Worksheet 5 - Exercise 4

Foundations of Bayesian Methodology

Wenjie Tu

Spring Semester 2022

Setup

```
# original covariate values
x <- c(0.0028, 0.0028, 0.0056, 0.0112, 0.0225, 0.0450)

# the centered covariate values (centered dose) from the Mice data from Collett
x_centered <- x - mean(x)

# number of mice deaths
y <- c(26, 9, 21, 9, 6, 1)

# total number of mice
n <- c(28, 12, 40, 40, 40, 40)
```

```
d.mice <- data.frame(
  x, y, n, x_centered, y/n, n-y
)
colnames(d.mice) <- c("$x$", "$y$", "$n$", "centered $x$", "$p$", "$alive$")
knitr::kable(d.mice, align="c", caption="Mice data from Collett (2003)")
```

Table 1: Mice data from Collett (2003)

x	y	n	centered x	p	*alive*
0.0028	26	28	-0.0121833	0.9285714	2
0.0028	9	12	-0.0121833	0.7500000	3
0.0056	21	40	-0.0093833	0.5250000	19
0.0112	9	40	-0.0037833	0.2250000	31
0.0225	6	40	0.0075167	0.1500000	34
0.0450	1	40	0.0300167	0.0250000	39

Logistic model:

$$\text{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta x_i$$

$$p_i = \frac{\exp(\alpha + \beta x_i)}{1 + \exp(\alpha + \beta x_i)}$$

Classic Approach

```
fit.classic <- glm(cbind(y, (n-y)) ~ x_centered, data = d.mice, family = binomial)
summary(fit.classic)
```

```
##
## Call:
## glm(formula = cbind(y, (n - y)) ~ x_centered, family = binomial,
##     data = d.mice)
##
## Deviance Residuals:
##       1        2        3        4        5        6
##  3.0784   0.4474  -0.9319  -2.2893   0.7546   1.3344
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.9800     0.2399  -4.085 4.41e-05 ***
## x_centered  -146.6927    26.3630  -5.564 2.63e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 92.287  on 5  degrees of freedom
## Residual deviance: 18.136  on 4  degrees of freedom
## AIC: 40.805
##
## Number of Fisher Scoring iterations: 5
```

```
knitr::kable(coef(summary(fit.classic)), align="c", caption="Summary results for classic approach")
```

Table 2: Summary results for classic approach

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-0.9800475	0.2399331	-4.084669	4.41e-05
x_centered	-146.6927209	26.3629619	-5.564349	0.00e+00

```
## Disaggregate the data
d.mice1 <- data.frame(y_binary=rep(c(1, 0), c(sum(y), sum(n)-sum(y))),
                      x_centered=c(rep(round(x_centered, 5), y),
                                   rep(round(x_centered, 5), n-y)))

knitr::kable(table(d.mice1))
```

	-0.01218	-0.00938	-0.00378	0.00752	0.03002
0	5	19	31	34	39
1	35	21	9	6	1

```
fit.glm <- glm(y_binary ~ x_centered, data = d.mice1, family = binomial)
summary(fit.glm)
```

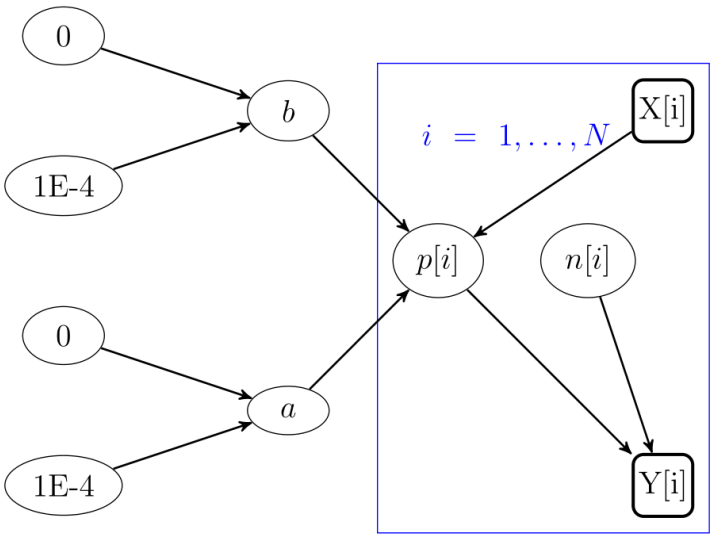
```
##
## Call:
## glm(formula = y_binary ~ x_centered, family = binomial, data = d.mice1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5337  -1.0030  -0.0957   0.8589   3.2826
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.9796     0.2399  -4.084 4.43e-05 ***
## x_centered  -146.6927    26.3629  -5.564 2.63e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 261.37  on 199  degrees of freedom
## Residual deviance: 187.22  on 198  degrees of freedom
## AIC: 191.22
##
## Number of Fisher Scoring iterations: 6
```

Bayesian Approach

```
library(rjags)
library(coda)
library(ggplot2)
```

```
modelString <- "model{
  for (i in 1:length(y)) {
    y[i] ~ dbin(p[i],n[i])
    p[i] <- ilogit(alpha + beta * x[i])
  }

  alpha ~ dnorm(0, 1.0E-04)
  beta ~ dnorm(0, 1.0E-04)
}"

writeLines(modelString, con="LogitModel.txt")
```

```
## Alternatively
modelString <- modelString <- "model{
  for (i in 1:length(y)) {
    y[i] ~ dbern(p[i])
    p[i] <- ilogit(alpha + beta * x[i])
  }

  alpha ~ dnorm(0, 1.0E-04)
  beta ~ dnorm(0, 1.0E-04)
}"

writeLines(modelString, con="LogitModel.txt")
```




```
## Set seed for reproducible results
set.seed(44566)

## Generate initial values based on estimates in classical logistic regression
inits.alpha <- coef(summary(fit.classic))[1, 1] +
  coef(summary(fit.classic))[1, 2] * rnorm(4)

inits.beta <- coef(summary(fit.classic))[2, 1] +
  coef(summary(fit.classic))[2, 2] * rnorm(4)
```

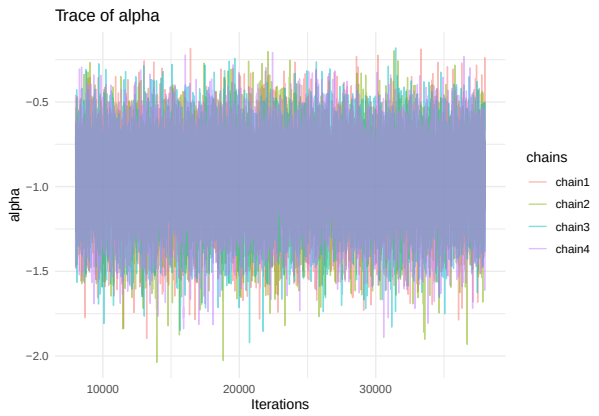
```
## Generate data list for JAGS
dat.jags <- list(y=y, x=x_centered, n=n)

## Set initial values and random seed for reproducible results
inits.jags <- list(list(alpha=inits.alpha[1], beta=inits.beta[1],
                        .RNG.name="base::Wichmann-Hill", .RNG.seed=314159),
                   list(alpha=inits.alpha[2], beta=inits.beta[2],
                        .RNG.name="base::Marsaglia-Multicarry", .RNG.seed=159314),
                   list(alpha=inits.alpha[3], beta=inits.beta[3],
                        .RNG.name="base::Super-Duper", .RNG.seed=413159),
                   list(alpha=inits.alpha[4], beta=inits.beta[4],
                        .RNG.name="base::Mersenne-Twister", .RNG.seed=143915))

## Compile JAGS model
model.jags <- jags.model(
  file = "LogitModel.txt",
  data = dat.jags,
  inits = inits.jags,
  n.chains = 4,
  n.adapt = 4000
)
```

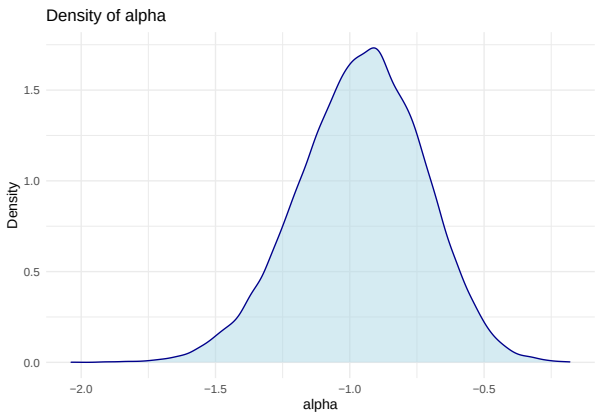
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 6
##    Unobserved stochastic nodes: 2
```

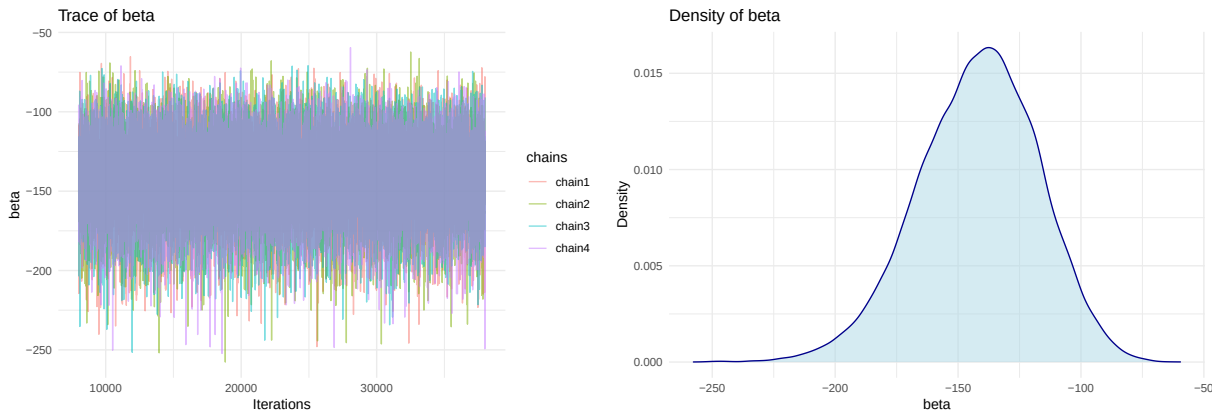
```
##    Total graph size: 37
##
## Initializing model
```

```
## Burn-in
update(model.jags, n.iter = 4000)

## Sampling
fit.bayesian <- coda.samples(
  model = model.jags,
  variable.names = c("alpha", "beta"),
  n.iter = 30000,
  thin = 3
)
```

```
m.fit.bayesian <-as.matrix(fit.bayesian)
d.chains <- data.frame(
  iterations = rep(seq(8003, 38000, by=3), times=4),
  alpha = m.fit.bayesian[, "alpha"],
  beta = m.fit.bayesian[, "beta"],
  chains = rep(c("chain1", "chain2", "chain3", "chain4"), each=10000)
)

ggplot(d.chains, aes(x=iterations, y=alpha, color=chains)) + geom_line(alpha=0.5) +
  labs(title="Trace of alpha", x="Iterations") + theme_minimal()

ggplot(d.chains, aes(x=alpha, y=..density..)) +
  geom_density(color="darkblue", fill="lightblue", alpha=0.5) +
  labs(title="Density of alpha", y="Density") + theme_minimal()

ggplot(d.chains, aes(x=iterations, y=beta, color=chains)) + geom_line(alpha=0.5) +
  labs(title="Trace of beta", x="Iterations") + theme_minimal()

ggplot(d.chains, aes(x=beta, y=..density..)) +
  geom_density(color="darkblue", fill="lightblue", alpha=0.5) +
  labs(title="Density of beta", y="Density") + theme_minimal()
```





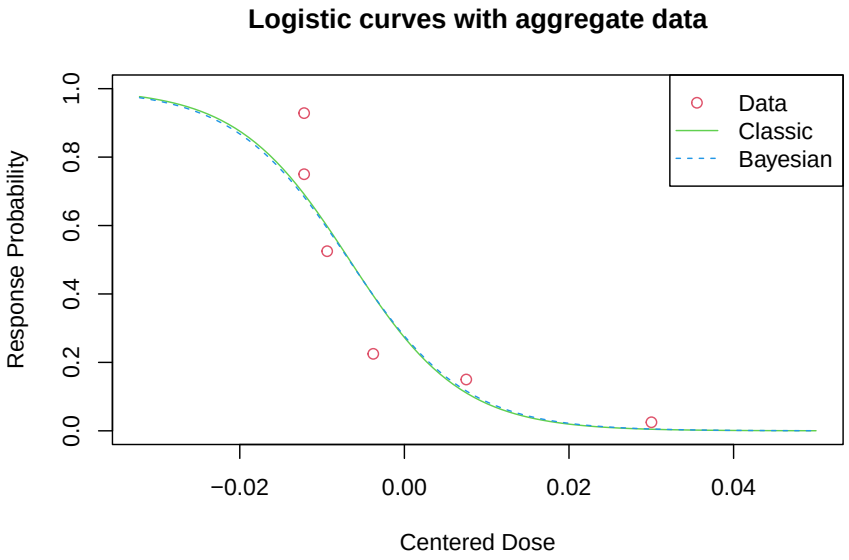



```
d.summary <- t(rbind(
  colMeans(m.fit.bayesian),
  apply(m.fit.bayesian, 2, function(x) sd(x)),
  apply(m.fit.bayesian, 2, function(x) quantile(x, probs=c(0.025, 0.5, 0.975)))
))

colnames(d.summary) <- c("Mean", "SD", "2.5%", "Median", "97.5%")
knitr::kable(d.summary, align="c", caption="Summary results for Bayesian approach")
```

Table 4: Summary results for Bayesian approach

	Mean	SD	2.5%	Median	97.5%
alpha	-0.9588435	0.2320144	-1.438487	-0.9498307	-0.5303068
beta	-142.0166273	24.5390488	-192.955187	-140.7837071	-97.0036504

```
## Inverse logit function
ilogit <- function(alpha, beta, x) {
  tmp <- exp(alpha + beta * x)
  pi <- tmp / (1 + tmp)
  return(pi)
}

## Extract estimates from classic and Bayesian models
alpha.classic <- coef(summary(fit.classic))[1, 1]
beta.classic <- coef(summary(fit.classic))[2, 1]
alpha.bayesian <- d.summary[1, 1]
beta.bayesian <- d.summary[2, 1]

x.grid <- seq(min(x_centered)-0.02, max(x_centered)+0.02, length.out=100)
y.pred.classic <- ilogit(alpha=alpha.classic, beta=beta.classic, x=x.grid)
y.pred.bayesian <- ilogit(alpha=alpha.bayesian, beta=beta.bayesian, x=x.grid)

plot(y.pred.classic ~ x.grid, col=3, type="l", ylim=c(0, 1), xlab="Centered Dose",
     ylab="Response Probability", main="Logistic curves with aggregate data")
lines(y.pred.bayesian ~ x.grid, col=4, lty=2)
points(x=x_centered, y=y/n, col=2)
legend("topright", legend=c("Data", "Classic", "Bayesian"),
       col=2:4, lty=c(NA, 1, 2), pch=c(1, NA, NA))
```


Logistic curves with aggregate data
Response Probability
0.0
0.2
0.4
0.6
0.8
1.0
−0.02
0.00
0.02
0.04
Centered Dose
Data
Classic
Bayesian