Detecting Changes in Slope With an L_0 Penalty

Bastien LE CHENADEC bastien.le-chenadec@eleves.enpc.fr
Sofiane EZZEHI sofiane.ezzehi@eleves.enpc.fr

January 9, 2024

What is expected for these mini-projects? *The goal of the exercise is to read (and understand) a research article, implement it (or find an implementation), test it on real data and comment on the results obtained. Depending on the articles, the task will not always be the same: some articles are more theoretical or complex, others are in the direct line of the course, etc... It is therefore important to balance the exercise according to the article. For example, if you have reused an existing implementation, it is obvious that you will have to develop in a more detailed way the analysis of the results, the influence of the parameters etc... Do not hesitate to contact us by email if you wish to be guided.*

The report *The report must be at most FIVE pages and use this template (excluding references). If needed, additional images and tables can be put in Appendix, but must be discussed in the main document. The report must contain a precise description of the work done, a description of the method, and the results of your tests. Please do not include source code! The report must clearly show the elements that you have done yourself and those that you have reused only, as well as the distribution of tasks within the team (see detailed plan below.)*

The source code *In addition to this report, you will have to send us a Python notebook allowing to launch the code and to test it on data. For the data, you can find it on standard sites like Kaggle, or the site https://timeseriesclassification.com/ which contains a lot of signals!*

The oral presentations *They will last 10 minutes followed by 5 minutes of questions. The plan of the defense is the same as the one of the report: presentation of the work done, description of the method and analysis of the results.*

Deadlines *Two sessions will be available :*

- ***Session 1***
  - *Deadline for report: December 18th (23:59)*
  - *Oral presentations: December 20th and 22th (precise times TBA)*
- ***Session 2***
  - *Deadline for report: January 9th (23:59)*
  - *Oral presentations: January, 11th and 12th (precise times TBA)*

1 Introduction and contributions

The Introduction section (indicative length : less than 1 page) should detail the scientific context of the article you chose, as well as the task that you want to solve (especially if you apply it on novel data). ***The last paragraph of the introduction must contain the following information****:*

- *Repartition of work between the two students*
- *Use of available source code or not, percentage of the source code that has been reused, etc.*
- *Use of existing experiments or new experiments (e.g. test of the influence of parameter that was not conducted in the original article, application of the method on a novel task/data set etc.)*
- *Improvement on the original method (e.g. new pre/post processing steps, grid search for optimal parameters etc.)*

2 Method

The Method section (indicative length : 1 to 2 pages) should describe the mathematical aspects of the method in a summarized manner. Only the main steps that are useful for understanding should be highlighted. If relevant, some details on implementation can be provided (but only marginally).

A word on notations : if we have a sequence $x = x_1, \dots, x_n$, we denote $x[s : t]$ the subsequence $x_s, \dots, x_t$.

2.1 Model

Let $y = y_1, \dots, y_n$ be successive data points in time. The goal of this method is to find m changepoints $\tau_1, \dots, \tau_m$ such that the data is divided in $m + 1$ segments. We let $\tau_0 = 0$ and $\tau_{m+1} = n$; the segment j is then $y_{\tau_{j-1}+1}, \dots, y_{\tau_j}$.

The method considers a piecewise linear model to fit the data. We denote ϕ_{τ_j} the value taken by the model at time τ_j. Under these assumptions, the model writes :

$$\forall 0 \leq i \leq m, \forall \tau_i + 1 \leq t \leq \tau_{i+1}, \quad Y_t = \phi_{\tau_i} + \frac{\phi_{\tau_{i+1}} - \phi_{\tau_i}}{\tau_{i+1} - \tau_i}(t - \tau_i) + Z_t \tag{1}$$

where $(Z_t)_{1 \leq t \leq n}$ is assumed to be a gaussian white noise with variance σ^2.

2.2 Penalized cost approach

The method uses a penalized cost approach to find the changepoints. The cost function is a squarred error loss, penalized by a term that depends on the segments length, and an L_0 penalty on the number of changepoints. The cost function naturally writes :

$$\sum_{i=0}^{m} \left[\frac{1}{\sigma^2} \sum_{t=\tau_i+1}^{\tau_{i+1}} \left(y_t - \phi_{\tau_i} - \frac{\phi_{\tau_{i+1}} - \phi_{\tau_i}}{\tau_{i+1} - \tau_i}(t - \tau_i) \right)^2 + h(\tau_{i+1} - \tau_i) \right] + \beta m \tag{2}$$

By introducing a segment cost, for fitting a segment that takes the value ϕ at time s and ψ at time t, on the data $y_{s+1}, \dots, y_t$, that writes :

$$\mathcal{C}(y[s + 1 : t], \phi, \psi) = \frac{1}{\sigma^2} \sum_{j=s+1}^{t} \left(y_j - \phi - \frac{\psi - \phi}{t - s}(j - s) \right)^2 \tag{3}$$

the penalized cost function can be rewritten as :

$$\min_{\substack{m \\ \tau_1,\dots,\tau_m \\ \phi_0,\dots,\phi_{m+1}}} \beta m + \sum_{i=0}^{m} \mathcal{C}(y[\tau_i + 1 : \tau_{i+1}], \phi_{\tau_i}, \phi_{\tau_{i+1}}) + h(\tau_{i+1} - \tau_i) \tag{4}$$

2.3 Dynamic programming

We want to solve the problem (4) by dynamic programming. Compared to other changepoints detection methods, the difficulty lies in the continuity constraint between segments. Thus the authors introduce $f^t(\phi)$ as the minimum cost for segmenting $y_1, \dots, y_t$ in $k+1$ segments conditionally on the model taking the value ϕ at time t :

$$\begin{aligned} f^t(\phi) = \min_{\substack{\tau_1,\dots,\tau_k \\ \phi_0,\dots,\phi_k}} & \sum_{i=0}^{k-1} \mathcal{C}(y[\tau_i + 1 : \tau_{i+1}], \phi_{\tau_i}, \phi_{\tau_{i+1}}) + h(\tau_{i+1} - \tau_i) \\ & + \mathcal{C}(y[\tau_k + 1 : t], \phi_{\tau_k}, \phi) + h(t - \tau_k) + \beta k \end{aligned} \tag{5}$$

which they are able to express recursively :

$$f^t(\phi) = \min_{\phi', s} f^s(\phi') + \mathcal{C}(y[s+1:t], \phi', \phi) + h(t-s) + \beta \tag{6}$$

They also introduce $f^t_{\boldsymbol{\tau}}(\phi)$ which is the minimum cost for segmenting $y_1, \dots, y_t$ in $k+1$ segments conditionally on the model taking the value ϕ at time t and the changepoints being $\boldsymbol{\tau} = (\tau_0, \tau_1, \dots, \tau_k, \tau_{k+1})$. This quantity is a quadratic polynomial in ϕ and the authors are able to express its coefficients recursively. All of this allows them to compute $f^n(\phi)$ in $\mathcal{O}(n!/(n-m)!)$.

2.4 Pruning

Given the high complexity of the dynamic programming algorithm (exploring the space of all possible changepoints), the authors introduce multiple pruning techniques to reduce the complexity.

Functional pruning The authors introduce $\mathcal{T}_t$ the set of all possible changepoint vectors at time t, and $\mathcal{T}_t^* = \left\{ \boldsymbol{\tau} \in \mathcal{T}_t \,\middle|\, \exists \phi \in \mathbb{R}, f^t(\phi) = f^t_{\boldsymbol{\tau}}(\phi) \right\}$ the set of changepoint vectors at time t that are optimal for some ϕ.

Theorem 1. *Let $\boldsymbol{\tau} \in \mathcal{T}_s$ such that $\boldsymbol{\tau} \notin \mathcal{T}_s^*$. Then $\boldsymbol{\tau} \notin \mathcal{T}_t^*$ for all $t > s$.*

This theorem allows us to only explore $\hat{\mathcal{T}}_t = \left\{ (\boldsymbol{\tau}, s) \,\middle|\, 0 \le s \le t-1, \boldsymbol{\tau} \in \mathcal{T}_s^* \right\}$ when computing $f^t(\phi)$.

Inequality based pruning Let $K = 2\beta + h(1) + h(n)$. Suppose that h is non-negative and non-decreasing.

Theorem 2. *Let $\tau \in \mathcal{T}_s$ such that :*

$$\min_{\phi} f^s_{\boldsymbol{\tau}}(\phi) > K + \min_{\phi'} f^s(\phi')$$

Then $\boldsymbol{\tau} \notin \mathcal{T}_t^*$ for all $t > s$.

This theorem allows to further prune the set $\hat{\mathcal{T}}_t$.

2.5 Algorithm

The two difficulties of the implementation are storing the coefficients of the polynomials $f_{\tau}^{t}(\phi)$, and implementing the pruning techniques. The authors provided relatively detailed pseudo-code for the algorithm, but we still found it difficult to implement.

The goal of this first algorithm is to only keep segmentations that are optimal for some values of ϕ. Segmentation costs are polynomials, so this algorithm starts from the optimum segmentation for $\phi = -\infty$ and iteratively finds the next optimum segmentation by looking at the points where the polynoms intersect.

Algorithm 1 Functional pruning algorithm

Input: Candidate segmentations $\hat{\mathcal{T}}_t$, costs $f_{\tau}^{t}(\phi)$ for $\tau \in \hat{\mathcal{T}}_t$.
Initialize: $\phi_{\text{curr}} = -\infty \quad \tau_{curr} = \arg\min_{\tau \in \hat{\mathcal{T}}_t} \left[f_{\tau}^{t}(\phi_{curr}) \right] \quad \mathcal{T}_{temp} = \hat{\mathcal{T}}_t \setminus \{\tau_{curr}\} \quad \mathcal{T}_t^* = \{\tau_{curr}\}$
while $\mathcal{T}_{temp} \neq \emptyset$ **do**
    for $\tau \in \mathcal{T}_{temp}$ **do**
        $x_{\tau} = \min \left\{ \phi : f_{\tau}^{t}(\phi) - f_{\tau_{curr}}^{t}(\phi) = 0 \ \ \& \ \ \phi > \phi_{curr} \right\}$
        if $x_{\tau} = \emptyset$ **then**
            $\mathcal{T}_{temp} = \mathcal{T}_{temp} \setminus \{\tau\}$
    $\tau_{curr} = \arg\min_{\tau \in \mathcal{T}_{temp}} (x_{\tau})$
    $\phi_{curr} = x_{\tau_{curr}}$
    $\mathcal{T}_t^* = \mathcal{T}_t^* \cup \{\tau_{curr}\}$
Output: $\mathcal{T}_t^*$

Then we can define the full algorithm that uses both pruning techniques.

Algorithm 2 CPOP

Input: Data $y = y_1, \ldots, y_n$, penalty β and segment cost h.
Initialize: $\hat{\mathcal{T}}_1 = \{\{0\}\}$
for $t = 1, \ldots, n$ **do** ▷ Cost computation
    for $\tau \in \hat{\mathcal{T}}_t$ **do**
        if $\tau = \{0\}$ **then** ▷ No changepoint
            $f_{\tau}^{t}(\phi) = \min_{\phi'} \mathcal{C}(y[1:t], \phi', \phi) + h(t)$
        else ▷ Compute recursively
            $f_{\tau}^{t}(\phi) = \min_{\phi'} \left(f_{\tau_1, \ldots, \tau_{k-1}}^{\tau_k}(\phi') + \mathcal{C}(y[\tau_k + 1 : t], \phi', \phi) + h(t - \tau_k) + \beta \right)$
    $\mathcal{T}_t^* = \text{functional\_pruning}(\hat{\mathcal{T}}_t, f^t)$
    $\hat{\mathcal{T}}_{t+1} = \left\{ \tau \in \hat{\mathcal{T}}_t \middle| \min_{\phi} f_{\tau}^{t}(\phi) \leq \min_{\phi', \tau'} f_{\tau'}^{t}(\phi') + K \right\} \cup \left\{ (\tau, t) \middle| \tau \in \mathcal{T}_t^* \right\}$
Output: $\arg\min_{\tau \in \hat{\mathcal{T}}_n} [\min_{\phi} f_{\tau}^{n}(\phi)]$

We note that we only need to store the coefficients for $f_{\tau_1, \ldots, \tau_{k-1}}^{\tau_k}$.

3 Data

3.1 Data description : Tunisian stock market

Financial data, and more specifically stock prices across time, are a good candidate to test the method. Indeed, stock prices, taken at a relatively low frequency (daily) are known to exhibit clear trends, while being noisy enough to make the detection of changepoints interesting. And since the method is designed to detect changepoints in a piecewise linear model, it is well suited for this kind of data.

We test the method on the stock prices of all the listed companies on the Bourse de Tunis, which is Tunisia stock exchange. The data was taken from Kagglee, and is available at https://www.kaggle.com/datasets/amariaziz/tunisian-stock-market. It contains the Open, High, Low and Close prices of 88 companies, as well as the volume of transactions. We only use the Close prices, which are the prices at the end of the trading day.

3.2 Data preprocessing

Since the data was partially collected by web scraping, it naturally requires some cleaning and preprocessing. We detail here the steps we took to clean the data.

Stocks with too few data points We see on figure 2 that the number of data points per company is very variable. We remove the stocks that have less than 200 data points, since they are not interesting for our analysis. We are left with 85 companies.

Missing values and duplicates Missing values are days where we have corrupted data for a given stock. We see in table 1 that the data is relatively clean of missing values, with only 3 companies having no more than a few percent of missing values. We see on figure 3 that each stock has between 1% and 3% of duplicates. We remove the missing values as well as the duplicates.

Missing days Missing days are days where we have no data at all for a given stock. We see on figure 4 that the number of missing days is clearly the biggest issue with the data, with most stocks having more than 40% of missing days. We deal with this issue by interpolating the missing days, using a linear interpolation between the last and next available data points. We note that this is not a perfect solution, since it assumes that the stock price evolves linearly between two days, which is never the case for stock prices. However, it is a simple solution that allows us to keep the data for all the stocks.

Data shrinkage by sub-sampling Since our method is relatively slow for time series longer than a few thousand points, and since our available computational power is limited, we sub-sample the data to reduce its size. We sub-sample the data by only keeping at most 1000 points per stock, by sampling uniformly on the time axis. We see on figure 5 that this sub-sampling very slightly affects the data.

4 Results

We present here a few experiments that we conducted on the 85 Tunisian stocks we preprocessed. First, we zoom on a few stocks to see how the method behaves and we qualitatively compare it to a best-fitting piecewise constant mean model. Second, we investigate the influence of the penalty parameter β on the number of changepoints detected. Third, we investigate the influence of the segment cost h on the number of changepoints detected. Finally, we briefly discuss the computational complexity of the method.

In all the experiments, we use a segment cost h of the form $h(s) = \gamma \log(s)$, where γ is a parameter to fix and s is the length of the segment. We also fix the parameter σ depending on the stock, by using the very classical Median Absolute Deviation (MAD) estimator first introduced in [**?**].

4.1 Qualitative comparison with a piecewise constant mean model

We compare the method to a piecewise constant mean model, which is the simplest model that can be used to detect changepoints. We note that the nature of the data (stock prices) puts the piecewise constant mean model at a great disadvantage, since stock prices are known to exhibit piecewise linear trends. We only perform this comparison qualitatively.

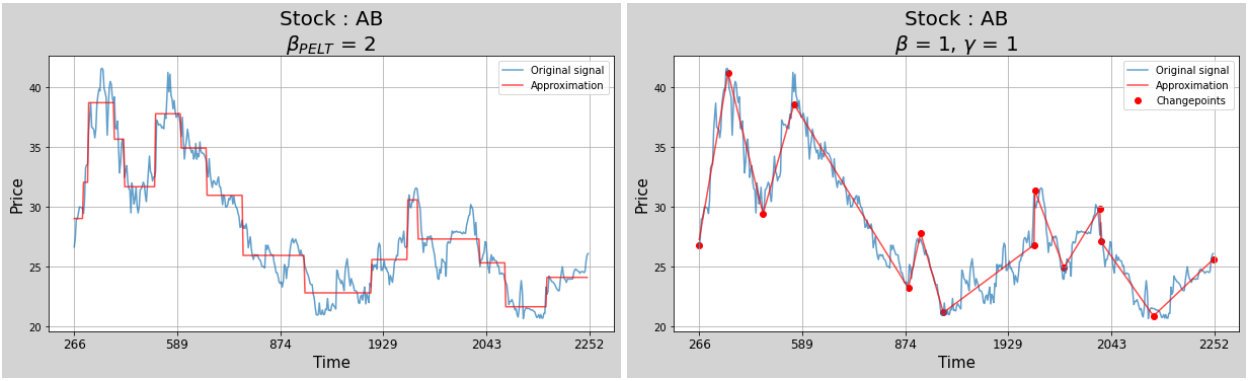

Figure 1: Comparison of the PELT method (left) and the CPOP method (right) on the stock of the company "Amen Bank" (AB).

We see on figure 1 that the CPOP method is able to detect the changepoints much more accurately than the PELT method. More precisely, the PELT method exhibits a clear tendency to detect multiple changepoints between two consecutive CPOP changepoints. This is due to the fact that the PELT method is, by design, only able to detect changes in the mean of the signal; Therefore, along a steep trend, as the signal rapidly and monotonically varies, it will detect multiple changepoints, while the CPOP method will only detect one changepoint at the beginning of the trend and one at the end.

4.2 Influence of the penalty parameter β

Next, we investigate the influence of the penalty parameter β on the number of changepoints detected. We see on figure 6 that the number of changepoints is a decreasing function of β, which is natural since β penalizes the number of changepoints. We visualize the approximations

obtained for values of $\beta \in \{0.1, 1, 5, 10, 20\}$ and $\gamma = 1$, on figure 7.

In particular, on figure 7, we see that almost all the data points are being detected for $\beta = 0.1$ which, again, is a very normal behaviour, since we get a very minimal mean-square error $\sum C$ as well as a very small cost for the segments $\sum h$ if we allow for a lot of changepoints.

Furthermore, we see that the number of changepoints detected is the same for $\beta = 10$ and $\beta = 20$. This is due to the fact that the segment cost $\sum h$, which penalizes the length of the segments, as well as the mean-square error $\sum C$, which penalizes the distance between the data and the approximation, are both dominant compared to the effect of penalization for β values in this $[10, 20]$ range.

4.3 Influence of the segment cost h

We used the same methodology to investigate the influence of the segment cost h on the number of changepoints detected. We recall that the segment cost h, which penalizes the length of the segments, was chosen of the form,

$$h(s) = \gamma \log(s),$$

where γ is a parameter.

We see on figure 8 the influence of γ on the number of changepoints detected. An interesting phenomenon is that we could at first expect that, in order to reduce h, the algorithm would tend to split the largest segments into smaller segments, but we see that this is not the case. Indeed, since the segment cost is logarithmic, the algorithm tends to split the smallest segments into even smaller segments, which is why we see that the number of changepoints detected doesn't necessarily decrease when γ increases. Furthermore, this explains why, as γ increases, we see distinct parts of the signal with a denser number of changepoints.

4.4 Computational complexity

5 Appendix

5.1 Correction of the computed coefficients of the polynomials $f_{\boldsymbol{\tau}}^t(\phi)$

The authors provide a calculation of the coefficients of the polynomials $f_{\boldsymbol{\tau}}^t(\phi)$ in Appendix C of the supplementary material of [**?**]. However, we found several mistakes in their calculations. We detail here the corrections we made.

The computation of the coefficients $A, \dots, F$ of $\mathcal{C}(y_{\tau_k+1:t}, \phi', \phi)$ is correct. We therefore take,

$$\mathcal{C}(y_{\tau_k+1:t}, \phi', \phi) = A\phi^2 + B\phi'\phi + C\phi + D + E\phi' + F\phi'^2, \tag{7}$$

with the coefficients $A, \dots, F$ given by the authors.

We have to solve the following minimization problem,

$$f_{\boldsymbol{\tau}}^t(\phi) = \min_{\phi'} \underbrace{\left(f_{\tau_1,\dots,\tau_{k-1}}^{\tau_k}(\phi') + \mathcal{C}(y_{\tau_k+1:t}, \phi', \phi) + h(t-\tau_k) + \beta\right)}_{:=g_\phi(\phi')}. \tag{8}$$

We note that g_ϕ is a quadratic polynomial in ϕ' since $\mathcal{C}$ is a quadratic polynomial in ϕ' and $f_{\tau_1,\dots,\tau_{k-1}}^{\tau_k}$ is a quadratic polynomial in ϕ' by induction hypothesis. We define, to simplify the notations,

$$f_{\tau_1,\dots,\tau_{k-1}}^{\tau_k}(\phi') = \hat{a} + \hat{b}\phi' + \hat{c}\phi'^2.$$

We can therefore write,

$$g_\phi(\phi') = \hat{a} + \hat{b}\phi' + \hat{c}\phi'^2 + A\phi^2 + B\phi'\phi + C\phi + D + E\phi' + F\phi'^2 + h(t-\tau_k) + \beta.$$

First, we compute the derivative of g_ϕ,

$$g_\phi'(\phi') = \hat{b} + 2\hat{c}\phi' + B\phi + E + 2F\phi',$$

and we set it to 0 to find the minimum of g_ϕ, which gives,

$$\phi' = -\frac{E+\hat{b}}{2(F+\hat{c})} - \frac{B}{2(F+\hat{c})}\phi.$$

To further simplify the notations, we define,

$$\alpha_1 = -\frac{E+\hat{b}}{2(F+\hat{c})} \quad \text{and} \quad \alpha_2 = -\frac{B}{2(F+\hat{c})} \quad \Longrightarrow \quad \phi' = \alpha_1 + \alpha_2\phi.$$

We can now compute the value of g_ϕ at the minimum, which gives us the value of $f_{\boldsymbol{\tau}}^t(\phi)$ of the form,

$$f_{\boldsymbol{\tau}}^t(\phi) = a + b\phi + c\phi^2.$$

where,

$$\begin{cases} a = \hat{a} + \hat{b}\alpha_1 + \hat{c}\alpha_1^2 + D + E\alpha_1 + F\alpha_1^2 + \beta + h(t-\tau_k), \\ b = \hat{b}\alpha_2 + 2\hat{c}\alpha_1\alpha_2 + B\alpha_1 + C + E\alpha_2 + 2F\alpha_1\alpha_2, \\ c = \hat{c}\alpha_2^2 + A + B\alpha_2 + F\alpha_2^2. \end{cases}$$

5.2 Preprocessing figures and tables

Stock	Number of missing values	Percentage of missing values
AST	10	1.51%
PLTU	10	3.19%
SIMPA	2	0.1%

Table 1: Number and percentage of missing values per stock.

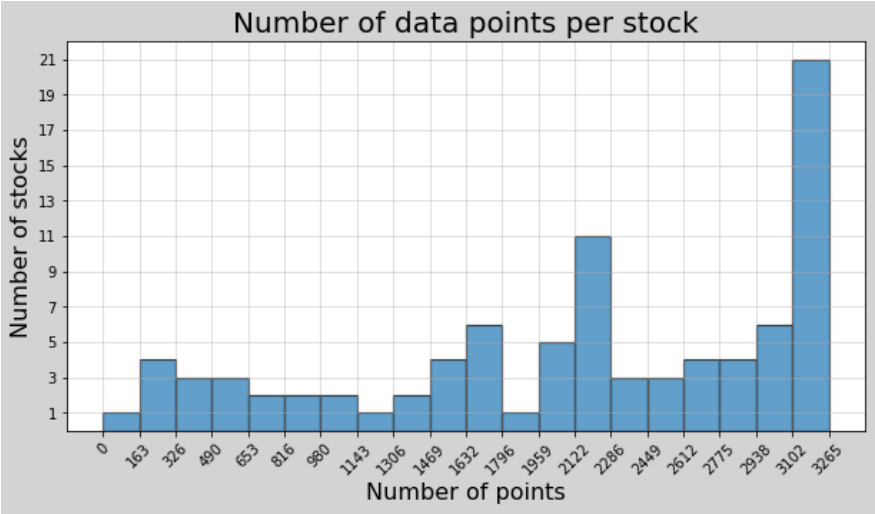

Figure 2: Histogram of the number of data points per stock.

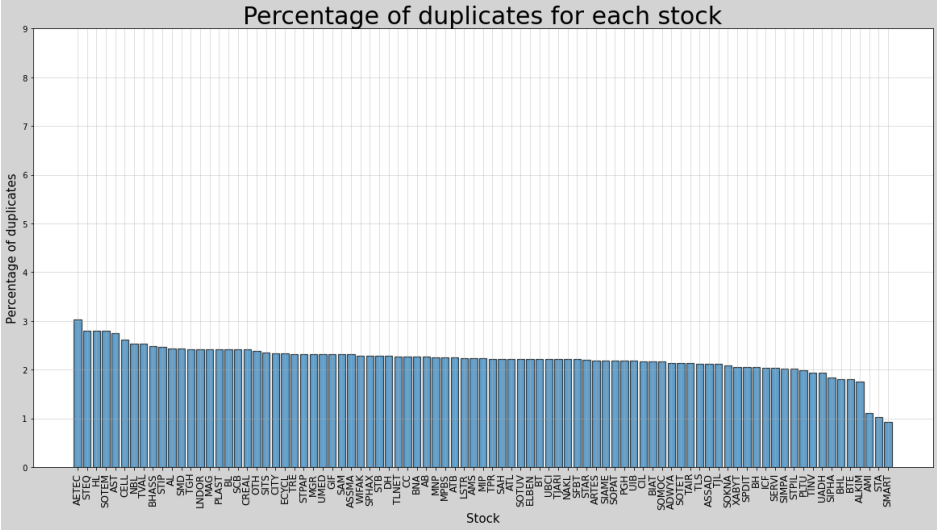

Figure 3: Percentage of missing values per stock. The stocks are sorted by decreasing percentage of missing values.

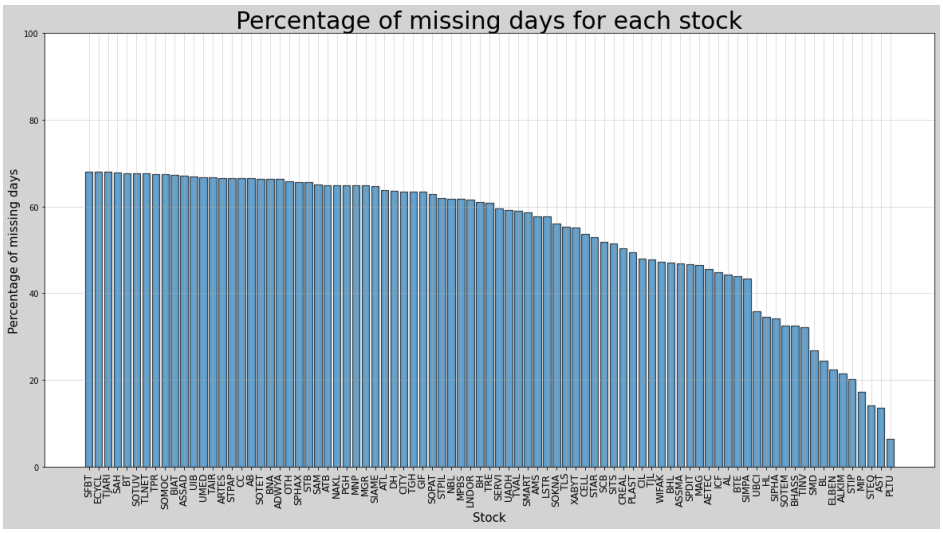

Figure 4: Percentage of missing days per stock. The stocks are sorted by decreasing percentage of missing days.

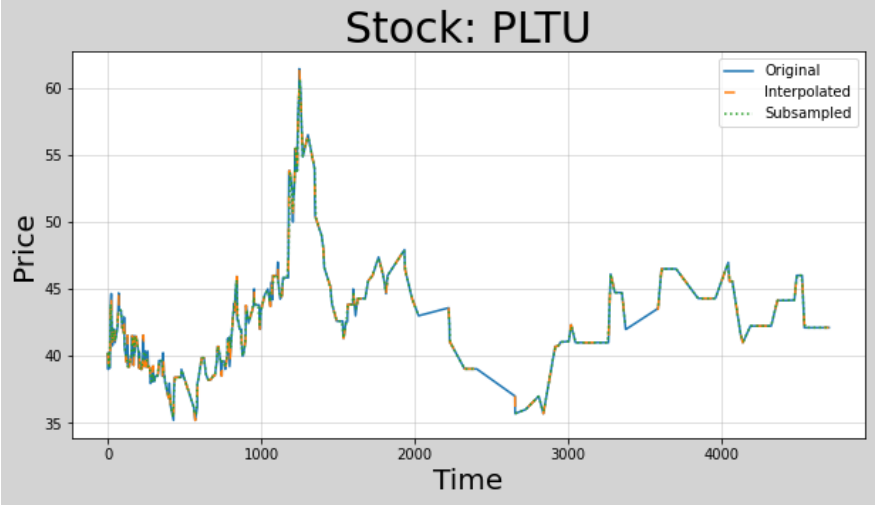

Figure 5: Comparison of the original data and the sub-sampled data on the "Placement de Tunisie - Sicaf" (PLTU) stock.

5.3 Results figures

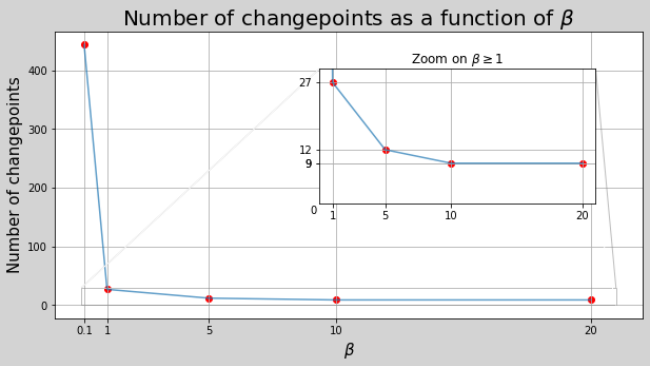

Figure 6: Number of changepoints detected as a function of the penalty parameter β.

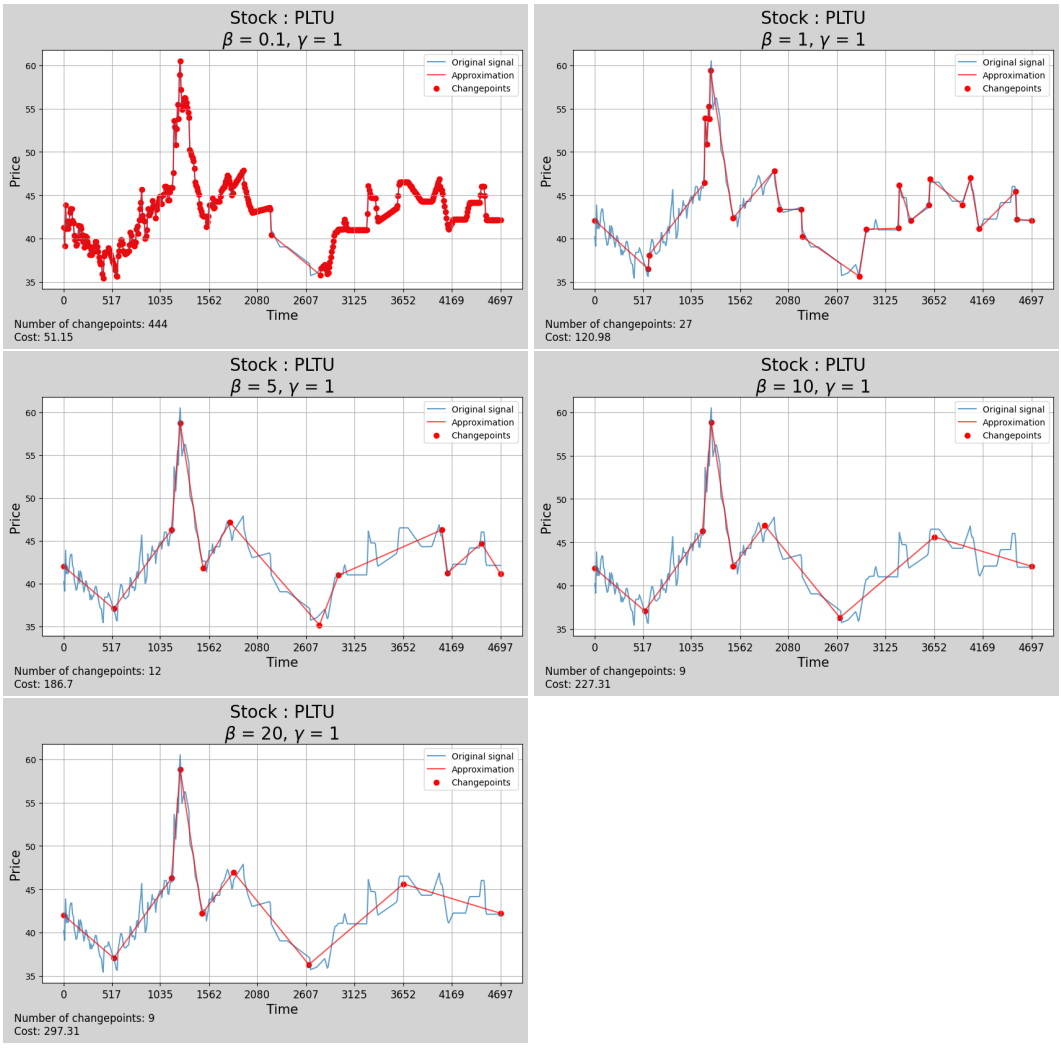

Figure 7: Approximations obtained for values of $\beta \in \{0.1, 1, 5, 10, 20\}$ and $\gamma = 1$, on the "Placement de Tunisie - Sicaf" (PLTU) stock.

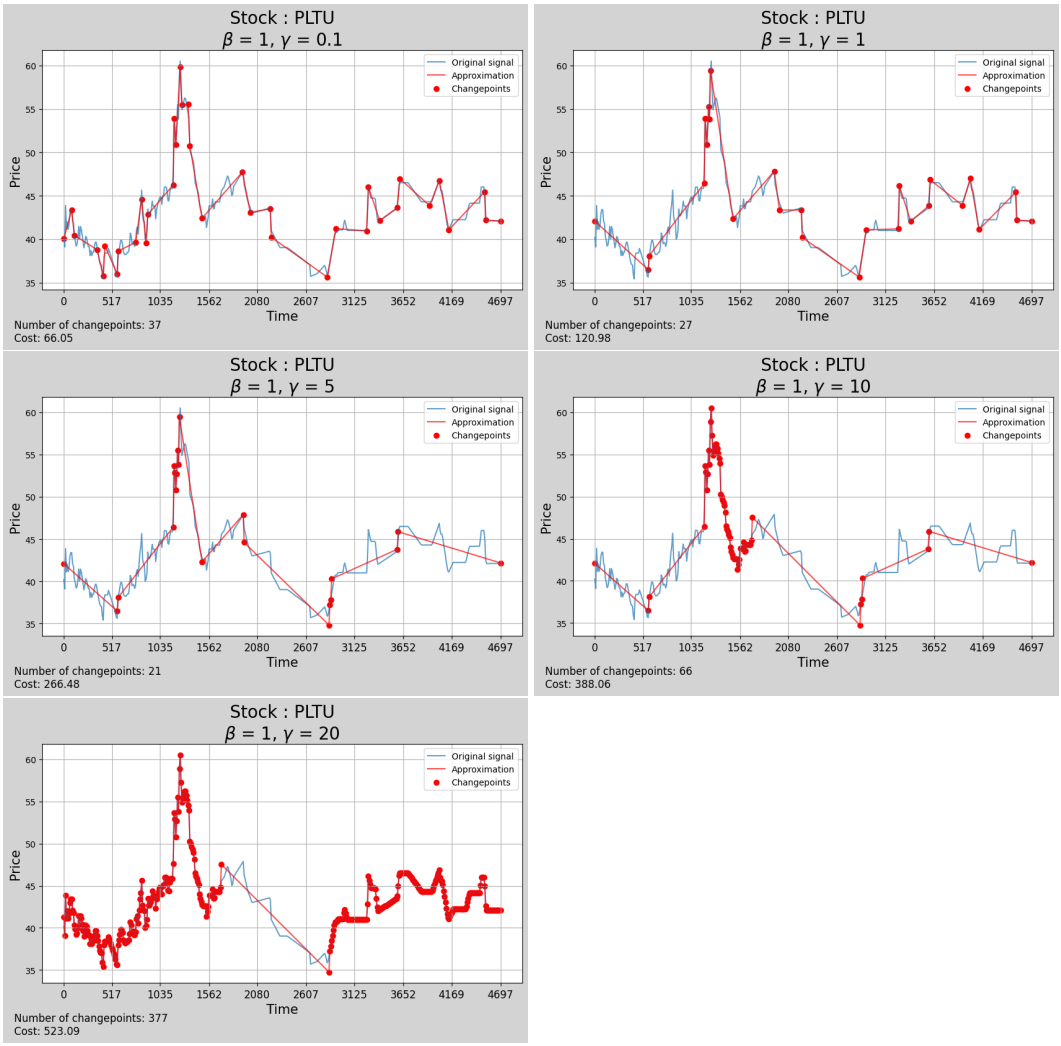

Figure 8: Approximations obtained for values of $\gamma \in \{0.1, 1, 5, 10, 20\}$ and $\beta = 1$, on the "Placement de Tunisie - Sicaf" (PLTU) stock.