Homework 6

Alex Ziyu Jiang

```
rm(list = ls())
library(rjags)
```

```
## Loading required package: coda

## Linked to JAGS 4.3.0

## Loaded modules: basemod,bugs
```

```
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
```

```
library(ggplot2)
library(rethinking)
```

```
## Loading required package: rstan

## Loading required package: StanHeaders

## rstan (Version 2.21.3, 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)

##
## Attaching package: 'rstan'

## The following object is masked from 'package:coda':
##
##     traceplot

## Loading required package: cmdstanr

## This is cmdstanr version 0.5.1.9000

## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr

## - CmdStan path: /Users/alexziyujiang/.cmdstan/cmdstan-2.29.2

## - CmdStan version: 2.29.2
## Loading required package: parallel
## rethinking (Version 2.21)
##
## Attaching package: 'rethinking'
## The following object is masked from 'package:rstan':
##
##     stan
## The following object is masked from 'package:stats':
##
##     rstudent
```

```
library(betareg)
```

Data Preprocessing

```
# read data
d <- read.csv(file = "qog_jan16.csv")

# transform variables
d$wdi_mortinftot <- d$wdi_mortinftot/1000
d$epi_watsup <- d$epi_watsup - mean(d$epi_watsup)

# y and x for jags
y <- d$wdi_mortinftot
x <- d$epi_watsup
```

Jags Code

```
linear_model_code <- "
  model{

   for(i in 1:n){
      # likelihood
      y[i] ~ dbeta(a[i], b[i])

      # conditional mean (logit link)
      logit(mu[i]) <- beta0 + beta1*x[i]

      # transform back to a and b
      a[i] <- mu[i] * phi
      b[i]  <- (1-mu[i]) * phi

      # compute partial derivative numerically
      ## plus 0.001
      logit(mup[i]) <- beta0 + beta1*(x[i]+0.001)
      ## minus 0.001
      logit(mum[i]) <- beta0 + beta1*(x[i]-0.001)
      ## difference
      apdx[i] <- ((mup[i] - mum[i])/(2*0.001))
      # posterior predictive
      ynew[i] ~ dbeta(a[i], b[i])
   }

    # priors
    beta0 ~ dnorm(0, 1)
    beta1 ~ dnorm(0, 1)
    phi ~ dunif(0, 500)
  }
"
```

Jags model compilation

```
# fit model, 4 chains
model <- jags.model(file = textConnection(linear_model_code),
                    data = list(x = x,
                                y = y,
                                n = length(y)), n.chains = 4)
```

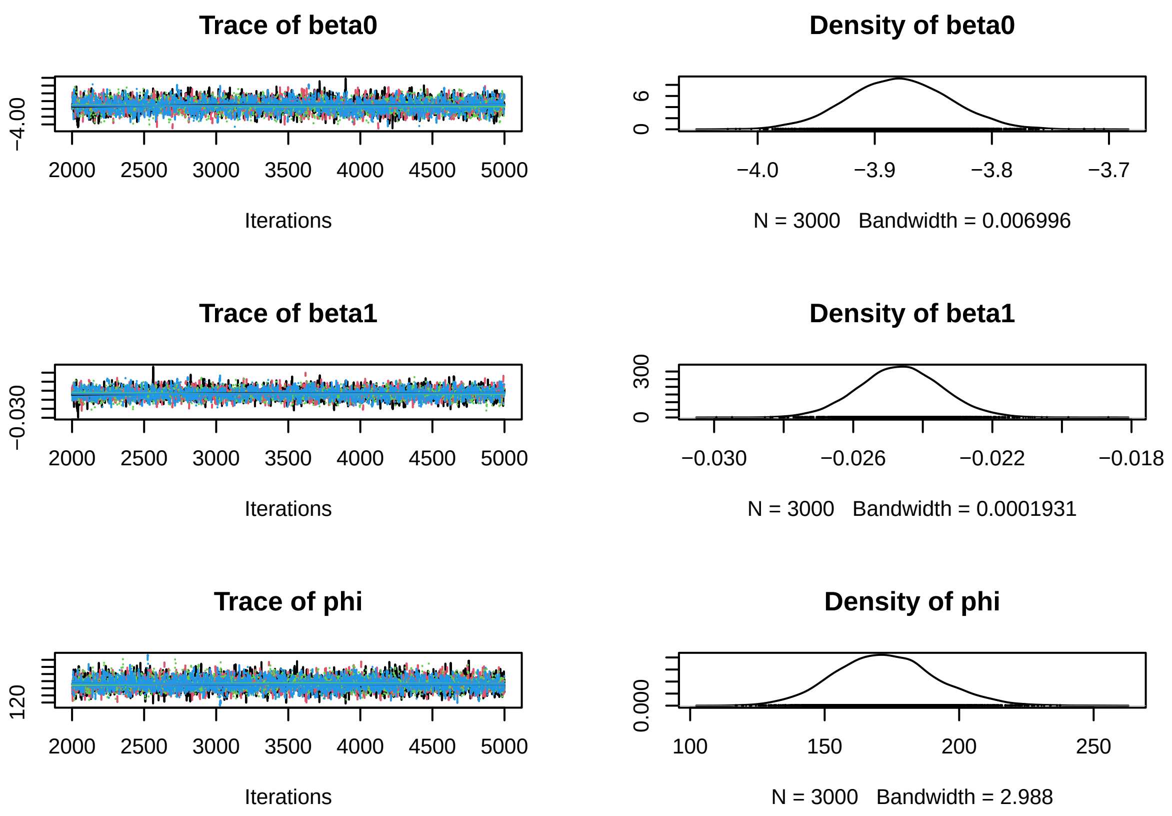
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 184
##    Unobserved stochastic nodes: 187
##    Total graph size: 3058
##
## Initializing model
```

```
# burn-in 1e3
update(model, n.iter = 1e3)

# sample betas + phi
betas <- coda.samples(model,
                      variable.names = c("beta0", "beta1", "phi"),
                      n.iter = 3e3)
```

model diagnostics

```
# check trace plot
plot(betas)
```



```
# effective sample size
effectiveSize(betas)
```

```
##    beta0    beta1      phi
## 3124.312 3572.105 4872.169
```

```
# Rhat
gelman.diag(betas)
```

```
## Potential scale reduction factors:
##
##       Point est. Upper C.I.
## beta0          1       1.01
## beta1          1       1.01
## phi            1       1.00
##
## Multivariate psrf
##
## 1
```

model results

```
# posterior mean + credible intervals
precis(as.data.frame(betas[[1]]), digits = 4, prob = .95) %>% data.frame %>% dplyr::select(c(-histogram
```

```
##               mean           sd        X2.5.       X97.5.
## beta0  -3.87687146  0.043187710  -3.96167833  -3.79340665
## beta1  -0.02460657  0.001181068  -0.02696522  -0.02223647
## phi   172.49675252 19.161328668 137.46313917 212.38878593
```

```r
# compare with betareg
breg <- betareg(wdi_mortinftot ~ epi_watsup, data = d)
summary(breg)
```

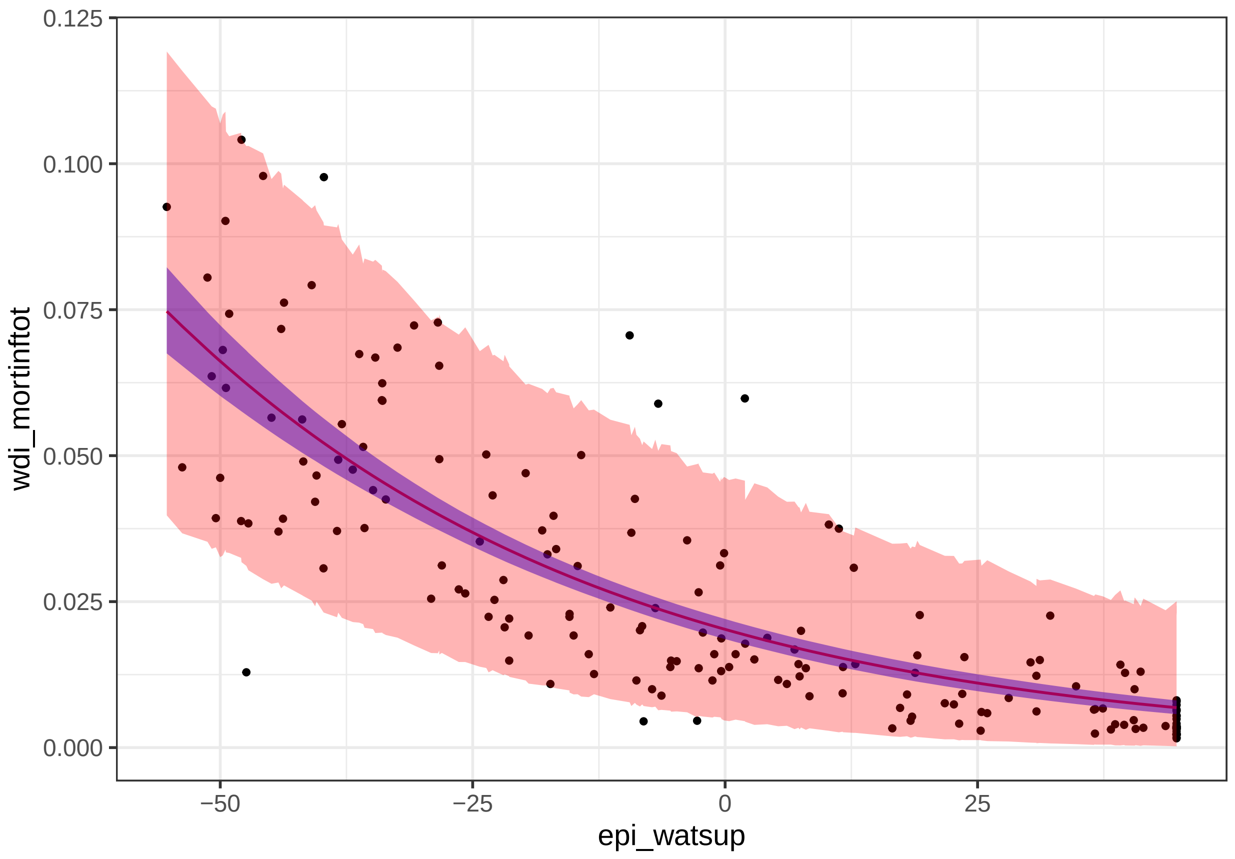
```
##
## Call:
## betareg(formula = wdi_mortinftot ~ epi_watsup, data = d)
##
## Standardized weighted residuals 2:
##     Min      1Q  Median      3Q     Max
## -4.9881 -0.5567 -0.0691  0.6217  2.3326
##
## Coefficients (mean model with logit link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.885477   0.043278  -89.78   <2e-16 ***
## epi_watsup  -0.024754   0.001207  -20.51   <2e-16 ***
##
## Phi coefficients (precision model with identity link):
##       Estimate Std. Error z value Pr(>|z|)
## (phi)   173.67      18.71   9.281   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Type of estimator: ML (maximum likelihood)
## Log-likelihood:   603 on 3 Df
## Pseudo R-squared: 0.7595
## Number of iterations: 85 (BFGS) + 2 (Fisher scoring)
```

posterior predictive checks

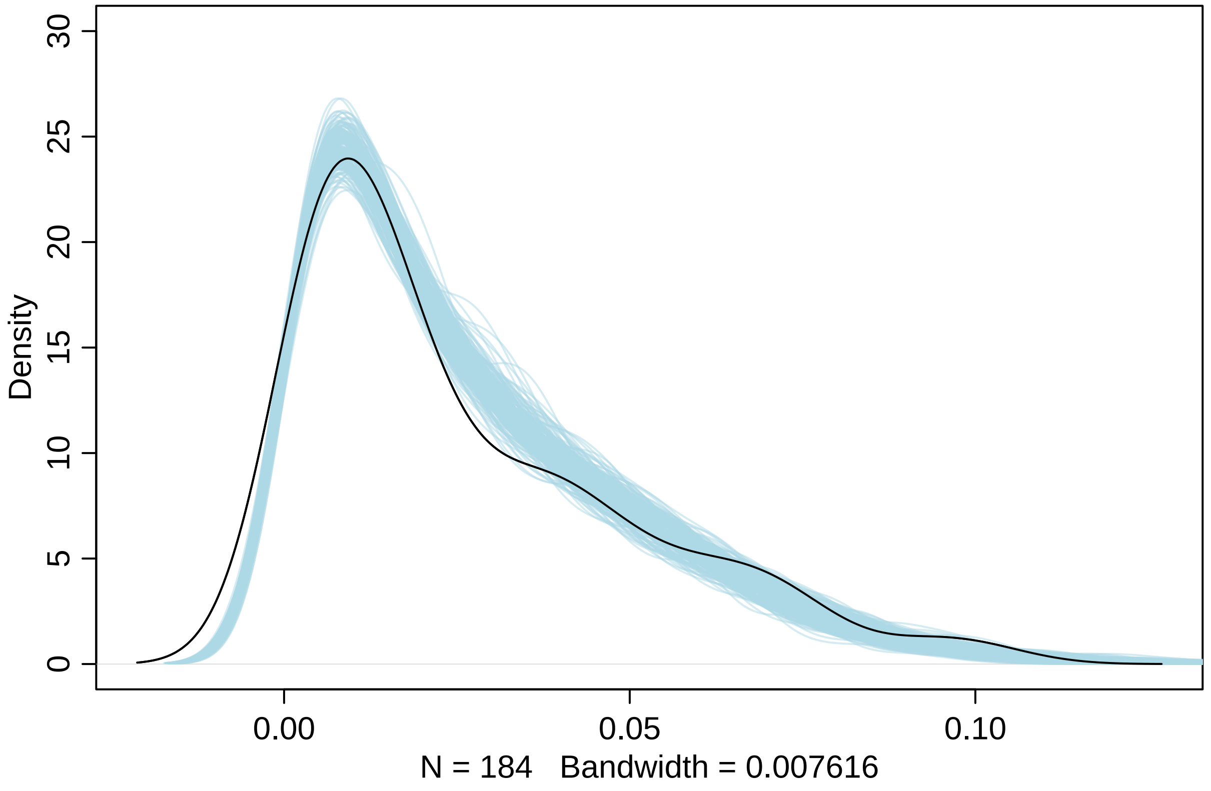
```r
# sample mu and ynew
ymu.samples <- jags.samples(model,
                            variable.names = c("mu", "ynew"),
                            n.iter = 1e3)

# add posterior mean and quantiles to data.frame
d$mu.mean   <-  apply(ymu.samples$mu,  1, mean)
d$mu.lwr    <-  apply(ymu.samples$mu, 1, quantile, .025)
d$mu.upr    <-  apply(ymu.samples$mu, 1, quantile, .975)
d$ynew.lwr <-  apply(ymu.samples$ynew,1, quantile, .025)
d$ynew.upr <-  apply(ymu.samples$ynew,1, quantile, .975)

# ggplot
ggplot(d, aes(y = wdi_mortinftot, x = epi_watsup))  +
  geom_point(col = alpha("black", 1), cex = .8) +
  geom_line(aes(y = mu.mean), col = "red") +
  geom_ribbon(aes(ymin = mu.lwr, ymax = mu.upr), fill = alpha("blue", .5)) +
  geom_ribbon(aes(ymin = ynew.lwr, ymax = ynew.upr), fill = alpha("red", .3))  +
  theme_bw()
```



```r
# density overlay
dens(y, ylim = c(0, 30), adj = 1, col = "black")
for(i in 1:200){
  dens(ymu.samples$ynew[,i,], adj = 1, col = alpha("lightblue", alpha = 0.5), add = T)
}
dens(y, adj = 1, col = "black", add = T)
```



APD

```
# apd
apdx <- coda.samples(model, "apdx", n.iter = 1e3)
apdx <- do.call("rbind", apdx)

library(gtools)
```

```
##
## Attaching package: 'gtools'

## The following object is masked from 'package:rethinking':
##
##     logit
```

```
bb_weights <- rdirichlet(nrow(apdx), alpha = rep(1, ncol(apdx)))
apd <- apply(bb_weights*apdx, 1, sum)
apd <- apd*1000
precis(apd)  %>% data.frame %>% dplyr::select(c(-histogram))
```

```
##           mean         sd      X5.5.     X94.5.
## apd -0.6280232 0.04891955 -0.7102447 -0.5531013
```