CE Assignment 03/24

Yitong Wu

March 24, 2020

```r
library(readr)
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
```

```r
library(ggplot2)
library(fastDummies)
knitr::opts_chunk$set(echo = TRUE)
def.chunk.hook  <- knitr::knit_hooks$get("chunk")
knitr::knit_hooks$set(chunk = function(x, options) {
  x <- def.chunk.hook(x, options)
  ifelse(options$size != "normalsize", paste0("\\", options$size,"\n\n", x, "\n\n \\normalsize"), x)
})
require(runjags)
```

```
## Loading required package: runjags
```

```r
require(coda)
```

```
## Loading required package: coda
```

```r
data <- read.csv("CEdata.csv")
data$Rural = fastDummies::dummy_cols(data$UrbanRural)[,names(fastDummies::dummy_cols(data$UrbanRural)) =
data$Race_Black = fastDummies::dummy_cols(data$Race)[,names(fastDummies::dummy_cols(data$Race)) == ".dat
data$Race_NA = fastDummies::dummy_cols(data$Race)[,names(fastDummies::dummy_cols(data$Race)) == ".data_3
data$Race_Asian = fastDummies::dummy_cols(data$Race)[,names(fastDummies::dummy_cols(data$Race)) == ".dat
data$Race_PI = fastDummies::dummy_cols(data$Race)[,names(fastDummies::dummy_cols(data$Race)) == ".data_5
data$Race_M = fastDummies::dummy_cols(data$Race)[,names(fastDummies::dummy_cols(data$Race)) == ".data_6
data$logInc <- log(data$Income)
```

```r
modelString <-"
model {
## sampling
for (i in 1:N){
y[i] ~ dnorm(beta0 + beta1*x_rural[i] +
beta2*x_race_B[i] + beta3*x_race_N[i] +
beta4*x_race_A[i] + beta5*x_race_P[i] + beta6*x_race_M[i], invsigma2)
}
## priors
beta0 ~ dnorm(mu0, g0)
beta1 ~ dnorm(mu1, g1)
beta2 ~ dnorm(mu2, g2)
beta3 ~ dnorm(mu3, g3)
beta4 ~ dnorm(mu4, g4)
beta5 ~ dnorm(mu5, g5)
```

```
beta6 ~ dnorm(mu6, g6)
invsigma2 ~ dgamma(a, b)
sigma <- sqrt(pow(invsigma2, -1))
}
"
```

```
y = as.vector(data$logInc)
x_rural = as.vector(data$Rural)
x_race_B = as.vector(data$Race_Black)
x_race_N = as.vector(data$Race_NA)
x_race_A = as.vector(data$Race_Asian)
x_race_P = as.vector(data$Race_PI)
x_race_M = as.vector(data$Race_M)
N = length(y)
```

```
the_data <- list("y" = y,
                 "x_rural" = x_rural, "x_race_B" = x_race_B,
                 "x_race_N" = x_race_N, "x_race_A" = x_race_A,
                 "x_race_P" = x_race_P, "x_race_M" = x_race_M,
                 "N" = N,
                 "mu0" = 0, "g0" = 1, "mu1" = 0, "g1" = 1,
                 "mu2" = 0, "g2" = 1, "mu3" = 0, "g3" = 1,
                 "mu4" = 0, "g4" = 1, "mu5" = 0, "g5" = 1,
                 "mu6" = 0, "g6" = 1,
                 "a" = 1, "b" = 1)
```

```
initsfunction <- function(chain){
  .RNG.seed <- c(1,2)[chain]
  .RNG.name <- c("base::Super-Duper",
                 "base::Wichmann-Hill")[chain]
  return(list(.RNG.seed=.RNG.seed,
              .RNG.name=.RNG.name))
}
```

```
posterior_MLR <- run.jags(modelString,
                      n.chains = 1,
                      data = the_data,
                      monitor = c("beta0", "beta1", "beta2",
                                  "beta3", "beta4", "beta5",
                                  "beta6", "sigma"),
                      adapt = 1000,
                      burnin = 5000,
                      sample = 5000,
                      thin = 1,
                      inits = initsfunction)
```

```
## Warning: Convergence cannot be assessed with only 1 chain
```

```
post <- as.mcmc(posterior_MLR)
```

```
synthesize <- function(X_rural, X_RB, X_RN, X_RA, X_RP, X_RM, index, n){
  mean_Y <- post[index, "beta0"] +  X_rural * post[index, "beta1"] + X_RB * post[index, "beta2"] + X_RN * post[index, "beta3"] +
  synthetic_Y <- rnorm(n, mean_Y, post[index, "sigma"])
  data.frame(synthetic_Y, X_rural, X_RB, X_RN, X_RA, X_RP, X_RM)
}
```

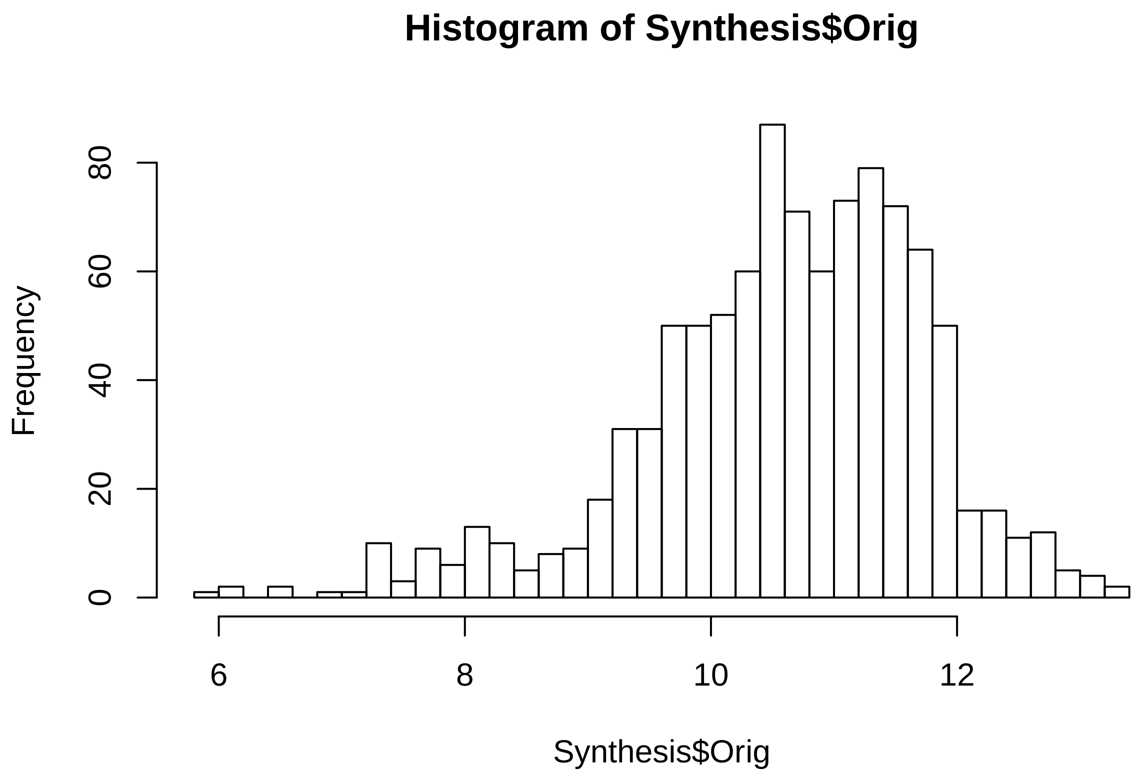
```
n <- dim(data)[1]
Syndata <- synthesize(data$Rural, data$Race_Black, data$Race_NA, data$Race_Asian, data$Race_PI, data$Rac
names(Syndata) <- c("logInc", "Rural", "RB", "RN", "RA", "RP", "RM")
```

```
Synthesis <- cbind(data$logInc, Syndata$logInc)
Synthesis <- data.frame(Synthesis)
names(Synthesis) <- c("Orig", "Syn")
```

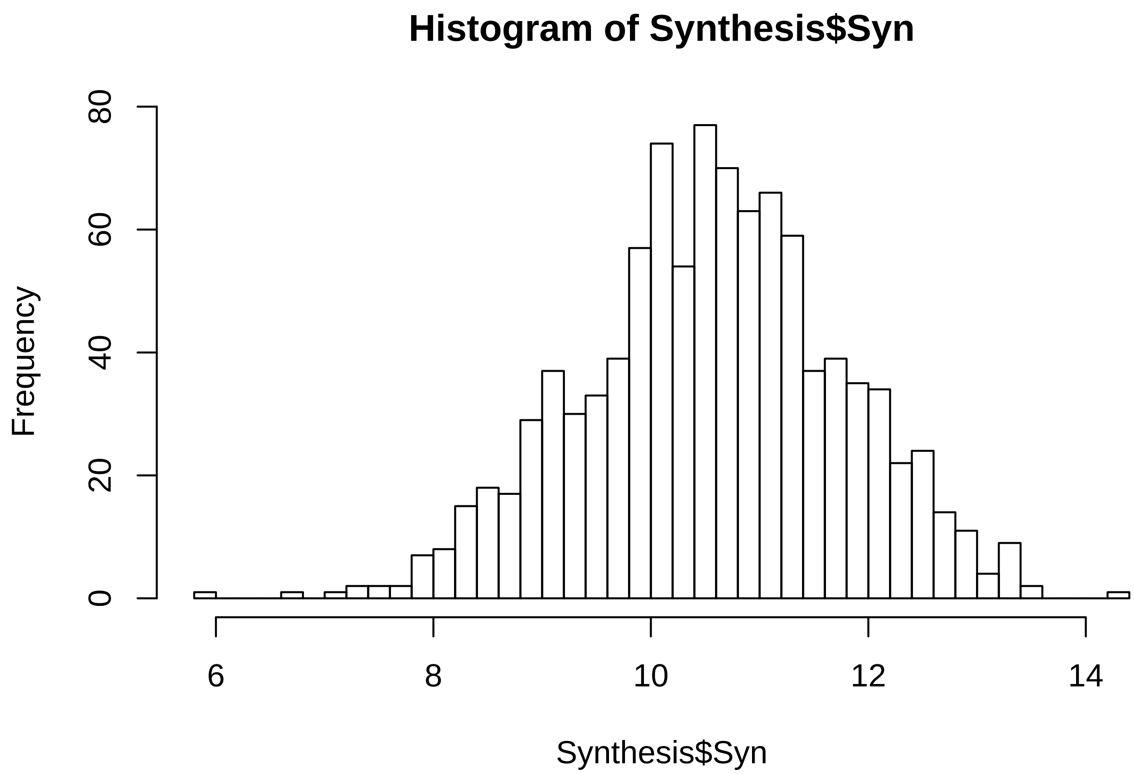
```
Synthesis <- cbind(Synthesis, data$UrbanRural, data$Race)
```

```r
Synthesis <- data.frame(Synthesis)
names(Synthesis) <- c("Orig", "Syn", "Urban", "Race")
```

```r
hist(Synthesis$Orig, breaks = 50)
```




```r
hist(Synthesis$Syn, breaks = 50)
```




```
CalculateKeyQuantities <- function(origdata, syndata, known.vars, syn.vars, n){
  origdata <- origdata
  syndata <- syndata
  n <- n

  c_vector <- rep(NA, n)
  T_vector <- rep(NA, n)

  for (i in 1:n){
    match <- (eval(parse(text=paste("origdata$",syn.vars,"[i]==
                                    syndata$",syn.vars,sep="",collapse="&")))&
                eval(parse(text=paste("origdata$",known.vars,"[i]==
                                      syndata$",known.vars,sep="",collapse="&"))))
    match.prob <- ifelse(match, 1/sum(match), 0)

    if (max(match.prob) > 0){
      c_vector[i] <- length(match.prob[match.prob == max(match.prob)])
    }
    else
      c_vector[i] <- 0
      T_vector[i] <- is.element(i, rownames(origdata)[match.prob == max(match.prob)])
  }

  K_vector <- (c_vector * T_vector == 1)
  F_vector <- (c_vector * (1 - T_vector) == 1)
  s <- length(c_vector[c_vector == 1 & is.na(c_vector) == FALSE])

  res_r <- list(c_vector = c_vector,
                T_vector = T_vector,
```

```r
                K_vector = K_vector,
                F_vector = F_vector,
                s = s
  )
  return(res_r)
}
```

```r
IdentificationRisk <- function(c_vector, T_vector, K_vector, F_vector, s, N){

  nonzero_c_index <- which(c_vector > 0)

  exp_match_risk <- sum(1/c_vector[nonzero_c_index]*T_vector[nonzero_c_index])
  true_match_rate <- sum(na.omit(K_vector))/N
  false_match_rate <- sum(na.omit(F_vector))/s

  res_r <- list(exp_match_risk = exp_match_risk,
                true_match_rate = true_match_rate,
                false_match_rate = false_match_rate
  )
  return(res_r)
}
```

```r
Synthesis$Orig1 <- round(Synthesis$Orig, digits=1)
Synthesis$Syn1 <- round(Synthesis$Syn, digits=1)
Synthesis$Orig2 <- round(Synthesis$Orig, digits=2)
Synthesis$Syn2 <- round(Synthesis$Syn, digits=2)
```

```r
org1 <- cbind(Synthesis$Orig1, Synthesis$Urban, Synthesis$Race)
org1 <- data.frame(org1)
names(org1) <- c("inc", "urb", "race")
```

```r
syn1 <- cbind(Synthesis$Syn1, Synthesis$Urban, Synthesis$Race)
syn1 <- data.frame(syn1)
names(syn1) <- c("inc", "urb", "race")
```

```r
org2 <- cbind(Synthesis$Orig2, Synthesis$Urban, Synthesis$Race)
org2 <- data.frame(org2)
names(org2) <- c("inc", "urb", "race")
```

```r
syn2 <- cbind(Synthesis$Syn2, Synthesis$Urban, Synthesis$Race)
syn2 <- data.frame(syn2)
names(syn2) <- c("inc", "urb", "race")
```

```r
known.vars <- c("urb", "race")
syn.vars <- c("inc")
n <- dim(org1)[1]
```

```r
KeyQuantities <- CalculateKeyQuantities(org1, syn1,
                                        known.vars, syn.vars, n)
c_vector <- KeyQuantities[["c_vector"]]
T_vector <- KeyQuantities[["T_vector"]]
K_vector <- KeyQuantities[["K_vector"]]
F_vector <- KeyQuantities[["F_vector"]]
s <- KeyQuantities[["s"]]
N <- n
```

```r
ThreeSummaries <- IdentificationRisk(c_vector, T_vector, K_vector, F_vector, s, N)
ThreeSummaries[["exp_match_risk"]]
```

```
## [1] 3.938998
```

```r
ThreeSummaries[["true_match_rate"]]
```

```
## [1] 0.002012072
```

```r
ThreeSummaries[["false_match_rate"]]
```

```
## [1] 0.9726027
```

```r
KeyQuantities <- CalculateKeyQuantities(org2, syn2,
                                        known.vars, syn.vars, n)
c_vector <- KeyQuantities[["c_vector"]]
T_vector <- KeyQuantities[["T_vector"]]
K_vector <- KeyQuantities[["K_vector"]]
F_vector <- KeyQuantities[["F_vector"]]
s <- KeyQuantities[["s"]]
N <- n
ThreeSummaries <- IdentificationRisk(c_vector, T_vector, K_vector, F_vector, s, N)
ThreeSummaries[["exp_match_risk"]]
```

```
## [1] 1
```

```r
ThreeSummaries[["true_match_rate"]]
```

```
## [1] 0.001006036
```

```r
ThreeSummaries[["false_match_rate"]]
```

```
## [1] 0.9956332
```