Coupled Markov chains with applications to Approximate Bayesian Computation for model based clustering

E. Bertoni, M. Caldarini, F. Di Filippo, G. Gabrielli, E. Musiari

Politecnico di Milano

February 14, 2022

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1. Introduction

The aim of our project is to deal with two problems that can affect the same computation. The first one is to speed up Markov chain Monte Carlo (MCMC) methods using parallelization, while the second one is to manage the case in which the likelihood is computationally intractable. To solve the first problem, we used the Unbiased Markov chain Monte Carlo methods with couplings, instead for the second problem we used the Approximate Bayesian Computation.

2. Methods

2.1. Unbiased Markov chain Monte Carlo methods with couplings

Markov chain Monte Carlo (MCMC) methods provide consistent approximations of high dimensional integrals, namely as the number of iterations goes to infinity. However, these estimators can be potentially biased for any fixed number of iterations, hence the aim is to propose a general construction to produce unbiased estimators of integrals with respect to a target probability distribution.

Glynn and Rhee **[glynn2014exact]** illustrated a construction on Markov chains represented by iterated random functions; in their approach only two chains must be coupled for the proposed estimator to be unbiased, without further assumptions on the state space or target distribution. Indeed, we need an unbiased estimator in order to make the parallelization possible, that will lead to a faster computation of the MCMC.

Hence, the goal is to estimate

$$\mathbb{E}_\pi[h(X)] = \int h(x)\pi(dx).$$

The estimator is based on a coupled pair of Markov chains, $(X_t)_{t\geq 0}$ and $(Y_t)_{t\geq}$, which marginally start from π_0 and evolve accordingly to P.

Some assumptions must be considered:

1. as $t \to \infty$,
$$\mathbb{E}[h(X_t)] \to \mathbb{E}_\pi[h(X)];$$
and there exists $\eta > 0$ and $D < \infty$ such that $\mathbb{E}[|h(X_t)|^{2+\eta}] \leq D$ for all $t \geq 0$;
2. the chains are such that the meeting time
$$\tau = \inf\{t \geq 1 : X_t = Y_{t-1}\}$$
satisfies $\mathbb{P}(\tau > t) \leq C\delta^t$ for all $t \geq 0$, for some constants $C < \infty$ and $\delta \in (0, 1)$;
3. the chains stay together after meeting:
$$X_t = Y_{t-1} \quad \forall t \geq \tau.$$

Thanks to the previous assumptions it can be proved that:

$$\mathbb{E}_\pi[h(X)] = \mathbb{E}[h(X_k) + \sum_{t=k+1}^{\tau-1} \{h(X_t) - h(Y_{t-1})\}];$$

and the Rhee–Glynn estimator can be defined as:

$$H_k(X, Y) = h(X_k) + \sum_{t=k+1}^{\tau-1} \{h(X_t) - h(Y_{t-1})\}$$

which is unbiased by construction.

Time-averaged estimator can be considered as a special case of Rhee–Glynn estimator, in which $H_k(X, Y)$ can be computed for several values of k from the same realization of the coupled chains, with unbiased average of them. It can used to allow computation, evaluated as:

$$H_{k:m}(X, Y) = MCMC_{k:m} + BC_{k:m}$$

where:

$$MCMC_{k:m} = \frac{1}{m-k+1} \sum_{l=k}^{m} h(X_l)$$

is the standard MCMC average;

$$BC_{k:m} = \sum_{l=k+1}^{\tau-1} \min(1, \frac{l-k}{m-k+1})\{h(X_l) - h(Y_{l-1})\}$$

is the bias correction;

m is the number of iterations, smaller than the case of a biased classical MCMC method; $k-1$ is the burn-in number; τ is a random variable for the meeting time.

The algorithm of the time-average estimator can be written as follows:

1. draw X_0 and Y_0 from an initial distribution π_0 and draw $X_1 \sim P(X_0, \cdot)$;
2. set $t = 1$: while $t < \max\{m, \tau\}$ and:
   a draw $(X_{t+1}, Y_t) \sim \bar{P}\{(X_t, Y_{t-1}), \cdot\}$;

b set $t \leftarrow t+1$;

3. compute the time-averaged estimator:

$$H_{k:m}(X,Y) = \frac{1}{m-k+1}\sum_{l=k}^{m} h(X_l)$$

$$+ \sum_{l=k+1}^{\tau-1} \min(1, \frac{l-k}{m-k+1})\{h(X_l) - h(Y_{l-1})\}.$$

where $\bar{P}$ must be evaluated before.

Metropolis–Hasting algorithm allows the calculation of the coupled kernel $\bar{P}\{(X_t, Y_{t-1}), \cdot\}$:

1. sample $(X^\star, Y^\star)|(X_t, Y_{t-1})$ from a maximal coupling of $q(X_t, \cdot)$ and $q(Y_{t-1}, \cdot)$;
2. sample $U \sim \mathcal{U}([0,1])$;
3. if
$$U \leq \min\left\{1, \frac{\pi(X^\star)q(X^\star, X_t)}{\pi(X_t)q(X_t, X^\star)}\right\}$$
then $X_{t+1} = X^\star$; otherwise $X_t = X_{t-1}$;
4. if
$$U \leq \min\left\{1, \frac{\pi(Y^\star)q(Y^\star, Y_t)}{\pi(Y_t)q(Y_t, Y^\star)}\right\}$$
then $Y_{t+1} = Y^\star$; otherwise $Y_t = Y_{t-1}$.

and it can be noticed that the acceptance conditions are based on the same samples from the uniform.

A maximal coupling between two distributions p ad q on a space $\mathcal{X}$ is a distribution of a pair of random variables(X, Y) that maximizes $\mathbb{P}(X = Y)$, subject to the marginal constraints $X \sim p$ and $Y \sim q$.

We can proceed setting $p = \mathcal{N}(X_{t-1}, 1)$ and $q = \mathcal{N}(Y_{t-1}, 1)$, then:

1. sample $X_t \sim p$;
2. sample $W|X_t \sim \mathcal{U}\{[0, p(X_t)]\}$;
3. if $W \leq q(X_t)$ then output (X_t, X_t), otherwise:
   (a) sample $Y_t \sim q$;
   (b) sample $W^\star|Y_t \sim \mathcal{U}\{[0, q(Y_t)]\}$ until $W^\star > p(Y_t)$ and output (X_t, Y_t).

and hence the output follows a maximal coupling of p and q.

Moreover, the absence of bias allows the application on time budget-constrained parallel simulation: independent unbiased estimators with finite variance can be generated on separate machines and combined into consistent and asymptotically normal estimators, leading to better results with respect to the non-parallelized version.

2.1.1 Parallelized implementation results

The unbiased Markov chain Monte Carlo method with couplings can be implemented allowing parallelization. Indeed, this method consist in generating many couples of chains, any couple can be given to a single core to be processed, instead of generating each couple sequentially.

To understand how to exploit the unbiasness of time-averaged estimator $H_{k:n}(X, Y)$ and how to find a criteria for settings parameters like number of iterations or number of chains, here we report some results.

Considering the assumptions of the method, for all $k \geq 0$ and $m \geq k$, the estimator $H_{k:m}(X, Y)$ has expectation $\mathbb{E}_\pi[h(X)]$, a finite variance and a finite computation time. This means that the mean of many $H_{k:m}(X, Y)$ converges to $\mathbb{E}_\pi[h(X)]$ as their number increases, thus we can sample many independent copies of $H_{k:n}(X, Y)$ in parallel and average the result to get a good approximation of $\mathbb{E}_\pi[h(X)]$.

In order to have a good estimation we have to consider also the variance of the estimation. Recall that we wrote the time-averaged estimator as the sum of the Markov chain Monte Carlo average and a bias correction:

$$H_{k:n}(X,Y) = MCMC_{k:m} + BC_{k:m},$$

thanks to that form we can write the estimator's variance as:

$$\mathbb{V}[H_{k:m}(X,Y)] = \mathbb{E}[\{MCMC_{k_m} - \mathbb{E}_\pi[h(X)]\}^2] + 2\mathbb{E}[\{MCMC_{k:m} - \mathbb{E}[h(X)]\}BC_{k:m}] + \mathbb{E}[BC_{k:m}^2]$$

We wrap the mean-square error:

$$MSE_{k:m} = \mathbb{E}[\{MCMC_{k_m} - \mathbb{E}_\pi[h(X)]\}^2],$$

and, thanks to other results, we obtain the following bound for variance:

$$\mathbb{V}[H_{k:m}(X,Y)] \leq MSE_{k:m} + 2\sqrt{MSE_{k:m}C(m,k)} + C(m,k)$$

Where $C(m,k)$ is a function of m and k of the shape $\frac{A\,B^k}{m-k+1}$ where A and B are constants.

Therefore the variance is bounded by the mean-square error of the MCMC estimator plus additive therms that vanish polynomially in $m-k$.

Assuming that $m > \tau$ with large probability, it is proved that we can retrieve an asymptotic efficiency that is comparable with the underlying MCMC estimators with appropriate choices of k and m that depend on the distribution of the meeting time τ.

In other words, the bias of the MCMC algorithm can be removed at the cost of an increased variance, which can in turn be reduced by choosing sufficiently large values of k and m. This results in a trade-off with the desired level of parallelism: one might prefer to keep k and m small, yielding a suboptimal efficiency for $H_{k:m}(X,Y)$, but enabling more independent copies to be generated in a given computing time.

2.2. Approximate Bayesian Computation

Likelihood-free methods are used to solve the problem that concerns the intractable evaluation of the likelihood function. A way to solve this issue is to use Approximate Bayesian Computation method, in which samples are produced form an approximation of the posterios distribution. Doing so the likelihood doesn't need to be evaluated in teh algorithm.

Our focus was on ABC-Markov chain Monte Carlo samplers using Metropolis–Hastings algorithm.

In the well known Metropolis–Hastings algorithm given the current chain state $\theta^{(i)}$, the next value is obtained by sampling a candidate value θ' from a proposal distribution $\theta' \sim g(\theta^{(i-1)}, \theta)$, which is then accepted with probability

$$\alpha(\theta^{(i-1)}, \theta') = \min\{1, \frac{\pi(\theta'|y_obs)g(\theta', \theta^{(i-1)})}{\pi(\theta^{(i-1)}|y_obs)g(\theta^{(i-1)}, \theta')}\}$$

Whereas the likelihood can't be evaluated, in the standard ABC computation the posterior is substituted with the approximated version

$$\pi_{ABC}(\theta|s_obs) \propto \int K_h(\parallel s' - s_{obs} \parallel) Lik(\theta|s_obs)\pi(\theta)ds$$

where $K_h(u) = \frac{1}{h}K\left(\frac{u}{h}\right)$ is the standard smoothing Kernel function of parameter $h > 0$,$Lik(\theta|s_obs)$ is the intractable likelihood function and $s = S(y)$ is the low-dimensional vector of summary statistics

Therefore we can rewrite the acceptance probability, after some semplification, as:

$$\alpha(\theta^{(i-1)}, \theta') = \min\{1, \frac{\pi_{ABC}(\theta'|s_{obs})g(\theta', \theta^{(i-1)})}{\pi_{ABC}(\theta^{(i-1)}|s_{obs})g(\theta^{(i-1)}, \theta')}\}$$

$$= \min\{1, \frac{\pi_{ABC}(K_h(\parallel s' - s_{obs} \parallel)\pi(\theta')g(\theta', \theta^{(i-1)})}{K_h(\parallel s^{(i-1)} - s_{obs} \parallel)\pi(\theta^{(i-1)})g(\theta^{(i-1)}, \theta')}\}$$

sucht that there's no direct evaluation of the likelihood function.

A key task in this computation is to set efficiently both the scaling parameter h and the summary statistics: the former parameter must be small to obtain the resulting estimate of the posterior closer to the true posterior. While the latter must be large enough to contain informations and low enough to avoid curse of dimensionality. The best choice is sufficient statistics. As sufficient statistics can be much lower dimensional than the full dataset, it is clear that greater approximation accuracy can be achieved for the same computational overheads when using low dimensional statistics. Then the comparison $\| y - y_{obs} \|$ might be replaced by $\| s - s_{obs} \|$ without too much loss of information, but with the advantage that the dimension of S(y) is now much lower.

The algorithm can be written as:

Input:

- a target posterior density $\pi(\theta|y_{obs}) \propto p(y_{obs}|\theta)\pi(\theta)$, consisting of a prior distribution $\pi(\theta)$ and a procedure of generating data under the model $p(y_{obs}|\theta)$;
- a proposal density $g(\theta'|\theta)$
- an integer $N > 0$;
- a kernel function $K_h(u)$ and a scale parameter $h > 0$;
- a low dimensional vector of summary statistics $s = S(y)$.

Initialize: Repeat until

$$K_h(\| s^{(0)} - s_{obs} \|) > 0$$

1. Choose an initial parameter vector $\theta_{(0)} \sim \pi(\theta)$;
2. generate $y^{(0)} \sim p(y|\theta^{(0)})$ from the model and compute summary statistics $s^{(0)} = S(y^{(0)})$;

Sampling for $i = 1, ..., N$:

1. generate candidate vector $\theta' \sim g(\theta^{(i-1)}, \theta)$ from the proposal density g;
2. generate $y' \sim p(y|\theta')$ from the model and compute summary statistics $s' = S(y')$;
3. with probability

$$\min\{1, \frac{K_h(\| s' - s_{obs} \|)\pi(\theta')g(\theta', \theta^{(i-1)})}{K_h(\| s^{(i-1)} - s_{obs} \|)\pi(\theta^{(i-1)})g(\theta^{(i-1)}, \theta')}\}$$

set $(\theta^{(i)}, s^{(i)}) = (\theta', s')$. Otherwise set $(\theta^{(i)}, s^{(i)}) = (\theta^{(i-1)}, s^{(i-1)})$.

Output:

a set of correlated parameter vectors $\theta^{(1)}, ..., \theta^{(N)}$ from a Markov chain with stationary distribution $\pi_{ABC}(\theta|S_{obs})$.

3. Implementation

3.1. Approximate Bayesian Computation

3.1.1 Univariate case

The univariate ABC algorithm begins with generating 1000 samples from a Normal Distribution:

$$y_{obs} \sim \mathcal{N}(\mu_{obs}, \sigma^2_{obs})$$

and the chosen model is

$$y_i|\mu \overset{iid}{\sim} \mathcal{N}(\mu, \sigma^2_{obs}) \quad \text{for i=1,...,n}$$
$$\mu \sim \mathcal{N}(\mu_0, \sigma^2_0)$$

with σ_{obs} known and μ_0, σ_0 fixed.

The Gaussian Kernel function is

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$$

The distance is the L^2-norm of the difference between s_0 and s_{obs} and the summary statistics chosen are the sample mean and the vector of 9 quantiles. Both of them are sufficient statistics.

First case: sample mean

Input parameters: $\mu_0 = 8,\ \sigma_0^2 = 4,\ \mu_{obs} = 10,\ \sigma_{obs}^2 = 3,\ h = 0.2$

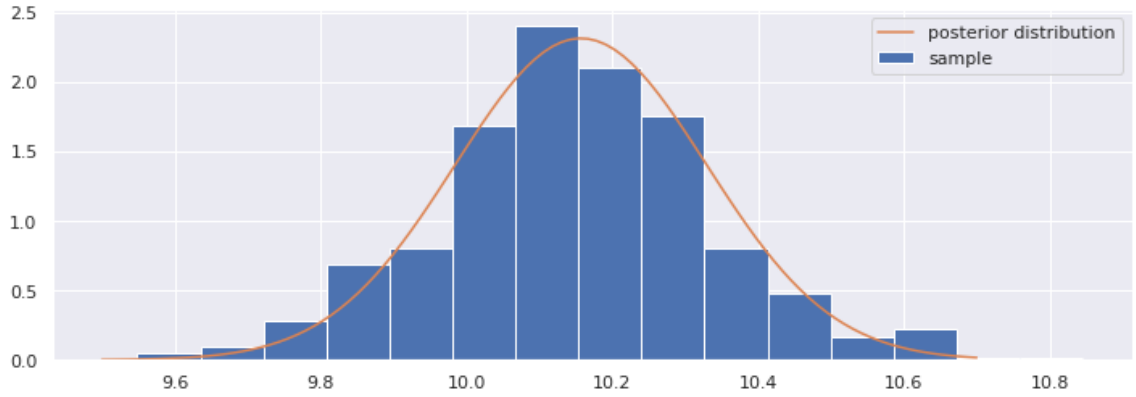

Figure 1

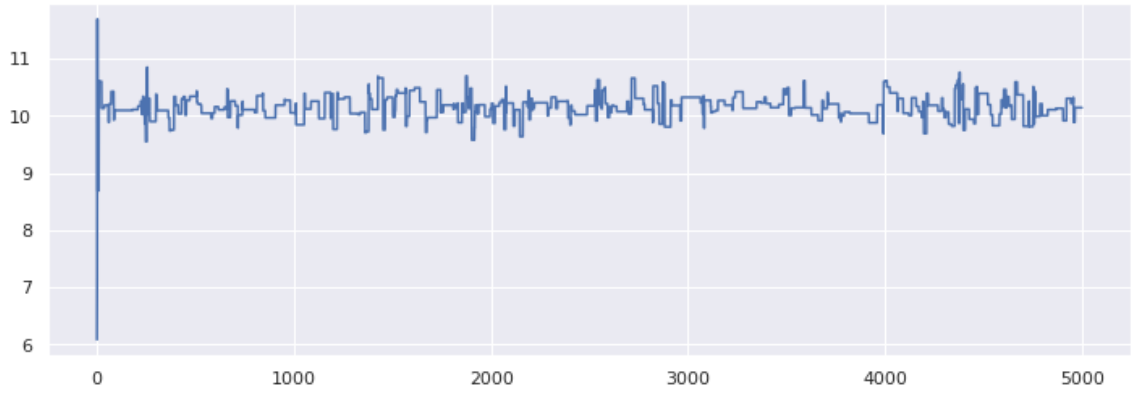

Figure 2

The posterior distribution has $\mu = 10.151$ and $\sigma^2 = 0.0298$

As mentioned in the chapter 2, the choice of parameter h is important.

Input parameters: $\mu_0 = 8,\ \sigma_0^2 = 4,\ \mu_{obs} = 10,\ \sigma_{obs}^2 = 3,\ h = 1.8$

As seen in 4, a larger h led to a poor approximation due to the acceptance of generated data that are far from the observed data.

Second case: vector of 9 quantiles

Input parameters: $\mu_0 = 8,\ \sigma_0^2 = 4,\ \mu_{obs} = 10,\ \sigma_{obs}^2 = 3,\ h = 0.5$

The posterior distribution has $\mu = 10.2527$ and $\sigma^2 = 0.0298$

Input parameters: $\mu_0 = 8,\ \sigma_0^2 = 4,\ \mu_{obs} = 10,\ \sigma_{obs}^2 = 3,\ h = 1.8$

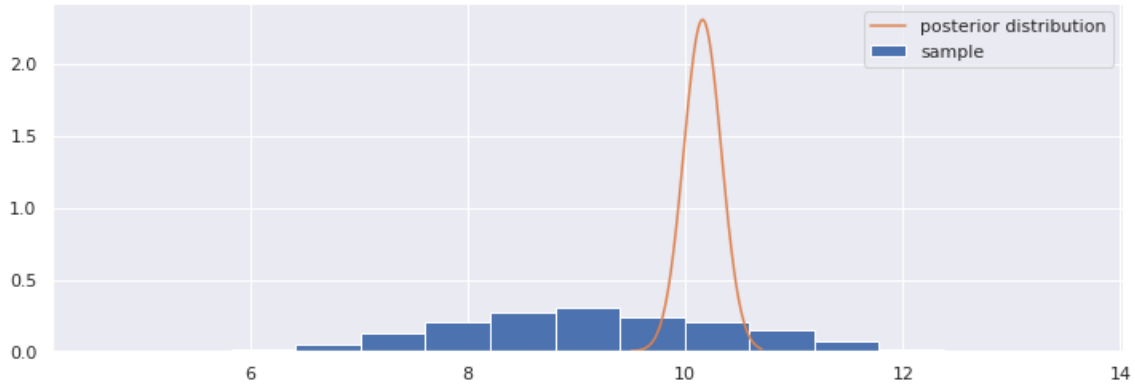

Figure 3

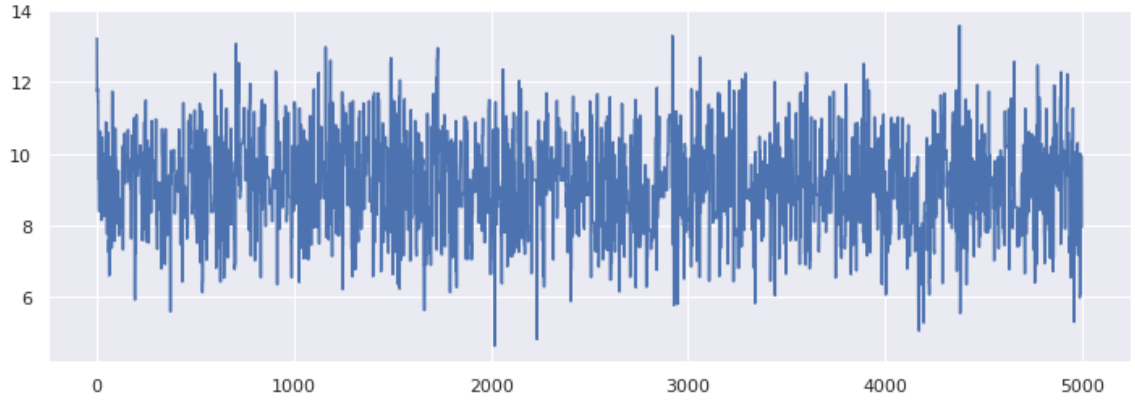

Figure 4

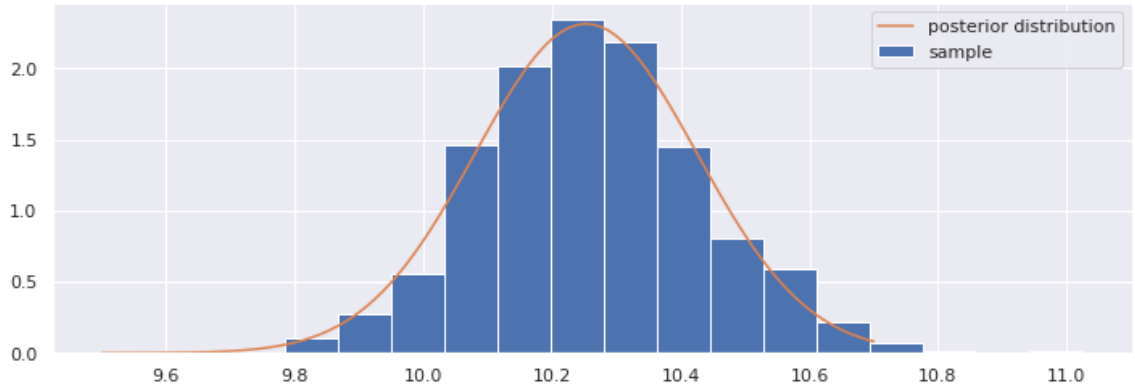

Figure 5

3.2. Multivariate case

The multivariate ABC method is an extension of the univariate one. Samples has been generated from a Multivariate Normal Distribution.

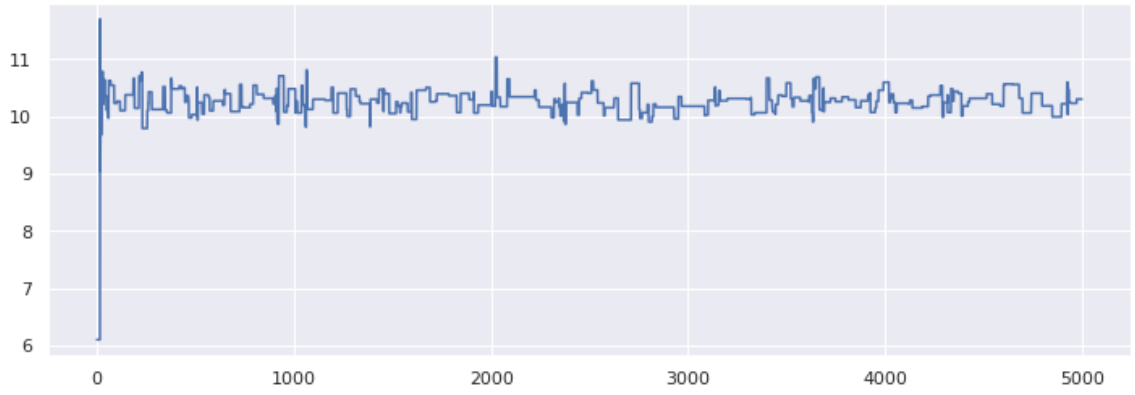

Figure 6

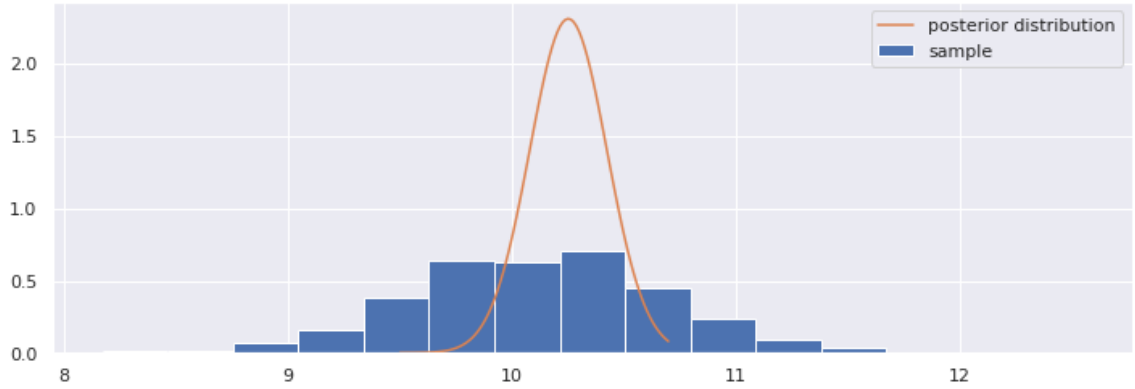

Figure 7

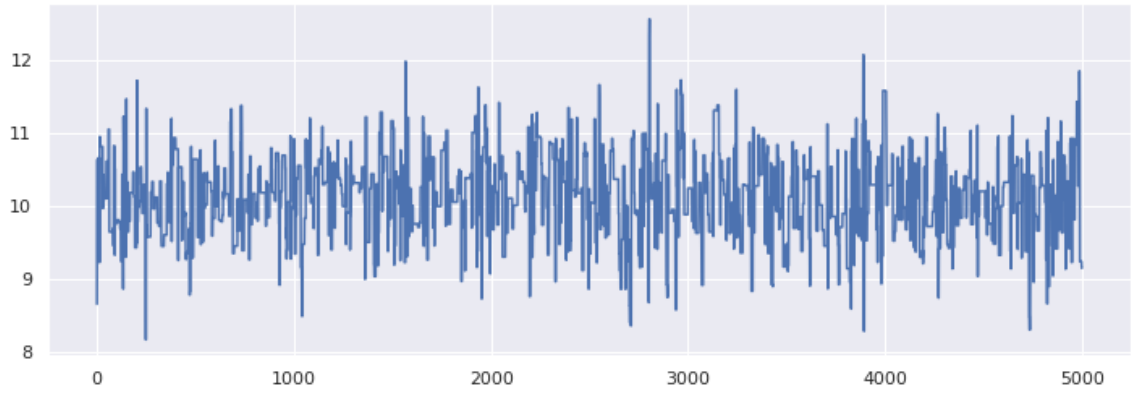

Figure 8

The distance is the Mahalanobis distance between s_0 and s_{obs} and the summary statistic chosen is the sample mean.

3.3. Unbiased Markov chain Monte Carlo methods with coupling

3.3.1 Univariate case

The idea is to apply the method of coupling to a Metropolis-Hasting MCMC, obtaining two chains X_t, Y_t that can be used to compute the time-average estimator $H_{k:m}(X, Y)$ defined above. In this way we can achieve an estimate of the unknow parameter k.

As we said before, the principal idea of coupling is founding chains that meet in a finite time and remain together after the meeting. In order to do that, candidates can be generated with a maximal coupling algorithm, which maximizes the probability of getting two samples that are the same from two given distributions

During the computation of the time-averaged estimator we used a Metropolis-Hastings algorithm to generate the pair of chains X_t, Y_t, computing

$$H_{k:m}(X,Y) = \frac{1}{m-k+1}\sum_{l=k}^{m} h(X_l) + \sum_{l=k+1}^{\tau-1} \min(1, \frac{l-k}{m-k+1})\{h(X_l) - h(Y_{l-1})\}$$

where m is the maximum number of iterations, k is the *burnin* (namely the number of iterations that the chain needs to become stable) and τ is a random variable that represents the meeting time.

The computation of the Metropolis-Hastings algorithm is similar to the version with a singular chain, but we highlight that next iteration candidate pair is obtained from a maximal coupling. Thanks to the fact that the time-averaged estimator is unbiased, we can parallelize the process running the algorithm on four different cores, putting together the estimates at the end, reducing the uncertainty.

In our studies we set the model as:

$$Y_i|\mu \sim iid\mathcal{N}(\mu, \sigma^2_{obs}) \quad \text{for i= 1,...,n}$$

$$\mu \sim \mathcal{N}(\mu_0, \sigma^2_0)$$

with $\mu_0 = 8$, $\sigma^2_0 = 4$. We generate 100 samples from a Gaussian distribution:

$$Y_{obs} \sim \mathcal{N}(\mu_{obs}, \sigma^2_{obs})$$

with $\mu_{obs} = 10$, $\sigma^2_{obs} = 3$.

We obtained the posterior distribution:

$$\mu|Y \sim \mathcal{N}(\mu_n, \sigma^2_n), \quad \mu_n \simeq 10.065, \quad \sigma^2_n \simeq 0.0298.$$

The method has been implemented with the following setting, with parallelization:
Couples of chains: 4;
Samples from each couple: 1000;
Burnin: 100.

Time Averaged Estimators mean obtained:

$$\mathbb{E}[H_{k:m}(X,Y)] = 10.042$$

that is correctly close to the observed mean.

The results are graphically shown in the following plots.

In figure 9 we can see that the chains meet after few iterations and remain together; in figure 11 we can notice that the posterior distribution is really closed to the theoretical one.

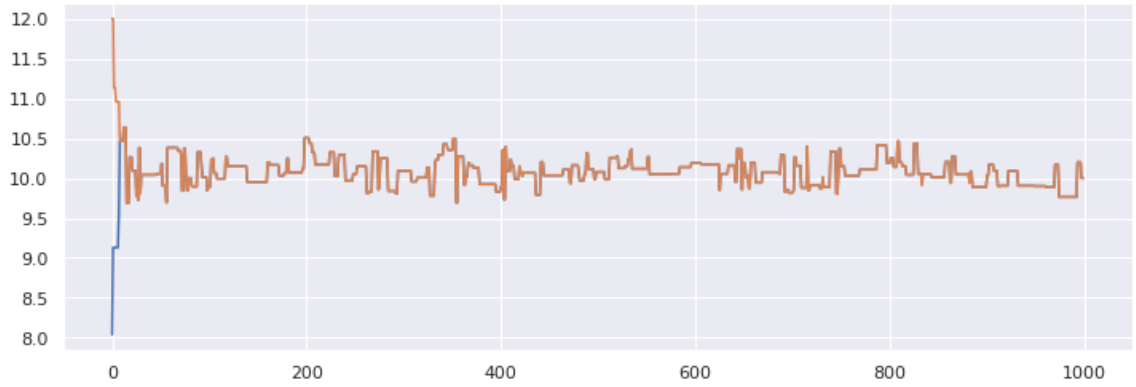

Figure 9: *Coupled chains*

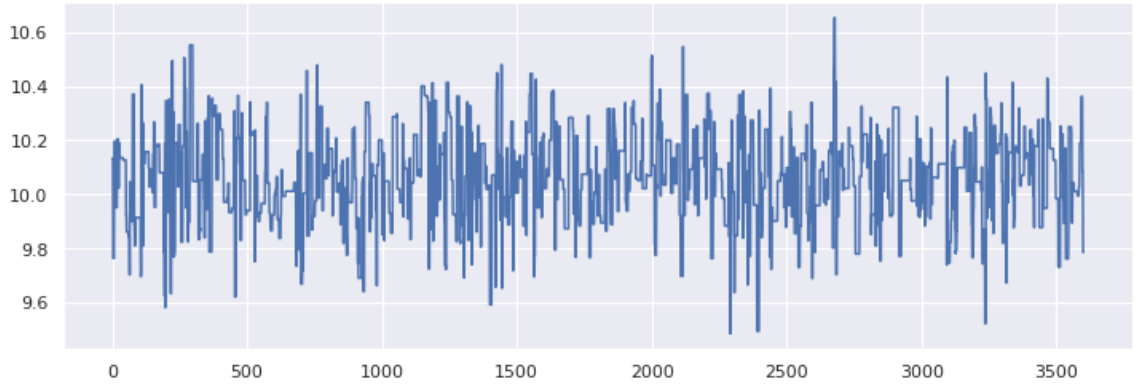

Figure 10: *Complete sampling*

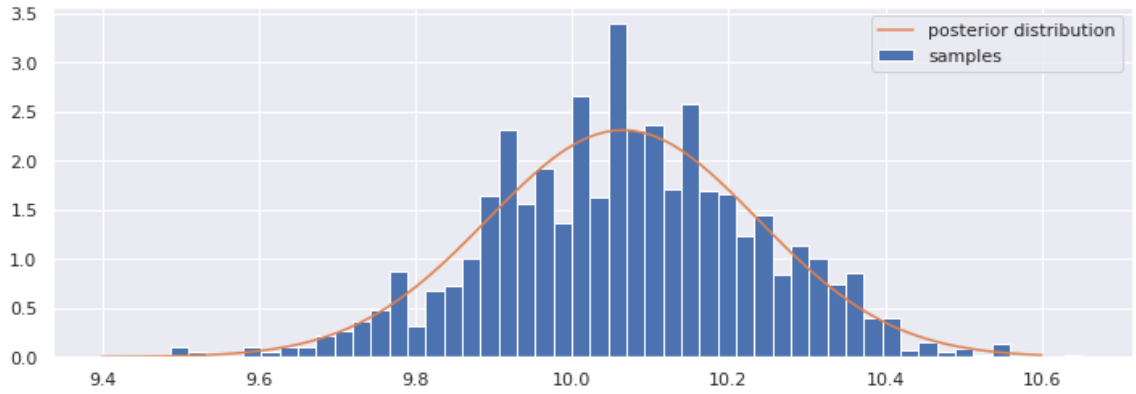

Figure 11: *Sampling histogram*

3.3.2 Multivariate case

The univariate case can be easily extended to multivariate dimension, we used the following bivariate model:

$$\boldsymbol{Y}|\boldsymbol{\mu} \overset{iid}{\sim} \mathcal{N}(\mu, \Sigma_{obs}) \quad \text{for } i = 1...n$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\mu_0, \sigma_0^2)$$

with $\boldsymbol{\mu}_0 = \begin{Bmatrix} 12 \\ 18 \end{Bmatrix}$, $\Sigma_0 = 3 \cdot \mathbb{I}$.

We generate 100 samples from a Gaussian distribution:

$$\boldsymbol{Y}_{obs} \sim \mathcal{N}(\boldsymbol{\mu}_{obs}, \Sigma_{obs})$$

with $\boldsymbol{\mu}_{obs} = \begin{Bmatrix} 10 \\ 20 \end{Bmatrix}$, $\Sigma_{obs} = 5 \cdot \mathbb{I}$.

The posterior distribution we obtain is as follows:

$$\boldsymbol{\mu}|\boldsymbol{Y}_i \sim \mathcal{N}(\boldsymbol{\mu}_n, \Sigma_n), \quad \boldsymbol{\mu}_n \simeq \begin{Bmatrix} 10.103 \\ 19.976 \end{Bmatrix}, \quad \Sigma_n \simeq 0.049 \cdot \mathbb{I}$$

The method has been implemented with the following setting, with parallelization:
Couples of chains: 4;
Samples from each couple: 800;
Burnin: 100.

Time Averaged Estimators mean obtained:

$$\mathbb{E}[H_{k:m}(\boldsymbol{X}, \boldsymbol{Y})] \simeq \begin{Bmatrix} 10.008 \\ 19.968 \end{Bmatrix}$$

Here the plot of the chains of the two variables, then the plots of the posterior sampling compared with the theoretical distribution::

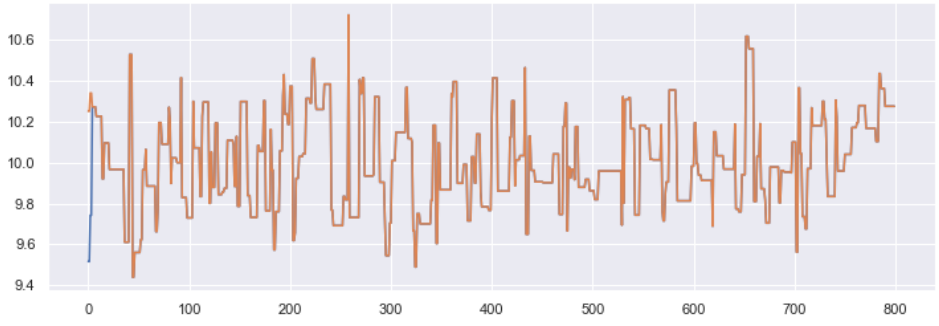

Figure 12: *First variable coupled chains*

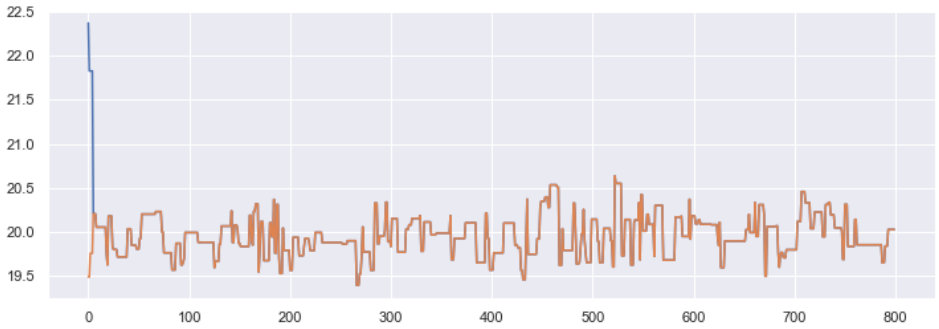

Figure 13: *Second variable coupled chains*

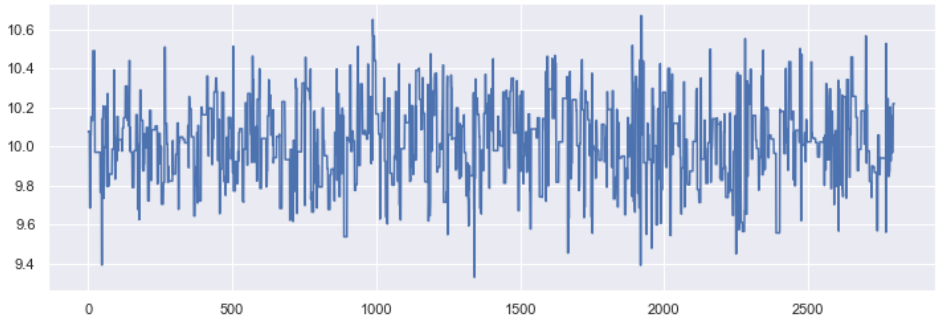

Figure 14: *First variable sampling (all chains together)*

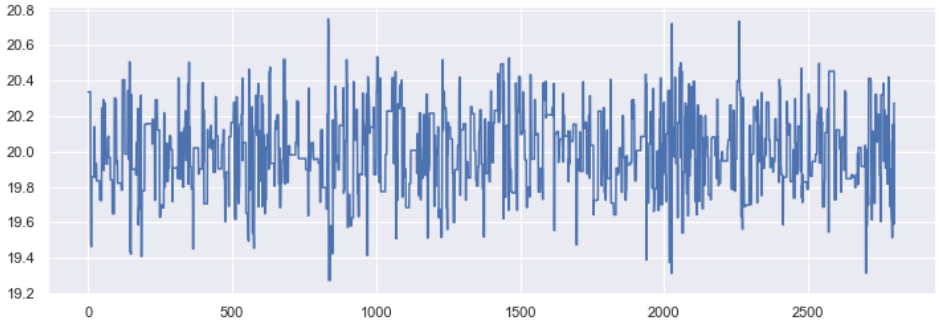

Figure 15: *Second variable sampling (all chains together)*

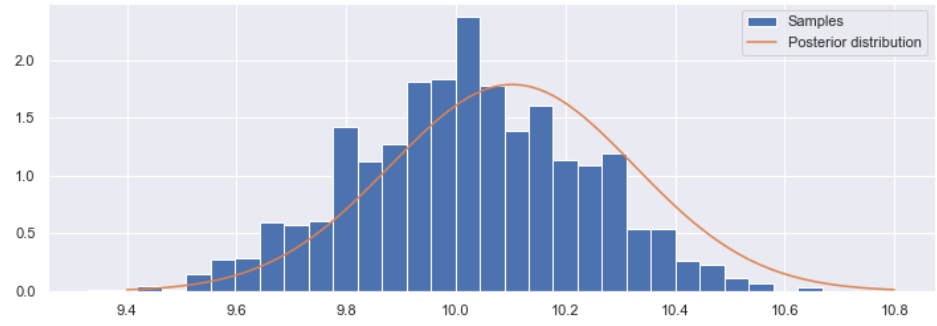

Figure 16: *First variable posterior*

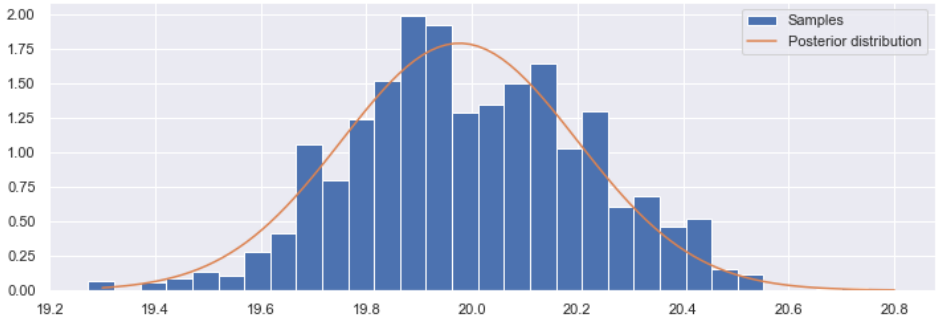

Figure 17: *Second variable posterior*

3.3.3 Technicalities on parallel implementation

In order to achieve parallelization, our choice is to use Python's standard package for parallelization, called multiprocessing (https://docs.python.org/3/library/multiprocessing.html). The Python's architecture does not allow multitreading parallelization, which would be optimal in terms of memory efficiency.

The main advantage of this technology is the possibility to have a full control of the process and the variable, but the main advantage consist of a limitation, in term of memory, of the data structure shared between processes. This limitation dumped on a trade-off between the number of coupled chain and the number of iterations: while this is acceptable for univariate sapling, such limits is too much limiting if sampling dimensionality is too high.

As we achieved the goal to parallelize the computation we do not attempt any other implementation, but there exist many other tools that can solve those issues and obtain a faster execution.

3.4. The complete method

3.4.1 Univariate case with fixed variance

We implemented an MCMC algorithm using Approximate Bayesian Computation to approximate the likelihood and Coupled Markov chanis. The algorithm we computed:

1. Compute $s_{obs} = S(y_{obs})$;
2. generate $\theta_x^{(0)} \sim \pi(\mu)$ and $\theta_y^{(0)} \sim \pi(\mu)$ from prior density;
3. generate with a maximal coupling two samples of N observations such that $y_{1i} \sim \mathcal{N}(\theta_x^{(0)}, \sigma_{obs}^2)$ and $y_{2j} \sim \mathcal{N}(\theta_y^{(0)}, \sigma_{obs}^2)$;
4. compute $s_x^{(0)} = S(y_1)$ and $s_y^{(0)} = S(y_2)$;
5. until $K_h(||s_x^{(0)} - s_{obs}||) > 0$:
   - generate $\theta_x^{(0)} \sim \pi(\mu)$ from prior density;
   - generate a sample of N observations such that $y_{1i} \sim \mathcal{N}(\theta_x^{(0)}, \sigma_{obs}^2)$;
   - compute $s_x^{(0)} = S(y_1)$;
6. until $K_h(||s_y^{(0)} - s_{obs}||) > 0$:
   - generate $\theta_y^{(0)} \sim \pi(\mu)$ from prior density;
   - generate a sample of N observations such that $y_{2j} \sim \mathcal{N}(\theta_y^{(0)}, \sigma_{obs}^2)$;
   - compute $s_y^{(0)} = S(y_2)$;
8. for i = 1,...,N:
   - generate $[\theta_x^{(i)}, \theta_y^{(i)}]$ from a maximal coupling given $[\theta_x^{(i-1)}, \theta_y^{(i-1)}]$;
   - generate from a maximal coupling two samples of N observations $y_1 \sim p(y|\theta_x^{(i)})$ and $y_2 \sim p(y|\theta_y^{(i)})$;
   - compute $s_x^{(i)} = S(y_1)$ and $s_y^{(i)} = S(y_2)$;
   - accept $\theta_x^{(i)}$ with probability
   $$\frac{K_h(||s_x^{(i)} - s_{obs}||)\pi(\theta_x^{(i)})}{K_h(||s_x^{(i-1)} - s_{obs}||)\pi(\theta_x^{(i-1)})}$$
   and accept $\theta_y^{(i)}$ with probability
   $$\frac{K_h(||s_y^{(i)} - s_{obs}||)\pi(\theta_y^{(i)})}{K_h(||s_y^{(i-1)} - s_{obs}||)\pi(\theta_y^{(i-1)})}.$$

As output:

$$\theta_x^{(1)}, ..., \theta_x^{(N)} \sim \pi_{ABC}(\theta|y_{obs});$$
$$\theta_y^{(1)}, ..., \theta_y^{(N)} \sim \pi_{ABC}(\theta|y_{obs}).$$

We highlight that the key point is how each iteration works: given the current values of θ, a pair of candidates is generated from a maximal coupling; two samples of dimension n are generated from the law of y given θ, maximizing the probability that the samples are the same; the summary statistics is computed and the candidate $\theta_x^{(i)}$, $\theta_y^{(i)}$ are accepted with the *Metropolis Hastings acceptance probability*. As output we get two sets of parameter vectors of samples from the posterior distribution of θ given S.

In our study case, we used the sample mean as summary statistics, a L^2-norm as distance function and a Gaussian kernel function:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}, \quad K_h(u) = \frac{K(\frac{u}{h})}{h}$$

The model is the same as the above implementation:

$$Y_i|\mu \sim iid\mathcal{N}(\mu, \sigma^2_{obs}) \quad \text{for i= 1,...,n}$$

$$\mu \sim \mathcal{N}(\mu_0, \sigma^2_0)$$

with $\mu_0 = 8,\ \sigma^2_0 = 4$. We generate $n = 100$ samples from a Gaussian distribution:

$$Y_{obs} \sim \mathcal{N}(\mu_{obs}, \sigma^2_{obs})$$

with $\mu_{obs} = 10,\ \sigma^2_{obs} = 3$.

We obtained the posterior distribution:

$$\mu|Y \sim \mathcal{N}(\mu_n, \sigma^2_n), \quad \mu_n \simeq 10.065, \quad \sigma^2_n \simeq 0.0298.$$

The results are shown in the following plots.

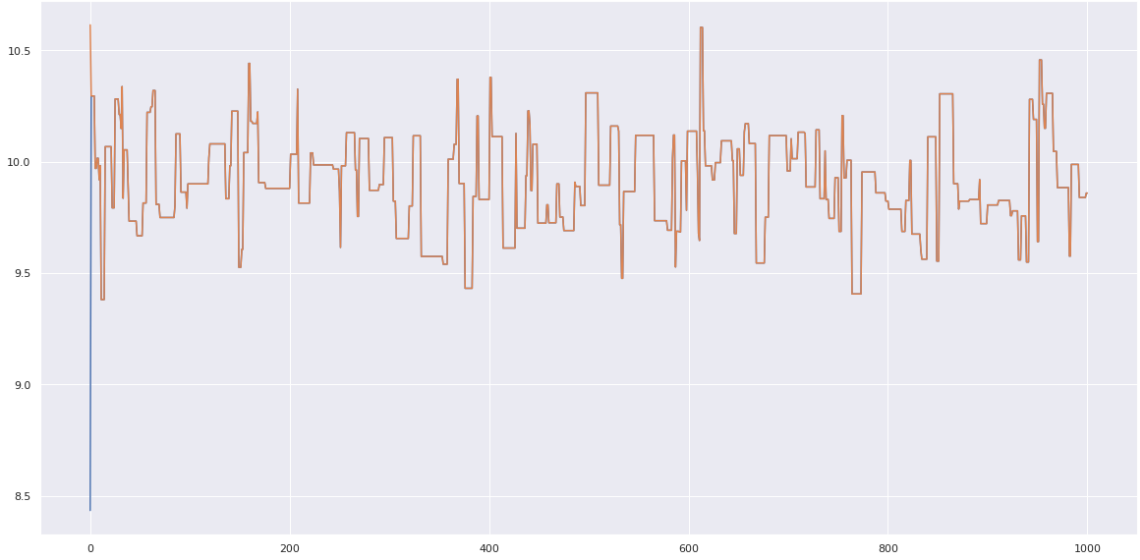

Figure 18: *Coupled chains*

We can see in figure 18 that the chains meet after few iterations remaining together while they converge to the posterior mean μ_n. In figure 19 we can see that our model approximates quite well the real distribution, even if the computation could be improved increasing the number of iterations performing a parallelization.

3.4.2 Multivariate case with fixed variance

The complete method can be easily extended to the multivariate version. We performed the same bivariate model that we used in the single multivariate coupling.

$$\boldsymbol{Y}|\boldsymbol{\mu} \overset{iid}{\sim} \mathcal{N}(\mu, \Sigma_{obs}) \quad \text{for } i = 1...n$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\mu_0, \sigma^2_0)$$

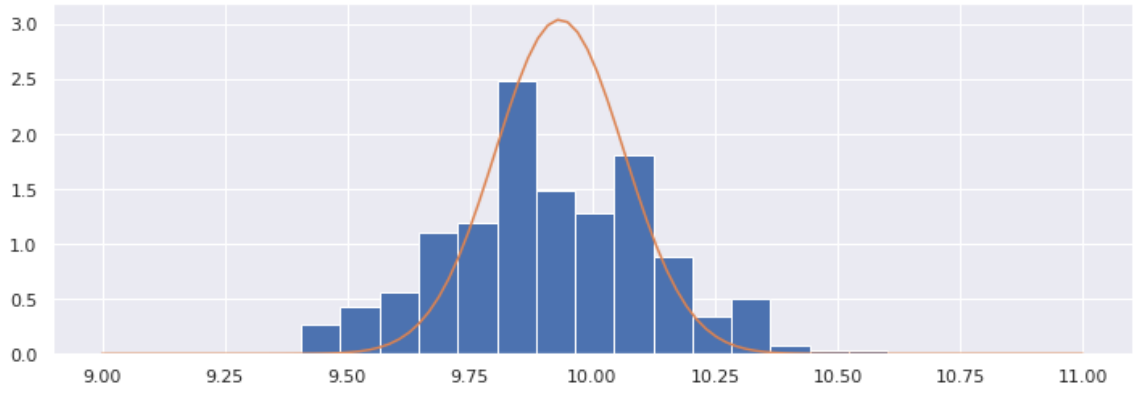

Figure 19: *Sampling histogram with real distribution*

with $\boldsymbol{\mu}_0 = \begin{Bmatrix} 12 \\ 18 \end{Bmatrix}$, $\Sigma_0 = 3 \cdot \mathbb{I}$.

We generate 100 samples from a Gaussian distribution:

$$\boldsymbol{Y}_{obs} \sim \mathcal{N}(\boldsymbol{\mu}_{obs}, \Sigma_{obs})$$

with $\boldsymbol{\mu}_{obs} = \begin{Bmatrix} 10 \\ 20 \end{Bmatrix}$, $\Sigma_{obs} = 5 \cdot \mathbb{I}$.

The posterior distribution we obtain is as follows:

$$\boldsymbol{\mu}|\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}_n, \Sigma_n), \quad \boldsymbol{\mu}_n \simeq \begin{Bmatrix} 9.556 \\ 20.595 \end{Bmatrix}, \quad \sigma_n^2 \simeq 0.050$$

Time Averaged Estimators mean obtained:

$$\mathbb{E}[H_{k:m}(\boldsymbol{X}, \boldsymbol{Y})] \simeq \begin{Bmatrix} 9.536 \\ 20.551 \end{Bmatrix}$$

Here the plot of the chains of the two variables, then the plots of the posterior sampling compared with the theoretical distribution::

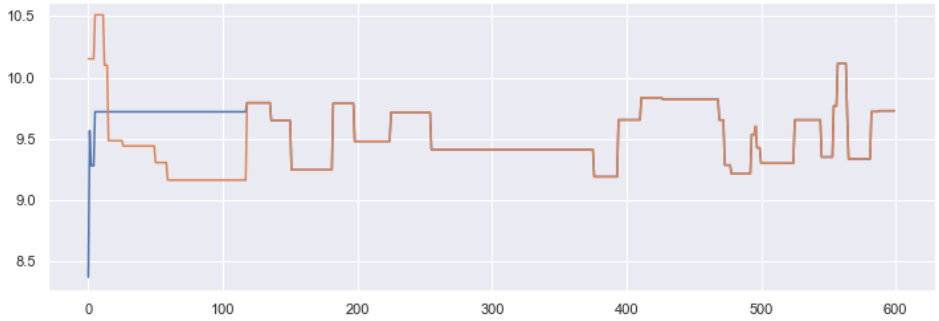

Figure 20: *First variable coupled chains*

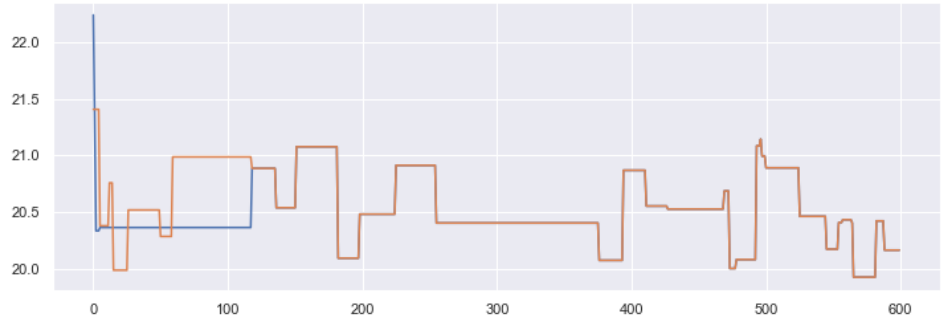

Figure 21: *Second variable coupled chains*

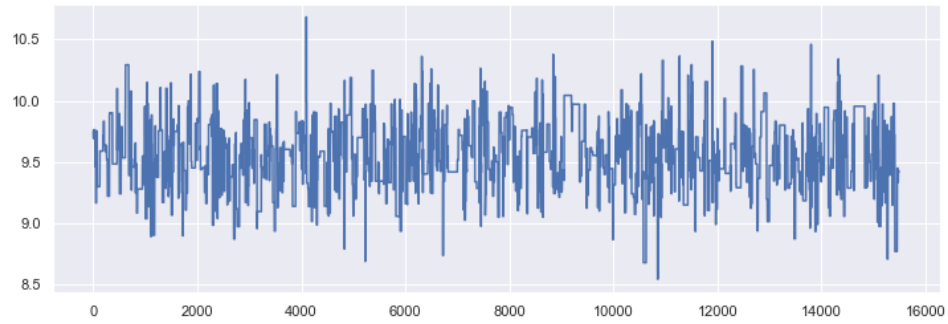

Figure 22: *First variable sampling (all chains together)*

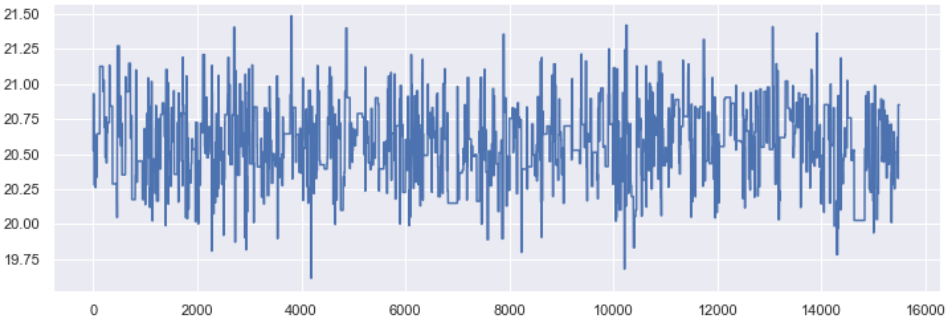

Figure 23: *Second variable sampling (all chains together)*

3.4.3 Univariate case with both mean and variance unknown

This is also an extension of the Univariate case with fixed variance where $\theta = (\mu, \sigma)$

We assume as model:

$$Y_i|\mu \sim iid\mathcal{N}(\mu, \sigma^2) \quad \text{for i= 1,...,n}$$
$$\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$$

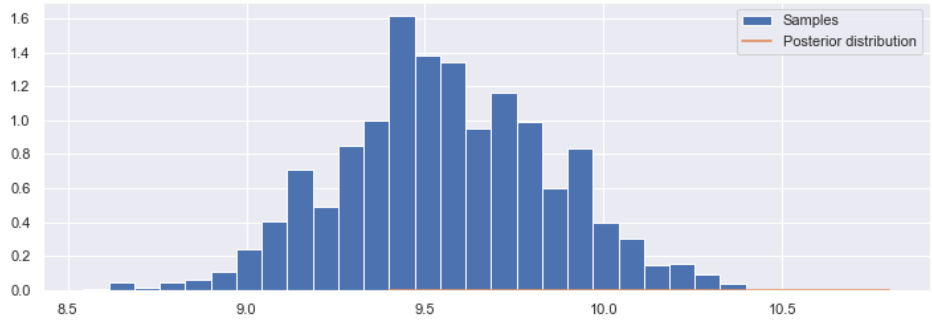

Figure 24: *First variable posterior*

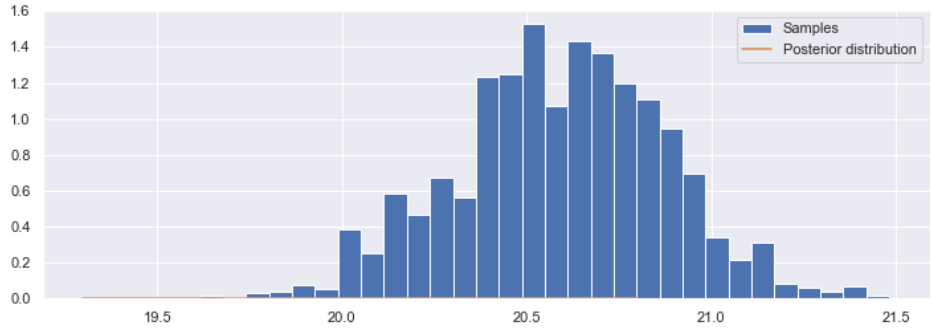

Figure 25: *Second variable posterior*

$$\sigma^2 \sim InvGa(a_0, b_0)$$

with $\mu_0 = 8, \sigma_0^2 = 4$ and $a_0 = 1, b_0 = 1$.

We generate $n = 100$ samples from a Gaussian distribution:

$$Y_{obs} \sim \mathcal{N}(\mu_{obs}, \sigma_{obs}^2)$$

with $\mu_{obs} = 10$, $\sigma_{obs}^2 = 3$. As sufficient summary statistics we set (sample mean, sample variance), a L^2-norm as distance function and a Gaussian kernel function

For the MH ABC and coupled MCMC algorithm we define as proposal distribution for the parameters' update:

$$\mu^{i+1} \sim \mathcal{N}(\mu^i, 0.1^2)$$
$$log(\sigma^{2(i+1)}) \sim \mathcal{N}(log(\sigma^{2(i)}), 0.1^2)$$

the logarithm trasformation is needed to keep the proposed value of σ^2 within its domain, which is $\mathbb{R}^+$

Posterior: It would be impossible to compute the marginal posterior distribution, thus we obtain the conditional posterior distribution, or full conditional distributions: Given $\mathbf{y} \in \mathbb{R}$

$$\mu|\sigma^2, \mathbf{y} \sim \mathcal{N}(\sigma_n \mu_n, \sigma_n^2)$$
$$\sigma^2)|\mu, \mathbf{y} \sim InvGa(a_n, b_n)$$
$$\text{where } \mu_n = n\bar{y}/\sigma_0^2 + \mu_0/\sigma_0^2$$

and $a_n = a + n/2n = 2, b_n = b + n(\mu - \bar{y})^2/2 + \sum_{i=1}^n (y - \bar{y})^2/2$

4. Numerical Experiments

In order to test our method on a more complex problem, we followed the common numerical experiment in the literature: ABC applied on the g-and-k distribution. The goal is to, not only, apply ABC algorithm to this problem, but also the Unbiased coupled Markov chain Monte Carlo method.

4.1. Univariate g-and-k distribution

A classical numerical experiments in the ABC literature is to test the algorithm on the univariate g-and-k distribution. The likelihood is intractable, therefore the distribution is defined in terms of its quantile function:

$$r \in (0,1) \longmapsto a + b(1 + 0.8\left(\frac{1 - exp(-g \cdot z(r))}{1 + exp(-g \cdot z(r))}\right)(1 + z(r)^2)^k \cdot z(r)$$

where $z(r)$ is the r-th quantile of the standard Normal distribution, while a and b are location and scale parameters and g and k are related to skewness and kurtosis.

Sampling from this distribution can be done by generating $z(r)$s as samples from a $\mathcal{N}(0,1)$: $z(r) \sim \mathcal{N}(0,1)$

To complete the model we assume prior probability distribution as follows:

$$\begin{aligned} a &\sim \mathcal{U}([0,10]) \\ b &\sim \mathcal{U}([0,10]) \\ g &\sim \mathcal{U}([0,10]) \\ k &\sim \mathcal{U}([0,10]) \end{aligned}$$

To test this distribution we sampled a set of observation y_{obs} imposing the following values to the parameters, following the path of Jacob et al.:

a=3, b=1, g=2, k=0.5

A key point in the ABC procedure is the decision of the summary statistics, which in this case, are taken as the 10 equidistant quantiles as well as the minimum.

As before we set

Distance:

$$L^2 - norm\ of\ the\ difference\ of\ S(y)\ and\ s_{obs}$$

Kernel function:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}, \quad K_h(u) = \frac{K(\frac{u}{h})}{h}$$

4.1.1 ABC and coupled Markov chain Monte Carlo method

given $\theta = (a, b, g, k)$

1. Initialization: for k in (1,2): $\theta_k^{(0)} \sim \mathcal{U}([0,10]^4)$
   - for j in (1,...,n_{obs}):
     - $z^{(0,j)}(r) \sim \mathcal{N}(0,1)$
     - $y_k^{(0,j)} \sim quantilefunction(z^{(0,j)}(r), \theta_k^{(0)})$
     - compute $s_k^{(0)} = S(y_k^{(0)})$
   - Untill $Kh(\|s_k^{(0)} - s_{obs}\|) > 0$:
     - Generate $\theta_k^{(0)} \sim \mathcal{U}([0,10]^4)$ from prior density.
     - Generate $z^{(0)}(r) \sim \mathcal{N}(0,1)$
     - Generate a sample of 1000 observations such that $y_k \sim quantilefunction(z(0)(r), \theta_k^0)$

- Compute $s_k^{(0)} = S(y_k)$

2. for i in (1,...,M):

- generate $\theta_1^{(i)}, \theta_2^{(i)}$ from maximalcoupling$(\theta_1^{(i-1)}, \theta_2^{(i-1)})$
  for j in (1,...,n):
  - $z^{(i,j)}(r) \sim \mathcal{N}(0,1)$
  - generate:
    $y_1^{(ij)} \sim quantilefunction(z^{(i,j)}(r), \theta_1^{(i-1)})$
    $y_2^{(ij)} \sim quantilefunction(z^{(i,j)}(r), \theta_2^{(i-1)})$
- Compute the summaries $s_k^{(i)} = S(y_k^{(i)})$ for k in $(1,2)$
- for k in (1,2): Accept $\theta_k^{(i)}$ with probability $\frac{Kh(\|s_k^{(i)} - s_{obs}\|)\pi(\theta_k^{(i)})}{Kh(\|s_k^{(i-1)} - s_{obs}\|)\pi(\theta_k^{(i-1)})}$ otherwise $\theta_k^{(i)} = \theta_k^{(i-1)}$; k in $(1,2)$

In the algorithm used before we generated the simulated datasets from $y|\theta \sim f(\cdot|\theta)$ with maximal coupling, which guarantees that the chains stay together after they've met. In this case that's not applicable because there is not a density function, so we used a different approach:

After sampling the paramters θ_1 and θ_2 from the maximal coupling algorithm, we generate n_{obs} samples $z(r) \sim \mathcal{N}(0,1)$ and then we compute the two Markov Chains

$$y_1 \sim quantilefunction(z(r), \theta_1)$$

$$y_2 \sim quantilefunction(z(r), \theta_2)$$

Therefore, if θ_1 and θ_2 from maximal coupling are identical, then also the chains y_1 and y_2 will stay together.