Statistical Methods in Data Science and Laboratory II: Assignment #2

Luca Tardella

Alessandro Gallo

Contents

Exercise 1

The essential ingredients to specify the probability law of a markov chain on a general state space are:

- An initial distribution μ to the initial state of the chain at time t=0
- A Transition kernel $K_t(x, A) = P\{X_{t+1} \in A | X_t = x\}$ for each t=1,2...

We can use them to derive any finite dimensional distribution. Let's say that the process starts to an initial time t=0 and a fixed state x_0 :

$$P_{X_0}(X_0 \in A) = \delta_{X_0}(A) = \int_A \mu(x)dx \tag{1}$$

We would like to know what is the probability at each t step, so:

For t=1:

$$P_{X_0}(X_1 \in A) = K(x_0, A) = \int_A K(x_0, dy) \tag{2}$$

And so on, for each t step:

For t=2

$$P_{X_0}\{X_1 \in A_1, X_2 \in A_2\} = \int_{A_1} K(y, A_2)K(x_0, dy) \tag{3}$$

For t=3

$$P_{X_0}\{X_1 \in A_1, X_2 \in A_2, X_3 \in A_3\} = \int_{A_2} K(z, A_3) \int_{A_1} K(y, dz)K(x_0, dy) \tag{4}$$

Assuming $A_1 = A_2 = ... = S$, the marginal distribution, in the general case, will be:

$$P^t_{X_0}(A_t) = P_{X_0}\{X_t \in A_t\} = K^t(x_0, A_t) = \int_S K(y, A_t)K^{t-1}(x_0, dy) \tag{5}$$

Exercise 2

We want to define stochastic processes in a way that we can approximate I with its Monte Carlo estimate:

$$\hat{I} = \frac{1}{t}\sum_{i=0}^{t} g(\theta_i) \to E_\pi[g(\theta)] = I \qquad t \to \infty \tag{6}$$

This can be done if the stationary distribution π has ergodic properties.
A Markov Chain satisfy them if it is:

- Aperiodicity: p is the period of the chain and the largest integer d such that the chain has period d. Being aperiodic, p=1. It means that, starting to any state, the chain can return to that state only at multiples of the period d.

- Irreducibility: starting from any state, it's possible to arrive to any other state with non-zero probability.

- Harris recurrence: a markov chain X_t with kernel K, ψ-irreducible (ψ maximal) is Harris recurrent if for each $x_0 \in$ S and for each A $\in$ B(S) such that $\psi(A) > 0$ we have that:

$$P\left\{\sum_{t=1}^{\infty} I_A(X_t) = \infty | X_0 = x_0\right\} = 1 \tag{7}$$

For the approximation error, we look at the variance of the MCMC estimator. We define:

$$Var_\pi[\hat{I}_t] = Var_\pi[\frac{1}{t}\sum_{i=1}^{t} h(X_i)] = \frac{1}{t^2} Var_\pi[\sum_{i=1}^{t} h(X_i)] = \tag{8}$$

$$= \frac{1}{t^2}\left\{\sum_{i=1}^{t} Var_\pi[h(X_i)] + 2\sum_{i=1}^{t}\sum_{j=1}^{t} Cov[h(X_i), h(X_j)]\right\} =$$

$$= \frac{1}{t^2}\left\{t\sigma^2 + \frac{2t\sigma^2}{t\sigma^2}\sum_{k=1}^{t-1}(t-k)\gamma_k\right\} = \frac{t\sigma^2}{t^2}\left[1 + 2\sum_{k=1}^{t-1}\frac{t-k}{t}\frac{\gamma_k}{\sigma^2}\right] = \frac{\sigma^2}{t}\left[1 + 2\sum_{k=1}^{t-1}\frac{t-k}{t}\rho_k\right]$$

$$\approx \frac{\sigma^2}{t}\left[1 + 2\sum_{k=1}^{\infty}\rho_k\right] = \frac{\sigma^2}{\frac{t}{1+2\sum_{k=1}^{\infty}\rho_k}} = \sigma^2_{\hat{I}} \tag{9}$$

A natural estimator of the quantity γ_k is

$$\hat{\gamma}_k = \frac{1}{t-k}\sum_{i=1}^{t-k}(h(X_i) - \hat{I}_t)(h(X_{i+k} - \hat{I}_t) \tag{10}$$

and to estimate the asymptotic variance of $\sqrt{t}_t$ we have:

$$\hat{\tau}^2_{\hat{i}} = \gamma_0 + 2\sum_{k=1}^{t-1}\hat{\gamma}_k \tag{11}$$

which is not consistent for τ^2.

As alternative solution, we can use:

- estimators based on weighted sum of $\hat{\gamma}_k$

- batch means

- sub-sampling

- sequential stopping criteria.

Exercise 3

(a)

Considering the following transition scheme:

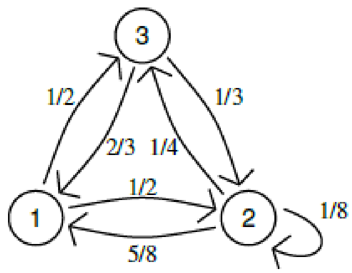
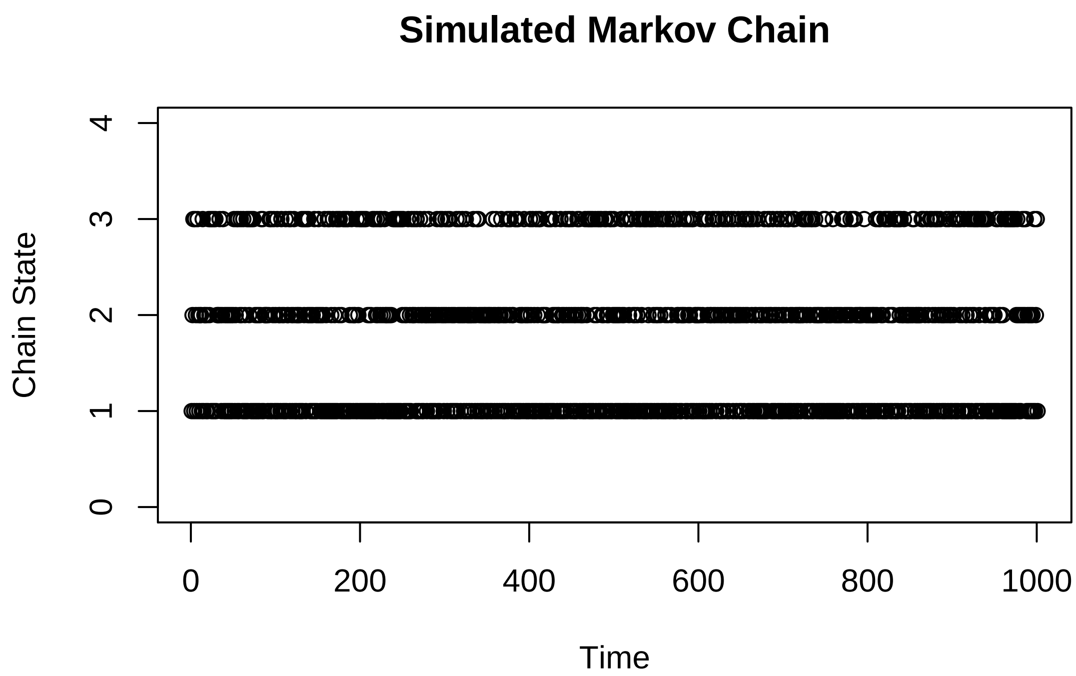


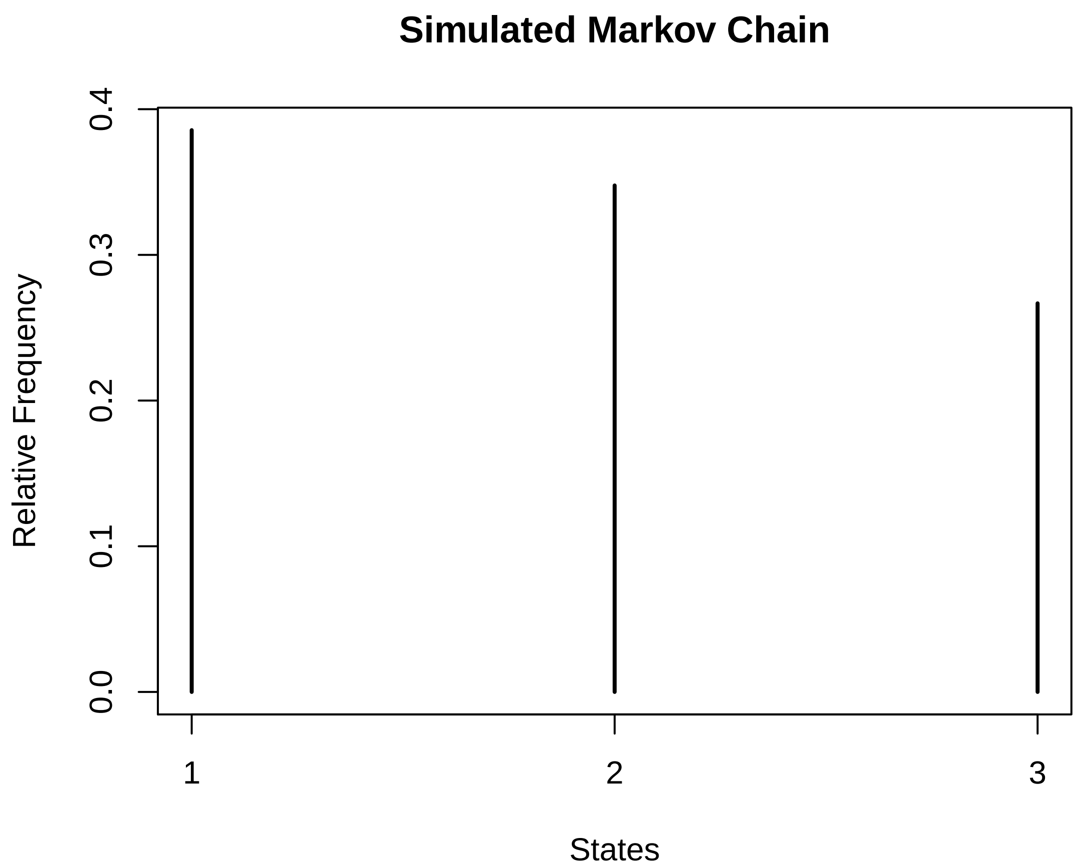
Figure 1: Transition Matrix

```r
set.seed(123)

# Transition Matrix
mpt<-matrix(c(0,0.5,0.5,
              5/8,1/8,1/4,
              2/3,1/3,0),
            nrow=3,byrow=T)
S=c(1,2,3)
# Initial State at initial time t=0
X0<-1

# Number of simulations
nsim <- 1000
# Markov Chain simulation function
mc_fun = function(mat, nsim, states, startingstate){
  chain<-rep(NA,nsim+1)
  # First state of the chain
  chain[1]<-startingstate
  # Simulation
  for(t in 1:nsim){
    chain[t+1]<-sample(states,size=1,prob=mat[chain[t],])
  }
  return(chain)
}
mc_sim1 = mc_fun(mpt,nsim,S,X0)
plot(mc_sim1, ylim=c(0,4), ylab = "Chain State", xlab = "Time",
     main = "Simulated Markov Chain")
```



b)

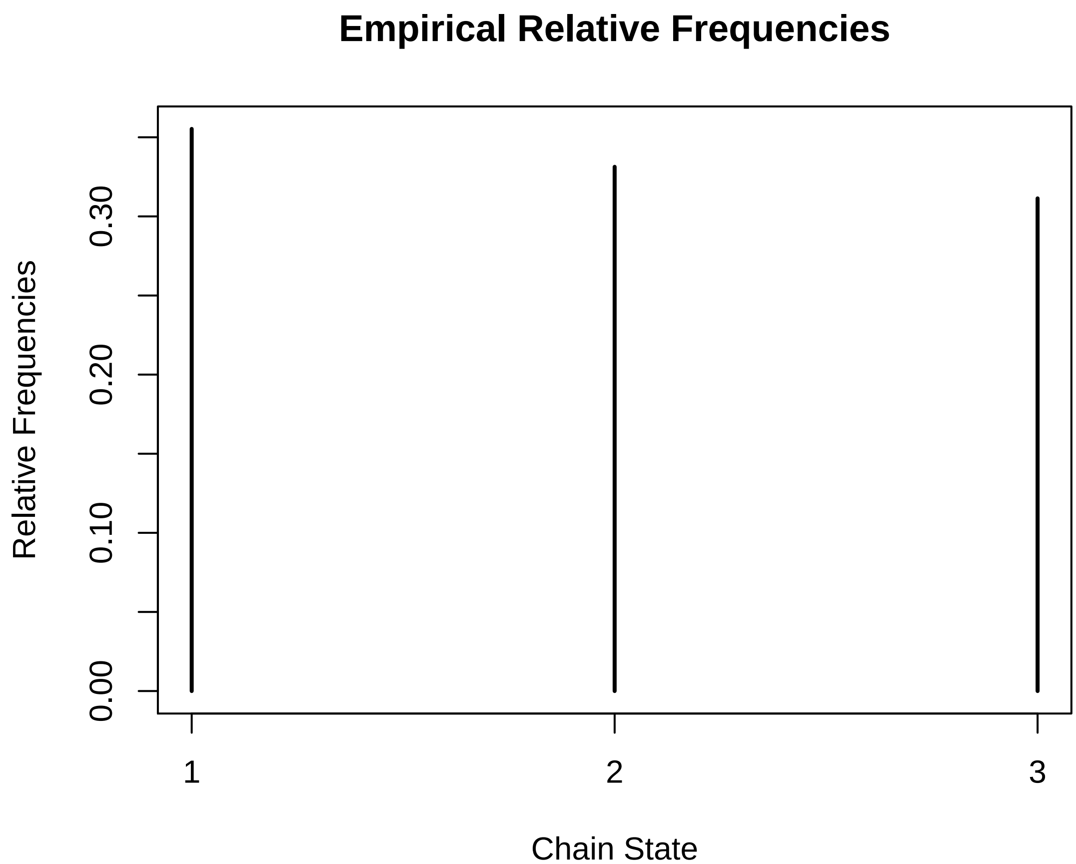
```
# Relative frequencies
rel_freq_chain1 = table(mc_sim1)/(nsim+1)
cat('Relative Frequencies of the chain with X0=1: ', rel_freq_chain1)

## Relative Frequencies of the chain with X0=1:  0.3856144 0.3476523 0.2667333

plot(rel_freq_chain1,xlab="States",ylab="Relative Frequency",
     main = "Simulated Markov Chain")
```



c)

```
rep_mc_fun<- function(nrepetition, mat, nsim, states,
                                      startingstate){
  final.state = rep(NA, nrepetition)
  for (r in 1:nrepetition){
    chain_temp = rep(NA, nsim+1)
    chain_temp[1] = startingstate
    for(t in 1:nsim){
      chain_temp[t+1] = sample(S, size=1, prob=mat[chain_temp[t],])
    }
    final.state[r] = chain_temp[nsim+1]
  }
  return(final.state)
}

nrepetit = 500
rep_mc_sim1 = rep_mc_fun(nrepetit,mpt,nsim,S,X0)
```

```
rel_freq_sim1 = table(rep_mc_sim1)/(nrepetit+1)
cat('Relative frequencies of the simulation with X0=1: ', rel_freq_sim1)

## Relative frequencies of the simulation with X0=1:  0.3552894 0.3313373 0.3113772

plot(rel_freq_sim1,xlab = 'Chain State',ylab = 'Relative Frequencies',
     main='Empirical Relative Frequencies')
```



In this case the resulting vector is not a Markov Chain because it's only a vector of final states but for the convergence to the stationary distribution π of an ergodic MC, and looking also the the relative frequencies, we can assert that It is still a good approximation of π.

d)

The theoretical stationary distribution can be computed solving the linear system:

$$\vec{\pi} P^T = \vec{\pi} \tag{12}$$

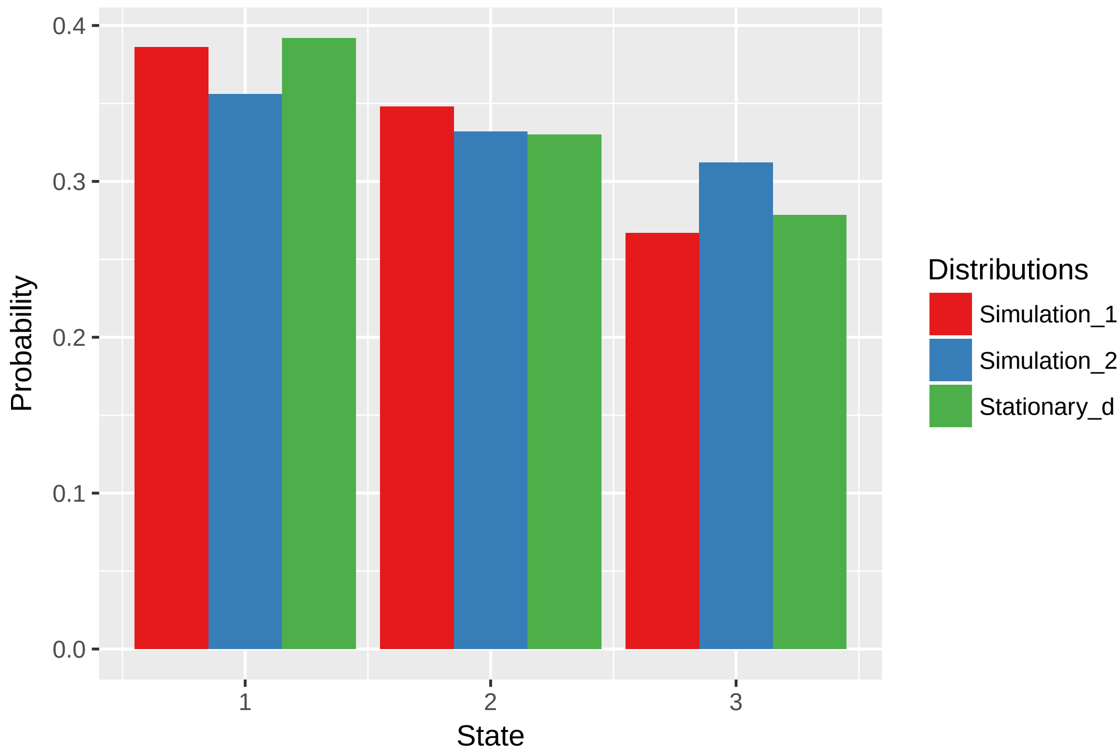
adding another equation: $\sum \pi_i = 1$

```r
st_fun <- function(matrix1, matrix2){
  pi = solve(matrix1,matrix2)
  return(pi)
}

m_1 = matrix(c(-1,5/8,2/3,1/2,-0.875,1/3,1,1,1),nrow=3,byrow = T)
m_2 = matrix(data=c(0,0,1), nrow=3, ncol=1, byrow=FALSE)
pi = st_fun(m_1,m_2)
cat('Stationary distribution: ', pi)

## Stationary distribution:  0.3917526 0.3298969 0.2783505
```

e)

Let's make bar plots of the previous results.

```r
Distr_plots <- read.table(
  header=TRUE, text='Distributions        State Probability
  1   Stationary_d       1      0.3917526
  2   Stationary_d       2      0.3298969
  3   Stationary_d       3      0.2783505
  4   Simulation_1       1      0.386
  5   Simulation_1       2      0.348
  6   Simulation_1       3      0.267
  7   Simulation_2       1      0.356
  8   Simulation_2       2      0.332
  9   Simulation_2       3      0.312')

library(ggplot2)
ggplot(Distr_plots, aes(State, Probability, fill = Distributions)) +
  geom_bar(stat="identity", position = "dodge") +
  scale_fill_brewer(palette = "Set1")
```



We can see how it is well approximated by simulated empirical relative frequencies.

f)

```
set.seed(123)

X0 <- 2

mc_sim2 = mc_fun(mpt,nsim,S,X0)
rel_freq_chain2 = table(mc_sim2)/(nsim+1)
cat('Relative frequencies of the chain with X0=2: ', rel_freq_chain2)

## Relative frequencies of the chain with X0=2:  0.3856144 0.3476523 0.2667333

rep_mc_sim2 = rep_mc_fun(nrepetit,mpt,nsim,S,X0)
rel_freq_sim2 = table(rep_mc_sim2)/(nrepetit+1)
cat('Relative frequencies of the simulation with X0=2: ', rel_freq_sim2)

## Relative frequencies of the simulation with X0=2:  0.3552894 0.3313373 0.3113772

cat('Relative frequencies of the chain with X0=1:', rel_freq_chain1)

## Relative frequencies of the chain with X0=1: 0.3856144 0.3476523 0.2667333

cat('Relative frequencies of the chain with X0=2:', rel_freq_chain2)
```

```
## Relative frequencies of the chain with X0=2: 0.3856144 0.3476523 0.2667333

cat('Relative frequencies of the simulation with X0=1:', rel_freq_sim1)

## Relative frequencies of the simulation with X0=1: 0.3552894 0.3313373 0.3113772

cat('Relative frequencies of the simulation with X0=2:', rel_freq_sim2)

## Relative frequencies of the simulation with X0=2: 0.3552894 0.3313373 0.3113772

cat('Stationary distribution: ', pi)

## Stationary distribution:  0.3917526 0.3298969 0.2783505
```

We started from a different point and we obtained the same results, showing how the frequencies approximates (convergence property) both the stationary distribution.

Exercise 4

a)

Priors:

$$\pi(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} e^{-\beta\lambda}\lambda^{\alpha-1} \cdot I_{(0,+\infty)}(\lambda) \propto e^{-\beta\lambda}\lambda^{\alpha-1} \cdot I_{(0,+\infty)}(\lambda) \tag{13}$$

$$\pi(\phi) = \frac{b^\alpha}{\Gamma(\alpha)} e^{-b\phi}\phi^{\alpha-1} \cdot I_{(0,+\infty)}(\phi) \propto e^{-b\phi}\phi^{\alpha-1} \cdot I_{(0,+\infty)}(\phi) \tag{14}$$

$$\pi(m) = \frac{1}{n-1} I_{\{1,\dots,n-1\}}(m) \propto I_{\{1,\dots,n-1\}}(m) \tag{15}$$

$$\pi(\lambda, \phi, m) \ \propto \ e^{-\beta\lambda - b\phi} \cdot \lambda^{\alpha-1} \cdot \phi^{\alpha-1} \cdot I_{(0,+\infty)}(\lambda) \cdot I_{(1,n-1)}(m)$$

Likelihood:

$$L(\lambda, \phi, m) = \prod_{i=1}^{m} \frac{e^{-\lambda}\lambda^{y_i}}{y_i!} \prod_{j=m+1}^{n} \frac{e^{-\phi}\phi^{y_j}}{y_j!} I_{\{1,\dots,n-1\}}(m) I_{(0,\infty)}(\lambda) I_{(0,\infty)}(\phi) \tag{17}$$

$$\propto e^{-m\lambda-(n-m)\phi}\lambda^{\sum_{1=1}^{m} y_i}\phi^{\sum_{j=m+1}^{n} y_j} I_{\{1,\dots,n-1\}}(m) I_{(0,+\infty)}(\lambda) I_{(0,+\infty)}(\phi) \tag{18}$$

Posterior:

$$\pi(\lambda, \phi, m|y) \propto L(\lambda, \phi, m)\pi(m)\pi(\lambda)\pi(\phi) \tag{19}$$

$$\propto e^{-(m+\beta)\lambda} e^{-(n-m-b)\phi} \lambda^{\sum_{i=1}^{m} y_i + \alpha - 1} \phi^{\sum_{j=m+1}^{n} y_j + \alpha - 1} I_{\{1,\dots,n-1\}}(m) I_{(0,+\infty)}(\lambda) I_{(0,+\infty)}(\phi) \tag{20}$$

b)

Full conditionals:

$$\pi(\lambda|\phi, m, y) \propto e^{-(m+b)\lambda}\lambda^{\sum_{i=1}^{m} y_i+\alpha-1} \tag{21}$$

$$\sim \Gamma\left(\sum_{i=1}^{m} y_i + \alpha, m+b\right) \tag{22}$$

$$\pi(\phi|\lambda, m, y) \propto e^{(n-m+b)\phi}\phi^{\sum_{j=m+1}^{n} y_j+\alpha-1} \tag{23}$$

$$\sim \Gamma\left(\sum_{j=m+1}^{n} y_j + \alpha, n-m+b\right) \tag{24}$$

$$\pi(m|\lambda, \phi, y) \propto e^{-(m)\lambda}e^{-(-m)\phi}\lambda^{\sum_{i=1}^{m} y_i}\phi^{\sum_{j=m+1}^{n} y_j} \tag{25}$$

$$\propto e^{m(\phi-\lambda)}\lambda^{\sum_{i=1}^{m} y_i}\phi^{\sum_{i=1}^{n} y_i-\sum_{i=1}^{m} y_i} \tag{26}$$

$$\propto e^{m(\phi-\lambda)}(\frac{\lambda}{\phi})^{\sum_{i=1}^{m} y_i} \tag{27}$$

c)

```
y=c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,3,5,4,5,3,1,4,
    4,1,5,5,3,4,2,5,2,2,3,4,2,1,3,2,1,1,1,1,1,3,0,0,
    1,0,1,1,0,0,3,1,0,3,2,2,0,1,1,1,0,1,0,1,0,0,0,2,
    1,0,0,0,1,1,0,2,2,3,1,1,2,1,1,1,1,2,4,2,0,0,0,1,
    4,0,0,0,1,0,0,0,0,0,1,0,0,1,0,0)

n.obs = length(y)
n.iter=10000

stat.y.firstperiod=cumsum(y)[-length(y)]
stat.y.secondperiod=sum(y)-stat.y.firstperiod

n.iter=10000
lambda=rep(NA,n.iter+1)
phi=rep(NA,n.iter+1)
m=rep(NA,n.iter+1)
m.support=seq(1,n.obs-1)
```

```
#starting values
lambda.start=lambda[1]=5
phistart=phi[1]=9
m.start=m[1]=50

#hyperparameters
ip.alpha=0.001
ip.beta=0.001
ip.a=0.001
ip.b=0.001

for(gibbs in 1:n.iter){
  lambda[gibbs+1]=rgamma(1,shape=ip.alpha+sum(y[1:m[gibbs]]),
                   rate=ip.beta+m[gibbs])
  phi[gibbs+1]=rgamma(1,shape=ip.a+sum(y[(m[gibbs]+1):n.obs]),
                   rate=ip.b+n.obs-m[gibbs])
  logci = - lambda[gibbs+1]*m.support + stat.y.firstperiod*log(lambda[gibbs+1]) +
    log(phi[gibbs+1])*stat.y.secondperiod + phi[gibbs+1]*m.support
  m.full.conditional.nn=exp(logci-max(logci))
  m[gibbs+1]=sample(x=m.support,size=1,prob=m.full.conditional.nn)
}

lambda=lambda[-(1:1001)]
phi=phi[-(1:1001)]
m=m[-(1:1001)]
```

d)

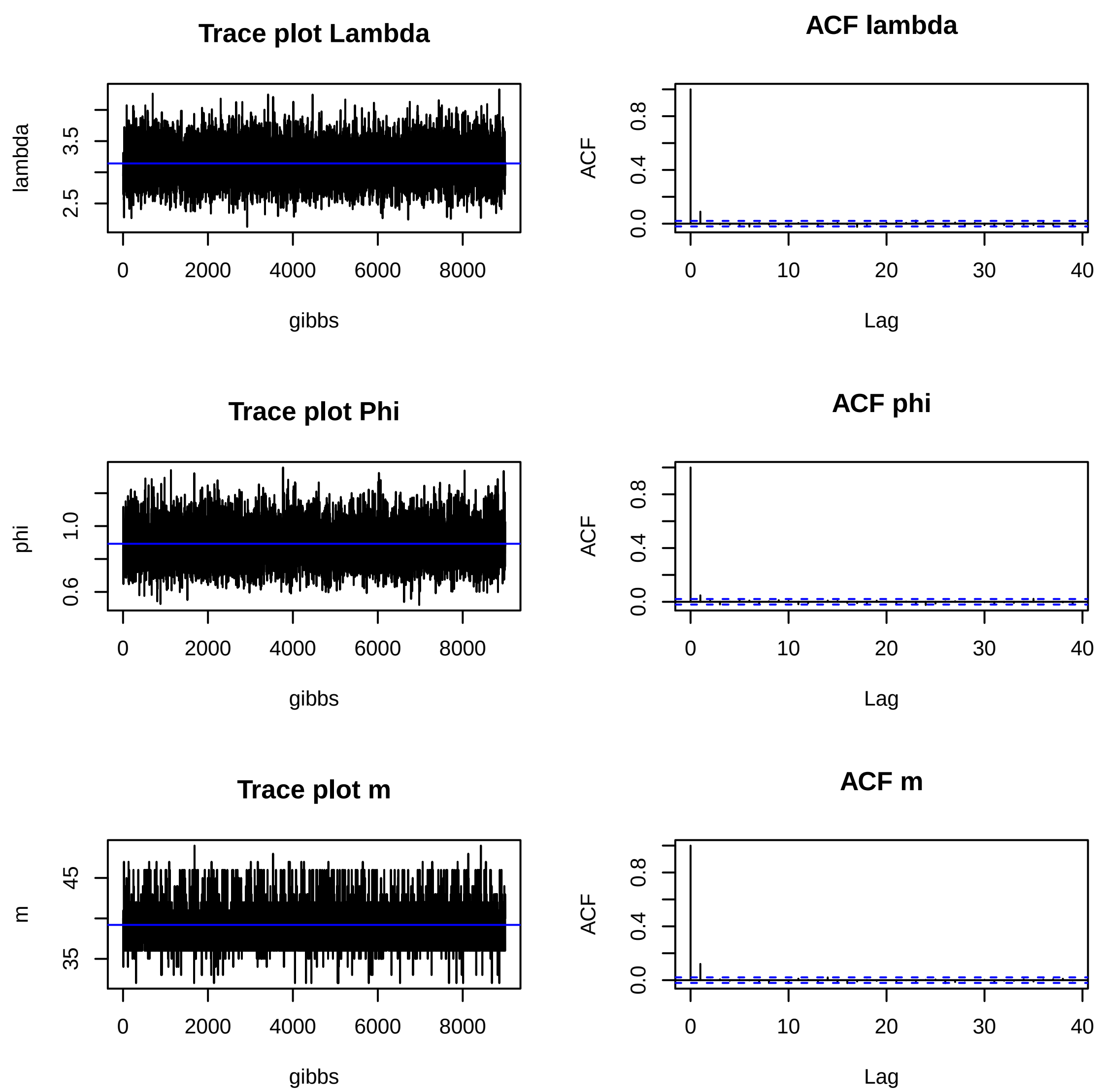
The estimated values of the parameters are:

$$\hat{\lambda} = 3.142331$$
$$\hat{\phi} = 0.892409$$
$$\hat{m} = 39.24511$$

We're going now to check before the trace plots and the auto-correlation function of the parameters:

```
par(mfrow=c(3,2))
plot(lambda,type="l",main="Trace plot Lambda", xlab="gibbs",
     ylab="lambda")
abline(h=mean(lambda),col='blue')
acf(lambda, main='ACF lambda')
plot(phi,type="l",main="Trace plot Phi", xlab="gibbs",
```

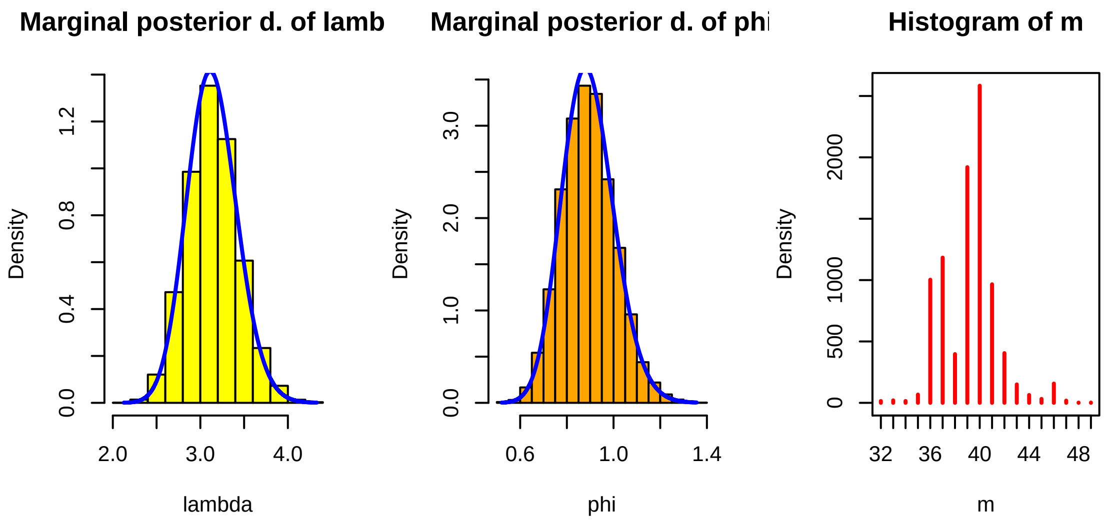
```
     ylab="phi ")
abline(h=mean(phi),col='blue')
acf(phi, main='ACF phi')
plot(m,type="l",main="Trace plot m", xlab="gibbs",
     ylab="m")
abline(h=mean(m),col='blue')
acf(m, main='ACF m')
```



Then the marginal posterior distributions of the parameters:

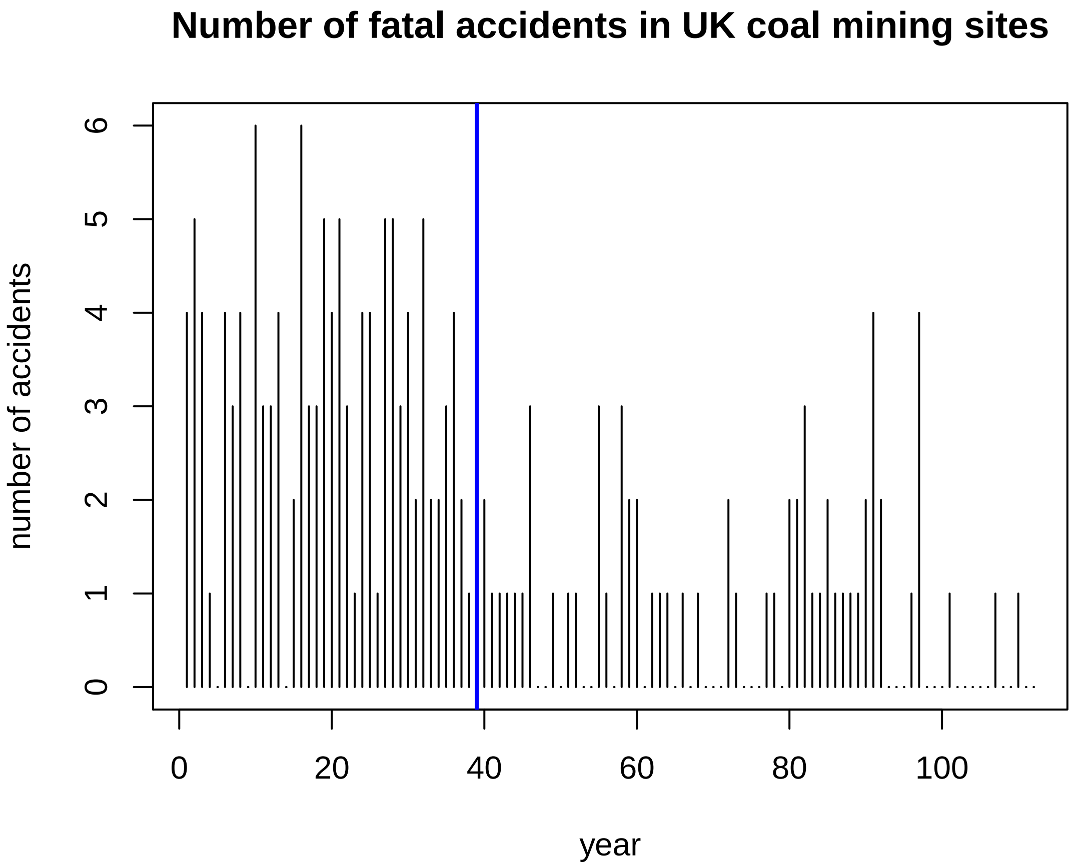
```
par(mfrow=c(1,3))
hist(lambda,freq = F,col='yellow',main='Marginal posterior d. of lambda')
x1 = seq(min(lambda), max(lambda), length=length(lambda))
y1 = dgamma(x1, ip.alpha+sum(y[1:mean(m)]), ip.beta+mean(m))
```

```
lines(x1, y1, col="blue", lwd=2)
hist(phi,freq = F,col='orange',main='Marginal posterior d. of phi')
x2 = seq(min(phi), max(phi), length=length(phi))
y2 = dgamma(x2, sum(y[(mean(m)+1):n.obs])+ip.b, n.obs-mean(m)+ip.b)
lines(x2, y2, col="blue", lwd=2)
plot(table(m),type="h",main='Histogram of m',
     col='red',ylab="Density")
```



e)

```
plot(y,main = "Number of fatal accidents in UK coal mining sites",
     xlab = "year", ylab = "number of accidents", type = "h")
abline(v = (round(mean(m))), col = "blue",lwd=2)
```



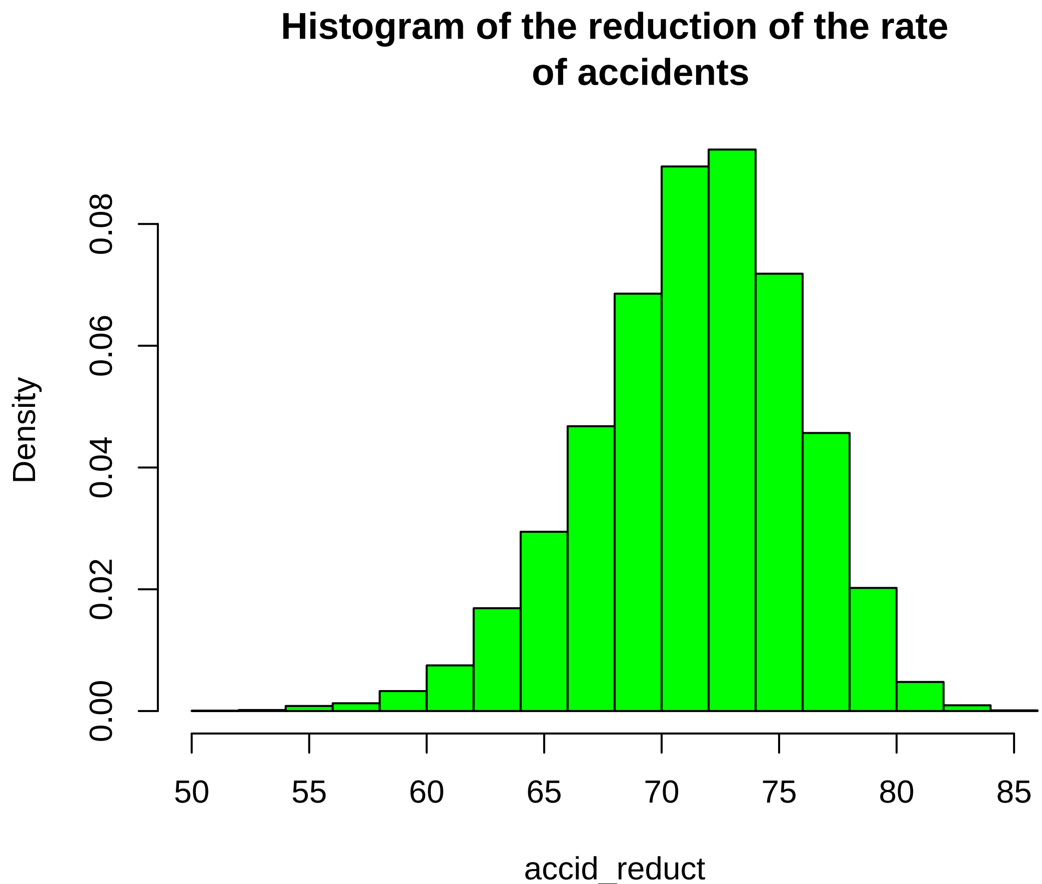

```
cat('There has been a change in the year: ', 1851 + (round(mean(m))))

## There has been a change in the year:  1890
```

f)

The expected reduction (in percentage) of the rate of accidents in two periods can be computed with the formula:

$$r = \frac{\hat{\lambda} - \hat{\phi}}{\hat{\phi}} \tag{28}$$

```
accid_reduct = (lambda-phi)/lambda*100

cat('The expected reduction is: ', mean(accid_reduct),'%')

## The expected reduction is:  71.36253 %

hist(accid_reduct, freq = F,col='green',main='Histogram of the reduction of the rate
     of accidents')
```




```
cat('The expected reduction is: ', mean(accid_reduct),'%')

## The expected reduction is:  71.36253 %
```

Exercise 3

a)

Starting from the likelihood:

$$L(\alpha, \beta, \gamma, \tau^2) \propto (\tau^2)^{-\frac{n}{2}} exp\left[-\sum_{i=}^{n} \frac{(y_i - \alpha + \beta\gamma^{x_i})^2}{2\tau^2}\right] \tag{29}$$

And writing the priors:

$$\pi(\alpha) \propto exp\left\{-\frac{\alpha^2}{2\sigma_\alpha^2}\right\} \tag{30}$$

$$\pi(\beta) \propto \left\{-\frac{\beta^2}{2\sigma_\beta^2}\right\} \tag{31}$$

$$\pi(\gamma) \propto 1 \tag{32}$$

$$\pi(\tau^2) \propto exp\left\{-\frac{b}{\tau^2}\right\} \tag{33}$$

We derive the full conditionals:

Full α:

$$\pi(\alpha|\beta,\gamma,\tau^2,y) \propto exp\left\{-\frac{1}{2\tau^2}\sum_i (y_i - \alpha + \beta\gamma^{x_i})^2\right\} exp\left\{-\frac{\alpha^2}{2\sigma_\alpha^2}\right\} \tag{34}$$

$$exp\left\{-\frac{1}{2\tau^2}\sum_i (y_i^2 + \alpha^2 + \beta^2\gamma^{2x_i} - 2\alpha y_i + 2\beta\gamma^{x_i}y_i - 2\alpha\beta\gamma^{x_i})\right\} exp\left\{-\frac{\alpha^2}{2\sigma_\alpha^2}\right\} \tag{35}$$

$$exp\left\{-\frac{1}{2\tau^2}\left(\sum_i y_i^2 + n\alpha^2 + \beta^2\sum_i \gamma^{2x_i} - 2\alpha\sum_i y_i + 2\beta\sum_i \gamma^{x_i}y_i - 2\alpha\beta\sum_i \gamma^{x_i}\right) - \frac{\alpha^2}{2\sigma_\alpha^2}\right\} \tag{36}$$

$$exp\left\{-\frac{1}{2\tau^2}\left(n\alpha^2 - 2\alpha\sum_i y_i - 2\alpha\beta\sum_i \gamma^{x_i}\right) - \frac{\alpha^2}{2\sigma_\alpha^2}\right\} \tag{37}$$

$$exp\left\{-\frac{1}{2}\left[\left(\frac{n\sigma_\alpha^2 + \tau^2}{\tau^2\sigma_\alpha^2}\right)\alpha^2 - 2\frac{\alpha}{\tau^2}\left(\sum_i y_i + \beta\sum_i \gamma^{x_i}\right)\right]\right\} \tag{38}$$

$$exp\left\{\frac{1}{2}\left[\frac{\alpha^2 - 2\alpha\left(\frac{\sum_i y_i + \beta\sum_i \gamma^{x_i}}{\tau^2}\right)\frac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2+\tau^2}}{\frac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2+\tau^2}}\right]\right\} \tag{39}$$

$$\sim \mathcal{N}\left(\frac{\sigma_\alpha^2\sum_i (y_i + \beta\gamma^{x_i})}{n\sigma_\alpha^2 + \tau^2}, \frac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2 + \tau^2}\right) \tag{40}$$

Full β:

$$\pi(\beta|\alpha,\gamma,\tau^2,y) \propto exp\left\{-\frac{1}{2\tau^2}\sum_i (y_i - \alpha + \beta\gamma^{x_i})^2\right\} exp\left\{-\frac{\beta^2}{2\sigma_\beta^2}\right\} \tag{41}$$

$$exp\left\{-\frac{1}{2\tau^2}\sum_i (y_i^2 + \alpha^2 + \beta^2\gamma^{2x_i} - 2\alpha y_i + 2\beta\gamma^{x_i}y_i - 2\alpha\beta\gamma^{x_i})\right\} exp\left\{-\frac{\beta^2}{2\sigma_\beta^2}\right\} \tag{42}$$

$$exp\left\{-\frac{1}{2\tau^2}\left(\sum_i y_i^2 + n\alpha^2 + \beta^2\sum_i \gamma^{2x_i} - 2\alpha\sum_i y_i + 2\beta\sum_i \gamma^{x_i}y_i - 2\alpha\beta\sum_i \gamma^{x_i}\right) - \frac{\beta^2}{2\sigma_\beta^2}\right\} \tag{43}$$

$$exp\left\{-\frac{1}{2\tau^2}\left(\beta^2\sum_i\gamma^{2x_i}+2\beta\sum_i\gamma^{x_i}y_i-2\alpha\beta\sum_i\gamma^{x_i}\right)-\frac{\beta^2}{2\sigma_\beta^2}\right\} \tag{44}$$

$$exp\left\{-\frac{1}{2}\left[\frac{\sigma_\beta^2\sum_i\gamma^{2x_i}+\tau^2}{\tau^2\sigma_\beta^2}\beta^2-2\beta\frac{\sum_i(\alpha\gamma_i^x-\gamma^{x_i}y_i}{\tau^2}\right]\right\} \tag{45}$$

$$exp\left\{-\frac{1}{2}\left[\frac{\beta^2-2\beta\left(\frac{\sum_i(\alpha\gamma_i^x-\gamma^{x_i}y_i)}{\tau^2}\right)\frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_i\gamma^{2x_i}+\tau^2}}{\frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_i\gamma^{2x_i}+\tau^2}}\right]\right\} \tag{46}$$

$$\sim\mathcal{N}\left(\frac{\sigma_\beta^2\sum_i(\alpha\gamma^{x_i}-\gamma^{x_i}y_i)}{\sigma_\beta^2\sum_i\gamma^{2x_i}+\tau^2},\frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_i\gamma^{2x_i}+\tau^2}\right) \tag{47}$$

Full γ:

$$\pi(\gamma|\alpha,\beta,\tau^2,y)\propto exp\left\{-\frac{1}{2\tau^2}\sum_i(y_i-\alpha+\beta\gamma^{x_i})^2\right\}I_{[0,1]}(\gamma) \tag{48}$$

Full τ^2:

$$\pi(\tau^2|\alpha,\beta,\gamma,y)\propto\tau^{2(-\frac{n}{2})}exp\left\{-\frac{1}{2\tau^2}\sum_i(y_i-\alpha+\beta\gamma^{x_i})^2\right\}\frac{e^{-\frac{b}{\tau^2}}}{\tau^{2(\alpha+1)}} \tag{49}$$

$$\propto\tau^{2(-\frac{n}{2}-\alpha-1)}exp\left\{-\frac{1}{2\tau^2}\left[2b+\sum_i(y_i-\alpha+\beta\gamma^{x_i})^2\right]\right\} \tag{50}$$

$$I\Gamma\left(\frac{n}{2}+\alpha,\frac{2b+\sum_i(y_i-\alpha+\beta\gamma^{x_i})^2}{2}\right) \tag{51}$$

b)

We recognize all the distributions but not the γ one.

c)

```
# Dugongs data

# Length
x = c( 1.0, 1.5, 1.5, 1.5, 2.5, 4.0, 5.0, 5.0, 7.0, 8.0, 8.5,
                          9.0, 9.5, 9.5, 10.0, 12.0, 12.0, 13.0, 13.0, 14.5, 15.5,
                          15.5, 16.5, 17.0, 22.5, 29.0, 31.5)

# Age
Y = c(1.80, 1.85, 1.87, 1.77, 2.02, 2.27, 2.15, 2.26, 2.47, 2.19,
                         2.26, 2.40, 2.39, 2.41, 2.50, 2.32, 2.32, 2.43, 2.47, 2.56,
```

```
                     2.65, 2.47, 2.64, 2.56, 2.70, 2.72, 2.57)
N = 27

n = length(x)
```

```
#Hyperparameters
sigma.a = 100
sigma.b = 100
a = 0.001
b = 0.001

set.seed(1234)
library(pscl)
library(MCMCpack)


#Hyperparameters
sigma.a = 100
sigma.b = 100
a = 0.001
b = 0.001

# Full alpha conditional
full.cond.alpha = function(beta, gamma, tau){
  # Alpha mu
  mu = (sigma.a^2 * sum(beta*gamma^x + Y)) / (tau + n*sigma.a^2 )
  # Alpha var
  var = (tau*sigma.a^2) / (tau + n*sigma.a^2)
  a.constr = 0
  while(a.constr < 1){
    a.constr = rnorm(1, mu, sqrt(var))
  }
  return(a.constr)
}

# Full beta conditional
full.cond.beta = function(alpha, gamma, tau){
  # Beta mu
  mu = (sigma.b^2*sum(alpha*gamma^x - Y*gamma^x))/(tau + sigma.b^2*sum(gamma^(2*x)))
  # Beta var
  var = (tau*sigma.b^2) / (tau + sigma.b^2*sum(gamma^(2*x)))
  b.constr = 0
```

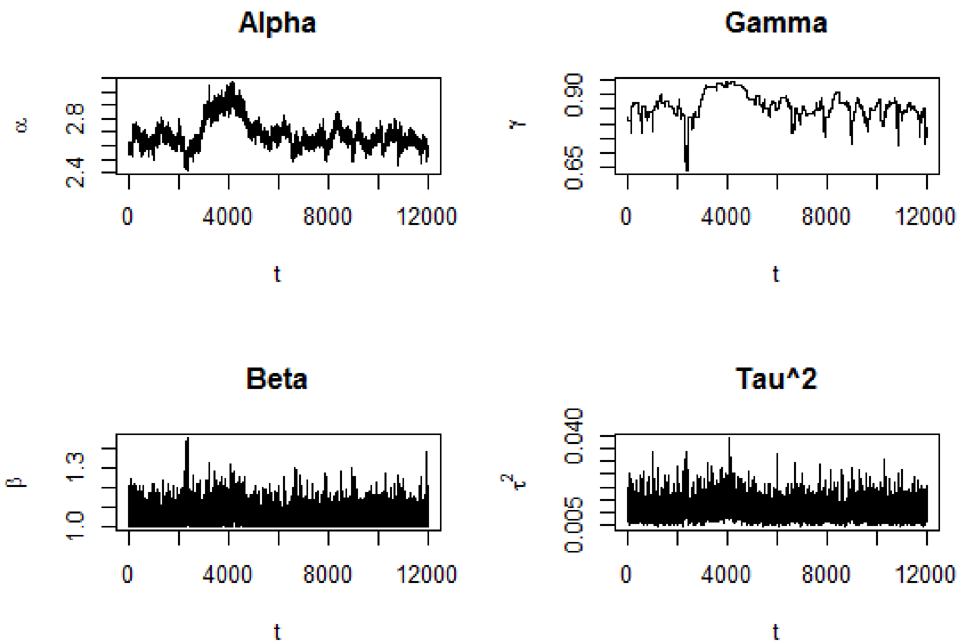
```
  while(b.constr < 1){
    b.constr = rnorm(1, mu, sqrt(var))
  }
  return(b.constr)
}

# Full tau conditional
full.cond.tau = function(alpha, beta, gamma){
  # Tau Shape
  shape = n/2 + a
  # Tau rate
  rate = b + (sum((beta*gamma^x + Y - alpha)^2))/2
  tau = rigamma(1, shape, rate)
  return(tau)
}

# Full gamma conditional
full.cond.gamma = function(alpha, beta, tau, gamma){
  arg = -1/(2*tau) * sum((beta*gamma^x + Y - alpha)^2)
  return(exp(arg))
}


# Metropolis-within-Gibbs algorithm
metro_fun = function(alpha.old, beta.old, gamma.old, tau.old, n.sim){

  for (gibbs in 1:n.sim){
    alpha.old[gibbs + 1] = full.cond.alpha(beta.old[gibbs], gamma.old[gibbs],
                                 tau.old[gibbs])
    beta.old[gibbs + 1] = full.cond.beta(alpha.old[gibbs + 1], gamma.old[gibbs],
                                 tau.old[gibbs])
    gamma.cand = runif(1, 0, 1)
    gamma.prob.old = full.cond.gamma(alpha.old[gibbs + 1], beta.old[gibbs + 1],
                                 tau.old[gibbs], gamma.old[gibbs])
    gamma.prob.cand = full.cond.gamma(alpha.old[gibbs + 1], beta.old[gibbs + 1],
                                 tau.old[gibbs], gamma.cand)
    if (gamma.prob.cand/gamma.prob.old<1) {
      p = gamma.prob.cand/gamma.prob.old
    } else {
      p = 1
    }
    gamma.old[gibbs + 1] = sample(c(gamma.cand, gamma.old[gibbs]), size = 1,
                                 prob = c(p, 1-p))
```

```
    tau.old[gibbs + 1] = full.cond.tau(alpha.old[gibbs + 1], beta.old[gibbs + 1],
                                        gamma.old[gibbs + 1])
  }
  return(list(alpha.old,beta.old,gamma.old,tau.old))
}

n.sim = 13000
a = 0.1

alpha.old = rep(NA,n.sim+1)
beta.old = rep(NA,n.sim+1)
gamma.old = rep(NA,n.sim+1)
tau.old = rep(NA, n.sim+1)

alpha.old[1] = 2
beta.old[1] = 1
gamma.old[1] = 0.5
tau.old[1] = 1

mf =  metro_fun(alpha.old,beta.old,gamma.old,tau.old,n.sim)

# Burn-in procedure
alpha = mf[[1]][-(1:1000)]
beta = mf[[2]][-(1:1000)]
gamma = mf[[3]][-(1:1000)]
tau = mf[[4]][-(1:1000)]

alpha.hat = mean(alpha)
beta.hat = mean(beta)
gamma.hat = mean(gamma)
tau.hat = mean(tau)
```

d)

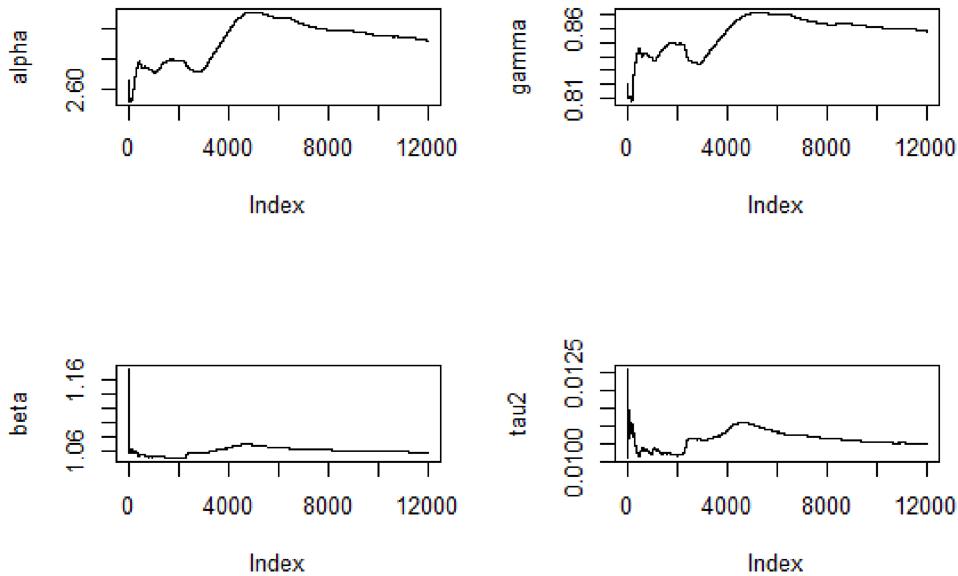
```
par(mfcol=c(2,2))
plot(seq(1, length(alpha)), alpha, type = "l", xlab = "t", ylab = expression(alpha)
     ,main='Alpha')
plot(seq(1, length(beta)), beta, type = "l", xlab = "t", ylab = expression(beta)
     ,main='Beta')
plot(seq(1, length(gamma)), gamma, type = "l", xlab = "t", ylab = expression(gamma)
     ,main='Gamma')
plot(seq(1, length(tau)), tau, type = "l", xlab = "t", ylab = expression(tau^2)
```

```
    ,main='Tau^2')
```



e)

```
par(mfcol=c(2,2))
plot(cumsum(alpha)/1:length(alpha), type="l", ylab = "alpha")
plot(cumsum(beta)/1:length(beta), type="l", ylab = "beta")
plot(cumsum(gamma)/1:length(gamma), type="l", ylab = "gamma")
plot(cumsum(tau)/1:length(tau), type="l", ylab = "tau2")
```



f)

We want to estimate the variance of that has the covariance component so we can not use the naive estimator of Exercise 2. The chosen method will be the Batched Means: we divide t simulation in B groups with equal size $\frac{t}{B}$.
The overall empirical mean can be seen as an average of the B group means.

```
library(batchmeans)

library(batchmeans)
app.err.alpha = bm(alpha)$se
app.err.beta = bm(beta)$se
app.err.gamma = bm(gamma)$se
app.err.tau = bm(tau)$se
```

$\alpha_{err} = 0.009478047$
$\beta_{err} = 0.002174508$
$\gamma_{err} = 0.003979647$
$\tau^2_{err} = 0.000110519$

g)

The posterior uncertainty is measured with:

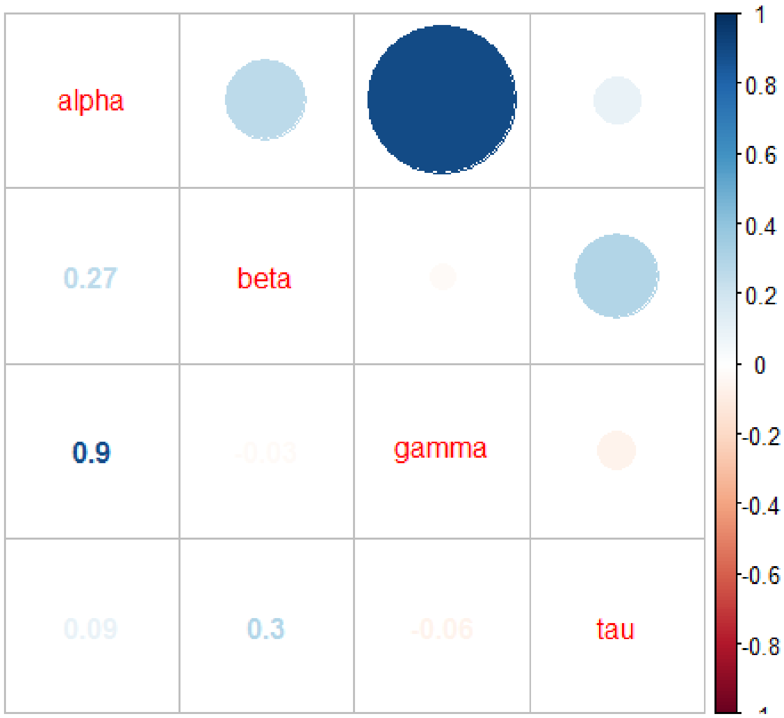
$$\frac{\sqrt{Var}}{\mu} \tag{52}$$

```
pu.alpha = sqrt(var(alpha))/mean(alpha)
pu.alpha = sqrt(var(beta))/mean(beta)
pu.alpha = sqrt(var(gamma))/mean(gamma)
pu.alpha = sqrt(var(tau))/mean(tau)
```

$\alpha_{pu} = 0.0389941$
$\beta_{pu} = 0.04879854$
$\gamma_{pu} = 0.05131915$
$\tau^2_{pu} = 0.3236271$

h)

```
correlation.par = cor(data.frame(alpha, beta, gamma, tau))

par(mfcol=c(1,1))
df.corr = data.frame(alpha,beta,gamma,tau)
library(corrplot)
correl <- cor(df.corr)
corrplot.mixed(correl)
```



i)

```
pos.pred.distr = function(x){
  mu = alpha - beta*gamma^x
  arg = rnorm(length(alpha), mu, tau)
  return(arg)
}

new_dug1 = pos.pred.distr(20)
new_dug_mean1 = mean(new_dug1)
```

The posterior predictive length of a dugong with age of 20 years is: 2.603696

j)

```
new_dug2 = pos.pred.distr(30)
new_dug_mean2 = mean(new_dug2)
```

The posterior predictive length of a dugong with age of 30 years is: 2.652631

k)

Let's use again the batched means:

```
app.err1.dug1 = bm(new_dug1)$se
app.err2.dug2 = bm(new_dug2)$se
pos.unc1.dug1 = sqrt(var(new_dug1))/mean(new_dug1)
pos.unc2.dug2 = sqrt(var(new_dug2))/mean(new_dug2)
```

The approximation error and posterior uncertainty of the 1 dugong are: 0.002664751 and 0.01501401

The approximation error and posterior uncertainty of the 2 dugong are: 0.005902201 and 0.02617365

As we can see, both the approximate error and the posterior uncertainty are lower in the case of the 20 years old dugong.