Hw7

Cheng

6/4/2019

# Question1

fit Poisson regression using femininity as a predictor

# load data  
data(Hurricanes)  
df <- Hurricanes; remove(Hurricanes)

m1.1: Possion model of deaths using frmininity as a predictor

# Setup  
m1.1 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem[N];  
 }  
  
 parameters {  
 real a;  
 real bf;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,10);  
 bf ~ normal(0,10);  
 for(i in 1:N) lambda[i] = a + bf \* fem[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + bf \* fem[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem = df$femininity  
)  
fit1.1 = stan(model\_code = m1.1,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

m1.2: Possion model of deaths using intercept only

m1.2 <- '  
 data {  
 int N;  
 int deaths[N];  
 }  
  
 parameters {  
 real a;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,10);  
 for(i in 1:N) lambda[i] = a ;  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a;  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths  
)  
fit1.2 = stan(model\_code = m1.2,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

Output

print(fit1.1, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 79cff282806f576952fc98a21b54dfb9.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.50 0.00 0.06 2.41 2.50 2.57 267 1  
## bf 0.07 0.00 0.01 0.06 0.07 0.08 270 1  
## lp\_\_ 3899.09 0.05 0.95 3897.91 3899.39 3899.95 331 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:14:33 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).

print(fit1.2, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 6c50c596a582515f2940a5405bece3e3.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 3.03 0.00 0.02 3.00 3.03 3.06 357 1.00  
## lp\_\_ 3852.34 0.03 0.64 3851.52 3852.56 3852.81 409 1.01  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:15:10 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).

Model Comparision using WAIC

fit\_list <- list(fit1.1, fit1.2)  
ll\_list <- lapply(fit\_list, extract\_log\_lik)   
exp\_ll\_list <- lapply(ll\_list, exp)  
rel\_n\_eff\_list <- lapply(exp\_ll\_list, relative\_eff, chain\_id = c(rep(1, 500), rep(2, 500)))  
# loo  
waic\_list <- list()  
for(i in 1:2) {  
waic\_list[[i]] <- waic(ll\_list[[i]], r\_eff = rel\_n\_eff\_list[[i]], cores = 4)  
}  
names(waic\_list) <- c('fit1.1', 'fit1.2')  
loo::compare(x = waic\_list)

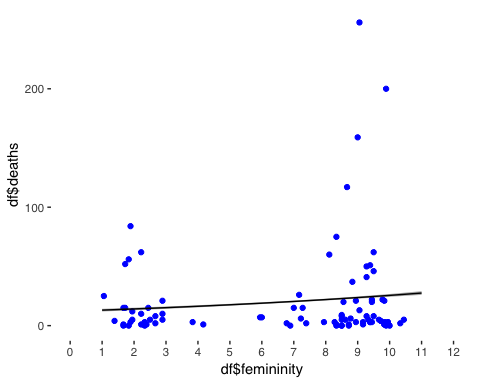
## elpd\_diff se   
## -20.5 71.7

According to WAIC comparison model with femininity is better and take all score, but the dispersion of the difference and SE of WAIC itself is huge. The difference is less than standard error of the difference.

post1 = as.data.frame(fit1.1)  
nd <- expand.grid(fem = seq(1, 11))  
  
# get posterior parameters of each  
f\_mu\_2 <- function(fem) with(post1, a + bf \* fem)  
mu <- mapply(f\_mu\_2, fem = nd$fem)  
mu <- exp(mu)  
mu\_mean <- colMeans(mu)   
mu\_hpdi <- apply(mu, 2, rethinking::PI)  
nd <- nd %>%  
 mutate(mu = mu\_mean,  
 mu\_hpdi\_l = mu\_hpdi[1,],  
 mu\_hpdi\_h = mu\_hpdi[2,])

Plot m1.1:

ggplot() +   
geom\_point(data = df, aes(x = df$femininity, y = df$deaths), colour = 'blue') +  
geom\_ribbon(data = nd, aes(x = fem, ymin = mu\_hpdi\_l, ymax = mu\_hpdi\_h), alpha = 0.4) +  
geom\_line(data = nd, aes(x = fem, y = mu)) +  
scale\_color\_manual(values = "black", aesthetics = "colour") +  
scale\_x\_continuous(breaks = seq(0, 12), limits=c(0, 12))



Intuitively there is some hidden variable that better explains deaths.Visually relation induced by the model looks suspicious for me because it looks like being caused by several outliers.

# Question2

Using gamma-Poisson model to predict deaths using femininity:

m2.1 <- "   
data {  
 int N;  
 int deaths[N];  
 real fem[N];  
}  
parameters {  
 real a;  
 real bf;  
 real shape;  
}  
model {  
 vector[N] mu;  
  
 a ~ normal(0,5);   
 bf ~ normal(0,5);  
 shape ~ exponential(1);  
   
 for(i in 1:N) mu[i] = a + bf \* fem[i];  
  
 target += neg\_binomial\_2\_log\_lpmf( deaths | mu, shape); // use log link  
}  
"

Run

dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem = df$femininity  
)  
fit2.1 = stan(model\_code = m2.1,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

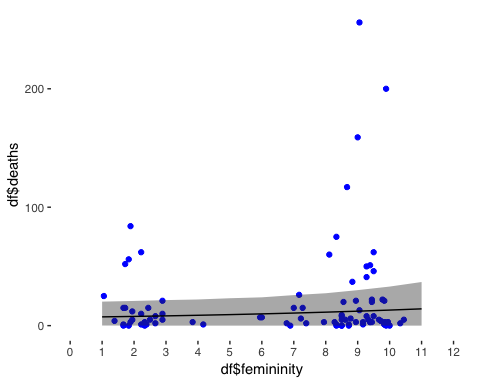
print(fit2.1)

## Inference for Stan model: 1c862463f17a7ef510939b003804d438.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 2.5% 25% 50% 75% 97.5%  
## a 0.79 1.78 1.79 -0.98 -0.98 0.36 2.55 3.16  
## bf -0.60 0.67 0.67 -1.27 -1.27 -0.68 0.07 0.14  
## shape 0.22 0.23 0.23 0.00 0.00 0.15 0.44 0.55  
## lp\_\_ -941.85 588.46 588.76 -1532.13 -1529.99 -944.70 -353.09 -352.12  
## n\_eff Rhat  
## a 1 8.25  
## bf 1 23.80  
## shape 1 6.07  
## lp\_\_ 1 657.07  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:16:05 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).

Plot

post2 = as.data.frame(fit2.1)  
nd <- expand.grid(fem = seq(1, 11))  
  
# get posterior parameters of each  
f\_mu\_2 <- function(fem) with(post2, a + bf \* fem)  
mu <- mapply(f\_mu\_2, fem = nd$fem)  
mu <- exp(mu)  
mu\_mean <- colMeans(mu)   
mu\_hpdi <- apply(mu, 2, rethinking::PI)  
nd <- nd %>%  
 mutate(mu = mu\_mean,  
 mu\_hpdi\_l = mu\_hpdi[1,],  
 mu\_hpdi\_h = mu\_hpdi[2,])

ggplot() +   
geom\_point(data = df, aes(x = df$femininity, y = df$deaths), colour = 'blue') +  
geom\_ribbon(data = nd, aes(x = fem, ymin = mu\_hpdi\_l, ymax = mu\_hpdi\_h), alpha = 0.4) +  
geom\_line(data = nd, aes(x = fem, y = mu)) +  
scale\_color\_manual(values = "black", aesthetics = "colour") +  
scale\_x\_continuous(breaks = seq(0, 12), limits=c(0, 12))



why the association diminished in strength? I think this is because a much larger range of values of (alpha, beta\_femininity) can produce, through the Gamma distribution, the same expectedvalues lambda via the variance in the Gamma process itself.

# Question3

Fit a series of models with varying interactions:

# Standardize predictors  
df = df %>%  
 mutate(fem\_std = as.vector(scale(femininity))) %>%  
 mutate(min\_std = as.vector(scale(min\_pressure))) %>%  
 mutate(dam\_std = as.vector(scale(damage\_norm)))

m3.1: No interaction: lambda ~ a + b\_fem*fem\_std + b\_dam*dam\_std + b\_mp\*min\_std

# Setup  
m3.1 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem\_std[N];  
 real dam\_std[N];  
 real min\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_fem;  
 real b\_dam;  
 real b\_min;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_fem ~ normal(0,1);  
 b\_dam ~ normal(0,1);  
 b\_min ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem\_std = df$fem\_std,  
 dam\_std = df$dam\_std,  
 min\_std = df$min\_std  
)  
fit3.1 = stan(model\_code = m3.1,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

m3.2: with interaction: b\_fem\_dam*fem\_std[i]*dam\_std[i]

# Setup  
m3.2 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem\_std[N];  
 real dam\_std[N];  
 real min\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_fem;  
 real b\_dam;  
 real b\_min;  
 real b\_fem\_dam;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_fem ~ normal(0,1);  
 b\_dam ~ normal(0,1);  
 b\_min ~ normal(0,1);  
 b\_fem\_dam ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i] +  
 b\_fem\_dam\*fem\_std[i]\*dam\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i] +  
 b\_fem\_dam\*fem\_std[i]\*dam\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem\_std = df$fem\_std,  
 dam\_std = df$dam\_std,  
 min\_std = df$min\_std  
)  
fit3.2 = stan(model\_code = m3.2,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

m3.3: w/ interaction, b\_fem\_min*fem\_std[i]*min\_std[i]

# Setup  
m3.3 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem\_std[N];  
 real dam\_std[N];  
 real min\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_fem;  
 real b\_dam;  
 real b\_min;  
 real b\_fem\_min;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_fem ~ normal(0,1);  
 b\_dam ~ normal(0,1);  
 b\_min ~ normal(0,1);  
 b\_fem\_min ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i] +  
 b\_fem\_min\*fem\_std[i]\*min\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_fem\*fem\_std[i] + b\_dam\*dam\_std[i] + b\_min\*min\_std[i] +  
 b\_fem\_min\*fem\_std[i]\*min\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem\_std = df$fem\_std,  
 dam\_std = df$dam\_std,  
 min\_std = df$min\_std  
)  
fit3.3 = stan(model\_code = m3.3,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

m3.4: no feminiity

# Setup  
m3.4 <- '  
 data {  
 int N;  
 int deaths[N];  
 real dam\_std[N];  
 real min\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_dam;  
 real b\_min;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_dam ~ normal(0,1);  
 b\_min ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_dam\*dam\_std[i] + b\_min\*min\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_dam\*dam\_std[i] + b\_min\*min\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 dam\_std = df$dam\_std,  
 min\_std = df$min\_std  
)  
fit3.4 = stan(model\_code = m3.4,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

Output

print(fit3.1, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: cbf786d609ffca5f09c4362b2d221789.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.70 0.00 0.03 2.66 2.70 2.73 751 1  
## b\_fem 0.27 0.00 0.02 0.23 0.27 0.30 1002 1  
## b\_dam 0.33 0.00 0.01 0.31 0.33 0.34 701 1  
## b\_min -0.43 0.00 0.03 -0.46 -0.43 -0.40 654 1  
## lp\_\_ 4630.34 0.05 1.22 4628.61 4630.60 4631.67 492 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:16:40 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).

print(fit3.2, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 5d4bf6c4722b4f8e8dee45254eefcbfb.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.71 0.00 0.03 2.67 2.70 2.74 812 1  
## b\_fem 0.17 0.00 0.03 0.14 0.17 0.21 678 1  
## b\_dam 0.30 0.00 0.01 0.28 0.30 0.32 782 1  
## b\_min -0.46 0.00 0.02 -0.49 -0.46 -0.43 709 1  
## b\_fem\_dam 0.08 0.00 0.01 0.07 0.08 0.10 842 1  
## lp\_\_ 4647.84 0.07 1.49 4645.80 4648.12 4649.47 480 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:17:15 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).

print(fit3.3, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: cdd71e4214f0ed7464f865884b1f4b86.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.70 0.00 0.03 2.66 2.70 2.74 838 1  
## b\_fem 0.24 0.00 0.03 0.19 0.24 0.28 770 1  
## b\_dam 0.33 0.00 0.02 0.31 0.33 0.35 509 1  
## b\_min -0.42 0.00 0.03 -0.45 -0.42 -0.38 451 1  
## b\_fem\_min -0.04 0.00 0.02 -0.07 -0.04 -0.01 772 1  
## lp\_\_ 4631.02 0.08 1.58 4628.82 4631.34 4632.74 385 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:17:50 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).

print(fit3.4, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: c1b352d705e46a308680345315c13142.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.74 0.00 0.03 2.70 2.74 2.77 684 1  
## b\_dam 0.31 0.00 0.01 0.30 0.31 0.33 596 1  
## b\_min -0.43 0.00 0.03 -0.46 -0.43 -0.40 511 1  
## lp\_\_ 4572.50 0.05 1.12 4571.05 4572.78 4573.60 465 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:18:28 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).

Model comparision using WAIC

fit\_list <- list(fit3.1, fit3.2, fit3.3, fit3.4)  
ll\_list <- lapply(fit\_list, extract\_log\_lik)   
exp\_ll\_list <- lapply(ll\_list, exp)  
rel\_n\_eff\_list <- lapply(exp\_ll\_list, relative\_eff, chain\_id = c(rep(1, 500), rep(2, 500)))  
# loo  
waic\_list <- list()  
for(i in 1:4) {  
waic\_list[[i]] <- waic(ll\_list[[i]], r\_eff = rel\_n\_eff\_list[[i]], cores = 4)  
}  
names(waic\_list) <- c('fit3.1', 'fit3.2', 'fit3.3', 'fit3.4')  
loo::compare(x = waic\_list)

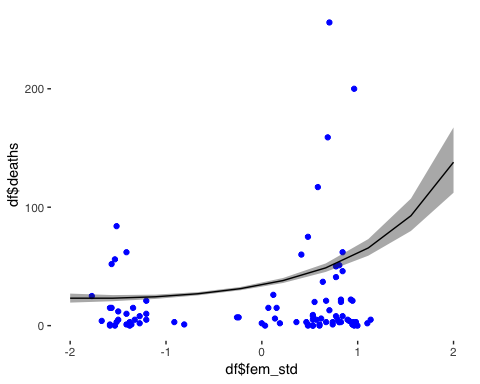
## elpd\_diff se\_diff elpd\_waic p\_waic waic   
## fit3.2 0.0 0.0 -1579.8 280.5 3159.7  
## fit3.1 -12.1 35.7 -1591.9 268.2 3183.9  
## fit3.4 -63.2 96.9 -1643.0 264.1 3286.0  
## fit3.3 -71.7 43.5 -1651.5 341.5 3303.0

Best model according to the WAIC is m3.2 which is using damage and feminiity as interaction term.

Plot

post3.2 = as.data.frame(fit3.2)  
nd = expand.grid(fem\_std = seq(-2,2,length.out = 10)) %>%  
 mutate(dam\_std = seq(-1,5,length.out = 10)) %>%  
 mutate(min\_std = seq(-2,1,length.out = 10))  
  
# get posterior parameters of each  
f\_mu\_2 <- function(fem\_std, dam\_std, min\_std) with(post3.2, a + b\_fem\*fem\_std + b\_dam\*dam\_std + b\_min\*min\_std + b\_fem\_dam\*fem\_std\*dam\_std)  
mu <- mapply(f\_mu\_2, fem\_std = nd$fem\_std, dam\_std = nd$dam\_std, min\_std = nd$min\_std)  
mu <- exp(mu)  
mu\_mean <- colMeans(mu)   
mu\_hpdi <- apply(mu, 2, rethinking::PI)  
nd <- nd %>%  
 mutate(mu = mu\_mean,  
 mu\_hpdi\_l = mu\_hpdi[1,],  
 mu\_hpdi\_h = mu\_hpdi[2,])

ggplot() +   
geom\_point(data = df, aes(x = df$fem\_std, y = df$deaths), colour = 'blue') +  
geom\_ribbon(data = nd, aes(x = fem\_std, ymin = mu\_hpdi\_l, ymax = mu\_hpdi\_h), alpha = 0.4) +  
geom\_line(data = nd, aes(x = fem\_std, y = mu)) +  
scale\_color\_manual(values = "black", aesthetics = "colour")



# Question 4

df$log\_dam\_std <- as.vector(scale(log(df$damage\_norm)))

m4.1: Try

# Setup  
m4.1 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem\_std[N];  
 real log\_dam\_std[N];  
 real min\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_fem;  
 real b\_ldam;  
 real b\_min;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_fem ~ normal(0,1);  
 b\_ldam ~ normal(0,1);  
 b\_min ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_fem\*fem\_std[i] + b\_ldam\* log\_dam\_std[i] + b\_min\*min\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_fem\*fem\_std[i] + b\_ldam\*log\_dam\_std[i] + b\_min\*min\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

Run log(damage\_norm)

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem\_std = df$femininity,  
 log\_dam\_std = df$log\_dam\_std,  
 min\_std = df$min\_std  
)  
fit4.1 = stan(model\_code = m4.1,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

m4.2: the setting is same as 3.1 but using log(damage\_norm)

# Setup  
m4.2 <- '  
 data {  
 int N;  
 int deaths[N];  
 real fem\_std[N];  
 real log\_dam\_std[N];  
 }  
  
 parameters {  
 real a;  
 real b\_fem;  
 real b\_ldam;  
 real b\_fem\_ldam;  
 }  
   
 model {  
 vector[N] lambda;  
 a ~ normal(0,5);  
 b\_fem ~ normal(0,1);  
 b\_ldam ~ normal(0,1);  
 b\_fem\_ldam ~ normal(0,1);  
  
 for(i in 1:N) lambda[i] = a + b\_fem\*fem\_std[i] + b\_ldam\*log\_dam\_std[i] +  
 b\_fem\_ldam\*fem\_std[i]\*log\_dam\_std[i];  
 deaths ~ poisson\_log(lambda);  
 }  
  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] lambda;  
 for(i in 1:N) {  
 lambda[i] = a + b\_fem\*fem\_std[i] + b\_ldam\*log\_dam\_std[i] +  
 b\_fem\_ldam\*fem\_std[i]\*log\_dam\_std[i];  
 log\_lik[i] = poisson\_log\_lpmf(deaths[i] | lambda[i]);  
 }  
 }  
 }  
'

# Run  
dat = list(  
 N = nrow(df),  
 deaths = df$deaths,  
 fem\_std = df$fem\_std,  
 log\_dam\_std = df$log\_dam\_std  
)  
fit4.2 = stan(model\_code = m4.2,   
 data = dat,   
 iter = 1000,   
 chains = 2,   
 cores = 2)

Output

print(fit4.1, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: f5580a04e326be66f1a42a39bbf1108f.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 1.55 0.00 0.08 1.45 1.55 1.64 470 1  
## b\_fem 0.09 0.00 0.01 0.08 0.09 0.10 506 1  
## b\_ldam 1.56 0.00 0.05 1.50 1.56 1.62 573 1  
## b\_min -0.01 0.00 0.03 -0.05 -0.01 0.03 636 1  
## lp\_\_ 5018.73 0.08 1.50 5016.82 5019.08 5020.23 398 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:19:05 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).

print(fit4.2, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 0f6b38cebd5fcfc877c20012f0377992.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a 2.18 0.00 0.04 2.13 2.17 2.23 558 1  
## b\_fem 0.01 0.00 0.04 -0.04 0.01 0.07 539 1  
## b\_ldam 1.51 0.00 0.04 1.46 1.51 1.56 547 1  
## b\_fem\_ldam 0.30 0.00 0.04 0.25 0.30 0.35 561 1  
## lp\_\_ 5043.73 0.07 1.39 5041.81 5044.09 5045.21 437 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:19:42 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).

Model Comparsion musing WAIC

fit\_list <- list(fit4.1, fit4.2, fit3.1, fit3.2)  
ll\_list <- lapply(fit\_list, extract\_log\_lik)   
exp\_ll\_list <- lapply(ll\_list, exp)  
rel\_n\_eff\_list <- lapply(exp\_ll\_list, relative\_eff, chain\_id = c(rep(1, 500), rep(2, 500)))  
# loo  
waic\_list <- list()  
for(i in 1:4) {  
waic\_list[[i]] <- waic(ll\_list[[i]], r\_eff = rel\_n\_eff\_list[[i]], cores = 4)  
}  
names(waic\_list) <- c('fit4.1', 'fit4.2', 'fit3.1', 'fit3.2')  
loo::compare(x = waic\_list)

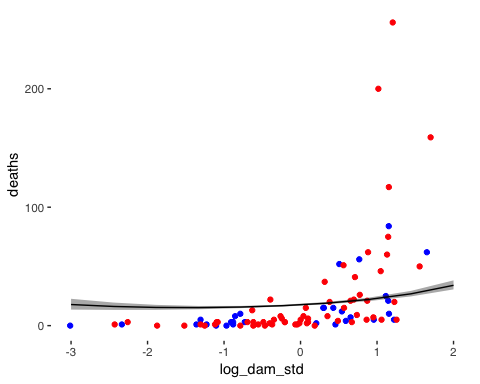
## elpd\_diff se\_diff elpd\_waic p\_waic waic   
## fit4.2 0.0 0.0 -1047.9 107.2 2095.7  
## fit4.1 -187.1 125.2 -1234.9 311.9 2469.9  
## fit3.2 -532.0 343.0 -1579.8 280.5 3159.7  
## fit3.1 -544.1 331.0 -1591.9 268.2 3183.9

Model 4.2 performs is the best model in this case.

Plot

post3.2 = as.data.frame(fit3.2)  
nd = expand.grid(fem\_std = seq(-2,2,length.out = 10)) %>%  
 mutate(log\_dam\_std = seq(-3,2,length.out = 10)) %>%  
 mutate(min\_std = seq(-2,1,length.out = 10))  
  
# get posterior parameters of each  
f\_mu\_2 <- function(fem\_std, dam\_std, min\_std) with(post3.2, a + b\_fem\*fem\_std + b\_dam\*dam\_std + b\_min\*min\_std + b\_fem\_dam\*fem\_std\*dam\_std)  
mu <- mapply(f\_mu\_2, fem\_std = nd$fem\_std, dam\_std = nd$log\_dam\_std, min\_std = nd$min\_std)  
mu <- exp(mu)  
mu\_mean <- colMeans(mu)   
mu\_hpdi <- apply(mu, 2, rethinking::PI)  
nd <- nd %>%  
 mutate(mu = mu\_mean,  
 mu\_hpdi\_l = mu\_hpdi[1,],  
 mu\_hpdi\_h = mu\_hpdi[2,])

df.male <- filter(df, female!=1)  
df.female <- filter(df, female==1)  
ggplot() +   
 geom\_point(data = df.male , aes(x = log\_dam\_std, y = deaths), color = 'blue') +  
 geom\_point(data = df.female , aes(x = log\_dam\_std, y = deaths), color = 'red') +  
 geom\_ribbon(data = nd, aes(x = log\_dam\_std, ymin = mu\_hpdi\_l, ymax = mu\_hpdi\_h), alpha = 0.4) +  
 geom\_line(data = nd, aes(x = log\_dam\_std, y = mu)) +  
 scale\_color\_manual(values = "black", aesthetics = "colour")



With log scale of damage there is no need for min\_pressure variable.Model becomes more accurate in predictions. According to coefficients it’s only interaction of femininicity and log(damage\_norm) that really matters.

# Question 5

Load data

data("bangladesh")  
d <- bangladesh

Data pre-processing

d$district\_id <- as.integer(as.factor(d$district))  
d$use\_contraception <- d$use.contraception  
d$age\_centered <- d$age.centered  
d$living\_children <- d$living.children  
d <- select(d,-use.contraception, -age.centered, -district, -living.children)

m5.1: fixed effect model using dummy variables for district

m5.1='  
 data {  
 int N;  
 int K; // number of clusters  
 int dis\_id[N];  
 int use\_con[N];  
 }  
 parameters {  
 vector[K] a\_dis;  
 }  
 model {  
 vector[N] p;  
 target += normal\_lpdf(a\_dis | 0, 5);  
  
 for (i in 1:N) p[i] = inv\_logit( a\_dis[dis\_id[i]]);   
 target += binomial\_lpmf(use\_con | 1, p);  
 }  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] p;  
 for(i in 1:N) {  
 p[i] = inv\_logit( a\_dis[dis\_id[i]] );  
 log\_lik[i] = binomial\_lpmf(use\_con[i] | 1, p[i]);  
 }  
 }  
 }'

Run the model

dat <- list(  
 N = nrow(d),  
 use\_con = d$use\_contraception,  
 K = max(d$district\_id),  
 dis\_id = d$district\_id  
)  
fit5.1 <- stan(model\_code = m5.1, data = dat, iter = 1000, chains = 2, cores = 2)

m5.2: multilevel model with varing intercepts for district

m5.2='  
 data {  
 int N;  
 int K; // number of clusters  
 int dis\_id[N];  
 int use\_con[N];  
 }  
 parameters {  
 vector[K] a\_dis;  
 real alpha;  
 real <lower=0> sigma;  
 }  
 model {  
 vector[N] p;  
 target += normal\_lpdf(alpha | 0, 1);  
 target += cauchy\_lpdf(sigma | 0, 1);  
 target += normal\_lpdf(a\_dis | alpha, sigma);  
  
 for (i in 1:N) p[i] = inv\_logit( a\_dis[dis\_id[i]]);   
 target += binomial\_lpmf(use\_con | 1, p);  
 }  
 generated quantities {  
 vector[N] log\_lik;  
 {  
 vector[N] p;  
 for(i in 1:N) {  
 p[i] = inv\_logit( a\_dis[dis\_id[i]] );  
 log\_lik[i] = binomial\_lpmf(use\_con[i] | 1, p[i]);  
 }  
 }  
 }'

dat <- list(  
 N = nrow(d),  
 use\_con = d$use\_contraception,  
 K = max(d$district\_id),  
 dis\_id = d$district\_id  
)  
fit5.2 <- stan(model\_code = m5.2, data = dat, iter = 1000, chains = 2, cores = 2)

Output

print(fit5.1, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 37460738b544146aa9e27bafd6627ae2.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a\_dis[1] -1.08 0.01 0.22 -1.36 -1.07 -0.79 1868 1  
## a\_dis[2] -0.63 0.01 0.45 -1.24 -0.61 -0.05 2525 1  
## a\_dis[3] 4.61 0.11 3.28 0.91 4.02 9.13 967 1  
## a\_dis[4] 0.00 0.01 0.38 -0.48 0.00 0.46 3075 1  
## a\_dis[5] -0.60 0.01 0.33 -1.04 -0.59 -0.18 2213 1  
## a\_dis[6] -0.89 0.00 0.27 -1.24 -0.89 -0.56 3058 1  
## a\_dis[7] -0.99 0.01 0.53 -1.65 -0.98 -0.34 2912 1  
## a\_dis[8] -0.50 0.01 0.36 -0.98 -0.50 -0.04 3574 1  
## a\_dis[9] -0.86 0.01 0.45 -1.44 -0.84 -0.29 2761 1  
## a\_dis[10] -2.77 0.04 1.12 -4.15 -2.62 -1.51 736 1  
## a\_dis[11] -6.20 0.09 2.64 -9.93 -5.67 -3.34 860 1  
## a\_dis[12] -0.66 0.01 0.40 -1.19 -0.66 -0.15 2806 1  
## a\_dis[13] -0.34 0.01 0.42 -0.88 -0.34 0.20 2903 1  
## a\_dis[14] 0.52 0.00 0.19 0.27 0.52 0.76 2483 1  
## a\_dis[15] -0.57 0.01 0.44 -1.14 -0.56 0.00 2595 1  
## a\_dis[16] 0.21 0.01 0.46 -0.38 0.21 0.79 2952 1  
## a\_dis[17] -0.93 0.01 0.46 -1.51 -0.93 -0.36 1601 1  
## a\_dis[18] -0.67 0.01 0.32 -1.07 -0.67 -0.28 1826 1  
## a\_dis[19] -0.49 0.01 0.41 -1.03 -0.48 0.02 3036 1  
## a\_dis[20] -0.43 0.01 0.58 -1.17 -0.43 0.29 2995 1  
## a\_dis[21] -0.49 0.01 0.52 -1.14 -0.50 0.17 1756 1  
## a\_dis[22] -1.45 0.01 0.57 -2.20 -1.42 -0.73 1939 1  
## a\_dis[23] -1.08 0.01 0.58 -1.78 -1.07 -0.41 2084 1  
## a\_dis[24] -2.87 0.03 1.12 -4.20 -2.74 -1.63 1125 1  
## a\_dis[25] -0.20 0.00 0.23 -0.50 -0.20 0.10 2213 1  
## a\_dis[26] -0.51 0.01 0.64 -1.32 -0.48 0.25 2331 1  
## a\_dis[27] -1.53 0.01 0.41 -2.05 -1.51 -1.00 2886 1  
## a\_dis[28] -1.15 0.01 0.34 -1.58 -1.14 -0.74 2243 1  
## a\_dis[29] -0.98 0.01 0.40 -1.48 -0.97 -0.48 2356 1  
## a\_dis[30] -0.02 0.01 0.27 -0.38 -0.01 0.32 2525 1  
## a\_dis[31] -0.19 0.01 0.34 -0.61 -0.18 0.25 1877 1  
## a\_dis[32] -1.38 0.01 0.49 -2.02 -1.34 -0.77 3396 1  
## a\_dis[33] -0.29 0.01 0.54 -0.96 -0.30 0.35 1906 1  
## a\_dis[34] 0.67 0.01 0.37 0.20 0.67 1.14 2189 1  
## a\_dis[35] 0.00 0.01 0.29 -0.38 0.00 0.39 2641 1  
## a\_dis[36] -0.64 0.01 0.56 -1.38 -0.62 0.05 2349 1  
## a\_dis[37] 0.16 0.01 0.58 -0.57 0.16 0.94 3155 1  
## a\_dis[38] -0.97 0.01 0.63 -1.76 -0.94 -0.18 2572 1  
## a\_dis[39] 0.00 0.01 0.40 -0.50 0.00 0.50 3472 1  
## a\_dis[40] -0.15 0.01 0.31 -0.56 -0.13 0.25 2445 1  
## a\_dis[41] 0.00 0.01 0.41 -0.51 0.00 0.52 3919 1  
## a\_dis[42] 0.19 0.01 0.64 -0.64 0.19 1.02 2401 1  
## a\_dis[43] 0.14 0.01 0.30 -0.24 0.14 0.52 2876 1  
## a\_dis[44] -1.31 0.01 0.46 -1.93 -1.28 -0.75 2146 1  
## a\_dis[45] -0.71 0.01 0.35 -1.16 -0.70 -0.26 1593 1  
## a\_dis[46] 0.09 0.00 0.22 -0.20 0.09 0.36 3802 1  
## a\_dis[47] -0.14 0.01 0.55 -0.83 -0.13 0.54 2178 1  
## a\_dis[48] 0.11 0.01 0.30 -0.27 0.10 0.48 3357 1  
## a\_dis[49] -4.95 0.10 2.85 -8.83 -4.45 -1.76 785 1  
## a\_dis[50] -0.09 0.01 0.45 -0.66 -0.10 0.47 1913 1  
## a\_dis[51] -0.16 0.01 0.32 -0.56 -0.17 0.26 2336 1  
## a\_dis[52] -0.23 0.01 0.25 -0.57 -0.23 0.09 2356 1  
## a\_dis[53] -0.31 0.01 0.49 -0.93 -0.31 0.31 2550 1  
## a\_dis[54] -1.97 0.04 1.25 -3.61 -1.82 -0.53 1253 1  
## a\_dis[55] 0.32 0.01 0.29 -0.06 0.30 0.69 1970 1  
## a\_dis[56] -1.56 0.01 0.49 -2.15 -1.54 -0.97 2328 1  
## a\_dis[57] -0.17 0.01 0.35 -0.62 -0.18 0.30 2277 1  
## a\_dis[58] -2.48 0.03 1.10 -3.96 -2.33 -1.21 1610 1  
## a\_dis[59] -1.30 0.01 0.44 -1.86 -1.27 -0.76 2647 1  
## a\_dis[60] -1.33 0.01 0.38 -1.83 -1.32 -0.82 2566 1  
## lp\_\_ -1388.38 0.31 5.83 -1396.25 -1388.04 -1381.06 357 1  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:20:22 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).

print(fit5.2, include = F, pars = 'log\_lik', probs = c(.1, .5, .9))

## Inference for Stan model: 4b86a80683fdfa0f622e0fd08b51bdb6.  
## 2 chains, each with iter=1000; warmup=500; thin=1;   
## post-warmup draws per chain=500, total post-warmup draws=1000.  
##   
## mean se\_mean sd 10% 50% 90% n\_eff Rhat  
## a\_dis[1] -1.00 0.01 0.19 -1.23 -1.01 -0.76 1054 1.00  
## a\_dis[2] -0.59 0.01 0.37 -1.08 -0.60 -0.11 1093 1.00  
## a\_dis[3] -0.23 0.02 0.51 -0.87 -0.25 0.42 994 1.00  
## a\_dis[4] -0.18 0.01 0.28 -0.55 -0.18 0.20 1030 1.00  
## a\_dis[5] -0.57 0.01 0.28 -0.92 -0.56 -0.20 922 1.00  
## a\_dis[6] -0.81 0.01 0.25 -1.13 -0.82 -0.50 769 1.00  
## a\_dis[7] -0.75 0.01 0.36 -1.22 -0.75 -0.30 1177 1.00  
## a\_dis[8] -0.52 0.01 0.30 -0.91 -0.52 -0.14 1230 1.00  
## a\_dis[9] -0.70 0.01 0.31 -1.09 -0.69 -0.33 995 1.00  
## a\_dis[10] -1.13 0.02 0.45 -1.70 -1.10 -0.57 750 1.00  
## a\_dis[11] -1.53 0.02 0.46 -2.10 -1.51 -0.97 745 1.00  
## a\_dis[12] -0.61 0.01 0.30 -1.02 -0.62 -0.23 1314 1.00  
## a\_dis[13] -0.41 0.01 0.32 -0.80 -0.42 -0.01 1339 1.00  
## a\_dis[14] 0.40 0.01 0.18 0.16 0.39 0.64 938 1.00  
## a\_dis[15] -0.55 0.01 0.33 -0.98 -0.54 -0.14 1279 1.00  
## a\_dis[16] -0.10 0.01 0.34 -0.53 -0.12 0.35 1016 1.00  
## a\_dis[17] -0.77 0.01 0.33 -1.18 -0.76 -0.37 1045 1.00  
## a\_dis[18] -0.64 0.01 0.28 -0.98 -0.64 -0.28 1139 1.00  
## a\_dis[19] -0.50 0.01 0.32 -0.89 -0.50 -0.10 837 1.00  
## a\_dis[20] -0.46 0.01 0.37 -0.94 -0.46 0.01 1361 1.00  
## a\_dis[21] -0.49 0.01 0.38 -0.96 -0.48 -0.01 1065 1.00  
## a\_dis[22] -0.96 0.01 0.38 -1.43 -0.96 -0.47 927 1.00  
## a\_dis[23] -0.77 0.01 0.39 -1.27 -0.76 -0.30 1241 1.00  
## a\_dis[24] -1.17 0.01 0.42 -1.71 -1.17 -0.65 858 1.00  
## a\_dis[25] -0.26 0.01 0.22 -0.55 -0.26 0.02 1234 1.00  
## a\_dis[26] -0.51 0.01 0.36 -0.97 -0.51 -0.03 1342 1.00  
## a\_dis[27] -1.18 0.01 0.30 -1.56 -1.15 -0.81 849 1.00  
## a\_dis[28] -0.96 0.01 0.27 -1.32 -0.96 -0.65 1130 1.00  
## a\_dis[29] -0.81 0.01 0.32 -1.23 -0.80 -0.39 1222 1.00  
## a\_dis[30] -0.15 0.01 0.23 -0.43 -0.14 0.13 1024 1.00  
## a\_dis[31] -0.30 0.01 0.29 -0.68 -0.29 0.07 1125 1.00  
## a\_dis[32] -0.96 0.01 0.37 -1.43 -0.96 -0.51 1127 1.00  
## a\_dis[33] -0.44 0.01 0.39 -0.95 -0.43 0.06 1029 1.00  
## a\_dis[34] 0.28 0.01 0.30 -0.10 0.28 0.66 571 1.00  
## a\_dis[35] -0.13 0.01 0.27 -0.47 -0.12 0.20 1060 1.00  
## a\_dis[36] -0.58 0.01 0.37 -1.04 -0.58 -0.10 1128 1.00  
## a\_dis[37] -0.23 0.01 0.37 -0.71 -0.24 0.25 922 1.00  
## a\_dis[38] -0.71 0.01 0.37 -1.19 -0.71 -0.25 1004 1.00  
## a\_dis[39] -0.21 0.01 0.32 -0.63 -0.22 0.22 1191 1.00  
## a\_dis[40] -0.26 0.01 0.27 -0.61 -0.25 0.09 864 1.00  
## a\_dis[41] -0.20 0.01 0.31 -0.60 -0.20 0.19 981 1.00  
## a\_dis[42] -0.21 0.01 0.41 -0.71 -0.22 0.31 1081 1.00  
## a\_dis[43] -0.03 0.01 0.27 -0.38 -0.04 0.30 967 1.00  
## a\_dis[44] -0.96 0.01 0.33 -1.39 -0.95 -0.55 1035 1.00  
## a\_dis[45] -0.66 0.01 0.28 -1.01 -0.66 -0.29 958 1.00  
## a\_dis[46] -0.01 0.01 0.19 -0.25 -0.01 0.24 1157 1.00  
## a\_dis[47] -0.35 0.01 0.37 -0.80 -0.35 0.13 1248 1.00  
## a\_dis[48] -0.07 0.01 0.25 -0.39 -0.08 0.25 1507 1.00  
## a\_dis[49] -0.88 0.02 0.49 -1.50 -0.85 -0.28 1026 1.00  
## a\_dis[50] -0.29 0.01 0.36 -0.76 -0.29 0.18 1349 1.00  
## a\_dis[51] -0.26 0.01 0.27 -0.60 -0.26 0.08 978 1.00  
## a\_dis[52] -0.30 0.01 0.24 -0.60 -0.30 0.02 1218 1.00  
## a\_dis[53] -0.42 0.01 0.36 -0.89 -0.43 0.03 1342 1.00  
## a\_dis[54] -0.79 0.01 0.45 -1.41 -0.76 -0.25 1142 1.00  
## a\_dis[55] 0.09 0.01 0.25 -0.23 0.10 0.42 1054 1.00  
## a\_dis[56] -1.05 0.01 0.37 -1.52 -1.03 -0.60 1172 1.00  
## a\_dis[57] -0.31 0.01 0.29 -0.67 -0.30 0.05 978 1.00  
## a\_dis[58] -1.01 0.02 0.46 -1.60 -0.99 -0.44 897 1.00  
## a\_dis[59] -0.99 0.01 0.33 -1.43 -0.98 -0.58 970 1.00  
## a\_dis[60] -1.04 0.01 0.29 -1.40 -1.03 -0.66 1224 1.00  
## alpha -0.54 0.00 0.09 -0.65 -0.53 -0.43 723 1.00  
## sigma 0.52 0.01 0.09 0.41 0.52 0.63 306 1.01  
## lp\_\_ -1286.56 0.50 7.45 -1295.92 -1286.42 -1276.91 219 1.01  
##   
## Samples were drawn using NUTS(diag\_e) at Thu Jun 6 03:21:03 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).

Model Comparision using WAIC

library(loo)  
ll12\_1 <- extract\_log\_lik(fit5.1)  
ll12\_2 <- extract\_log\_lik(fit5.2)  
reff1 <- relative\_eff(ll12\_1, chain\_id = c(rep(1, 500), rep(2, 500)), cores =2)  
reff2 <- relative\_eff(ll12\_2, chain\_id = c(rep(1, 500), rep(2, 500)), cores =2)  
waic5.1 <- waic(ll12\_1, r\_eff = reff1, cores = 2)  
waic5.2 <- waic(ll12\_2, r\_eff = reff2, cores = 2)  
loo::compare(waic5.1, waic5.2)

## elpd\_diff se   
## 8.5 5.9

c(waic5.1$waic,waic5.2$waic)

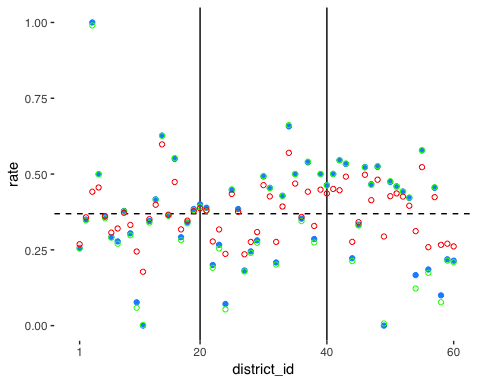
## [1] 2531.163 2514.082

Multilevel model performs better.

Plot

d.res <- d %>%   
 group\_by(district\_id) %>%   
 summarise(  
 cnt=n(),   
 ttl\_use\_c=sum(use\_contraception),  
 rate=ttl\_use\_c/cnt  
 ) %>%   
 as.data.frame() %>%  
 mutate(  
 d\_label = reorder(as.factor(paste0(district\_id,'/n=',cnt)), cnt)  
 ) %>%   
 arrange(district\_id)

post5.2 <- as.data.frame(fit5.2, pars = c('alpha', 'sigma', 'a\_dis'))  
post5.1 <- as.data.frame(fit5.1, pars = c('a\_dis'))  
d.res$pred5.1 <- post5.1 %>% colMeans %>% plogis  
d.res$pred5.2 <- post5.2[,-c(1:2)] %>% colMeans %>% plogis  
  
ggplot(d.res) +   
 geom\_vline(xintercept = c(20, 40)) +  
 geom\_hline(yintercept = plogis(mean(post5.2$alpha)), linetype = 'dashed') +  
 geom\_point(aes(x = district\_id, y = rate), colour = 'dodgerblue') +  
 geom\_point(aes(x = district\_id, y = pred5.1), shape = 21, colour = 'green') +  
 geom\_point(aes(x = district\_id, y = pred5.2), shape = 21, colour = 'red') +  
 scale\_x\_continuous(limits = c(0, 60), breaks = c(1, 20, 40, 60)) +  
 labs(x = 'district\_id', y = 'rate')



blue: actual data green: pooled model red: fixed model