ARCalculation

Sarah Boese

4/6/2020

```
library(ProbBayes)
library(dplyr)
library(ggplot2)
require(gridExtra)
library(reshape)
library(runjags)
library(coda)
library(tidyverse)
library(fastDummies)
crcblue <- "#2905a1"
```

```
CEdata<-read.csv("CEsample2.csv")
names(CEdata)<-c("UrbanRural", "Income", "Race", "Expenditure")
CESample <- read.csv("CEsample2.csv")
```

```
CEdata$LogIncome <- log(CEdata$Income)
CEdata$LogExpenditure <- log(CEdata$Expenditure)
```

```
CESample <- CESample %>%
  mutate(LogTotalIncome = log(TotalIncomeLastYear))
CESample <- CESample %>%
  mutate(LogTotalExp = log(TotalExpLastQ))
```

- Pass the data and hyperparameter values to JAGS:

```
y_income = as.vector(CESample$LogTotalIncome)
x_exp = as.vector(CESample$LogTotalExp)
x_rural = as.vector(CESample$Rural)
x_race_B = as.vector(CESample$Race_Black)
x_race_N = as.vector(CESample$Race_NA)
x_race_A = as.vector(CESample$Race_Asian)
x_race_P = as.vector(CESample$Race_PI)
x_race_M = as.vector(CESample$Race_M)
N = length(y_income)  # Compute the number of observations
```

- Pass the data and hyperparameter values to JAGS:
- Pass the data and hyperparameter values to JAGS:
- Run the JAGS code for this model:

```
n<-nrow(CEdata)
synthetic_list<-synthesize_log_income(1)
```

```
## Calling the simulation...
## Welcome to JAGS 4.3.0 on Tue Apr  7 13:02:01 2020
## JAGS is free software and comes with ABSOLUTELY NO WARRANTY
## Loading module: basemod: ok
## Loading module: bugs: ok
## . . Reading data file data.txt
## . Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 994
##    Unobserved stochastic nodes: 9
##    Total graph size: 9984
## . Reading parameter file inits1.txt
## . Initializing model
## . Adaptation skipped: model is not in adaptive mode.
## . Updating 5000
## -------------------------------------------------| 5000
## ************************************************** 100%
## . . . . . . . . . . Updating 50000
## -------------------------------------------------| 50000
## ************************************************** 100%
## . . . . Updating 0
## . Deleting model
## .
## Note: the model did not require adaptation
## Simulation complete.  Reading coda files...
## Coda files loaded successfully
## Calculating summary statistics...

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

## Finished running the simulation
```

```
synthetic_one<- synthetic_list[[1]]
post<-synthetic_list[[2]]
```

```
CEdata_org <- CEdata[, 1:4]
CEdata_syn <- as.data.frame(cbind(CEdata_org[, "UrbanRural"],
                            exp(synthetic_one[, "LogIncome"]),
                            cbind(CEdata_org [, c("Race","Expenditure")])))
names(CEdata_syn) <- c("UrbanRural", "Income", "Race", "Expenditure")
```

```
CEdata_org$LogIncome <- round(log(CEdata_org$Income), digits = 1)
CEdata_org$LogExpenditure <- round(log(CEdata_org$Expenditure), digits = 1)
CEdata_syn$LogIncome <- round(log(CEdata_syn$Income), digits = 1)
CEdata_syn$LogExpenditure <- round(log(CEdata_syn$Expenditure), digits = 1)
```

```
compute_logsumexp <- function(log_vector){
  log_vector_max <- max(log_vector)
  exp_vector <- exp(log_vector - log_vector_max)
  sum_exp <- sum(exp_vector)
```

```
  log_sum_exp <- log(sum_exp) + log_vector_max
  return(log_sum_exp)
}
```

```
calc_prob_rank<-function(i, H, post){
  y_i <- CEdata_org$LogIncome[i]
  y_i_guesses <- seq((y_i - 2.5), (y_i + 2.5), 0.5)
  X_i <- CEdata_syn$LogExpenditure[i]
  G <- length(y_i_guesses)
  beta0_draws <- post[1:H, "beta0"]
  beta1_draws <- post[1:H, "beta1"]
  sigma_draws <- post[1:H, "sigma"]

  CU_i_logZ_all <- rep(NA, G)
  for (g in 1:G){
    q_sum_H <- sum((dnorm(y_i_guesses[g],
    mean = (beta0_draws + beta1_draws * X_i),
    sd = sigma_draws)) /(dnorm(y_i, mean = (beta0_draws + beta1_draws * X_i), sd = sigma_draws)))
    log_pq_h_all <- rep(NA, H)
    for (h in 1:H){
      log_p_h <- sum(log(dnorm(CEdata_syn$LogIncome, mean = (beta0_draws[h] + beta1_draws[h] *
                                                              CEdata_syn$LogExpenditure), sd = sigma_d
      log_q_h <- log(((dnorm(y_i_guesses[g], mean = (beta0_draws[h] + beta1_draws[h] * X_i), sd = sigma
                    (dnorm(y_i, mean = (beta0_draws[h] + beta1_draws[h] * X_i),
                                sd = sigma_draws[h]))) / q_sum_H)
      log_pq_h_all[h] <- log_p_h + log_q_h
    }
    CU_i_logZ_all[g] <- compute_logsumexp(log_pq_h_all)
  }
  prob <- exp(CU_i_logZ_all - max(CU_i_logZ_all)) / sum(exp(CU_i_logZ_all - max(CU_i_logZ_all)))
  outcome <- as.data.frame(cbind(y_i_guesses, prob))
  names(outcome) <- c("guess", "probability")
  outcome[order(outcome$probability, decreasing = TRUE), ]
  rank<-which(outcome[,1]==y_i)
  rank_df<-slice(outcome, rank)
  probability<-as.numeric(rank_df$probability)
  out<-data.frame(rank, probability)
  names(out)<-c("rank","probability")
  return(out)
}
```

```
prob_rank_df<-calc_prob_rank(1, 50,post)
for(i in 2:n){
  prob_rank_df_i<-calc_prob_rank(i,50,post)
  prob_rank_df<-bind_rows(prob_rank_df, prob_rank_df_i)
}
```