STAT 695 Bayesian Data Analysis

Wei Deng

Sep 2, 2017

Part 1, Computational problem

Consider the following model for the binary responses of a certain process:

$Pr(y_i = 1|\beta, x_i) = \frac{\beta_0+\beta_1 x_i}{1+exp\beta_0+\beta_1 x_i}$

where i is the experimental unit index, y_i is the response (with possible values of 1 for success and 0 for failure), x_i is a treatment level, and $\beta = (\beta_0, \beta_1)^T$ is a vector of unknown parameters. We interpret β_0 as the logarithm of the odds of success for an experimental unit whose treatment level is zero, and β_1 as the dierence in the logarithm of the odds of success for a level change in the treatment.

The estimand of interest is $exp(\beta_1)$, which captures the multiplicative change in the odds of success for a level change in the treatment.

Set working directory first.

```
set.seed(26)
setwd("C:/Users/Wei/Desktop/STAT 695/HW1")
```

Q1. A person who is knowledgeable about this process believes that the odds of success for $x_i = 0$ is equal to 0.3, with a 10% chance that it could exceed 0.4. She also believes that the odds of success changes by a factor of 1.5 for each treatment level increase, with a 5% chance that it could exceed 2. Specify a prior for β based on this information, and provide a visualization of it.

Suppose we have normal priors for β_0 and β_1 with $\mu = 0.3$ and $\mu = 1.5$ respectively, use Monte Carlo method to determine their standard deviation, from the quantile below, we can roughly say that $\sigma = 0.077$ for β_0 and $\sigma = 0.305$ for β_1

```
set_sol = seq(0, 3, 0.001)
thres_beta_0 = sapply(set_sol, function(sigma) abs(quantile(rnorm(1000, log(0.3), sigma), 0.9) - 0.4))
sigma_0 = set_sol[which.min(thres_beta_0)]

thres_beta_1 = sapply(set_sol, function(sigma) abs(quantile(rnorm(1000, log(1.5), sigma), 0.95) - 2))
sigma_1 = set_sol[which.min(thres_beta_1)]
```

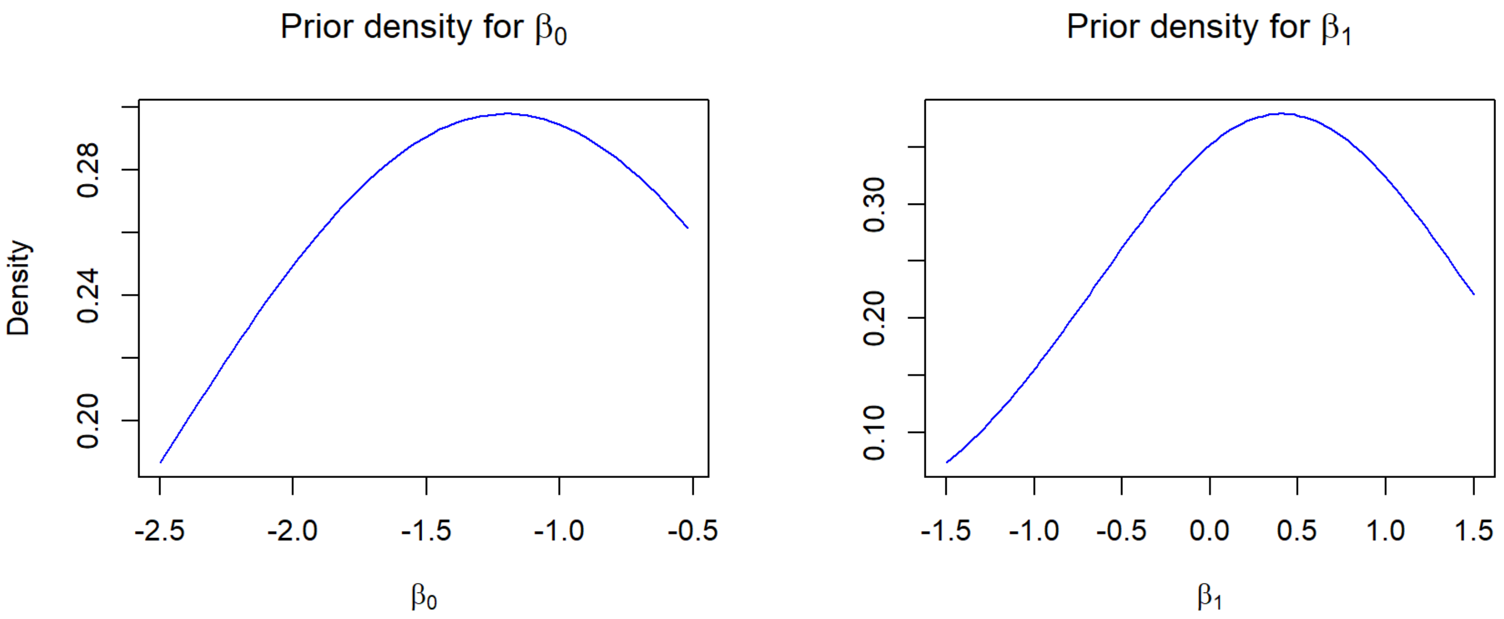
Using prior knowledge, we believe β_0~$N(log(0.3), sigma_0^2)$, β_1~$N(log(1.5), sigma_1^2)$, visualization goes like the following

```r
prior_beta_0 = function(beta_0) dnorm(beta_0, log(0.3), sigma_0)
prior_beta_1 = function(beta_1) dnorm(beta_1, log(1.5), sigma_1)

x_grid = seq(-2.5, -0.5, 0.03)
y_grid = seq(-1.5, 1.5, 0.03)

points_beta_0 = sapply(x_grid, prior_beta_0)
points_beta_1 = sapply(y_grid, prior_beta_1)

par(mfrow=c(1,2))
plot(x_grid, points_beta_0, xlab=expression(beta[0]), ylab='Density', type="l", col="blue",
main=expression(paste("Prior density for ", beta[0])))
plot(y_grid, points_beta_1, xlab=expression(beta[1]), ylab='', type="l", col="blue",
main=expression(paste("Prior density for ", beta[1])))
```



Q2. The file *computationdata. csv* contains data generated from this process. Construct the likelihood and log-likelihood functions for β based on this data, and provide visualizations of both.

Good example here (https://www.mathworks.com/help/stats/examples/bayesian-analysis-for-a-logistic-regression-model.html?requestedDomain=www.mathworks.com)

Read data

```r
data = read.csv(file="computation_data.csv", header=TRUE)
x = data$x
y = data$y
```

Get odd ratio function

```r
odd_ratio = function(beta_0, beta_1) 1 / (1 + exp(- beta_0 - beta_1 * x))
```

Compute log likelihood and derive likelihood function using it. **Writing log likelihood function first is to avoid numerical issues in likelihood function**

```r
log_likelihood = function(beta_0, beta_1) sum(log(dbinom(y, 1, odd_ratio(beta_0, beta_1))))
```

```
library(scatterplot3d)
```

Expand 2-d grids

```
x_grid = seq(-2.5, -0.5, 0.03)
y_grid = seq(-1.5, 0, 0.03)
grids = expand.grid(x_grid,y_grid) # create a grid for the 2-d tuples
```

Compute (log) likelihood values over these grids.

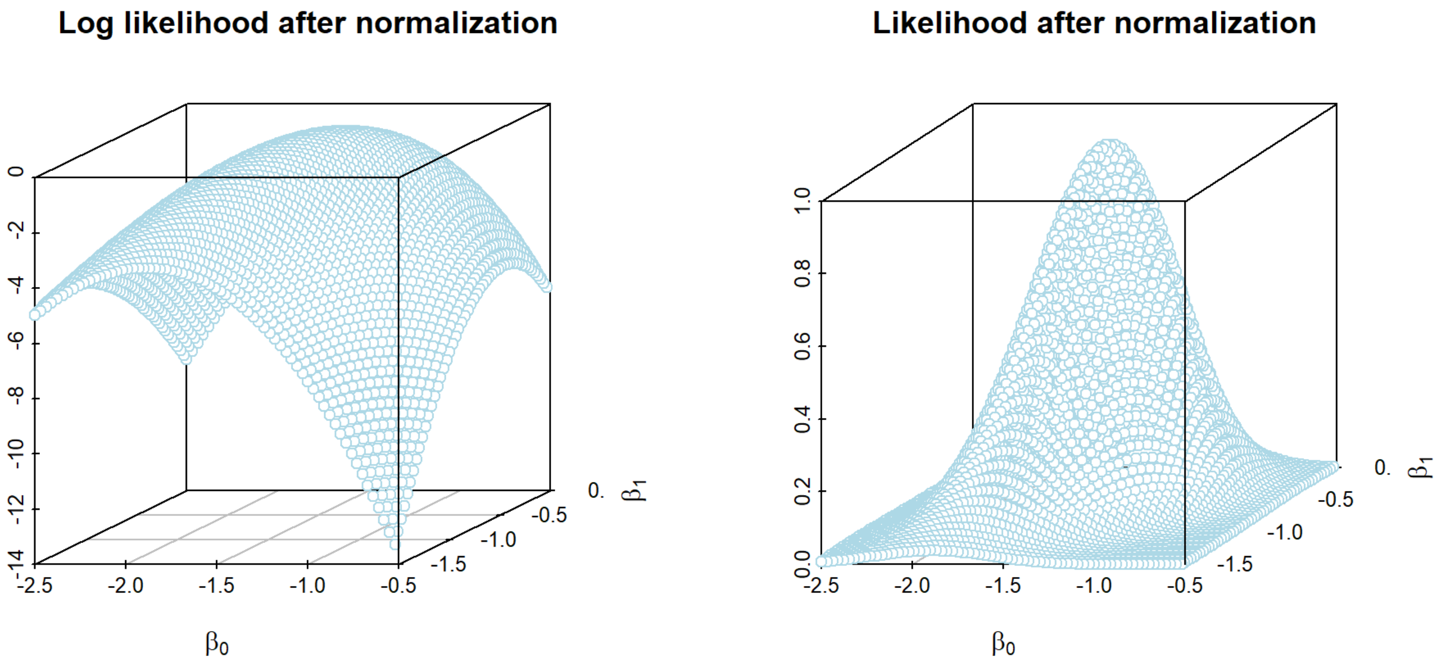
```
val_log_likelihood = mapply(log_likelihood, grids$Var1, grids$Var2)
```

Scale log likelihood to normalize it.

```
val_log_likelihood = val_log_likelihood - max(val_log_likelihood)
```

Plot scatterplot.

```
par(mfrow=c(1,2))
scatterplot3d(grids$Var1, grids$Var2, val_log_likelihood, color="lightblue",pch=21,
xlab=expression(beta[0]), ylab=expression(beta[1]), zlab="", main="Log likelihood after normalization")
scatterplot3d(grids$Var1, grids$Var2, exp(val_log_likelihood), color="lightblue",pch=21, xlab=expression(be
ta[0]), ylab=expression(beta[1]), zlab="", main="Likelihood after normalization")
```


Log likelihood after normalization

Likelihood after normalization

Q3. Use the $optim$ function in R to find both the mode and the Hessian at the mode for the logarithm of the posterior density of β. Construct the corresponding Multivariate Normal approximation to the posterior of β (hint: read pages 83 and 84 in $BayesianDataAnalysis$). Provide a visualization comparing this approximation to the actual posterior of β.

Build posterior first

```
log_posterior = function(beta) {
  return (log(prior_beta_0(beta[1]))
        + log(prior_beta_1(beta[2]))
        + log_likelihood(beta[1], beta[2]))
}
```

Compute optimal value and get mode, hessian matrix, covariance matrix

```
optimal = optim(c(0, 0), log_posterior, control=list(fnscale=-1), hessian=TRUE)

u = optimal$par
hessian = optimal$hessian
cov = -solve(hessian)
u
```

```
## [1] -1.3785813 -0.5412964
```

```
hessian
```

```
##           [,1]      [,2]
## [1,] -12.63266   3.59332
## [2,]   3.59332 -12.82534
```

```
cov
```

```
##            [,1]       [,2]
## [1,] 0.08601477 0.02409906
## [2,] 0.02409906 0.08472257
```

Q4. Draw 1000 values of β from its posterior using a discrete grid approximation to the posterior. Be sure to describe your choice of discrete grid (hint: the previous step can be helpful in this regard).

```
num_draws = 1000
```

Rewrite the posterior using vector input to sample data

```
sample_posterior = function(beta) exp(log_posterior(beta))
```

Use MCMC to draw samples from density with multiple parameters

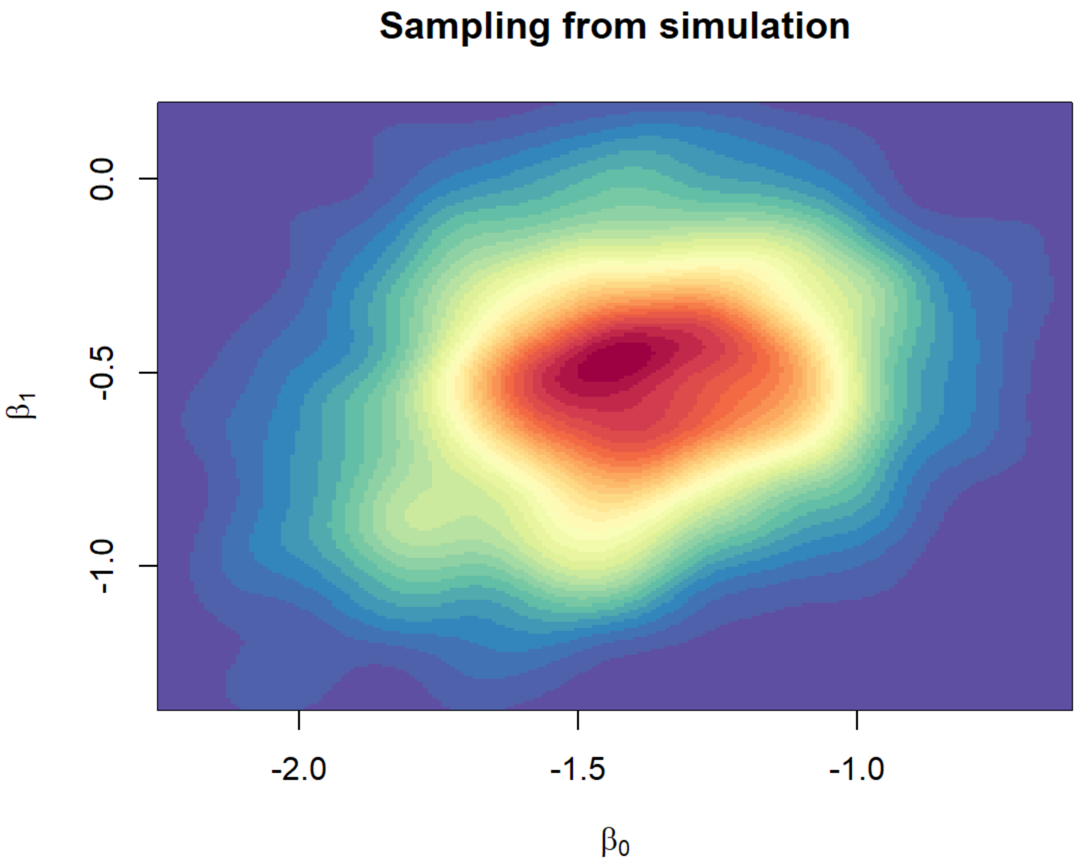
```
# library(DPpackage)
# mcmc = list(nburn=2000, nsave=num_draws, ndisplay=500)
# grids = expand.grid(seq(0, 0.5, 0.01), seq(0, 0.5, 0.01))
# support = cbind(grids$Var1, grids$Var2)
# fit = PTsampler(sample_posterior, dim.theta=2, mcmc=mcmc, support=support)
# samples_mcmc_posterior = fit$save.state$thetasave
```

Use sample function to sample from density function

```
trials = 100000
seq0 = sample(seq(u[1] - sqrt(cov[1,1]) * 3, u[1] + sqrt(cov[1,1]) * 3, 0.0001), replace=TRUE, size = trials)
seq1 = sample(seq(u[2] - sqrt(cov[2,2]) * 3, u[2] + sqrt(cov[2,2]) * 3, 0.0001), replace=TRUE, size = trials)
grid = rbind(seq0,seq1)
den_prop = apply(grid, 2, sample_posterior)
idx = sample(seq(1, trials), size =1000,replace = TRUE, prob = den_prop)

samples_posterior = t(grid[,idx])
```

```
library(RColorBrewer)
rf = colorRampPalette(rev(brewer.pal(11,'Spectral')))
r = rf(32)
library(MASS)
k = kde2d(samples_posterior[,1], samples_posterior[,2], n=200)
image(k, col=r, ylab=expression(beta[1]), xlab=expression(beta[0]), main="Sampling from simulation")
```



Q5. Draw 1000 values of β from the Multivariate Normal approximation to the posterior.

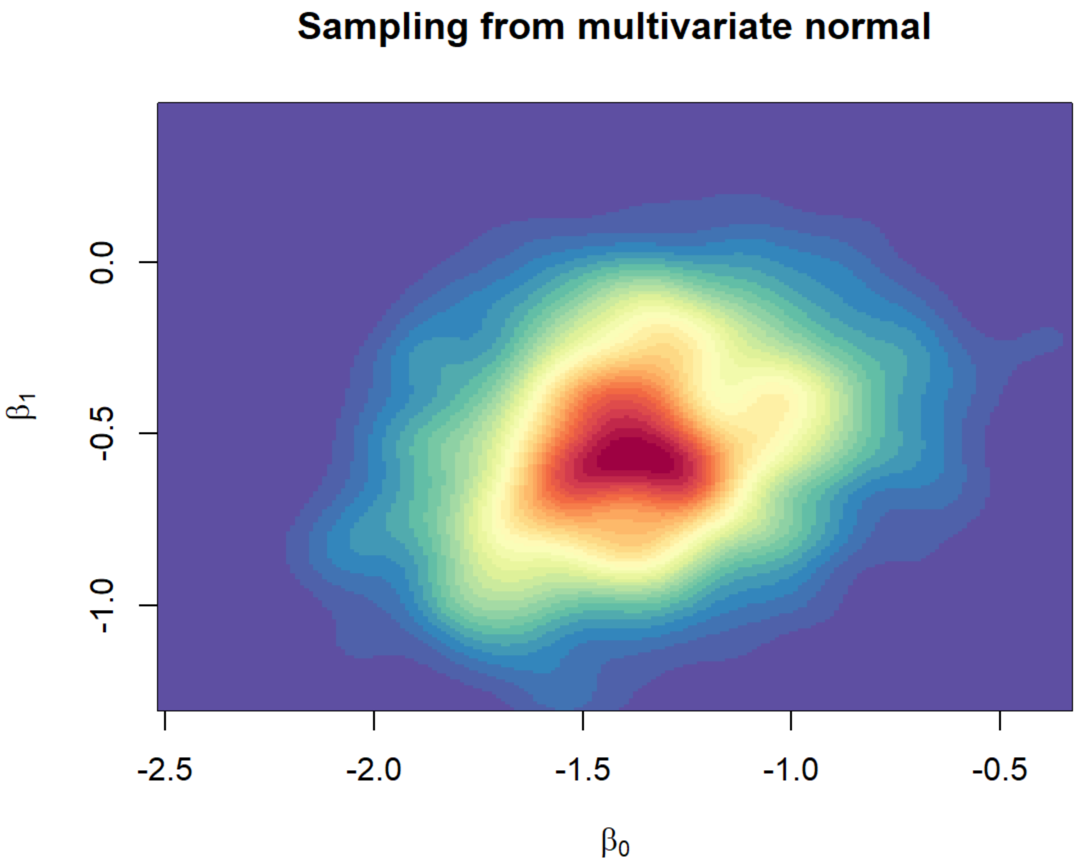
We can sample directly from multivariate normal

```
# library(mvtnorm)
# samples_2d_normal = rmvnorm(num_draws, myMode, sqrt(covariance))
```

Or indirectly by first sampling two series of independent standard normal distributions

```
C = chol(cov)
mu = matrix(rep(u, num_draws), nrow=2)
yi = rbind(rnorm(num_draws), rnorm(num_draws))
samples_2d_normal = t(mu + C %*%  yi)

par(mfrow=c(1,1))
k = kde2d(samples_2d_normal[,1], samples_2d_normal[,2], n=200)
image(k, col=r, ylab=expression(beta[1]), xlab=expression(beta[0]), main="Sampling from multivariate norma
l")
```



Q6. For each entry in β, compare the two sets of draws by calculating the following summaries for each: 0.025, 0.25, 0.5, 0.75, 0.975 quantiles, mean, standard deviation, skewness, and kurtosis.

```
library(moments)
mySummary = function(dat) {
  values = round(c(quantile(dat, probs=c(0.025,0.25,0.5,0.75,0.975)),
             mean(dat[,1]), mean(dat[,2]), sd(dat[,1]), sd(dat[,2]), skewness(dat), kurtosis(dat)), 4)
  return(setNames(values, c("2.5%", "25%", "50%", "75%", "97.5%",
                            "u_beta_0", "u_beta_1", "sd_beta_0", "sd_beta_1", "skew_1", "skew_2", "kurt_1",
 "kurts_2")))
}

mySummary(samples_posterior)
```

```
##      2.5%       25%       50%       75%     97.5%  u_beta_0  u_beta_1
##   -1.8950   -1.4213   -1.0026   -0.5390   -0.0814   -1.4230   -0.5450
## sd_beta_0 sd_beta_1    skew_1    skew_2    kurt_1   kurts_2
##    0.2819    0.2858   -0.0729   -0.0841    2.7908    2.7522
```

```
mySummary(samples_2d_normal)
```

```
##      2.5%       25%       50%       75%     97.5%  u_beta_0  u_beta_1
##   -1.8673   -1.3744   -0.9243   -0.5493   -0.0802   -1.3658   -0.5395
## sd_beta_0 sd_beta_1    skew_1    skew_2    kurt_1   kurts_2
##    0.3099    0.2801    0.1488    0.0532    3.0903    2.8684
```

Q7. Summarize your inferences on $exp(\beta_1)$ based on the above two sets of draws. Describe the merits (if any) of the Multivariate Normal approximation for the posterior distribution in this problem.

Drawing samples from multivariate normal is quite fast, from the two summaries below, most of the attributes are better

```
mySummary = function(dat) {
  values = round(c(quantile(dat, probs=c(0.025,0.25,0.5,0.75,0.975)),
                   mean(dat), sd(dat), skewness(dat), kurtosis(dat)), 4)
  return(setNames(values, c("2.5%", "25%", "50%", "75%", "97.5%", "mean", "sd", "skew", "kurt")))
}

# inference for exp(beta_1)
mySummary(exp(samples_posterior[,2]))
```

```
##   2.5%    25%    50%    75%  97.5%   mean     sd   skew   kurt 
## 0.3305 0.4790 0.5835 0.7062 1.0175 0.6038 0.1729 0.6637 3.3477
```

```
mySummary(exp(samples_2d_normal[,2]))
```

```
##   2.5%    25%    50%    75%  97.5%   mean     sd   skew   kurt 
## 0.3425 0.4787 0.5772 0.7041 0.9971 0.6064 0.1741 0.9039 4.4946
```

Q8. Define y^{rep} as replicated data that we could see if the process that produced today's data were replicated with the same model, treatment levels, and value of β that produced the observed data. Describe how you would approximate the posterior predictive distribution of yrep using simulation. Perform this approximation and summarize your draws with a plot of the posterior predictive means and 95% central posterior predictive intervals of y^{rep} as a function of the treatment.

Use samples of (β_0, β_1) drawn from posterior to generate odd ratios.

```
simulated_odd = matrix(NA, nrow = num_draws, ncol = length(y))
for (i in 1: num_draws)
  simulated_odd[i,] = odd_ratio(samples_posterior[i,1], samples_posterior[i,2])
```

Compute credit interval:

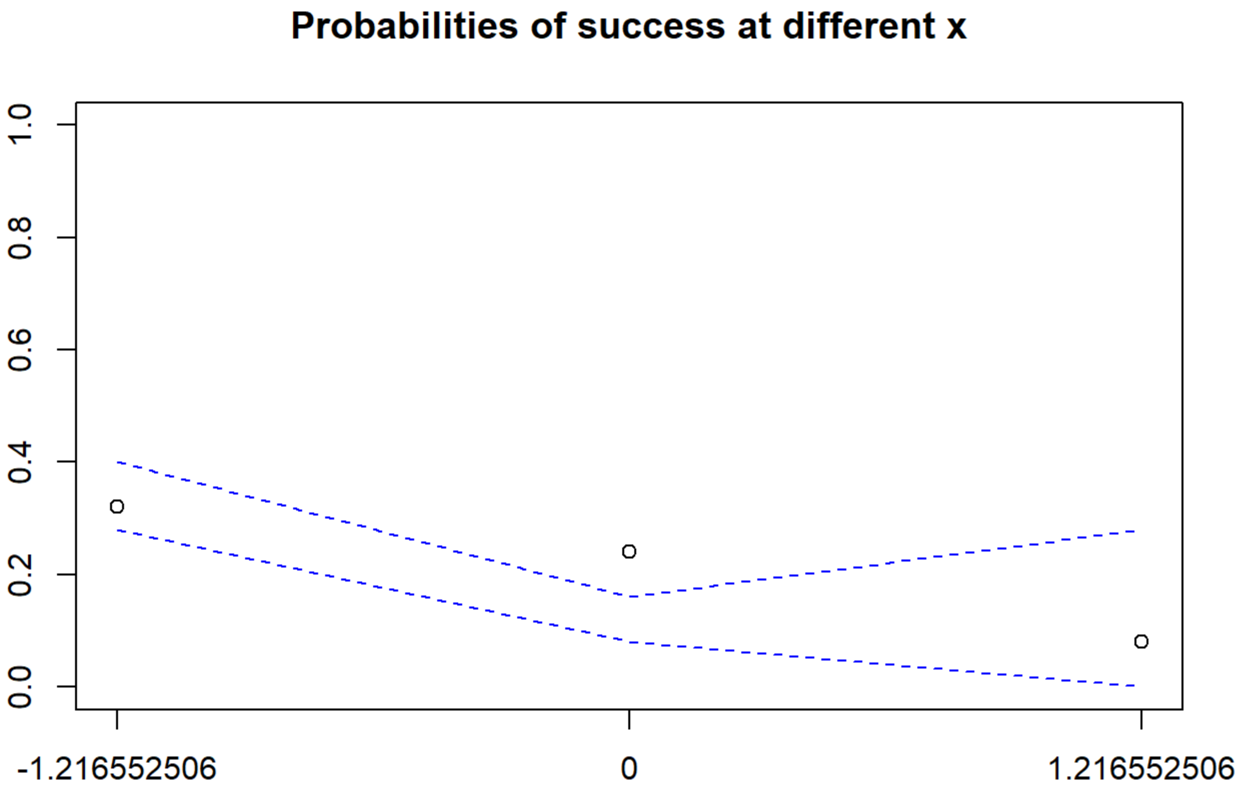
```
simulation = list(mu = c(), up = c(), low = c())
for (i in 1:length(y)) {
  simulation$mu = c(simulation$mu, rbinom(1, 1, mean(simulated_odd[,i])))
  simulation$up = c(simulation$up, rbinom(1, 1, quantile(simulated_odd[,i], probs=0.975)))
  simulation$low = c(simulation$low, rbinom(1, 1, quantile(simulated_odd[,i], probs=0.025)))
}
```

Evaluate the result and visualize it.

```
simulation$x = x
simulation$y = y
eval = list(p_real=c(), p_mu=c(), p_up=c(), p_low=c())
unique_v = c(simulation$x[1], 0, simulation$x[71])

for (v in unique_v) {
    eval$p_real = c(eval$p_real, sum(simulation$y[simulation$x==v]) /
length(simulation$y[simulation$x==v]))
    eval$p_mu = c(eval$p_mu, sum(simulation$mu[simulation$x==v]) / length(simulation$mu[simulation$x==v]))
    eval$p_up = c(eval$p_up, sum(simulation$up[simulation$x==v]) / length(simulation$up[simulation$x==v]))
    eval$p_low = c(eval$p_low, sum(simulation$low[simulation$x==v]) /
length(simulation$low[simulation$x==v]))
}

plot(eval$p_real, pch=21, ylim=c(0, 1), xaxt="n", ylab="", xlab="", main="Probabilities of success at diffe
rent x")
axis(1, at=1:3, labels=unique_v)
lines( eval$mu)
points(eval$p_up, type="l", col="blue", lty=2)
points(eval$p_low, type="l", col="blue", lty=2)
```



Q9.Compare the observed data to these posterior predictive summaries. What can you say about the model fit? Does the model appear appropriate? If not, how you would suggest modifying the model?

The real probability of success at $x = 0$ is a little higher than predicted, the model is basically correct. It seems that if we change the mean of $beta_1$ from log(1.5) to log(2), the result would better match the real data.

Compute posterior and sample from it.

```
prior_beta_0 = function(beta_0) dnorm(beta_0, log(0.3), sigma_0)
prior_beta_1 = function(beta_1) dnorm(beta_1, log(2.0), sigma_1)

sample_posterior = function(beta) {
  return (exp(log(prior_beta_0(beta[1]))
        + log(prior_beta_1(beta[2]))
        + log_likelihood(beta[1], beta[2])))
}
inner_trials = 100000
seq0 = sample(seq(u[1] - sqrt(cov[1,1]) * 3, u[1] + sqrt(cov[1,1]) * 3, 0.0001), replace=TRUE, size = inner_trials)
seq1 = sample(seq(u[2] - sqrt(cov[2,2]) * 3, u[2] + sqrt(cov[2,2]) * 3, 0.0001), replace=TRUE, size = inner_trials)
grid = rbind(seq0,seq1)
den_prop = apply(grid, 2, sample_posterior)
idx = sample(seq(1,inner_trials), size =1000, replace = TRUE, prob = den_prop)
samples_posterior = t(grid[,idx])
```

Compute credit interval

```
simulation_samples = matrix(NA, nrow = num_draws, ncol = length(y))

for (i in 1: num_draws)
  simulation_samples[i,] = odd_ratio(samples_posterior[i,1], samples_posterior[i,2])

simulation = list(mu = c(), up = c(), low = c())
for (i in 1:length(y)) {
  simulation$mu = c(simulation$mu, rbinom(1, 1, mean(simulation_samples[,i])))
  simulation$up = c(simulation$up, rbinom(1, 1, quantile(simulation_samples[,i], probs=0.975)))
  simulation$low = c(simulation$low, rbinom(1, 1, quantile(simulation_samples[,i], probs=0.025)))
}
```

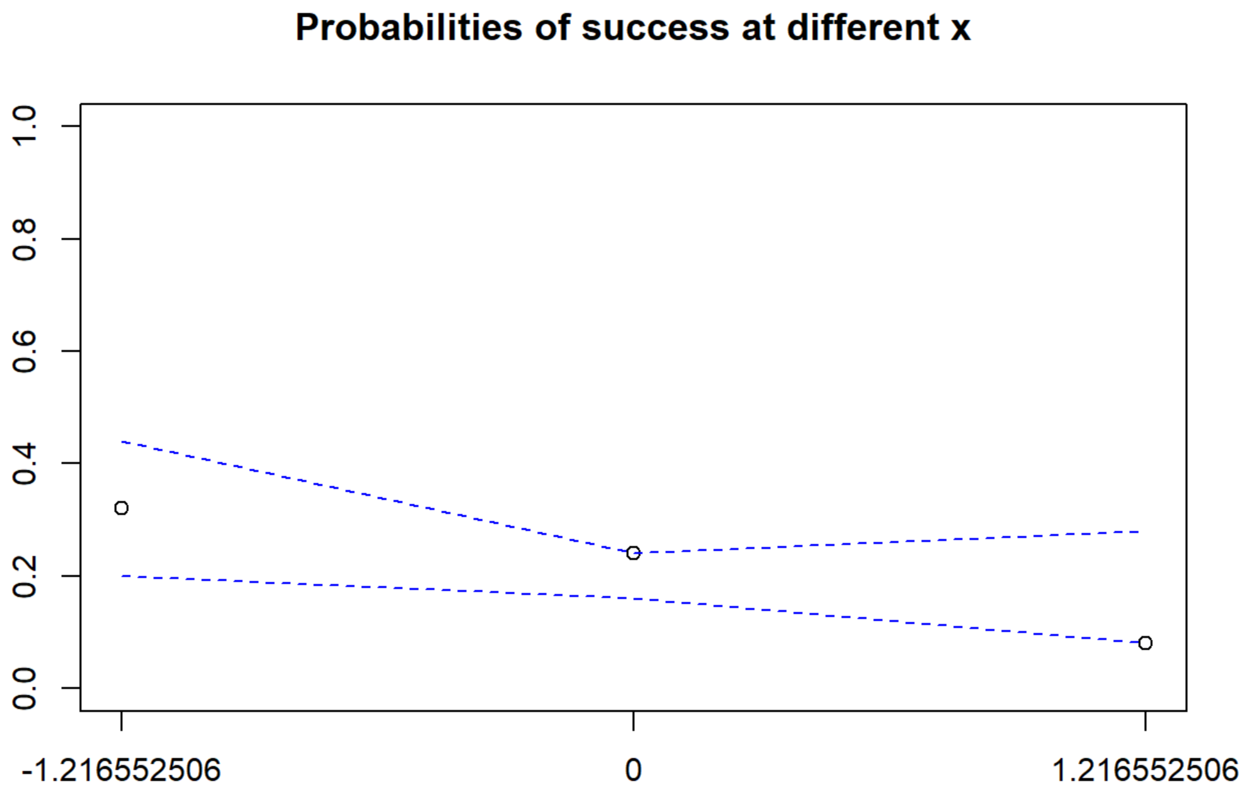
Evaluate the result and make visualization.

```
simulation$x = x
simulation$y = y
eval = list(p_real=c(), p_mu=c(), p_up=c(), p_low=c())

unique_v = c(simulation$x[1], 0, simulation$x[71])

for (v in unique_v) {
    eval$p_real = c(eval$p_real, sum(simulation$y[simulation$x==v]) /
length(simulation$y[simulation$x==v]))
    eval$p_mu = c(eval$p_mu, sum(simulation$mu[simulation$x==v]) / length(simulation$mu[simulation$x==v]))
    eval$p_up = c(eval$p_up, sum(simulation$up[simulation$x==v]) / length(simulation$up[simulation$x==v]))
    eval$p_low = c(eval$p_low, sum(simulation$low[simulation$x==v]) /
length(simulation$low[simulation$x==v]))
}

plot(eval$p_real, pch=21, ylim=c(0, 1), xaxt="n", ylab="", xlab="", main="Probabilities of success at different x")
axis(1, at=1:3, labels=unique_v)
lines( eval$mu)
points(eval$p_up, type="l", col="blue", lty=2)
points(eval$p_low, type="l", col="blue", lty=2)
```



Applied question

A researcher in Purdue University's College of Health and Human Sciences is conducting a study to shed light on less medieval remedial policy measures and reforms to reduce the incidence of teenage preg-nancy. She is particularly interested to learn whether *controllable* factors (e.g., parental supervision) play a significant role on teenage pregnancy. The researcher's data for her study consisted of survey responses on 2006 teenage girls in grades 7 - 12 from 1995 - 1996 obtained from the National Longitudinal Study of Adolescent Health. Their first interviews (referred to as Wave I) took place during the 1995 school year, and their second interviews (referred to as Wave II) took place the following year. Nearly 500 controllable and uncontrollable variables, measuring a broad spectrum of characteristics such as race, the importance of religion, and the amount of time spent each week playing sports, were collected for each respondent. The response variable for each teenage girl is an indicator for whether she became pregnant during Wave II. Of the 2006 respondents, 105 got pregnant during Wave II. The researcher sent you a subset of her collected data, contained in pregnancy data subset.csv, and requests that you analyze it to assess the association between parental supervision and teenage pregnancy. This data consists of the following:

- mother_op_sex: how the respondent thinks her mother would feel about her having sex at this time in her life (0 if disapprove, 1 if approve)
- night_wo_perm: indicator for whether the respondent spent a night away from home without permission in the last 12 months (0 if no, 1 if yes)
- preg: indicator for whether the respondent was pregnant or not during Wave II (0 if no, 1 if yes)

Q1. Explain why the researcher used as her response variable whether a respondent got pregnant during Wave II, as opposed to whether a respondent got pregnant during Wave I.

Because Wave II happens one year later after Wave I, there would be more pregnant respondent. In addition, the predictor variable was collected from Wave II, therefore, using response variable in wave II is more related to variable for whether the respondent spent a night away from home without permission in the last 12 months.

Q2. Do you believe that this data is suficient to address the researcher’s questions? Why or why not? If not, what other data would you request from the researcher?

Roughly sufficient, even though we don’t have too much knowledge about priors on variables_mother_op_sex and night_wo_perm, we could still use non-informative prior, e.g. t/Cauchy/Lapace prior, to solve this problem.

Q3. Specify a parametric model for this data. Explain all the assumptions involved with your specified model, and whether you believe them to be valid. In particular, provide arguments in favor or against an assumption of exchangeability among respondents conditional on the given information.

Use the response and predictors are all binary, we will use binary logistic regression. Since we don’t have too much information about the priors, weekly informative prior Cauchy prior is used.

$$Pr(yi = 1|\beta; xi) = \frac{exp(\beta_0+\beta_1 mother\_op\_sex+\beta_2 night\_wo\_perm)}{1+exp(\beta_0+\beta_1 mother\_op\_sex+\beta_2 night\_wo\_perm)}$$

Q4. Identify possible estimand(s) of interest with respect to your model.

The estimands of interest are exp (beta_0), which is the odds of success for an experimental unit when two behaviors are not allowed, exp (beta_1), which captures the multiplicative change in the odds of success if $mother\_op\_sex = 1$, exp (beta_2), which captures the multiplicative change in the odds of success if $night\_wo\_perm = 1$

Q5. Specify a prior for all of the model parameters. You can choose any prior you like, but you must justify your choice based on information available before Wave II. Provide relevant citations and references for any information that you use to construct your prior.

This reference suggested to use Cauchy prior, since we don’t have too much information about priors, reference is here (https://tgmstat.wordpress.com/2014/03/19/weakly-informative-priors-for-logistic-regression/)

```r
set.seed(26)
setwd("C:/Users/Wei/Desktop/STAT 695/HW1")
data = read.csv(file="pregnancy_data_subset.csv", header=TRUE)
```

Choose Cauchy prior

```r
prior_intercept = function(beta_0) dcauchy(beta_0)
prior_mother_op_sex = function(beta_1) dcauchy(beta_1)
prior_night_wo_perm = function(beta_2) dcauchy(beta_2)
```

Build posterior

```r
p_ratio = function(beta_0, beta_1, beta_2) 1 / (1 + exp(- beta_0 - beta_1 * data$mother_op_sex - beta_2 *
data$night_wo_perm))
log_likelihood = function(beta_0, beta_1, beta_2) sum(log(dbinom(data$preg, 1, p_ratio(beta_0, beta_1, beta
_2))))

log_posterior = function(beta) {
  return (log(prior_intercept(beta[1]))
        + log(prior_mother_op_sex(beta[2]))
        + log(prior_night_wo_perm(beta[3]))
        + log_likelihood(beta[1], beta[2], beta[3]))
}
```

Compute the mode and covariance

```r
optimal = optim(c(0, 1, 1), log_posterior, control=list(fnscale=-1), method="L-BFGS-B", hessian=TRUE)
optimal
```

```
## $par
## [1] -3.2290229  0.9835995  1.1054136
##
## $value
## [1] -398.2499
##
## $counts
## function gradient
##       16       16
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
##
## $hessian
##           [,1]       [,2]       [,3]
## [1,] -97.16046 -24.625232 -20.856399
## [2,] -24.62523 -24.642041  -8.262012
## [3,] -20.85640  -8.262012 -20.766492
```

```
u = optimal$par
hessian = optimal$hessian
cov = solve(-hessian)
```

Q6. Calculate and provide a visualization of the posterior distribution(s) of the estimand(s). Summarize the posterior distribution(s) numerically.

```
trials = 10000
seq1 = sample(seq(u[1]-sqrt(cov[1,1])*3, u[1]+sqrt(cov[1,1])*3, 0.0001), replace=TRUE, size = trials)
seq2 = sample(seq(u[2]-sqrt(cov[2,2])*3, u[2]+sqrt(cov[2,2])*3, 0.0001), replace=TRUE, size = trials)
seq3 = sample(seq(u[3]-sqrt(cov[3,3])*3, u[3]+sqrt(cov[3,3])*3, 0.0001), replace=TRUE, size = trials)
# grid = rbind(rep(u[1], trials), seq2, seq3)
grid = rbind(seq1, seq2, seq3)
dLogProp = apply(grid, 2, log_posterior)
cNormorlize = max(dLogProp)

sample_posterior = function(beta) exp(log_posterior(beta) - cNormorlize)
dProp = apply(grid, 2, sample_posterior)
idx = sample(seq(1, trials), size =1000, replace = TRUE, prob = dProp)
samples_posterior = t(grid[,idx])
vals = dProp[idx]

library(threejs)
```

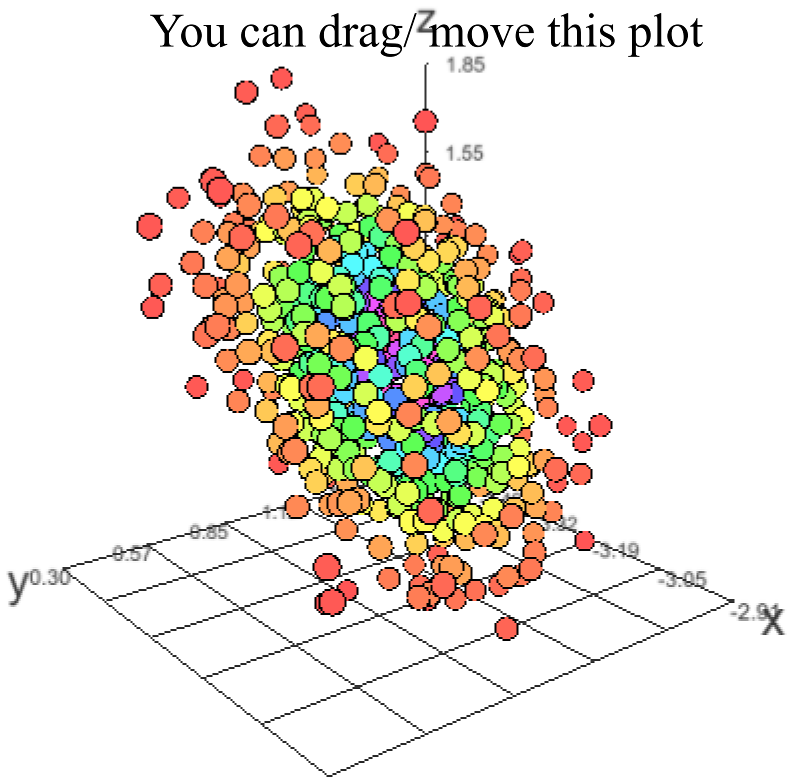
```
## Loading required package: igraph
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
ra = ceiling(256 * vals / max(vals))
col = rainbow(256, 2/3)

scatterplot3js(x=samples_posterior[,1], y=samples_posterior[,2], z=samples_posterior[,3], size=0.4, color
= col[ra], main="You can drag/ move this plot")
```



The summary of posterior is as follows

```
library(moments)
c(mean(samples_posterior[,1]), mean(samples_posterior[,2]), mean(samples_posterior[,3]))
```

```
## [1] -3.2340991  0.9784608  1.0947297
```

```
var(samples_posterior)
```

```
##              seq1         seq2         seq3
## seq1  0.013639692 -0.009906662 -0.009105411
## seq2 -0.009906662  0.055944470 -0.013800255
## seq3 -0.009105411 -0.013800255  0.064760199
```

```
skewness(samples_posterior)
```

```
##       seq1       seq2       seq3 
## -0.1132591 -0.0729713 -0.1549070
```

```
kurtosis(samples_posterior)
```

```
##     seq1     seq2     seq3 
## 2.842814 2.925106 2.857610
```

Q7. Conduct a posterior predictive check to diagnose the fit of your model. Does your model appear appropriate? If not, modify it to address the observed inadequacies, and perform a posterior predictive check on the new model to confirm that it provides a better fit to the observed data.

Use the samples drawn from posterior to simulate data from the function of binary response.

```
num_draws = 1000
simulated_p = matrix(NA, nrow = num_draws, ncol = dim(data)[1])

for (i in 1: num_draws)
  simulated_p[i,] = p_ratio(samples_posterior[i,1], samples_posterior[i,2], samples_posterior[i,3])

simulation = list(mu = c(), up = c(), low = c())

for (i in 1: dim(data)[1]) {
  simulation$mu = c(simulation$mu, mean(simulated_p[,i]))
  simulation$up = c(simulation$up, quantile(simulated_p[,i], probs=0.975))
  simulation$low = c(simulation$low, quantile(simulated_p[,i], probs=0.025))
}
```

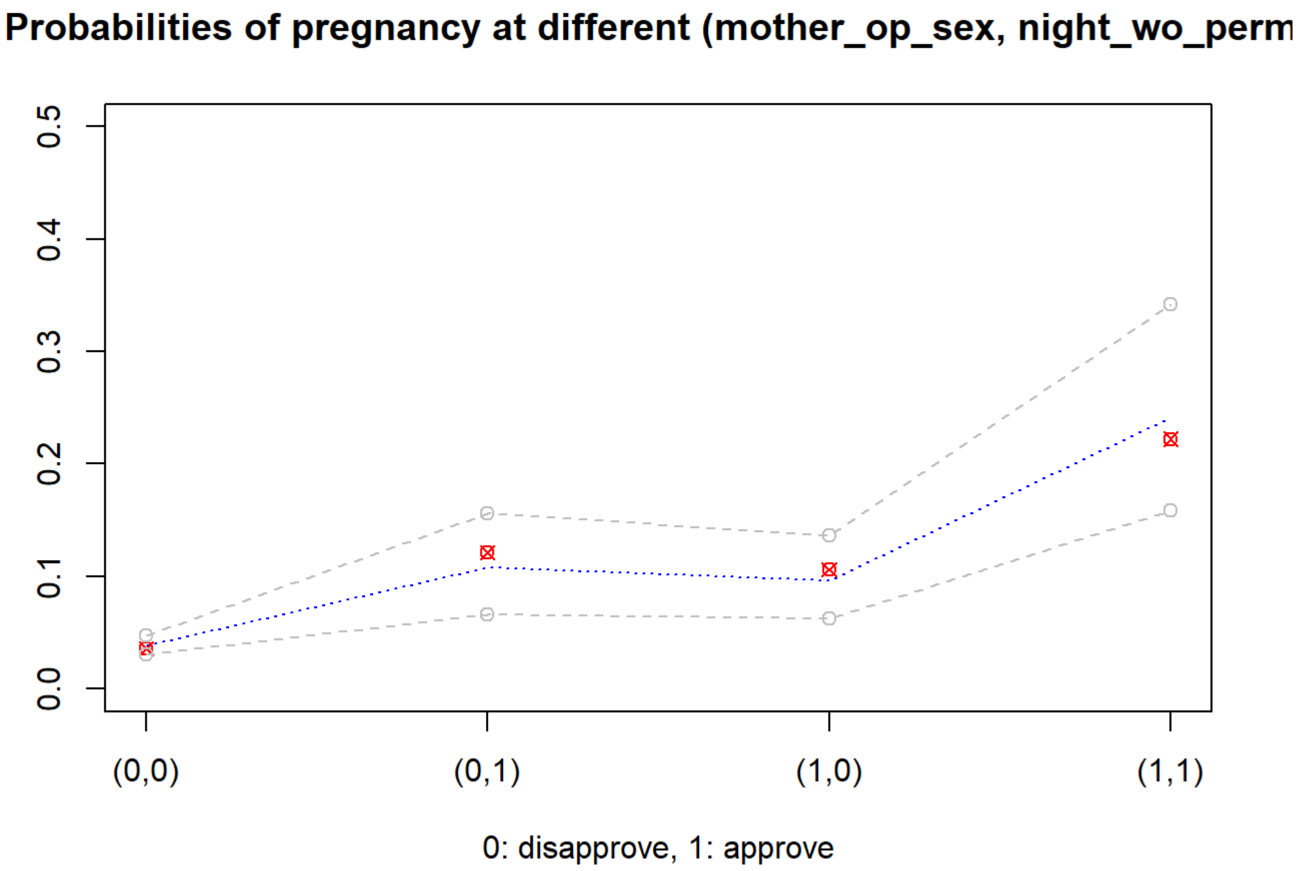
Compute approximated probabilities using random sampled data

```
xx = data$mother_op_sex
yy = data$night_wo_perm
simulation$real = data$preg

eval = list(p_real=c(), p_mu=c(), p_low=c(), p_up=c())
unique_v = c(0, 1)
for (v in unique_v) {
  for (w in unique_v) {
    eval$p_real = c(eval$p_real, mean(simulation$real[xx == v & yy == w]))
    eval$p_mu = c(eval$p_mu, mean(simulation$mu[xx == v & yy == w]))
    eval$p_low = c(eval$p_low, mean(simulation$low[xx == v & yy == w]))
    eval$p_up = c(eval$p_up, mean(simulation$up[xx == v & yy == w]))
  }
}
```

Visualization

```
plot(eval$p_real, pch=13, col="red", ylim=c(0, 0.5), xaxt="n", ylab="", xlab="0: disapprove, 1: approve", m
ain="Probabilities of pregnancy at different (mother_op_sex, night_wo_perm)")
axis(1, at=1:4, labels=c("(0,0)", "(0,1)", "(1,0)", "(1,1)"))
points(eval$p_mu, type="l", col="blue", lty=3)
points(eval$p_up, type="o", col="grey", lty=2)
points(eval$p_low, type="o", col="grey", lty=2)
```



From the figure shown above, the model seems quite appropiate.

Q8. Provide a non-technical explanation of your findings for the researcher, who has minimal statistical training.

Parental supervision could reduce teenage pregnancy rate as much as 30% if her mother disapprove her having sex at this time in her life and disapprove teenage spent a night away from home.

Theory

1. Define the Kullback-Leibler (KL) divergence $KL(p(y)||q(y))$ between two probability densities p(y) and q(y). Explain why the KL divergence is not a metric.

Metric should be symmetric, however $D_{KL}(P||Q) = D_{KL}(Q||P)$ means $\int Plog(P)\ dx - \int PlogQ\ dx = \int Qlog(Q)\ dx - \int QlogP\ dx$, this gives $\int log(P)\ dx = \int log(Q)\ dx$, however, when P is not identical to Q, the equation doesn't always hold. This suggests that KL divergence is not a metric.

2. In Question 3 of the Computation problem, you were tasked with approximating a probability density using a Normal distribution. Another approach to approximate a density $p(y)$ using a Normal distri-bution with mean μ and σ_2 is to choose μ and σ^2 so as to minimize $KL(p(y)||N(y|(\mu, \sigma^2))$, where $N(y|(\mu, \sigma^2))$ denotes the Normal probability density function. Derive the expressions for the optimum μ and σ_2. What are the advantages of this approximation compared to that in the Computation problem? What are the disadvantages?

Let $q(y)$ denote the probability density function for a $N(\mu, \sigma^2)$ random variable. Then

$$KL(p(y)||q(y)) = \int p(y)logp(y)dy - \int p(y)\{-\frac{(y-u)^2}{2\sigma^2} - \frac{log(2\pi\sigma^2)}{2}\}dy$$

Thus, $\frac{d}{d\mu}KL(p(y)||q(y)) = 0$ if and only if $\int p(y)\frac{y-\mu}{\sigma^2}dy = 0$ if and only if $\mu = \int yp(y)dy$. Similarly, $\frac{d}{d\sigma^2}KL(p(y)||q(y)) = 0$ if and only if $\sigma^2 = \int (y-\mu)^2 p(y)dy$.

3. Suppose that p(y) has support on the positive real line, and you wish to approximate it using an Exponential distribution. Describe how will you set the parameter of the Exponential distribution. Do you believe that the Exponential distribution provides a sufficiently flexible approximation? Why or why not? If not, suggest an alternative to the Exponential distribution.

Let q(y) denote the probability density function for an Exponential random variable with rate parameter λ. Then

$$KL(p(y)||q(y)) = \int p(y)log(p(y))dy + \int p(y)(\lambda y)dy - \lambda$$

Thus, $\frac{d}{d\lambda}KL(p(y)||q(y)) = 0$ if and only if $\int yp(y)dy = \lambda$. The Exponential distribution may not provide a sufficiently flexible approximation, and you may wish to specify the variance too, for example, by using the Gamma distribution instead.