurricanes),  
 fem = Hurricanes$femininity\_std,  
 d = Hurricanes$deaths,  
 mp = Hurricanes$min\_pressure\_std,  
 dm = Hurricanes$damage\_norm\_std  
)

For the first model, let’s assume that there’s an interaction effect between min\_pressure and femininity. The model is written as below.

model.04.1.code <-"  
data{  
 vector[92] dm;  
int N;  
int d[92];  
vector[92] mp;  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
real bmp;  
real bfmp;  
real phi;  
}  
model{  
vector[92] lambda;  
phi ~ cauchy( 0 , 1 );  
bfmp ~ normal( 0 , 3 );  
bf ~ normal( 0 , 3 );  
bmp ~ normal( 0 , 3 );  
a ~ normal( 0 , 3 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bmp \* mp[i] + bfmp \* fem[i] \* mp[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ neg\_binomial\_2( lambda , phi );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bmp \* mp[i] + bfmp \* fem[i] \* mp[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = neg\_binomial\_2\_lpmf( d[i] | lambda[i] , phi );  
}  
"

The model code is shown below.

model.04.1 <- stan(model\_code = model.04.1.code, data = d.04, cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

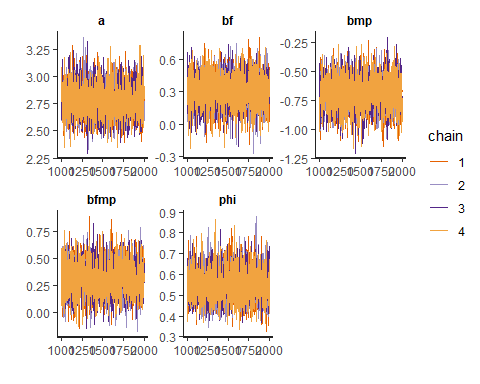
print(model.04.1, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

## Inference for Stan model: 061a279b45af5a68e397a52f6d66e0c0.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.79 0.00 0.15 2.60 2.78 2.97 4412 1  
## bf 0.30 0.00 0.15 0.12 0.30 0.48 4585 1  
## bmp -0.69 0.00 0.14 -0.87 -0.68 -0.51 3983 1  
## bfmp 0.31 0.00 0.15 0.12 0.31 0.50 4588 1  
## phi 0.55 0.00 0.08 0.46 0.55 0.66 4228 1  
## lp\_\_ 5688.49 0.04 1.56 5686.44 5688.80 5690.16 1876 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:16:59 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 4000, which implys an extremely solid model. We can see that femininity again is positively related to deaths tolls, while min\_pressure is strongly negatively related to deaths. To add on, the interaction effects is positively related to deaths, all above with the intercept of 2.78, not too far away from the model without these two new predictors.

Before moving on to the next model, let’s take a look at the traceplot to evaluate the decentness of the model.

traceplot(model.04.1, pars = c('a','bf','bmp','bfmp','phi'))

 From the traceplot above we can easily tell that the model converges.

Let’s build the second model, assuming there’s an interaction effect between damage\_norm and femininity. The model code is written as below.

model.04.2.code <-"  
data{  
 vector[92] mp;  
int N;  
int d[92];  
vector[92] dm;  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
real bdm;  
real bfdm;  
real phi;  
}  
model{  
vector[92] lambda;  
phi ~ cauchy( 0 , 1 );  
bfdm ~ normal( 0 , 3 );  
a ~ normal( 0 , 3 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* dm[i] + bfdm \* fem[i] \* dm[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ neg\_binomial\_2( lambda , phi );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* dm[i] + bfdm \* fem[i] \* dm[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = neg\_binomial\_2\_lpmf( d[i] | lambda[i] , phi );  
}  
"

The model code is shown below.

model.04.2 <- stan(model\_code = model.04.2.code, data = d.04, cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

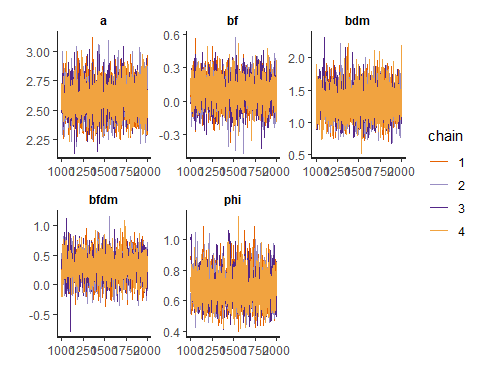
print(model.04.2, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

## Inference for Stan model: 90e2139c80953ae0aac274572512b36a.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.59 0.00 0.14 2.42 2.59 2.77 5951 1  
## bf 0.09 0.00 0.13 -0.08 0.10 0.25 4486 1  
## bdm 1.32 0.00 0.23 1.03 1.31 1.63 4157 1  
## bfdm 0.32 0.00 0.21 0.06 0.33 0.58 3513 1  
## phi 0.68 0.00 0.10 0.56 0.68 0.82 4008 1  
## lp\_\_ 5698.94 0.04 1.65 5696.82 5699.28 5700.67 1681 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:18:03 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 4000, which implys an extremely solid model. Within this model we can figure out these ideas: 1. Femininity is slightly positively related to death tolls. In this case is not that obvious compared to the formal ones. 2. damage\_norm is strongly positively related to death tolls, which is quite intuitive by the nature of the predictor. 3. The interaction effects again is positively related to the death toll.

Before moving on to the next model, let’s take a look at the traceplot to evaluate the decentness of the model.

traceplot(model.04.2, pars = c('a','bf','bdm','bfdm','phi'))

 From the traceplot above we can easily tell that the model converges.

Let’s move on to the third model, with two interaction effects all included. The model is described below.

model.04.3.code <-"  
data{  
 int N;  
int d[92];  
vector[92] mp;  
vector[92] dm;  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
real bdm;  
real bmp;  
real bfdm;  
real bfmp;  
real phi;  
}  
model{  
vector[92] lambda;  
phi ~ cauchy( 0 , 1 );  
bf ~ normal( 0 , 3 );  
bmp ~ normal( 0 , 3 );  
bdm ~ normal( 0 , 3 );  
bfmp ~ normal( 0 , 3 );  
bfdm ~ normal( 0 , 3 );  
a ~ normal( 0 , 3 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* dm[i] + bmp \* mp[i] + bfdm \* fem[i] \* dm[i] + bfmp \* fem[i] \* mp[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ neg\_binomial\_2( lambda , phi );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* dm[i] + bmp \* mp[i] + bfdm \* fem[i] \* dm[i] + bfmp \* fem[i] \* mp[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = neg\_binomial\_2\_lpmf( d[i] | lambda[i] , phi );  
}  
"

The model code is shown below.

model.04.3 <- stan(model\_code = model.04.3.code, data = d.04, cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

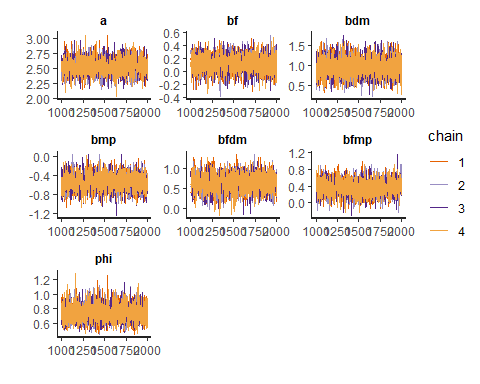
print(model.04.3, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

## Inference for Stan model: bda8047e6e2efbc6bf25830a17cd0db7.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.54 0.00 0.13 2.37 2.54 2.70 4121 1  
## bf 0.11 0.00 0.13 -0.05 0.12 0.28 4154 1  
## bdm 0.95 0.00 0.22 0.66 0.94 1.24 3124 1  
## bmp -0.54 0.00 0.17 -0.76 -0.54 -0.33 2540 1  
## bfdm 0.64 0.00 0.21 0.37 0.64 0.90 2775 1  
## bfmp 0.37 0.00 0.18 0.14 0.37 0.60 2458 1  
## phi 0.75 0.00 0.11 0.60 0.74 0.90 3378 1  
## lp\_\_ 5703.48 0.04 1.88 5700.96 5703.81 5705.58 1832 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:19:12 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 2000, which implys an solid model. Within this model we can figure out these ideas: 1. Femininity is slightly positively related to death tolls. In this case still it is not that obvious. 2. Damage\_norm is still strongly positively related to death tolls, which is quite intuitive by the nature of the predictor. However, this correlation is slightly lower than the previous model, which does not include the effect of min\_pressure. 3. Min\_pressure is negatively related to the death tolls, which is similar to the one using only min\_pressure as predictor. 4. Both the interaction effects shows that it is positively related to the death toll.

Before moving on to the next model, let’s take a look at the traceplot to evaluate the decentness of the model.

traceplot(model.04.3, pars = c('a','bf','bdm', 'bmp', 'bfdm','bfmp','phi'))

 From the traceplot above we can easily tell that the model converges.

Let’s compare those models. The code is shown below.

rethinking::compare(model.04.1, model.04.2, model.04.3)

## WAIC pWAIC dWAIC weight SE dSE  
## model.04.3 663.9335 8.777622 0.000000 9.723632e-01 35.25648 NA  
## model.04.2 671.0547 7.040856 7.121177 2.763660e-02 34.05445 8.129309  
## model.04.1 694.3168 8.326558 30.383342 2.455668e-07 37.95956 14.360682

From the output we can tell that the one with only damage\_norm as well as its interaction effect is the best model with the lowest WAIC.

To better understand the model, let’s make some counterfactual plots with two variables - min\_pressure and damage\_norm. To start off, let’s do some data preprocessing. The code is shown as below.

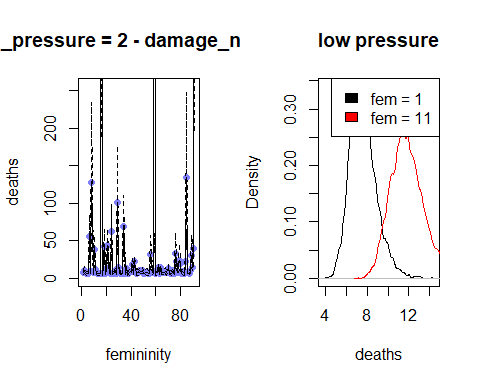
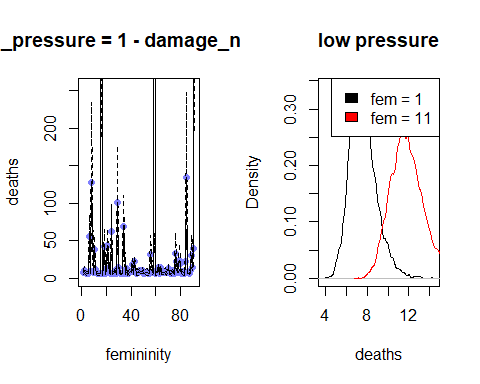
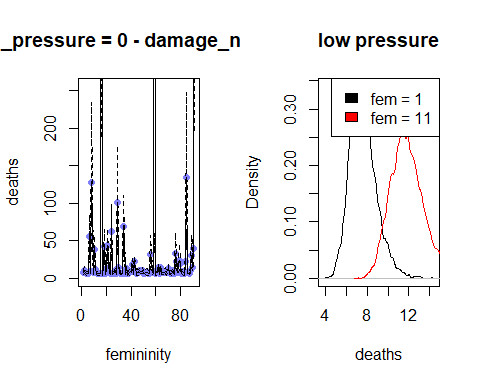
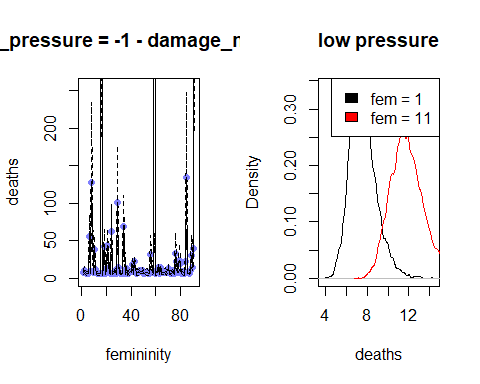
cp <- extract.samples(model.04.2)  
cp.mean <- colMeans(cp$lambda)  
cp.PI <- apply(cp$lambda, 2, PI, prob=.89)

Now is the code for counterfactural plots.

counterfactual\_plot <- function(model, mp=0, dm=0){  
   
 cp\_data=data.frame(min\_pressure=rep(mp,92), damage\_norm=rep(dm,92), femininity=c(1:92))  
   
 test <- extract.samples(model)  
 test.mean <- colMeans(test$lambda)  
 test.PI <- apply(test$lambda, 2, PI, prob=.89)  
   
 # plot the model predictions for `y` on the scale of actual deaths.  
 title <- paste0("min\_pressure = ",mp," - damage\_norm = ",dm)  
 plot(y = test.mean, x=cp\_data$femininity, col=rangi2, ylab="deaths", xlab="femininity", pch=16, ylim=c(0,max(Hurricanes$deaths)), main=title)  
   
 lines(y = test.mean, x=cp\_data$femininity)  
 lines( test.PI[1,], x=cp\_data$femininity, lty=2 )  
 lines( test.PI[2,], x=cp\_data$femininity, lty=2 )  
   
 dens(test$lambda[,1], main="low pressure", xlab="deaths")  
 dens(test$lambda[,11], add=T, col="red")  
 legend("topright", legend=c("fem = 1","fem = 11"), fill = c("black","red"))  
}

After all the preparation, now let’s do the plotting!

vargrid <- expand.grid(mp= -1:2,dm = 0)  
  
par(mfrow=c(1,2))  
  
for(i in 1:NROW(vargrid)){  
 counterfactual\_plot(model = model.04.2, mp = vargrid$mp[i], dm = vargrid$dm[i])  
}



< Question 4 > In this question we want to substitute damage\_norm by log(damage\_norm). First we need to form a dataframe for the model. The code is as below.

Hurricanes$log\_damage\_norm\_std <- log(Hurricanes$damage\_norm)  
Hurricanes$log\_damage\_norm\_std <-   
 (Hurricanes$log\_damage\_norm\_std - mean(Hurricanes$log\_damage\_norm\_std)) / sd(Hurricanes$log\_damage\_norm\_std)  
  
d.04.log <- list(  
 N = NROW(Hurricanes),  
 fem = Hurricanes$femininity\_std,  
 d = Hurricanes$deaths,  
 log\_dm = Hurricanes$log\_damage\_norm\_std  
)

We first change dm into log\_dm and then standardize it. Now it’s time to formulate the model. The code is as below.

model.04.2.log.code <-"  
data{  
int N;  
int d[92];  
vector[92] log\_dm;  
vector[92] fem;  
}  
parameters{  
real a;  
real bf;  
real bdm;  
real bfdm;  
real phi;  
}  
model{  
vector[92] lambda;  
phi ~ cauchy( 0 , 1 );  
bfdm ~ normal( 0 , 3 );  
a ~ normal( 0 , 3 );  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* log\_dm[i] + bfdm \* fem[i] \* log\_dm[i];  
lambda[i] = exp(lambda[i]);  
}  
d ~ neg\_binomial\_2( lambda , phi );  
}  
generated quantities{  
vector[92] log\_lik;  
vector[92] lambda;  
for ( i in 1:92 ) {  
lambda[i] = a + bf \* fem[i] + bdm \* log\_dm[i] + bfdm \* fem[i] \* log\_dm[i];  
lambda[i] = exp(lambda[i]);  
}  
for ( i in 1:92 ) log\_lik[i] = neg\_binomial\_2\_lpmf( d[i] | lambda[i] , phi );  
}  
"

The model code is shown below.

model.04.2.log <- stan(model\_code = model.04.2.log.code, data = d.04.log, cores = 4, chains = 4)

The descriptions of the model is shown below, using print statement.

print(model.04.2.log, include=FALSE, pars=c('log\_lik','lambda'), probs = c(.1, .5, .9))

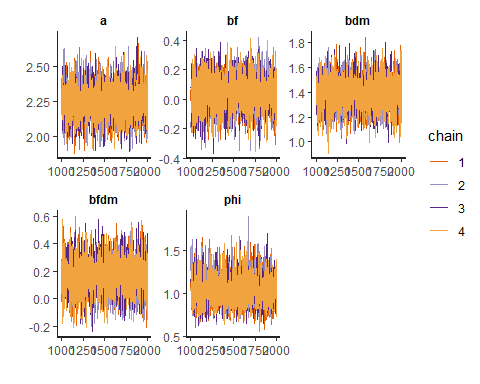
## Inference for Stan model: f4b6f0e1ca396c9d8289f4db90169a64.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.24 0.00 0.12 2.09 2.24 2.40 4445 1  
## bf 0.04 0.00 0.12 -0.11 0.04 0.19 4421 1  
## bdm 1.39 0.00 0.13 1.23 1.39 1.55 3588 1  
## bfdm 0.17 0.00 0.13 0.01 0.17 0.33 3880 1  
## phi 1.05 0.00 0.17 0.84 1.04 1.28 3925 1  
## lp\_\_ 5717.79 0.04 1.63 5715.57 5718.14 5719.55 2151 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:20:22 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 1800, which implys an solid model.

Compared to the model using damage\_norm directly, we find out some interesting features. 1. The relationship between log\_damage\_norm and death toll is greater than that of damage\_norm and death toll. 2. The interaction effects is somewhat smaller for the model using log\_damage\_norm.

Before moving on to the next model, let’s take a look at the traceplot to evaluate the decentness of the model.

traceplot(model.04.2.log, pars = c('a','bf','bdm','bfdm','phi'))

 From the traceplot above we can easily tell that the model converges.

Let’s compare those models. The code is shown below.

rethinking::compare(model.04.2, model.04.2.log)

## WAIC pWAIC dWAIC weight SE dSE  
## model.04.2.log 630.7954 5.368087 0.00000 1.000000e+00 30.77069 NA  
## model.04.2 671.0547 7.040856 40.25921 1.810601e-09 34.05445 14.21581

From the output we can tell that the one with log\_damage\_norm has a smaller WAIC comparing to the one using damage\_norm directly. We can conclude that death toll doesn’t increase exponentially with a linear increase in damage\_norm. It is much more possible that a linear relationship is the case. However, we cannot know the true generating process. From the comparison we can only tell that the model using log\_damage\_norm is much likely to be the case.

< Question 5 > For this question the dataset is bangladesh. The goal is to do modeling using use.contraception as the predictor. For the first model, we use use.contraception using dummy variables for district. Firstly let’s import the data needed.

library(rethinking)  
data("bangladesh")  
d5 <- bangladesh; rm(bangladesh)  
d5$district\_id <- as.integer(as.factor(d5$district))

Similarly to avoid stan from crashing, let’s make up a new dataset. The code is shown below.

d.05 <- list(  
 uc = d5$use.contraception,  
 dist\_id = d5$district\_id  
)

The model code is shown below.

model.05.code <-"  
data{  
 int uc[1934];  
int dist\_id[1934];  
}  
parameters{  
vector[60] a;  
}  
model{  
vector[1934] p;  
a ~ normal( 0 , 10 );  
for ( i in 1:1934 ) {  
p[i] = a[dist\_id[i]];  
p[i] = inv\_logit(p[i]);  
}  
uc ~ binomial( 1 , p );  
}  
generated quantities{  
vector[1934] log\_lik;  
vector[1934] p;  
for ( i in 1:1934 ) {  
p[i] = a[dist\_id[i]];  
p[i] = inv\_logit(p[i]);  
}  
for ( i in 1:1934 ) log\_lik[i] = binomial\_lpmf( uc[i] | 1 , p[i] );  
}  
"  
  
model.05 <- stan(model\_code = model.05.code,  
 data = d.05, cores = 4, chains = 4)

## Warning: There were 3 divergent transitions after warmup. Increasing adapt\_delta above 0.8 may help. See  
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: Examine the pairs() plot to diagnose sampling problems

The descriptions of the model is shown below, using print statement.

print(model.05, include=FALSE, pars=c('log\_lik','a','p'), probs = c(.1, .5, .9))

## Inference for Stan model: 2365a1c43d497ee7dfa8ca2853f7d837.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## lp\_\_ -1235.4 0.15 5.6 -1242.62 -1235.07 -1228.34 1417 1.01  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:21:48 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 1800, which implys an solid model.

Now let’s construct a multilevel model. The code is shown below.

model.06.code <-"  
data{  
 int uc[1934];  
int dist\_id[1934];  
}  
parameters{  
vector[60] a;  
real mu;  
real<lower=0> sigma;  
}  
model{  
vector[1934] p;  
sigma ~ cauchy( 0 , 1 );  
mu ~ normal( 0 , 5 );  
for ( i in 1:1934 ) {  
p[i] = a[dist\_id[i]];  
p[i] = inv\_logit(p[i]);  
}  
a ~ normal( mu , sigma );  
uc ~ binomial( 1 , p );  
}  
generated quantities{  
vector[1934] log\_lik;  
vector[1934] p;  
for ( i in 1:1934 ) {  
p[i] = a[dist\_id[i]];  
p[i] = inv\_logit(p[i]);  
}  
for ( i in 1:1934 ) log\_lik[i] = binomial\_lpmf( uc[i] | 1 , p[i] );  
}  
"  
  
model.06 <- stan(model\_code = model.06.code, data = d.05, cores = 4, chains = 4)

print(model.06, include=FALSE, pars=c('log\_lik','a','p'), probs = c(.1, .5, .9))

## Inference for Stan model: d037ef08c6c03c3cd9a4eb7fd3447501.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## mu -0.54 0.00 0.09 -0.65 -0.54 -0.43 4215 1  
## sigma 0.52 0.00 0.09 0.42 0.51 0.63 1216 1  
## lp\_\_ -1229.60 0.24 7.25 -1238.87 -1229.27 -1220.59 883 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 06 01:23:30 2019.  
## 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).

Rhat = 1 means that the model can reach convergence in this Markov Chain. n\_eff is greater than 1000, which implys an solid model.

Now let’s compare the two models using compare function.

rethinking::compare(model.05, model.06)

## WAIC pWAIC dWAIC weight SE dSE  
## model.06 2514.639 35.80915 0.00000 9.999086e-01 25.06489 NA  
## model.05 2533.240 62.54190 18.60131 9.135623e-05 32.91853 12.94583

We can easily tell that the multilevel model is better than the model using dummy variables since it has a lower WAIC value.

Before ending this question, let’s plot those predictions out to get a clearer overall picture. First we need to extract mean and PI value for both dummy variable model and multilevel mode. Code for data preprocessing is shown below.

library(dplyr)

##   
## Attaching package: 'dplyr'

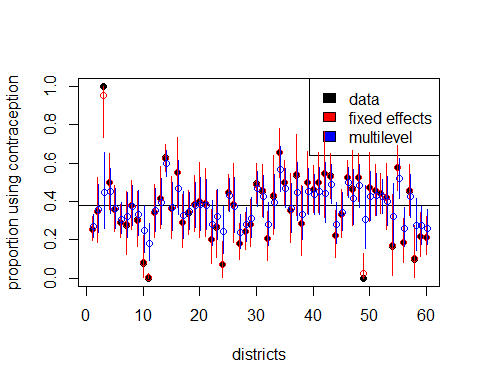
## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

d5\_new <- as.tbl(d5) %>% group\_by(district\_id) %>% summarize(p\_use\_contraception=mean(use.contraception))  
  
post.dum <- extract.samples(model.05)  
post.dum.mean <- apply(inv\_logit(post.dum$a), 2, mean)  
post.dum.PI <- apply(inv\_logit(post.dum$a), 2, PI, prob=.89)  
  
post.mul <- extract.samples(model.06)  
post.mul.mean <- apply(inv\_logit(post.mul$a), 2, mean)  
post.mul.PI <- apply(inv\_logit(post.mul$a), 2, PI, prob=.89)

The code for creating plots for the dummy variable model is shown as below.

plot(y=d5\_new$p\_use\_contraception, x=d5\_new$district\_id, pch=19, xlab="districts", ylab="proportion using contraception")  
abline(h=median(d5\_new$p\_use\_contraception))  
legend("topright",legend=c("data","fixed effects", "multilevel"),fill=c("black","red","blue"))  
  
points(y=post.dum.mean,x=1:60, col="red")  
for(i in 1:60){segments(x0=i,x1=i,y0=post.dum.PI[1,i],y1=post.dum.PI[2,i],col="red")}  
  
points(y=post.mul.mean,x=(1:60)+0.2, col="blue")  
for(i in 1:60){segments(x0=i+0.2,x1=i+0.2,y0=post.mul.PI[1,i],y1=post.mul.PI[2,i],col="blue")}



From the plot we can see that multilevel model is a better one since the extreme value are all brought back to the median. The distance between median also shrinks.