Statistical Methods for Data Science Laboratory Computational Statistics (2015/2016)

Assignment # 2

Paolo Tamagnini

July 2016

Contents

1 Exercise 1 **3**

2 Exercise 2 **4**
2.1 Irreducibility . . . 4
2.2 Aperiodicity . . . 5
2.3 Recurrence . . . 6
2.4 Ergodic Theorem . . . 7
2.5 The Approximation Error . . . 8

3 Exercise 3 **9**
3.1 Question (a) . . . 9
3.2 Question (b) . . . 9
3.3 Question (c) . . . 9
3.4 Question (d) . . . 10
3.5 Question (e) . . . 11
3.6 Question (f) . . . 11

4 Exercise 4 **12**
4.1 Question (a) . . . 12
4.2 Question (b) . . . 13
4.3 Question (c) . . . 13
4.4 Question (d) . . . 14
4.5 Question (e) . . . 19
4.6 Question (f) . . . 20

5 Exercise 5 **20**
5.1 Question (a) . . . 20
5.1.1 The model . . . 20
5.1.2 The likelihood function . . . 20
5.1.3 The priors formulas . . . 20
5.1.4 The posterior distribution . . . 21
5.1.5 The full-conditional for α . . . 21
5.1.6 The full-conditional for β . . . 21
5.1.7 The full-conditional for γ . . . 22
5.1.8 The full-conditional for τ^2 . . . 22
5.2 Question (b) . . . 23
5.3 Question (c) . . . 23
5.4 Question (d) . . . 26
5.5 Question (e) . . . 28
5.6 Question (f) . . . 30
5.7 Question (g) . . . 32
5.8 Question (h) . . . 32
5.9 Question (i) . . . 33
5.10 Question (j) . . . 33
5.11 Question (k) . . . 33

1 Exercise 1

A markov chain on a general state space $\mathcal{S} \subset \mathbb{R}^k$ is a stochastic process that uses a discrete index t.

$$\{X_0, X_1, X_2, .., X_t, ..\} \quad t = 0, 1, ..$$

The essential ingredients to specify the probability law of a Markov chain are the following:

- An initial distribution $\mu(x)$ from which the first value x_0 of the chain can be drawn.
- A transition kernel with which we can draw all the other values for each $t \in \{1, 2, ..\}$ as follows:

  $$K_t(x, A) = Pr\{X_{t+1} \in A | X_t = x\}$$

  The kernel function is indeed able to give us a probability measure given the value of the precedent $X_{t-1} = x$ with $x \in \mathcal{S}$ for any set of values $A \in \mathcal{B}(\mathcal{S})$. So we are able to draw each X_t in such a way that it will be dependent just from the precedent value X_{t-1} and not also from all the previous ones $\{X_0, X_1, X_2, .., X_{t-2}\}$.

Thanks to those two ingredients we can define the probability law as follows:

$X_0 \sim \mu(x)$

$Pr\{X_{t+1} \in A | X_0 = x_0, X_1 = x_1, X_2 = x_2, .., X_t = x, ..\} =$

$= Pr\{X_{t+1} \in A | X_t = x\} = K_t(x, A)$

So in other words to build a stochastic process that is a Markov chain we need a distribution able to give us a suitable starting value and a function able to compute probabilities in such a way that each random variable in the chain depends just on the precedent one.
We can derive any finite dimensional distribution thanks to those same elements.
First we must draw, thanks to $\mu(x)$, and then fix the starting state x_0.

$x_0 \sim \mu(x) \rightarrow P_{x_0}\{X_0 \in A\} = \delta_{x_0}(A) = \int_A \mu(x)dx$

Then we can compute each of the distributions of interest in the process:

$P_{x_0}\{X_1 \in A\} = K(x_0, A) = \int_A K(x_0, dy_1)$

$P_{x_0}\{X_1 \in A_1, X_2 \in A_2\} = \int_{A_1} K(y_1, A_2)K(x_0, dy_1)$

$P_{x_0}\{X_1 \in A_1, X_2 \in A_2, X_3 \in A_3\} = \int_{A_1} \int_{A_2} K(y_2, A_3)K(y_1, dy_2)K(x_0, dy_1)$

$P_{x_0}\{X_1 \in A_1, .., X_t \in A_t\} = \int_{A_1} .. \int_{A_{t-1}} K(y_{t-1}, A_t)K(y_{t-2}, dy_{t-1}) .. K(x_0, dy_1)$

If we denote:

$K^1(x_0, A) = K(x_0, A) = P_{x_0}\{X_1 \in A\}$

then the kernel for t transitions is given by:

$$P_{x_0}\{X_t \in A\} = K^t(x_0, A) = \int_S K^{t-1}(y, A)K(x_0, dy)$$

2 Exercise 2

The Monte Carlo Markov Chain (MCMC) can be used to approximate the finite quantity I. We want to define the ingredients for which a Markov chain $\{\theta_1, \theta_2, .., \theta_{T_0}, .., \theta_t\}$ behaves as follows:

$$\hat{I} = \frac{1}{t}\sum_{i=T_0}^{T_0+t} h(\theta_i) \;\rightarrow\; \mathbb{E}_\pi[x] = I$$

Where T_0 defines the index used to burn in the chain, meaning removing the first T_0 values, where the chain is more unstable and the dependence from the initial point θ_0 is higher.
The distribution $\pi(x)$ is the target distribution that the chain is supposed to simulate. Such distribution has to be invariant with respect to the kernel $K(x, A)$, meaning that:

$$\pi(A) = \int_S K(y, A)\pi(dy)$$

meaning we have to forge a kernel such that each state is always drawn from the same distribution π:

$$X_t \sim \pi \;\rightarrow\; X_{t+1} \sim \pi$$

$\pi(x)$ will also be used to draw our first value x_0.
A chain then is said to be homogeneous when the probability of drawing a certain value h, given a fixed past one k, doesn't change with the index t:

$$Pr\{X_{t+1} = h|X_t = k\} = Pr\{X_t = h|X_{t-1} = k\} \quad \forall\, t$$

So in order to simulate a MCMC able to approximate I, we need to have an homogeneous Markov chain with an invariant distribution π with respect to the kernel $K(A, x)$ that satisfies the following **ergodic properties**:

- Irreducibility
- Aperiodicity
- Recurrence

2.1 Irreducibility

This features describes the ability of a chain to be able to go from any state of the state space $\mathcal{S}$ to any other existing space.

This is easy to describe in a discrete case, where the chain can be represented by a graph and each state as a node. Indeed in the discrete case a Markov chain is irreducible if there is positive probability of starting from any fixed state and arriving at any other state in a finite number of steps. Just by analysing the transition probability matrix P of the graph, we will be

able to tell if the directed graph is fully connected and so if the chain itself is irreducible.
In the continuous case instead, the situation is more tricky. To determine if the chain is able to move freely in the continuous state space $\mathcal{S}$ we find a measure ϕ of $\mathcal{S}$ for which the Kernel always measure a probability greater than 0 for finite t. A Markov chain will be ϕ-irreducible if:

$$\exists\ t < \infty : P_s\{X_t \in A\} = K^t(s, A) > 0$$

$$\forall s \in \mathcal{S},\ \forall A \in \sigma(\mathcal{S}) : \phi(A) > 0$$

It can be shown it exists a uniquely defined measure ψ, called *maximal*, such that if a chain is ϕ-irreducible is also ψ-irreducible. Such measure ψ will be always be dominating any other measure for which the chain is irreducible and it can be determined thanks to the candidate measure ϕ.

$$\psi(A) = \int_{\mathcal{S}} K(y, A)\phi(dy)$$

To better understand the nature of ψ we must look at the definition of *atom*. An *atom* A is a subset of $\mathcal{B}(\mathcal{S})$ from which no matter what starting point I pick, I will always end up in the subset B with the same non null probability $\nu(B)$. In formulas:

$$K(x, B) = \nu(B) \quad \forall\ x \in A,\ \ \forall\ B \in \mathcal{B}(\mathcal{S})$$

This is related to ψ because an atom is said to be accessible if $\psi(A) > 0$, meaning if we are able to end up in A in the first place or not. Indeed if $\psi(A) = 0$ it means it doesn't exist a t for which $K^t(s, A) > 0$, then the atom A is not accessible.
In particular we can define a subset C *small set* if, for any starting point in C, we can always reach any subset $B \in \sigma(\mathcal{S})$ in a finite number m of steps and with a non-null probability measure ν_m:

$$K^m(x, B) \geq \nu_m(B) \quad \forall\ x \in C,\ \ \forall\ B \in \sigma(\mathcal{S})$$

We will use those notions later in the next sections.

2.2 Aperiodicity

This features of a Markov chain is important because it guarantees that there aren't convergence problems due to periodicity of the chain route in the state space $\mathcal{S}$. This periodicity appears as the chain values oscillates between sets of values rather then converging in a single one. This is due to the transitions between those set that force the chain to repeat such cycle over and over again. It is possible to measure such cycle with the measure d. The following statements mostly apply for the discrete case, but they can also be adapted for the continuous case. Given a state $g \in \mathcal{S}$, a Markov chain has cycle of length d passing through g where:

$$d = GCD\{m : P_g\{X_m = g\} > 0\} = GCD\{m : K^m(g, g) > 0\}$$

To explain why d can measure the size of a cycle we need to inspect the nature of the set of m on which we compute the greatest common divisor. Starting with $x_0 = g$ the chain moves in the state space and when ever it visits again the state g, we append to the set the number of

passed steps m_i. The chain will visit the state g multiple times, then, once we have enough m_i, we can see whether there is a pattern in these values. If there is periodicity in the behaviour of the chain, the sequence of m_i has a pattern. The most direct way to inspect that is to see if all m_i are multiples of a same number. The higher this number the more relevant is the cycle. The value d is indeed the greatest common divisor of such sequence.
If $d = 1$ the cycle going through the state g has size equal to 1 and this implies there is no cycle at all, as the sequence of m_i has no pattern, with no divisor greater than 1. If the chain is irreducible will have to pass through each state. This means that to inspect the presence of cycles, it will be sufficient to measure d for a single state g, rather then computing it for all states $j \in \mathcal{S}$. If we compute then a single state g with $d = 1$, all the other states will have also $d = 1$. In that case no cycle is present and the Markov chain is then aperiodic.

2.3 Recurrence

Irreducibility and aperiodicity are not enough to ensure convergence. What we mean for convergence is that for any starting point we can pick from the state space, we always converge to the same result for $t \to \infty$. An aperiodic and irreducible Markov chain will go eventually through each state without oscillating, but it is also important it visits each state a suitable number of times in order to have an asymptotic behaviour and the desired approximation.
This feature is described by the recurrence which is related to the expected number of times a chain will visit the state. Given a starting value $x \in \mathcal{S}$ and a subset $A \in \sigma(\mathcal{A})$, we can indeed compute the expected value $\eta_x(A)$ of the number of times that the chains that starts in x will visit the set A.

$$\eta_x(A) = \mathbb{E}\left[\sum_{t=1}^{\infty} I_A(X_t)\right]$$

where $I_A(X_t) = 1$ if $X_t \in A$, otherwise $I_A(X_t) = 0$. The idea of (weak) recurrence is about the expectation that an infinite chain visits the states and infinite number of times. From this formula we can derive the following definitions of weak recurrence:

- The state x is recurrent if: $\eta_x(\{x\}) = \infty$
- The subset A is recurrent if: $\eta_x(A) = \infty$
- The chain $\{X_1, .., X_t, ..\}$, for $t \in (1, \infty)$, is recurrent if:
  - The chain is ψ-irreducible.
  - $\eta_x(A) = \infty \quad \forall x \in A, \ \forall A : \psi(A) > 0$

Then a chain is recurrent if it is expected that for $t \to \infty$ each state, and not just some, will be visited an infinite number of times, and this is strongly related to irreducibility, without which there might be some states that are not even reached once.
If X_t is ψ-irreducible than it will exists a small subset C, such that any chain starting in C will always return in C with a finite number of steps. This is:

$$P_x(\tau_C < \infty) = 1 \ \forall x \in C$$

where τ_C it is indeed the first index value t of the first $X_t \in C$.

$$\tau_C = inf\{t \geq 0 : X_t \in C\}$$

This formula already talks about a feature very important for convergence, but it is not enough since it limits the ability to always reach a subset C just from chains starting from C and not from any $x \in \mathcal{S}$. This is due to weak recurrence which implies that it is just expected that we will visit each state an infinite number of times. We want instead this to happen probability $= 1$ and to achieve this we need that our chain is Harris recurrent.
A ψ-irreducible Markov chain is Harris recurrent if:

$$Pr\Big\{ \sum_{t=1}^{\infty} I_A(X_t) = \infty \,|\, X_0 = x_0 \Big\} = 1$$
$$\forall x_0 \in \mathcal{S},\ \ \forall A \in \mathcal{B}(S) : \psi(A) > 0$$

stating that such chain will visit all possible states with certain probability starting from all possible states. Thanks to this a ψ-irreducible and Harris recurrent Markov chain can reach a small subset C $\forall x_0 \in \mathcal{S}$ as follows:

$$P_x(\tau_C < \infty) = 1 \ \ \forall x \in \mathcal{S}$$

$$\tau_C = inf\{t \geq 0 : X_t \in C\}$$

In words, the probability that from, any state of the state space, an Harris recurrent chain visits always to the same set of values in a finite number of steps, is certain.

2.4 Ergodic Theorem

An aperiodic, Harris recurrent (then also irreducible) Markov chain with invariant probability measure π is such that:

$$|P_{x_0}\{X_t \in A\} - \pi(A)| \to 0 \quad \forall x_0 \in \mathcal{S},\ \ \forall A \in \sigma(\mathcal{S})$$
$$|K^t(x_0, A) - \pi(A)| \to 0 \quad \forall x_0 \in \mathcal{S},\ \ \forall A \in \sigma(\mathcal{S})$$
$$|\hat{I}_{t,x_0} - I| \to 0 \quad \forall x_0 \in \mathcal{S},\ \ t \to \infty$$
$$\hat{I} = \frac{1}{t} \sum_{i=T_0}^{T_0+t} h(\theta_i) \ \to \ \mathbb{E}_\pi[x] = I \quad t \to \infty$$

2.5 The Approximation Error

As it is shown in the last formulas the more $P_{x_0}\{X_t \in A\}$ is close to the target distribution $\pi(A)$, the better the chain simulates the target distribution, the more $\hat{I}$ will get close to I and the error will decrease.
To measure such error we should consider:

$$\mathbb{E}\left[(\hat{I}_t - I)^2\right] = Var_\pi(\hat{I}_t) =$$

$$= Var_\pi\left(\frac{1}{t}\sum_{i=1}^{t} h(\theta_i)\right) =$$

$$= \frac{1}{t}\left(\gamma_0 + 2\sum_{k=1}^{t-1}\gamma_k\right)$$

$\gamma_0 = Var(h(\theta_i))$

$\gamma_k = Cov(h(\theta_0), h(\theta_k))$

Unfortunately we need to deal with the dependency between the variables of the simulation that we did not have in other Monte Carlo simulations. Anyway we are able to deal with this thanks to the following estimator of the variance of our approximation.

$$\hat{Var}_\pi(\hat{I}_t) = \frac{1}{t}\left(\hat{\gamma}_0 + 2\sum_{k=1}^{t-1}\hat{\gamma}_k\right)$$

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

Then we can say that the approximation error depends from the variance of $\hat{I}$ and that its estimate will depend on how much $\hat{I}$ differs from each $h(X_i)$. Most of all the approximation error will decrease if t increases.

3 Exercise 3

3.1 Question (a)

The first thing to do was to get the probability transition matrix P from the graph:

$$P_{(3\times3)} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{5}{8} & \frac{1}{8} & \frac{1}{4} \\ \frac{2}{3} & \frac{1}{3} & 0 \end{pmatrix}$$

As we can see the sum of each row of the matrix is equal to 1, as it was expected since from each state the next step will have to pick an out-going edge with certain probability.
The following R code has been used:

```
seme = 123
set.seed(seme)
mpt<-matrix(c(0,1/2,1/2,5/8,1/8,1/4,2/3,1/3,0),nrow=3,byrow=T)
S=c(1,2,3)
x0<-1
nsample<-1000
chain<-rep(NA, nsample+1)
chain[1]<-x0
for(t in 1:nsample){
  chain[t+1]<-sample(S, size=1,prob=mpt[chain[t],]) }
resultB = table(chain)/nsample
```

3.2 Question (b)

The computed frequencies are the following:

-	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{\pi}_3$
(b)	0.386	0.348	0.267

3.3 Question (c)

To use the last values of 500 different chain the following code has been used:

```
set.seed(seme)

mpt<-matrix(c(0,0.5,0.5,0.625,0.125,0.25,2/3,1/3,0),nrow=3,byrow=T)
S=c(1,2,3)
x0<-1
nsample<-1000
nchains = 500
```

```
lastVal<-rep(NA,nchains)

for (k in 1:nchains) {

  chain<-rep(NA,nsample+1)
  chain[1]<-x0

  for(t in 1:nsample){
    chain[t+1]<-sample(S,size=1,prob=mpt[chain[t],]) }

  lastVal[k] = chain[nsample+1] }

resultC = table(lastVal)/nchains
```

Let's update our result chart:

-	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{\pi}_3$
(b)	0.386	0.348	0.267
(c)	0.358	0.332	0.310

Those results differs because (b) and (c) are different kinds of simulations. While (b) is a MCMC, where each element is connected to the previous one, and that satisfies ergodic properties, (c) instead is not Markov chain at all. Indeed (c) contains only the last value of multiple Markov chains, which have a better quality than the first values. Anyway by drawing the samples independently, (c) looses all the properties of the Markov chains and those properties (irreducibility, aperiodicity, recurrence, invariance, etc.) are the reason why MCMC are used to make good simulations.

As we pointed out before a Markov chain is such that the probability measure of drawing a value belonging to the set A at time index t will be:

$$P_{x_0}\{X_t \in A\} = K^t(x_0, A)$$

In the case (b) for $t_b \in [1, 1000]$ we pick X_{t_b} using the probability measure $K^{t_b}(x_0, A)$, this way we build our simulation $\{X_1, .., X_{t_b}, .., X_{1000}\}$. Instead for (c) we always use the same probability measure $K^{1000}(x_0, A)$ to draw each sample X_{t_c} with $t_c \in [1, 500]$. This explain us why the results are different and we know that (b) will be closer to π because:

$$|K^{t_b}(x_0, A) - \pi(A)| \to 0 \quad \forall x_0 \in \mathcal{S}, \ \forall A \in \sigma(\mathcal{S})$$

while it is a different case for (c) where this error is:

$$|K^{1000}(x_0, A) - \pi(A)| \quad \forall \ t_c$$

3.4 Question (d)

The theoretical stationary distribution can be computed with the following linear system where each equations describes for each node the flows of probability in the graph. Indeed we will be using the transition probability matrix where π is our vector of unknown variables: $\pi = \pi P$.

We have to add another equation so that the overall probability in the graph is always equal to 1.

$$\pi = P^T\pi \rightarrow \begin{cases} \pi_1 = \frac{5}{8}\pi_2 + \frac{2}{3}\pi_3 \\ \pi_2 = \frac{1}{2}\pi_1 + \frac{1}{8}\pi_2 + \frac{1}{3}\pi_3 \\ \pi_3 = \frac{1}{2}\pi_1 + \frac{1}{4}\pi_2 \\ 1 = \pi_1 + \pi_2 + \pi_3 \end{cases} \rightarrow \begin{cases} -\pi_1 + \frac{5}{8}\pi_2 + \frac{2}{3}\pi_3 = 0 \\ \frac{1}{2}\pi_1 + -\frac{7}{8}\pi_2 + \frac{1}{3}\pi_3 = 0 \\ \frac{1}{2}\pi_1 + \frac{1}{4}\pi_2 - \pi_3 = 0 \\ \pi_1 + \pi_2 + \pi_3 = 1 \end{cases} \rightarrow M\pi = c$$

$$M_{(4\times3)} = \begin{pmatrix} -1 & \frac{5}{8} & \frac{2}{3} \\ \frac{1}{2} & -\frac{7}{8} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{4} & -1 \\ 1 & 1 & 1 \end{pmatrix} \qquad c = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Now we have all what's necessary to let R solve this linear system:

```
M <- matrix(c(-1,5/8,2/3,1/2,-7/8,1/3,1/2,1/4,-1,1,1,1),nrow=4,byrow=
    red↪ T)
c = c(0,0,0,1)
resultPi = qr.solve(M,c)
```

Let's update our chart:

-	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{\pi}_3$
(b)	0.386	0.348	0.267
(c)	0.358	0.332	0.310
-	π_1	π_2	π_3
(d)	0.3917526	0.3298969	0.2783505

3.5 Question (e)

Below a table with the errors of (b) and (c) respect to (d), which is the target distribution π, computed as $\Delta\hat{\pi} = \pi - \hat{\pi}$

.

-	$\Delta\hat{\pi}_1$	$\Delta\hat{\pi}_2$	$\Delta\hat{\pi}_3$	mean($\lvert\Delta\hat{\pi}\rvert$)
(b)	0.03375	-0.00210	-0.03165	0.0117354
(c)	0.00575	-0.01810	0.01135	0.02250172

As expected the error is greater for the simulation (c), which it wasn't at all a MCMC.

3.6 Question (f)

Even if we change the starting value x_0 from 1 to 2, the result stays the same. This is explained by the ergodic properties of our MCMC. In particular we see that our chain will converge always towards π for any x_0 because it satisfies the ergodic theorem. First of all we can see clearly from the graph that the chain is irreducible, meaning that for sure from any of the 3 node we can reach any other. It is of course a consequence that the directed graph is connected. Our

MCMC is irreducible and it is also Harris recurrent because, as it is obvious from the graph, with probability = Ω any chain from any starting value will reach any other state in a finite number of steps. It could be done empirically but it is obvious as well that by measuring the length of the cycle for one of the three values, we will find there is no pattern in the list of indexes of the states that visit such value. We can find then that $d = 1$ for $x_0 = 1$ for example. With such result and being the chain irreducible this will tell us that it is also aperiodic.
Being the MCMC aperiodic and Harris recurrent with an invariant probability measure π we can explain why we obtain always the same approximation, no matter which starting value we pick.

4 Exercise 4

From 1851 to 1962 the number of accident in the UK coal minings is descrived by variable Y_i as follows:

$$(Y_1, ..., Y_{m-1}, Y_m, Y_{m+1}, ..., Y_n), \;\; n = 112$$

where m is the year after which there should have been a change in the rate of accident. Since those Y are described by a Poisson distribution such change is modeled in a change of parameter for the Y as follows:

$$Y_i \sim Pois(\lambda), i = 1, 2, ..., m$$
$$Y_j \sim Pois(\phi), i = m+1, m+2, ..., n$$

The unknown parameters will be (λ, ϕ, m) and their prior distribution the following:

$$\lambda \sim Gamma(\alpha, \beta), \lambda \in (0, +\infty) \rightarrow \pi(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} e^{-\beta\lambda} \lambda^{\alpha-1} I_{(0,+\infty)}(\lambda) \propto e^{-\beta\lambda} \lambda^{\alpha-1} I_{(0,+\infty)}(\lambda)$$

$$\phi \sim Gamma(a, b), \phi \in (0, +\infty) \rightarrow \pi(\phi) = \frac{b^a}{\Gamma(a)} e^{-b\phi} \phi^{a-1} I_{(0,+\infty)}(\phi) \propto e^{-b\phi} \phi^{a-1} I_{(0,+\infty)}(\phi)$$

$$m \sim Unif\left\{1, 2, ..., n-1\right\}, m = 1, 2, ..., n-1 \rightarrow \pi(m) = \frac{1}{n-1} I_{1,2,...,n-1}(m) \propto I_{1,2,...,n-1}(m)$$

4.1 Question (a)

We can now compute the Likelihood function of our model.

$$\begin{aligned} L(\lambda, \phi, m | y_1, ..., y_n) = & \\ & = \prod_{i=1}^{m} f(y_i|\lambda) \prod_{i=m+1}^{n} f(y_i|\phi) = \\ & = \prod_{i=1}^{m} \frac{e^{-\lambda} \lambda^{y_i}}{y_i!} \prod_{i=m+1}^{n} \frac{e^{-\phi} \phi^{y_i}}{y_i!} \propto \\ & \propto e^{-m(\lambda-\phi)-n\phi} \lambda^{\sum_{i=1}^{m} y_i} \phi^{\sum_{i=m+1}^{n} y_i} \end{aligned}$$

$$(\lambda, \phi, m) \in (0, \infty) \times (0, \infty) \times \{1, 2, ..., n-1\}$$

4.2 Question (b)

With the likelihood and priors we can compute the posterior distribution:

$$\begin{aligned}\pi(\lambda, \phi, m|y_1, ..., y_n) \propto \\ &\propto L(\lambda, \phi, m|y_1, ..., y_n)\pi(\lambda)\pi(\phi)\pi(m) \propto \\ &\propto \left[e^{-m\lambda}\right]\left[e^{-(n-m)\phi}\right]\left[\lambda^{\sum_{i=1}^{m} y_i}\right]\left[\phi^{\sum_{i=m+1}^{n} y_i}\right]\left[e^{-\beta\lambda}\right]\left[\lambda^{\alpha-1}\right]\left[e^{-b\phi}\right]\left[\phi^{a-1}\right]\end{aligned}$$

$$(\lambda, \phi, m) \in (0, \infty) \times (0, \infty) \times \{1, 2, ..., n-1\}$$

By selecting from the priors the factors that contain each parameter, we build the full-conditionals:

- $\pi(\lambda|\phi, m, y) \propto e^{-(\beta+m)\lambda}\lambda^{\alpha+\sum_{i=1}^{m} y_i-1} I_{(0,+\infty)}(\lambda) \sim Gamma(\alpha + \sum_{i=1}^{m} y_i, \beta + m)$
- $\pi(\phi|\lambda, m, y) \propto e^{-(b+n-m)\phi}\phi^{a+\sum_{i=m+1}^{n} y_i-1} I_{(0,+\infty)}(\phi) \sim Gamma(a + \sum_{i=m+1}^{n} y_i, b + n - m)$
- $\pi(m|\lambda, \phi, y) \propto e^{(\phi-\lambda)m}\lambda^{\sum_{i=1}^{m} y_i}\phi^{\sum_{i=m+1}^{n} y_i} I_{(1,2,...,n-1)}(m)$

4.3 Question (c)

It follows the code that implemented Gibbs Sampling for the model.

```
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 = length(Y)
alfa<-2
beta<-1
a<-2
b<-1
nchain<-11000

post <- function(theta){
  lamb = theta[1]
  ph = theta[2]
  mm = theta[3]
  prod = exp(-mm*lamb)
  prod = exp(-(n-mm)*ph)*prod
  prod = lamb^(sum(Y[1:mm]))*prod
  prod = phi^(sum(Y[(mm+1):n]))*prod
  prod = exp(-beta*lamb)*prod
  prod = lamb^(a-1)*prod
  prod = exp(-beta*ph)*prod
  prod = ph^(a-1)*prod
  return(prod)}

mChainDist<-function(t1, t2, X) {
  n<-length(X)
  #print(n)
  probVector<-rep(0,n-1)
```

```
  for (i in 1:n-1) {
    uno<-exp((t2-t1)*i)
    due<-t1^sum(X[1:i])
    tre<-t2^sum(X[(i+1):n])
    probVector[i]<-uno*due*tre }
  probVector<-probVector/sum(probVector)
  return(probVector) }

lambdaChain<-rep(0, nchain)
phiChain<-rep(0, nchain)
mChain<-rep(0, nchain)
lambda<-rgamma(1, alfa, beta)
phi<-rgamma(1, a, b)
m<-sample(1:n, 1)


for (s in 1:nchain) {
  lambdaChain[s]<-lambda<-rgamma(1, alfa+sum(Y[1:m]), beta+m)
  phiChain[s]<-phi<-rgamma(1, a+sum(Y[(m+1):n]), b+n-m)
  probVect<-mChainDist(lambda, phi, Y)
  mChain[s]<-m<-sample(1:(n-1), 1, prob=probVect) }

lambdaMean = mean(lambdaChain[1001:nchain])
phiMean = mean(phiChain[1001:nchain])
m = mean(mChain[1001:nchain])
t = table(mChain)/length(mChain)
moda = strtoi(names(t[which(t == max(t))[1]]))
mYearsMode = moda + 1850
mYearsMean = m + 1850
postValue = post(c(lambdaMean,phiMean,moda))
```

4.4 Question (d)

To perform Gibbs Sampling, three Markov chains have been built, one for each parameter (λ,ϕ,m). For each chain, we draw the next value using as parameters for the full-conditionals the most recent values from the other chains. For λ and ϕ chains, it was easier because the full-conditionals are already recognized probability distribution. Instead for m, where its full-conditional is not a probability distribution, we had to define the function *mChainDist* able to give a probability vector with which the next m in the chain was picked. Indeed such probability vector has a component for each possible value of $m \in [1, n-1]$, and its sum is equal to 1. To compute such probability vector, the process is really straight-foward. Given the most recent λ and ϕ, we compute the measure given by the full-conditional of m for each possible value of $m \in [1, n-1]$, building a vector. Then we normalize such vector dividing each component for its sum. This way the vector has a sum equal to 1 and it has all component values between 0 and 1.

After we have all chains 11'000 values long, we stop the iteration, we burn in the first 1000 values, we compute the mean values and for m also the mode. The results are as follows where $t = 10000$ and $T_0 = 1000$:

$$\bar{\lambda} = \frac{1}{t} \sum_{i=T_0}^{T_0+t} \hat{\lambda}_i = 3.111577$$

$$\bar{\phi} = \frac{1}{t}\sum_{i=T_0}^{T_0+t} \hat{\phi}_i = 0.9077287$$

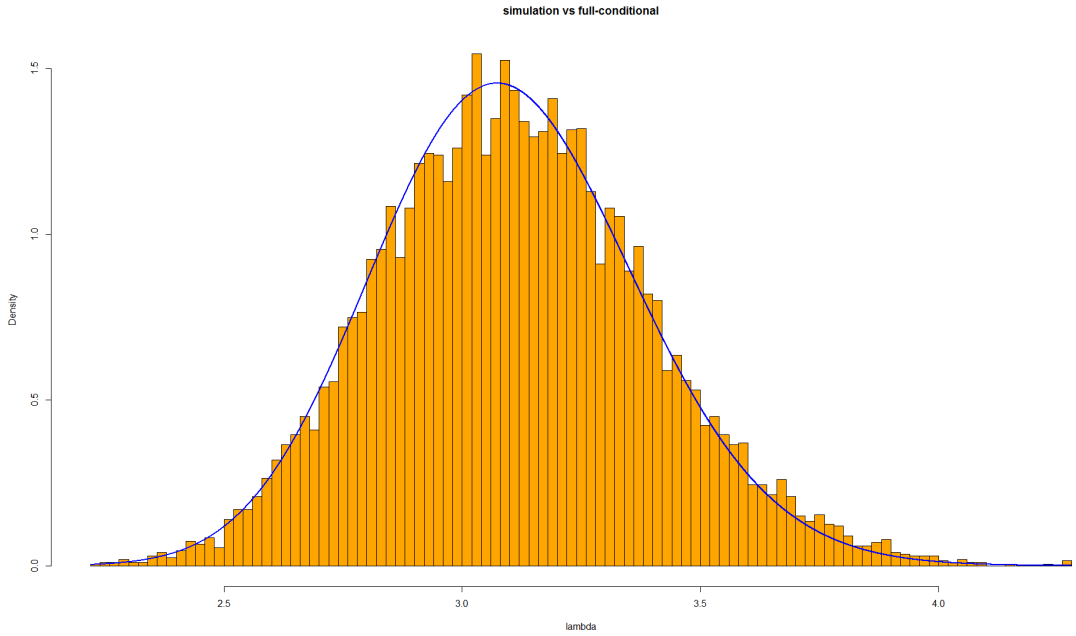
$$\bar{m} = \frac{1}{t}\sum_{i=T_0}^{T_0+t} \hat{m}_i = 39.1908$$

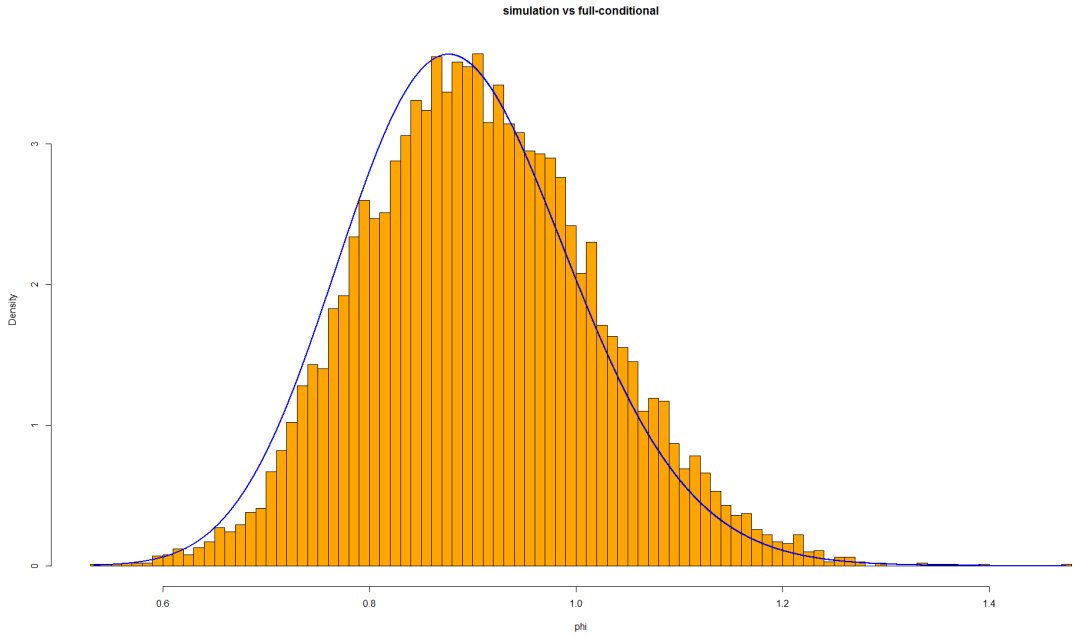
$$\dot{m} = Mode(\hat{m}_1, \hat{m}_2, .., \hat{m}_t,) = 40$$

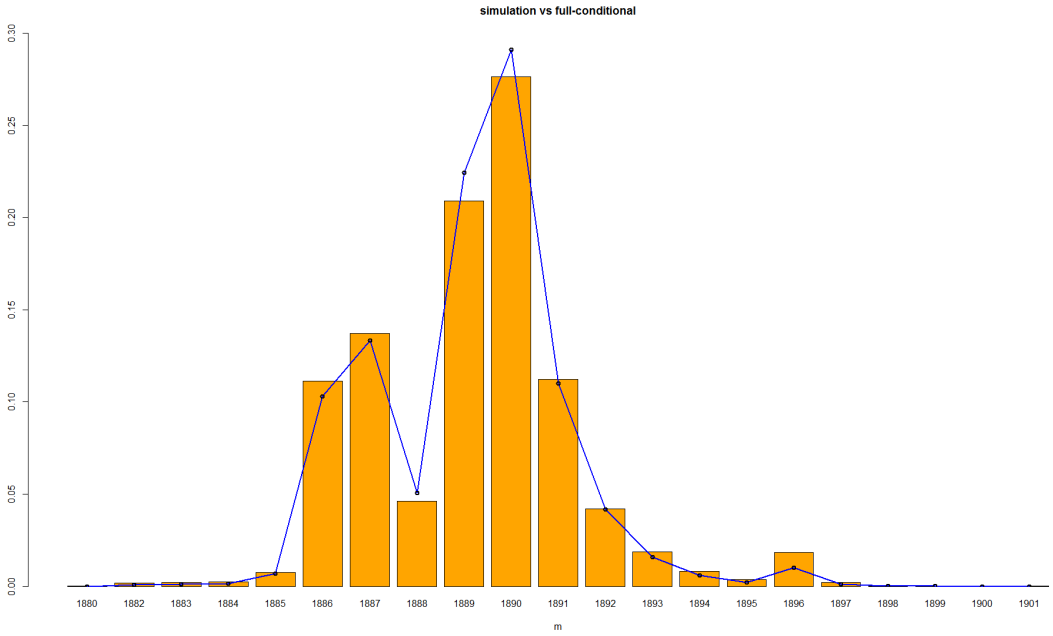
We can compare now each simulation to the full-conditionals used. The fitting is not obvious because each value of the chain was drawn always from the same full-conditional, but the used parameters were changing every iteration. We will fix the parameters for the full-conditional as follows:

$$(\bar{\lambda}, \bar{\phi}, \dot{m}) = (3.111577, 0.9077287, 40)$$

```
lamdaSim = lambdaChain[1001:nchain]

hist(lamdaSim,freq = F,breaks = 75, col = 'orange', main = 'simulation vs full-
    red↪ conditional',xlab='lambda')
xfit<-seq(min(lamdaSim),max(lamdaSim),length=length(lamdaSim))
yfit<-dgamma(xfit,alfa+sum(Y[1:moda]),beta+moda)
lines(xfit, yfit, col="blue", lwd=2)

phiSim = phiChain[1001:nchain]

hist(phiSim,freq = F,breaks = 75, col = 'orange', main = 'simulation vs full-conditional
    red↪ ',xlab='phi')
xfit<-seq(min(phiSim),max(phiSim),length=length(phiSim))
yfit<-dgamma(xfit,a+sum(Y[(moda+1):n]),b+n-moda)
lines(xfit, yfit, col="blue", lwd=2)

tavol = table(mSim)/length(mSim)
tbl <- barplot(tavol,col = 'orange', main = 'simulation vs full-conditional',ylim=c
    red↪ (0,0.30),xlab='m')
listM = c(30,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51)
yfit<-mChainDist(lambdaMean,phiMean,Y)[listM]
lines(x = tbl, y = yfit, col="blue",lwd=2)
points(x = tbl, y = yfit,lwd=2)
```

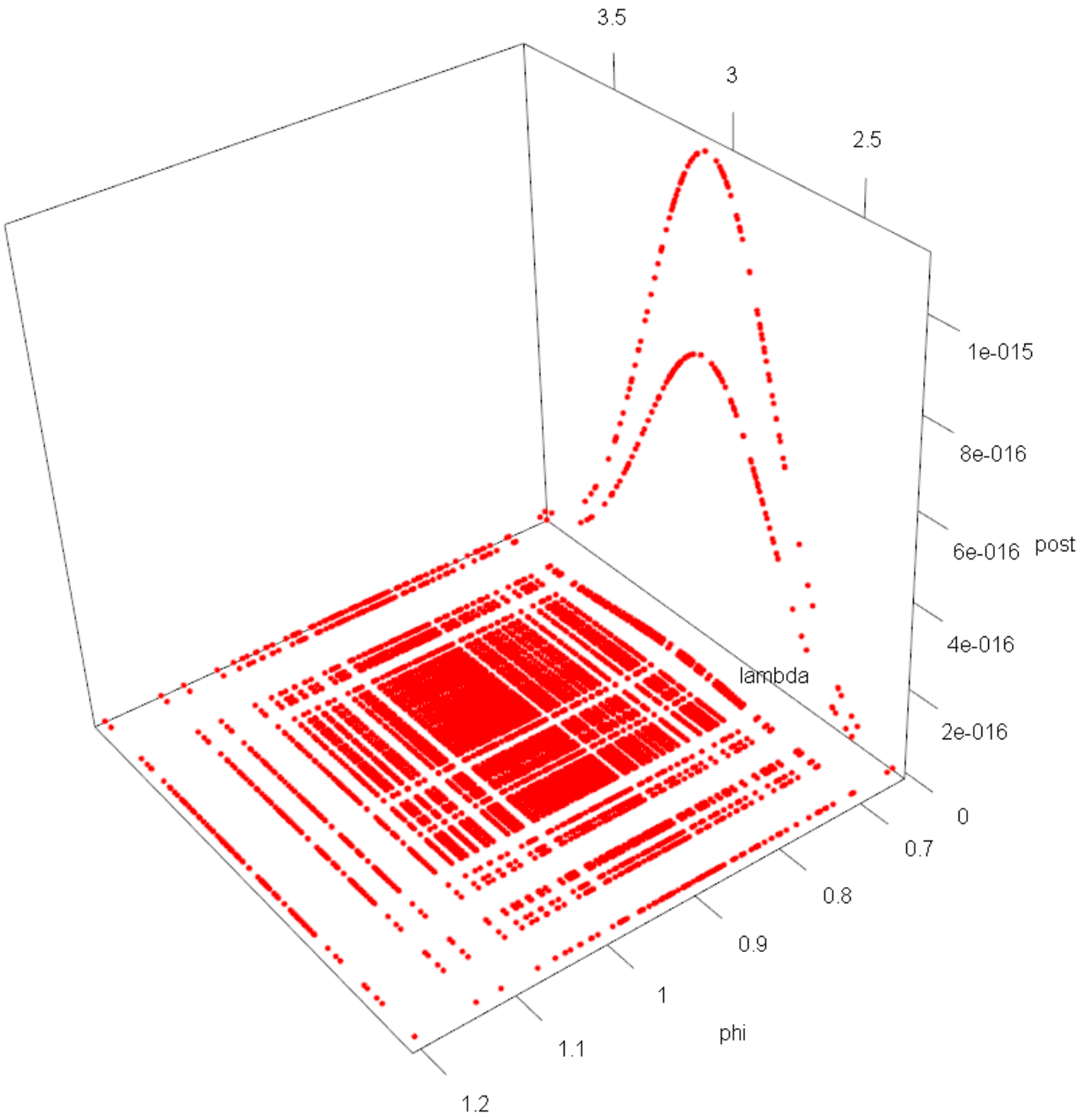

simulation vs full-conditional
Density
0.0
0.5
1.0
1.5
2.5
3.0
3.5
4.0
lambda


simulation vs full-conditional
Density
0
1
2
3
0.6
0.8
1.0
1.2
1.4
phi

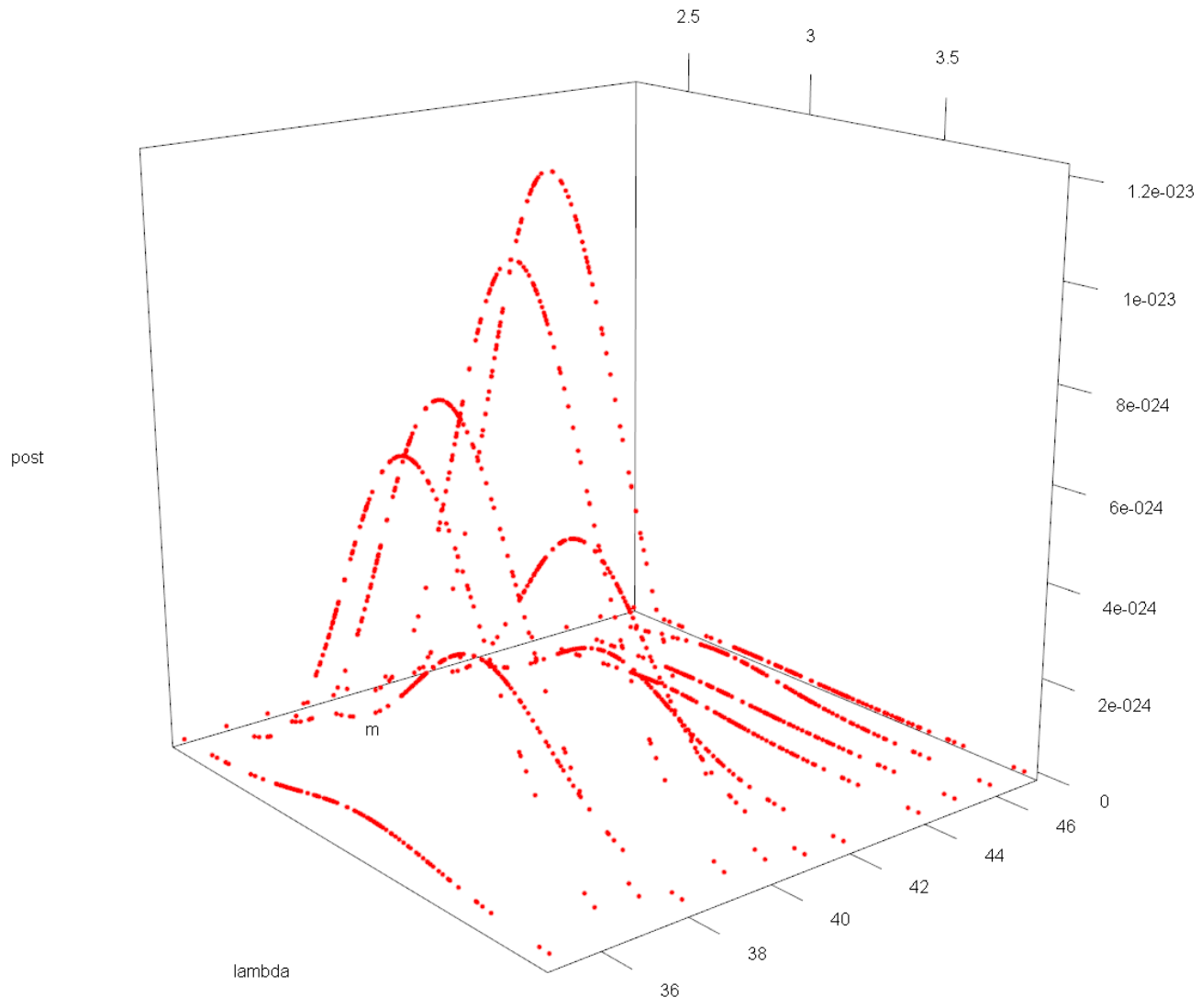


Furthermore we can now see how well the simulation approximate the posterior distribution. To make a scatter-plot with the posterior distribution we need to fix at least one variable, since its domain has three dimensions. We are going to fix then $m = \dot{m} = 40$ and see how the combinations of $\hat{\lambda}$ and $\hat{\phi}$ behave in the posterior distribution. For computational reason we will use just the last 100 values of the chains.

```
nccz= length(lamdaSim)
loll = nccz - 100
xp <- lamdaSim[loll:nccz]
yp <- phiSim[loll:nccz]
np = length(xp)
Zp<-rep(0, np^2)
Xp<-rep(0, np^2)
Yp<-rep(0, np^2)
zi = 1
cc = 1
for (ip in 1:np) {
  for (jp in 1:np) {
    Zp[zi]=post(c(xp[ip],yp[jp],moda))
    Xp[zi]=xp[ip]
    Yp[zi]=yp[jp]
    zi = zi + 1}
  cc = cc + 1}
plot3d(x=Xp, y=Yp, z=Zp, type="p", col="red", xlab="lambda", ylab="phi", zlab="post",
       size=5, lwd=15, box=F)
```



To better explore the 3D model please run the code and change zoom and angle on the screen. The maximum in the plot doesn't overlap perfectly with our approximation, but the fact that our simulation is close to it, makes our simulation more trustworthy.
It is also interesting the shape for the 3D scatter-plot when we fix $\lambda = \bar{\lambda} = 0.9077287$. The situation is anyway similar to the previous one.



4.5 Question (e)

We already showed that if we take the approximation of m equal to the mode $\dot{m}$, our approximation will be 40. Then the last year where $Y_i \sim Pois(\lambda)$ is Y_{40}. The next year Y_{41} has a different distribution $Pois(\phi)$. So for $m = 41$ there has been a change. By summing then 1850 to 40 and 41 we get that the most likely years between which we had a change in the rate of mining disaster is 1890-1891, (where of course 1891 is precisely the year where there has been the change).
Maybe it is not a coincidence that the *Wikipedia* page of *History of coal mining* reports that:
"Since 1890, coal mining has also been a political and social issue. Coal miners' labour and trade unions became powerful in many countries in the 20th century, and often, the miners were leaders of the Left or Socialist movements."
Then we can say that it's even more likely that between those exact two years there has been a change in rate of disasters, because probably in the same years the issue was brought to attention to the entire society.

4.6 Question (f)

To compute the expected reduction in percentage of the rate of accidents we will use $\bar{\lambda}$ and $\bar{\phi}$ as they give an estimate of the expected number of accidents in the two set of years. The expected reduction will be then computed as follows:

$$\frac{\bar{\phi} - \bar{\lambda}}{\bar{\lambda}} \cdot 100 = -70.82738\%$$

5 Exercise 5

5.1 Question (a)

Below is reported the full process to derive all the four full-conditionals.

5.1.1 The model

$$\mu_i = \alpha - \beta\gamma^{x_i}$$
$$Y_i \sim N(\mu_i, \tau^2)$$

where Y_i is the length and x_i is the age of the specimen of dugong.

$$\alpha \sim N(0, \sigma_\alpha^2) \quad \alpha \in (1, +\infty)$$
$$\beta \sim N(0, \sigma_\beta^2) \quad \beta \in (1, +\infty)$$
$$\gamma \sim Unif(0, 1) \quad \gamma \in [0, 1]$$
$$\tau^2 \sim /\Gamma(a, b) \quad \tau^2 \in (0, +\infty)$$

5.1.2 The likelihood function

$$L(\alpha, \beta, \gamma, \tau^2 | Y, x) = \left(\frac{1}{\sqrt{2\pi\tau^2}}\right)^n exp\left\{-\frac{1}{2\tau^2}\sum_{i=1}^{n}(Y_i - \alpha + \beta\gamma^{x_i})^2\right\}$$
$$(\alpha, \beta, \gamma, \tau^2) \in (1, +\infty) \times (1, +\infty) \times [0, 1] \times (0, +\infty)$$

5.1.3 The priors formulas

$$\pi(\alpha) = \frac{1}{\sqrt{2\pi\sigma_\alpha^2}} exp\left\{-\frac{\alpha^2}{2\sigma_\alpha^2}\right\} \propto exp\left\{-\frac{\alpha^2}{2\sigma_\alpha^2}\right\}$$
$$\pi(\beta) = \frac{1}{\sqrt{2\pi\sigma_\beta^2}} exp\left\{-\frac{\beta^2}{2\sigma_\beta^2}\right\} \propto exp\left\{-\frac{\beta^2}{2\sigma_\beta^2}\right\}$$
$$\pi(\gamma) = I_{[0,1]}(\gamma)$$
$$\pi(\tau^2) = \frac{b^a}{\Gamma(a)} \tau^{2(-a-1)} exp\left\{-\frac{b}{\tau^2}\right\} \propto \tau^{2(-a-1)} exp\left\{-\frac{b}{\tau^2}\right\}$$

5.1.4 The posterior distribution

$$\pi(\alpha,\beta,\gamma,\tau^2|Y,x) \propto$$

$$\propto L(\alpha,\beta,\gamma,\tau^2|Y,x)\cdot\pi(\alpha)\cdot\pi(\beta)\cdot\pi(\gamma)\cdot\pi(\tau^2)\propto$$

$$\propto \left[\left(\tfrac{1}{\sqrt{2\pi\tau^2}}\right)^n exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}\right]\cdot\left[exp\left\{-\tfrac{\alpha^2}{2\sigma_\alpha^2}\right\}\right]\cdot\left[exp\left\{-\tfrac{\beta^2}{2\sigma_\beta^2}\right\}\right]\cdot\left[\tau^{2(-a-1)}exp\left\{-\tfrac{b}{\tau^2}\right\}\right]$$

$$(\alpha,\beta,\gamma,\tau^2)\in(1,+\infty)\times(1,+\infty)\times[0,1]\times(0,+\infty)$$

5.1.5 The full-conditional for α

By picking from the posterior all the factors containing α we find that:

$$\pi(\alpha|\beta,\gamma,\tau^2,Y,x)\propto$$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}\cdot exp\left\{-\tfrac{\alpha^2}{2\sigma_\alpha^2}\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(\cancel{Y_i^2}+\alpha^2+\cancel{\beta^2\gamma^{2x_i}}-2\alpha Y_i+\cancel{2\beta Y_i\gamma^{x_i}}-2\alpha\beta\gamma^{x_i})-\tfrac{\alpha^2}{2\sigma_\alpha^2}\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\left[n\alpha^2-2\alpha\textstyle\sum_{i=1}^n(Y_i+\beta\gamma^{x_i})\right]-\tfrac{\alpha^2}{2\sigma_\alpha^2}\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2}\left[\tfrac{\alpha^2(n\sigma_\alpha^2+\tau^2)-2\alpha\sigma_\alpha^2\sum_{i=1}^n(Y_i+\beta\gamma^{x_i})}{\tau^2\sigma_\alpha^2}\right]\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2}\left[\tfrac{\alpha^2-2\alpha(\frac{\sigma_\alpha^2\sum_{i=1}^n(Y_i+\beta\gamma^{x_i})}{n\sigma_\alpha^2+\tau^2})}{\frac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2+\tau^2}}\right]\right\}\sim N\left(\tfrac{\sigma_\alpha^2\sum_{i=1}^n(Y_i+\beta\gamma^{x_i})}{n\sigma_\alpha^2+\tau^2},\tfrac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2+\tau^2}\right)$$

$$\alpha\in(1,+\infty)$$

Given the bounded domain, the full-conditional for α is a truncated normal.

5.1.6 The full-conditional for β

With a really similar method for β we find that:

$$\pi(\beta|\alpha,\gamma,\tau^2,Y,x)\propto$$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}\cdot exp\left\{-\tfrac{\beta^2}{2\sigma_\beta^2}\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(\cancel{Y_i^2}+\cancel{\alpha^2}+\beta^2\gamma^{2x_i}-\cancel{2\alpha Y_i}+2\beta Y_i\gamma^{x_i}-2\alpha\beta\gamma^{x_i})-\tfrac{\beta^2}{2\sigma_\beta^2}\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2}\left[\tfrac{\beta^2\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+2\beta\sigma_\beta^2\sum_{i=1}^n Y_i\gamma^{x_i}-2\alpha\beta\sigma_\beta^2\sum_{i=1}^n\gamma^{x_i}+\tau^2\beta^2}{\tau^2\sigma_\beta^2}\right]\right\}\propto$$

$$\propto exp\left\{-\tfrac{1}{2}\left[\tfrac{\beta^2(\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2)+2\beta(\sigma_\beta^2\sum_{i=1}^n Y_i\gamma^{x_i}-\alpha\sigma_\beta^2\sum_{i=1}^n\gamma^{x_i})}{\tau^2\sigma_\beta^2}\right]\right\}\propto$$

$$\propto exp\left\{-\frac{1}{2}\left[\frac{\beta^2-2\beta(\frac{-\sigma_\beta^2\sum_{i=1}^n Y_i\gamma^{x_i}+\alpha\sigma_\beta^2\sum_{i=1}^n\gamma^{x_i}}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2})}{\frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2}}\right]\right\}\sim N\left(\frac{\sigma_\beta^2\sum_{i=1}^n(\alpha-Y_i)\gamma^{x_i}}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2},\frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2}\right)$$

$$\beta\in(1,+\infty)$$

Also β has then a truncated normal full-conditional.

5.1.7 The full-conditional for γ

For γ is pretty straightforward:

$\pi(\gamma|\alpha,\beta,\tau^2,Y,x)\propto$

$$\propto exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}I_{[0,1]}(\gamma)$$

which cannot be recognized within any standard parametric family of distribution.

5.1.8 The full-conditional for τ^2

$\pi(\tau^2|\alpha,\beta,\gamma,Y,x)\propto$

$$\propto\left(\tfrac{1}{\sqrt{2\pi\tau^2}}\right)^n exp\left\{-\tfrac{1}{2\tau^2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}\tau^{2(-a-1)}exp\left\{-\tfrac{b}{\tau^2}\right\}\propto$$

$$\propto\tfrac{1}{\tau^{2(\frac{n}{2})}\tau^{2(a+1)}}exp\left\{\tfrac{-\frac{1}{2}\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2-b}{\tau^2}\right\}\propto$$

$$\propto\tfrac{1}{\tau^{2(\frac{n}{2}+a+1)}}exp\left\{-\tfrac{b+\frac{1}{2}\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2}{\tau^2}\right\}\sim/\Gamma\left(\tfrac{n}{2}+a,b+\tfrac{1}{2}\textstyle\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right)$$

$$\tau^2\in(0,+\infty)$$

Then the full-condition of γ has an inverse gamma distribution.

5.2 Question (b)

To summarize:

$$\pi(\alpha|\beta,\gamma,\tau^2,Y,x) \sim N\left(\frac{\sigma_\alpha^2\sum_{i=1}^n(Y_i+\beta\gamma^{x_i})}{n\sigma_\alpha^2+\tau^2}, \frac{\tau^2\sigma_\alpha^2}{n\sigma_\alpha^2+\tau^2}\right)$$
$$\alpha \in (1,+\infty)$$

$$\pi(\beta|\alpha,\gamma,\tau^2,Y,x) \sim N\left(\frac{\sigma_\beta^2\sum_{i=1}^n(\alpha-Y_i)\gamma^{x_i}}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2}, \frac{\tau^2\sigma_\beta^2}{\sigma_\beta^2\sum_{i=1}^n\gamma^{2x_i}+\tau^2}\right)$$
$$\beta \in (1,+\infty)$$

$$\pi(\gamma|\alpha,\beta,\tau^2,Y,x) \propto exp\left\{-\frac{1}{2\tau^2}\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right\}$$
$$\gamma \in [0,1]$$

$$\pi(\tau^2|\alpha,\beta,\gamma,Y,x) \sim /\Gamma\left(\frac{n}{2}+a, b+\frac{1}{2}\sum_{i=1}^n(Y_i-\alpha+\beta\gamma^{x_i})^2\right)$$
$$\tau^2 \in (0,+\infty)$$

Then the only full-conditional without recognizable distribution is the one for γ. So we can now implement Metropolis-within-Gibbs in which we perform a Metropolis–Hastings random walk for drawing simulation for γ.

5.3 Question (c)

The code:

```
#install.packages('msm', repos = 'http://cran.r-project.org')
library(msm)
set.seed(123)
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,
    red↪ 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)
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,
    red↪  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 = length(y)

sigmaAlfa2 = 10000
sigmaBeta2 = 10000
a = 0.001
b = 0.001

nchain = 10000

alfaChainOrig<-rep(0, nchain)
```

```
betaChainOrig<-rep(0, nchain)
gammaChainOrig<-rep(0, nchain)
tau2ChainOrig<-rep(0, nchain)
Y20ChainOrig<-rep(0, nchain)
Y30ChainOrig<-rep(0, nchain)

alfa<-rnorm(1, mean = 0, sd = sqrt(sigmaAlfa2))

while(!is.finite(alfa) || alfa<1){
  alfa<-rnorm(1, mean = 0, sd = sqrt(sigmaAlfa2)) }

beta<-rnorm(1, mean = 0, sd = sqrt(sigmaBeta2))

while(!is.finite(beta) || beta<1){
  beta<-rnorm(1, mean = 0, sd = sqrt(sigmaBeta2)) }

gamma <- runif(1)

tau2<-1/rgamma(1,0.001, 0.001)

while(!is.finite(tau2)|| tau2<0){
  tau2<-1/rgamma(1,a, b) }

alfaMu <-function(gamma, tau2, beta) {
  num = sigmaAlfa2*sum(beta*gamma^x+y)
  den = (n*sigmaAlfa2+tau2)
  return(num/den) }

betaMu <-function(gamma, tau2, alfa) {
  num = sigmaBeta2*sum((alfa-y)*gamma^x)
  den = (sigmaBeta2*sum(gamma^(2*x))+tau2)
  return(num/den) }

alfaVar <-function(tau2) {
  num = tau2*sigmaAlfa2
  den = sigmaAlfa2*n+tau2
  return(num/den) }

betaVar <-function(tau2,gamma) {
  num = tau2*sigmaBeta2
  den = sigmaBeta2*sum(gamma^(2*x))+tau2
  return(num/den) }



metrWithGibbsUnif <-function(alfa,beta,gamma,tau2) {
  delta=10^(-2)
  h = runif(1,gamma-delta,gamma+delta)

  if (h<0 || h>1) {
    numh = 0 }

  numh = -sum((y-alfa+beta*h^(x))^2)
  numg = -sum((y-alfa+beta*gamma^(x))^2)
  den = 2*tau2

  fnum = exp(numh/den)
  fden = exp(numg/den)
```

```
  p = min(1,(fnum)/(fden))
  if(!is.finite(p)){
    p = 1   }
  if(runif(1)<p){
    return(h)
  }
  else {
    return(gamma)   }}


for (s in 1:nchain) {
  print(s)
  alfaChainOrig[s]<-alfa<-rtnorm(1, mean = alfaMu(gamma,tau2,beta), sd = sqrt(alfaVar(
      red↪ tau2)),lower = 1, upper = Inf)
  betaChainOrig[s]<-beta<-rtnorm(1, mean = betaMu(gamma,tau2,alfa), sd = sqrt(betaVar(
      red↪ tau2,gamma)),lower = 1, upper = Inf)
  gammaChainOrig[s]<-gamma<-metrWithGibbsUnif(alfa,beta,gamma,tau2)
  tau2ChainOrig[s]<-tau2<-1/rgamma(1,shape = a+n/2, rate=(b + sum((y-alfa+beta*gamma^(x)
      red↪ )^2)/2))
  Y20ChainOrig[s]<-rnorm(1,alfa-beta*gamma^20,sqrt(tau2))
  Y30ChainOrig[s]<-rnorm(1,alfa-beta*gamma^30,sqrt(tau2))}

lb = 1001

alfaChain = alfaChainOrig[lb:10000]
betaChain = betaChainOrig[lb:10000]
gammaChain = gammaChainOrig[lb:10000]
tau2Chain = tau2ChainOrig[lb:10000]
Y20Chain = Y20ChainOrig[lb:10000]
Y30Chain = Y30ChainOrig[lb:10000]


alfa = mean(alfaChain)
beta = mean(betaChain)
gamma = mean(gammaChain)
tau2 = mean(tau2Chain)
Y20 = mean(Y20Chain)
Y30 = mean(Y30Chain)
```

We could say that the Metropolis-within-Gibbs technique is represented by the function *metrWithGibbsUnif* that uses a Metropolis–Hastings random walk by drawing from a uniform with interval centered in the past γ and of length 2δ where $\delta = 10^{-2}$. With such drawn value we either decide to use it to update our chain of γ_i, or to not use it and to update the chain with the past γ we had, meaning $\gamma_t = \gamma_{t-1}$. To do that I used the random walk Metropolis-Hastings algorithm. We can see now the results comparing them with the MLE, which is also equal to the MAP, found in the past assignment of this course.

```
> ####MLE#########
> # alfa = 2.65  #
> # beta = 0.96  #
> # gamma = 0.87 #
> # tau2 = 0.008 #
> ################
>
> print(alfa)
```

```
[1] 2.679797
> print(beta)
[1] 1.053117
> print(gamma)
[1] 0.860189
> print(tau2)
[1] 0.01038834
```

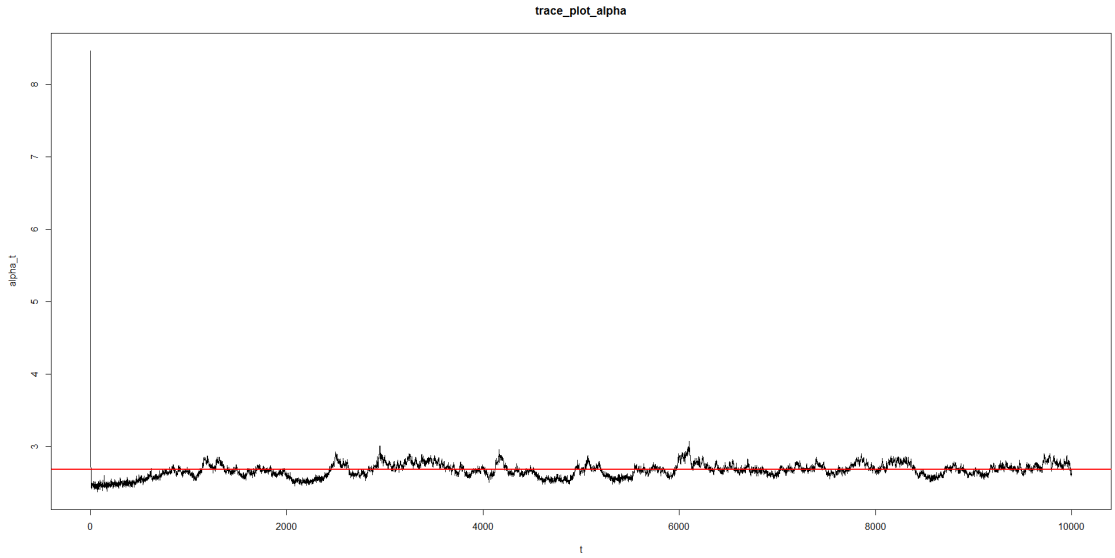
We can see how similar they are and this points out that Metropolis-within-Gibbs should be correct.

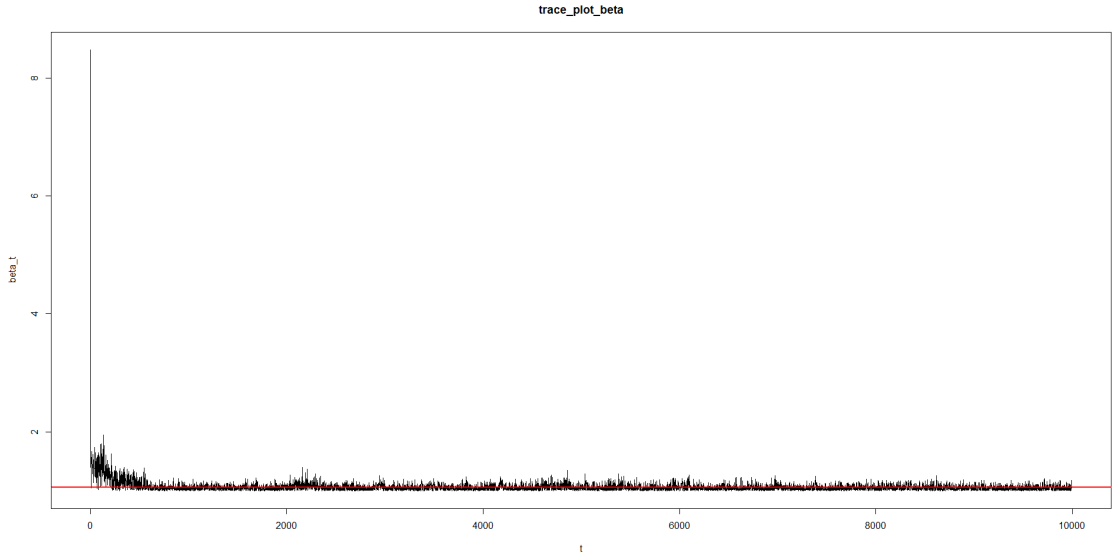
5.4 Question (d)

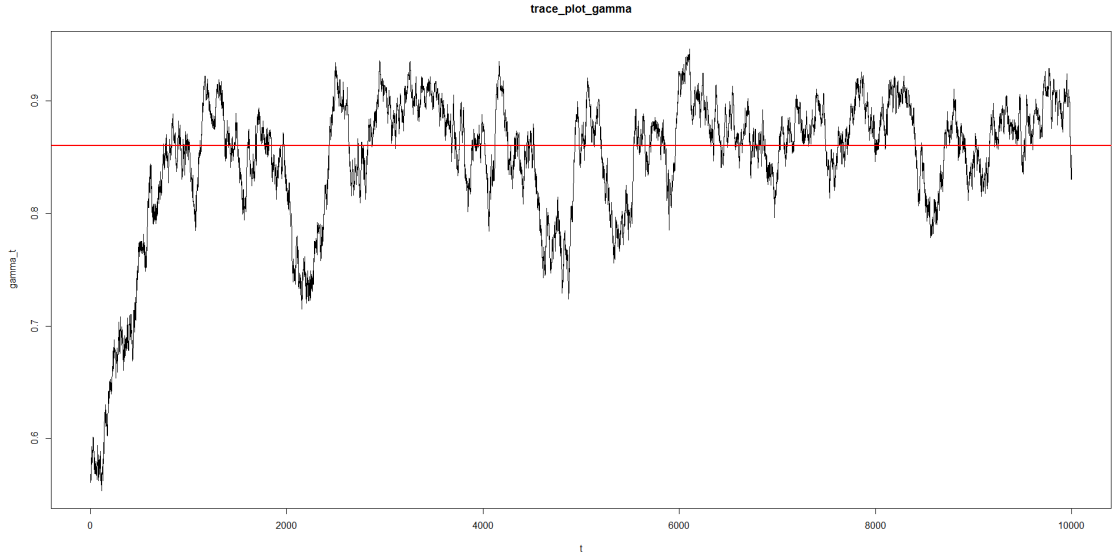
The trace-plots and the code to plot them is below. I decided to take away few starting points in the chain because they were really far from convergence, being drawn from the priors.

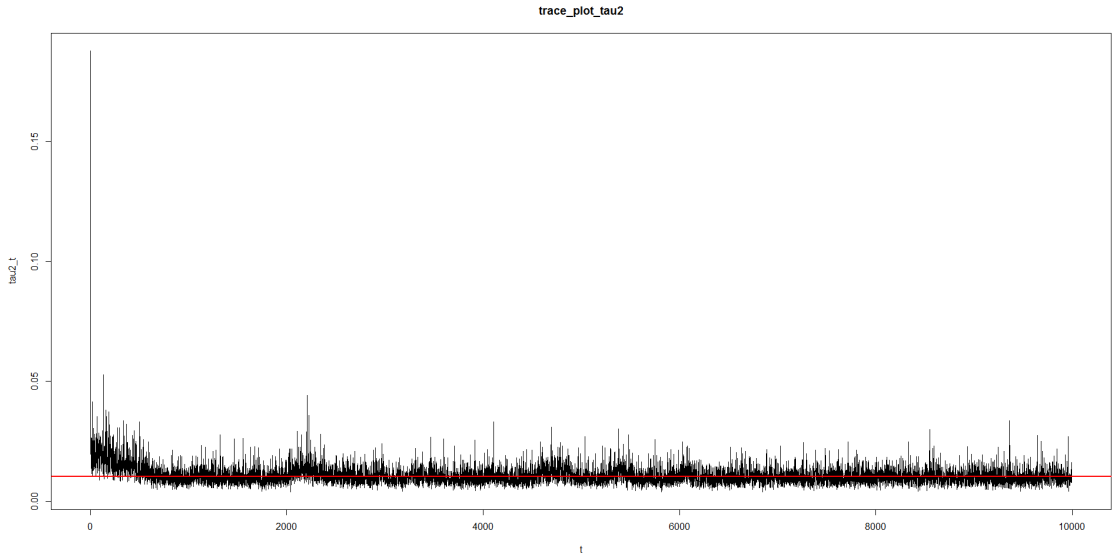
```
lb = 0

alfaChain = alfaChainOrig[lb:10000]
gammaChain = gammaChainOrig[lb:10000]
Y20Chain = Y20ChainOrig[lb:10000]
Y30Chain = Y30ChainOrig[lb:10000]


itVec=seq(1,length(alfaChain))
plot(itVec,alfaChain,type = 'l',main='trace_plot_alpha',xlab = 't',ylab = 'alpha_t')
abline(h=alfa, col='red', lw = 2)


betaChain = betaChainOrig[3:10000]
itVecbeta=seq(1,length(betaChain))

plot(itVecbeta,betaChain,type = 'l',main='trace_plot_beta',xlab = 't',ylab = 'beta_t')
abline(h=beta, col='red', lw = 2)

plot(itVec,gammaChain,type = 'l',main='trace_plot_gamma',xlab = 't',ylab = 'gamma_t')
abline(h=gamma, col='red', lw = 2)


tau2Chain = tau2ChainOrig[4:10000]
itVectau=seq(1,length(tau2Chain))

plot(itVectau,tau2Chain,type = 'l',main='trace_plot_tau2',xlab = 't',ylab = 'tau2_t')
abline(h=tau2, col='red', lw = 2)
```

The red line stands for the approximation values we compared before with the MLE, that are indeed the results of our simulation.


trace_plot_alpha
alpha_t
3
4
5
6
7
8
0
2000
4000
6000
8000
10000
t


trace_plot_beta
beta_t
2
4
6
8
0
2000
4000
6000
8000
10000
t


trace_plot_gamma
gamma_t
0.6
0.7
0.8
0.9
0
2000
4000
6000
8000
10000
t



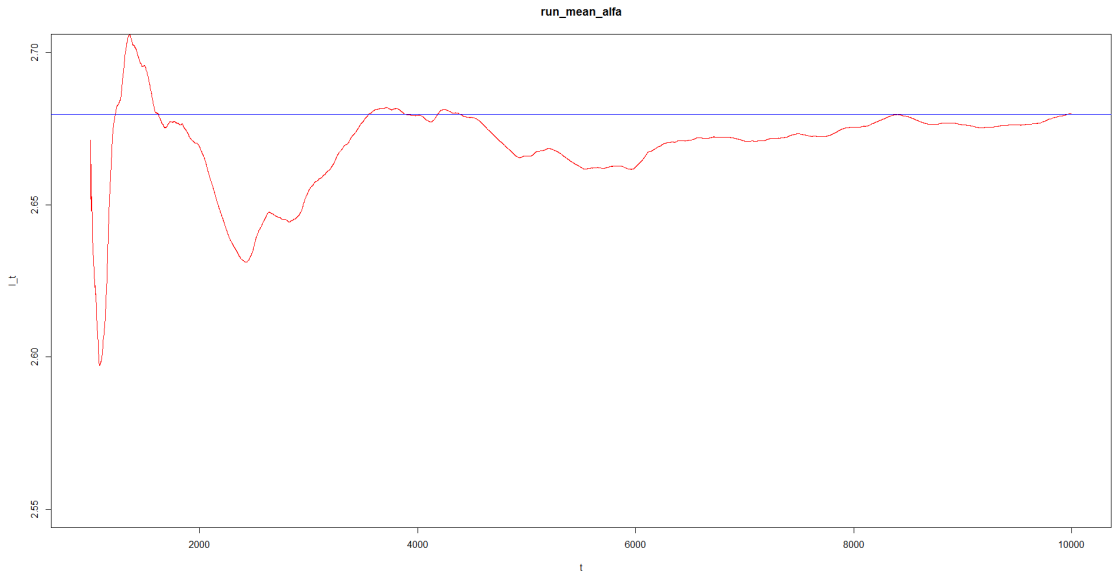
5.5 Question (e)

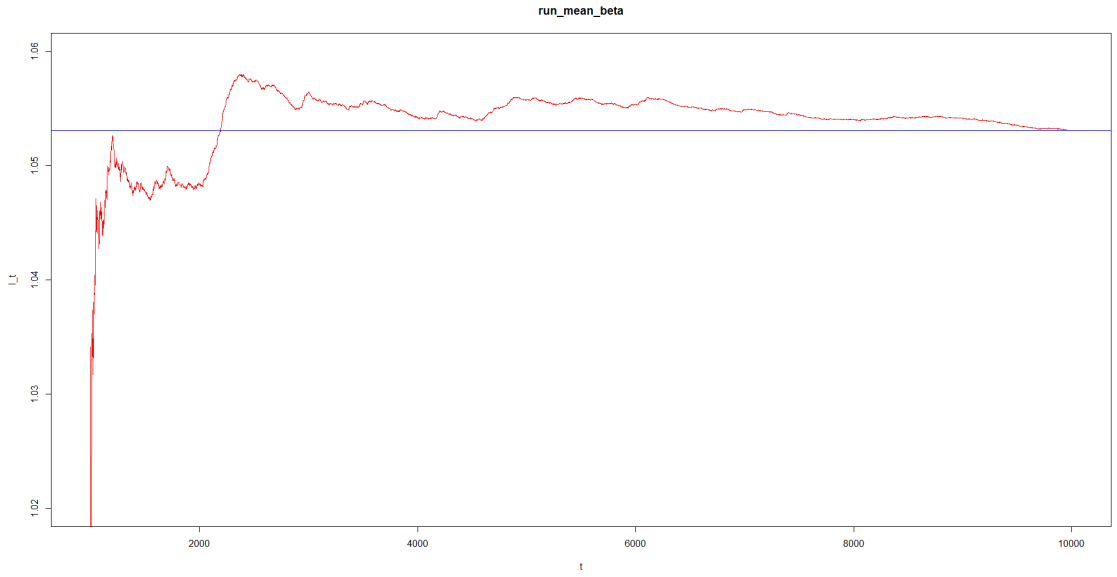
Those are the codes and plots for the running means of our simulations. Such plots could look different from what expected because I applied burn-in with $T_0 = 1000$.

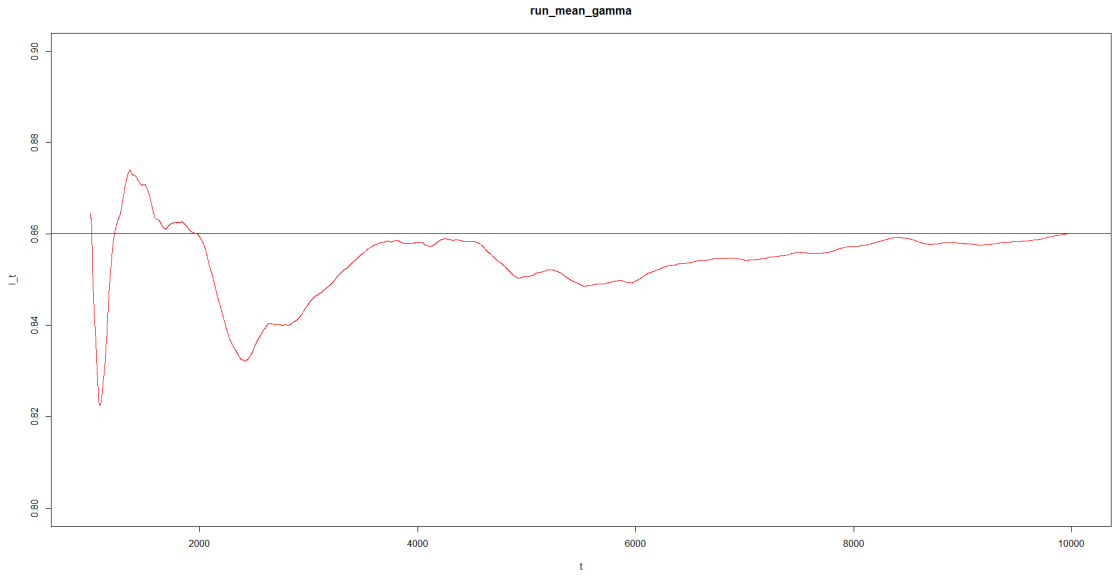
```
lb = 1001

alfaChain = alfaChainOrig[lb:10000]
betaChain = betaChainOrig[lb:10000]
gammaChain = gammaChainOrig[lb:10000]
tau2Chain = tau2ChainOrig[lb:10000]
Y20Chain = Y20ChainOrig[lb:10000]
Y30Chain = Y30ChainOrig[lb:10000]

#fancy:
x = alfaChain
iter = (lb):(length(x)+(lb-1))
runningmeans=cumsum(x)/(1:length(x))
plot(iter,runningmeans,type="l",col='red', main='run_mean_alfa', xlab = 't',ylab = 'I_t
    red↪ ',ylim = c(2.55,2.7))
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=alfa, col='green')

x = betaChain
runningmeans=cumsum(x)/(1:length(x))
plot(iter,runningmeans,type="l",col='red', main='run_mean_beta', xlab = 't',ylab = 'I_t
    red↪ ', ylim = c(1.02,1.06))
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=beta, col='green')

x = gammaChain
runningmeans=cumsum(x)/(1:length(x))
plot(iter,runningmeans,type="l",col='red', main='run_mean_gamma', xlab = 't',ylab = 'I_t
    red↪ ', ylim = c(0.8,0.9))
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=gamma, col='green')

x = tau2Chain
runningmeans=cumsum(x)/(1:length(x))
```

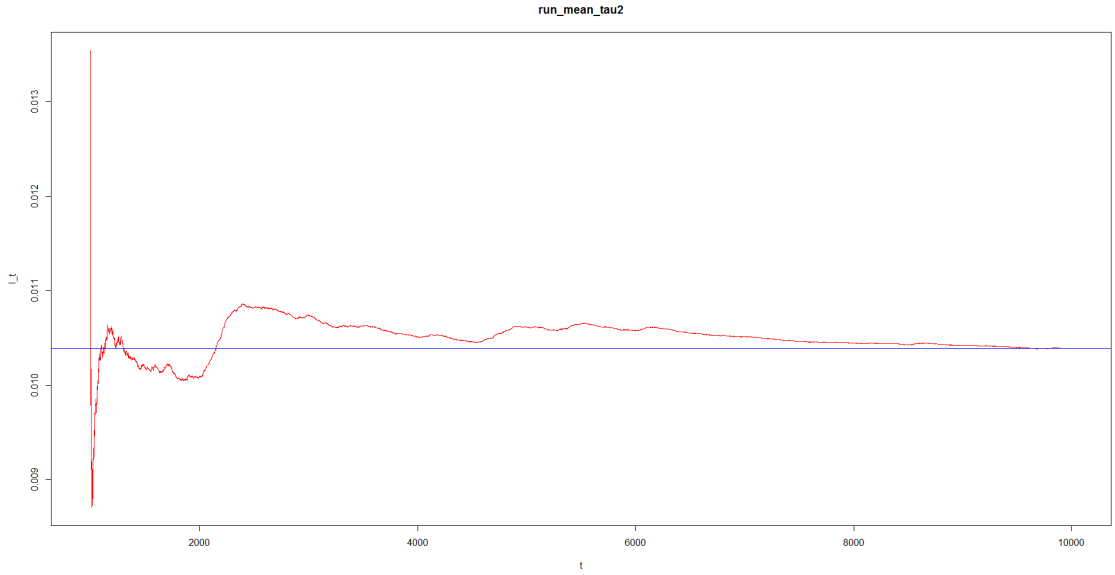
```
plot(iter,runningmeans,type="l",col='red', main='run_mean_tau2', xlab = 't',ylab = 'I_t
    red ↪ ')
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=tau2, col='green')

x = Y20Chain
runningmeans=cumsum(x)/(1:length(x))
plot(iter,runningmeans,type="l",col='red', main='run_mean_Y20', xlab = 't',ylab = 'I_t')
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=Y20, col='green')

x = Y30Chain
runningmeans=cumsum(x)/(1:length(x))
plot(iter,runningmeans,type="l",col='red', main='run_mean_Y30', xlab = 't',ylab = 'I_t')
abline(h=runningmeans[length(runningmeans)], col='blue')
abline(h=Y30, col='green')
```

The blue line stands for the final mean and of course each plot converges to it.









5.6 Question (f)

We already showed our approximation in section 5.3, but to be more precise we will show them once again together with the approximation error.

```
> print(alfa)
[1] 2.679797
> print(errAlfa)
[1] -0.0002872513
> print(beta)
[1] 1.053117
> print(errBeta)
[1] -3.772156e-06
> print(gamma)
```

```
[1] 0.860189
> print(errGamma)
[1] -8.810475e-05
> print(tau2)
[1] 0.01038834
> print(errTau2)
[1] -2.16205e-08
```

To compute the approximation we just made the average of our simulations taking away the first 1000 values.

$$\hat{I} = \sum_{t=1001}^{10000} X_i$$

To compute approximation error instead we used an estimator of the variance of $\hat{I}$ using the following formulas, also explained in section 2.5.

$$\mathbb{E}\left[(\hat{I}_t - I)^2\right] = Var_\pi(\hat{I}_t) \simeq \hat{Var}_\pi(\hat{I}_t) = \frac{1}{t}\left(\hat{\gamma}_0 + 2\sum_{k=1}^{t-1}\hat{\gamma}_k\right)$$

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

To do this it required more computational time than in any other computation. The following function has been used in R:

```
gamma_k <- function(k,vet,Icap) {
  t = length(vet)
  somma = 0
  for (ki in 1:(t-k)) {
    somma = somma + (vet[ki] - Icap)*(vet[ki+k] - Icap) }
  return(somma/(t-k))}


varCap <- function(vet,Icap) {
  somma = 0
  tVar = length(vet)
  print('Summing up all gamma_k, from k = 1 to:')
  print(tVar)
  print('Starting now..')
  print(1)
  for (klol in 1:(tVar-1)) {
    if (klol%%500==0) {
      print(klol) }
    somma = somma + gamma_k(klol,vet,Icap) }
  num = gamma_k(0,vet,Icap) + 2*somma
  return(num/tVar) }
```

5.7 Question (g)

To measure the posterior uncertainty of each parameter we used the following coefficient c:

$$c = \frac{\sqrt{\hat{Var}(\hat{I}_t)}}{\hat{I}_t}$$

Such coefficient measures how big the error is compared to the value of the approximation. The idea is that given two parameters with the same error, there is more uncertainty on the parameter with smaller value.
The results are:

```
> coeffAlfa = sqrt(abs(errAlfa))/alfa
> coeffBeta = sqrt(abs(errBeta))/beta
> coeffGamma = sqrt(abs(errGamma))/gamma
> coeffTau2 = sqrt(abs(errTau2))/tau2
>
> coeffAlfa
[1] 0.006324541
> coeffBeta
[1] 0.001844243
> coeffGamma
[1] 0.01091204
> coeffTau2
[1] 0.01415424
```

Then the parameter with higher posterior uncertainty is τ^2.

5.8 Question (h)

Below the computed correlations in absolute values:

```
cor_Alfa_Beta
[1] 0.09874042

cor_Alfa_Gamma
[1] 0.9223362

cor_Alfa_Tau2
[1] 0.06812645

cor_Beta_Gamma
[1] 0.1191487

cor_Beta_Tau2
[1] 0.2426529

cor_Gamma_Tau2
[1] 0.1676563
```

The highest correlation is then between α and γ.

5.9 Question (i)

If you look back at the code in section 5.3 you can see that the Markov chains implemented were 6 while the parameters we are estimating are only 4. This because to predict $\hat{Y}_{20}$ and $\hat{Y}_{30}$ we run their simulation along the other ones, drawing each $\hat{Y}_{20,t}$ and each $\hat{Y}_{30,t}$ from respectively $N(\alpha_t - \beta_t\gamma_t^{20}, \tau_t^2)$ and $N(\alpha_t - \beta_t\gamma_t^{30}, \tau_t^2)$ with $t \in [1, 10000]$.
The results for $\hat{Y}_{20}$ are below:

```
> print(Y20)
[1] 2.606334
> print(errY20)
[1] -3.371105e-05
```

5.10 Question (j)

The results for $\hat{Y}_{30}$ are below:

```
> print(Y30)
[1] 2.655799
> print(errY30)
[1] -0.0002040558
```

5.11 Question (k)

To evaluate the precision P of a prediction we use the estimate of the approximation error computed as follows:

$$P = \frac{1}{|\hat{Var}(\hat{I}_t)|}$$

We find that:

```
> precY20 = 1 / abs(errY20)
> precY30 = 1 / abs(errY30)
>
> print(precY20)
[1] 29663.87
> print(precY30)
[1] 4900.62
```

Then, since $\hat{Y}_{20}$ has higher P, it is more precise than $\hat{Y}_{30}$.