Estimating genomic copy number with hidden Markov Models

Rayan Charrier, Fang Goh, Sofiane Hadji
Supervisor : Élodie Vernet
Reviewer : Erwan Scornet

Abstract

La délétion ou l'insertion de gènes dans le génome est l'une des principales causes de cancer. La détection de variations du nombre de copies de gènes est ainsi devenue un enjeu majeur dans la lutte contre le cancer, nécessitant des outils statistique adaptés à la qualité des données habituellement récoltées. Une des méthodes développées récemment est celle présentée par Yau et al. en 2011 [1], utilisant des modèles de Markov caché. Nous avons essayé de reproduire les résultats de l'article dans le cas de distributions gaussiennes, en utilisant les mêmes données expérimentales.

Nous avons obtenu des résultats similaires, où la méthode statistique permet de détecter les variations de copies avec un nombre élevé de faux-positifs. Ces résultats mettent en évidence la nécessité d'utiliser des distributions plus complexes comme des mélanges de gaussiennes ou des distributions non-paramétriques - ce qui est aussi fait dans l'article - afin d'améliorer la spécificité de la méthode.

Both duplication and deletion of genes which were present in a genome have been found responsible for increasing the risk of cancer. As a result, detecting copy-number variations (CNVs) has become instrumental in fighting cancer, which in turn requires better statistical tools to handle the quality of data. One recent paper, by Yau et al. in 2011 [1], develops a new method based on Hidden Markov models to estimate CNVs. We studied the case of Gaussian distributions, using the same experimental data.

In the end, we obtained results similar to those in the paper. We detected most CNVs but with a high number of false-positives. In order to refine our findings, it would be necessary to use more complex distributions such as Gaussian mixture models or non-parametric distributions - as proposed in the paper - in order to improve the specificity of the method.

Contents

1 Problem statement

The genetic information of each individual is stored in every cell of its body under the form of a DNA molecule. The DNA molecule of human beings is made up of more than 3 billions pairs of nucleotides, representing about 27 000 genes. We refer to the term "pairs of nucleotides" because the human genome is made up of 23 pairs of chromosomes : 23 chromosomes from the mother and 23 from the father (see Figure 1).

For a long time, scientists had held that genes were always present in two copies in a genome. However, recent discoveries have demonstrated that copy numbers can widely vary in large sections of DNA. For some individuals, genes can occur in one, three or more than three copies, sometimes even missing altogether.

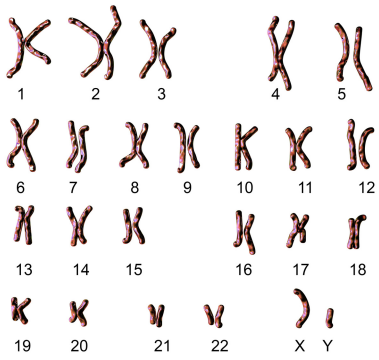


Figure 1: Karyotype of a human male

In particular, the duplication or the deletion of some genes can favor or hinder the developement of cancers. Some genes called "tumour suppressor genes" indeed tend to hinder the abnormal proliferation of cancer cells, while oncogenes favor that proliferation. Deleting or inserting such genes can thus cause cancers and detecting them is a crucial issue for research in cancer control.

In order to detect those variations, Lakshmi et al. developed in 2006 [2] a technique of *comparative genomic hybridization* to measure more precisely copy-number variations.

This technique use microarrays in order to target specific genes. The intensity of the signal returned by those microarrays is proportional to the number of genes in the analyzed sequence, and the intensity ratio between two different groups of cells can be used to study copy-number variations.

The data we worked on consisted in the intensities of control cells and of cancerous mouse cells. In order to study those variations we thus define the following quantity:

$$Y = log(\frac{I_{tumour}}{I_{no\ tumour}})$$

2 Modeling choice

2.1 Change detection

Detecting deletions or insertions is basically a challenge of **change detection**: we try to distinguish between sequences of constant mean (see Figure 2).

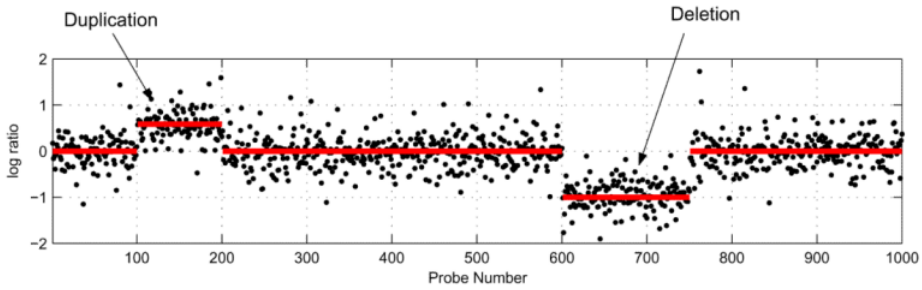


Figure 2: Logarithm of the ratio for a sequence of ADN

However, measuring directly these segmental changes with traditional methods is difficult, because of their sensitivity to DNA quality and instrumental noise [1]. As a result, we are going to exploit more efficient statistical methods (hidden Markov models and Bayesian statistics) in order to take advantage of correlation among data.

2.2 Hidden Markov model

2.2.1 Presentation

A hidden Markov model (or *HMM*) is a couple of stochastic processes $((X_t)_{t\geq 1}, (Y_t)_{t\geq 1})$ such that

- $(X_t)_{t\geq 1}$ is a Markov chain on a finite state space
- conditional on $(X_t)_{t\geq 1}$, $(Y_t)_{t\geq 1}$ are independent and Y_t only depends on X_t.

$(Y_t)_{t\geq 1}$ can be interpreted as observations, taken at a time t, of a system which is in state X_t. Figure 3 is a graphical representation of this model.

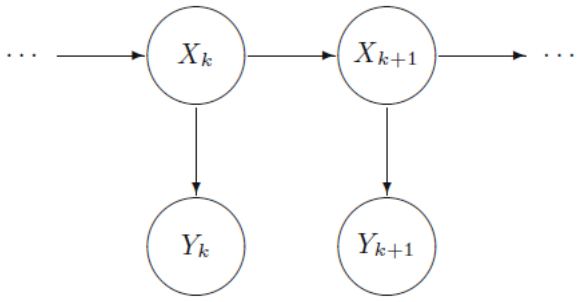


Figure 3: Dependency structure of a HMM

HMM are caracterized by how observations are generated : we suppose that the distribution of Y_t depends on the state X_t. For example, we could assume that Y_t is generated with a gaussian distribution $\mathcal{N}(0, 3)$ if the system is in state 1 ($X_t = 1$), and with a beta distribution $\beta(1, 1)$ if the system is in state 2 ($X_t = 2$). On figure 4, we can see a rudimentary HMM diagram.

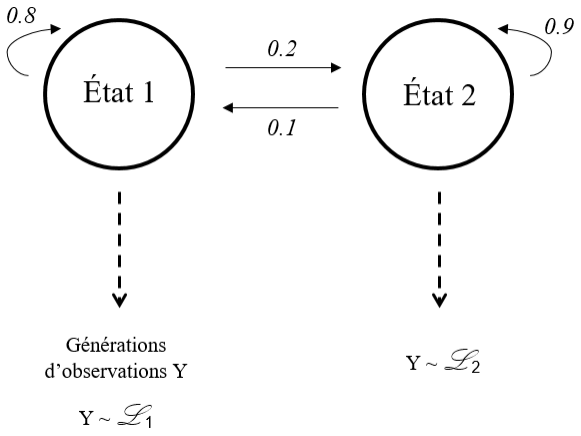


Figure 4: Observations generation diagram of a chain with 2 states

In HMM models, the observations $(Y_t)_{t\geq 1}$ are known, but not the states $(X_t)_{t\geq 1}$ (which is why the Markov chain is said to be *hidden*). The main goal is therefore to find out these hidden states $(X_t)_{t\geq 1}$.

2.2.2 Model used

In our case, t represents the position of sites all along the genome ($t \in [1, n]$ where n is the length of the genome), and the observations $(Y_t)_{t\geq 1}$ are those introduced in Part 1 (logarithm of the ratio). The hidden states $(X_t)_{t\geq 1}$ are the gene copy numbers :

$$1 : \text{deletion}, 2 : \text{no change}, 3 : \text{insertion}.$$

Finally, we assume here that the probability distributions of the Y_t given the X_t are Gaussian (Yau et al. [1] also consider Gaussian mixtures and non- parametric distributions, but we didn't have enough time to try this out).

2.2.3 Parameters of the chain

Finally, we also need to estimate the parameters of the chain (a vector noted θ) in order to find the hidden states. These are the following :

- the probability distribution of chain ν
- the transition matrix Q
- les lois des observations selon les états Markoviens.

The number of states is supposed known (2, 3,...). The space state is $\mathcal{X}$.

3 Estimation of the model parameters

In this part, we explain how we estimated θ, the vector of the chain parameters, and the hidden states $(X_t)_{t\geq 1}$. We used an MCMC method implementing a random walk on the Markov chain : Gibbs sampling. Each parameter (an element of θ or $(X_t)_{t\geq 1}$) is generated iteratively, given the values of the other parameters.

3.1 Notations

Transition matrix

Let us consider an homogeneous Markov chain. The transition matrix Q is thus independant of t.

For $\mathcal{X} = \{1, 2, \ldots, d\}$ and i, j in $\mathcal{X}$ we have :

$$Q(i, j) = P(X_t = j | X_{t-1} = i) \tag{1}$$

Distributions of Y given X

As written previously, we consider in our work a model in which the distributions of the Y_t given the state X_t are Gaussian. Let us note respectively $\mu = (\mu_1, \mu_2, \ldots, \mu_d)$ et $v = (v_1, v_2, \ldots, v_d)$ the means and variances of the Gaussian laws corresponding to each state.

Taking the same notations as Cappé et al. in *Inference in Hidden Markov Models* (2005) [3], we introduce for every i i in $\mathcal{X}$ and t in $\{1 : n\}$ $g_t(i)$, defined as such :

$$g_t(i) = \frac{1}{\sqrt{2\pi v_i}} e^{-\frac{(y_t - \mu_i)^2}{2v_i}} \tag{2}$$

.

$g_t(i)$ corresponds to the conditional density function of Y_t given $X_t = i$ (evaluated in y_t). If the Y were discrete, we would interpret $g_t(i)$ as $P(Y_t = y_t | X_t = i)$ (cf. 3.3.2 case of discrete observations).

3.2 Bayesian inference

Before going into further details into Gibbs sampling, we thought useful to first introduce the notion of Bayesian inference.

By considering a random variable Z, which distribution depends on a parameter θ (probability distribution p_θ), this method enables to estimate the parameter θ given the realizations of Z.

We call **posterior** distribution the distribution of θ given Z, noted $\mathcal{L}(\theta|Z)$ and of density $f_{\theta|Z}$.

In order to estimate this distribution, we choose a **prior** distribution Π on θ of density π. This distribution contains all available information on θ before any input data : it is generally determined by scientific experts and/or experimental results.

With Bayes's law, we obtain (with μ measure on the probability space):

$$f_{\theta|Z=z}(\theta) = \frac{p_\theta(x)\pi(\theta)}{\int p_\theta(x)\pi(\theta)d\mu(\theta)} \tag{3}$$

In most cases, the posterior distribution is hard to determine. Choosing *conjugate* distributions allows us to simplify calculations.

Indeed, a distributions family is said to be conjugate if each law of this family - treated as a prior distribution - gives posterior distributions in the same family. *The distribution of Z given θ also matters, but contrary to the prior distribution, we do not choose it.*

For instance, the Gaussian family is conjugate to itself. We can also mention the Beta distribution and Inverse-Gamma distributions.

3.3 Gibbs sampling

Let us remind that $\theta = (\theta_1, \dots, \theta_l)$ is the vector of parameters of the Markov chain (cf Partie 2). We also note $X = (X_1, \dots, X_n)$.

We define $\tilde{\theta}$ =$(\theta_1, \dots, \theta_l, X) = (\nu, Q, \mu_1, \dots, \mu_d, v_1, \dots, v_d, X)$.

We are looking for an estimate of the expression of $\tilde{\theta}$, but a direct computation with the likelihood ends up with an expression which is too complicated : this is why we use a MCMC method (Markov chain Monte Carlo), in order to sample according to the distribution of $\tilde{\theta}$. More specifically, we used Gibbs sampling, which defines a random walk on a Markov chain whose stationnary distribution is the one we want to simulate (the distribution of $\tilde{\theta}$ [4]).

The principle of the method is to iteratively generate each of the coordinates of $\tilde{\theta}$, conditionnaly to the others. At each step k, we then need to estimate the distribution of :

- $\tilde{\theta_1} \mid \tilde{\theta_2} = \tilde{\theta_2^k}, \dots, \tilde{\theta_L} = \tilde{\theta_L^k}$ to generate $\theta_1^{\tilde{k+1}}$
- $\tilde{\theta_2} \mid \tilde{\theta_1} = \theta_1^{\tilde{k+1}}, \tilde{\theta_3} = \tilde{\theta_3^k}, \dots, \tilde{\theta_L} = \tilde{\theta_L^k}$ to generate $\theta_2^{\tilde{k+1}}$
- ...
- $\tilde{\theta_L} \mid \tilde{\theta_1} = \theta_1^{\tilde{k+1}}, \dots, \tilde{\theta_{L-1}} = \theta_{L-1}^{\tilde{k+1}}$ to generate $\theta_L^{\tilde{k+1}}$

with $L = l + 1$ size of vector $\tilde{\theta}$.

Even though the vector ν of the initial probabilities of the chain states appears in the expressions of the posteriors, it does not really influence the estimations and we choose is as constant $\nu = (\frac{1}{d}, \dots, \frac{1}{d})$ (we suppose that n, the number of observations, is big enough to assume a quasi-stationnary distribution of the states).

For the other parameters of the chain, we can compute their **posterior** distributions by choosing conjugate **prior** distributions (3.3.1). However, for the states X, this is more complicated and we need to use an algorithm called *Forward-Backward* (3.3.2).

Once the chain parameters have been generated enough times, we use their average as an estimate. For X, each hidden state X_t is estimated by the most probable state (generated the most times).

In our study, the number of iterations is always K= 10 000, and we only keep the values generated after the 1000^th iteration (we consider the algorithm becomes stationnary after 1000 iterations).

3.3.1 Posterior distributions (chain parameters)

In this part, we present the posterior distributions for Markov models with two or three states.

I) Markov model with 2 states

We choose to define a transition matrix of the shape

$Q = \begin{bmatrix} q_{11} & 1-q_{11} \\ 1-q_{22} & q_{22} \end{bmatrix}$.

Then, we have $\theta = (q_{11}, q_{22}, \mu_1, \mu_2, v_1, v_2)$.

Prior distributions

As mentionned before, we choose conjugate distributions. Let us fix $\alpha, \beta, m, w, c, d$ such that the prior distribution of the parameters are the following :

$$q_{11}, q_{22} \sim Beta(\alpha, \beta), \quad \mu_1, \mu_2 \sim \mathcal{N}(m, w), \quad v_1, v_2 \sim IG(c, d). \tag{4}$$

Posterior distributions

For each parameter, we get the following posterior distributions :

$$q_{11} : Beta(\alpha + \sum_{i=1}^{n-1} 1_{\{X_i=1, X_{i+1}=1\}}, \beta + \sum_{i=1}^{n-1} 1_{\{X_i=1, X_{i+1}=2\}}) \tag{5}$$

$$q_{22} : Beta(\alpha + \sum_{i=1}^{n-1} 1_{\{X_i=2, X_{i+1}=2\}}, \beta + \sum_{i=1}^{n-1} 1_{\{X_i=2, X_{i+1}=1\}}) \tag{6}$$

$$\mu_1 : \mathcal{N}(\frac{mv_1 + S_1 w}{v_1 + N_1 w}, \frac{wv_1}{v_1 + N_1 w}) \tag{7}$$

$$\mu_2 : \mathcal{N}(\frac{mv_2 + S_2 w}{v_2 + N_2 w}, \frac{wv_2}{v_2 + N_2 w}) \tag{8}$$

$$v_1 : IG(c + \frac{N_1}{2}, d + \frac{1}{2} \sum_{i=1}^{n} (Y_i - \mu_1)^2 1_{X_i=1}) \tag{9}$$

$$v_2 : IG(c + \frac{N_2}{2}, d + \frac{1}{2} \sum_{i=1}^{n} (Y_i - \mu_2)^2 1_{X_i=2}) \tag{10}$$

où $N_1 = \sum_{i=1}^{n} 1_{X_i=1}$ et $N_2 = \sum_{i=1}^{n} 1_{X_i=2}$, $S_1 = \sum_{i=1}^{n} Y_i 1_{X_i=1}$ et $S_2 = \sum_{i=1}^{n} Y_i 1_{X_i=2}$.

Moreover, in order to avoid any issue of *label switching* which could hold back convergence of the algorithm (state 1 becomes state 2 and vice-versa), we keep on generating μ_1 et μ_2 as long as the condition $\mu_1 < \mu_2$ is not satisfied. As such, we generate μ_1, μ_2 with $\mu_1 < \mu_2$.

This results are proved in appendix I.

II) Markov model with 3 states

We then choose Q with the shape $Q = \begin{bmatrix} 1-\rho & \frac{\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{2} & 1-\rho & \frac{\rho}{2} \\ \frac{\rho}{2} & \frac{\rho}{2} & 1-\rho \end{bmatrix}$.

We have $\theta = (\rho, \mu_1, \mu_2, \mu_3, v_1, v_2, v_3)$.

Prior distributions

Taking the same notations as before, we choose the following prior distribution for ρ :

$$\rho \sim Beta(\alpha, \beta). \tag{11}$$

The prior distributions for μ et v are the same as before.

Posterior distribution

We have the following posterior distribution for ρ :

$$\rho : Beta(\alpha + \sum_{i=1}^{n-1} 1_{\{X_i \neq X_{i+1}\}}, \beta + \sum_{i=1}^{n-1} 1_{\{X_i = X_{i+1}\}}) \tag{12}$$

The posterior distributions for μ et v are the same as before.

Here to avoid label switching issue, we demand that the condition $\mu_1^l < \mu_2^l < \mu_3^l$ be satisfied.

3.3.2 Forward-Backward (states (X_t))

During all this subsection, we assume that θ the vector of the Markov chain parameters is fixed.

Also, $Y_a, Y_{a+1}, \ldots, Y_{b-1}, Y_b$ will be noted from now $Y_{a:b}$.

We seek here to determine the distribution of X given θ et $Y_{1:n}$. For pedagogical reasons, we first present the Forward-Backward algorithm assuming that the observations $Y_{1:n}$ are discrete (which is not the case in our situation). We will then generalize our formula to the continuous case.

Discrete observations

During all this part, $P(Y_k = y_k)$ will be noted $P(Y_k)$ to simplify notations.

We are looking for the distribution of X. Let us first remark that :

$$P(X_k = x \mid Y_{1:n}) = \frac{P(X_k = x, Y_{1:n})}{P(Y_{1:n})} = \frac{P(X_k = x, Y_{1:n} \mid X_k = x, Y_{1:k})\ P(X_k = x, Y_k)}{P(Y_{1:n})}$$
$$= P(Y_{k+1:n} \mid X_k, Y_{1:k})\ P(X_k = x, Y_{1:k})\ (\times \frac{1}{P(Y_{1:n})})$$
$$= \beta_k(x)\ \alpha_k(x)\ (\times \frac{1}{P(Y_{1:n})})$$

where $\alpha_k(x) = P(X_k = x, Y_{1:k})$ et $\beta_k(x) = P(Y_{k+1:n} \mid X_k = x)$ (independence of $(Y_k)_{k\geq 1}$ conditionally to the $(X_k)_{k\geq 1}$).

The $\alpha_k(x)$ can be dynamically computed ($1 \leq k \leq n$ and x din $\mathcal{X}$).

$$\alpha_{k+1}(x) = P(X_{k+1} = x, Y_{1:k+1})$$
$$= \sum_{x'} P(X_{k+1} = x, Y_{k+1}, Y_{1:k}, X_k = x')$$
$$= \sum_{x'} P(X_{k+1} = x, Y_{k+1} | Y_{1:k}, X_k = x')\ P(X_k = x', Y_{1:k})$$
$$= \sum_{x'} P(Y_{k+1} | X_{k+1} = x, Y_{1:k}, X_k = x')\ P(X_{k+1} = x | Y_{1:k}, X_k = x')\ P(X_k = x', Y_{1:k})$$
$$= \sum_{x'} P(Y_{k+1} | X_{k+1} = x)\ P(X_{k+1} = x | X_k = x')\ \alpha_k(x')$$
$$= \sum_{x'} g_{k+1}(y_{k+1})\ Q(x', x)\ \alpha_k(x')$$

The β_k are also dynamically computed and we have:

$$\beta_{k-1}(x) = P(Y_{k:n} | X_{k-1} = x)$$
$$= \sum_{x'} P(Y_k, Y_{k+1:n} | X_k = x', X_{k-1} = x)\ P(X_k = x' | X_{k-1} = x)$$
$$= \sum_{x'} P(Y_k, Y_{k+1:n} | X_k = x')\ Q(x, x')$$
$$= \sum_{x'} P(Y_k | X_k = x')\ P(Y_{k+1:n} | X_k = x')\ Q(x, x')$$
$$= \sum_{x'} g_k(y_k)\ \beta_k(x')\ Q(x, x')$$

.

However, to determine the distribution of X consists in determining the joint distribution of the $X_{1:n}$ et not only their marginal distributions. The previous computations are not enough.

We proceed as such :

$$P_y(X_1 = x_1, ..., X_n = x_n) \ = \ P_y(X_1 = x_1)\ P_y(X_2 = x_2 \mid X_1 = x_1) \ ...\ P_y(X_n = x_n \mid X_{n-1} = x_{n-1} \ ...\ X_1 = x_1) \tag{13}$$

with P_y probability P given $Y_1 = y_1, \ldots, Y_n = y_n$.

We then introduce

$$\begin{aligned} F_{k|n}(x', x) &= P(X_{k+1} = x | X_k = x', Y_{k+1:n}) \\ &= \frac{P(Y_{k+1:n}, X_{k+1} = x | X_k = x')}{P(Y_{k+1:n} | X_k = x')} \\ &= \frac{P(Y_{k+1:n} \mid X_{k+1} = x, X_k = x') P(X_{k+1} = x | X_k = x')}{\beta_k(x')} \\ &= \frac{P(Y_{k+1} \mid X_{k+1} = x) P(Y_{k+2:n} \mid X_{k+1} = x) P(X_{k+1} = x | X_k = x')}{\beta_k(x')} \\ &= \frac{g_{k+1}(x') \beta_{k+1}(x) Q(x', x)}{\beta_k(x')}. \end{aligned}$$

In the case where $\beta_k(x') = P(Y_{k+1:n} | X_k = x') = 0$ we define $F_{k|n}(x', \ .) = 0$. For $k = n$ we write $F_{k|n} = Q$.

We are now able to estimate the joint distribution of the $(X_k)_{1 \leq k \leq n}$ from $(\beta_k(x))_{t,x}$ (through the use of $F_{k|n}$). Indeed we have

$$P(X_1 = x_1, \ldots, X_n = x_n \mid Y_{1:n}) = \nu(x_1) \prod_{t=1}^{n-1} F_{k|n}(x_t, x_{t+1}) \tag{14}$$

Finally, we only need to take care of the normalization of the (α_k) and (β_k) for each k. Indeed the coefficients usually quickly converge exponentially to 0 ou $+\infty$. Numerical computations fast become impossible.

However, the normalization of the $(\beta_k)_k$ doesn't modify the value of the $F_{k|n}$, since the normalization factor is common to all the $(\beta_k(x))_{xin(X)}$ at k fixed and the sum of the $F_{k|n}(x', x)$ for $xin(X)$ is 1).

Continuous observations

In the continuous case, we define the α_t and β_t as such ($1 \leq t \leq n$ and $x \in \mathcal{X}$) :

$$\alpha_{\nu,t}(x) = \sum_{(x_1, \ldots, x_{t-1}) \in \mathcal{X}} \nu(x_1) g_1(x_1) \left(\prod_{s=1}^{t-2} Q(x_s, x_{s+1}) g_{s+1}(x_{s+1}) \right) Q(x_{s-1}, x) g_s(x) \tag{15}$$

with the convention that the product is evaluated to 1 for $t = 2$ (0 for $t \leq 1$), and

$$b_{t|n}(x) = \sum_{(x_{t+1}, \ldots, x_n) \in X^{n-t}} Q(x, x_{t+1}) g_{t+1}(x_{t+1}) \prod_{s=t+2}^{n} Q(x_{s-1}, x_s) g_s(y_s) \tag{16}$$

with $b_{n|n} = 1$.

The recursion formula are the same than for the discrete case (demonstration generalized to the continous case).

For the normalization, we define the sequence $(c_t)_{1\leq t\leq n}$ where :

$$c_1 = \sum_{x\in\mathcal{X}} \nu(x)g_1(x), \quad c_t = \sum_{x,x'\in\mathcal{X}} \bar{\alpha}_{t-1}(x)Q(x,x')g_t(x')$$

with $\bar{\alpha}_t = \frac{\alpha_t}{c_t}$ the normalized coefficient. We do not compute separately the α_t and the c_t given the interdependency.

However the c_t can be computed without knowing beforehand the values of β_t. We then compute $(\bar{\alpha})_{1\leq t\leq n}$ and $(c_t)_{1\leq t\leq n}$ to then normalize the β_t at each step with $\bar{\beta}_t = \frac{\beta_t}{c_t}$.

Finally, we compute the joint distribution of the $(X_t)_{1\leq t\leq n}$ the same way as in the discrete case (formula (13)).

4 Evaluate the quality of the estimation

Once we are able to estimate the parameters of the chain, it is important to be able to evaluate the quality of our estimates.

4.1 Evaluation criteria

A) States $(X_t)_{t\geq 1}$

In order to evaluate the prediction of the $(X_t)_{t\geq 1}$ we look at the proportion of well predicted states :

$$Precision(X_{1:n}) = \frac{\sum_{i=1}^{n} 1_{\hat{X}_i = X_i^0}}{n}$$

where $\hat{X}$ is the estimate of X and X^0 the value to estimate.

B) Chain parameters

The chain parameters which are estimated are mostly the transition matrix Q and the Gaussian distributions parameters μ et v (we do not really care about ν vector of initial probabilities). We chose two main criteria for these parameters in our study :

- relative precision
- l'appartenance de la vraie valeur à un intervalle de crédibilité
- the length of this interval.

To be able to compute the relative precision, it would be necessary to know the true values of the parameters we estimate. As such, we only measure it when using simulated data.

Let us keep the notations from Part 2, with $(\theta)_{1\leq l\leq L}$ the parameters vector, θ_l^0 the value to estimate and $\hat{\theta}_l$ its estimation. As a reminder, $\hat{\theta}_l$ is defined as the mean of the values of (θ_l) generated by the algorithm after the 1000^{th} itération.

We define the relative precision as such :

$$Relative\ precision\ (\theta_l)\ =\ \frac{|\hat{\theta}_l - \theta_l^0|}{\theta_l^0}. \tag{17}$$

The relative precision cannot take care of the case where $\theta_l^0 = 0$ and works badly when θ_l^0 is small, nethertheless it enables us to renormalize the parameters. Figure 5 shows the relative precisions of the parameters for 16 random simulations de différents paramètres ($\mu_2 = 0.1$).

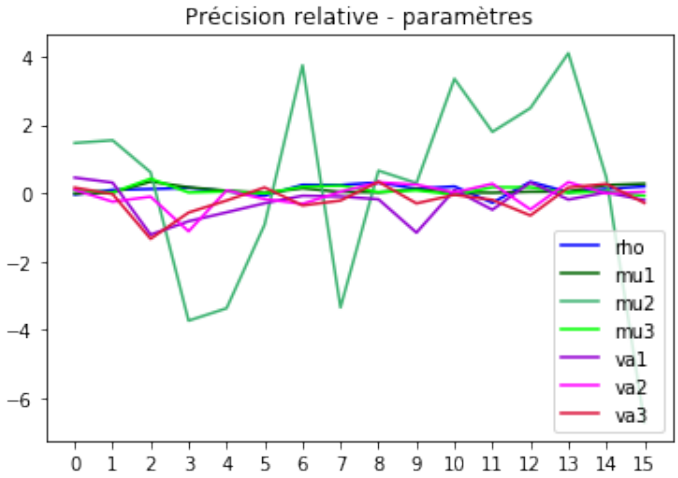


Figure 5: Relative precisions for 16 simulations

The confidence interval corresponds to the interval inside which we can find 95% of the generated values of θ_l (quantiles at 2,5% and at 97,5%). In case of a good estimatin, the distribution of the generated values is supposed to be tight around the true value : we expect the confidence interval to be small and to contain the value we had to predict. However, we didn't try to define what was considered to be acceptable for the length of a confidence interval.

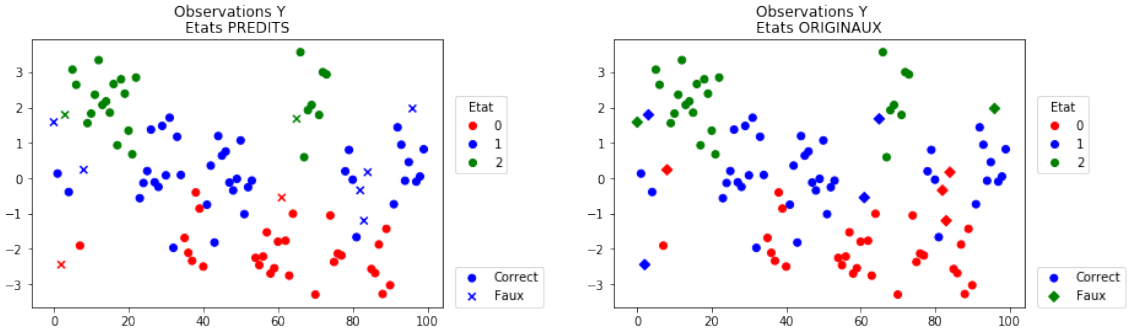
4.2 Simulations for 3 states

In order to estimate the performances of our algortihmn we simulated a Markov chain and its observations with the following parameters:

- number of states of the chain = 3
- number of observations $n = 100$
- $\rho = 0.2$
- $\mu = (-2, 0.1, 2)$
- $v = (1, 1, 1)$.

We simulated 3 times with these parameters and we obtained the following precisions : 0.90, 0.83 and 0.78. Regarding the other criterions, we had better limit ourselves to a single simulation, and we chose the one with the best precision (90%).

We begin by graphically displaying with colors the states corresponding to each observation (predicted states vs original states). The results is introduced in the Figure 6.



(a) Predicted states

(b) Original states

Figure 6: Observations of the color of the state

Rgearding the credibility intervals, the Figures 7 and 8 graphically display the distribution of parameters generated after the 1000^{th} iteration.

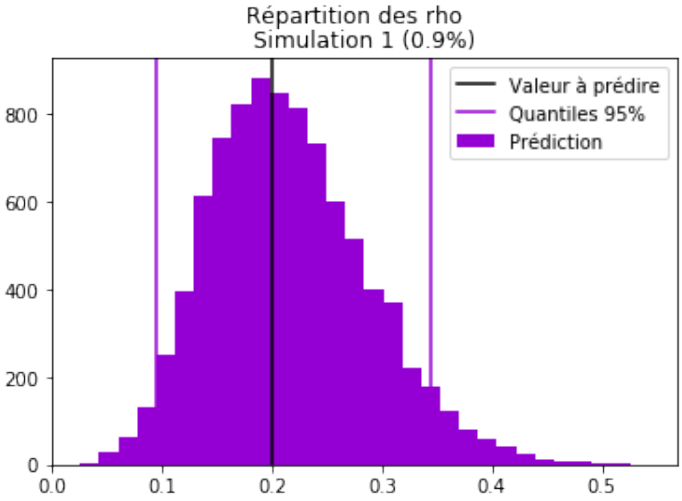
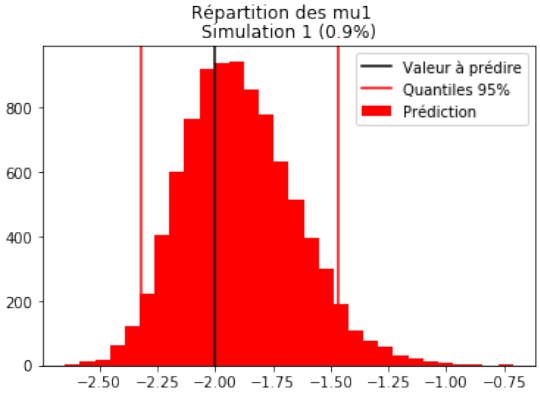
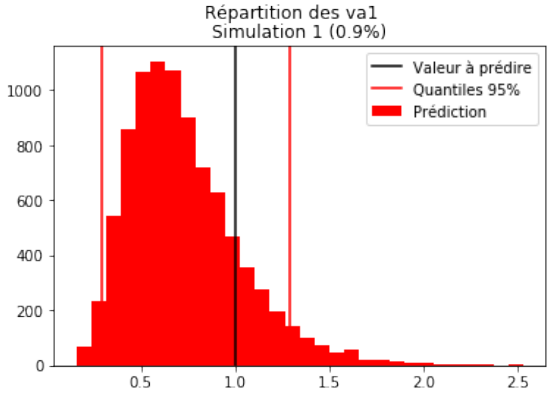


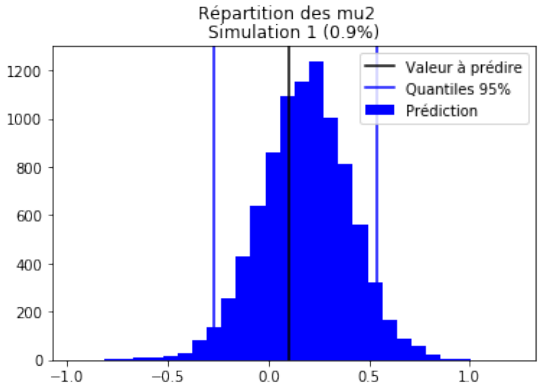
Figure 7: Distribution of ρ



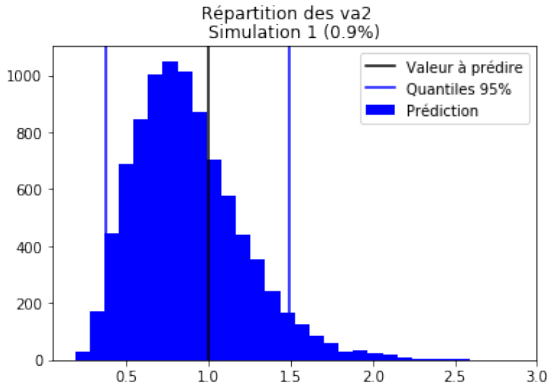
(a) Histogram of μ_1



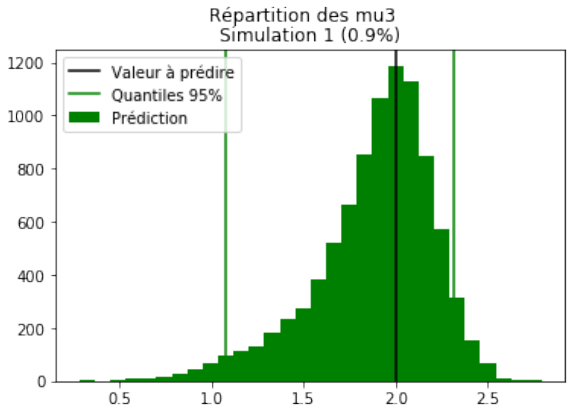
(b) Histogram of v_1



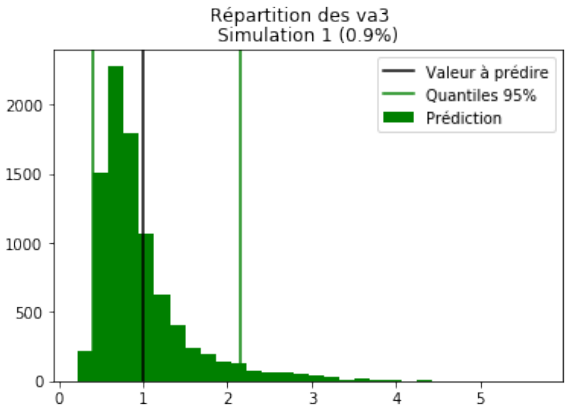
(c) Histogram of μ_2



(d) Histogram of v_2



(e) Histogram of μ_3



(f) Histogram of v_3

Figure 8: Distributions of μ and v

The credibility intervals - already displayed in the Figures 7 and 8 - are grouped in the Table 1.

For that simulation the credibility intervals contain all the real values. The width aren't that small but remain reasonable. Finally, we displayed the evolution of the generated values (Appendix II) in order to check in a graphical way the covnergence of the algorithm.

Parameter	ρ	μ_1	μ_2	μ_3	v_1	v_2	v_3
True value	0.2	-2	0.1	2	1	1	1
Credibility interval	[0.09,0.34]	[-2.32,-1.47]	[-0.27,0.54]	[1.08,2.32]	[0.3,1.29]	[0.37,1.5]	[0.4,2.15]

Table 1: Credibility interval at 95%

The values we generate aren't supposed to converge towards a final value as they are randomly generated at each iteration (given a distribution whom we estimate the parameters). Nonetheless we expect them to be distributed around a stable mean value without being too far from it (and this is the case in that simulation).

Rgearding the computation time, the algorithm took roughly half an hour for a simulation with $n = 100$ observations and 10000 iterations. We did many other simulations with different values for ρ, μ and v. The results for some of them are on Table 2 (and their relative precisions are in Figure 5 by the way).

All the variances are equal to 1 here, and all the expected values can be expressed as follows : $(-\mu_0, 0.1, \mu_0)$.

ρ	0.4	0.4	0.4	0.4	0.3	0.3	0.3	0.2	0.2	0.2	0.2	0.1	0.1	0.1	0.1
μ_0	5	4	3	2	5	3	2	5	4	3	2	5	4	3	2
Precision	0.99	0.98	0.85	0.83	0.98	0.92	0.86	1.0	0.99	0.98	0.84	1.0	0.99	0.98	0.94

Table 2: States précisions for 16 simulations

5 Running our algorithm with the data of the paper

5.1 Results

After having evaluated the performances of our algorithm, we ran it with the data introduced in Part 1. We have selected a portion of the Chromosome 5 for that - the one chose by Yau et al. in their paper [1] - and the deletion zone is highlighted in red on the Figure 9.

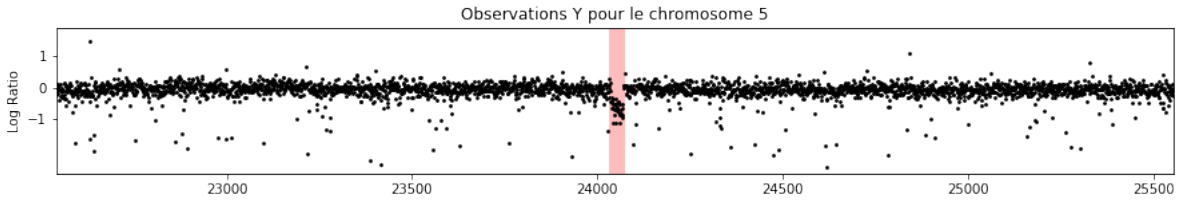
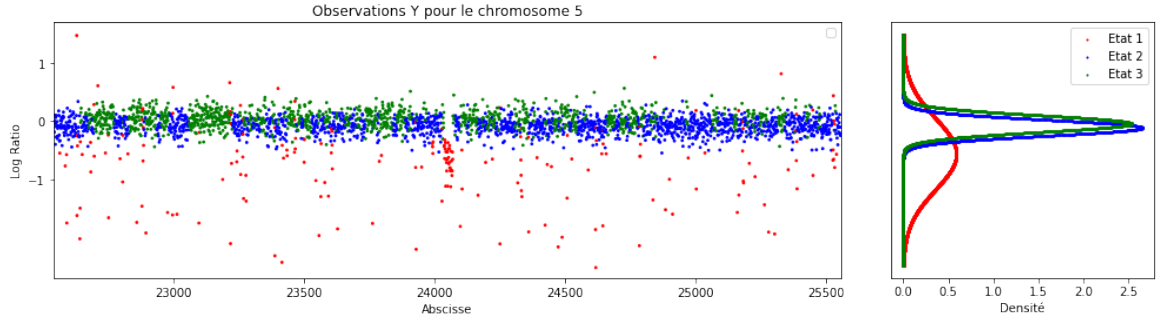


Figure 9: Observations for chromosome 5

The algorithm predicts the states introduced on Figure 10a. We added to this graph the Gaussian distributions corresponding to each state (Figure 10b).



(a) Observations colored according to the state

(b) Distributions for each state

Figure 10: States predicted with a Gibbs sampling

The graph of the Figure 11 is basically the superimposition of the means μ_1, μ_2 and μ_3 and of the values of the observations we get. One may notice that the values of μ_2 and μ_3 are very close.

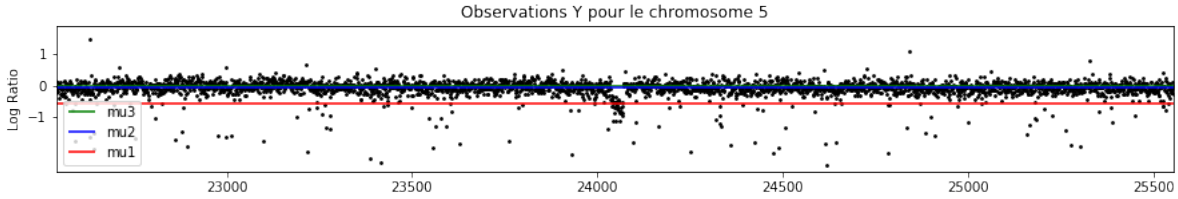


Figure 11: Observations et espérances des gaussiennes par état

The graph of the Figure 12 consists in the deletion probabilities for each site. As in our model a deletion is modeled by the state 1, we get the deletion probability of any position t by counting the proportion of 1's generated by the algorithm for X_t.

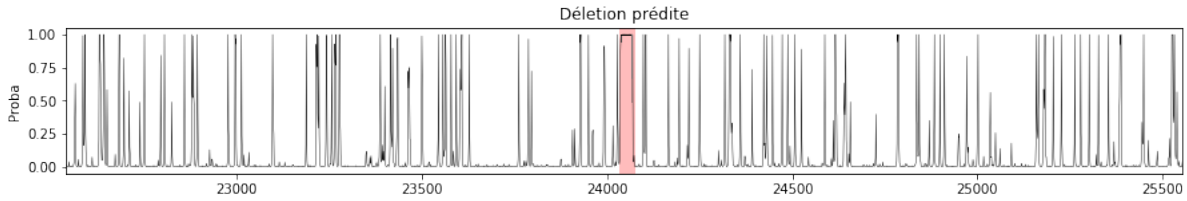


Figure 12: Deletion probabilities for each site

Figure 13 compares the deletion probabilities we get to those of the paper (just Gaussian distribtions).

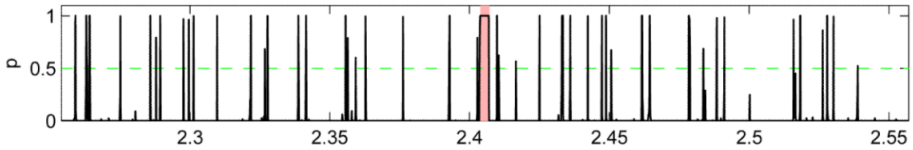


Figure 13: Deletion probabilities for each site (Article)

One may notice that our results have a high number of false positives, more than in the results of the paper. It may be due to the estimation of ρ : Yau et al. don't estimate that value and set it to 0.05, while we find 0.13 when we estimate it. Indeed, a higher value for ρ leads to a number of states attributed to 1 which is higher.

Nonetheless, even if Yau et al. declare getting their results in less than one hour, we didn't manage to be that fast. Our algorithm required a computation time of around half an hour for $n = 100$ observations, explaining why the estimations of hidden states with $n = 30\ 000$ observations couldn't be done in few hours (the temporal complexity is linear in n). For this reason, we couldn't run the algorithm several times with the data, and see what would have been the results with ρ set to 0.05 for instance.

We thus didn't get enough time to run the algorithm with other DNA sequences, despite the fact that we implemented a general algorithm with three states (*deletion, insertion, modification*). We could have indeed limited ourselves to two states (*deletion, insertion*) as it's done in the paper, instead of considering the states 2 and 3 separately. The authors of the paper suppose that the input sequences don't contain simultaneously deletions **and** insertions, and that we can thus just consider two states (that are deletion / *insertion* and *no modification*).

That may explain why we have very close means for the states 2 and 3 ($\mu_2 = -0.07$ and $\mu_3 = 0.006$ with estimated variances of 0.02).

5.2 Assessing performance (simulated data with the estimated parameters)

Once we had estimated the parameters of the model with the sequence from chromosome 5, we decided to assess the performance of our algorithm in that precise framework: we simulated a HMM with $n = 3000$ observations as well as a Markov chain whom parameters were those previously estimated.

Running our algorithm with the simulated data gave us a precision of 74% for the predicted states. The distributions of the parameters as well as their credibility intervals are introduced in the Appendix III.

6 Conclusion

We managed to implement Gibbs sampling in the case of Gaussian distributions and to get results similar to those of the paper of Yau et al. (2006) [1]. The algorithm perfoms quite well despite a quite long running time (accelerating the run using C code for instance or librarires such as *numba* might have helped us).

Nevertheless, given the results of the paper it seems necessary to consider more complex distributions such as mixture models or non-parametric distributions. Gaussian distributions are actually very sensitive to asymmetry, fat tails (for the extreme values) and outliers. For instance, we saw with the chromosome 5 that outliers due to the measure noise were pretty often associated to the state 1 of higher variance.

7 Appendix

I) An example of calculus in the situation with two possible states for X (3.3)

Let's recall that $g_t(i) = \frac{1}{\sqrt{2\pi v_i}} e^{-\frac{(y_t-\mu_i)^2}{2v_i}}$ when $X_t = i$.

With the formula (3) we get that the density of the posterior distribution of θ verifies:

$$f_{\theta|X=x,Y=y}(\theta) \propto \pi(\theta)\ p_\theta(x,y) \tag{18}$$

Hence

$f_{\theta|X,Y}(q_{11}, q_{22}, \mu_1, \mu_2, v_1, v_2) \propto$

$$q_{11}^{\alpha-1}(1-q_{11})^{\beta-1} q_{22}^{\alpha-1}(1-q_{22})^{\beta-1} e^{-\frac{(\mu_1-m)^2}{2w}} e^{-\frac{(\mu_2-m)^2}{2w}} v_1^{-c-1} e^{-\frac{d}{v_1}} v_2^{-c-1} e^{-\frac{d}{v_2}}$$
$$\times\ \nu(x_1) g_1(x_t) \prod_{t=1}^{n-1} Q(x_t, x_{t+1}) g_t(x_t)$$

Posterior distribution of q_{11}:

$$f_{q_{11}|q_{22},\mu,v,X,Y}(q_{11}) = \int f_{\theta|X,Y}(q_{11}, q_{22}, \mu_1, \mu_2, v_1, v_2)\ dq_{22}\ d\mu_1\ d\mu_2\ dv_1\ dv_2$$
$$\propto q_{11}^{\alpha-1}(1-q_{11})^{\beta-1} q_{11}^{\sum 1_{\{X_i=1, X_{i+1}=1\}}} (1-q_{11})^{\sum 1_{\{X_i=1, X_{i+1}=2\}}}$$

Given the uniqueness of the posterior distribution, it is thus a Beta distribution $Beta(\alpha + \sum 1_{\{X_i=1,X_{i+1}=1\}}, \beta + \sum 1_{\{X_i=1,X_{i+1}=2\}})$

Posterior distribution of μ_1:

$$f_{\mu_1|Q,\mu_2,v,X,Y}(\mu_1) = \int f_{\theta|X,Y}(q_{11}, q_{22}, \mu_1, \mu_2, v_1, v_2)\ dq_{11}\ dq_{22}\ d\mu_2\ dv_1\ dv_2$$
$$\propto e^{-\frac{(\mu_1-m)^2}{2w}} e^{-\frac{1}{2}\sum_{i=1}^n \frac{(Y_i-\mu_1)^2}{v_1} 1_{X_i=1}}$$

For the same reasons as above (uniqueness of the distribution), the distribution we identify is a Gaussian distribution $\mathcal{N}(\frac{mv_1+S_1w}{v_1+N_1w}, \frac{wv_1}{v_1+N_1w})$.

Posterior distribution of v_1:

$$f_{v_1|Q,\mu,v_2,X,Y}(v_1) = \int f_{\theta|X,Y}(q_{11}, q_{22}, \mu_1, \mu_2, v_1, v_2)\ dq_{11}\ dq_{22}\ d\mu_1\ d\mu_2\ dv_2$$
$$\propto v_1^{-c-1} e^{-\frac{d}{v_1}} e^{-\frac{1}{2}\sum_{i=1}^n \frac{(Y_i-\mu_1)^2}{v_1} 1_{X_i=1}} \left(\frac{1}{\sqrt{(v_1)}}\right)^{N_1}$$

The posterior distribution is thus an Inverse-Gamma $IG(c + \frac{N_1}{2}, d + \frac{1}{2}\sum_{i=1}^{n}(Y_i - \mu_1)^2 1_{X_i=1})$

We get the posterior distributions of $\{q_{22}, \mu_2, v_2\}$ in like manner.

II) Evolution of the generated values for the different parameters (4.2)

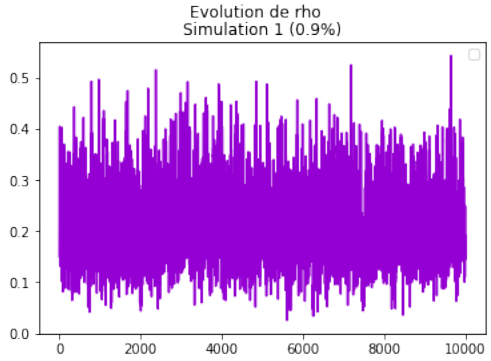
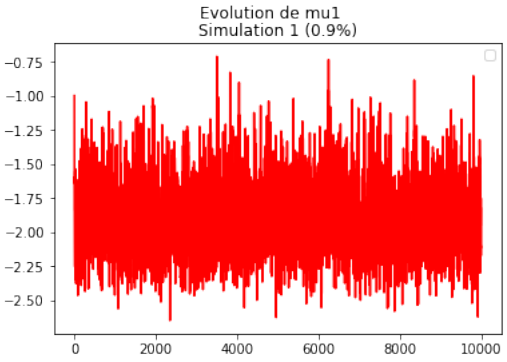
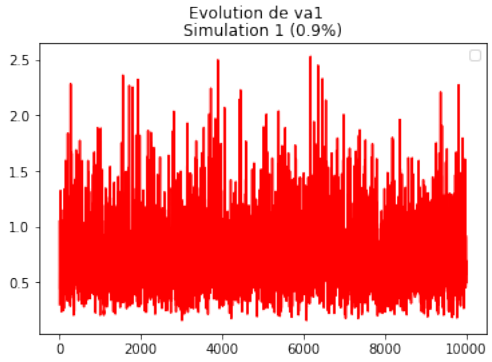


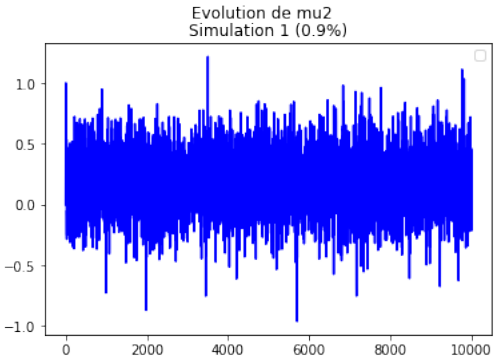
Figure 14: Evolution of ρ



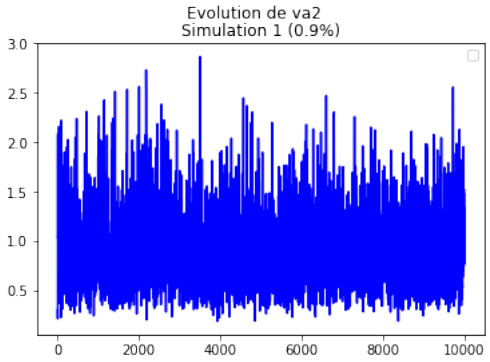
(a) Histogram of μ_1



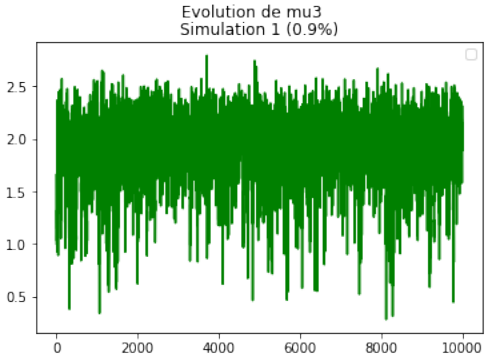
(b) Histogram of v_1



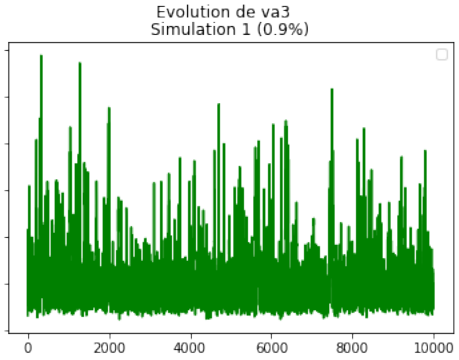
(c) Histogram of μ_2



(d) Histogram of v_2



(e) Histogram of μ_3



(f) Histogram of v_3

Figure 15: Evolution of μ and v

III) Distribution of the generated values for data simulated with the estimated parameters (5.2)

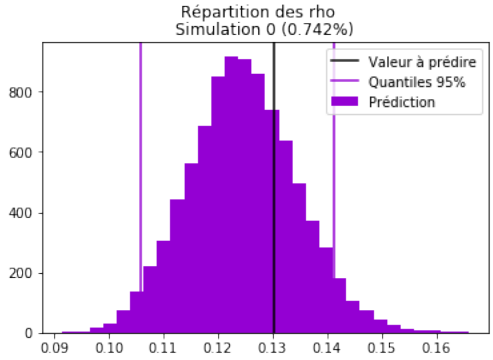
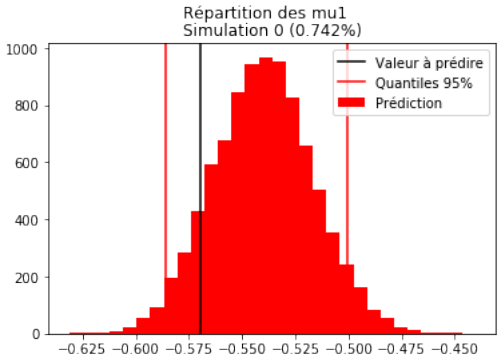
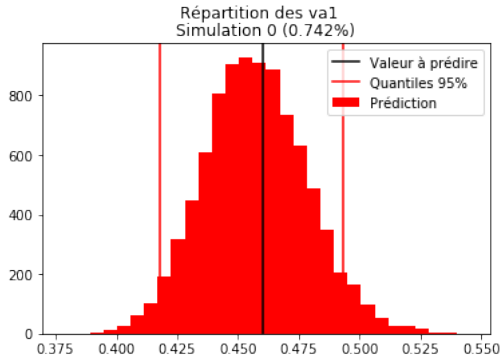


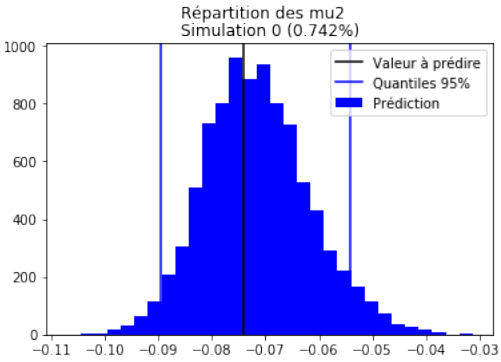
Figure 16: Distribution of ρ



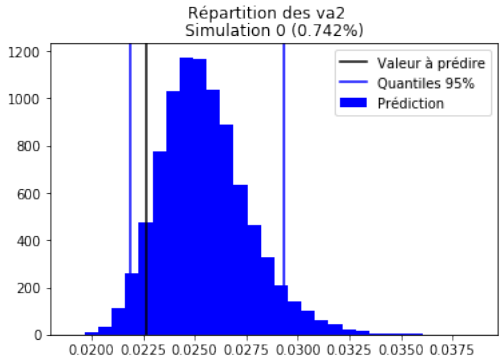
(a) Histogram of μ_1



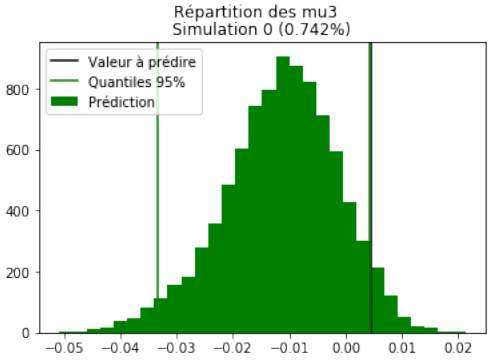
(b) Histogram of v_1



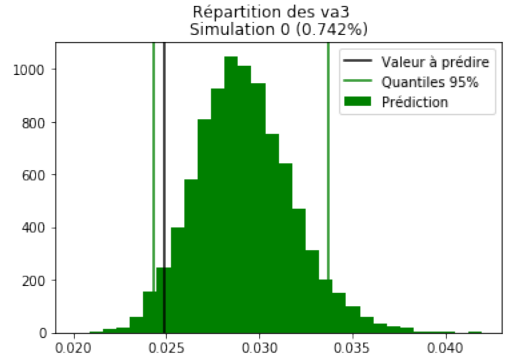
(c) Histogram of μ_2



(d) Histogram of v_2



(e) Histogram of μ_3



(f) Histogram of v_3

Figure 17: Distribution of μ and v

References

[1] C.Yau,O.Papaspiliopoulos, G.O.Roberts and C.Holmes *Bayesian Nonparametric Hidden Markov Models with application to the analysis of copy-number-variation in mammalian genomes* ,Journal of the Royal Statistical Society. Series B, Statistical methodology vol. 73,1 (2011): 37-57

[2] Lakshmi, B et al. *Mouse genomic representational oligonucleotide microarray analysis: detection of copy number variations in normal and tumor specimens*, Proceedings of the National Academy of Sciences of the United States of America vol. 103,30 (2006): 11234-9

[3] Olivier Cappé, Eric Moulines, Tobias Rydén *Inference in Hidden Markov Models* , Springer, 2005

[4] Tom Kennedy *Chapter 8: Markov chain Monte Carlo*, 2016, https://www.math.arizona.edu/ tgk/mc/index.html