# Point process and resolution of temporal data

**Redwan Mekrami**

## Abstract

Over the last few decades, the amount of data available on certain very liquid markets has allowed the development of the field of high frequency finance. Indeed, the challenge is to model and calibrate the micro structure of the markets as precisely as possible, and this obviously involves modeling the arrival of different events by taking into account the finest details. Until the 1990s, market dynamics were largely described by discrete-time models. Hasbrouck, Engle and Russel suggested the use of point processes in order to model financial data in a continuous way in time. In this context, they introduced the ACD model by representing the point processes by means of their intensity function which also aim to illustrate the conditional probabilities of arrival of different events on the market in the near future. A.G Hawkes introduced Hawkes processes during the 1970's to describe seismic events. The arrival of a wave was thus modeled using a Poisson process whose intensity depends on past historical data, thus the latter increases as it goes along and thus becomes important when an earthquake has just taken place. This property of historical dependence as well as the simplicity and flexibility of these processes have led to the use of Hawkes processes in the field of high frequency finance. This study was first introduced by Bowsher who suggested a Hawkes process model to describe the joint dynamics of trades and mid-price changes at the NYSE. During our study, we will place ourselves in the framework of Hawkes' model. The aim of our study is to study the effect of randomization on statistical inference in the framework of Hawkes processes. Indeed, in high frequency finance, the temporal data to be modeled often require a very fine resolution and can lead to the arrival of two events at the same time.

It is in this context that the randomization procedure comes into play in order to select the event to keep. We will first study these processes from a mathematical point of view, both in one dimension and in multidimensional, this study will allow us to simulate and estimate the different parameters of our model numerically. Afterwards, we will perform experiments to illustrate the impact of randomization. To do so, we reduce the quality of the simulated data, first by naive rounding, then by discretization of a temporal grid with variable time step, in order to study the impact on the different parameters obtained by estimation.

## 1 One-dimensional Hawkes process

In this section we will first study theoretically the Hawkes processes as well as the simulation algorithm, for that it is necessary at first to define the point and Poisson processes.

### 1.1 Poisson processes

**Definition 1** (Point Process)**.**
A point process is a random process whose elements are arrival times on a given set. Indeed, let $t$ be the physical time and $\{t_i\}_{i\in\{1,2,\cdots\}}$ a randomly increasing sequence of arrival times, $0 \leqslant t_i \leqslant t_{i+1}, \forall i$. Then the sequence is a point process on the segment $[0, \infty)$.

It is possible to generalize by defining a point pattern as a locally finite counting measure.

**Definition 2** (Counting process)**.**
Let $(N(t))_{t\geqslant 0}$ be a real-valued stochastic process such that $N(t) = \sum_{i\geqslant 1} 1_{\{t_i \leqslant t\}}$. We say that $N$ is a counting process if a.s. : $N(0) = 0$, $N$ is right continuous and $N$ is increasing.

**Definition 3** (Poisson process)**.**
The Poisson process of intensity $\lambda$ (strictly positive real) is a counting process of occurrences which verifies the following conditions :

- The numbers of occurrences in disjoint time intervals are independent: for any $t, s \geqslant 0$, the random variables $N(t+s) - N(s)$ are independent of the sigma algebra $\sigma\left(N_u, u \leqslant s\right)$ (independent increase).
- The probability of an occurrence in a small interval of time is proportional to the length of this interval, the proportionality coefficient being $\lambda$ : for all $t, s \geqslant 0, N(t+s) - N(t)$ follows a Poisson distribution of parameter $\lambda s$ (stationarity).
- The probability that there is more than one occurrence in a small time interval is negligible.

We are going to represent on the graph below, the distribution function of Poisson processes with different intensities $\lambda \in \{1, 4, 10\}$

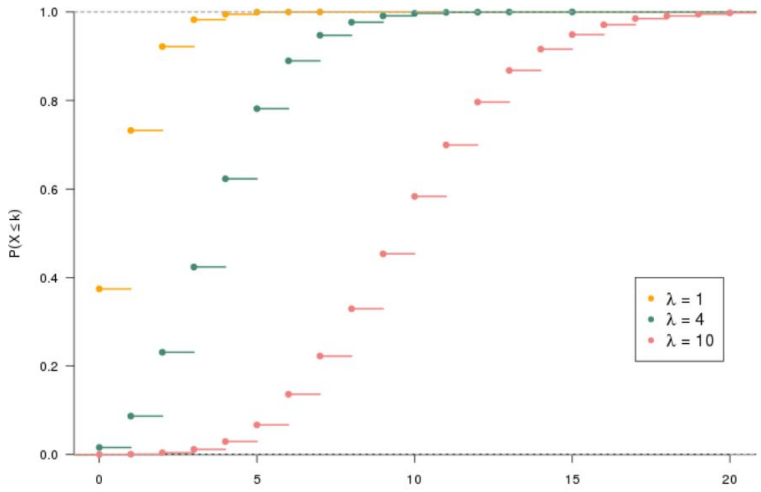


Figure 1: Cumulative distribution function of Poisson Processes

Poisson process can be used to model the Merton Jump-to-Ruin model. The stochastic differential equation of this model is the following:

$$dS_t = \lambda S_t dt + \sigma S_t dW_t - S_{t^-} dN_t$$

where $W$ is a Brownian and $N$ is a Poisson process of intensity $\lambda$. which allows to model a ruin of a stock (which when it reaches 0 can't be redeemed anymore). This is one example among many others. We will study the Hawkes process which is also a point process.

**Definition 4** (Intensity)**.**
Let $(\Omega, \mathcal{F}, P)$ be a probability space and $N$ be a counting process adapted to the filtration $\mathcal{F}_t$. The intensity process is continuous on the left and limited to right is defined by,

$$\begin{aligned}\lambda\left(t \mid \mathcal{F}_t\right) &= \lim_{h\to 0} \mathbb{E}\left[\frac{N(t+h) - N(t)}{h} \mid \mathcal{F}_t\right] \\ &= \lim_{h>0} \mathbb{P}\left[\left\{\frac{N(t+h) - N(t)}{h}\right\} > 0 \mid \mathcal{F}_t\right]\end{aligned}$$

The intensity characterizes the evolution of the process $N(t)$ conditional on its natural filtration. It can also be interpreted as the conditional probability of observing an event at the next instant. Note also that a point process adapted to the $\mathcal{F}_t$-filtration is an $\mathcal{F}_t$-martingale. And in particular, it is an super-martingale. Then, according to the Doob-Meyer decomposition, any $\mathcal{F}_t$-super-martingale can be decomposed (uniquely) as the sum of a zero-mean $\mathcal{F}_t$-martingale $M(t)$ and a $\mathcal{F}_t$-increasing predictable process $\Lambda(t)$,

$$N(t) = M(t) + \Lambda(t)$$

The predictable part involved in the Doob-Meyer decomposition is called the compensator and is defined by,

$$\Lambda(t) = \int_0^t \lambda\left(s \mid \mathcal{F}_t\right) \mathrm{d}s$$

By rearranging the previous expression we can give an alternative definition of the intensity,

$$N(t) - \int_0^t \lambda\left(s \mid \mathcal{F}_t\right) \mathrm{d}s = M(t)$$

$M(t)$ being a martingale, we obtain,

$$\mathbb{E}\left[N(t) \mid \mathcal{F}_s\right] = \mathbb{E}\left[\int_0^t \lambda\left(u \mid \mathcal{F}_u\right) \mathrm{d}u \mid \mathcal{F}_s\right], \text{ p.s.}$$

However,

$$\mathbb{E}\left[N(t) - N(s) \mid \mathcal{F}_s\right] = \mathbb{E}\left[\int_s^t \lambda\left(u \mid \mathcal{F}_u\right) \mathrm{d}u \mid \mathcal{F}_s\right] = \mathbb{E}\left[\Lambda(s,t) \mid \mathcal{F}_t\right], \text{p.s.}$$

with $\Lambda(s,t) = \Lambda(t) - \Lambda(s)$.

The compensator has several applications, notably in the calibration of the Hawkes process (in one-dimensional or multidimensional) and for random time change methods.

### 1.2 One-dimensional Hawkes process Mathematical Framework

In this section, we will introduce the different mathematical results essential to the study of Hawkes processes. Let us first define them:

**Definition 5** (One-dimensional Hawkes process)**.**
Let $N$ be a counting process adapted to the $\mathcal{F}_t$ filtration, then if its intensity is of the type,

$$\lambda(t) = \nu + \int_0^t w(t-s)N(ds) = \nu + \sum_{ti<t} w(t-ti),$$

with $\nu \geqslant 0$ and $w : \mathbb{R}^+ \to \mathbb{R}^+$, the process $N$ is a Hawkes process. $w$ is the penalty kernel, it is the kernel that will give the variations of the intensity according to the arrival times. It expresses the effect of the previous events $t_i$ on the value of the intensity and thus allows to control the speed of growth and relaxation of the intensity of the process linked to the different excitations. $\nu$ is the "fixed" intensity ($\lambda(t) \geqslant \nu$).

In the framework of Hawkes processes, we often find exponential penalty kernels (exponantial kernel) which have the following form, with $\alpha > 0$ :

$$w(t) = \alpha \exp^{-\beta t}$$

Thus, the farther $t_i$ is from time $t$, the more the function tends to 0 . On the contrary, the closer $t_i$ is to $t$, the closer the function gets to $\alpha$. We thus have a temporary increase of the intensity at each new arrival of $t_i$.

### 1.3 Thinning method and Ogata's algorithm for the simulation of the Hawkes process

#### 1.3.1 Thinning method

**Theorem 1** (Thinning method for stochastic intensity processes)**.**
Let $N$ be a point process of intensity $\lambda_t$ on the interval $[0, T]$. We suppose that we can find a process

$\lambda_t^*$ such that $\lambda_t \leqslant \lambda_t^*$. Let $(t_1^*, \ldots, t_n^*) \in (0, T]$, be the points $du$ process $N^*$ of intensity $\lambda_t^*$. Then, for each of these points, we associate a weight $p = 1$ with a probability

$$\frac{\lambda_{t_j^*}}{\lambda_{t_j^*}^*}$$

and $p = 0$ otherwise. Therefore the points of weight $p = 1$ constitute a process identical to $N$.

We will first recall the form of the intensity of a Hawkes process in one dimension:

$$\lambda(t) = \lambda(0) + \sum_{t_i < t} \alpha e^{-\beta(t - t_i)}$$

Ogata's algorithm will allow us to generate arrival times that follow the Hawkes process law. The latter takes as input the parameters $\lambda_0, \alpha$, and $\beta$. We will first detail the general working principle of the algorithm:

- Initialization: The algorithm generates the first arrival time that we note s, of exponential law $\lambda_0$ (initial intensity). As we place ourselves on the interval $[0, T]$, the generated time must be included in this interval. For the simulation we will generate a uniform law $U \sim \mathcal{U}([0, 1])$ and we will have that $s \leftarrow -\frac{1}{\lambda_0} \log(U)$
- In a recurrent way, the algorithm will modify the maximum intensity by adding the parameter $\alpha$ so that at the time n we have $(\lambda_* = \lambda(t_{n1}) + \alpha)$ and generates a new event which follows an exponential law of parameter $\lambda_*$ while verifying the same condition as during initialization. This event can be seen as the next arrival time of our Hawkes process.
- Always during the step n, the algorithm will carry out a rejection attribution test, indeed we accept the new time with a probability $\frac{\lambda(t_i)}{\lambda*}$ otherwise we return to the step above of simulation of an event following an exponential law.
- At the output of the algorithm, we obtain the different arrival times which represent our Hawkes process in one dimension.

**Theorem 2** (Algorithm of Ogata (1981))**.**
Below we will detail the procedure explained above in a more precise and clear way:

1. Initialization : $i \leftarrow 1$ and $\lambda^* \leftarrow \lambda_0$.
2. First event: Generate $s \sim \mathcal{P}(\lambda^*)$ (Generate $U \sim \mathcal{U}([0, 1])$ and do $s \leftarrow -\frac{1}{\lambda^*} \log(U)\big)$.
   (a) If $s > T$, go to the last step.
   (b) Otherwise: $t_1 \leftarrow s$.
3. Main loop: As long as $t_i \leqslant T$ : do $i \leftarrow i + 1$
   (a) Update the maximum intensity: do $\lambda^* \leftarrow \lambda(t_{i-1}) + \alpha$.
   (b) New event: generate $s \sim \mathcal{P}(\lambda^*)$. If $s > T$, then go to the last step.
   (c) Assignment-rejection test: generate $D \sim \mathcal{U}([0, 1])$ :
      i. If $D \leqslant \frac{\lambda(s)}{\lambda^*}$, then $t_i \leftarrow s$ and repeat the main loop (acceptance),
      ii. Otherwise, update $\lambda^* \leftarrow \lambda(s)$ and return to step 3.b. to generate a new event s (rejection).
4. Output: return the simulated point process $(\{t_i\}_i)$ on $[0, T]$.

Below, we can see the result obtained by using Ogata's algorithm to simulate a Hawkes process of parameters $\lambda_0 = 6, \alpha = 0.6, \beta = 1$ :

### 1.4 Influence of the parameters on the simulation of the Hawkes process

We will now study the influence of each parameter of the model on the simulation of the Hawkes process by applying the Ogata algorithm while varying each parameter and fixing the two others.

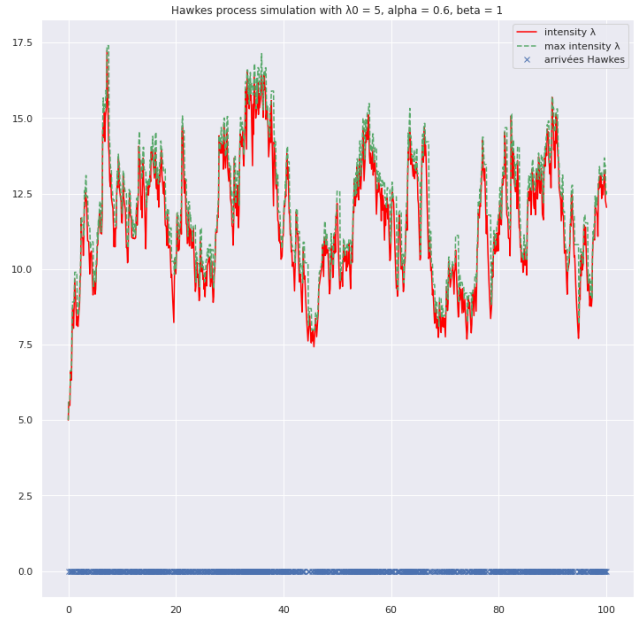


Figure 2: Hawkes Process Simulation

### 1.4.1 Influence of $\lambda_0$

We will first study the influence of the parameter $\lambda_0$ which represents the initial intensity of our process. Thus if the value of this parameter increases the intensity of the arrivals increases consequently the number of events as well as their average value grows according to this parameter it is what we can note from the graphs below, where we chose to represent $\lambda_0 = 0.1, \lambda_0 = 1, \lambda_0 = 10$ while keeping $\alpha = 0.6, \beta = 1$

### 1.4.2 Influence of $\alpha$

In the expression of the intensity, $\alpha$ is the multiplying coefficient that we find in front of the exponential, consequently we can expect that the intensity of the arrivals of the Hawkes process increases according to $\alpha$ and thus the more this parameter is big and the more the peaks will be consequent. This is what we observe on the results obtained below representing $\alpha = 0.01, \alpha = 0.1, \alpha = 0.9$ with $\lambda_0 = 0.1, \beta = 1$

### 1.4.3 Influence of $\beta$

The parameter $\beta$ influences the exponential decay rate of the intensity of our process. Indeed, this term is a factor in the exponential decay and will therefore parameterize the decay rate. To illustrate this, we have chosen to plot the results obtained for $\beta = 0.1, \beta = 3, \beta = 10$.

## 1.5 Estimation of the Hawkes process parameters

In this section, we will study the estimation of the different parameters of the one-dimensional Hawkes model: $\lambda_0, \alpha, \beta$. We will first define the likelihood of a point process and more particularly that of the Hawkes model. Then we will use the maximum likelihood estimator in order to have an estimate of the parameters of the model and we will finally study the properties of this estimator and its convergence.

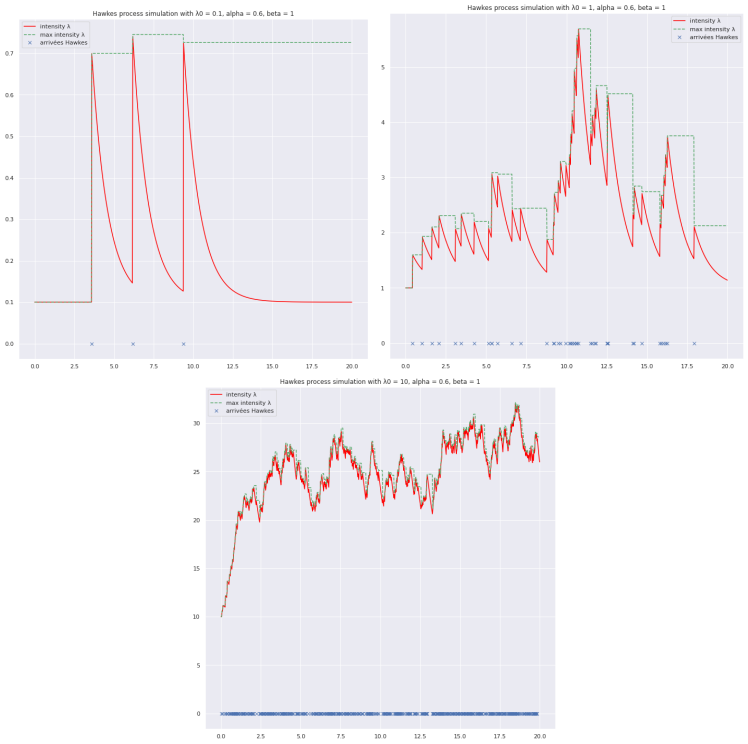


Figure 3: Simulation of a one-dimensional Hawkes process with $\lambda_0 = 0.1, \lambda_0 = 1, \lambda_0 = 10$

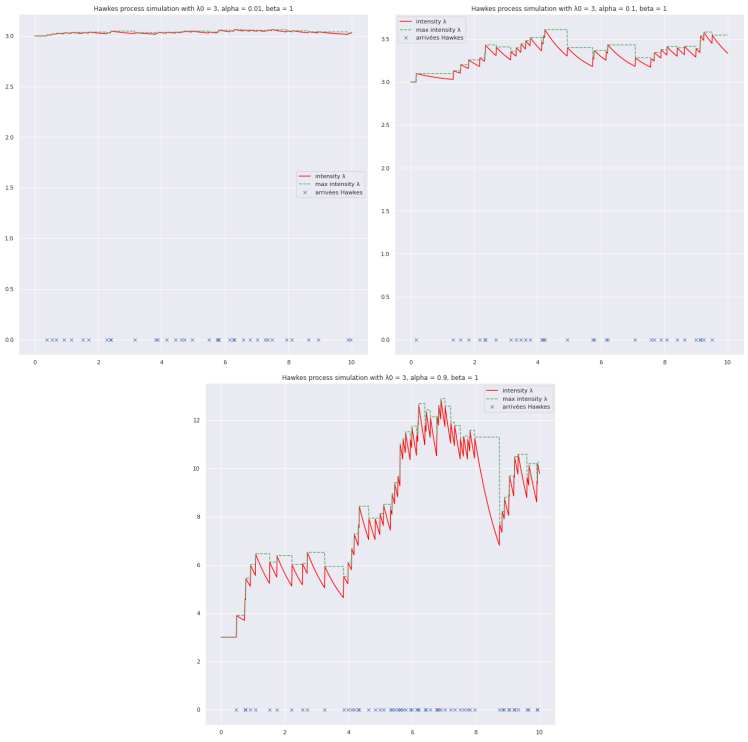


Figure 4: Simulation of a one-dimensional Hawkes process with $\alpha = 0.01, \alpha = 0.1, \alpha = 0.9$

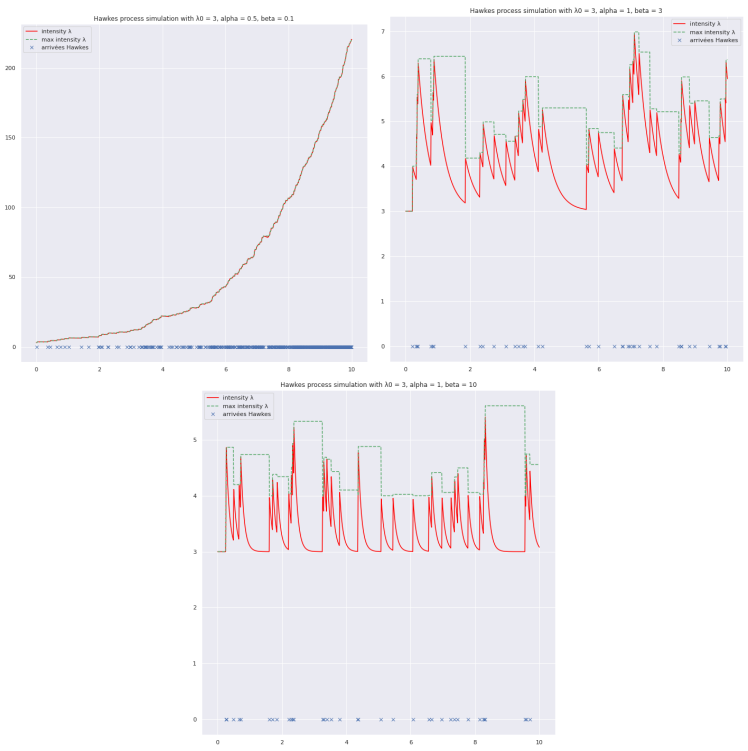


Figure 5: Simulation of a one-dimensional Hawkes process with $\alpha = 0.1, \alpha = 3, \alpha = 1$

### 1.5.1 Maximum likelihood

**Theorem 3** (The log-likelihood of a point process)**.**
The log-likelihood of a point process $(N_t)_{t\geqslant 0}$ having intensity $(\lambda_t)_{t\geqslant 0}$ that we observe on $[0, T]$ is :

$$\log\left(\mathcal{L}_T\right) = \int_0^T \log\left(\lambda_s\right) \mathrm{d}N_s - \int_0^T \lambda_s \, \mathrm{d}s$$

**Theorem 4** (The log-likelihood of a Hawkes process with exponential kernel)**.**
For an exponential Hawkes process, the log-likelihood is written:

$$\begin{aligned}\log\left(\mathcal{L}\left(\alpha, \beta, \lambda_0 \mid t\right)\right) &= -T\lambda_0 - \sum_{i=1}^{n} \frac{\alpha}{\beta}\left(1 - e^{-\beta(T-t_i)}\right) + \sum_{i=1}^{n} \log\left(\lambda_0 + \sum_{t_k<t_i} \alpha e^{-\beta(t_i-t_k)}\right) \\ &= -T\lambda_0 - \sum_{i=1}^{n} \frac{\alpha}{\beta}\left(1 - e^{-\beta(T-t_i)}\right) + \sum_{i=1}^{n} \log\left(\lambda_0 + \alpha R(i)\right)\end{aligned}$$

where $R(1) = 0$ and $R(i) = e^{-\beta(t_i - t_{i-1})}(1 + R(i-1))$

One can notice from the expression below that it is difficult to obtain an explicit expression of the maximum likelihood that could be easily implemented. We have therefore chosen to use the "minimize" function of the Scipy.optimize module by defining as linear constraints: $\alpha \leqslant \beta$ and all values are positive.

### 1.5.2 Study of the maximum likelihood estimator

The previous implementation allows us to obtain the estimate of the maximum likelihood estimator:

$$\hat{\theta}_{EMV} = \arg \max_{alpha,\beta,\lambda_0} \log\left(\mathcal{L}\left(\alpha, \beta, \lambda_0 \mid t\right)\right)$$

**Convergence**: We will first study the convergence of our estimators by varying $mathrmT$ and the number of occurrences. We notice that the estimators converge to the parameters used to simulate the data with the Ogata algorithm $\lambda_0 = 5, \alpha = 0.6, \beta = 1$.

**Unbiased estimators**: We are now going to study the distribution of the three estimators, we notice by plotting the corresponding histograms for $n = 1000$ that the three estimators follow a centered distribution around the value we are trying to estimate, therefore they are indeed unbiased estimators.

| T | Number of occurrences | $\alpha$ | $\beta$ | $\lambda_0$ |
|---|---|---|---|---|
| 1000 | 13433 | 0.675 | 1.050 | 4.797 |
| 2000 | 25340 | 0.540 | 1.101 | 5.462 |
| 3000 | 37058 | 0.600 | 1.006 | 4.995 |
| 4000 | 50788 | 0.559 | 0.933 | 5.100 |

Table 1: Results obtained for $\alpha = 0.6$, $\beta = 1$, $\lambda_0 = 5$

As specified above, the estimators have a normal distribution centered around the value to be estimated and for each estimator $\hat{\theta}_{n_1}\sqrt{n}\left(\hat{\theta}_n - \theta^*\right)$ converges in law to $\mathcal{N}\left(0, \frac{1}{I(\theta)}\right)$ where $I(\theta)$ is the Fisher information.

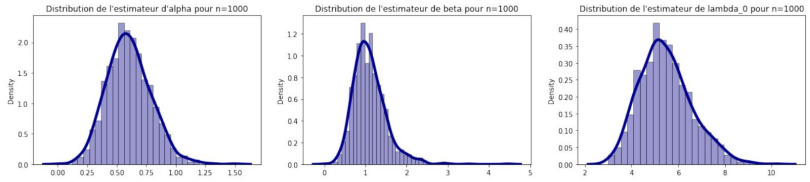


Figure 6: Histograms of the distributions of the maximum likelihood estimators with as data for the simulation $\beta = 1$, $\alpha = 0.6$, $\lambda_0 = 5$

## 2 Multidimensional Hawkes

### 2.1 Theoretical framework

To better model an order book or the variations of any financial asset at very high frequency, we need to extend the definition of a Hawkes process to the multivariate case. Indeed, several phenomena that must be taken into account, and that can influence the phenomenon to be modeled, come into play and themselves can be modeled by Hawkes processes. For example, the orders that move the price are not only market orders, limit orders and cancellations also have an important role on the variations observed on the price.

**Definition 6** (Multivariate Hawkes process)**.**
Let $\boldsymbol{N} = (N_1(t), \cdots, N_M(t))$ be a filtration-matched counting process $\mathcal{F}_t$ associated with the point process $(t_{i,1}, \cdots, t_{i,M}), M \in N$, then if the conditional intensity takes the form:

$$\lambda^j\left(t \mid \mathcal{F}^t\right) = \lambda_0^j(t) + \sum_{n=1}^{M} \int_0^t g_{mn}(t-s) N_n(\mathrm{d}s)$$

where $\lambda_0(t) = \left(\lambda_0^1(t), \cdots, \lambda_0^M(t)\right)$ is a vector of exogenous intensities and $\boldsymbol{G} = g_{mn}(t)_{m,n=1,.M}$ is the kernel matrix, then $(N_1, \cdots, N_M)$ is a multivariate Hawkes process.
The matrix $\boldsymbol{G} = g_{mn}(t)_{m,n=1,..,M}$ verifies the following properties:

- $\forall m, n = 1, \cdots, M, \quad g_{mn}(t) \geqslant 0.$
- If $t < 0, \quad$ then $g_{mn}(t) = 0 \ \forall m, n = 1, \cdots, M.$
- $\forall m, n = 1, \cdots, M \quad g_{mn}(t) \in \boldsymbol{L}^1.$

**Theorem 5** (Stability condition)**.**
The process $N_t$ has asymptotically stationary increments and $\lambda(t)$ is asymptotically stationary if the kernel matrix satisfies the following condition, called the stability condition:

$$\max\{\sigma(\boldsymbol{G})\} < 1$$

where $\sigma(.)$ is the spectral radius. To calibrate the model parameters, we introduce the likelihood of the Hawkes process:

$$\ln L = \sum_{j=1}^{d} \int_0^T \ln \lambda_j\left(t \mid \mathcal{F}_t\right) N(\mathrm{d}t) - \sum_{j=1}^{d} \Lambda_j(T)$$

### 2.2 Simulation of a multidimensional Hawkes process

As defined earlier, a M-dimensional Hawkes process is a point process $N_t = \left(N_t^1, .., N_t^M\right)$ with associated intensities $(\lambda_t^m)_{t \geqslant 0}, m = 1, ..,$ such that:

$$\lambda^m(t) = \lambda_0^m(t) + \sum_{n=1}^{M} \int_0^t g_{mn}(t-s) dN_s^n$$

where $\lambda_0 : [0, \infty) \to \mathbb{R}^+$ and $g_{mn} : [0, \infty) \to \mathbb{R}^+, m, n = 1, ..M$ are deterministic functions. Using vector notations:

$$\underline{\lambda}(t) = \underline{\lambda_0}(t) + \int_0^t \underline{G}(t-s) \cdot \underline{N}_s.$$

For the simulation, we will use exponential kernel functions of the form:

$$\underline{G}(t) = \left(\alpha^{mn} e^{-\beta^{mn} t}\right)_{m,n=1,\ldots,M},$$

and

$$\lambda^m(t) = \lambda_0^m(t) + \sum_{n=1}^{M} \sum_{t_n < t} \alpha^{mn} e^{-\beta^{mn}(t-t_n)}.$$

A M dimensional Hawkes process is said to be stable if the spectral radius of the matrix $\Gamma = \int_0^\infty \underline{G}(u)du < 1$. In the case of the exponential kernel: $\Gamma = \left(\frac{\alpha^{mn}}{\beta^{mn}}\right)_{m,n=1,..,M}$ and to simplify, let $\underline{\lambda_0}(t) = \underline{\lambda_0}$.

We define $I^K(t) = \sum_{n=1}^K \lambda^n(t)$ the sum of the intensities of the first $K$ components of the multivariate process. $I^M(t)$ is then the sum of the intensities of all the components (total intensity). The simulation algorithm of a multivariate Hawkes process uses the "thinning" principle in the following way:

1. Initialization: $i \leftarrow 1, i^1 \leftarrow 1, .., i^M \leftarrow 1$ and $I* \leftarrow I^M(0) = \sum_{n=1}^M \lambda_0^n(0) = \sum_{n=1}^M \lambda_0^n$.
2. First event: Generate $s \sim \mathcal{P}(I^*)$ (either by using the numpy.random.exponential function on python or generate $U \sim \mathcal{U}([0,1])$ and do $s \leftarrow -\frac{1}{I*}\log(U)\big)$.
   (a) If $s > T$, go to the last step.
   (b) Otherwise: attribution test: generate $D \sim \mathcal{U}([0,1])$ and do $t_1^{n_0} \leftarrow s$ where $n_0$ is such that $D \sim \mathcal{U}([0,1])$ and faire $t_1^{n_0} \leftarrow s$ où $n_0$ est tel que $\frac{I^{n_0-1}(0)}{I*} < D \leqslant \frac{I^{n_0}(0)}{I*}$, and $t_1 \leftarrow t_1^{n_0}$.
3. Main loop: do $i^{n_0} \leftarrow i^{n_0} + 1$ and $i \leftarrow i + 1$
   (a) Update the maximum intensity: do $I^* \leftarrow I^M(t_{i-1}) + \sum_{n=1}^M \alpha^{nn_0}$.
   (b) New event: generate $s \sim \mathcal{P}(I^*)$. If $s > T$, then go to the last step.
   (c) Assignment-rejection test: generate $D \sim \mathcal{U}([0,1])$ :
      i. If $D \sim \frac{I^M(s)}{I*}$, then $t_{i^{n_0}}^{n_0} \leftarrow s$ where $n_0$ is such that $\frac{I^{n_0-1}(s)}{I*} < D \leqslant \frac{I^{n_0}(s)}{I*}$ et $t_i \leftarrow t_{i^{n_0}}^{n_0}$ and repeat the main loop (acceptance) while checking the condition $t_i \leqslant T$,
      ii. Otherwise, update $I^* \leftarrow I^M(s)$ and return to step 3.b. to generate a new event $s$ (rejection).
4. Output: return the simulated point process $\left(\{t_i^n\}_i\right)_{n=1,\dots,M}$ on $[0,T]$.

Figure 6 represents the simulation of a Hawkes process in 2 d with the parameters below. We check that the stability condition is respected (the spectral radius is $0.537$).

$$\lambda_0 = \begin{bmatrix} 0.1 \\ 0.5 \end{bmatrix}, \quad \alpha = \begin{bmatrix} 0.1 & 0.2 \\ 0.3 & 0.4 \end{bmatrix}, \quad \beta = \begin{bmatrix} 1.0 & 1.0 \\ 1.0 & 1.0 \end{bmatrix}, \quad T = 80$$

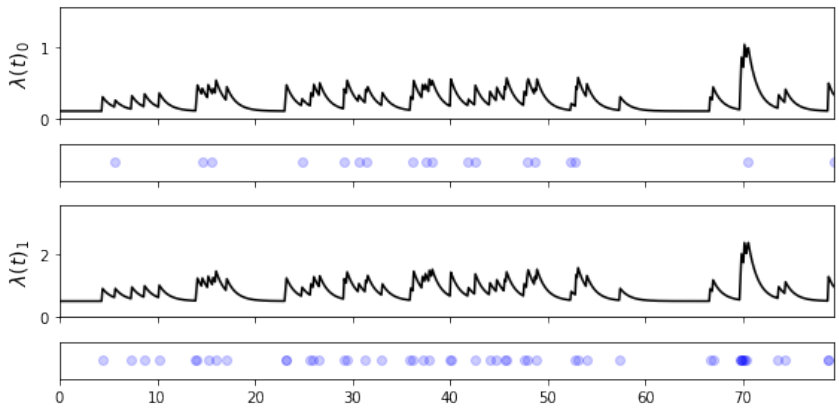


Figure 7: Simulation of a Hawkes process in 2 dimensions. The graph on top corresponds to the first component of the process, and the one on the bottom corresponds to the second.

### 2.3 Estimation of a multidimensional Hawkes process

#### 2.3.1 Maximum likelihood

The estimation of the parameters of a Hawkes process is done in a similar way to the one-dimensional case with a likelihood method. In the multidimensional case, the log-likelihood can be computed as the sum of the likelihoods along each coordinate and we obtain the following formula:

$$\log \mathcal{L}\left(\{t_i\}_{i=1,\dots,N}\right) = \sum_{m=1}^{M} \log \mathcal{L}^m\left(\{t_i\}\right)$$

Where the term following each coordinate is calculated with its associated intensity with :

$$\log \mathcal{L}^m\left(\{t_i\}\right) = \int_0^T \left(1 - \lambda^m(s)\right) ds + \int_0^T \log \lambda^m(s) dN^m(s).$$

And by injecting this formula into the first expression, the final expression for the loglikelihood is:

$$\log \mathcal{L}^m\left(\{t_i\}\right) = -\lambda_0^m T - \sum_{i=1}^{N}\sum_{n=1}^{M} \frac{\alpha^{mn}}{\beta^{mn}}\left(1 - e^{-\beta^{mn}(T-t_i)}\right) + \sum_{t_i^m} \log\left(\lambda_0^m + \sum_{n=1}^{M} \alpha^{mn} R^{mn}(i)\right)$$

$$\text{Avec } R^{mn}(0) = 0 \text{ et } R^{mn}(i) = \begin{cases} e^{-\beta\left(t_i^m - t_{i-1}^m\right)}\left(1 + R^{mn}(i-1)\right) & \text{si } m = n \\ e^{-\beta\left(t_i^m - t_{i-1}^m\right)} R^{mn}(i-1) + \sum_{t_{i-1}^m \leqslant t_k^m < t_i^m} e^{-\beta^{mn}(t_i^m - t_k^n)} & \text{sinon} \end{cases}.$$

However, one can easily notice that the log-likelihood function is not strictly concave, which makes its optimization more complicated. Moreover, it has several local minima. We have tried to minimize the log-likelihood function by using the following implementations of the Scipy library of python: the Differential Evolution DE algorithm and the trust-constr method applied to constrained optimization problems. Nevertheless, we encountered convergence problems of the algorithms when calibrating the M-dimensional Hawkes models. Indeed, when starting from the true value of the parameters as the initial point for the "trust-constr" method method, the algorithm converges to a point different from the true value of the parameters.

#### 2.3.2 EM estimation

Another method that we tried to implement was an EM estimator and which avoids the problems that could be encountered in very high dimension by using maximum likelihood (given the dimension explosion problems in the MLE). Another advantage is that the EM has closed formulas for the parameters within each iteration of the estimation (given in the literature) and makes good use of the micro-structure of Hawkes processes. The EM agorithm is an iterative algorithm for finding the parameters that maximize the likelihood of a probabilistic model when the model depends on unobservable latent variables. It consists of two steps that are repeated at each iteration: an expectation evaluation step (E-step), where the expectation of the likelihood of the model is computed taking into account the last observed variables, and a maximization step (M-step), where the maximum likelihood of the parameters is estimated by maximizing the likelihood found in step E.

In our case, the latent variable is the branching matrix $Q$ describing the parent-descent relationship for each event of the process. More precisely, we note $Q = [q_{ij}]$ our latent branching matrix such that $q_{ij} = 1$ if the event $i$ was caused by $j$ (0 otherwise), and $q_{ii} = 1$ implies that $i$ is a background event. We will seek to maximize the posterior law of the complete data, which is defined as follows:

**Definition 7** (Complete Data posterior of a Hawkes process)**.**
For a sequence $\tau = \{(t_i, u_i)\}_{i=1}^N$, a branching matrix $Q = [q_{ij}]$, and parameters $\Theta$, the complete data posterior is:

$$p(\Theta \mid \tau, Q) \propto p(\tau, Q \mid \Theta) p(\Theta)$$

With these notations, we can now define the log-likelihood of the data completed by [2]:

$$\mathcal{L}(\tau, Q; \Theta) = \log p(\tau, Q; \Theta) + \log p(\Theta)$$

It remains now to express the expectation of the completed likelihood of the data:

$$\mathbb{E}[\log p(\tau, Q \mid \Theta)] = \sum_{i=1}^{N} p_{ii} \log \frac{\mu_{u_i}}{p_{ii}} + \sum_{i=1}^{N} \sum_{j=1}^{i-1} p_{ij} \log \frac{\alpha_{u_i u_j} g\left(t_i - t_j\right)}{p_{ij}}$$
$$- T \sum_{n=1}^{M} \lambda_0^n - \sum_{n=1}^{M} \sum_{j=1}^{N} \alpha_{n n_j} G\left(T - t_j\right)$$

Where $T$ is the final date of the observed times, $N$ the number of events, $M$ the dimension of the model and $G = \int_0^t g(s)ds$. However, the drawback of this estimator is that the kernel $g$ is known in advance. We have therefore applied it to our problem by taking constant $\beta$ and given as input. We note $P = [p_{ij}]$ the expectation of $Q$ in the expectation evaluation step E:

$$P^{(k+1)} = \mathbb{E}\left[Q \mid \tau, \Theta^{(k)}\right]$$

which will be

$$p_{ii}^{(k+1)} = \frac{\lambda_0^{u_i(k)}}{\lambda_0^{i(k)} + \sum_{j=1}^{i-1} \alpha_{u_i u j}^{(k)} g\left(t_i - t_j\right)}$$
$$p_{ij}^{(k+1)} = \frac{\alpha_{u_i u_j}^{(k)} g\left(t_i - t_j\right)}{\lambda_0^{i(k)} + \sum_{j=1}^{i-1} \alpha_{u_i u j}^{(k)} g\left(t_i - t_j\right)}$$

In the $M$ step of the algorithm, we update the formulas by solving the condition

$$\Theta^{(k+1)} = \operatorname{argmax}\left\{\mathbb{E}\left[\mathcal{L}\left(\tau, Q^{(k)}; \Theta, V\right) \mid Q^{(k)} = P^{(k+1)}\right]\right\}$$
$$= \operatorname{argmax}\left\{\mathbb{E}\left[\log p\left(\tau, Q^{(k)}; \Theta\right) \mid Q^{(k)} = P^{(k+1)}\right] + \mathbb{E}\left[\log p\left(\Theta^{(k)}; V\right)\right]\right\}$$

Thus:

$$\lambda_0^{(k+1)} = \frac{\sum_{i:u_i=u} p_{ii}^{(k)}}{T}$$
$$\alpha_{uu'}^{(k+1)} = \frac{\sum_{i,u=u_i} \sum_{j,uj=u',j<i} p_{ij}^{(k)}}{\sum_{i=1}^{N} \sum_{j,u_j=u',j<i} G\left(T - t_j\right) + t_{uu'}}$$

# 3 Numerical experimentation

Now that we have our generator/estimator tandem working for 1D Hawkes processes, we will try to reduce the quality of the data to see how our estimator performs. The interesting metric in a Hawkes process is the ratio $\frac{\alpha}{\beta}$. Thus, for all experiments, we decided to set the following values:

- $\lambda_0 = 1$
- $T = 1000$
- $\beta = 1$

We will then vary $\alpha$ for our experiments.

## 3.1 Data quality via truncation

### 3.1.1 Study of the bias of the estimator

We first decided to truncate the precision of the data with the round function of Python. Recall that we decided to take $T = 1000$. In order to generalize this to financial data that could be analyzed, it will be appropriate to reduce the time range to $T = 1000$ to compare with the results presented in this report. First, we set $\alpha = 0.6$. Then we rounded the values of the arrival times from 0 decimal places to 15 . We obtained the following results for the estimators of $\lambda_0, \alpha, \beta$.

| $\mathrm{n}_round$ | $\hat{\lambda_0}$ | $\hat{\alpha}$ | $\hat{\beta}$ |
|---|---|---|---|
| 15 | 1.024176295910912 | 0.9022450523726567 | 1.0066012595875753 |
| 14 | 1.0059472520268309 | 0.9009946586760513 | 1.0038649030957116) |
| 13 | 0.9981054964917139 | 0.9003175340415785 | 1.002568189656568 |
| 12 | 1.0070882829046848 | 0.9010349686405843 | 1.004038204446988 |
| 11 | 1.0241010861756725 | 0.9022235975122409 | 1.0065637208906544 |
| 10 | 1.0149606678091048 | 0.9015906556750841 | 1.0052045935711116 |
| 9 | 1.0055542924459449 | 0.9008306866100675 | 1.003554893373597 |
| 8 | 1.0150989654007059 | 0.9016375198670772 | 1.005238241222761 |
| 7 | 1.0058748675333025 | 0.9009150904315902 | 1.003766267886462 |
| 6 | 1.0241329582455732 | 0.9022450643488448 | 1.0065993928819594 |
| 5 | 1.0055642894285193 | 0.9008822421271322 | 1.003730119990898 |
| 4 | 1.0050206643604076 | 0.9010169647497115 | 1.0038226936858452 |
| 3 | 1.0151727888783 | 0.9016389120195971 | 1.005246860084167 |
| 2 | 1.0150909965137331 | 0.9018321305316858 | 1.005491424993567 |
| 1 | 3.5153541249198206 | 14649.325958032385 | 24841.87767672238 |
| 0 | 0.9999867705926502 | 105773.25221377045 | 729334.8799217555 |

Table 2 : Results of estimators for $\frac{\alpha}{\beta} = 0.6$ for different values of rounding

We observe that for $\frac{\alpha}{\beta} = 0.6$, the estimator behaves very well up to a rounding of 2 units. On the other hand, when the quality of the data is degraded further, the estimator behaves very badly.

### 3.1.2 Study of the mean square error of the estimator

Let's look at the standard deviation of this estimator in the same way to see if the rounding of the values influences it. For the same values as the previous table, we find the following results:

| $\mathrm{n}_round$ | $\mathrm{Stdev}(\hat{\lambda_0})$ | $\mathrm{Stdev}(\hat{\alpha})$ | $\mathrm{Stdev}(\hat{\beta})$ |
|---|---|---|---|
| 15 | 0.13638392839392027 | 0.04218732290583605 | 0.050577489398704496 |
| 14 | 0.19713603586618803 | 0.04347355146192414 | 0.05627038395695842 |
| 13 | 0.2199685109080231 | 0.044589189540269754 | 0.0594884089515575 |
| 12 | 0.1962056285399635 | 0.044140866544452866 | 0.057185562956676726 |
| 11 | 0.13634779897444882 | 0.042132454694469586 | 0.050513509479699625 |
| 10 | 0.16985161075341468 | 0.043030253676374496 | 0.053836518973286014 |
| 9 | 0.19729545158739048 | 0.04274969078752582 | 0.05523980845010386 |
| 8 | 0.16966738854715896 | 0.04270696251949367 | 0.05328082328141898 |
| 7 | 0.19716892671847397 | 0.04349441104698056 | 0.05630351975273242 |
| 6 | 0.13646991432013403 | 0.042162889534350705 | 0.05058324838170425 |
| 5 | 0.19731759809786809 | 0.043341080837748416 | 0.0560856664955203 |
| 4 | 0.19743569907844713 | 0.04355559786923395 | 0.05641758105512187 |
| 3 | 0.16967046466457025 | 0.04270992576763246 | 0.053274712781128455 |
| 2 | 0.16980882123682897 | 0.04299285642957189 | 0.05377643256527985 |
| 1 | 2.3236237033870077 | 40531.83194633352 | 70082.97767294706 |
| 0 | 0.1452882388008451 | 292258.7137368637 | 5124995.616614064 |

Table 3: Standard deviation of estimators for $\frac{\alpha}{\beta} = 0.6$ for different rounding values

We observe exactly the same thing as for the quality of the estimator. The standard deviation remains constant up to a rounding of 2. So we can degrade the quality of the data up to a certain point. We also tried to noisy the data with a Gaussian noise. We obtain exactly the same results as when we truncate. When the Gaussian noise is smaller than 10e-2, the estimator is not perturbed. On the other hand, when the noise is of the order of a unit or a tenth of a unit, the estimator no longer works.

The same calculations were performed for different values of $\frac{alpha}{\beta}$ and we find that the estimator degrades for higher values of rounding when the ratio increases.

## 3.2 Data quality according to a grid

The rounding precision is not precise enough. We will now analyze by splitting the grid into several points. We use the same values as before, and in particular $\mathrm{T} = 1000$. In view of the previous

part, we expect a degradation of the estimator between 1000 and 10000 points (which corresponds respectively to a rounding to the unit and to the tenth of unit).

How does this new way of degrading data work? It is very simple. We cut the time range (here between 0 and 1000) in a number of $n\_point$ previously defined. For example, for $n\_point = 1000$, this gives us the following grid for $\mathrm{T} = 1000 : [1, 2, 3, 4 \ldots .., 998, 999, 1000]$. Then each arrival time value is transformed into its closest value on the grid. For example, in our example, $567.48393857393$ will have the value 567. This allows us to have a finer step of data degradation than just rounding. Moreover, it allows us to adapt more easily to different values of T that we could meet in financial data. So we created a python script that we ran on a super-calculator (a little more than 3 days of calculation). We took the following values:

- $\lambda_0 = 1$
- $T = 1000$
- $\beta = 1$
- $\alpha = [0.5, 0.6, 0.7, 0.8, 0.82, 0.84, 0.86, 0.88, 0.9, 0.91, 0.92, 0.93, 0.94, 0.95, 0.96]$
- $n\_points = [1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 12500, 15000, 17500, 20000, 25000, 30000, 40000, 50000, 75000, 10000]$

### 3.2.1 Special case $\frac{\alpha}{\beta} = 0.6$

Let us first present the results for the ratio $\frac{\alpha}{\beta} = 0.6$. As we said before, we expect to see a degradation of the estimator between 1000 and 10000 points. Let's plot $\hat{\lambda_0} - \lambda_0, \hat{\alpha} - \alpha$ as well as $\hat{\beta} - \beta$ according to $n\_point$. For scaling purposes on the graphs, we put a max value for the error. What we are interested in here is when the value of the estimate becomes good and not how far away from the true value we are.

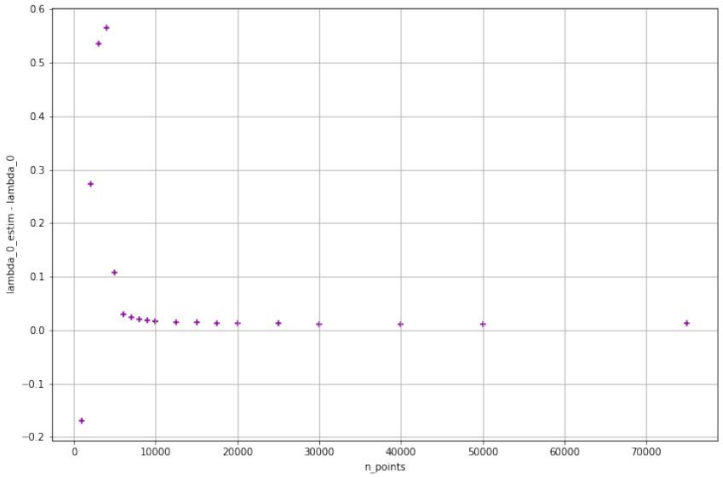


Figure 8: $\hat{\lambda_0} - \lambda_0$ as a function of $n\_point$ for $\frac{\alpha}{\beta} = 0.6$

On figures $8, 9, 10$ we see that for the 3 estimators, for a value of $\frac{\alpha}{\beta} = 0.6$, we need a minimum of 6000 points on the grid in order to keep an estimator which works. We can see that the quality of the data is important to have a good estimation of the parameters. This quality is of course directly proportional to the time range studied.

### 3.2.2 General case

Let's draw now the same graphs with different values of $\frac{\alpha}{\beta}$ in order to see if it influences the minimum value of points to have on the grid. We used a specific color code to recognize the curves. The smallest values of $\frac{\alpha}{\beta}$ are in purple and evolve progressively to dark red. Remember that for the sake

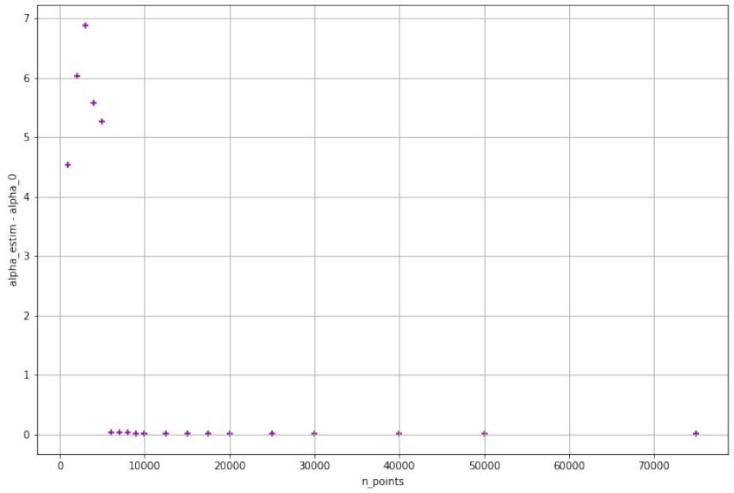


Figure 9: $\hat{\alpha_0} - \alpha_0$ as a function of $n\_point$ for $\frac{\alpha}{\beta} = 0.6$

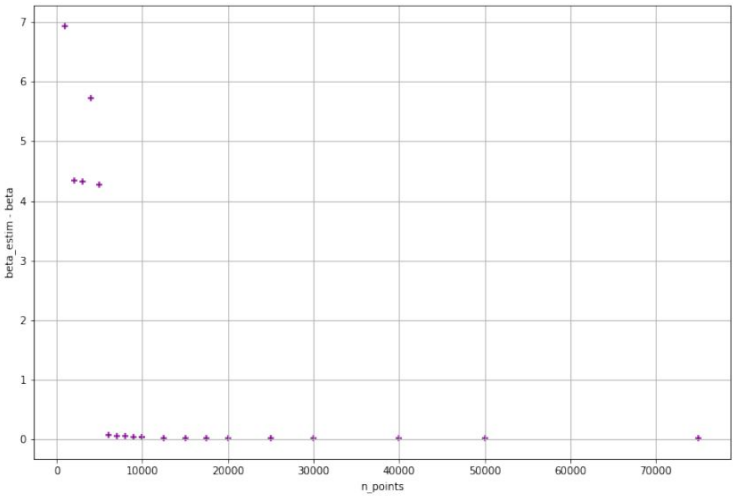


Figure 10: $\hat{\beta_0} - \beta_0$ as a function of $n\_point$ for $\frac{\alpha}{\beta} = 0.6$

of readability, we have captured the errors on $\hat{\alpha}$ and $\hat{\beta}$ at $0.2$ plus a Gaussian noise. Thus, all values greater than $0.2$ can in reality be much further away from $0$ . But here, we are just interested in knowing if the estimator is efficient or not.

For the 3 estimators, we can already notice that from a certain $n\_point$, the 3 estimators give a satisfactory result. More precisely, for $n\_point \geqslant 20000$, the estimators of $\lambda_0$ and $\beta$ work for any value of $\frac{\alpha}{\beta}$. For $n\_point \geqslant 30000$ , the estimator of $\alpha$ also works.

Another tendency can be observed: the higher the ratio $\frac{\alpha}{\beta}$ is, the more precise the data must be in order for the estimators to give a good result.

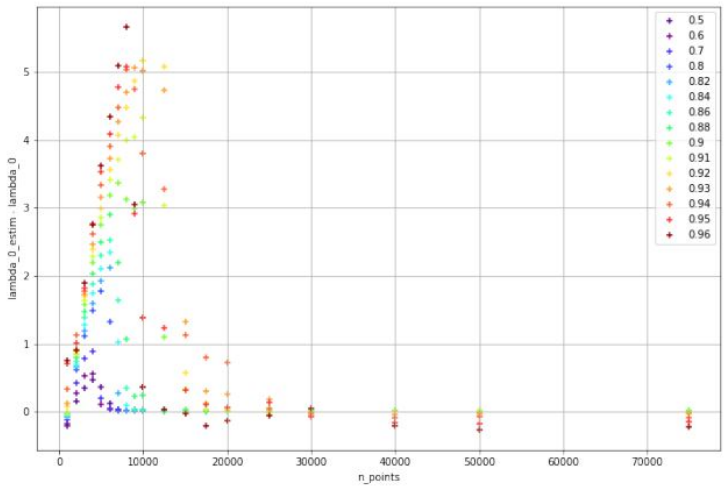


Figure 11: $\hat{\lambda_0} - \lambda_0$ as a function of $n\_point$ for different $\alpha$

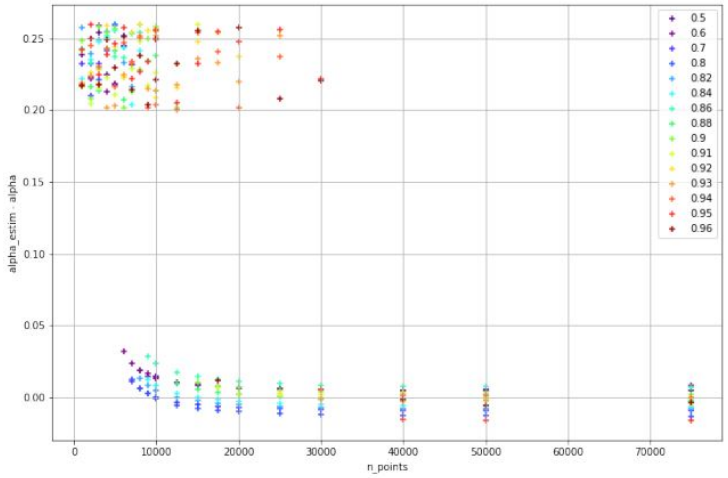


Figure 12: $\hat{\alpha_0} - \alpha_0$ as a function of $n\_point$ for different $\alpha$

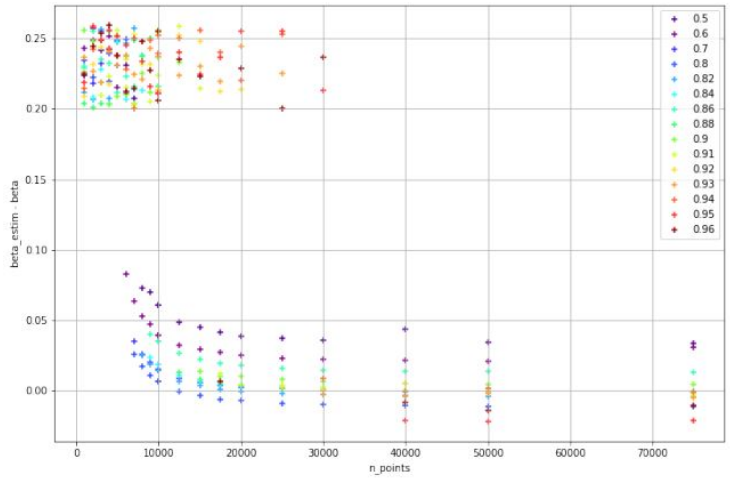


Figure 13: $\hat{\beta_0} - \beta_0$ as a function of $n\_point$ for different $\alpha$

## References

D. B. R. A. P. Dempster, N. M. Laird. Maximum likelihood from incomplete data via the em algorithm. In *Advances in neural information processing systems*.

D. S. V. Filimonov. Quantifying reflexivity in financial markets: towards a prediction of flash crashes. In *Physical Review 85.056108*, pages 13–14, 2021.

S. M. Marta C. Gonzalez, Sharon Xu. Modeling human dynamics and lifestyle using digital traces. pages 593–600, 2020.