LSTAT2130 Project

Nicephore Bayekula, Justin Davaux, Patrick Guerin

May 27, 2018

1 Responses

1. The general equality for our regression is

$$log(\mu_i(x_i, z_i)) = (\beta_0 + \alpha z_i) + (\beta_1 + \tau z_i)(x_i - 39). \tag{1}$$

Setting $z_i = 0$ and $x_i = 39$, we obtain the equality

$$log(\mu_i(x_i, z_i)) = \beta_0. \tag{2}$$

Therefore, β_0 can be interpreted as *the natural logarithm of the expected number of days of absence of a thirty-nine-year old man*. (So, $exp(\beta_0)$ is the expected number of days of absence of a thirty-nine-year old man.)
Simply setting $z_i = 0$, we obtain the equality

$$log(\mu_i(x_i, z_i)) = \beta_0 + \beta_1(x_i - 39). \tag{3}$$

Given that

$$\beta_1 = log(\mu_i(x_i + 1, z_i)) - log(\mu_i(x_i, z_i))\left(= log\left(\frac{\mu_i(x_i + 1, z_i)}{\mu_i(x_i, z_i)}\right)\right) \tag{4}$$

under the hypothesis $z_i = 0$, β_1 can be interpreted as *the increase of the natural logarithm of the expected number of days of absence of a man if he grows one year older*. (So, $exp(\beta_1)$ is the ratio of the expected number of days of absence of a man to the analogous number for a man who is one year younger.)
Setting $z_1 = 1$ and $x_i = 39$, we obtain the equality

$$\alpha = log(\mu_i(x_i, z_i)) - \beta_0. \tag{5}$$

Using our interpretation of β_0, we interpret α as the difference between *the natural logarithm of the expected number of days of absence of a thirty-nine-year old woman* and *the natural logarithm of the expected number of*

days of absence of a thirty-nine-year old man. (So, $exp(\alpha)$ is the ratio of the expected number of days of absence of a thirty-nine-year old woman to the analogous number for a thirty-nine-year old man.)
Simply setting $z_1 = 1$, we obtain the equality

$$\tau = \big(log(\mu_i(x_i + 1, z_i))\big) - log(\mu_i(x_i, z_i)) - \beta_1. \tag{6}$$

Using our interpretation of β_1, we interpret τ as the difference between *the increase of the natural logarithm of the expected number of days of absence of a woman if she grows one year older* and *the increase of the natural logarithm of the expected number of days of absence of a man if he grows one year older.*
2. Since the $Y_i's$ are independent, the likelihood function for the reduced sample (the sample without the discarded rows), is given by the product

$$\mathcal{L}(\alpha, \beta_0, \beta_1, \tau \mid \mathcal{D}) = \prod_{i=1}^{92} \frac{e^{-\mu_i(z_i,x_i)}\mu_i(z_i, x_i)^{y_i}}{y_i}$$

$$= \prod_{i=1}^{92} \frac{e^{-[e^{(\beta_0+\alpha z_i)+(\beta_1+\tau z_i)(x_i-39)}]}}{y_i!}[e^{(\beta_0+\alpha z_i)+(\beta_1+\tau z_i)(x_i-39)}]^{y_i}.$$

Its (natural) logarithm equals

$$\sum_{i=1}^{92} -[e^{(\beta_0+\alpha z_i)+(\beta_1+\tau z_i)(x_i-39)}] + y_i((\beta_0 + \alpha z_i) + (\beta_1 + \tau z_i)(x_i - 39)) - log(y_i!).$$

```
loglikelihood <- function(V){
  alpha <- V[1]
  beta_0 <- V[2]
  beta_1 <- V[3]
  tau <- V[4]

  q <- exp(beta_0+alpha*z+(beta_1+tau*z)*(x-39))
  result <- sum((-q)+y*log(q)-log(factorial(y)))
  return(result)
}
```

Since the R function *nlm* performs minimisation, we must apply it to the opposite of the logarithm of the likelihood rather than to the logarithm of likelihood itself. Having used the R function nlm with the following options:

Options	Values
Starting parameters	$(0, 0, 0, 0)$
Gradtol	$1e - 6$
fscale	1
stepmax	10
iterlim	50

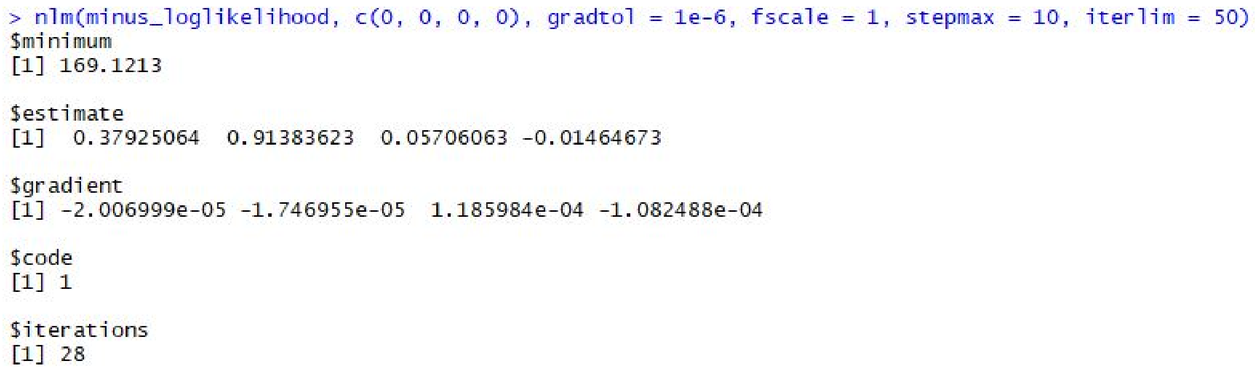
,

we obtained the following maximum likelihood estimators (rounded to the nearest hundredth):

Parameters	Values
α	0.38
β_0	0.91
β_1	0.06
τ	-0.01

.

Here is the R output produced. According to R documentation, code 1 suggests that current iterate is probably solution because 'relative gradient is close to zero'.



```
> nlm(minus_loglikelihood, c(0, 0, 0, 0), gradtol = 1e-6, fscale = 1, stepmax = 10, iterlim = 50)
$minimum
[1] 169.1213

$estimate
[1]  0.37925064  0.91383623  0.05706063 -0.01464673

$gradient
[1] -2.006999e-05 -1.746955e-05  1.185984e-04 -1.082488e-04

$code
[1] 1

$iterations
[1] 28
```

3. For simplicity, we chose a normal distribution with mean 0 and standard deviation 10^6 for each parameter, i.e.

$$\pi(p) = \frac{1}{10^6\sqrt{2\pi}} e^{\frac{-p^2}{2\times 10^{12}}} \tag{7}$$

for $p \in \{\alpha, \beta_0, \beta_1, \tau\}$. In order to obtain the joint prior, we combine the marginal priors multiplicatively as if the parameters were independent a priori:

$$\pi(\alpha, \beta_0, \beta_1, \tau) = \pi(\alpha)\pi(\beta_0)\pi(\beta_1)\pi(\tau).$$

```
prior <- function(V){
  A <- V[1]
  B <- V[2]
  C <- V[3]
  D <- V[4]
  cst <- 10^6
  result <- pnorm(A,0,cst)*pnorm(B,0,cst)*pnorm(C,0,cst)*pnorm(D,0,cst)
  return(result)
}
```

Relying on the previously created functions *loglikelihood* and *prior*, we create the required function logposterior as follows:

```
logposterior <- function(V){
  alpha <- V[1]
  beta_0 <- V[2]
  beta_1 <- V[3]
  tau <- V[4]
  result <- loglikelihood(V) + log(prior(V))
  return(result)
}
```

If we cannot rely on previously created functions, then we create the required R function as follows:

```
log_posterior <- function(V){

  alpha <- V[1]
  beta_0 <- V[2]
  beta_1 <- V[3]
  tau <- V[4]

  q <- exp(beta_0+alpha*z+(beta_1+tau*z)*(x-39))
  loglikelihood <- sum((-q)+y*log(q)-log(factorial(y)))

  cst <- 10^6
  prior <- pnorm(alpha,0,cst)*pnorm(beta_0,0,cst)
    *pnorm(beta_1,0,cst)*pnorm(tau,0,cst)
  result <- loglikelihood + log(prior)
  return(result)
}
```

Note that all the quoted pieces of code were preceded by

```
absences <- na.omit(absences) #Removal of rows with unavailable values

x <- absences[,3]
y <- absences[,1]
z <- absences[,2]
```

4. (a) In order to be able to adjust the acceptance rates conveniently, we wrote a Metropolis function which takes for arguments the number of iterations, the burn-in, the initial value of α, the initial value of β_0, the initial value of β_1 and the initial value of τ, and the four-dimensional vector of standard deviations of the normal 'disruptions' and returns samples and acceptance rates for each parameter. A *set.seed(2018)* statement precedes the body of the function code for reproducibility.

We evaluated the Metropolis function with

Options	Values
M	25000
burn-in	15000
alphainit	0.15
beta0_init	0.15
beta1_init	0.15
tau	0.15
std_proposal	$(0.22, 0.14, 0.01, 0.016)$

,

and obtained the following acceptance rates:

Parameter	Acceptance rate
α	42.32%
β_0	42.12%
β_1	42.49%
τ	41.68%

,

which are very close to the prescribed rate.

(b) Next we needed to check the convergence of the chains using two different criteria. First, we used Geweke's diagnostic, yielding the following z-scores:

Parameter	Geweke z-score
α	0.753
β_0	-0.4376
β_1	1.473
τ	-1.909

,

Since each z-score is located within the $[z_{0.025}, z_{0.975}] \simeq [-1.96, 1.96]$ interval, the null hypothesis that the chains converged cannot be rejected according to Geweke's diagnostic for a significance level of 5% (and we do not need to use geweke.plot to see how much of any sample must be removed before achieving stationarity.)

Secondly, we used Heidelberger-Welch's convergence diagnostic to check the convergence of the chains. The results are as follows:

Parameter	Stationarity test	Halfwidth test	p-value for stationarity test
α	passed	passed	0.0824
β_0	passed	passed	0.508
β_1	passed	passed	0.154
τ	passed	passed	0.121

,

hence the null hypothesis that the chains converged cannot be rejected according to Heidelberger-Welch's diagnostic for a significance level of 5%.

The reader will find the traceplots of the chains in the Appendix. They suggest that the chains converged rapidly.

(c) We got the following point estimates for the model parameters, rounded to the nearest thousandth, and the following lower and upper bounds for the 95% credible intervals (using the R function HDPinterval):

Parameter	Median	Mean	Lower bound	Upper bound
α	0.383	0.385	0.124	0.645
β_0	0.906	0.905	0.733	1.080
β_1	0.057	0.057	0.044	0.068
τ	-0.014	-0.015	-0.033	0.004

.

5. Geweke's diagnostic yields the following z-scores:

Parameter	Geweke z-score
α	1.311
β_0	-1.177
β_1	0.583
τ	-0.849

.

Since each z-score is located within the $[z_{0.025}, z_{0.975}] \simeq [-1.96, 1.96]$ interval, the null hypothesis that the chains converged cannot be rejected according to Geweke's diagnostic for a significance level of 5% (and we do not need to use geweke.plot to see how much of any sample must be removed before achieving stationarity.)

Secondly, we ran Heidelberger-Welch's convergence diagnostic to check the convergence of the chains. The results were as follows:

Parameter	Stationarity test	Halfwidth test	p-value for stationarity test
α	passed	passed	0.136
β_0	passed	passed	0.062
β_1	passed	passed	0.081
τ	passed	passed	0.114

,

hence the null hypothesis that the chains converged cannot be rejected according to Heidelberger-Welch's diagnostic for a significance level of 5%. The reader will find the traceplots of the chains in the Appendix. They suggest that the chains converged rapidly.

We got the following point estimates for the model parameters, rounded to the nearest thousandth, and the following lower and upper bounds for the 95% credible intervals (using the R function HDPinterval)

Parameter	Median	Mean	Lower bound	Upper bound
α	0.382	0.381	0.108	0.640
β_0	0.907	0.906	0.735	1.086
β_1	0.057	0.057	0.045	0.07
τ	-0.015	-0.015	-0.033	0.003

.

It turns out that the respective conclusions of Geweke and Heidelberger-Welch's diagnostics are the same as before, and that corresponding point estimates are extremely close: they are located within one hundredth of one unit of one another. Also, the 95% credible intervals are very close for each parameter across implementations for question 4 and 5. Also, a cursory look at the traceplots shows a similar pattern of quick convergence for all parameters across implementations for question 4 and 5. In conclusion, the two methods yielded very similar results.

6. We set out to work with the medians obtained by using JAGS.

Parameter	Median
α	0.382
β_0	0.907
β_1	0.057
τ	-0.015

.

In light of our interpretation of the coefficients, the retained point estimates lead to the conclusions outlined in the next facts.

Fact 1. $\beta_0 = 0.907$ implies that the expected number of days of absence of a thirty-nine-year old man is $exp(0.907)$, i.e. approximately 2.48.

Secondly,

Fact 2. $\beta_1 = 0.057$ implies that the ratio of the expected number of days of absence of a man to the analogous number for a man who is one year younger is $exp(0.057)$, i.e. approximately 1.06. So, age is positively related to the expected number of days of absence of a man: the older a man is, the more days of absence he is expected to have.

Thirdly,

Fact 3. $\alpha = 0.382$ implies that the ratio of the expected number of days of absence of a thirty-nine-year old woman to the analogous number for a thirty-nine-year old man is $exp(0.382)$, i.e. approximately 1.47.

Fourthly,

Fact 4. $\tau = -0.015$ and $\beta_1 = 0.057$ altogether imply that $exp(\tau+\beta_1) = 1.043$; by (6), this means that the ratio of the expected number of days of absence of a woman to the analogous number for a woman who is one year younger is approximately 1.043. So, age is positively related to the expected number of days of absence of a woman: the older a woman is, the more days of absence she is expected to have.

Facts 2 and 4 put together imply that age is positively related to the expected number of days of absence.

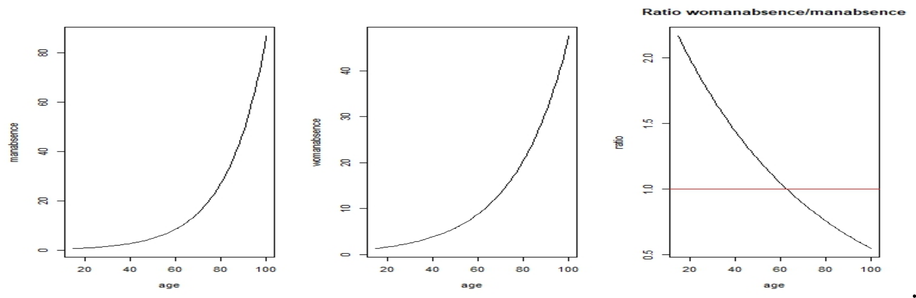
Consequently, if the function *manabsence* is the expected number of days of absence of a man as a function of age and the function *womanabsence* is the expected number of days of absence of a woman as a function of age,

$$manabsence(age) = 2.48 \times 1.06^{age-39} \tag{8}$$

and

$$womanabsence(age) = 2.48 \times 1.47 \times 1.043^{age-39}. \tag{9}$$

Here is a plot of the latter divided by the former:



.

This plot shows that, mostly, employment-aged women are expected to have more days of absence than men of the same age. So, while age is positively exponentially related to the expected number of days of absence, the female sex is more strongly associated with the number of days of absence than the male sex. Moreover, Fact 2 and Fact 3 show that the expected number of days of absence grows a bit faster (common ratio: 1.06) for men than for women (common ratio: 1.043).

7. (a) See code. (#7.(a))

(b) Here are the results we obtained:

subject	age	number of days of absence	percentage of ones in 'sex' sample	code of likely sex
93	30	1	(38)	(0)
94	59	4	(41)	(0)
95	20	1	(46)	(0)
96	29	6	(86)	(1)
97	(53)	6	0	0
98	(28)	1	100	1
99	(51)	5	0	0
100	(31)	2	100	1

.

The brackets mark the numbers that are results extracted from the samples produced to 'impute' the covariates. Since the proportions of 1's in the samples generated for the sex of subjects $93, 94$ and 95 are a little below

50%, they are a bit more likely to be men than women. By way of contrast, subject 96 is more likely to be a woman than a man because the proportion of 1's in the corresponding sample, which equals 86%, is a lot greater than 50.

It comes as no surprise that subjects 97 and 99 are likely to be relatively old (53 and 51 are the medians of the corresponding samples) because, while they are known to be men, they have high numbers of absences. Similarly, it comes as no surprise that subjects 98 and 100 are likely to be very young (28 and 31 are the medians of the correspondong samples) because, while they are known to be women, they have very low numbers of absences. These remarks rests on the observations of question 6.

The medians of the samples generated for the four parameters (in the usual sense), rounded to the nearest thousandth, are:

Parameter	Median
α	0.37
β_0	0.92
β_1	0.055
τ	-0.015

.

These numbers are very close to the ones we obtained in the model without the discarded data in question 6. So, the discarded data (only eight records among 100) had little influence on the parameters of the model.

2 Appendix

2.1 R Code

```
setwd("C:/Users/Nicephore/Documents/Q2/BayesianProject")
absences <-
  read.table("C:/Users/Nicephore/Documents/Q2/BayesianProject/absences.txt",
             header=TRUE, sep="", na.strings="NA", dec=".", strip.white=TRUE)

absences_whole_set <- absences
#Need to keep the whole data set for use in the last question

absences <- na.omit(absences) #Removal of rows with unavailable values

x <- absences[,3]
y <- absences[,1]
z <- absences[,2]

#2.
```

```
#This is the logarithm of the likelihood

loglikelihood <- function(V){
  alpha <- V[1]
  beta_0 <- V[2]
  beta_1 <- V[3]
  tau <- V[4]

  q <- exp(beta_0+alpha*z+(beta_1+tau*z)*(x-39))
  result <- sum((-q)+y*log(q)-log(factorial(y)))
  return(result)
}

#Trying to calculate the MLEs
#argmax f(x) = argmin ( -f(x) )

minus_loglikelihood <- function(V){
  result <- - loglikelihood(V)
  return(result)
}

nlm(minus_loglikelihood, c(0,0,0,0), gradtol = 1e-6,
  fscale = 1, stepmax = 10, iterlim = 50)



#Let's work with "uninformative" priors.

prior <- function(V){
  A <- V[1]
  B <- V[2]
  C <- V[3]
  D <- V[4]
  cst <- 10^6
  result <- pnorm(A,0,cst)*pnorm(B,0,cst)*pnorm(C,0,cst)*pnorm(D,0,cst)
  return(result)
}

#3.

logposterior <- function(V){

  alpha <- V[1]
  beta_0 <- V[2]
  beta_1 <- V[3]
  tau <- V[4]
  result <- loglikelihood(V) + log(prior(V))
  return(result)
}

#If one does not want to call another function
#within function logposterior then:

####logposterior <- function(V){

####  alpha <- V[1]
####  beta_0 <- V[2]
####  beta_1 <- V[3]
####  tau <- V[4]

####  q <- exp(beta_0+alpha*z+(beta_1+tau*z)*(x-39))
####  loglikelihood <- sum((-q)+y*log(q)-log(factorial(y)))
####  cst <- 10^6
####  prior <- pnorm(alpha,0,cst)*pnorm(beta_0,0,cst)
####  *pnorm(beta_1,0,cst)*pnorm(tau,0,cst)
####  result <- loglikelihood + log(prior)
####  return(result)
####}

#4. (a)

#Implementation inspired by an old assignment from a different module


#M: number of iterations
#burnin: number of initial elements of the samples which will be discarded
#in order to minimise the influence of the initial values fed to the Metropolis
#std_proposal: 4-vector of standard deviations of the normal 'disruptions'

Metropolis <- function( M,
                        burnin,
                        alphainit,
                        beta_0init,
                        beta_1init,
                        tau_init,
                        std_proposal
)
{
  set.seed(2018)
  result <- matrix(0, nrow = M, ncol = 4)
  accept <- matrix(0, ncol = 4, nrow = M)
  lcur <- logposterior( c(alphainit, beta_0init, beta_1init, tau_init))
  paramProp <- rep(0, 4)
  paramCur  <- c(alphainit, beta_0init, beta_1init, tau_init)

  for(cpt1 in 1:M){ # iteration loop
    for(cpt2 in 1:4) { # loop for number of parameters
      paramProp <- paramCur + rnorm(1,0, std_proposal[cpt2])*diag(1,4)[cpt2,]
      lprop <- logposterior(c(paramProp[1],paramProp[2],
            paramProp[3], paramProp[4]))

      if(runif(1) < min(1, exp(lprop - lcur))) {
        paramCur <- paramProp
        lcur <- lprop
        accept[cpt1, cpt2] <- 1
      }
    }
    result[cpt1, ] <- paramCur
  }

  return(list(alpha = result[(burnin+1):M,1],
              beta_0 = result[(burnin+1):M,2],
              beta_1 = result[(burnin+1):M,3],
              tau = result[(burnin+1):M,4],
              accept_Rate = list(alpha = mean(accept[(burnin+1):M,1]),
                                 beta_0 = mean(accept[(burnin+1):M,2]),
                                 beta_1 = mean(accept[(burnin+1):M,3])),
                                 tau = mean(accept[(burnin+1):M,4])))
}

M <- Metropolis(M=2.5*10^4, burnin = 1.5*10^4, alphainit = 0.15,
   beta_0init = 0.15, beta_1init = 0.15, tau_init = 0.15,
std_proposal = c(0.22, 0.14, 0.01, 0.016))

M

#All acceptance rates are between 41% and 43%

#Now, we have got to use two different criteria to
#check the convergence of the chains.

#(b).

library(coda)

MCMC_alpha <- mcmc(data = M$alpha)
MCMC_beta0  <- mcmc(data = M$beta_0)
```

```
MCMC_beta1 <- mcmc(data = M$beta_1)
MCMC_tau <- mcmc(data = M$tau)

geweke.diag(MCMC_alpha, frac1=0.1, frac2=0.5)
geweke.diag(MCMC_beta0, frac1=0.1, frac2=0.5)
geweke.diag(MCMC_beta1, frac1=0.1, frac2=0.5)
geweke.diag(MCMC_tau, frac1=0.1, frac2=0.5)


#The z-scores for all parameters  are located
#between -1.96 and 1.96, so the null hypothesis
#that the chains converged cannot be rejected according
#to Geweke diagnostic. (See p.15/26 in Chapter 3)

heidel.diag(MCMC_alpha)
heidel.diag(MCMC_beta0)
heidel.diag(MCMC_beta1)
heidel.diag(MCMC_tau)

#Traceplots

par(mfrow = c(2,2))
traceplot(MCMC_alpha, main = "alpha")
traceplot(MCMC_beta0, main = "beta_0")
traceplot(MCMC_beta1, main = "beta_1")
traceplot(MCMC_tau, main = "tau")

#All p-values are considerably larger 5%, so the null
#hypothesis that the chains converged cannot be rejected according
#to Heidelberger-Welch diagnostic.

#(c).

#Giving point estimates for each parameter

median(MCMC_alpha) #Estimator of the posterior mean of alpha
median(MCMC_beta0) #Estimator of the posterior mean of beta0
median(MCMC_beta1) #Estimator of the posterior mean of beta1
median(MCMC_tau) #Estimator of the posterior mean of tau

mean(MCMC_alpha) #Estimator of the posterior mean of alpha
mean(MCMC_beta0) #Estimator of the posterior mean of beta0
mean(MCMC_beta1) #Estimator of the posterior mean of beta1
mean(MCMC_tau) #Estimator of the posterior mean of tau

HPDinterval(MCMC_alpha, prob = 0.95) #HPD interval
```

```
#for the posterior distribution of alpha
HPDinterval(MCMC_beta0, prob = 0.95) #HPD interval
#for the posterior distribution of beta0
HPDinterval(MCMC_beta1, prob = 0.95) #HPD interval
#for the posterior distribution of beta1
HPDinterval(MCMC_tau, prob = 0.95) #HPD interval
#for the posterior distribution of tau


#5.

library(rjags)

M=length(x)
alpha <- rep(0,M)
beta_0 <- rep(0,M)
beta_1 <- rep(0,M)
tau <- rep(0,M)

mu = c()
beta_0
beta_1
tau

for (i in 1:length(x)){
  mu[i] =  exp(beta_0+alpha*z[i]+(beta_1+tau*z[i])*(x[i]-39))
}

mu


# initialization of the parameters


mod= "model{
for (i in 1:length(x)){
y[i] ~ dpois(mu[i])
mu[i] = exp(beta_0+alpha*z[i]+(beta_1+tau*z[i])*(x[i]-39))
}


# specify the priors (uninformative here)
alpha ~ dnorm(0,0.000001)
beta_0 ~ dnorm(0,0.000001)
beta_1 ~ dnorm(0,0.000001)
tau ~ dnorm(0,0.000001)
}"

alpha
beta_0
beta_1
tau

set.seed(1348)
colnames(absences)= c('y','z','x')
data = as.list(absences)
param=c("alpha","beta_0","beta_1","tau")

jags <- jags.model(textConnection(mod),
                   data = data,
                   n.chains = 4, # number of parallele chains
                   n.adapt = 1000 # burn-in
)
update(jags, 10000)


samples <- coda.samples(jags,
                        param,
                        10000)
#plot(samples)

samples
samples[[1]]
apply(samples[[1]],2,median)
apply(samples[[1]],2,mean)



S_alpha <- samples[[1]][,1]
S_beta0 <- samples[[1]][,2]
S_beta1 <- samples[[1]][,3]
S_tau <- samples[[1]][,4]

S_alpha <- mcmc(data = S_alpha)
S_beta0  <- mcmc(data = S_beta0)
S_beta1 <- mcmc(data = S_beta1)
S_tau <- mcmc(data = S_tau)

geweke.diag(S_alpha, frac1=0.1, frac2=0.5)
geweke.diag(S_beta0, frac1=0.1, frac2=0.5)
geweke.diag(S_beta1, frac1=0.1, frac2=0.5)
geweke.diag(S_tau, frac1=0.1, frac2=0.5)
```

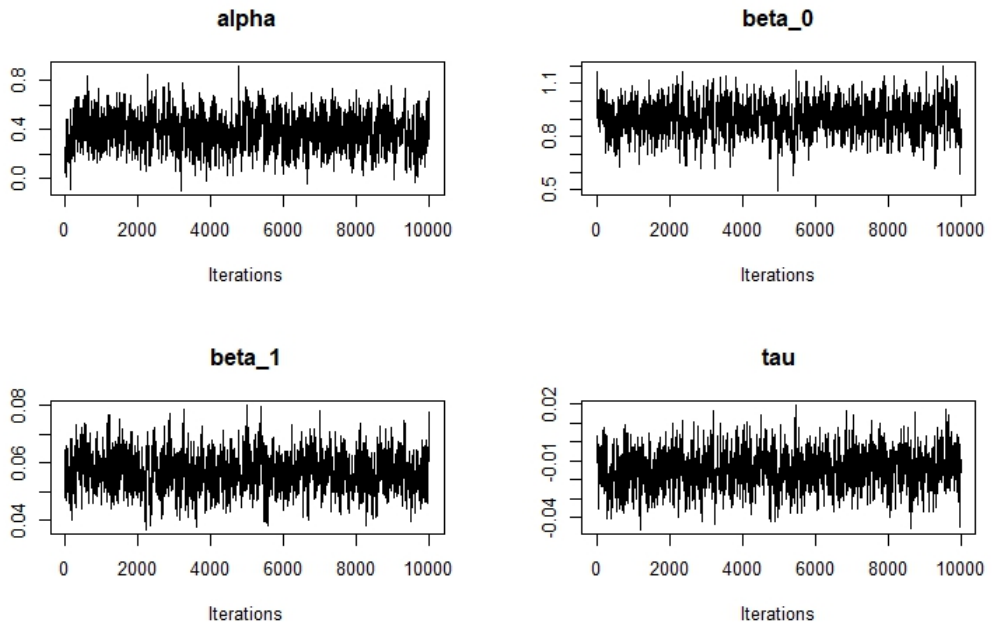
```
heidel.diag(S_alpha)
heidel.diag(S_beta0)
heidel.diag(S_beta1)
heidel.diag(S_tau)

par(mfrow=c(2,2))
traceplot(MCMC_alpha, main = "jags_alpha")
traceplot(MCMC_beta0, main = "jags_beta_0")
traceplot(MCMC_beta1, main = "jags_beta_1")
traceplot(MCMC_tau, main = "jags_tau")

median(S_alpha)
median(S_beta0)
median(S_beta1)
median(S_tau)

mean(S_alpha)
mean(S_beta0)
mean(S_beta1)
mean(S_tau)

HPDinterval(S_alpha, prob = 0.95) #HPD interval
#for the posterior distribution of alpha
HPDinterval(S_beta0, prob = 0.95) #HPD interval
#for the posterior distribution of beta0
HPDinterval(S_beta1, prob = 0.95) #HPD interval
#for the posterior distribution of beta1
HPDinterval(S_tau, prob = 0.95) #HPD interval
#for the posterior distribution of tau

#6.

manabsence <- function(age){
  result <- 2.48*(1.06^(age-39))
  return(result)
}

womanabsence <- function(age){
  result <- 2.48*1.47*(1.043^(age-39))
  return(result)
}

ratio <- function(age){
  result <- womanabsence(age)/manabsence(age)
  return(result)
```

```
}

par(mfrow = c(1,3))

plot(manabsence, xlab = "age", xlim = c(15,100))
plot(womanabsence, xlab = "age", xlim = c(15,100))
plot(ratio, xlab = "age", xlim = c(15,100), main =
"Ratio womanabsence/manabsence")
abline(a = 1, b = 0, col = "red")

#7.(a)

x <- absences_whole_set[,3]
y <- absences_whole_set[,1]
z <- absences_whole_set[,2]

colnames(absences_whole_set)= c('y','z','x')

M=length(x)
alpha <- rep(0,M)
beta_0 <- rep(0,M)
beta_1 <- rep(0,M)

tau <- rep(0,M)


mu = c()

for (i in 1:length(x)){
  mu[i] =  exp(beta_0+alpha*z[i]+(beta_1+tau*z[i])*(x[i]-39))
}

mu


# initialization of the parameters


mod= "model{
for (i in 1:length(x)){
y[i] ~ dpois(mu[i])
mu[i] = exp(beta_0+alpha*z[i]+(beta_1+tau*z[i])*(x[i]-39))
}


# specify the priors (uninformative here)
```

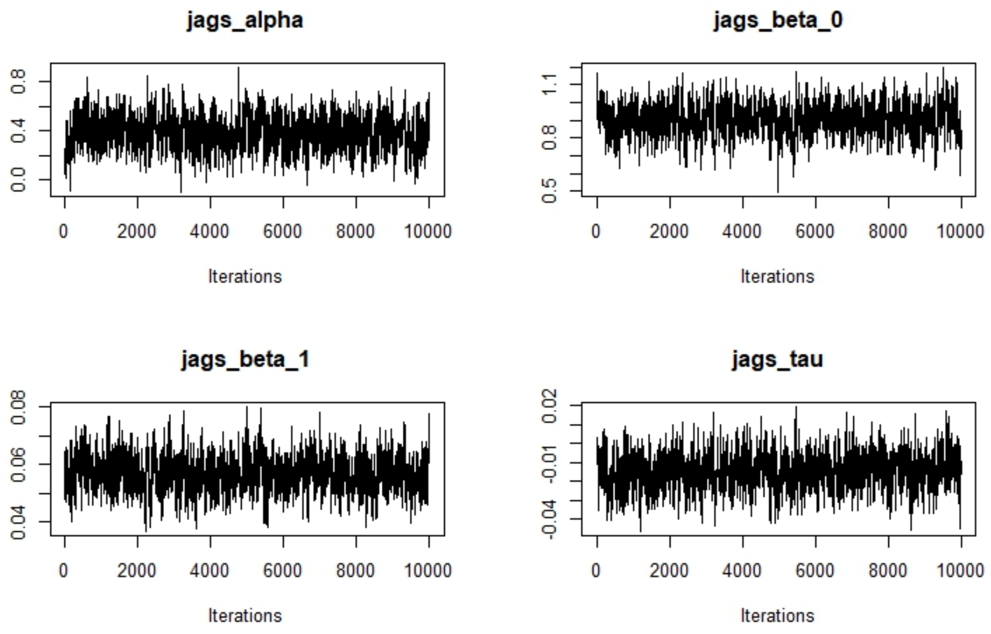
```
alpha ~ dnorm(0,0.000001)
beta_0 ~ dnorm(0,0.000001)
beta_1 ~ dnorm(0,0.000001)
tau ~ dnorm(0,0.000001)
z[93]  ~ dbern(0.5)
z[94]  ~ dbern(0.5)
z[95]  ~ dbern(0.5)
z[96]  ~ dbern(0.5)
x[97]  ~ dunif(18,65)
x[98]  ~ dunif(18,65)
x[99]  ~ dunif(18,65)
x[100]  ~ dunif(18,65)
}"

alpha
beta_0
beta_1
tau
z[93]
z[94]
z[95]
z[96]
x[97]
x[98]
x[99]
x[100]

set.seed(1348)
data = as.list(absences_whole_set)
param=c("alpha","beta_0","beta_1","tau", "z[93]", "z[94]",
"z[95]", "z[96]", "x[97]", "x[98]", "x[99]", "x[100]")

jags <- jags.model(textConnection(mod),
                   data = data,
                   n.chains = 4, # number of parallel chains
                   n.adapt = 1000 # burn-in
)
update(jags, 10000)


samples <- coda.samples(jags,
                        param,
                        10000)
#plot(samples)
#Samples are ordered as follows: alpha, beta_0, beta_1, tau, x[100],
#x[97], x[98], x[99], z[93], z[94], z[95], z[96]
```

```
samples

median(samples[[1]][,1]) #alpha
median(samples[[1]][,2]) #beta_0
median(samples[[1]][,3]) #beta_1
median(samples[[1]][,4]) #tau

median(samples[[1]][,5]) #x[100]
median(samples[[1]][,6]) #x[97]
median(samples[[1]][,7]) #x[98]
median(samples[[1]][,8]) #x[99]

mean(samples[[1]][,9]) #z[93]
mean(samples[[1]][,10]) #z[94]
mean(samples[[1]][,11]) #z[95]
mean(samples[[1]][,12]) #z[96]
```

2.2 Traceplots without JAGS



2.3 Traceplots with JAGS


jags_alpha
Iterations
jags_beta_0
Iterations
jags_beta_1
Iterations
jags_tau
Iterations