Monte Carlo Hand-in Assignment 3

Eliot Montesino Petrén, el6183mo-s, 990618-9130, LU Faculty of Engineering F19 , eliot.mp99@gmail.com,
Kasper Nordenram, ka0884no-s, 990212-7910, LU Faculty of Engineering F19

INTRODUCTION

In this assignment we are tasked with applying Monte Carlo and Empirical Methods for analyzing historical coal mine disasters in Great Britain and estimating significant wave-heights in the North Atlantic. Utilizing several methods and techniques in the field such as Gamma priors and parametric bootstrap methods, the goal is to model disaster intensity changes over time and predict extreme weather events. Matlab is used to conduct the data analysis.

PART 1: COAL MINE DISASTERS—CONSTRUCTING A COMPLEX MCMC ALGORITHM

The model has 3 (groups of) parameters. The first is the intensities λ_i, $i = 1, 2, \ldots, d$, sampled from $\Gamma(2,\ \theta)$ distributions. The second is the θ-parameter, sampled from a $\Gamma(2,\ \Psi)$ distribution. Finally, we have the breakpoints, sampled from a distribution proportional to

$$f(\mathbf{t}) = \begin{cases} \prod_{i=1}^{d}(t_{i+1} - t_i), & \text{for } t_1 < t_2 < \ldots < t_{d+1} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

We then also have the parameter d, the number of breakpoints $+1$ (giving d sections with different intensity), the hyperparameter Ψ, and the random walk parameter ρ. We have been given the conditional density

$$f(\tau|\lambda, \mathbf{t}, \theta) = \exp\left(-\sum_{i=1}^{d}\lambda_i(t_{i+1} - t_i)\right)\prod_{i=1}^{d}\lambda_i^{n_i(\tau)}. \tag{2}$$

We tried deriving this from scratch, we found an expression with some normalization factors that are not included here. The derivation has been put in Appendix III with a disclaimer, as we have not checked it too thoroughly. The model can be seen as the Bayesian hierarchical model of the form

$$f(\theta, \lambda, \mathbf{t}|\tau) = \frac{f(\tau|\theta, \mathbf{t}) \cdot f(\lambda|\theta) \cdot f(\theta)f(\mathbf{t})}{f(\tau)}. \tag{3}$$

a)

θ: Using Bayes theorem,

$$f(\theta|\lambda, \mathbf{t}, \tau) = \frac{f(\lambda|\theta, \mathbf{t}, \tau)f(\theta)}{f(\lambda)} \propto f(\lambda|\theta, \mathbf{t}, \tau)f(\theta). \tag{4}$$

Here, the density $f(\lambda|\theta, \mathbf{t}, \tau)$ is

$$f(\lambda|\theta, \mathbf{t}, \tau) = f(\lambda|\theta) = \prod_{i=1}^{d}\mathbb{P}(\lambda_i), \tag{5}$$

which with the Gamma(2, θ) distribution (shape-rate parametrization) gives

$$f(\lambda|\theta, \mathbf{t}, \tau) = \prod_{i=1}^{d}\frac{\theta^2}{\Gamma(2)}\lambda_i^{2-1}e^{-\lambda_i\theta} = \prod_{i=1}^{d}\theta^2\lambda_i e^{-\lambda_i\theta}, \tag{6}$$

while

$$f(\theta) = \frac{\Psi^2}{\Gamma(2)}\theta^{2-1}e^{-\theta\Psi} = \Psi^2\theta e^{-\theta\Psi}. \tag{7}$$

Put together, we get

$$f(\theta|\lambda, \mathbf{t}, \tau) \propto \Psi^2\theta e^{-\theta\Psi}\prod_{i=1}^{d}\theta^2\lambda_i e^{-\lambda_i\theta} \propto \theta^{2d+1}e^{-\theta\Psi}\prod_{i=1}^{d}e^{-\lambda_i\theta} = \theta^{2d+1}e^{-\theta\Psi-\theta\sum_{i=1}^{d}\lambda_i}, \tag{8}$$

for $\theta > 0$. We can see that this is proportional to a Γ distribution in θ with parameters

$$\text{shape} = 2d + 2 \tag{9a}$$

$$\text{rate} = \Psi + \sum_{i=1}^{d}\lambda_i, \tag{9b}$$

so this is the normalized distribution that we will sample from. When using `Matlab` to sample, we will invert the second parameter to get the shape-scale parametrization.

λ_i: Using Bayes theorem again,

$$f(\lambda|\theta, \mathbf{t}, \tau) = \frac{f(\tau|\theta, \mathbf{t}, \lambda)f(\lambda|\theta)}{f(\tau)} \propto f(\tau|\theta, \mathbf{t}, \lambda)f(\lambda|\theta). \tag{10}$$

We find $f(\tau|\theta, \mathbf{t}, \lambda)$ in Equation 2, and $f(\lambda|\theta)$ in Equation 6. Put together, we get

$$f(\lambda|\theta,\mathbf{t},\tau) \propto$$
$$\exp\left(-\sum_{i=1}^{d}\lambda_i(t_{i+1}-t_i)\right)\prod_{i=1}^{d}\lambda_i^{n_i(\tau)}\cdot\prod_{i=1}^{d}\theta^2\lambda_i e^{-\lambda_i\theta} \propto$$
$$\prod_{i=1}^{d}\lambda_i^{n_i(\tau)}e^{-\lambda_i(t_{i+1}-t_i)}\cdot\prod_{i=1}^{d}\lambda_i e^{-\lambda_i\theta} =$$
$$\prod_{i=1}^{d}\lambda_i^{n_i(\tau)+1}e^{-\lambda_i[(t_{i+1}-t_i)+\theta]}. \quad (11)$$

We see that the distribution factorizes into

$$f(\lambda|\theta,\mathbf{t},\tau)=\prod_{i=1}^{d}f_\Gamma(\lambda_i;n_i(\tau)+2,t_{i+1}-t_i+\theta). \quad (12)$$

As the λ_i:s are clearly independent, we can sample from them independently of each other without losing performance. Thus, we sample

$$\lambda_i \in \Gamma(n_i(\tau)+2,t_{i+1}-t_i+\theta), \quad (13)$$

for all i. Also here, we will when sampling invert the second, the rate, parameter to get a shape-scale parametrization used in Matlab.

$\mathbf{t}$*:* We once again attack the problem using Bayes, giving

$$f(\mathbf{t}|\theta,\lambda,\tau)=\frac{f(\tau|\theta,\mathbf{t},\lambda)f(\mathbf{t})}{f(\tau)}\propto f(\tau|\theta,\mathbf{t},\lambda)f(\mathbf{t}). \quad (14)$$

$$\alpha=\min\left(1,\frac{f(X^*)r(X_k|X^*)}{f(X_k)r(X^*|X_k)}\right)=\min\left(1,\frac{f(X^*)}{f(X_k)}\right), \quad (15)$$

We have the density $f(\tau|\theta,\mathbf{t},\lambda)$ in Equation 2, and $f(\mathbf{t})$ in Equation 1. Put together, and assuming the points are ordered correctly (otherwise the density is 0), we get

$$f(\mathbf{t}|\theta,\lambda,\tau)^* \propto$$
$$\exp\left(-\sum_{i=1}^{d}\lambda_i(t_{i+1}-t_i)\right)\prod_{i=1}^{d}\lambda_i^{n_i(\tau)}\cdot\prod_{i=1}^{d}(t_{i+1}-t_i), \quad (16)$$

where $n_i(\tau)$ is dependent on the t_i:s. The star symbolizes the assumption that the points are ordered. This is not a well-known distribution, and does not factorize, so we will have to sample from it using the Metropolis-Hastings algorithm. This will require evaluating the ratios of two of these densities. The factors $\lambda_i^{n_i(\tau)}$ can be, for example, 4^600. This is larger than the largest possible float in `Matlab`, so it calls for a better way to handle the ratio. One way is to calculate the logarithm of the ratio and exponentiate this. We have instead rewritten it

$$\frac{f(\mathbf{t}^*|\theta,\lambda,\tau)}{f(\mathbf{t}_k|\theta,\lambda,\tau)}=$$
$$\frac{\exp\left(-\sum_{i=1}^{d}\lambda_i(t^*_{i+1}-t^*_i)\right)\prod_{i=1}^{d}\lambda_i^{n_i^*(\tau)}\cdot\prod_{i=1}^{d}(t^*_{i+1}-t^*_i)}{\exp\left(-\sum_{i=1}^{d}\lambda_i(t^k_{i+1}-t^k_i)\right)\prod_{i=1}^{d}\lambda_i^{n_i^k(\tau)}\cdot\prod_{i=1}^{d}(t^k_{i+1}-t^k_i)}=$$
$$\exp\left(-\sum_{i=1}^{d}\lambda_i(t^*_{i+1}-t^*_i-t^k_{i+1}+t^k_i)\right)$$
$$\prod_{i=1}^{d}\lambda_i^{n_i^*(\tau)-n_i^k(\tau)}\prod_{i=1}^{d}\frac{t^*_{i+1}-t^*_i}{t^k_{i+1}-t^k_i}, \quad (17)$$

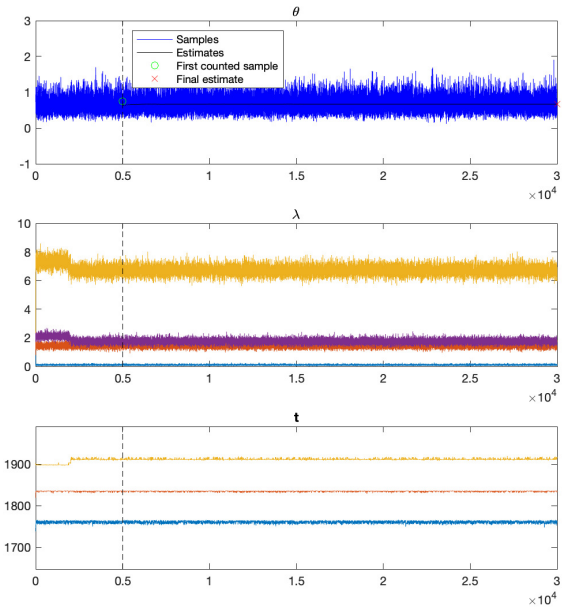
b)

We will sample from $f(\theta,\lambda,\mathbf{t}|\tau)$ using a hybrid MCMC algorithm. We will sample first from the conditional distribution for θ, then the conditional distributions of the λ_i:s one at a time, both using the inverse method. Finally, we will sample from the conditional distribution of $\mathbf{t}$, using a random walk proposal, one t_i at a time. We can see that the acceptance probability is

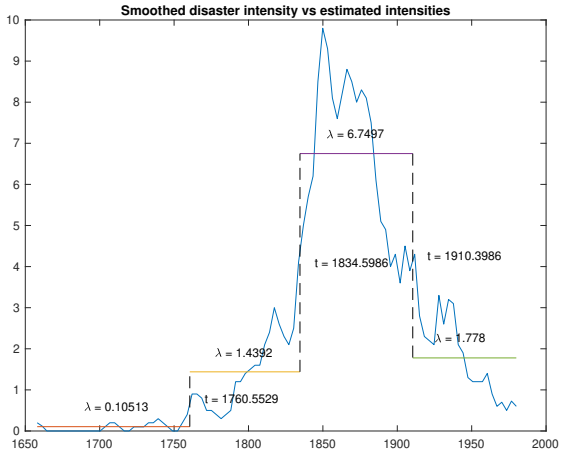
as the transition density $r(\cdot|\cdot)$ is symmetrical. We choose to do it one t_i at a time, as it seems like achieving proper mixing is harder when doing all proposals at a time - the acceptance probability goes to zero as t_i:s get close, and it seems unnecessary that two t_i:s getting close should stop the entire vector from being accepted. From a calculation point of view, this is heavier, as the densities have to be evaluated more times, although a lot of this can be rationalized away as there are many common factors in the densities if only one t_i moves. This gives a valid MCMC sampler according to [1]. The method employed boils down to a Gibbs algorithm, where each parameter θ, λ_i, and t_i is sampled conditional on all other parameters, with different sampling methods for the individual parameters based on what information is available (exact density or unnormalized density). (vad menar han med motivera?). The λ_i:s will be initialized as the average intensity, while the t_i:s will be initialized spread out uniformly . We see that the parameter Ψ decides how spread out we will allow the λ_i:s to be. If we set it very large, θ will be very large, and then the λ parameters are forced close to 0. An example run with Ψ at 5, ρ at 0.025, 5000 burn-in samples, 25000 samples to average over, and 3 breakpoints, is shown in Figure 1.

c)

Here, we set Ψ to 5 and ρ to 0.025. We used 5000 burn-in samples, and 25000 samples to average over to get the final estimates. In Figure 2, we see the found breakpoints when looking for different numbers of them, overlayed on a smoothing of the density, found as the moving average with a window length of 10 years, i.e. the number of counts in the window divided by the window length. An estimate by eye of this graph could give something like 5 breakpoints, but since the data seems very noisy this could be too much (or even too little). The breakpoint values are given in the graphs. With 6 breakpoints, there was a lot more movement



(a) Illustration of sampling process.



(b) Resulting estimates.

Fig. 1: Example of MCMC sampling.

of the breakpoints, indicating that the parameter density was less concentrated, possibly indicating that there were too many breakpoints. We also see that the first breakpoint end up in a quite uninformative spot. Therefore, 5 seems like a good estimate of the number of actual breakpoints. We could also use the likelihood ratio test, since we have the density (even normalized) of the measurements given the parameters.

d)

To evaluate how sensitive the posteriors are to the choice of Ψ, we have found them for different values of Ψ. We have used ρ set at 0.025, 5000 burn-in samples, 25000 samples to average over, and 3 breakpoints. In Figures 3 through 5, we see the found posteriors for $\Psi = \{1, 10, 100, 10000\}$.

We can begin by looking at the values of θ, as those are directly affected by the value of Ψ. We see that the posterior distribution of θ goes through a quite dramatic compression towards 0 with increasing Ψ, as is expected given that the posterior rate parameter of the θ Γ-distribution is

$$\text{rate}_\theta = \Psi + \sum_{i=1}^{d} \lambda_i. \tag{18}$$

A higher rate parameter compresses the distribution towards 0, exactly what we see. From this, we also see that for small values of Ψ, we essentially have a flat prior, as the rate parameter will be determined mostly by the λ-values.

Looking now at these λ_i:s, we know that they are affected by Ψ *through* θ. We see that the shift is much smaller than for θ. In fact, they seem essentially unaffected. Looking at how they are affected by θ, we see that it is through their rate parameter, which is

$$\text{rate}_{\lambda_i} = t_{i+1} - t_i + \theta. \tag{19}$$

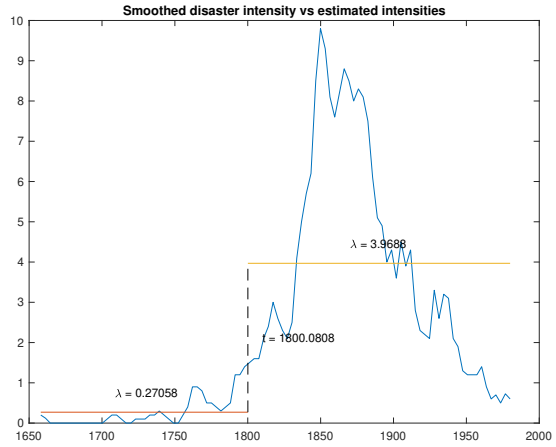
This makes it clear why we see such a small shift with an increasing Ψ. The typical distance between breakpoints is on a scale of 50 or more. Thus, a shift in θ of around 1 will have a small effect. When breakpoints get really close, the effect can be larger, but this is not a typical scenario, and for 3 breakpoints, a scenario that is not seen at all. We also see from this expression that for very high values of Ψ (giving very small values of θ), we essentially have a flat prior on the λ_i:s.

We can also look at the breakpoint locations, but since these are affected by Ψ only through the λ_i:s, which are already essentially unaffected, this is not necessary, to be able to draw the conclusion that the parameter Ψ has a quite small effect on the found piecewise constant intensity, which is what we are actually after. It is only a parameter in a prior, θ, that is affected in a major way, and the found value of this parameter is quite uninteresting for the application at hand.

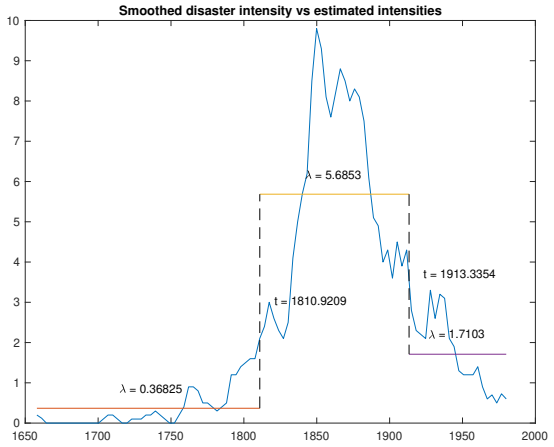
e)

The variance of the estimate using a MCMC sampler is dependent on the covariance function of the chain [2], such that

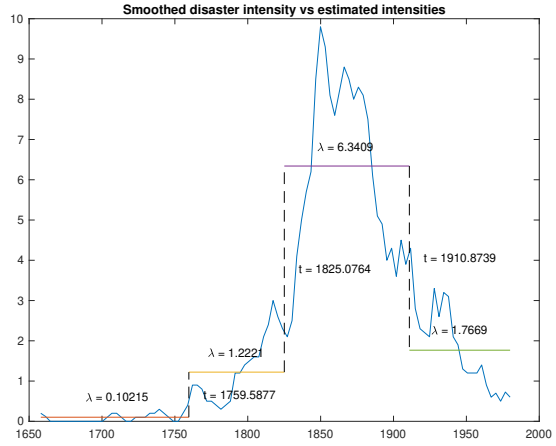
$$\sigma^2 = r(0) + 2\sum_{l=1}^{\infty} r(l). \tag{20}$$



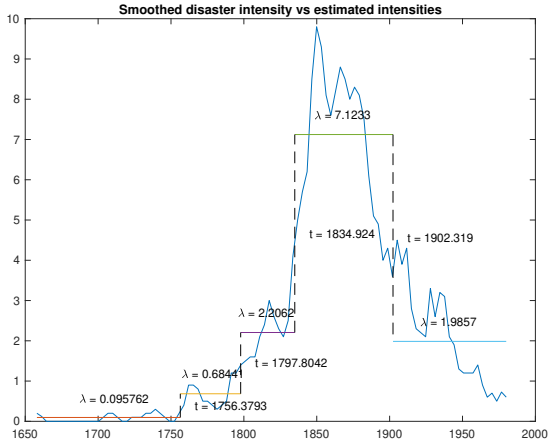
(a) Found piecewise constant intensity with 1 breakpoint



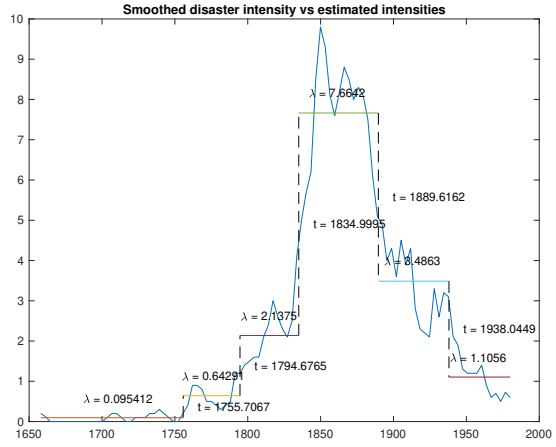
(b) Found piecewise constant intensity with 2 breakpoints



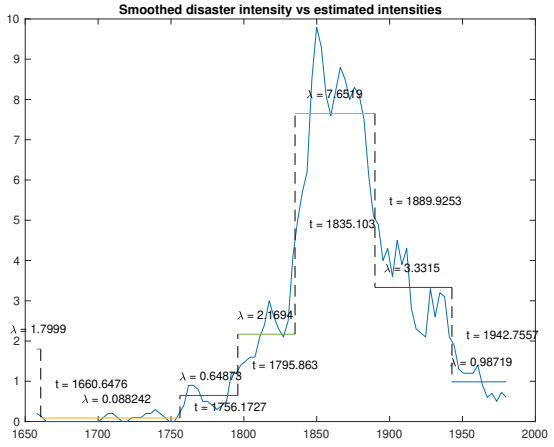
(c) Found piecewise constant intensity with 3 breakpoints



(d) Found piecewise constant intensity with 4 breakpoints



(e) Found piecewise constant intensity with 5 breakpoints



(f) Found piecewise constant intensity with 6 breakpoints

Fig. 2: Average of MCMC samples using different numbers of breakpoints.

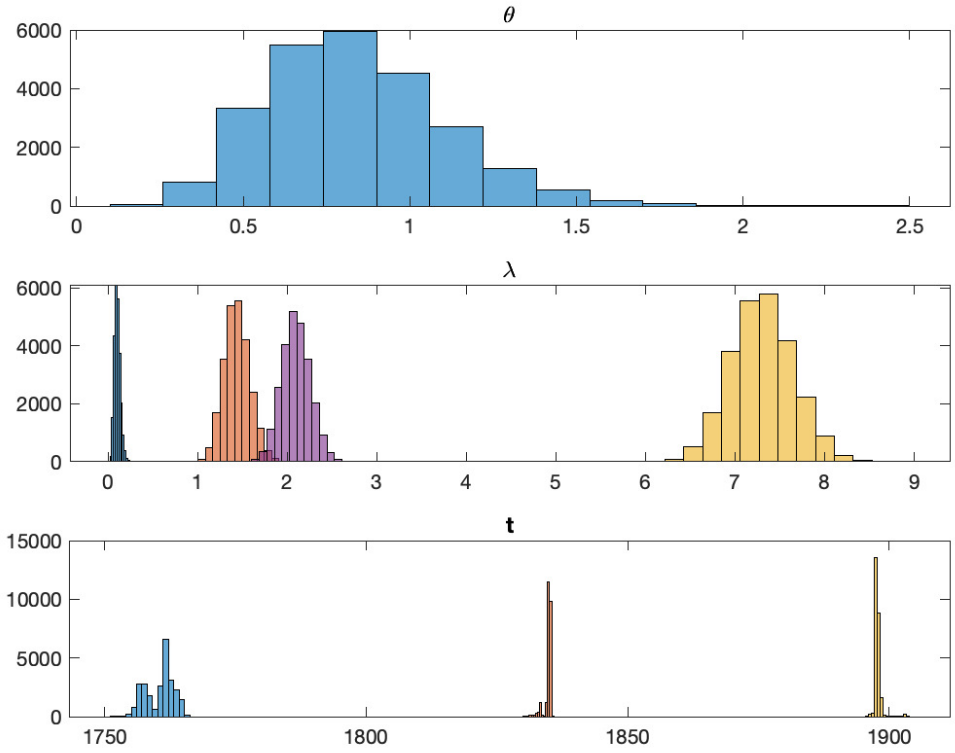


Fig. 3: Posteriors when Ψ is set to 1.

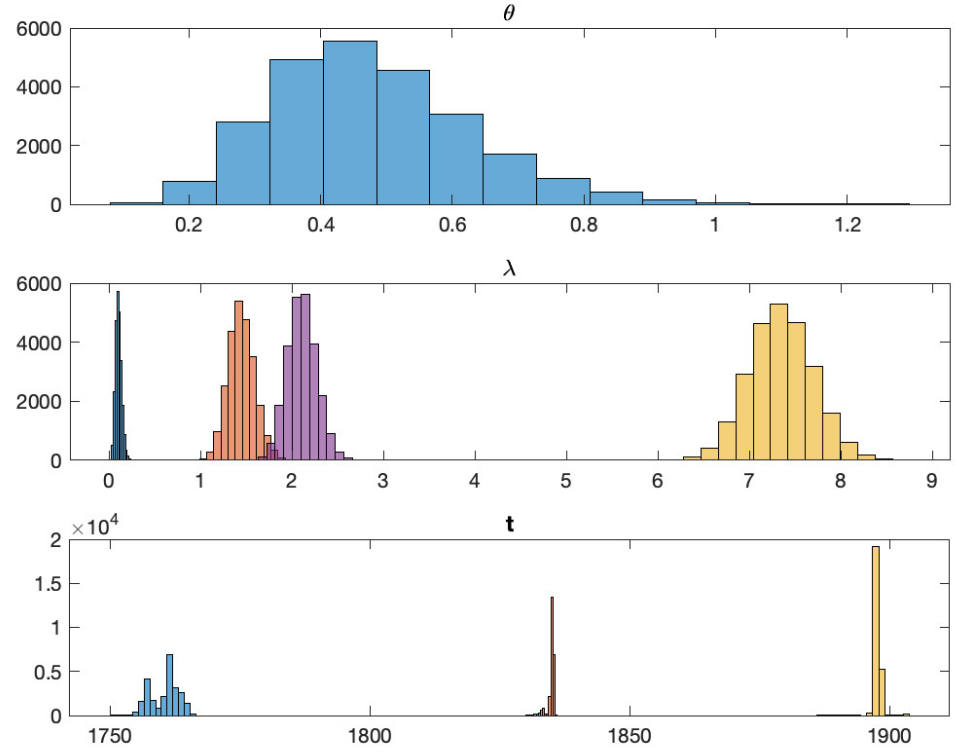


Fig. 4: Posteriors when Ψ is set to 10.

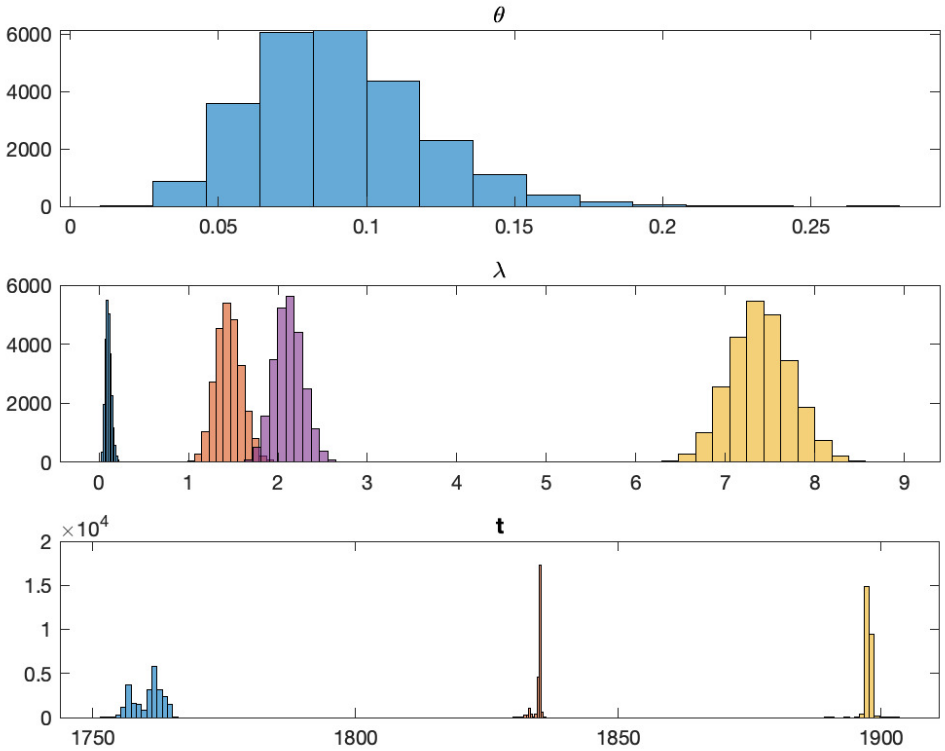


Fig. 5: Posteriors when Ψ is set to 100.

The faster the "forgetting" (lower covariance for larger lags), the lower the variance of the estimate. The rate of forgetting is called mixing, and clearly, higher mixing is better. To study the mixing, we estimate and investigate the empirical covariance function for the used samples (not the burn-in). We use Ψ set to 5, 5000 burn-in samples, 25000 samples to average over, and 3 breakpoints. In Figures 10 through 13 (in Appendix IV) we see the covariance functions for $\rho = 0.001, 0.01, 0.1, 1$. Estimating the actual variance is not worthwhile here, as we cannot capture the long-term dependencies very well with so few samples, but it is definitely possible to make a visual inspection. We see that the mixings of the parameters depend differently on the choice of ρ. The θ parameter seems to be the least affected by ρ, however, it appears to show the most white-noise-like covariance function at 0.01 and 0.1. λ_1 and λ_2 seem to have the best mixing at 0.1, while λ_3 and λ_4 seem to have the best mixing at 0.01. t_1 and t_2 both show the best mixing at 0.1, but t_3 shows very bad mixing here, it has the best mixing at 0.01. This mixing "quirk" seems like it might depend on the run of the algorithm, but it was in fact consistent between runs. With this in mind, it seems likely that the best overall mixing is found between 0.01 and 0.1.

Continuing the analysis, Figures 14 through 17 (also in Appendix IV) show the sampling process for these values of ρ. Here, we see the two problems we are trying to balance with a good choice of ρ. If ρ is too small, many movements in $\mathbf{t}$ are accepted, but they are so small that it takes a lot of movements to get anywhere, making the chain take a long time to explore the posterior space. However, if ρ is too large, very few movements are accepted, so even though the movement is large (meaning, quite uncorrelated points before and after the movement) it takes a lot of iterations to get any movement at all, meaning that two subsequent points are likely to be the same, thus the covariance is strong. With a balanced choice of ρ, we get a good number of movements, which are quite large (so between quite uncorrelated points). Comparing $\rho = 0.01$ and $\rho = 0.1$ here, we see that when it is 0.1, t_3 jumps between two values, while at 0.01 it seems to stick to a single value. This could indicate that the mixing is actually better when $\rho = 0.1$, and that the seemingly better mixing at 0.01 only appears because it is exploring a small groove in the posterior space where subsequent samples are quite independent. We also see some movement in t_1 for $\rho = 1$ that is not seen in any other case. This might indicate that, actually, only $\rho = 1$ reaches all parts of the parameter space. The estimated covariance functions are seemingly not good estimates of the actual covariances. It is therefore quite difficult to conclude what ρ has the best overall mixing, but it makes it clear that ρ has a large effect on the mixing. It is also likely that for other values of d, we may see different behaviors. It seems that using a uniform distribution as proposal distribution may not be the best choice, as this limits the largest possible "jumps" to a certain size, which may mean that the chain cannot easily get

out of some grooves. Using, for example, a normal distribution as the proposal distribution, may mean that the mixing is less sensitive to the spread parameter (in this case, the standard deviation).

PART 2: PARAMETRIC BOOTSTRAP FOR THE 100-YEAR ATLANTIC WAVE

a)

We are given the Gumbel distribution function in eq 21

$$F(x;\mu,\beta) = \exp\left(-\exp\left(-\frac{x-\mu}{\beta}\right)\right), \quad x \in \mathbb{R}, \tag{21}$$

Attempting to invert the function, we assert

$$F(F^{-1}(u;\mu,\beta)) = u, \tag{22}$$

and find

$$\begin{aligned} u = \exp\left(-\exp\left(-\frac{F^{-1}(u;\mu,\beta)-\mu}{\beta}\right)\right) \\ \iff \\ \ln(-\ln(u)) = -\frac{F^{-1}(u;\mu,\beta)-\mu}{\beta} \\ \iff \\ \mu - \beta\ln(-\ln(u)) = F^{-1}(u;\mu,\beta), \end{aligned} \tag{23}$$

so

$$F^{-1}(u;\mu,\beta) = \mu - \beta\ln(-\ln(u)). \tag{24}$$

In eq 24 we have obtained the inverse Gumbel distribution function.

b)

We are tasked to find a two-sided 99% confidence interval of the parameters of the Gumbel distribution function, utilising a bootstrapping method for estimating confidence intervals of parameters of the proposed parametric model: assuming a Gumbel distribution of the given time series. Bootstrap in Monte Carlo is a technique utilized for approximating the distribution of a statistic $\hat{\tau}$ and in this task it will be used to find a confidence interval for both $\hat{\mu}$ and $\hat{\beta}$.

It is always reasonable first step to have a naked look at the data, the measured time series data Y that is assumed to $Y \sim$ Gumbell(μ,β) as previously mentioned and it is visualized in fig 7.

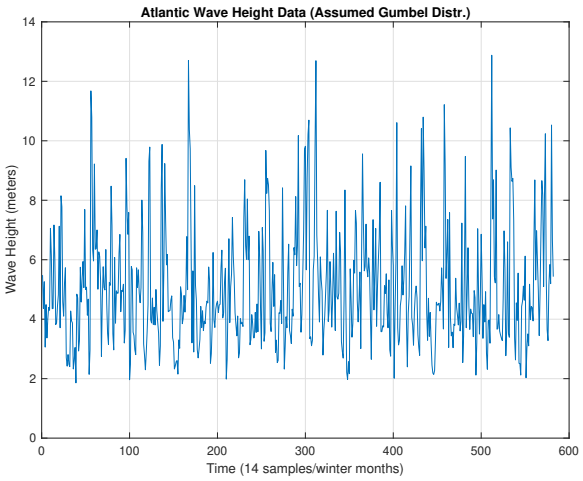


Fig. 7: Atlantic data provided in the task with highest recorded wave height (14 samples every month during winter months). The data is assumed to generated by a Gumbel distribution.

It is already given in the task a function (that we will not delve into further) that makes point-wise estimates of the parameters μ and β out of a time series named `est_gumbel`. In other words, we are essentially given the original observed statistics $\hat{\mu}$ and $\hat{\beta}$ from this Matlab function. Now it remains to derive a confidence interval for these.

A distribution of the statistics is produced by repeatedly sampling with replacement from the observed data and generating an empirical distribution function for each sample (using the inverse Gumbel distribution function derived earlier). This process allows for the calculation of new statistic values (called bootstrap estimates) for each sample (called bootstrap samples), which collectively serve as an approximation of the statistic's sampling distribution. Essentially, bootstrapping enables in this way the provided data to be self-sufficient in the analysis of statistical inference.

Specifically, let $Y_1^*, Y_2^*, \ldots, Y_N^*$ be N bootstrap samples (where we choose to sample $N = 10000$ number of times) obtained by drawing with replacement from the original time series Y. The bootstrap samples are simulated by the inverse Gumbel distribution function $F^{-1}(u;\hat{\mu},\hat{\beta})$ where $u \sim$ Uniform$(0,1)$. For each bootstrap sample, we calculate new observations of μ and β, the bootstrap replicate of the statistics which we will denote by $\hat{\mu}_n^*$ and $\hat{\beta}_n^*$ (where $n = 1, ..., N$) and we calculated them by using `est_gumbel` on all $Y_1^*, Y_2^*, \ldots, Y_N^*$ and these are visualized in the histograms seen in fig 8. We can see that a distribution has formed around the originally observed statistics $\hat{\mu}$ and $\hat{\beta}$. To calculate the appropriate percentiles from these bootstrap distributions we compute the sorted differences $\Delta\hat{\mu}_n^* = \hat{\mu}_n^* - \hat{\mu}$ and $\Delta\hat{\beta}_n^* = \hat{\beta}_n^* - \hat{\beta}$ and find the differences corresponding to the lower and upper bound of the specified confidence level $1-\alpha = 99\%$ (where $\alpha = 1\%$) to add these to $\hat{\mu}$ $\hat{\beta}$ and construct the desired confidence interval. [3]. The following confidence interval was obtained seen in table I and also visualized in fig 8. It could also be said that in the code implementation we set a random number generator seed $rng(10)$.

TABLE I: 99% Confidence Intervals for Gumbel Parameters. Number of bootstrap samples N = 10000.

Parameter	Lower Bound	Estimate	Upper Bound
μ	3.9807	4.1477	4.3075
β	1.3612	1.4858	1.6084

c)

In contrast to the previous task we are made to provide a one-sided upper bounded estimate of 99% confidence of the 100-year return value.

A hundred years would mean a total of $T = 3 * 14 * 100 = 4200$ observations (three winter months, 14 observations per month, 100 years). The wave height, the Tth return value is given by $F^{-1}\left(1 - \frac{1}{T}; \mu, \beta\right)$, where the inverse function is the same inverse Gumbel distribution function used earlier. This gives us an original observed estimate of the height of the wave based on the statistics $\hat{\mu}, \hat{\beta}$ by $\hat{h}_{wave} = F^{-1}\left(1 - \frac{1}{T}; \hat{\mu}, \hat{\beta}\right)$

We can utilize the $N = 10000$ bootstrap samples of the statistics $\hat{\mu}^*$ and $\hat{\beta}^*$ from task b) in a new bootstrap to generate $N = 10000$ new boot samples of replicas of the estimates of the wave height $\hat{h}^*_{wave,i} = F^{-1}\left(1 - \frac{1}{T}; \hat{\mu}^*_i, \hat{\beta}^*_i\right)$, where $i = 1, ..., N$ and those generated are visualised in figure 9. Thereafter we repeat the step of sorting the array containing the difference of all saved boot replicas $\Delta\hat{h}^*_{wave} = \hat{h}^*_{wave} - \hat{h}_{wave}$.

The final result of the one sided parametric bootstrap with 99% confidence confidence of the 100-year return value is seen in table II. Note that ”estimate” is simply $\hat{h}_{wave}$.

TABLE II: One-Sided Upper 99% Confidence Interval for the 100-Year Return Value. Number of bootstrap samples N = 10000.

Parameter	Lower Bound	Estimate	Upper Bound
100-Year Return Value	0	16.5436	17.6304

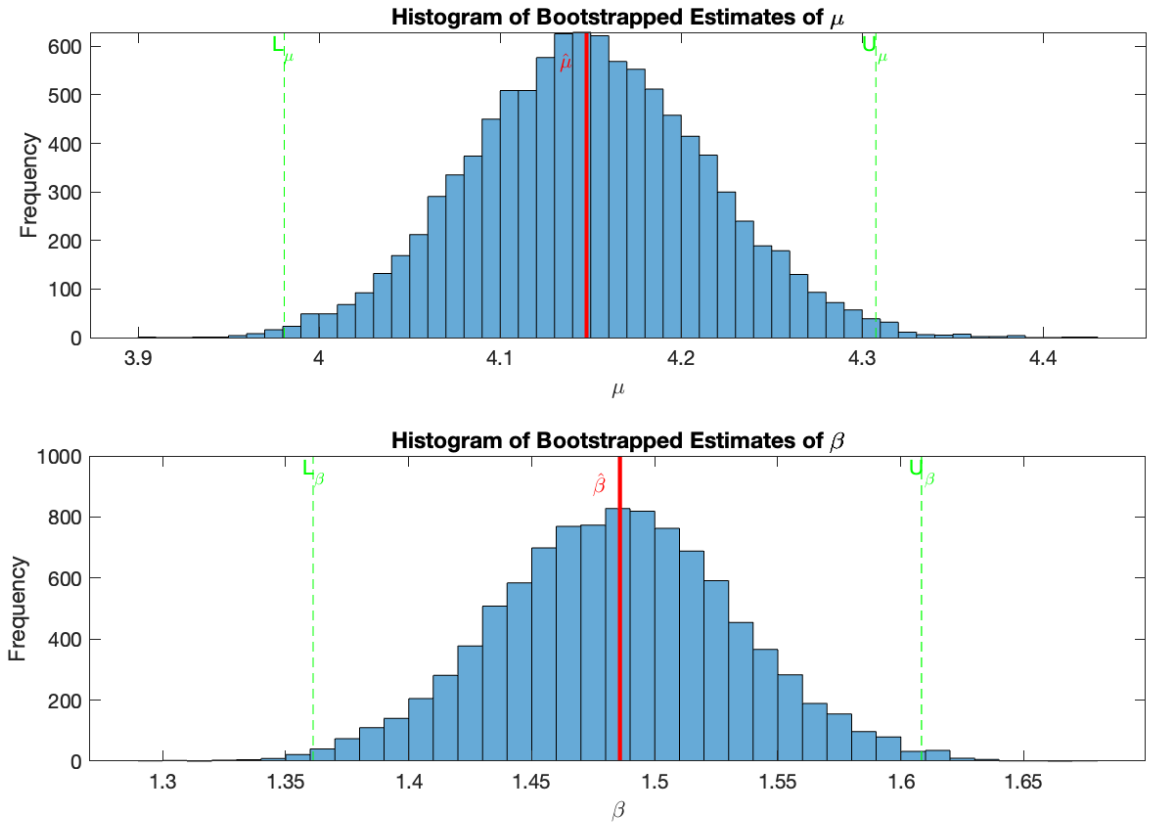


Fig. 8: Histogram of bootstrap samples and estimates with the original statistics value upper and lower bound 99% confidence interval. The top histogram has 53 bins and the lower histogram has 39 bins. Number of bootstrap samples N = 10000.

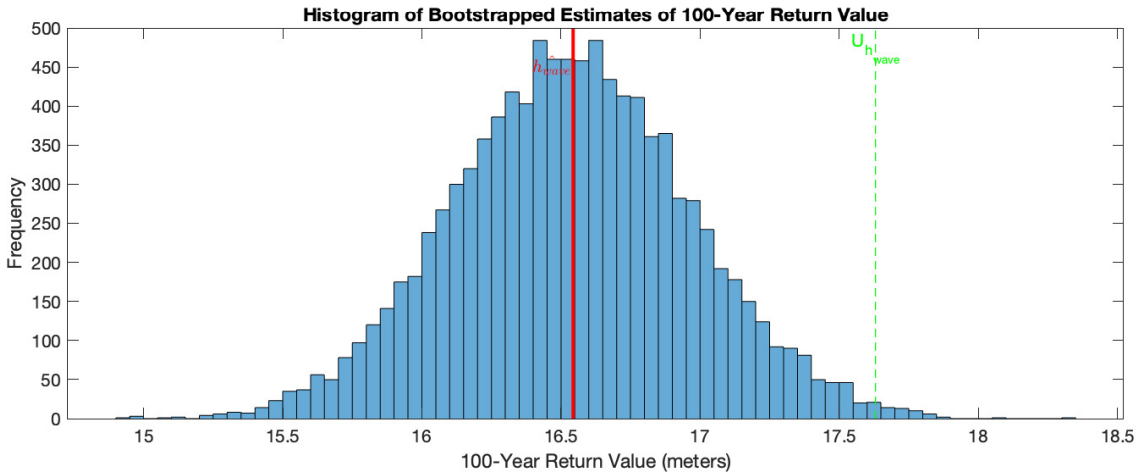


Fig. 9: Histogram of bootstrap samples and estimates with the original statistic and upper bound in 99% one-sided upper confidence bound. The histogram has 69 bins. Number of bootstrap samples N = 10000.

I. References

[1] Magnus Wiktorsson. *Lecture 11, slide 13*. 2024.
[2] Magnus Wiktorsson. *Lecture 11, slide 7*. 2024.
[3] Magnus Wiktorsson. *Lecture 13, slide 33*. 2024.

II. Appendix

III. Derivation of measurement density

The derivation here has not been checked thoroughly enough, so we have trusted the provided density. Anyway, the density of the accidents given all parameters would seem to be found as

$$f(\tau|\lambda,\mathbf{t},\theta)=\prod_{i=1}^{d}\mathbb{P}(n_i(\tau)|\lambda_i,t_{i+1}-t_i), \tag{25}$$

where $n_i(\tau)$ is the number of accidents in the time span $t_{i+1}-t_i$. Knowing that the number of accidents in each time span has a Poisson distribution with parameter $\lambda_i(t_{i+1}-t_i)$, we see that the probability distribution is

$$\begin{aligned} f(\tau|\lambda,\mathbf{t},\theta)=\prod_{i=1}^{d}\frac{[\lambda_i(t_{i+1}-t_i)]^{n_i(\tau)}}{n_i(\tau)!}e^{-\lambda_i(t_{i+1}-t_i)}= \\ \prod_{i=1}^{d}\frac{(t_{i+1}-t_i)^{n_i(\tau)}}{n_i(\tau)!}\lambda_i^{n_i(\tau)}e^{-\lambda_i(t_{i+1}-t_i)}. \end{aligned} \tag{26}$$

IV. Figures from 1e)

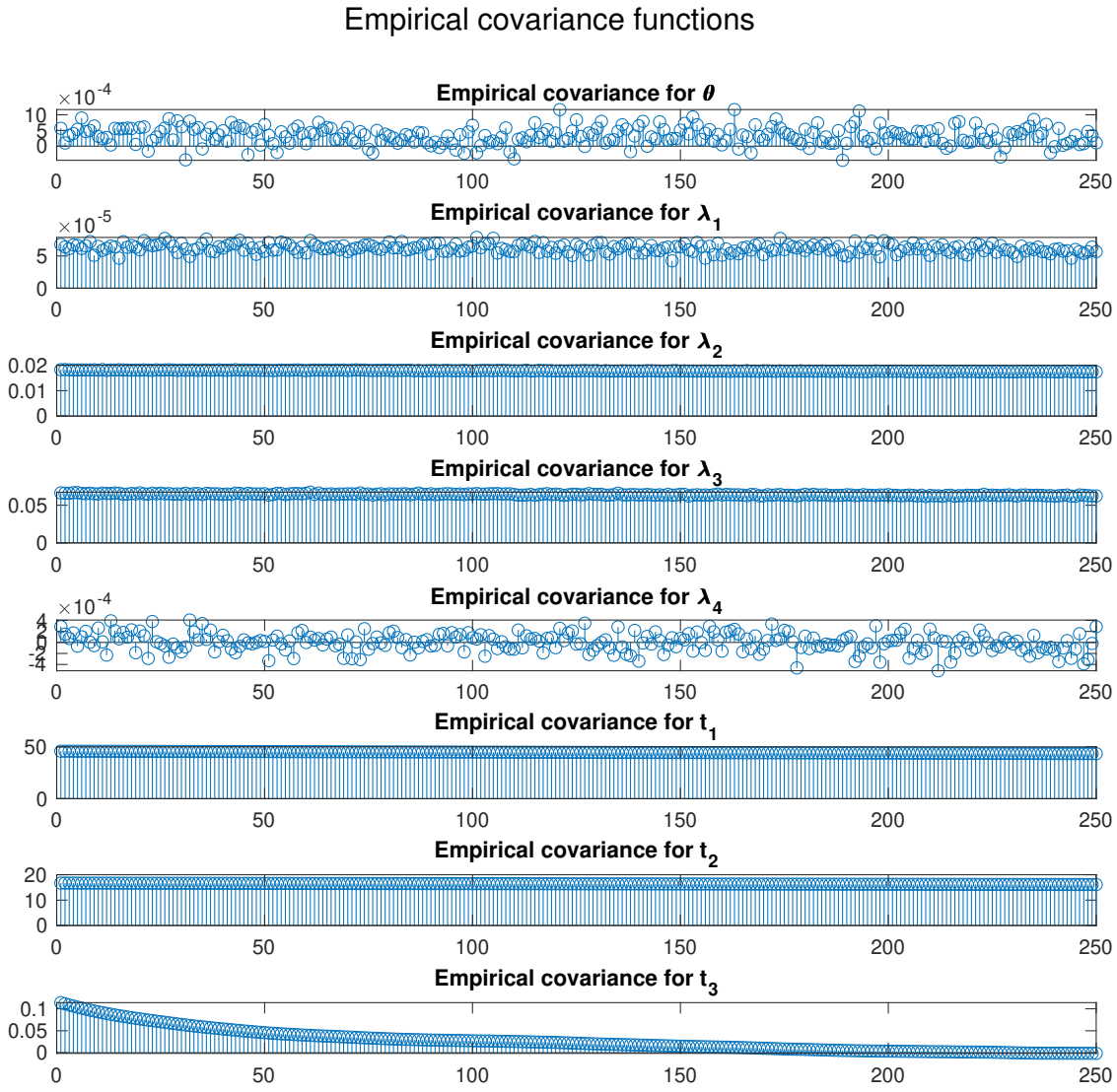


Fig. 10: Covariance function when ρ is set to 0.001.

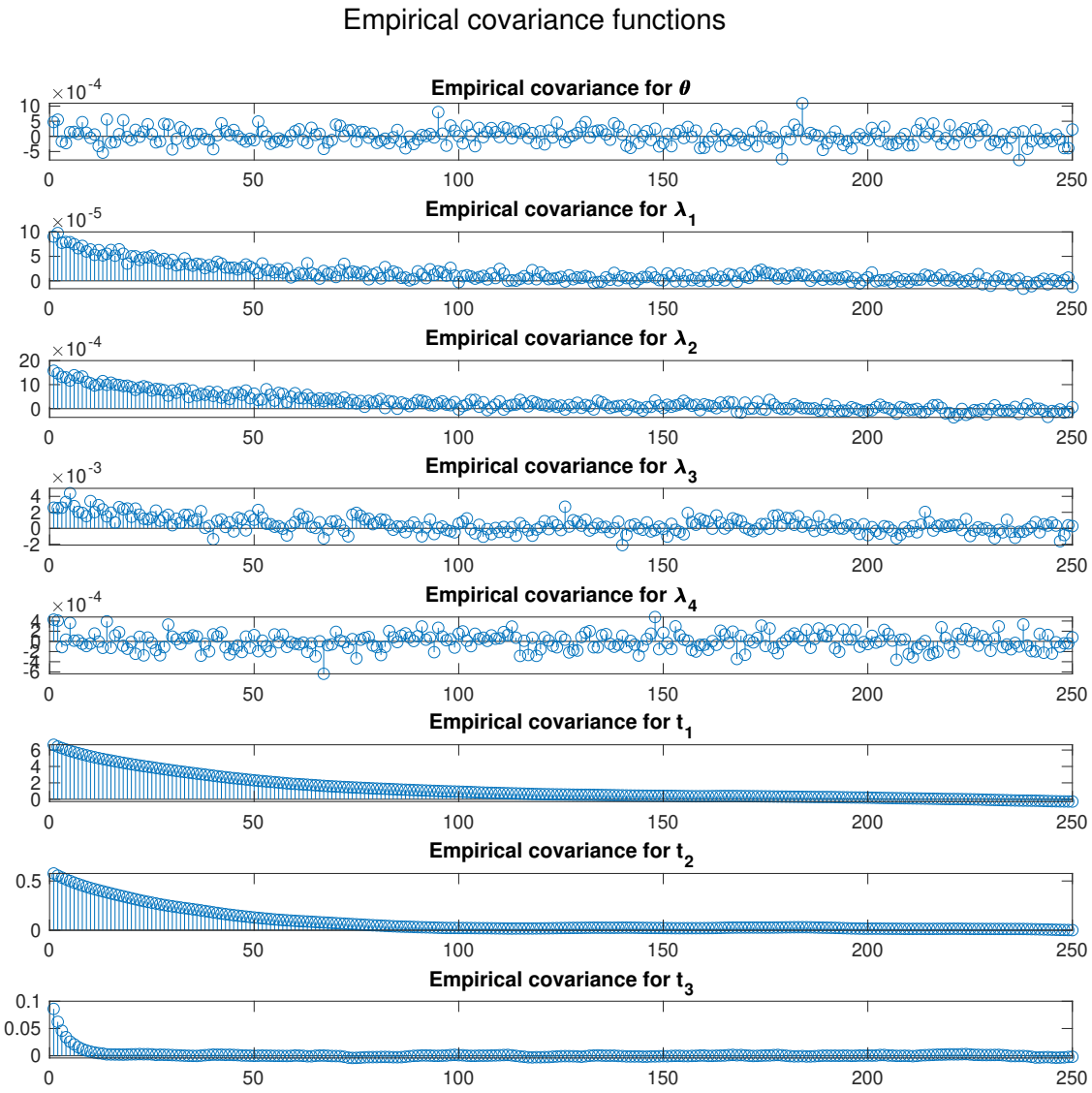


Fig. 11: Covariance function when ρ is set to 0.01.

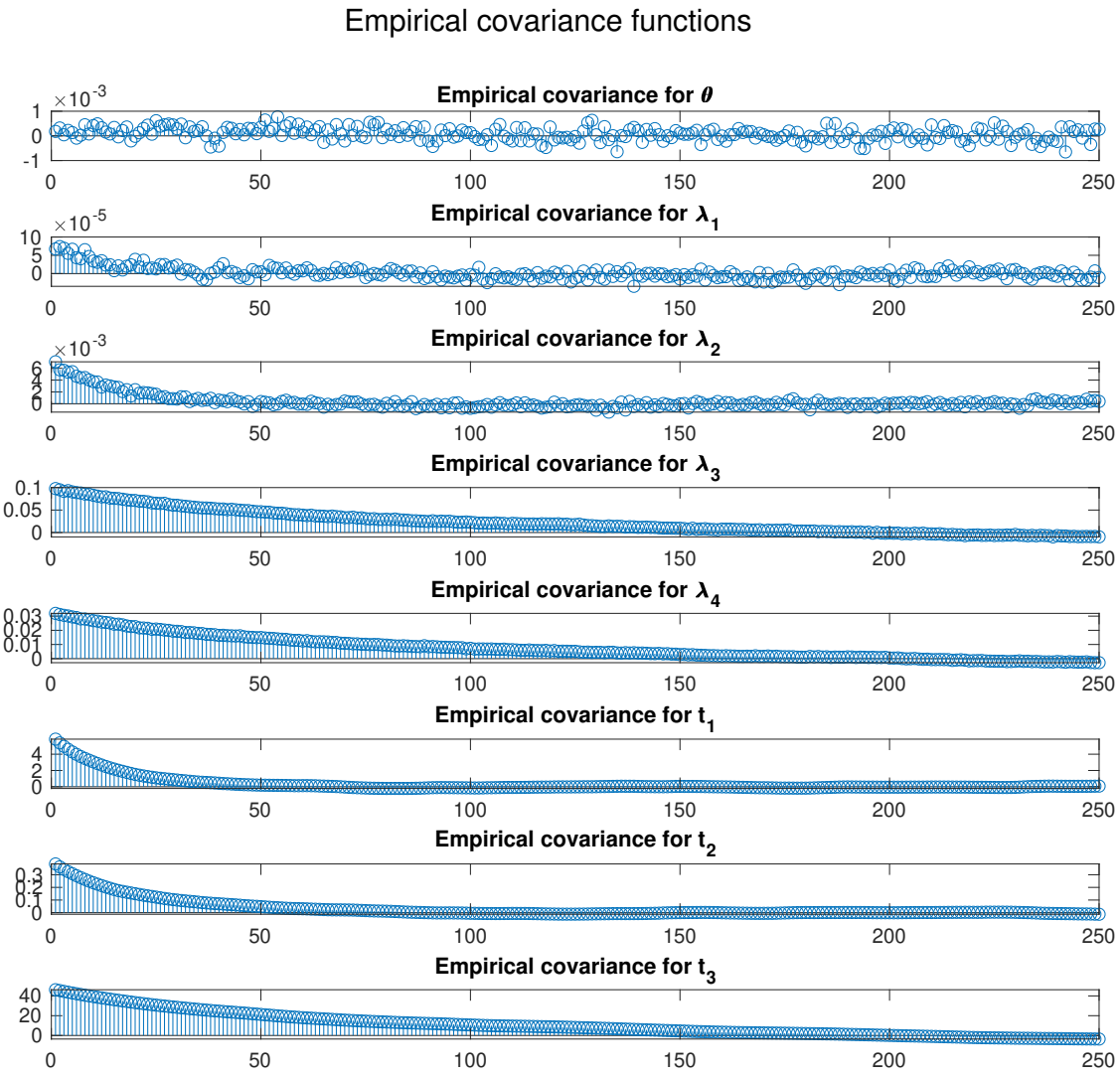


Fig. 12: Covariance function when ρ is set to 0.1.

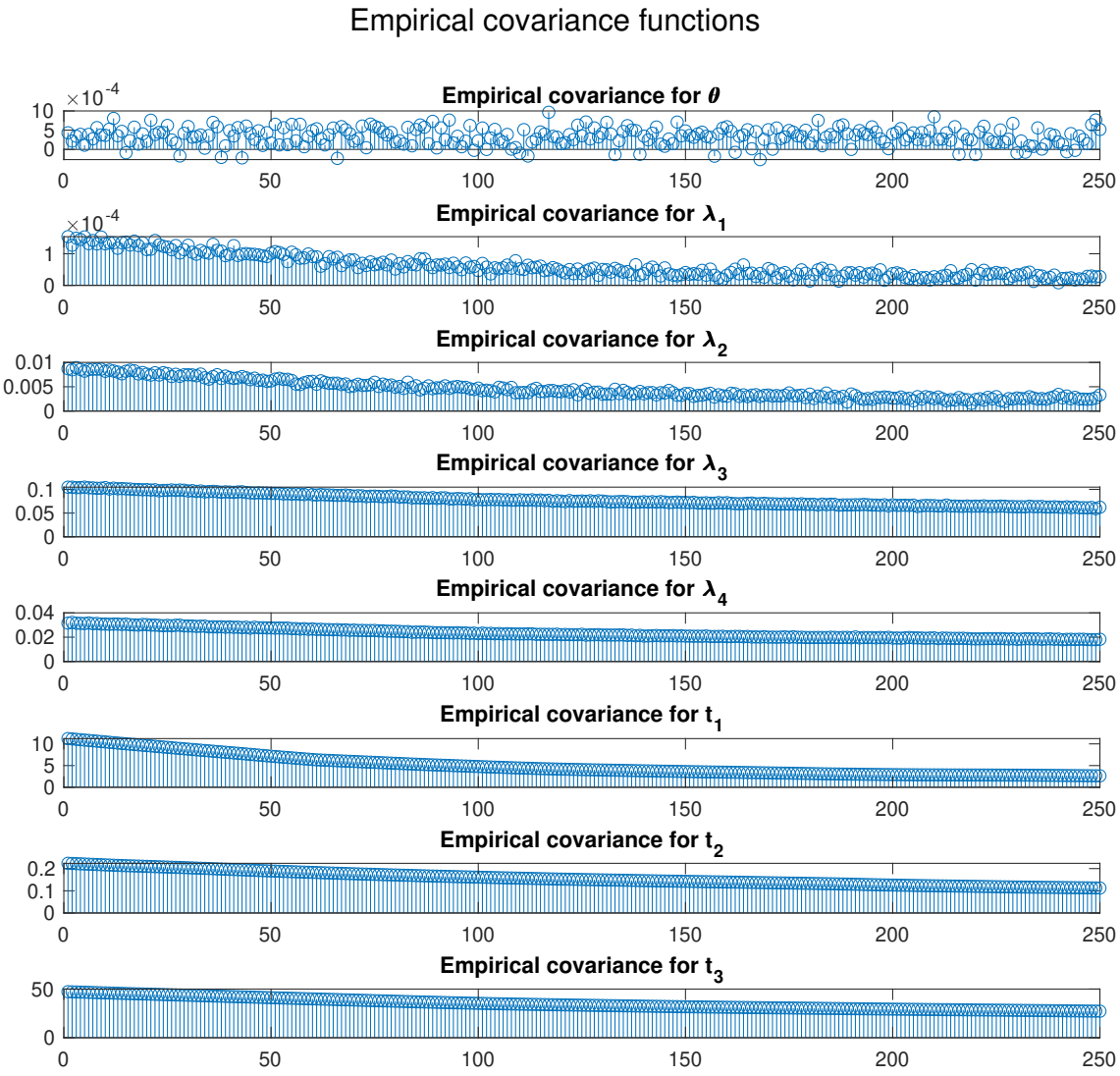


Fig. 13: Covariance function when ρ is set to 1.

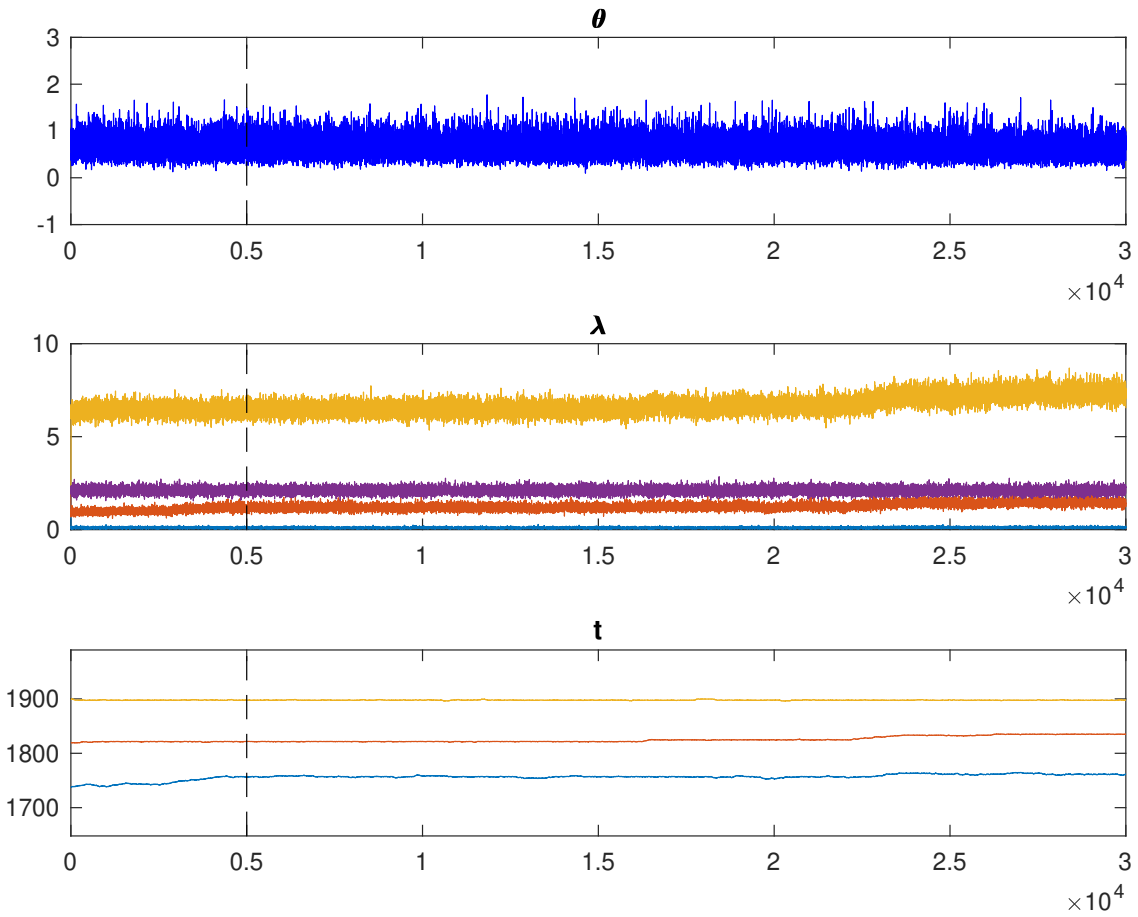


Fig. 14: Sampling process when ρ is set to 0.001.

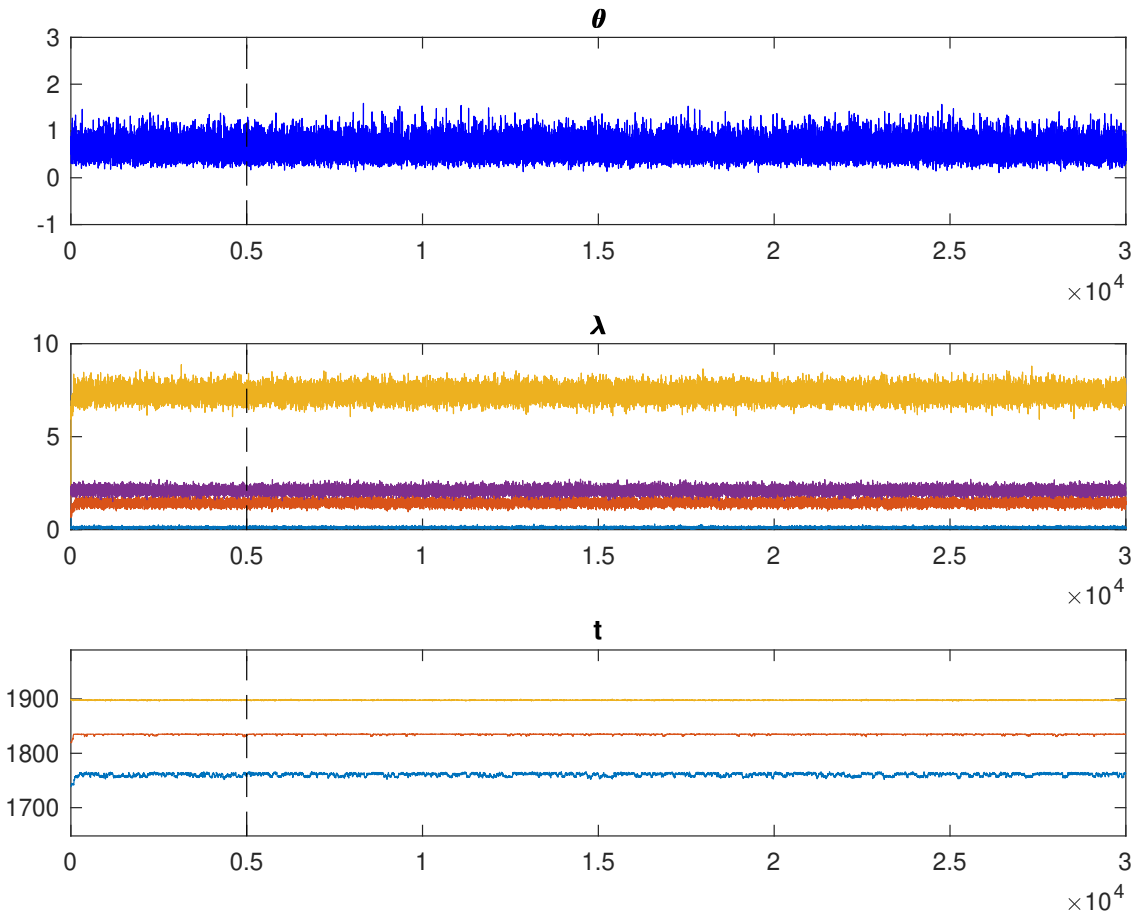


Fig. 15: Sampling process when ρ is set to 0.01.

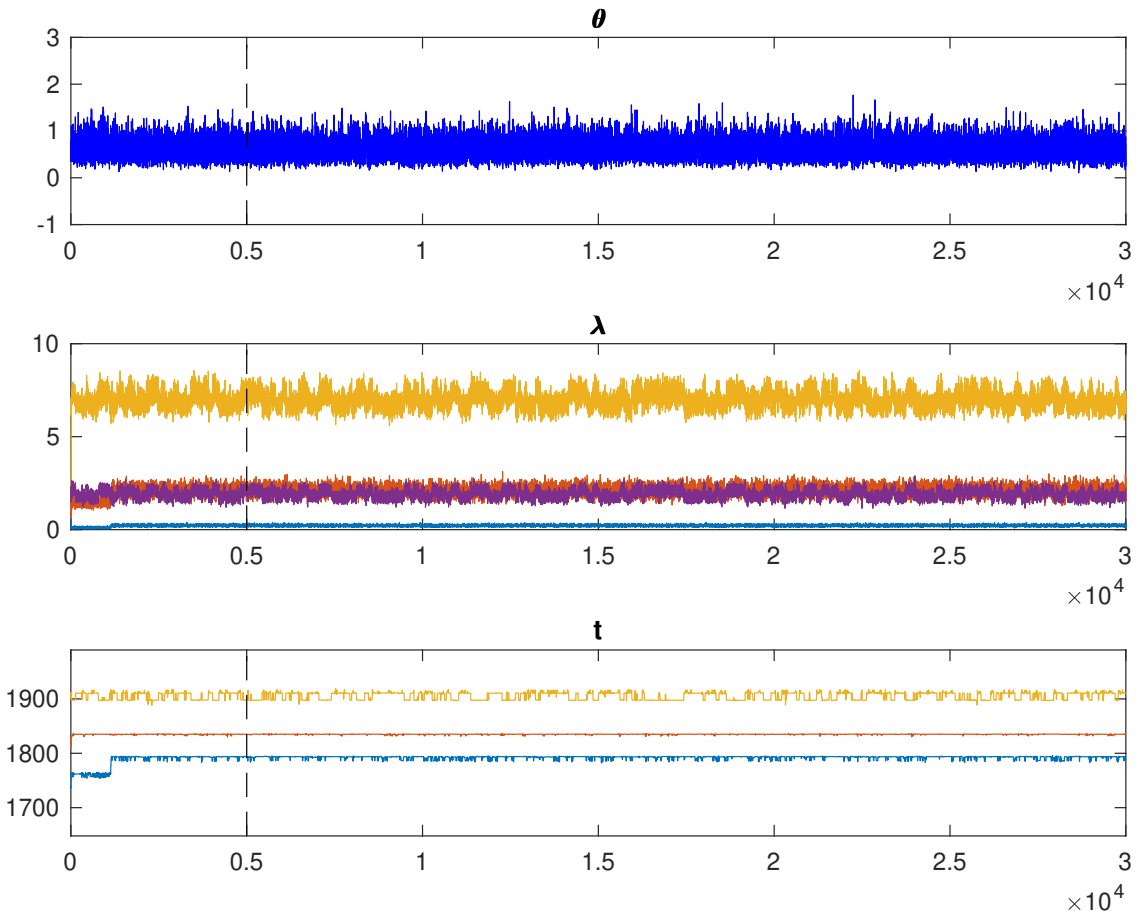


Fig. 16: Sampling process when ρ is set to 0.1.

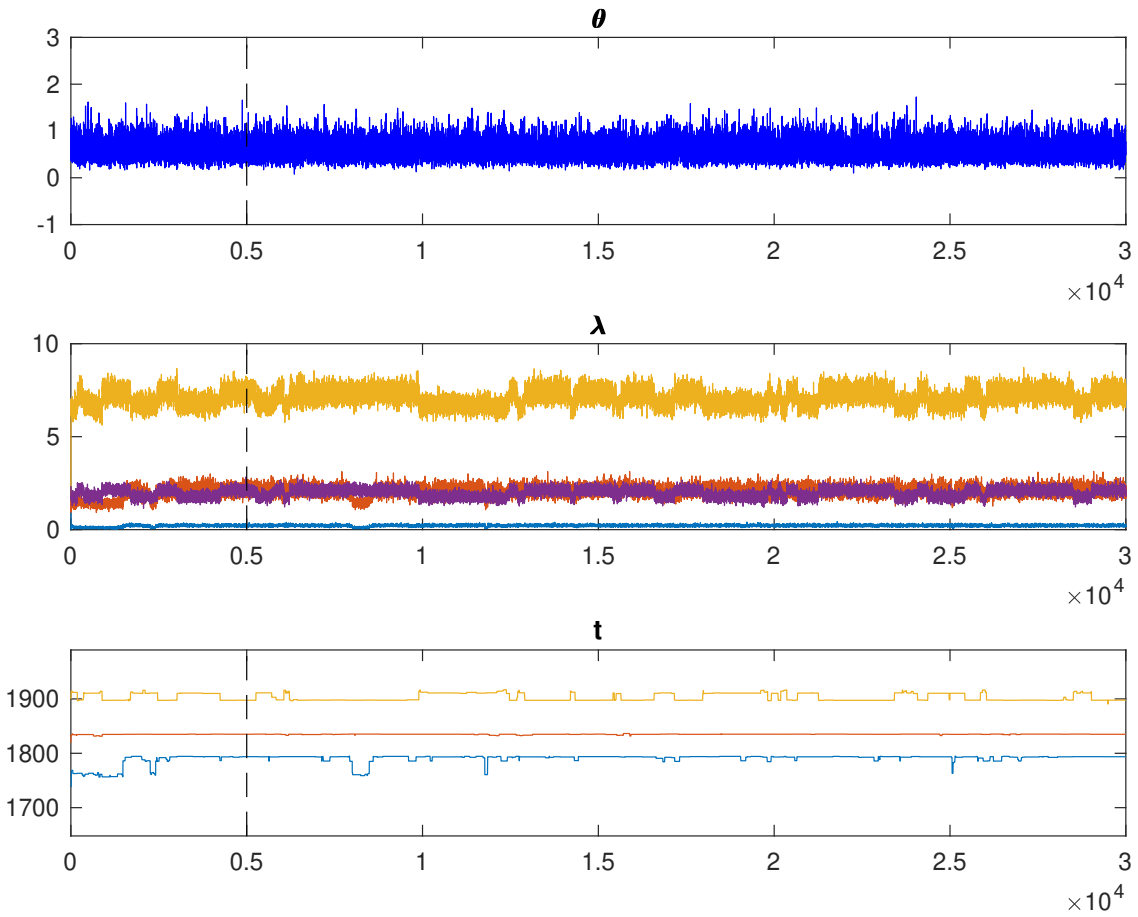


Fig. 17: Sampling process when ρ is set to 1.