# HW10_MissingDataAndMarkRecapture

Nathan Barrus

2024-04-04

## Purpose:

The purpose of this markdown document is to work through Homework 10 in Dr. Babcock's Bayesian Statistics Course at the University of Miami. Homework 10 deals with missing data problems and mark recapture models.

## General Start to Code

```
rm(list = ls())

######github#####
#note, only needed after 90 days from 1/16/2024

#  usethis::create_github_token()
#  gitcreds::gitcreds_set()

#####check for r updates#####
#note, updateing may take some time so plan accordingly

#require(installr)

#check.for.updates.R()

#updateR() #only if needed

#######check for package updates#####
#note, updateing may take some time so plan accordingly

#old.packages()

# update.packages() #make the decision to the update the packages
```

## Load packages

```
library(INLAutils)
library(INLA)
library(tidyverse)
```

```
library(R2jags)
library(rstan)
library(ggmcmc)
library(purrr)
library(magrittr)
library(here)
library(loo)
library(DHARMa)
library(lme4)
library(rstanarm)
library(shinystan)
library(BayesFactor)

theme_set(theme_bw(base_size=15))
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
```

## Data

For problem 1) the data represent capture/recapture histories for 27 birds

For problem 2) The data called swoPres.csv is presence or absence of swordfish in a fishery over time, with some data missing.

For problem 3) The data called pemax is a dataset with a y variable called pemax (maximum lung pressure), and predictor variables called weight, masspercent, expirevolume and residvolume. This is based on the cystic fibrosis data set in the ISwR library. The idea is to predict pemax from a multiple regression of the other measured variables for each patient. Some of the weight data is missing. All predictor variables have already been standardized.

```
#problem 1 data

markrecap.data <- read_csv(here("data", "dat10.1.csv"))
```

```
## Rows: 27 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): Y1, Y2, Y3, Y4, First
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(markrecap.data)

#problem 2 data

sword.data <- read_csv(here("data","swoPres.csv"))
```

```
## Rows: 500 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): Year, Presence, hbf
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(sword.data)

#problem 3 data

pemax.data <- read_csv(here("data","pemax.csv"))
```

```
## Rows: 25 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): pemax, weight, masspercent, expirevolume, residvolume
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(pemax.data)
```

## Problem 1) State space modeling of capture recapture histories.

**A) Using the data in dat10.1 and JAGS, estimate the probability of survival and the probability of being resighted, assuming that there is no variation between individuals or years in either parameter. See McCarthy box 7.1. Give the summary statistics and plot the densities.** *write and run model*

```
#create data in list format for jags

bird <- as.matrix(markrecap.data[,1:4])
bird

markrecap.ls <- list(Y = bird, Years = 4, N = length(markrecap.data$Y1),
                     First = markrecap.data$First)

#write the model

write("model
{
  for (i in 1:N)  # for each bird
  {
    alive[i, First[i]] <- 1  # 1 means it is alive the first time it was seen
    for (j in (First[i]+1):Years)  # for each year after the first
    {
#  palive = prob of remaining alive (=0 if previously dead, =surv if alive)
      palive[i,j] <- surv * alive[i, j-1]
# determine whether it is alive
      alive[i,j] ~ dbern(palive[i,j])
# probability of resighting depends on whether it is alive
      psight[i,j] <- resight * alive[i, j]
# actual resighting determined randomly
      Y[i, j] ~ dbern(psight[i,j])
```

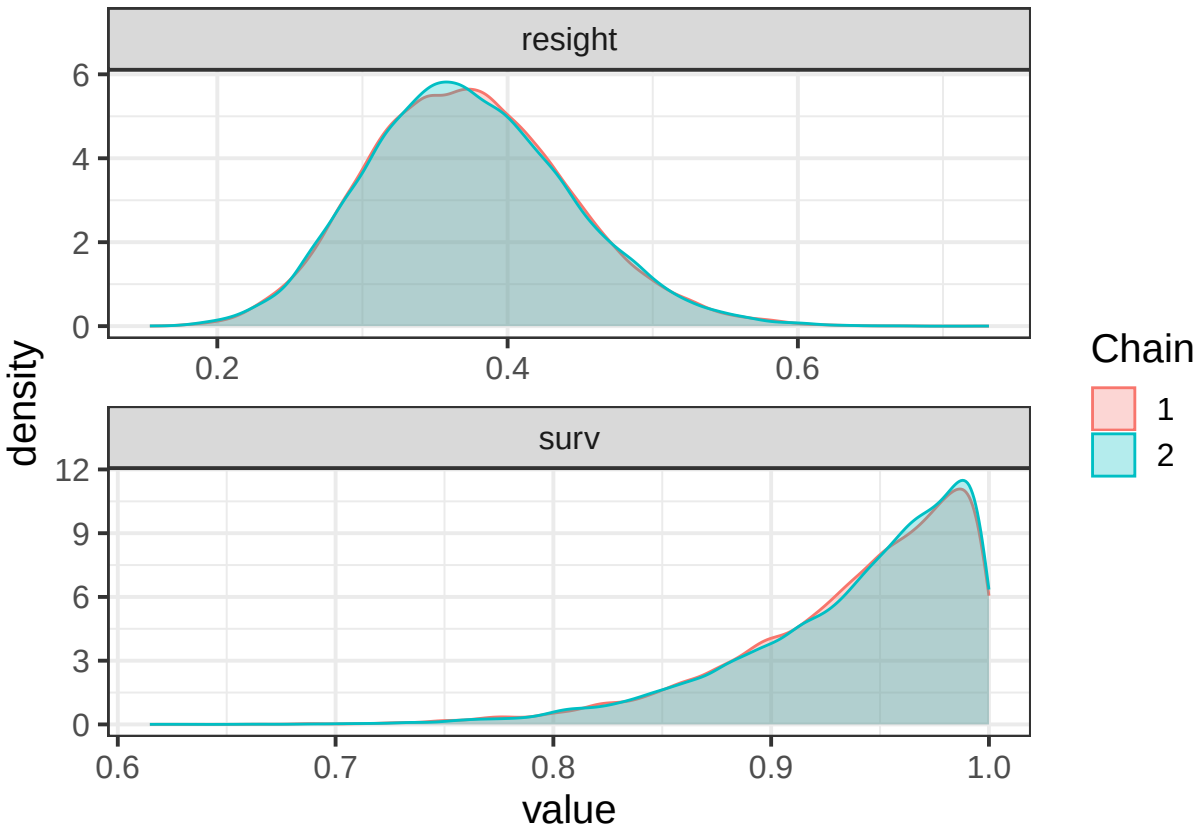
```
    }
   }
 # Uninformative priors for survival and resighting rates
  surv ~ dunif(0, 1)  # uninformative
  resight ~ dunif(0,1)  # resighting rate - uninformative
}",file=here("JAGS_mods","HW10_P1-A_markrecap.txt"))


#set up alive initialization

alive<-matrix(1,markrecap.ls$N,markrecap.ls$Years)

for(i in 1:markrecap.ls$Years){
 alive[,i][i<=markrecap.ls$First]=NA
}

alive[1:10,]  #To check it worked

init1<-list(surv=0.5,resight=0.5,alive=alive)
init2<-list(surv=0.8,resight=0.8,alive=alive)

#run the model

birds_markrecap_jags <- jags(markrecap.ls,list(init1,init2),
                parameters.to.save=c("surv","resight"),
                model.file=here("JAGS_mods","HW10_P1-A_markrecap.txt"),
                n.chains=2,n.iter=200000,n.burnin=10000,n.thin=10)
```

*summary results and plots*

```
round(birds_markrecap_jags$BUGSoutput$summary[c("surv","resight"),],2)
```

```
##         mean   sd 2.5%  25%  50%  75% 97.5% Rhat n.eff
## surv    0.94 0.05 0.81 0.91 0.95 0.98  1.00    1  7500
## resight 0.37 0.07 0.25 0.32 0.37 0.42  0.52    1 38000
```

```
bird1ggs<-ggs(as.mcmc(birds_markrecap_jags))
bird1ggs<-bird1ggs[bird1ggs$Parameter %in% c("surv","resight"),]
ggs_density(bird1ggs)
```




```
#write the model

write("model
{

  for (i in 1:N)  # for each bird
  {
    alive[i, First[i]] <- 1  # 1 means it is alive the first time it was seen
    for (j in (First[i]+1):Years)  # for each year after the first
    {
#  palive = prob of remaining alive (=0 if previously dead, =surv if alive)
      palive[i,j] <- surv[j-1] * alive[i, j-1]
# determine whether it is alive
      alive[i,j] ~ dbern(palive[i,j])
# probability of resighting depends on whether it is alive
      psight[i,j] <- resight * alive[i, j]
# actual resighting determined randomly
      Y[i, j] ~ dbern(psight[i,j])
     }
   }

 # Uninformative priors for survival and resighting rates
```

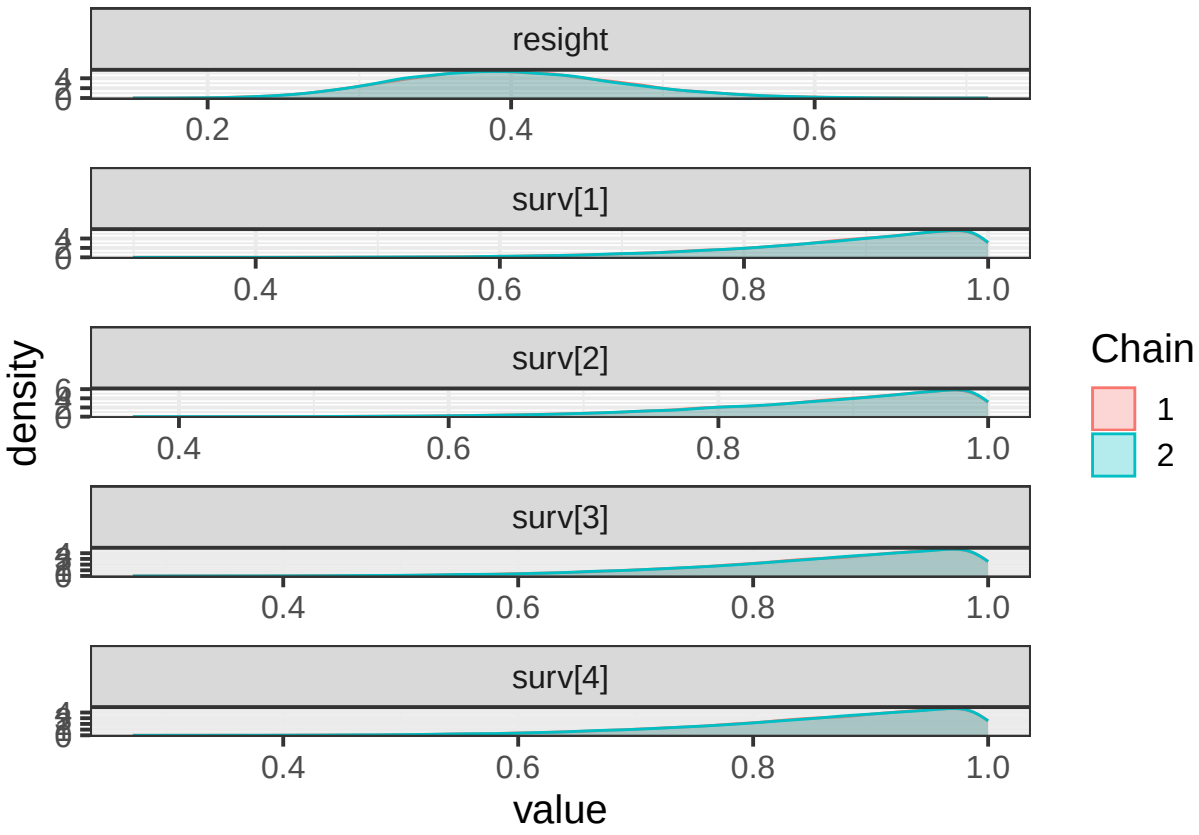
```
  surv[1] ~ dunif(0, 1)  # uninformative for y1
  surv[2] ~ dunif(0, 1)  # uninformative for y2
  surv[3] ~ dunif(0, 1)  # uninformative for y3
  surv[4] <-  surv[3]    # y4 unestimateable make same as y3
  resight ~ dunif(0,1)  # resighting rate - uninformative
}",file=here("JAGS_mods","HW10_P1-A_markrecap_2.txt"))


#set up alive initialization

alive<-matrix(1,markrecap.ls$N,markrecap.ls$Years)

for(i in 1:markrecap.ls$Years){
 alive[,i][i<=markrecap.ls$First]=NA
}

alive[1:10,]  #To check it worked

init1<-list(surv=rep(0.5,times =4),resight=0.5,alive=alive)
init2<-list(surv=rep(0.8,times =4),resight=0.8,alive=alive)

#run the model

birds_markrecap_jags_2 <- jags(markrecap.ls,list(init1,init2),
                parameters.to.save=c("surv","resight"),
                model.file=here("JAGS_mods","HW10_P1-A_markrecap_2.txt"),
                n.chains=2,n.iter=200000,n.burnin=10000,n.thin=10)
```

```
round(birds_markrecap_jags_2$BUGSoutput$summary[c("surv[1]","surv[2]","surv[3]","surv[4]","resight"),],1
```

**B) Now modify the code to allow the survival probability to have different values in each year (year 1- year 3). Assume that year 3 and year 4 have the same probability of survival, since survival in year for 4 is not estimable. Use a uniform(0,1) prior on survival in each year. Keep the probability of resighting the same across years. What are the survival probabilities and the probability of resighting? Give summary statistics and plot the densities.**

```
##         mean   sd 2.5%  25%  50%  75% 97.5% Rhat n.eff
## surv[1] 0.88 0.10 0.64 0.83 0.91 0.96  1.00    1 38000
## surv[2] 0.88 0.09 0.65 0.83 0.91 0.96  1.00    1 38000
## surv[3] 0.86 0.11 0.59 0.80 0.89 0.95  1.00    1 38000
## surv[4] 0.86 0.11 0.59 0.80 0.89 0.95  1.00    1 38000
## resight 0.40 0.07 0.27 0.35 0.40 0.45  0.55    1 34000
```

```
bird2ggs<-ggs(as.mcmc(birds_markrecap_jags_2))
bird2ggs<-bird2ggs[bird2ggs$Parameter %in% c("surv[1]","surv[2]","surv[3]","surv[4]","resight"),]
ggs_density(bird2ggs)
```




```
#write the model

write("model
{

#hyper priors
sigma.resight ~ dunif(0,1)
p.sight ~ dunif(0,1)
lp.sight <- log(p.sight/(1-p.sight))
tau.resight <- 1 / (sigma.resight * sigma.resight)

#individual variation in sighting
  for(i in 1:N) {
  eta[i] ~ dnorm(0,tau.resight)
  logit(resight[i]) <- lp.sight + eta[i]
  }


  for (i in 1:N)  # for each bird
  {
    alive[i, First[i]] <- 1  # 1 means it is alive the first time it was seen
    for (j in (First[i]+1):Years)  # for each year after the first
    {
```

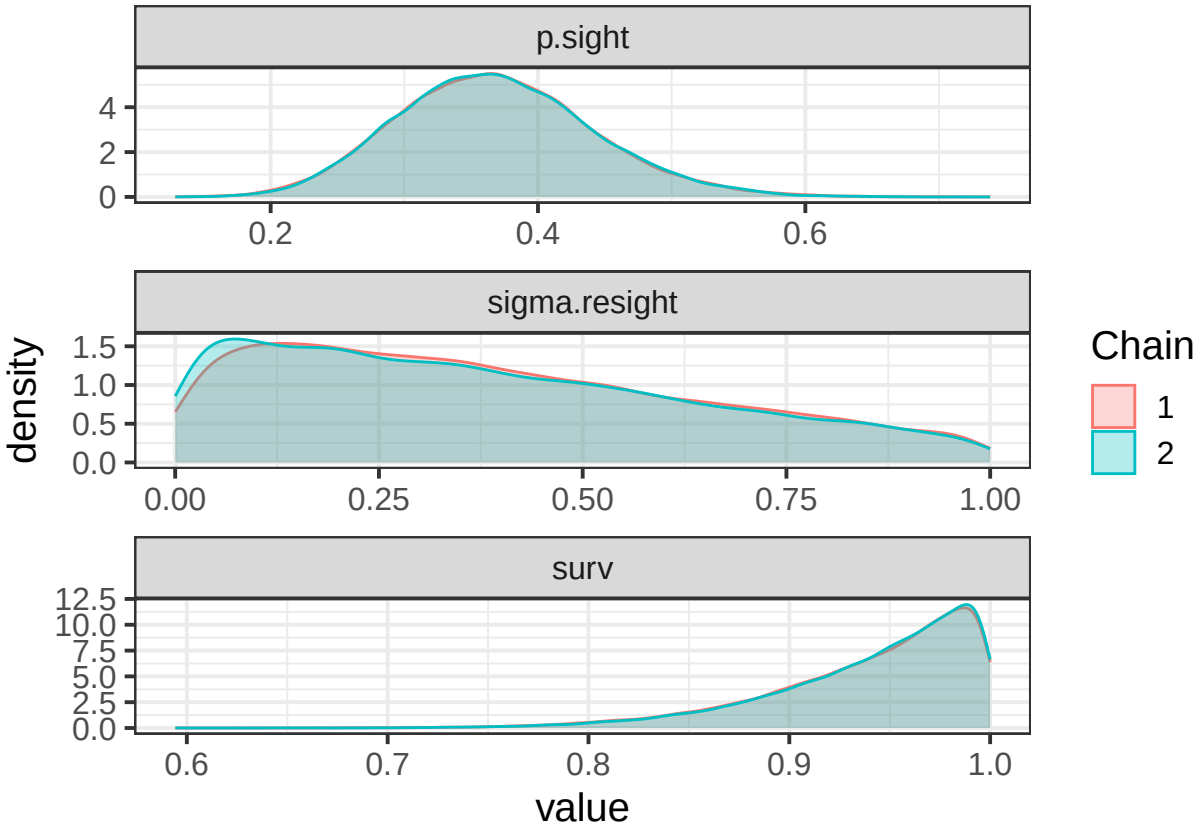
```
#  palive = prob of remaining alive (=0 if previously dead, =surv if alive)
      palive[i,j] <- surv * alive[i, j-1]
# determine whether it is alive
      alive[i,j] ~ dbern(palive[i,j])
# probability of resighting depends on whether it is alive
      psight[i,j] <- resight[i] * alive[i, j]
# actual resighting determined randomly
      Y[i, j] ~ dbern(psight[i,j])
     }
   }

 # Uninformative priors for survival and resighting rates

  surv ~ dunif(0,1)

}",file=here("JAGS_mods","HW10_P1-A_markrecap_3.txt"))


#set up alive initialization

alive<-matrix(1,markrecap.ls$N,markrecap.ls$Years)

for(i in 1:markrecap.ls$Years){
 alive[,i][i<=markrecap.ls$First]=NA
}

alive[1:10,]  #To check it worked

init1<-list(surv=0.5,p.sight = 0.5,sigma.resight=1,alive=alive)
init2<-list(surv=0.8,p.sight = 0.8,sigma.resight=0.5,alive=alive)

#run the model

birds_markrecap_jags_3 <- jags(markrecap.ls,list(init1,init2),
                parameters.to.save=c("surv","resight","p.sight","tau.resight","sigma.resight"),
                model.file=here("JAGS_mods","HW10_P1-A_markrecap_3.txt"),
                n.chains=2,n.iter=200000,n.burnin=10000,n.thin=10)
```

**C) Now add individual variation in sighting probability only. Use uninformative priors for the mean and variance of the normally-distributed individual variation in the logit of the resighting probability. Give summary statistics and plot the densities of the hyperparameters. Also, plot the individual variation in resighting probability (ggs_caterpiller is fine). How much do the individual detection probabilities vary?** *summary and density of surv and hyper parameters*

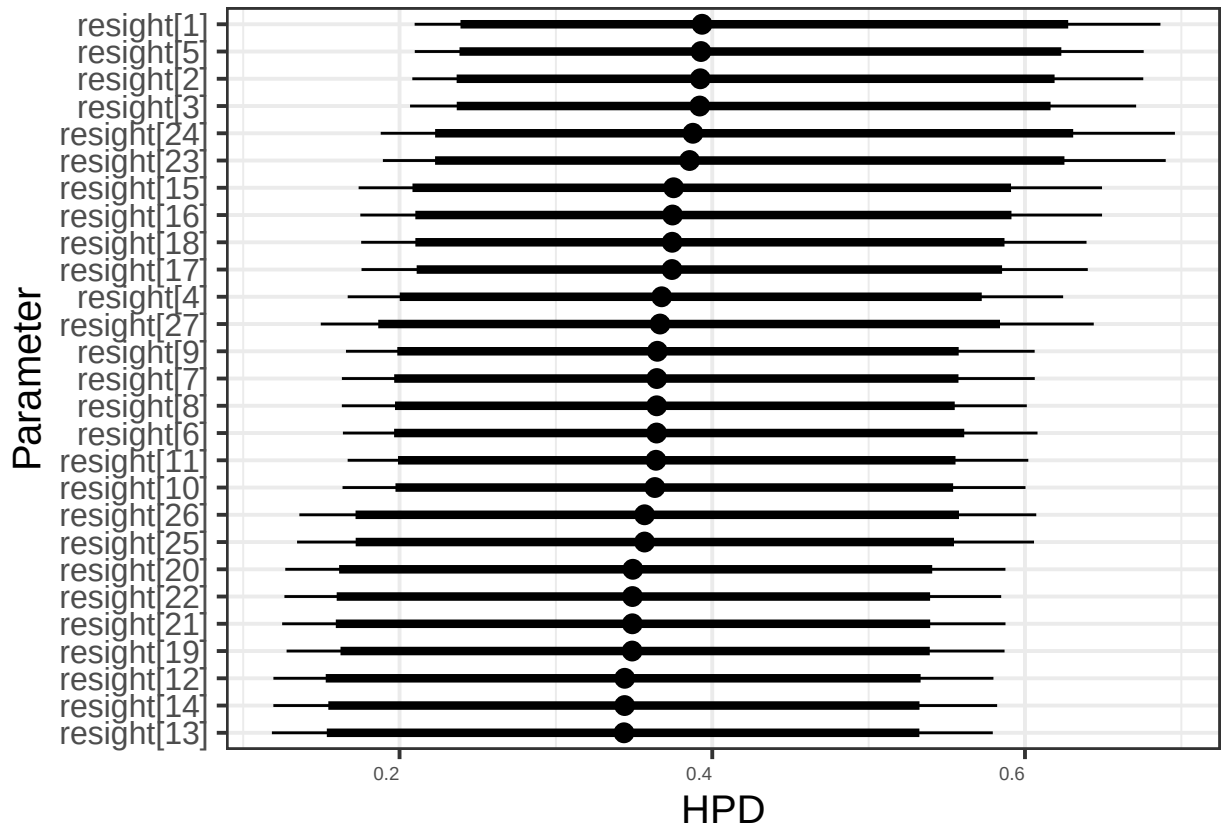
```
round(birds_markrecap_jags_3$BUGSoutput$summary[c("p.sight","sigma.resight", "surv"),],2)
```

```
##                mean   sd 2.5%  25%  50%  75% 97.5% Rhat n.eff
## p.sight        0.37 0.07 0.24 0.32 0.37 0.42  0.52 1.00 38000
## sigma.resight 0.38 0.26 0.02 0.16 0.34 0.57  0.93 1.01   730
## surv           0.94 0.05 0.81 0.91 0.95 0.98  1.00 1.00  5900
```

```
bird3ggs<-ggs(as.mcmc(birds_markrecap_jags_3))
bird3ggs<-bird3ggs[bird3ggs$Parameter %in% c("p.sight","sigma.resight", "surv"),]
ggs_density(bird3ggs)
```




```
#caterpillar plot

bird3ggs<-ggs(as.mcmc(birds_markrecap_jags_3))

bird3ggs |>
  filter(Parameter != "tau.resight") |>
  filter(Parameter != "deviance") |>
  filter(Parameter != "surv") |>
  filter(Parameter != "sigma.resight") |>
  filter(Parameter != "p.sight") |>
  ggs_caterpillar()
```




It doesn't seem like individual detection probability varies that much.

```
DIC.table <- tibble(model = c("Surv()Resight()", "Surv(Year)Resight()", "Surv()Resight(ind)"),
                    model.ls = c(list(birds_markrecap_jags), list(birds_markrecap_jags_2),list(birds_ma
  mutate(DIC = map_dbl(model.ls, c(2,24)),
         pD = map_dbl(model.ls, c(2,23)),
         deltaDIC = DIC - min(DIC),
         weight = round(exp(-2*deltaDIC)/sum(exp(-2*deltaDIC)),digits = 5))

DIC.table |> select(-model.ls) |> knitr::kable(caption = "DIC Table")
```

**D) Show the DIC values for the three models. Which one best represents the data?**

Table 1: DIC Table

| model | DIC | pD | deltaDIC | weight |
|---|---|---|---|---|
| Surv()Resight() | 85.6 | 9.62 | 2.47 | 0.007 |
| Surv(Year)Resight() | 83.1 | 10.84 | 0.00 | 0.992 |
| Surv()Resight(ind) | 86.5 | 11.08 | 3.32 | 0.001 |

The model with year effects best represents the data.

## Problem 2) Missing Y Data

We will fit a logistic regression to this model, with numerical year as an x variable, along with a numerical variable called hbf (hooks between floats).
logit(p)=a+b1 $Year$+$b2$hbf

**A) Fit the model using only the records where the data are not missing, which you can get with this R code:**

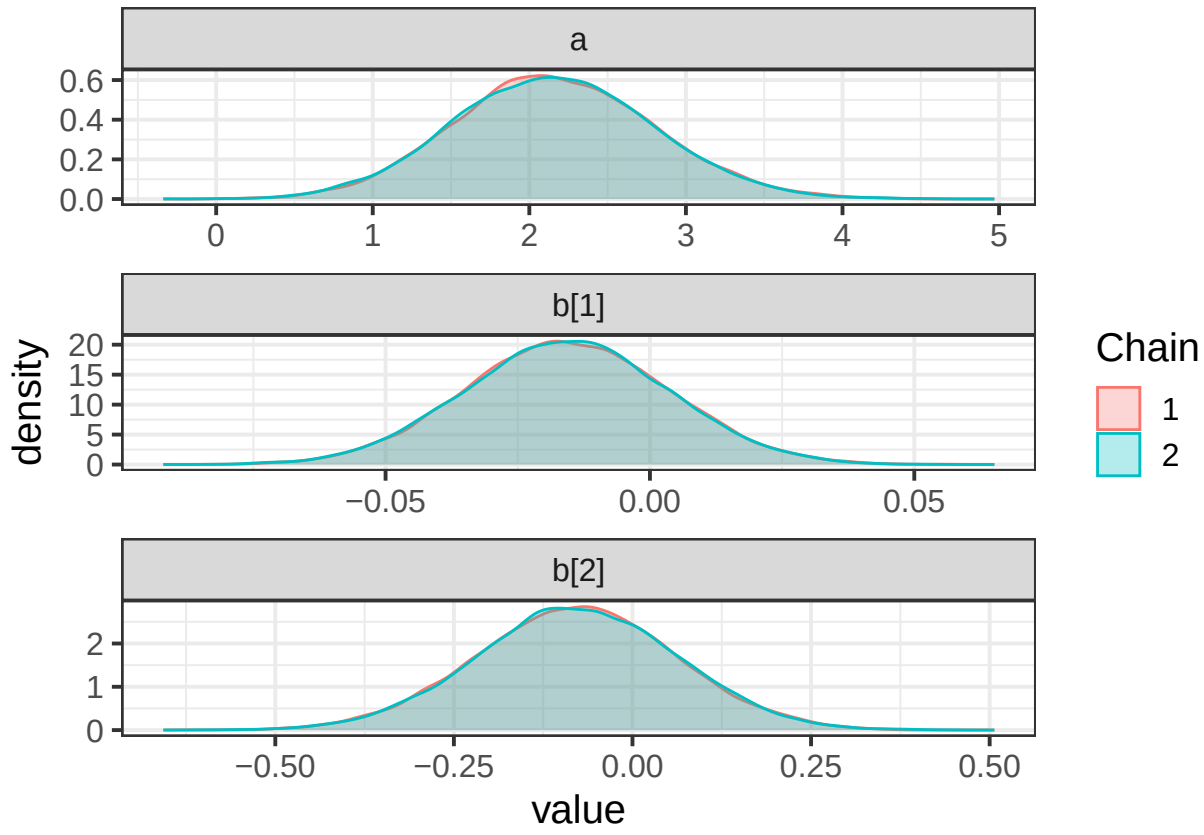
```
#write the model
write("model
{
 a~dnorm(0,1.0E-4)
for(i in 1:npredict) {
 b[i]~dnorm(0,1.0E-4)
}
for(i in 1:N)  {
  logit(p[i])<-a+b[1]*Year[i]+b[2]*hbf[i]
  present[i]~dbern(p[i])
  LL[i]<-log(present[i]*p[i]+(1-present[i])*(1-p[i]))
  simval[i]~dbern(p[i])
}
for(i in 1:nyear){
  logit(p.new[i]) <- a + b[1]*year.new[i] + b[2]*hbf.mean
}
}
",file=here("JAGS_mods","HW10_ProbB_LR_remove-miss.txt"))

#set up the data

sword.data.remove <- sword.data |>
  drop_na(Presence)

sword.ls.remove <- list(present = sword.data.remove$Presence,
                 Year = sword.data.remove$Year,
                 hbf = sword.data.remove$hbf,
                 N = length(sword.data.remove$Presence),
                 year.new = unique(sword.data.remove$Year),
                 hbf.mean = mean(sword.data.remove$hbf),
                 nyear = length(unique(sword.data.remove$Year)),
                 npredict = 2)

#run the model

sword_LR_removedMiss <-jags(sword.ls.remove,
  model.file=here("JAGS_mods","HW10_ProbB_LR_remove-miss.txt"),
  parameters.to.save=c("a","b","p","LL","simval","p.new"),
  n.chains=2,n.iter=100000,n.burnin=10000,n.thin=4)
```

**swoComplete<-na.omit(swoPres)**  i) Show the summary statistics for the three parameters.

```
round(sword_LR_removedMiss$BUGSoutput$summary[c("a","b[1]", "b[2]"),],2)
```

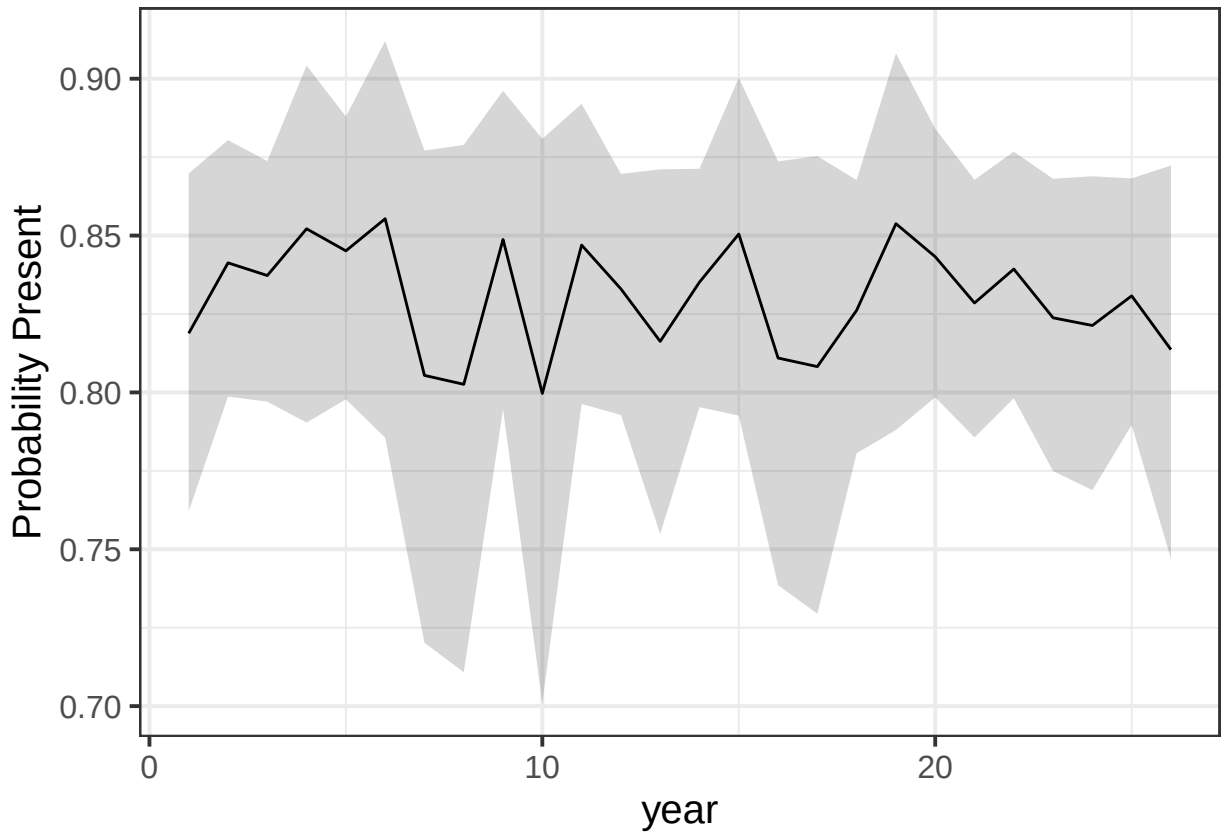
```
##         mean   sd  2.5%   25%   50%  75% 97.5% Rhat n.eff
## a       2.16 0.64  0.93  1.72  2.15 2.58  3.44    1 32000
## b[1] -0.02 0.02 -0.05 -0.03 -0.02 0.00  0.02    1 45000
## b[2] -0.08 0.14 -0.36 -0.17 -0.08 0.02  0.19    1 39000
```

```
sword1ggs<-ggs(as.mcmc(sword_LR_removedMiss))
sword1ggs<-sword1ggs[sword1ggs$Parameter %in% c("a","b[1]", "b[2]"),]
ggs_density(sword1ggs)
```




ii) Show the mean probabilities in each year with credible intervals (calculated with hbf at its mean value). What is the trend?

```
parameter <- paste0("p.new[",1:sword.ls.remove$nyear,"]")


df <- as_tibble(cbind(year = 1:sword.ls.remove$nyear,
             p = sword_LR_removedMiss$BUGSoutput$summary[parameter,c("2.5%","mean","97.5%")]))

df |>
  ggplot(aes(y = mean,x = year))+
  geom_ribbon(aes(ymin = `2.5%`, ymax = `97.5%`), alpha = 0.2)+
  geom_line()+
  labs(y = "Probability Present")
```




there seems to be little trend in the data or the probability of presence remains the same through time.

```
#write the model
write("model
{
 a~dnorm(0,1.0E-4)
for(i in 1:npredict) {
 b[i]~dnorm(0,1.0E-4)
}
for(i in 1:N)  {
  logit(p[i])<-a+b[1]*Year[i]+b[2]*hbf[i]
  present[i]~dbern(p[i])
  LL[i]<-log(present[i]*p[i]+(1-present[i])*(1-p[i]))
  simval[i]~dbern(p[i])
}
for(i in 1:nyear){
  logit(p.new[i]) <- a + b[1]*year.new[i] + b[2]*hbf.mean
}
}
",file=here("JAGS_mods","HW10_ProbB_LR_impute.txt"))

#set up the data
```

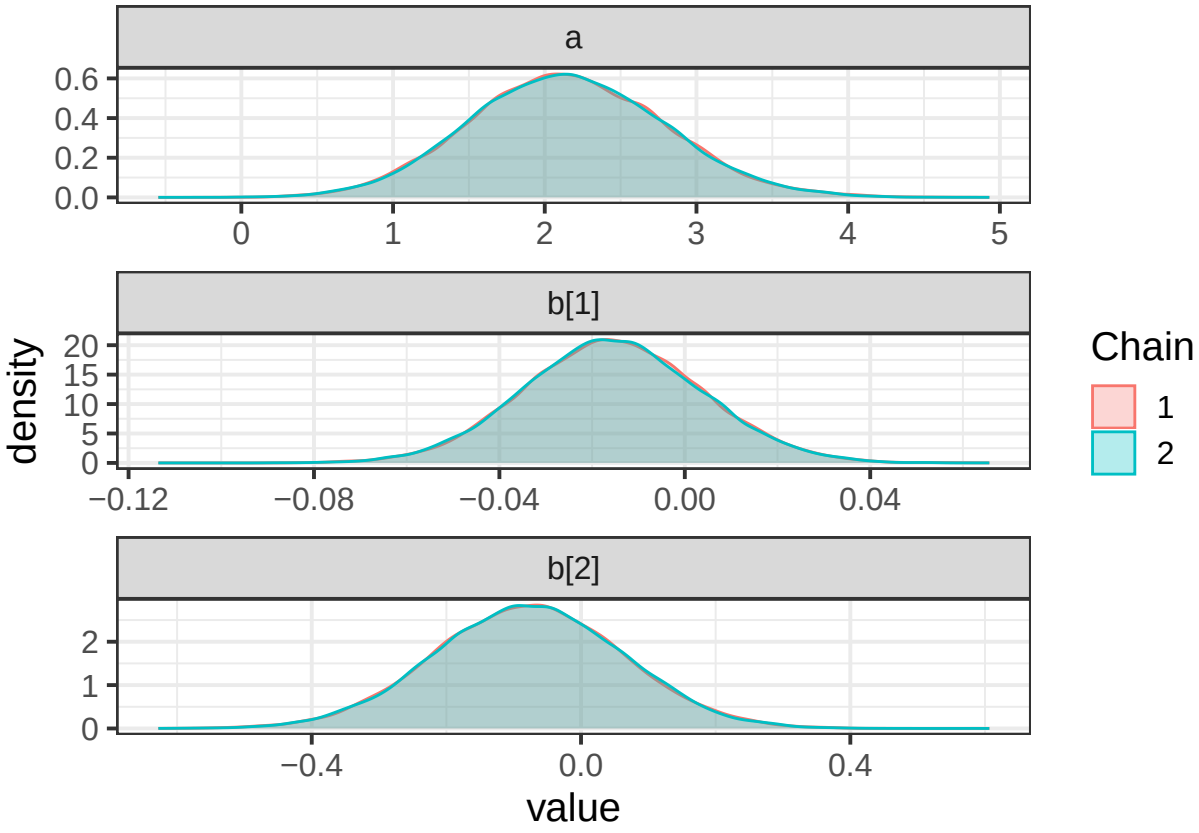
```
sword.ls <- list(present = sword.data$Presence,
                 Year = sword.data$Year,
                 hbf = sword.data$hbf,
                 N = length(sword.data$Presence),
                 year.new = unique(sword.data$Year),
                 hbf.mean = mean(sword.data$hbf),
                 nyear = length(unique(sword.data$Year)),
                 npredict = 2)

#run the model

sword_LR_impute <-jags(sword.ls,
  model.file=here("JAGS_mods","HW10_ProbB_LR_impute.txt"),
  parameters.to.save=c("a","b","p","LL","simval","p.new"),
  n.chains=2,n.iter=100000,n.burnin=10000,n.thin=4)
```

**B) Fit the model using both the complete records and those with missing y data.** i) Show the summary statistics for the three parameters.

```
round(sword_LR_impute$BUGSoutput$summary[c("a","b[1]", "b[2]"),],2)
```

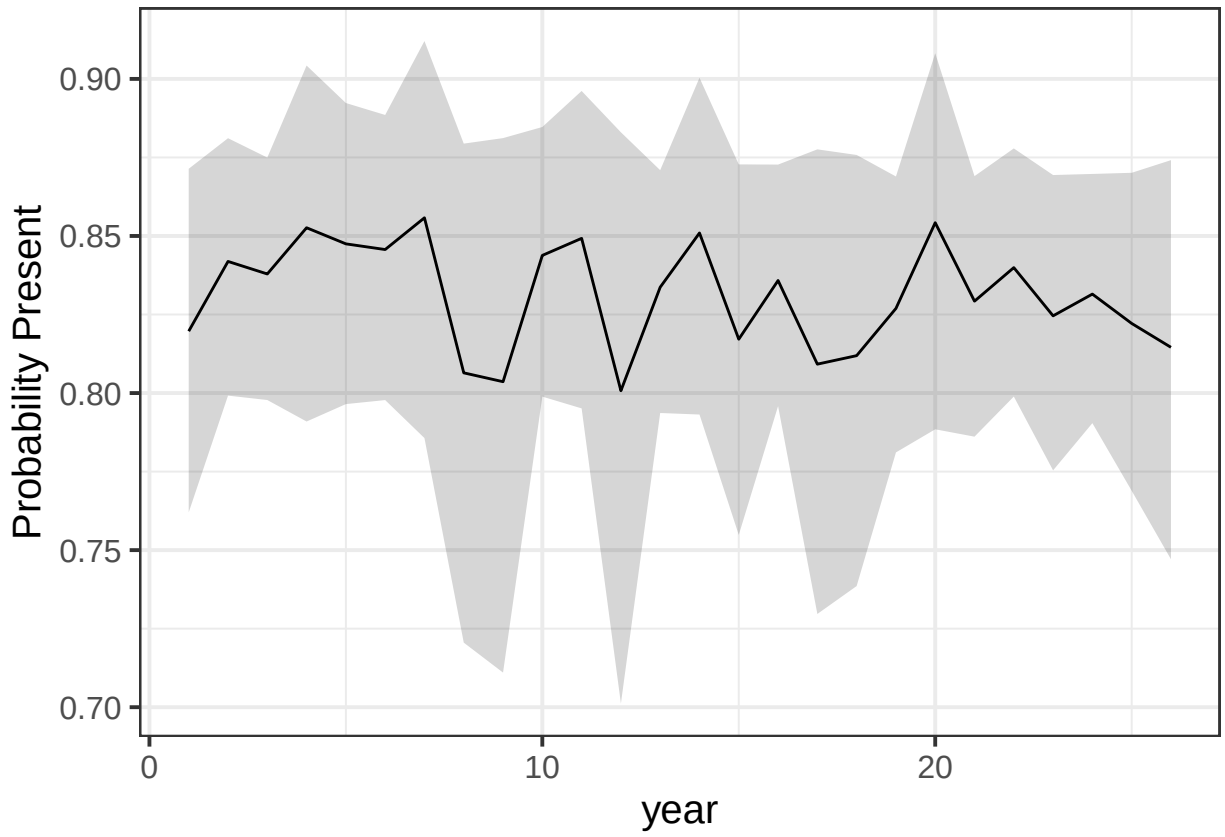
```
##       mean   sd  2.5%   25%   50%  75% 97.5% Rhat n.eff
## a     2.15 0.64  0.94  1.71  2.14 2.59  3.44    1 45000
## b[1] -0.02 0.02 -0.05 -0.03 -0.02 0.00  0.02    1 41000
## b[2] -0.08 0.14 -0.35 -0.17 -0.08 0.02  0.20    1 45000
```

```
sword2ggs<-ggs(as.mcmc(sword_LR_impute))
sword2ggs<-sword2ggs[sword2ggs$Parameter %in% c("a","b[1]", "b[2]"),]
ggs_density(sword2ggs)
```



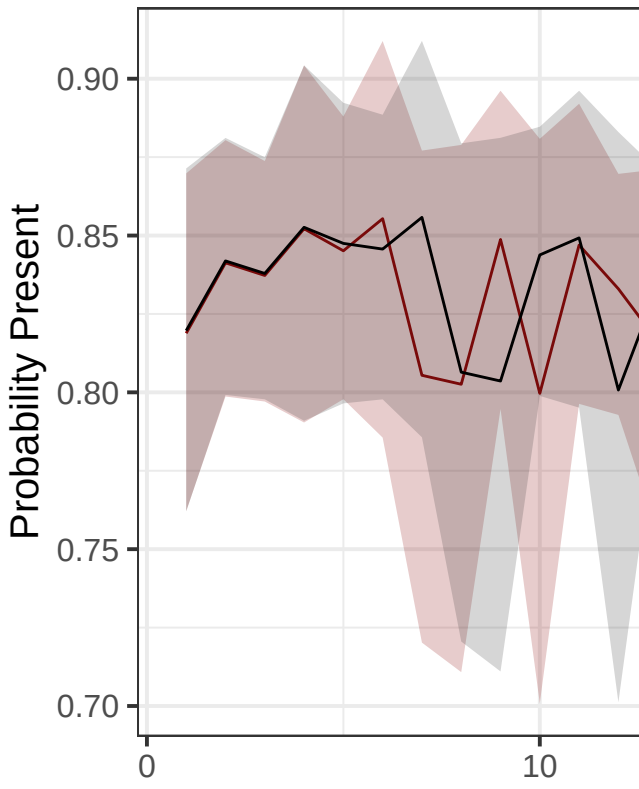

ii) Show the mean probabilities in each year with credible intervals. What is the trend?

```
parameter <- paste0("p.new[",1:sword.ls$nyear,"]")


df2 <- as_tibble(cbind(year = 1:sword.ls$nyear,
             p = sword_LR_impute$BUGSoutput$summary[parameter,c("2.5%","mean","97.5%")]))

df2 |>
  ggplot(aes(y = mean,x = year))+
  geom_ribbon(aes(ymin = `2.5%`, ymax = `97.5%`), alpha = 0.2)+
  geom_line()+
  labs(y = "Probability Present")
```




```
df2 |>
  ggplot(aes(y = mean,x = year))+
  geom_ribbon(data = df, aes(ymin = `2.5%`, ymax = `97.5%`), alpha = 0.2, fill = "darkred")+
  geom_line(data = df, aes(y = mean, x = year), color = "darkred")+
  geom_ribbon(aes(ymin = `2.5%`, ymax = `97.5%`), alpha = 0.2)+
  geom_line()+

  labs(y = "Probability Present")
```

**C) Did you get a similar result to the model with complete cases only and the model with missing data filled in? Which do you suppose is more accurate?**




The trends were similar in that there wasn't a trend. The imputed y data is likely more accurate.

## 3) Missing X data

**A) Fit a multiple regression with the all the variables but with only the rows with no missing data (complete cases). Show the summary statistics.** *write and run model*

```
#write the model

write("model
{
  for (i in 1:N)            # for each of the 73 sites
  {
    Y[i] ~ dnorm(ymean[i], prec)    # assume normal distribution
    ymean[i] <- a + b[1]*weight[i] + b[2]*masspercent[i] + b[3]*expirevolume[i] + b[4]*residvolume[i]
   pred.obs[i]~dnorm(ymean[i],prec)   # Predicted Y value
   resid[i]<-Y[i]-ymean[i]          # Residual
   sresid[i]<-(Y[i]-ymean[i])*sqrt(prec)  # Standardized residual
   sresid2[i]<-sresid[i]*sresid[i]  # Pearson residual squared
   rep.sresid2[i]<-(pred.obs[i]-ymean[i])*(pred.obs[i]-ymean[i])*prec
   LL[i]<--0.5*log(2*3.14159)+0.5*log(prec)-0.5*prec*(Y[i]-ymean[i])*(Y[i]-ymean[i])
  }
  # uninformative priors
  a ~ dnorm(0, 1.0E-6)
  for (i in 1:npredict)
```

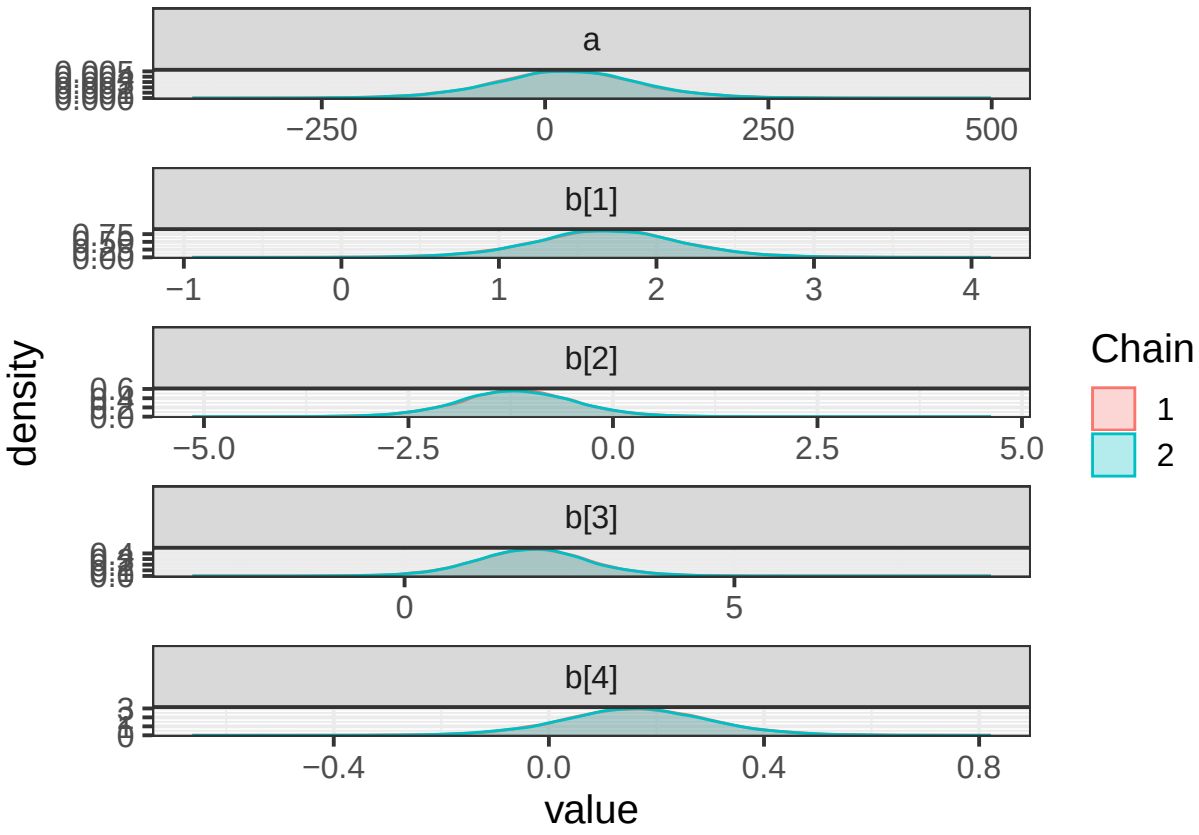
```
  {
    b[i] ~ dnorm(0, 1.0E-6)
  }
  prec ~ dgamma(0.001, 0.001)
  }
",file=here("JAGS_mods","HW10_probC_mOLS_remove.txt"))

#set up the data
pemax.data.remove <- pemax.data |>
  drop_na(weight)

pemax.ls.remove <- list(Y = pemax.data.remove$pemax,
                        weight = pemax.data.remove$weight,
                        masspercent = pemax.data.remove$masspercent,
                        expirevolume = pemax.data.remove$expirevolume,
                        residvolume = pemax.data.remove$residvolume,
                        N = length(pemax.data.remove$pemax),
                        npredict = 4)

#run the model

pemax_mOLS_remove <-jags(pemax.ls.remove,
  model.file=here("JAGS_mods","HW10_probC_mOLS_remove.txt"),
  parameters.to.save=c("a","b"),
  n.chains=2,n.iter=100000,n.burnin=10000,n.thin=4)
```

*summary*

```
round(pemax_mOLS_remove$BUGSoutput$summary[c("a","b[1]", "b[2]","b[3]","b[4]"),],2)
```

```
##       mean    sd    2.5%    25%   50%   75%  97.5% Rhat n.eff
## a    23.54 82.85 -142.56 -29.15 23.71 76.41 189.08    1 12000
## b[1]  1.69  0.49    0.73   1.38  1.69  2.00   2.66    1 45000
## b[2] -1.17  0.75   -2.64  -1.65 -1.18 -0.70   0.31    1 31000
## b[3]  1.93  0.88    0.17   1.37  1.94  2.50   3.70    1 45000
## b[4]  0.16  0.14   -0.12   0.07  0.16  0.25   0.45    1 12000
```

```
pemax1ggs<-ggs(as.mcmc(pemax_mOLS_remove))
pemax1ggs<-pemax1ggs[pemax1ggs$Parameter %in% c("a","b[1]", "b[2]","b[3]","b[4]"),]
ggs_density(pemax1ggs)
```




**B) Now impute the missing weight data as normal variable with an estimated mean and precision. Show the summary statistics for the regression coefficients and the missing values. Did you get the same values for the regression coefficients as before?** *write and run model*

```
#write the model

write("model
{
  for (i in 1:N)            # for each of the data
  {
    Y[i] ~ dnorm(ymean[i], prec)    # assume normal distribution
    ymean[i] <- a + b[1]*weight[i] + b[2]*masspercent[i] + b[3]*expirevolume[i] + b[4]*residvolume[i]
   pred.obs[i]~dnorm(ymean[i],prec)   # Predicted Y value
   resid[i]<-Y[i]-ymean[i]          # Residual
   sresid[i]<-(Y[i]-ymean[i])*sqrt(prec)  # Standardized residual
   sresid2[i]<-sresid[i]*sresid[i]  # Pearson residual squared
   rep.sresid2[i]<-(pred.obs[i]-ymean[i])*(pred.obs[i]-ymean[i])*prec
   LL[i]<--0.5*log(2*3.14159)+0.5*log(prec)-0.5*prec*(Y[i]-ymean[i])*(Y[i]-ymean[i])
   weight[i]~dnorm(mu.imp,prec.imp)
  }
  # uninformative priors
  a ~ dnorm(0, 1.0E-6)
  for (i in 1:npredict)
  {
    b[i] ~ dnorm(0, 1.0E-6)
```

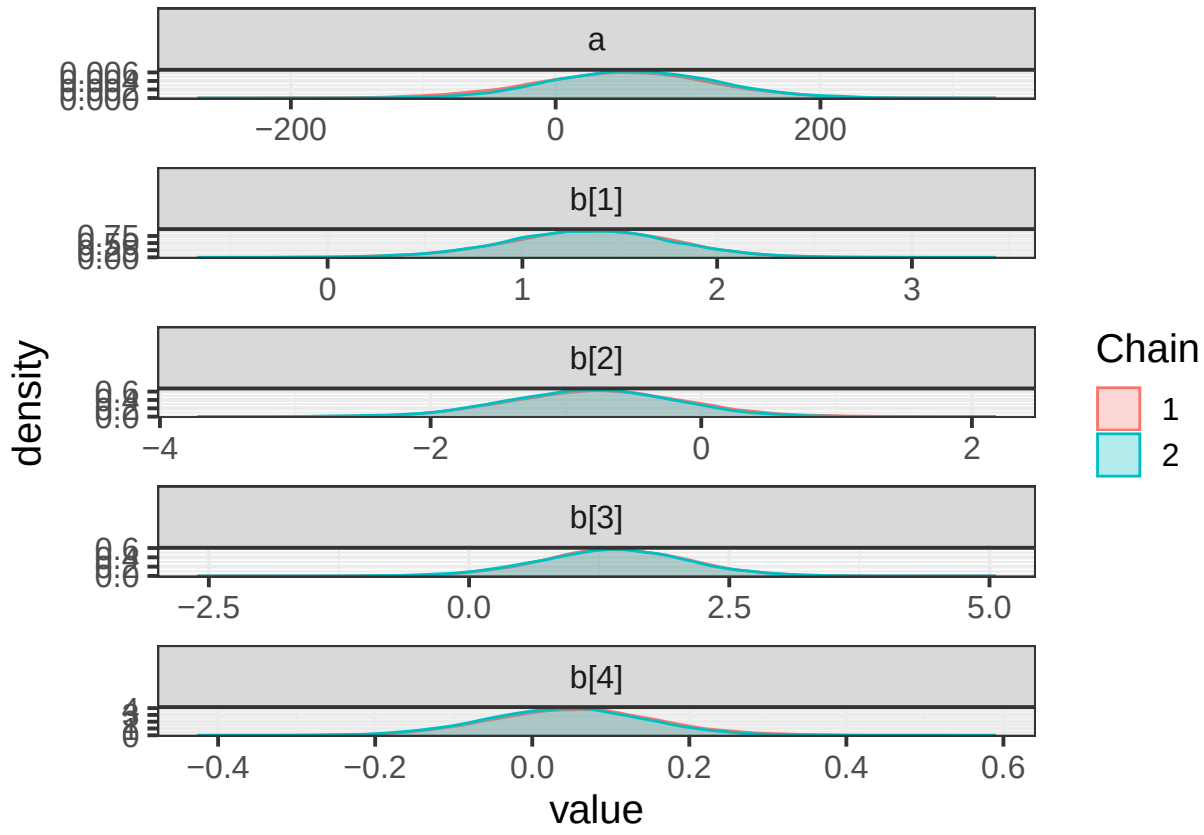
```
  }
  prec ~ dgamma(0.001, 0.001)

  #hyperpriors

  mu.imp ~ dnorm(0,1.0E-6)
  prec.imp ~ dgamma(0.001, 0.001)
  }
",file=here("JAGS_mods","HW10_probC_mOLS_impute.txt"))

#set up the data


pemax.ls <- list(Y = pemax.data$pemax,
                         weight = pemax.data$weight,
                         masspercent = pemax.data$masspercent,
                         expirevolume = pemax.data$expirevolume,
                         residvolume = pemax.data$residvolume,
                         N = length(pemax.data$pemax),
                         npredict = 4)

#run the model

pemax_mOLS_impute<-jags(pemax.ls,
  model.file=here("JAGS_mods","HW10_probC_mOLS_impute.txt"),
  parameters.to.save=c("a","b"),
  n.chains=2,n.iter=100000,n.burnin=10000,n.thin=4)
```

*summary*

```
round(pemax_mOLS_impute$BUGSoutput$summary[c("a","b[1]", "b[2]","b[3]","b[4]"),],2)
```

```
##       mean    sd   2.5%   25%   50%   75%  97.5% Rhat n.eff
## a    54.77 66.02 -76.74 11.33 54.91 98.59 183.21 1.01   160
## b[1]  1.34  0.43   0.47  1.06  1.35  1.63   2.17 1.00  8300
## b[2] -0.79  0.65  -2.06 -1.22 -0.79 -0.36   0.47 1.01   340
## b[3]  1.39  0.72  -0.05  0.92  1.39  1.86   2.80 1.00  2700
## b[4]  0.05  0.10  -0.15 -0.02  0.05  0.12   0.26 1.01   240
```

```
pemax2ggs<-ggs(as.mcmc(pemax_mOLS_impute))
pemax2ggs<-pemax2ggs[pemax2ggs$Parameter %in% c("a","b[1]", "b[2]","b[3]","b[4]"),]
ggs_density(pemax2ggs)
```




*comparison*

```
round(pemax_mOLS_remove$BUGSoutput$summary[c("a","b[1]", "b[2]","b[3]","b[4]"),],2)
```

```
##       mean    sd    2.5%    25%   50%   75%  97.5% Rhat n.eff
## a    23.54 82.85 -142.56 -29.15 23.71 76.41 189.08    1 12000
## b[1]  1.69  0.49    0.73   1.38  1.69  2.00   2.66    1 45000
## b[2] -1.17  0.75   -2.64  -1.65 -1.18 -0.70   0.31    1 31000
## b[3]  1.93  0.88    0.17   1.37  1.94  2.50   3.70    1 45000
## b[4]  0.16  0.14   -0.12   0.07  0.16  0.25   0.45    1 12000
```

```
round(pemax_mOLS_impute$BUGSoutput$summary[c("a","b[1]", "b[2]","b[3]","b[4]"),],2)
```

```
##       mean    sd   2.5%   25%   50%   75%  97.5% Rhat n.eff
## a    54.77 66.02 -76.74 11.33 54.91 98.59 183.21 1.01   160
## b[1]  1.34  0.43   0.47  1.06  1.35  1.63   2.17 1.00  8300
## b[2] -0.79  0.65  -2.06 -1.22 -0.79 -0.36   0.47 1.01   340
## b[3]  1.39  0.72  -0.05  0.92  1.39  1.86   2.80 1.00  2700
## b[4]  0.05  0.10  -0.15 -0.02  0.05  0.12   0.26 1.01   240
```

the regression coefficients from the model with imputed x data were different from the model with the the missing data omitted.