HAWKES ITERATIVE TIME-DEPENDENT ESTIMATION OF PARAMETERS

NIELS D. C. KOTLAREK

Abstract. We delve into the question of estimating exponential kernel Hawkes processes' parameters with a time dependency, using weighted maximum likelihood estimators. We compute the gradient and the Hessian of an M-dimensional log-likelihood and generalise it to the weighted case to perform a time-dependent estimation. Weighting the MLE can be valuable in many settings. Instead of truncating the data to perform multiple estimations, one can choose how the data influences the estimate through the choice of a window. This method could improve other existing methods based upon MLE like splines approximations.

Window estimation is often flawed due to the non-linearity of the underlying temporal dynamics. We propose an adaptation of the original idea of Silverman Adaptive Kernel Density Estimation into an iterative algorithm named ITiDeEP (Iterative Time-Dependent Estimation of Parameters).

This method does not rely on the structure of Hawkes processes and could be used in other settings where window estimation is performed.

Keywords. Hawkes Process, Weighted Maximum Likelihood, Point Process, Iterative Width Window.

1. Motivation and Related Work

1.1. **Why Endo-Exogenous Models?** Some phenomena exhibit randomness that is self-exciting: each event triggers future events with greater probability. Examples of such are the activity on social media and other online communities (see [35, 56]), in disease studies (see [44]), criminology (see [55]), seismology / earthquakes (see [4, 55]), tsunamis, wild fires, economic contagion (see a review of financial applications for Hawkes processes [6, 29]; applications [22]; trading with exponential kernel in [18, 36], another one for limit order books in [46, 47, 59], for optimal reinsurance-investment problem in [10], Vix modelling in [32], financial bubbles and endo-exo problems [34, 40, 41, 63, 68]...). Self-excitation and clustering are philosophically related topics. For this reason, these phenomena are well described by processes with cluster representation, which in turn may be split into mixed endogenous and exogenous models.

1.2. **The Advent of Hawkes Processes.** The advent of self-exciting processes to model clustering comes back to professor Hawkes in 1971 (see [30]). Historically, the original Hawkes process was studied with an exponential kernel. Recently there has been an emerging trend to consider general Hawkes processes such as the non-linear Hawkes (see [9, 36]) or the fractional Hawkes processes (see [12]).

A brief historical summary can be found in [39]. A global summary of the trends and different ideas behind Hawkes processes lies in [36]. No paper offers a comparative study of the different methods and the pros/cons of each. A non-exhaustive list of papers presenting certain ideas about Hawkes processes: presentation of the characteristics of the exponential kernel in [17] and [18], derivation of the likelihood function for multivariate Hawkes processes in [14], the different thinning algorithms are summarised in [16].

We summarise the current general and growing interest in linear Hawkes processes by:

- Hawkes processes are a class of self-exciting processes that is not mathematically too complicated,
- the simulation and calibration of such processes is tractable.

Also, the two classes of Hawkes processes that are mainly considered in practice as well as in academia are the power-law decay and the exponential decay. The latter is numerically easier to manipulate (see: [49]) and is known to have a Markovian structure for the conditional intensity (see: [1, 23, 43]), and of big interest in a wide range of topics. For this reason, we focus on the exponential kernel. It is for example still the most popular kernel in finance, and sometimes used to approximate the power-law kernel: [24, 27, 66, 67].

Date: October 9, 2022.

ETH Zuerich; work realised during a *semesterarbeit*. I am grateful to Ed Cohen from Imperial College for introducing the topic, and Patrick Cheridito from the Risk Lab at ETH Zuerich for supervising the paper.

1.3. **Time-Dependent Hawkes Processes.** Time-dependent point processes have been popular for a long time, see: [48]. Recently, people have emphasised how much improvement adding time dependency gives, see: [54]. However, the first occurrence of such generalisation for Hawkes processes seems to be in [35] in 2016, where time-dependent exogenous dynamics (and random marks) are used. Now, it is common also to consider a branching ratio (or intensity of the jump in the conditional intensity) which accounts for time-dependent exogenous dynamics (see: [63, 68]). Finally, time-dependent decays are still an open research field, which comes with the difficulty of complex next jumps' time.

We should mention that alongside trying to model more complex phenomena with more expressive Hawkes models, some research is related to non-parametric Hawkes processes (see: [33]). Another angle to non-parametric modelling is offered by deep learning (see: [21, 51]).

1.4. **Estimation of Time-Dependent Parameters.** We focus in this paper on MLE estimation using the Hessian. Ozaki in [52] in Section 3 mentions three methods for finding the maximum likelihood that respectively use the: gradient and Hessian (Newton-Raphson method), gradient (Davidon's procedure), likelihood function (direct method). Some research suggests using only the gradient (see: [42, 46, 50, 52]), Least-square estimation [11] or otherwise relying on the method of moments. A variant of the MLE is the EM (Expectation Maximisation: [37, 65, 68])). In [11], the authors summarise the different methods of parameters estimation for linear multivariate Hawkes processes.

We propose here a framework for time-dependent estimation. Estimation of time-dependent parameters for Hawkes processes has not been yet studied extensively. It was common to use the simple approach of dividing the whole observation area into sub-intervals of a short period where one estimates the parameter as a constant (see: [25, 42, 50]). However, our approach uses windows: we can choose the impact of each data point on the final estimation. Such an approach is closer to reality and allows for a fast reinterpretation of the original data (instead of cutting the original data set, a filter is applied) and is more flexible than a simple truncation, allowing for more complex filters (like a bell-curved window). Also, our approach can be used with square-shaped kernels giving the previously mentioned subdivided estimation. Another approach consists of estimating the value on each of the subdivisions and making a linear combination of the different estimations (see: [50]), or more generally, using splines [64]. It is possible to use our approach to increase the performance of these methods.

A global wrap-up of the related theory of WMLE can be found in [62] and the weighted likelihood comes back up to Hastie, and Tibshirani in [5, 58, 28]. The idea of using windows to estimate parameters is known (see: [8, 26]), it is, however, primarily used for spatial estimation instead of temporal estimation. One technical paper [69] proposes different methods to compute the residuals and opens a general framework for spatio-temporal weighted parameter estimation, but it has not been catching a lot of attention so far. This can be because the expressions they propose are in most cases, not tractable (with integrals to compute in the likelihood function).

1.5. **Original Contribution.** Our goal is to simulate marked and followable[1] multi-dimensional Hawkes processes with exponential kernel "perfectly" and "efficiently"[2], for time-dependent parameters.

Also, there is no clear and complete presentation of how to perform MLE for Multivariate Hawkes processes with the exponential kernel, nor any mathematical presentation of point process' parameters' weighted estimation. Our approach is intrinsically based upon weighted maximum likelihood by weighting every contribution of the likelihood (in the framework of [69]). We weigh the contribution of the inputs (the jumps in the process) and, based on this reshaped information, estimate the parameters at a different time (for each time the information is reshaped differently) using a maximum likelihood estimation. The idea could be used more generally for point processes. The results are gathered inside Section 3.1, and the Theorem 3.1 states the dynamics of the likelihood under a weighted paradigm.

We agree on the weakness of window estimation that suffers from the classical problem of window's bandwidth choice. We offer an attempt to solve this issue with the ITiDeEP in Section 4.

The idea of adaptive/iterative estimation is not new, and a reference to this could be Silverman in [57]. A shorter paper summarises the idea of improving KDE in [31]. This method could also be used in spatio-temporal point processes.

[1]When one can determine what process triggered the jump; See Appendix B.

[2]A clarification for the terminology lies in Appendix B.

2. Theoretical Background on Point Processes

2.1. **Notation.** We introduce the notation that we will keep throughout the paper.

We are going to use the double bracket notation, which is referring to the interval intersected with integer numbers: $\forall\, M \in \mathbb{N}_{>0}:\ \{1, \cdots, M\} = [\![1, M]\!]$.

Also, for shortness, we denote for any process with left limits N:

$$N_{t^-} := \lim_{\substack{s \to t \\ s<t}} N_s.$$

We use this notation as well for measures and conditional intensity.

Finally, we will talk about multivariate point processes with realisations $(t_1^m, \cdots, t_{n_m}^m)$; following the convention from [19] (page 269), we call the record of all the jumps' time generated by the stochastic processes "list history". Here, the process is $M \in \mathbb{N}_{>0}$ dimensional and $m \leq M$. In the case of multivariate marked point processes, the latter depends on the record of times and potentially on the marks too. For them, we include the previous marks inside the list history. We will write as a short-cut $\mathcal{T}^m_{t^m_{n_m}}$ or more conventionally, $\mathcal{F}^m_{t^m_{n_m}}$ as being the filtration generated by the conditional intensity of the m-th process, up to time $t^m_{n_m}$.

$\mathcal{T}^m$ shall correspond to the limiting case of the filtration, which then would correspond to the complete information we possess.

We will keep T as the upper bound of the observed window. Since we simulate processes on the finite time interval $[0, T]$, then $\mathcal{T}^m$ would represent the filtration generated by the conditional intensity of the m-th process on that interval.

The natural extension from marginal to global filtration is $\mathcal{F}_t := \mathcal{T}_t := \bigotimes_m \mathcal{T}^m_t$.

2.2. **Point Process Theory.** For a more complete description of Point process theory, we recommend the book from Daley [19]. Hereinafter, we give a small insight on some definitions that are at the core of the idea of Hawkes processes.

Definition 2.1 (Counting Process)**.** A **counting process** on a time interval I, $\{N_t\}_{t\in I}$, is a stochastic process satisfying the following conditions:

- $\forall\, t \in I, N_t \in \mathbb{N}_{i>0}$, (well defined)
- $N_0 = 0$, (starts at zero)
- $\lim_{s\to t^-} (N_t - N_s) \in \{0, 1\}$. (piecewise constant, incremental as well as orderly jumps and right-continuous)

Definition 2.2 (Point Process)**.** A sequence of random variables $T = \{T_i\}_{i\in\mathbb{N}_{\geq 0}}$ is called a **point process** on I if it satisfies the following conditions:

- $\forall\, i \in \mathbb{N}_{i>0}, T_i \in I$ a.s., (well defined)
- $T_i \leq T_j, \forall\, i \leq j$ a.s., (monotonicity).

We fix $T_0 = 0$.

Remark 2.3. The sequence of differences in times, $\tau_i = \{T_{i+1} - T_i\}_{i\in\mathbb{N}_{>0}}$ is called the **inter-arrival time sequence**.

Definition 2.4 (Simple Point Process)**.** A sequence of random variables $T = \{T_i\}_{i\in\mathbb{N}_{\geq 0}}$ is called a **simple point process** on I if it satisfies the following conditions:

- T is a point process on I as given by Definition 2.2, (point process)
- $\mathbb{P}(T_{i+1} - T_i = 0, \forall\, i \in \mathbb{N}_{\geq 0}) = 0$, (simple increment property).

Definition 2.5 (Marked Point Process)**.** We refer to a stochastic process N as a **marked point process**, with locations in $\mathbb{R}$, marks in $\mathcal{K} \subset \mathbb{R}$, whenever it is a point process on $\mathbb{R} \times \mathcal{K}$ with the property that the marginal process of locations, denoted N_g for ground process, is itself a point process:

$$\forall\, A \in \mathcal{B}(\mathbb{R}),\ \text{bounded},\ N_g(A) = N(A \times \mathcal{K}) < \infty. \tag{2.1}$$

Then we write the sequence of realisation as $\{(x_i, \kappa_i)\}_{i\in\mathbb{N}_{\geq 0}}$.

Multivariate point processes are the direct extension of the previous definitions. We write a d-variate point process N as:

$$N_t = \left(N_t^1, \cdots, N_t^d\right).$$

Definition 2.6 (Conditional Intensity Function)**.** When $N_\cdot$ is a point process with natural filtration $\mathcal{F}_\cdot$, we call the left-continuous and adapted process, "stochastic conditional intensity function of the point process", defined as:

$$\lambda(t \mid \mathcal{F}_{t^-}) = \lim_{h \to 0^+} \frac{\mathbb{P}(N_{t+h} - N_t > 0 \mid \mathcal{F}_{t^-})}{h} = \frac{\mathbb{P}(dN_t > 0 \mid \mathcal{F}_{t^-})}{dt}, \tag{2.2}$$

which for simple point processes can be rewritten in terms of a conditional expectation:

$$\lambda(t \mid \mathcal{F}_{t^-}) = \lim_{h \to 0^+} \frac{\mathbb{E}(N_{t+h} - N_t \mid \mathcal{F}_{t^-})}{h} = \frac{\mathbb{E}(dN_t \mid \mathcal{F}_{t^-})}{dt}. \tag{2.3}$$

Hereinafter, the conditioning is implied in the function λ for conciseness.

2.3. **Time Dependent Hawkes Processes.** We focus on the exponential kernel. The rationale for studying such a kernel lies in the easiness of understanding and simulating such a process and its robustness in approximating real-life phenomena. Defined by Pr. Hawkes in his paper [30] (also in [14]), we have for $M \in \mathbb{N}_{>0}$:

Definition 2.7 (non-homogeneous Hawkes process with exponential decay)**.** Let $N_t = \left(N_t^1, \cdots, N_t^M\right)$ be a simple multivariate counting process whose conditional intensity function satisfies for all $m \in [\![1, M]\!]$ as a left-continuous adapted stochastic process λ^m given by the Stieltjes integral:

$$\begin{aligned} \lambda^m(t \mid \mathcal{T}_{t^-}) &= \nu_m(t) + \sum_{n=1}^{M} \int_0^t \mu_{m,n}(t-s)\ dN_s^n \\ &= \nu_m(t) + \sum_{n=1}^{M} \sum_{\{k : t_k^n < t\}} \alpha_{m,n}(t_k^n) e^{-\beta_{m,n}(t_k^n)\cdot(t - t_k^n)}. \end{aligned} \tag{2.4}$$

μ is a kernel function called Hawkes' excitation function[3]. The usual condition on these parameters is that they are piecewise continuous. Essentially, we assume that parameters are smooth enough, so the simulations are correct so that the data reflects the underlying parameters and the estimations possible.

3. Weighting the MLE

3.1. **Estimation of Hawkes' Parameters with WMLE.** The process can be multivariate, and we would like:

- to estimate the parameters θ by $\hat{\theta} = (\hat{\nu}, \hat{\alpha}, \hat{\beta})$. θ shall belong to a set of possible parameters, named Θ.
- to estimate the parameters with some temporal dependency on them. We then replace our estimate by $\hat{\theta}_t$,

Since the log-likelihood of Hawkes processes is non-linear, we would like to use a non-linear optimisation technique. We are going to use the Newton-Raphson method. Even though it might be more expensive to compute the Hessian, it generally leads to better estimators.

For that purpose, we need to compute numerically the gradient and the Hessian of the log-likelihood. The formulas can be found in Appendix C.

3.2. **How to Weight the Likelihood?** The formula of likelihood for Point processes is well known (derived e.g. in [19] in 7.2 III p. 232 or in 7.3 III p. 251 for Marked Point processes). Thanks to it, we derive the log-likelihood function for Hawkes processes (shown in [14, 18]). It reads:

$$\log L^m(\Theta \mid \mathcal{T}) = -\int_0^T \lambda^m(s \mid \Theta, \mathcal{T}_s)\ ds + \int_0^T \log \lambda^m(s \mid \Theta, \mathcal{T}_s)\ dN_s^m. \tag{3.1}$$

This formula allows us to estimate the constant parameters of a (homogeneous) Hawkes process.

[3]In this paper, we focus on a common choice, the exponential case, where $\mu(t) = \alpha_{m,n}(t) e^{-\beta_{m,n}(t)t}$.

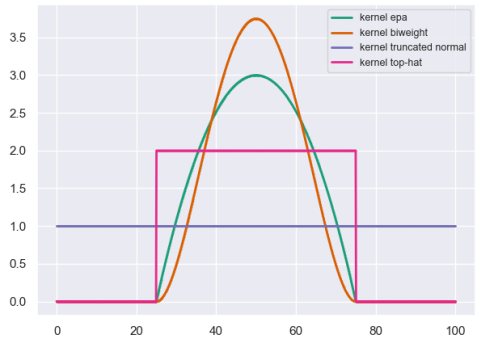

FIGURE 1. Various shapes for the kernel. Here the kernel w_τ is drawn for $\tau = 50$. We did not observe a noticeable impact of the shape upon the estimation's error. Numerically, one needs to be careful about edge cases.

It is, however, difficult to account for the non-homogeneity of the process in the likelihood. Instead of finding another way to estimate the parameters, we propose to shift the time dependency from the parameters to a window: by this way, we can use the same tools, namely the MLE, but with a time dependency. The window weighs the information of the likelihood relatively to how far (temporally) the data is to the temporal point when we are estimating the parameters. A window is centred around some point τ, which is the time point when the parameters are estimated. Fig. 1 shows an image of kernels. The weighted maximum likelihood estimators should give a good estimation of the parameters at the time τ: Θ_τ. Mimicking what was done in Section II from [3], and by coining the weight function w_τ (also called in the following kernel), we define the log weighted likelihood (coined $W_\tau L$) as:

$$\log W_\tau L^m(\Theta \mid \mathcal{T}) := -\int_0^T w_\tau(s)\ \lambda^m(s \mid \Theta, \mathcal{T}_s)\ ds + \int_0^T w_\tau(s)\ \log \lambda^m(s \mid \Theta, \mathcal{T}_s)\ dN_s^m. \tag{3.2}$$

Here the notation $W_\tau L$ should be understood as one word.

Essentially, we want to take advantage of the fact that a larger weight $w_\tau(s)$ gives observations in the infinitesimal region ds more influence during the estimation process.

The original part of this theorem is that we add in the equations a weighting parameter. The Likelihood becomes a Weighted Likelihood.

Theorem 3.1. *(Quoting [19] in 7.2 III p232): If N is a simple point process on $[0,T]$, $0 < T < \infty$, and we are aware of the realization of N over $[0,T]$ as being $\{t_i\}_{i \in \mathbb{N}_{>0}}$, then the likelihood L of N reads:*

$$L(\theta \mid \mathcal{T}) = \exp\left(-\int_0^T \lambda_\theta(s \mid \mathcal{T}_s) ds\right) \prod_{i=1}^{N(T)} \lambda_\theta(T_i = t_i \mid \mathcal{T}_{t_{i-1}^-}). \tag{3.3}$$

Following this, the likelihood function of the multivariate Hawkes process is given by the following equation:

$$\log W_\tau L(\theta \mid \mathcal{T}) = \sum_{m=1}^{M} \log W_\tau L^m(\theta \mid \mathcal{T}), \tag{3.4}$$

Details about it can be found, for example,e in [14].

Finally, our contribution would be that we completely characterise the likelihood of a multivariate Hawkes process with a time-dependent exponential kernel.

$$\log W_\tau L(\theta \mid \mathcal{T}, \lambda^m) = -\nu_m(\tau)T - \sum_{n=1}^{M} \frac{\alpha_{m,n}(\tau)}{\beta_{m,n}(\tau)} \sum_{\{k:t_k^n<T\}} w_\tau(t_k^n) \left(1 - e^{-\beta_{m,n}(\tau)\cdot(T-t_k^n)}\right)$$

$$+ \sum_{\{k:t_k^m<T\}} w_\tau(t_k^m) \log\left(\nu_m(\tau) + \sum_{n=1}^{M} \sum_{\{i:t_i^n<t_k^m\}} \alpha_{m,n}(\tau)\cdot e^{-\beta_{m,n}(\tau)\cdot(t_k^m - t_i^n)}\right). \tag{3.5}$$

Even though we rely on an approximation in the proof, the resultant estimator is unbiased and shows the same characteristics as the classical MLE, which can be observed empirically for linear parameters estimation, or big datasets.

Remark 3.2. Some papers also shift this log-likelihood by a constant T, as in [17, 59]. However, the formula proven in [19] is correct and should be the one used.

Remark 3.3. Weighting should be a way to rescale the information at our disposal. For that reason, we restrict the set of considered kernels to be of norm T:

$$\Omega_w := \{w\colon \mathbb{R} \to \mathbb{R},\ \text{such that}:\ \|w\|_{L^1} = T\}.$$

We do not need any other restriction on the considered functions. However, a customary choice is fixing the positivity of the kernel and its first moments for identifiability reasons.

Proof. We prove for the univariate case but modulo some notational changes, the proof is essentially the same in the multivariate case. The first thing is that we approximate the function parameters $(\alpha, \nu, \beta)(t)$ by the function evaluated at the point when we estimate the parameters $(\alpha, \nu, \beta)(\tau)$. From this, it appears that if we simply used the classical likelihood formula, we would get an averaged value as a result. Then, from (3.2):

$$\begin{aligned}\log W_\tau L(\Theta \mid \mathcal{T}) &= -\int_0^T w_\tau(s)\ \lambda(s \mid \Theta, \mathcal{T}_s)\ ds + \int_0^T w_\tau(s)\ \log\lambda(s \mid \Theta, \mathcal{T}_s)\ dN_s \\ &=: -\Lambda_T + \sum_{\{k:t_k<T\}} w_\tau(t_k)\ \log\left(\nu_m(\tau) + \sum_{\{i:t_i<t_k\}} \alpha(\tau) e^{-\beta(\tau)\cdot(t_k - t_i)}\right),\end{aligned}$$

(using the classical notation Λ of the compensator). We rewrite the compensator as:

$$\begin{aligned}\Lambda_T &= \int_0^T w_\tau(s)\ \lambda(s \mid \Theta, \mathcal{T}_s)\ ds \\ &= \int_0^T \nu(\tau)\cdot w_\tau(s)\ ds + \sum_{\{i:t_i<T\}} \int_{t_i}^{t_{i+1}} \sum_{\{j:t_j<s\}} w_\tau(s)\cdot\alpha(\tau)\cdot e^{-\beta(\tau)(s-t_j)}\ ds \\ &= T\nu(\tau) + \alpha(\tau) \sum_{\{i:t_i<T\}} \sum_{\{j:t_j<t_i\}} \int_{t_i}^{t_{i+1}} w_\tau(s)\cdot e^{-\beta(\tau)(s-t_j)}\ ds.\end{aligned}$$

The integral inside the double sum might be difficult to compute, and numerical integration would add up to the already high cost of the optimiser.

A natural approximation for $w_\tau(s)$ is $w_\tau(t_j)$ since it is the position of the jump for which we are considering the impact.

$$
\begin{aligned}
\Lambda_T &\approx T\nu(\tau) + \alpha(\tau) \sum_{\{i:t_i<T\}} \sum_{\{j:t_j<t_i\}} \int_{t_i}^{t_{i+1}} w_\tau(t_j) e^{-\beta(\tau)(s-t_j)}\, ds \\
&= T\nu(\tau) + \alpha(\tau) \sum_{\{i:t_i<T\}} \sum_{\{j:t_j<t_i\}} w_\tau(t_j) \int_{t_i}^{t_{i+1}} e^{-\beta(\tau)(s-t_j)}\, ds \\
&= T\nu(\tau) - \alpha(\tau)/\beta(\tau) \sum_{\{i:t_i<T\}} \sum_{\{j:t_j<t_i\}} w_\tau(t_j) \left(e^{-\beta(\tau)(t_{i+1}-t_j)} - e^{-\beta(\tau)(t_i-t_j)} \right) \\
&= T\nu(\tau) - \alpha(\tau)/\beta(\tau) \sum_{\{j:t_j<T\}} \left(w_\tau(t_j) \cdot e^{t_j} \right) \left(e^{-\beta(\tau)\cdot T} - e^{-\beta(\tau)\cdot t_j} \right) \\
&= T\nu(\tau) - \alpha(\tau)/\beta(\tau) \sum_{\{j:t_j<T\}} w_\tau(t_j) \left(e^{-\beta(\tau)(T-t_j)} - 1 \right).
\end{aligned}
$$

□

Remark 3.4. Notice that if all the weights are equal to unity, the resulting estimator is the usual maximum likelihood estimator, i.e. if $w_\tau \equiv 1$, then (3.2) reduces to (3.1). In this case, the norm of the weights is equal to T.

The idea behind using weighted likelihood is that by using a non-constant weight function, solving for the maximum likelihood estimator given by (3.2) will grant us the ability to estimate time-dependent parameters. In other words, despite using the likelihood for constant parameters, we are introducing a time dependency in the likelihood through a weighting function, the dependency being hidden behind the choice of the kernel. The gradient and Hessian of the likelihood are derived in Appendix C.

3.3. **Discussion about the Choice of the Kernel.** We would aim to find the best kernels for our task. If we want to find the estimates at a sequence of time $\{t\}_{t\in\mathcal{I}}$ where the index set is of length n, we need to find n kernels. Let's reflect on the two extreme cases for weights:

(1) A constant kernel considers all the jumps equally and hence computes an estimator that estimates the average over the whole simulation time $[0, T]$. Overall, the bias will be high, but the variability low.
(2) A kernel that considers only a small region will give a very precise estimate for that given region, but with much more variability over time.

This phenomenon is due to the well-known bias/variance trade-off.
The measure of error is commonly the so-called mean integrated square error:

$$
\mathrm{MISE}(\hat{f}) \triangleq \mathbb{E}[\int (\hat{f}(u) - f(u))^2\, du]. \tag{3.6}
$$

The theory of KDE informs us about the best kernels regarding the MISE.

It is possible to derive the optimal kernel for KDE (see: [61]). One can separate the dependency of the bandwidth and the kernel by a substitution. This showed that the shape of the kernel is not crucial for the estimation[4]. As a summary of the extended literature on kernel choice, the best choice would be the following class of kernels:

$$
K_a^*(t) := \frac{3}{4} \frac{1 - \frac{t^2}{5a^2}}{\sqrt{5}a} \mathbb{1}_{\{|t|\le\sqrt{5}a\}}.
$$

We usually refer to this class of kernel as the Epanechnikov kernels. It is usual to fix $a = 1$ and rescale the kernel such that it is defined on $|t| \le 1$, yielding:

$$
K^*(t) := \frac{3}{4}(1 - t^2)\, \mathbb{1}_{|t|\le 1}.
$$

[4]Theoretical fact that we observed while estimating different parameters, for the same bandwidth but different shapes of kernels.

Another popular scaled kernel for its regularity, with almost equal performances (see: [61]) is the bi-weight kernel defined as:

$$K_b(t) := \frac{15}{16}(1-t^2)^2 \mathbb{1}_{|t|\leq 1}\,. \tag{3.7}$$

It is the one we are going to use in the following.

The optimal width is more complicated to derive (perhaps impossible to derive without prior information about the parameters). The issue with the bandwidth in our scenario is that the kernel is first shaping our input data given to the optimiser algorithm. Thus, the relationship between the estimate and the kernels is more ambiguous.

Due to this uncertainty in the choice of the kernel's width, we propose an idea to solve this issue in the next section.

4. Iterative Width Window: ITiDeEP

4.1. **Idea Behind AKDE.** Historically, iterative width window choice comes from adaptive kernel density estimation (AKDE) and goes back to Silverman [57].

By using an iterative procedure, we should be able to find a better set of bandwidths for our kernels. The author refers to [31, 60] for some recent examples of AKDE in the literature. The classical method is a two-step procedure, where the first estimate gives some insights on what should be the optimal size of the kernels.

We write the idea of the historical algorithm, given we wish to estimate our function at the points of the sequence $\{t_i\}_{i\in\mathbb{N}_{\geq 0}}$:

Algorithm 1: Adaptive Kernel Estimation.

1. Find a pilot estimate coined $\widetilde{f}$. For a kernel w^* and a bandwidth h^*, the weighting is of the form
$$(t,t_i) \to \frac{1}{h^*} w^*\left(\frac{t-t_i}{h^*}\right).$$
2. $\forall\, t_i \in \{t_i\}_{i\in\mathbb{N}_{\geq 0}}$, after estimating the pilot estimate $\widetilde{f}$, create the local width factor λ_{t_i} according to (4.1).
3. Find the final estimate $\hat{f}$ by using a different kernel for each t_i. $\forall\, t \in \{t\}_{t\in\mathcal{I}}$, the new kernel shall be of the form:
$$(t,t_i) \to \frac{1}{\lambda_{t_i} h^*} K^*\left(\frac{t-t_i}{\lambda_{t_i} h^*}\right).$$

The interpretation of the scaling coefficient $\{\lambda_{t_i}\}_{i\in\mathbb{N}_{\geq 0}}$ can be stated as: the bigger the coefficient, the more the bandwidth is scaled up. A coefficient bigger (resp. smaller) than one is equivalent to widening (resp. tightening) the relative kernel.

The first accepted function yielding the scaling factors is the following. A brief reminder, the geometric mean for a sequence $\{a_i\}_{i\in\mathbb{N}_{\geq 0}}$ is commonly defined as

$$G_a := \left(\prod^{n} a_i\right)^{\frac{1}{n}},$$

and we define the sensitivity parameter satisfying $0 \leq \gamma \leq 1$. Then the scaling function is:

$$\forall\, t_i \in \{t_i\}_{i\in\mathbb{N}_{\geq 0}}, \qquad \lambda_{t_i} := \left(\frac{\widetilde{f}(t_i)}{G_{\widetilde{f}_i}}\right)^{-\gamma}, \tag{4.1}$$

where many sources recommend using $\gamma = \frac{1}{2}$, the so-called square root law (found in the book [57] or in the article [2]).

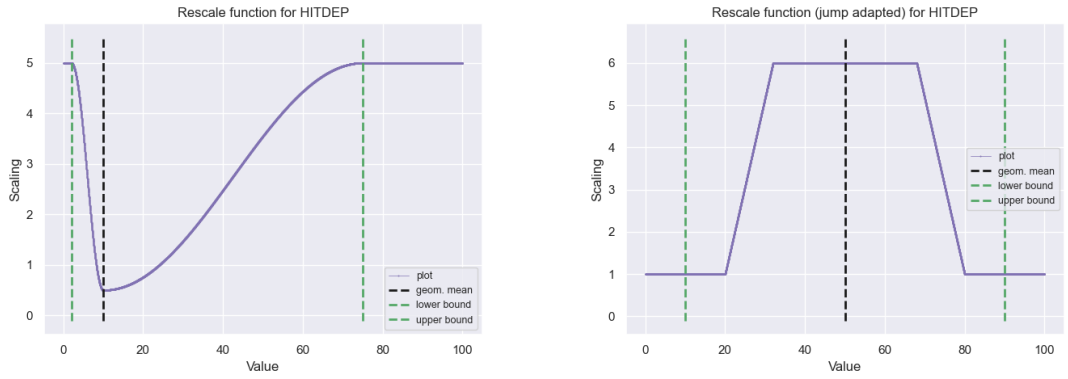

FIGURE 2. Left figure: Scaling output depending on how close the estimate is to the geometric mean. We called this function "g". The lower branch ends at 2, the mean is 10 and the upper branch finishes at 75. We constructed the function as a two sinusoidal branches function. Right figure: Scaling output depending on how close the estimate is to the geometric mean. The lower branch ends at 10, the mean is 50 and the upper branch finishes at 10. We constructed the function as a simple plateau.

4.2. **How to Rescale the Kernels?** Scaling could potentially be done either by

- a local argument: how much the function varies locally,
- a global argument: the one from AKDE, where we compare the current value with a global mean.

This paper focuses on the latter, even though it does not fit every case scenario.

Indeed, scenarios where a local argument would be better are for example functions with seasonality.

On the contrary, functions with simpler behaviours can be grasped with a global argument. Philosophically speaking, a global argument can work whenever it is possible to categorise each part of the function in either high, low or medium height and with little transition between these phases. Examples of such are a jump function with two heights, a sinus function (periodic) or a low-degree polynomial.

4.3. **Choice of a Rescale Function.** Restating the idea into words: an initial estimation with fixed width is computed, which gives a first idea of the time-dependent parameters. Then, we compute new bandwidths for a second estimation with the pilot estimate.

In the previous algorithm, by observing a first estimate of the density, there was an adaptive algorithm that could automatically rescale the windows and give more precise results.

However, for parameter estimation in WMLE, an idea would be to scale kernels depending on whether they are extremes by narrowing them and broadening kernels at points close to the average. This seems to be a natural choice, which would work very well in the case of constant parameters (where we would want broad kernels) and in the case where there would be a unique high and low peak in the estimation, in which case we would benefit from tighter bandwidths. In other words, the average behaviour is in the middle, with some extreme peaks (high or low) that are localised.

Even though the logic behind the algorithm is the same, one should choose with special care a rescale function that suits their needs. Here, we propose two rescale functions. The first one is the one described earlier, and another one is specially tailored for jump functions. Their outputs are both visible on Figure2.

General g rescale function:

$$g(x, G, L, R, h, l) = \frac{h-l}{2}\left(2 - \mathbb{1}_{[-\pi,0]}(u)\cos(u) - \mathbb{1}_{[0,\pi]}(v)\cos(v)\right) + l, \tag{4.2}$$

$$u = \pi\frac{x-G}{G-L},$$

$$v = \pi\frac{x-G}{R-G}$$

For that function, G is the junction point between the two branches. L and R are the ends of the varying part and hence outside of the interval $[L, R]$ the value of the function g is fixed. It is clear that L < G < R for the function to be well defined.

G shall correspond to the geometric mean of the pilot estimate, L and R are taken in the following as some quantiles of the data. h, l are added as the parameter of scaling. Recall that p-quantile for an ordered series of length n is defined as being equal to its $\lceil np \rceil$-value[5].

4.4. **Hawkes Iterative Time-Dependent Estimation of Parameters.** We now present the algorithm as a whole: the new algorithm is called ITiDeEP[6]. We keep the basic idea of the previous algorithm: we rescale the kernels based upon a function g, which takes as parameters one element of an array and the geometric mean.

Following the Algorithm 1, we write our final version of the adaptive weighting algorithm. The major changes that we apply are:

- rescale the parameters if necessary, according to the equation (4.3), for each dimension, excluding the dimensions that are non evolving[7]. The scaling we recommend is the positive mean-scaling:

$$\forall\, i \in [\![1, M]\!], \qquad \widetilde{\theta}_i = \frac{\theta_i - \bar{\theta}}{\max(\theta) - \min(\theta)} + 1. \tag{4.3}$$

  Not scaling the parameters could lead to the classic problem of data sciences related to feature standardisation. The features with high magnitudes will weigh much more in the distance calculations than features with low magnitudes, particularly with the Euclidean distance.
- Usage of the norm upon the pilot estimator $\theta^*_\cdot \in \mathbb{R}^{M+2M^2}$. We compute the geometric mean on $\theta^*_\cdot$'s norm.

The algorithm becomes:

Algorithm 2: Iterative Time-Dependent Estimation of Parameters (ITiDeEP).

1 Find a pilot estimate coined $\widetilde{\theta}_t$. For a kernel w^* and a bandwidth h^*, the weighting is of the form

$$(t, t_i) \to \frac{1}{h^*} w^* \left(\frac{t_i - t}{h^*} \right),$$

2 $\forall\, t \in \{t\}_{t \in \mathcal{I}}$, when we estimated the pilot estimate $\widetilde{\theta}_t$, create the local width factor λ_t according to:

$$\forall\, t \in \{t\}_{t \in \mathcal{I}}, \qquad \lambda_t := g\left(\|\widetilde{\theta}_t\|_{l^2}, G_{\|\widetilde{\theta}_\cdot\|_{l^2}} \right), \tag{4.4}$$

and $G_{\|\widetilde{\theta}_\cdot\|_{l^2}}$ is based upon the sequence $\{\|\widetilde{\theta}_\cdot\|_{l^2}\}$.

3 Find the final estimate $\hat{\theta}_t$ by using the rescaled kernels for each different time t. $\forall\, t \in [0, T]$, the new weight shall be of the form:

$$(t, t_i) \to \frac{1}{\lambda_t h^*} w^* \left(\frac{t - t_i}{\lambda_t h^*} \right).$$

Remark 4.1. The algorithm we proposed creates a single rescale parameter for all dimensions. In other words, we use windows that have a fixed bandwidth with respect to the dimensions of the process. However, a simple adaptation of it generating a scaling per dimension of the observed data would also be possible.

We recommend taking h in the interval $[1, 5]$ depending on the amount of data one has. The smaller the window, the more imprecise the estimate; In fact, it makes sense that l changes depending on the width of

[5] $\lceil \cdot \rceil$ is the ceil-function.

[6] ITiDeEP stands for Iterative Time-Dependent Estimation of Parameters. When applied to Hawkes estimation, we might call it the HITiDeEP.

[7] A problem appears with the estimation of constant parameters. In such a case, the time series of means is not indicating any tendency but is essentially noise. A solution we implemented is to not consider the evolution of parameters with no significant change, where our criterion considers how much variation is inside the vector of the different time estimations. We undergo this test for each dimension $(M + 2M^2)$ of the estimator.

Parameter	Sinus	Jump	Multivariate Mountain
Rescale Function	General g	Rescale Jump	General g
Number of Simulations and Estimation	150	150	150
T max	15 000	12 000	20 000
Initial Denominator of T max used as bandwidth	4	3	3
L	0.04	0.13	0.10
R	0.90	0.98	0.95
h	3.00	3.00	3.00
l	0.90	0.70	0.70

TABLE 1. Set of parameters for simulations and estimations using ITiDeEP.

the first kernels, so that the widest second step kernels cover the whole observation period. Hence, we use

$$l = \frac{\text{first width}}{2 * \text{T max}},$$

with such value, the widest second step kernels get the width T, length of the observation period.

L and R and quantiles should be chosen according to a guess about the shape of the function. 4% and 96% are good in practice, but should be increased (resp. lowered) if there are relatively many values (low values) of the type "small" for L and "high" for R.

The function g is flat on the sides and for that reason, choosing 2% and 98% could also be done.

One can see how the function g looks like on Figure 2.

5. Numerical Results

We verify the results in the following way. We consider different underlying parameters functions in the univariate and bivariate cases. We show the difference between a first and second estimation with respect to the same initial data (time series and estimations are available in the repository). We performed an estimation over the same[8] different realisations of the dynamics (in a Monte Carlo fashion), and we plot the mean estimation as well as 95% confidence interval on the plots (computed automatically with bootstrapping in the python package seaborn).

We also show how the global error is impacted by the rescaling of the window depending on the number of observed events.

For the sake of clarity, we present the two estimations on the same plot. We expect a reduction in the bias of the estimation at the price of higher variance. Also, the MLE estimation can become very difficult when the underlying parameters vary inside the observation window, leading to convergence failures. Hence, adapting the window to the variation of the function should also reduce the number of estimation failures.

The parameters used are gathered in Table 1.

We would like to thank the Cluster Euler available for ETH students, which hosted the simulations and estimations. The available resources reduced drastically the computational time. A repository with the code of the work can be found on Github[9].

5.1. **Uni-variate Estimation: Example of the Sinus Function.** We first consider the sinus function. There are multiple interests for such function: sinus function epitomises a recurrent behaviour whose peaks' values are constant through time and for which the choice of the bandwidth is not obvious. Results are visible on Figure 3. One should remark that the estimates are more precise and more estimations are successful: whereas the first estimation yields 18.65 failures in average per time estimation (for 150 estimations). After rescaling the kernels, the number of failures fell to 16.3 (for 150 estimations).

[8]We reuse the same data for the different methods.

[9]https://github.com/Code-Cornelius/ITiDeEP

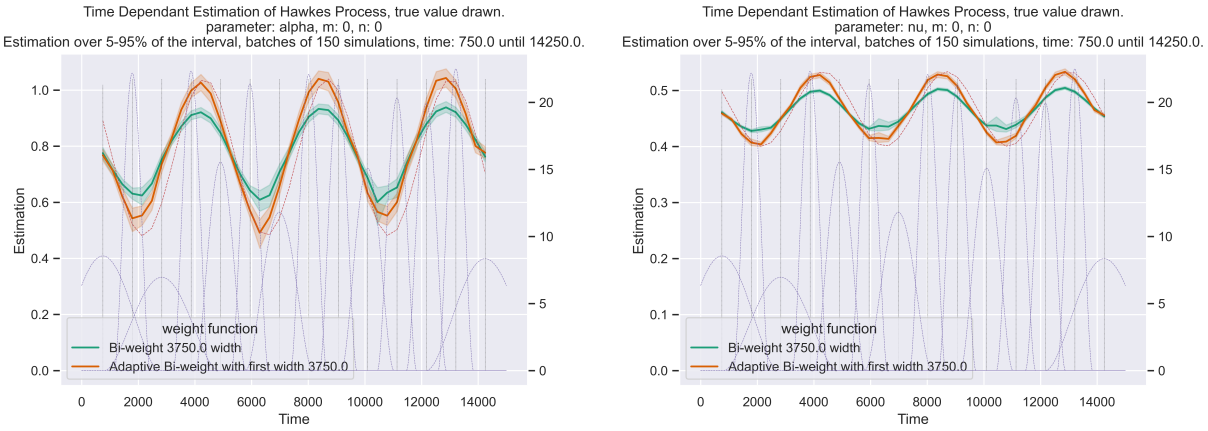

FIGURE 3. Comparison results of estimation before and after rescaling the bandwidths with ITiDeEP for uni-variate Hawkes. The underlying parameters follow a sinusoidal growth. We do not show beta's change in estimation since its value is constant.

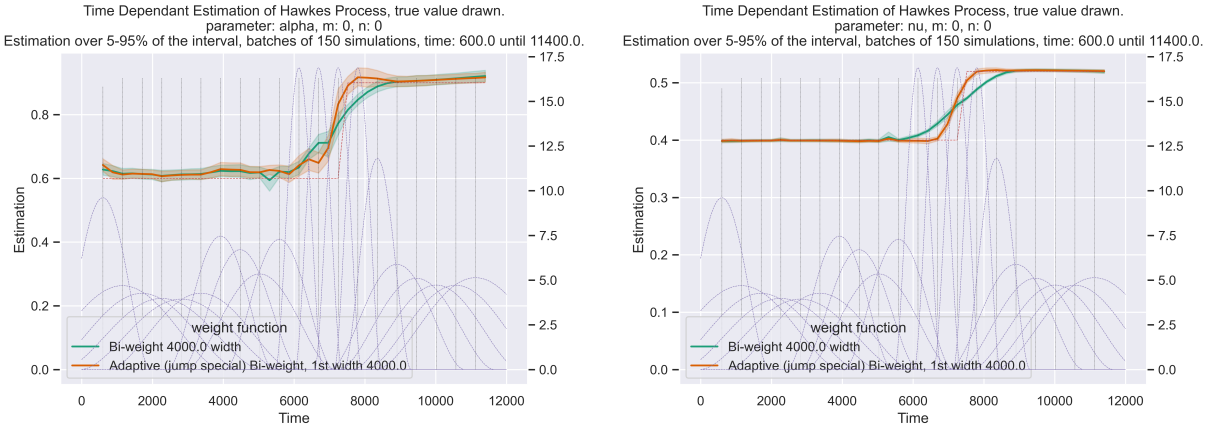

FIGURE 4. Comparison results of estimation before and after rescaling the bandwidths with ITiDeEP for uni-variate Hawkes. The underlying parameters follow a jump. We do not show beta's change in estimation since its value is constant.

5.2. **Uni-variate Estimation: Example of the Jump Function.** We now consider a jump function. The general rescale function is not suited for these kinds of processes, and we use the special jump rescale function visible in Figure 2. The other function focuses on making jumps tighter for "transition" points. One should remark that the estimates are more precise and estimation are successful: whereas the first estimation yields 7.75 failures in average per time estimation (for 150 estimations). After rescaling the kernels, the number of failures fell to 6.625 (for 150 estimations).

Hawkes processes with jumps in the parameters' functions are rampant and that is the reason why we focus on such processes.

5.3. **Bi-variate Estimation: Example of the Mountain Function.** Finally, we consider a bi-variate function where the parameters evolve with respect to a "mountain" dynamic. We decided to use such dynamics because it represents the combination of a jump evolution and linear growth.

Here, the choice of the kernel rescaling is not obvious, since it is a mix of the two precedent cases: peaks and jumps. We show however the performance the rescaling, and show all the estimations (α, β, ν) in Figure 5.

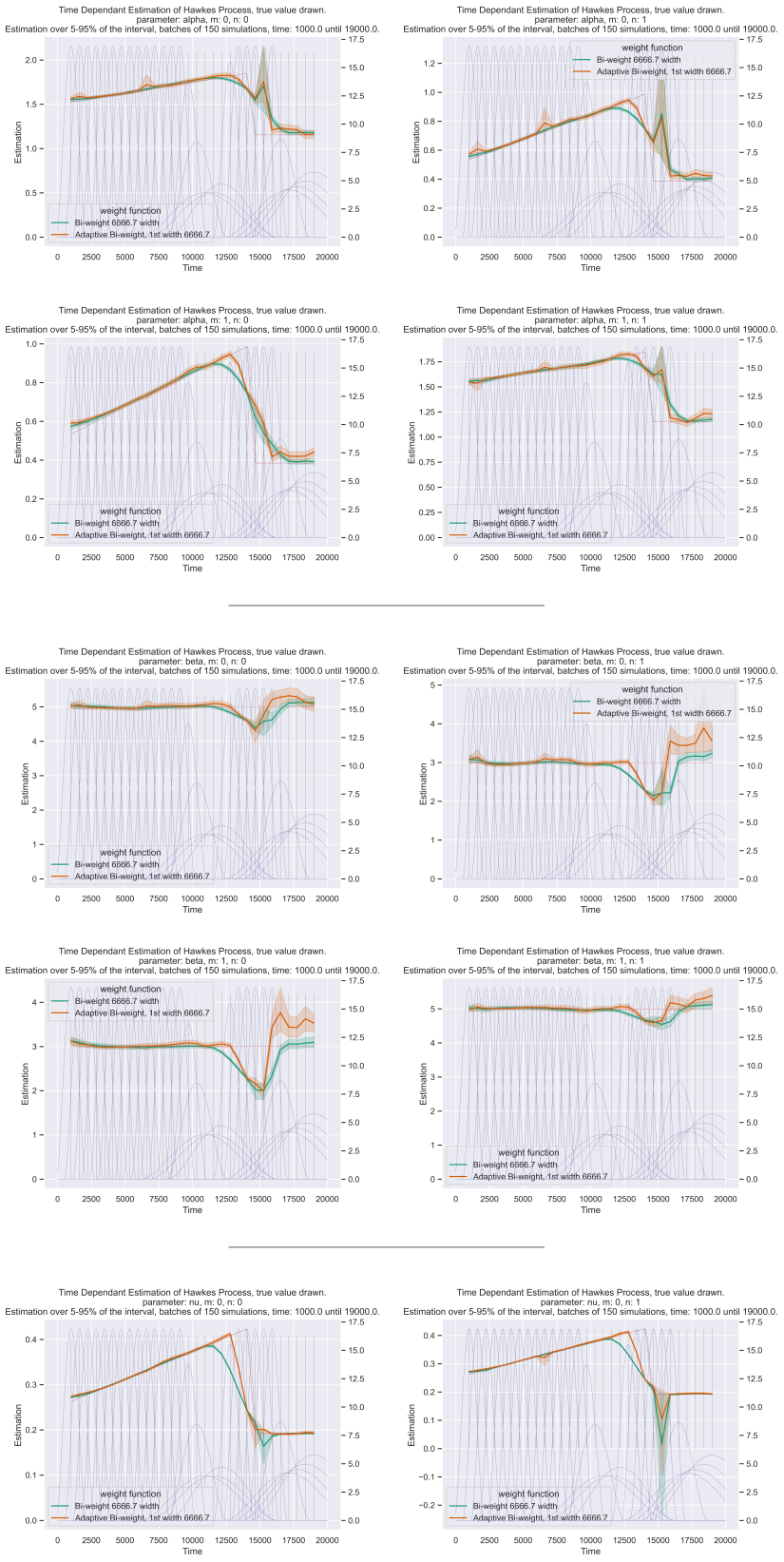

FIGURE 5. Comparison results of estimation before and after rescaling the bandwidths with ITiDeEP, for bi-variate Hawkes. The underlying parameters follow a mountain growth.

Parameter	Value
Rescale Function	Sinus
Number of Simulations and Estimation	100
Initial Denominator of T max used as bandwidth	2
L	0.04
R	0.90
h	2.50
l	1.00

TABLE 2. Set of parameters in the error computation.

5.4. **MISPE variation.** We show that numerically for one example, the ITiDeEP is a way to trade variance for bias, without harming the performance of the estimation and in the best case, even improving the L^2 error. To do that, we compare the error of a naive estimation and the error after applying a rescaling of the bandwidth.

The error we consider is as previously, the mean integrated squared error: $\mathbb{E}\left[\int_0^T \|\theta - \hat{\theta}\|_2^2 \, dt\right]$, and we approximate it and rescale it to make it more meaningful.

(1) First, we rescale the norm of the difference, to take into account the different relative errors.
(2) Second, we rescale with respect to the length of the integral. This is because we would like to take longer time series as observation and see how the error behaves.

All this together, we get the **Mean (time rescaled) Integrated Squared Percentage Error (MISPE)**:

$$\mathbb{E}\left[\frac{1}{T}\int_0^T \|\frac{\theta(t) - \hat{\theta}(t)}{\theta(t)}\|_2^2 \, dt\right], \tag{5.1}$$

where the division of vectors should be interpreted as division term by term. It is also worth considering the MIAPE where the L^2 norm is replaced by an L^1 norm in the expression above.

This expression is approximated by:

$$\frac{1}{N_{\text{simul}}}\frac{1}{N_{\text{times}}}\sum_{t_j \in \text{estim.'s time}} \|\frac{\theta(t_j) - \hat{\theta}(t_j)}{\theta(t_j)}\|_2^2, \tag{5.2}$$

where N_{simul} is the number of repeated estimations for different data, N_{times} the number of different time estimations for the parameters over the segment $[0, T]$ represented here by the set "estim.'s time" and finally M is the number of dimensions of the process.

We repeat the computation of the MISPE for different values of T. We also show the error in the case where we apply the ITiDeEP rescaling on the original bandwidth but with the second estimation's values. The second estimation has less bias and globally less mean square error (thanks to a first application of the ITiDeEP).

The parameters used are gathered in Table 2. The plots are on Figure 6. We plotted the error (in L^1 and L^2 sense) regarding first the parameters alpha and nu separately, and then the compounded error.

Remark 5.1. We could iteratively continue this procedure and rescale the original kernels' width with this more precise (second) estimation. This should yield a better result. In other words, it is possible to use the rescaling algorithm n times by using the $n-1$ estimate as a proxy of the true value to rescale the kernels accordingly. The rescale functions have to be changed at each iteration.

6. CONCLUSION

We have developed an algorithm in order to perform time-dependent estimation of parameters with MLE. This is valuable because it allows the user to extract the most out of the available information. Nevertheless,

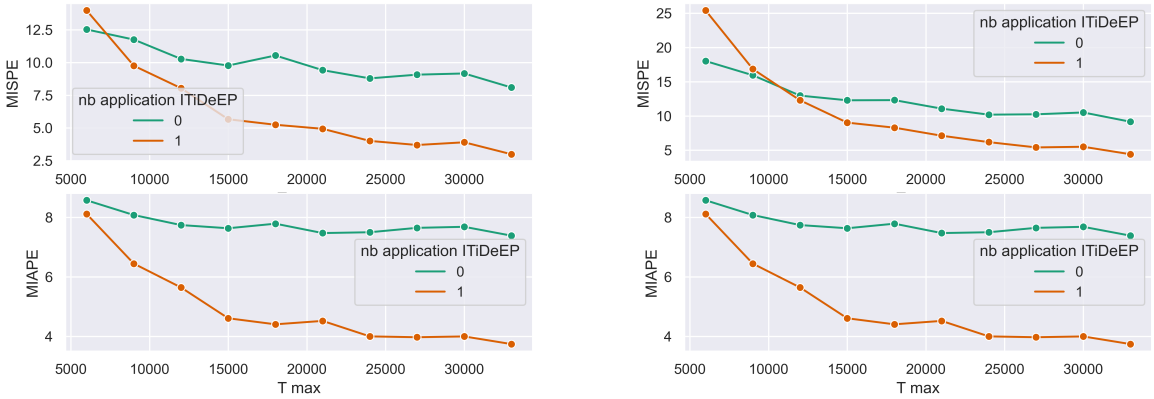

FIGURE 6. MIS/APError. We notice for both a convergence asymptotically to a better result. The only difference between the left and right panels is that on the left, we do not include the error for β as it should have increased because it is a constant parameter over the whole-time estimation period, hence the best window would be the widest one. We precise that the number of events in respective datasets are: $[3458, 5218, 6962, 10445, 12179, 13916, 15672, 17475, 19163]$ (approximately 0.6 events per unit of time). We observe that for both error types, adjusting the window size with HITiDeEP has successfully reduced the error. In particular, we note that asymptotically, the error is only due to the choice of the window, which induces a bias. A smaller window reduces the bias: it is the reason why we see an improvement asymptotically in the estimation.

it can be hard to find the right bandwidth, and in order to account for this, we propose the ItiDeEP algorithm. The algorithm improves performance and reduces the estimation bias when the windows are rescaled appropriately. The reduction of the bias yields a better asymptotic estimation.

APPENDIX A. SIMULATION OF HAWKES PROCESSES

A.1. **Notation.** The author refers to paper [39] for the chosen best solution to simulate Hawkes processes. There are different ways to simulate a Hawkes process, but they do not give as much expressivity as the one from [39], which is why we use it.

With the notation of the article, the inter-arrival time of the self-exciting part reads:

$$\forall\, m, j \in [\![1, M]\!], \qquad a^i_{m,j} \sim \mathcal{L}^i_{m,j},$$

where $\mathcal{L}^i_{m,j}$ is defined by the CDF:

$$\begin{aligned} F(s) &= 1 - \exp\left(-\int_{r_{j-1}}^{r_{j-1}+s} \lambda^i_m(t)dt\right) \\ &= 1 - \exp\left(-\frac{\lambda^i_m(r_{j-1})}{\beta^i_m}(1 - e^{-\beta^i_m s})\right). \end{aligned} \tag{A.1}$$

Hence we can use the inverse CDF method to generate the inter-arrival of the self-exciting part. The process is described in algorithm 3.

Observe a sample path for a penta-variate Hawkes process in Figure 7.

In the version we read, there was a typo in the inverse CDF method[10]. The equation reads, keeping the paper's notation:

[10]One can compare the line (A.2) with the equation (20) from [39].

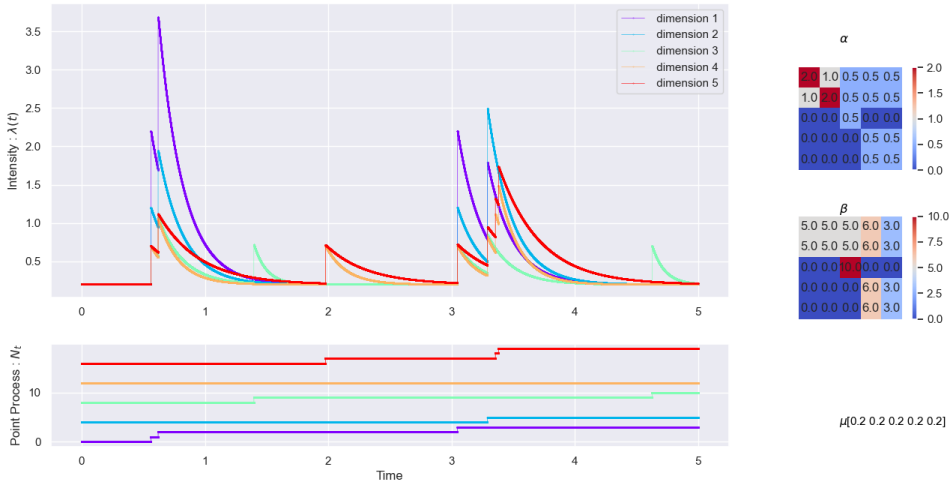

FIGURE 7. Sampled path from a penta-variate Hawkes process. Top: intensity of each process. Bottom: corresponding counting process. The processes N_t^m are shifted for readability.

Algorithm 3: Generate the waiting time of the self-exciting part.

$u \sim U(0,1)$,
if $u < 1 - \exp\left(-\frac{\lambda_m^i(r_{j-1})}{\beta_m^i}\right)$ **then**
  we let

$$a_{m,j}^i = -\frac{1}{\beta_m^i}\log\left(1 + \frac{\beta_m^i}{\lambda_m^i(r_{j+1})}\log(1-u)\right) \tag{A.2}$$

else
  $a_{m,j}^i = \infty$.
end

A.2. **Simulation for Time-Dependent Parameters.** The algorithm from [39] allows simulating multi-dimensional Hawkes processes, with different decay parameters, for an exponential kernel with marks, and constant immigration rates. From that, it is obvious that one can use the previous algorithm with deterministic jumps (α). On the other hand, the underlying intensity (ν) is easily changeable by using any method for simulating a non-homogeneous Poisson process.

However, the task is way harder concerning the decays (β).

To sum-up:

- deterministic $\alpha(\cdot)$ can be simulated thanks to the previous algorithm;
- deterministic $\beta(\cdot)$; The task with beta is quite harder. The problem lies in evaluating the integral; it is easy when β is not a function of the parameter with respect to which we integrate. Nevertheless, it is possible without too much effort to change the β deterministically at the random jumps time. Then, the parameters would be piecewise constant;
- deterministic $\nu(\cdot)$ requires a few changes in the algorithm. To make ν time-dependent, we are using a basic and classical thinning algorithm, which is the accepted method for deterministic inhomogeneous Poisson processes. One can find more information about such methods in Appendix part B.1.

Remark A.1. Thinning algorithms have the drawback of being a non-exact method. It would also be possible to compute the CDF of the inter-arrival times related to a function $\nu(\cdot)$ and inverse the formula, in the same fashion as we did in the first simulation algorithm to achieve a perfect method. We give an example in the following of the inversion of a model. However, such an approach would require solving by hand the inverse of some formulas, which does not always exist in closed form and require human intervention. Therefore, we decided to keep the non-exact but flexible thinning algorithm.

In order to simulate the underlying non-homogeneous Poisson process, we define the following law:

$$\forall\, m, j \in [\![1, M]\!], \qquad a^0_{m,j} \sim \mathcal{J}_m,$$

where $\mathcal{J}_m$ is defined by the CDF:

$$F(s) = 1 - \exp\left(- \int_{r_{j-1}}^{r_{j-1}+s} \nu_m(t)\, dt \right). \tag{A.3}$$

We also need the following equation, which deals with the deterministic α[11]:

$$\lambda^i_m(r_j) = \lambda^i_m(r_{j-1}) e^{-\beta^i_m \min_{m,i} a^i_{m,j}} + \alpha_{i,m}(r_j)\, \mathbb{1}_{i=m^*}\,. \tag{A.4}$$

Algorithm 4: Exact simulation of multidimensional Hawkes process. From [39].

for $j > 0$ **do**
  while $r_j < T$ **do**
    for $m \in [\![1, M]\!]$ **do**
      Sample $a^0_{m,j} \sim \mathcal{J}_m$ as defined by (A.3) using a thinning algorithm (see Subsection B.1 in appendix),
      for $i \in [\![1, M]\!]$ **do**
        Sample $a^i_{m,j} \sim \mathcal{L}^i_{m,j}$, as defined by (A.1),
      end
    end
    $r_j = r_{j-1} + \min_{m,i} a^i_{m,j}$,
    for $m \in [\![1, M]\!]$ **do**
      Update $\lambda^i_m(r_j)$ according to equation (A.4),
      Update $N^m(r_j) = N^m(r_{j-1}) + \mathbb{1}_{m=m^*}$,
    end
    $t^{m^*}_k = r_j$ where $k = N^{m^*}(r_j)$ and $m^*, i^* = \operatorname{argmin}_{i,m} a^i_{m,j}$,
  end
  Discard the last $r_j >$ T.
end

One can see a few examples of realised univariate processes where the parameters change over time in Figure 8. Looking at the pictures, the impact of time upon the parameters is crystal clear.

Appendix B. Methods for Simulating a Hawkes Process

We describe a few algorithms for simulating Hawkes processes. We will give particular attention to the following criteria:

- Exact, without having to use approximation. One also defines "exact" as a method of drawing an unbiased associated estimator throughout the entire simulation process,
- Efficient (no waste, not using the rejection sampling)[12],
- Different decays (all parameters in Θ as defined previously: $\Theta = (\nu, \alpha, \beta)$ can be used)
- Followable (source triggering the event is known), we define it in this way,
- Perfect (take past history into account, i.e. no "edge effect"),
- Markable (jumps can be marked; marked Hawkes process),
- Multivariate (the algorithm supports multi-variate Hawkes process),

The criteria "different decays" and "followable" are only interesting for multivariate algorithms.

Because a Hawkes process can be seen either as a non-homogeneous Poisson process or a branching process, two essential classes of simulating methods exist: intensity-based and cluster-based simulation. We will talk about the ones that have been applied and consensually accepted in the literature. They gather into

[11]The only difference is that now we consider α as a function of the time.
[12]An algorithm that uses the rejection sampling method is called non-efficient.

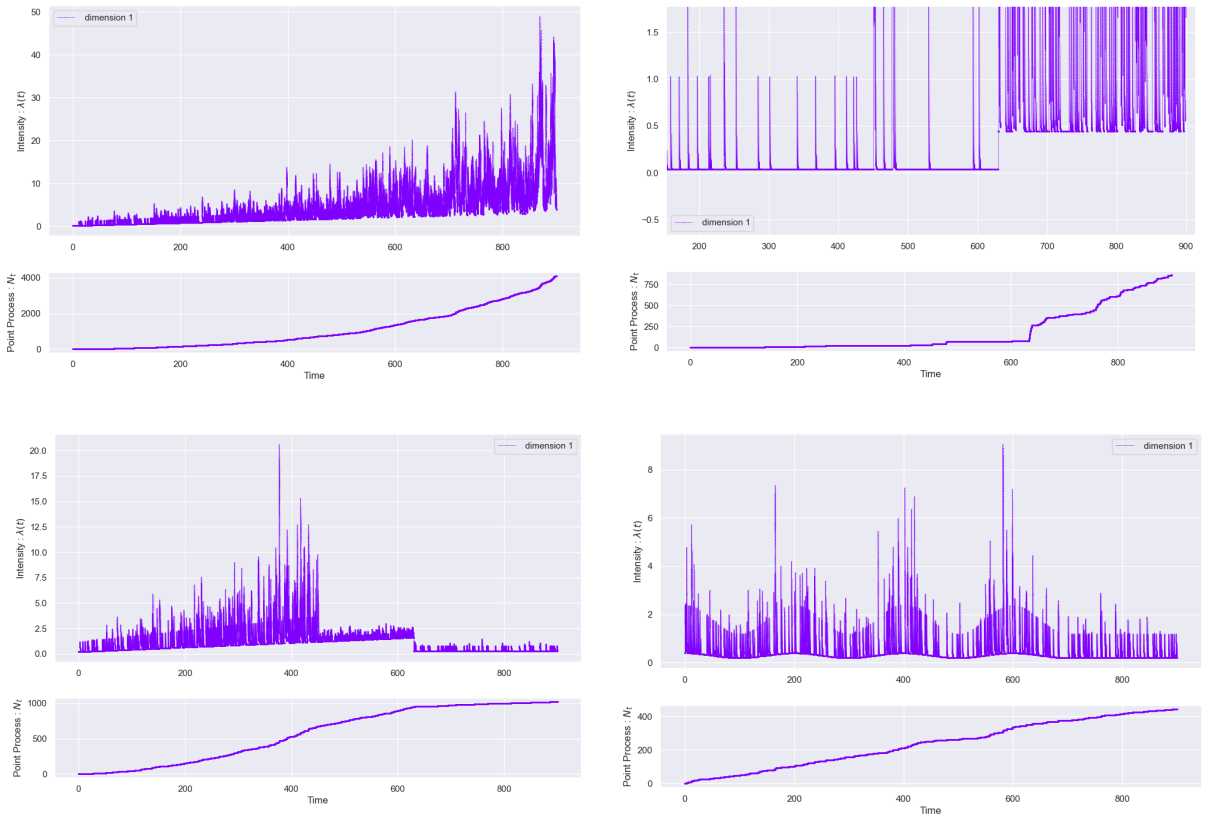

FIGURE 8. Plot of four sampled paths of a univariate Hawkes process with time-dependent parameters. We modify α and ν with respect to time. One can notice the jumps, as well as the underlying intensity, changing with respect to time. From left to right, top to bottom: linear increase, jump, linear and jump, and sinusoidal change. For the one corresponding to the "one jump" function, we zoomed in to emphasise the phenomena.

three categories: inversion, order-statistics methods and rejection sampling. We refer to [53] for a detailed description of the categories (a dozen algorithms are also proposed).

B.1. **Intensity Based Simulation.** Hawkes process generation can be viewed as the simulation of an inhomogeneous Poisson process. Indeed, the intensity is globally random, with deterministic segments in between jumps ($\forall\, k \in \mathbb{N}_{i>0},]t_k, t_{k+1}]$ is a deterministic segment). For that reason, one can use the classic thinning algorithm (also referred to as the Lewis-Shedler thinning algorithm), whose pseudo-code can be found in [38] and whose simplest version is well described in [56] (1.18).

Lewis' algorithm works well enough in the settings of fixed intensity functions. Its main drawback is the need for a bound on the intensity that holds over an interval for all histories of the process. This is a tremendous flaw for applications to processes with random conditional intensities (as Hawkes processes). This difficulty has been overcome with Ogata, whose algorithm requires only a local boundedness condition on the conditional intensity. One can read about it in [49] (the so-called Ogata's modified thinning algorithm).

A clear sum-up is available at the end of [36]. One can also find a very clear and detailed review of intensity-based algorithms in [19], Section 7.5. In particular, they mention in Algorithm 7.5.II. the Lewis algorithm; algorithm 7.5.IV. the Ogata's polishing and at 7.5.V. the thinning algorithm for marked point processes.

Thinning methods fall into the category of Acceptance-Rejection methods and are not described as efficient for the reason that for n events, depending on the input, the complexity can rise to $\mathcal{O}(n^2)$.

B.2. **Superposition Based Simulation.** Another interesting method for simulating Poisson processes is based upon the immigration-birth representation. The idea is first to simulate the immigrants as a Poisson process. Conditional on the number of immigrants, the time at which the immigrants appear is an order statistics, identically and independently uniformly distributed over the segment (proof in [53] as Theorem 2). The descendants then form an inhomogeneous Poisson process, which can be simulated thanks to another trick. Since we know that the branching representation of the Hawkes process has α/β births on average, we simulate the number of descendants with a Poisson law with parameter α/β and then add the new jumps by computing the inter-arrival time, immigrant-son, of a homogeneous Poisson process with parameter β.

The algorithm can be found in [36]. We would like, however, to point out that the algorithm is incomplete. The algorithm generates the immigrants as well as the first generation of the offspring and then stops. A for loop should be added on the next generations until no new offspring is generated.

B.3. **Other Used Methods in the Literrature.** We highlight the algorithm proposed by Møller and Rasmussen [45], which is a perfect method in the sense of our definition at the beginning of the section.

An exciting evolution of thinning has been given in [20], where the authors use superposition and thinning to reduce the algorithm's complexity. The methods are vulgarised in [56] (1.19). The authors used the Markov property of the Hawkes process (granted by the exponential kernel). They also include in the algorithm the possibility of having a different starting intensity than the asymptotic underlying intensity.

The biggest flaw of that algorithm is that all the decays need to be identical in the multivariate case. There is, however, another algorithm that ggeneralises[20]'s one (in the sense they are also using thinning and superposition) but they did manage to get the desired flexibility. The algorithm is very well stated inside [39]. Also, usually, only a cluster-based algorithm conveys information regarding the source that triggers the event times. For that reason, [39] is an exciting algorithm from an innovative perspective.

All the previously mentioned algorithms are summed-up in Table 9.

B.4. **Github Repository State of the Art Thinning Algorithm in Python.** The paper [7] offers a state-of-the-art implementation in Python of thinning algorithms for Hawkes processes and other temporal point processes through the associated repository. A broad range of implemented algorithms is proposed, which despite not being optimised for the exponential kernel, is very useful in common situations.

Name of the Method	Type	Author	Exact	Efficient	Perfect	Markable	Multivariate	Different Decays	Followable
Ozaki's Algorithm	Intensity	[52]	×	✓	×	×	×	–	–
Thinning Algorithm	Intensity	[38], [16]	✓	×	×	×	×	–	–
Ogata's Modified Thinning Algorithm	Intensity	[49], [16], [15]	✓	×	×	×	✓	✓	×
Algorithm 7.5. Daley;	Intensity	[19]	✓	×	×	✓	×	–	–
Perfect simulation	Cluster	[45], improved version: [13]	✓	✓	✓	✓	×	–	–
Exact simulation	Mixed	[20]	✓	✓	×	✓	✓	×	×
Laub's Cluster algorithm	Cluster	[36]	✓	✓	×	×	×	–	–
Superposed exact simulation	Mixed	[39]	✓	✓	×	✓	✓	✓	✓

FIGURE 9. Overview of a few different methods for Hawkes process simulation.

Appendix C. Likelihood Function Expression

C.1. **Derivation of the (weighted) Likelihood Function.** Even though the likelihood can already be explicitly found in the related literature (see [14, 17]), formulas are usually for the univariate case and its derivatives are never explicitly written. We dedicate ourselves to writing the explicit formulas hereinafter. For this reason, we derive the gradient and the Hessian of the log-likelihood in the following.

C.2. **Gradient and Hessian's Values.** We fix m, n, m', n' as being inside $[\![1, M]\!]^2, M \in \mathbb{N}_{>0}$. We drop the subscript on the W and on the weight function w for clarity.

Likelihood Function:

We introduce the function R inside Theorem 3.1 for computational reasons. The log-likelihood we described in closed form earlier with R reads:

$$\begin{aligned}\log WL^m(\theta \mid \mathcal{T}) = -\nu_m T - \sum_{n=1}^{M} \frac{\alpha_{m,n}}{\beta_{m,n}} \sum_{\{k:t_k^n<T\}} w(t_k^n)\,\left(1-\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\right) \\ + \sum_{\{k:t_k^m<T\}} w(t_k^m)\,\log\left(\nu_m + \sum_{j=1}^{M}\alpha_{m,j}R_{m,j}(k)\right),\end{aligned} \tag{C.1}$$

where $R_{m,n}$ is defined as $R_{m,n}(1) = 0$ and for $k \geq 1$ as

$$\forall\, k \geq 2, \qquad R_{m,n}(k) = \sum_{\{i:t_i^n<t_k^m\}} \exp\left(-\beta_{m,n}\cdot(t_k^m - t_i^n)\right).$$

We will prefer to this expression the recursion which leads to faster computations:

$$\begin{aligned} k=1, \quad & R_{m,n}(k) = 0, \\ k\geq 2, \quad & R_{m,n}(k) = \exp(-\beta_{m,n}\cdot(t_k^m - t_{k-1}^m))R_{m,n}(k-1) \\ & \qquad + \sum_{\{i:t_i^n\in[\ t_{k-1}^m, t_k^m\ [\ \}} \exp(-\beta_{m,n}(t_k^m - t_i^n)). \end{aligned}$$

While the first approach is computed in $\mathcal{O}(n^2)$, the latter costs $\mathcal{O}(n)$. The origin and computations of the recursion can be found in [18, 49].

Remark C.1. Notice that it is possible to use this less expensive expression in the case $m = n$:

$$k \geq 2, \quad R_{m,n}(k) = \exp(-\beta_{m,n}\cdot(t_k^m - t_{k-1}^m))(1 + R_{m,n}(k-1)).$$

Furthermore, we will also need the derivatives of those expressions for which no useful recurrent expression has been found yet.

$$\begin{aligned} k=1, \quad & R'_{m,n}(k) = 0, \\ k\geq 2, \quad & R'_{m,n}(k) = \sum_{\{i:t_i^n<t_k^m\}} (t_k^m - t_i^n)\exp\left(-\beta_{m,n}\cdot(t_k^m - t_i^n)\right), \end{aligned}$$

$$\begin{aligned} k=1, \quad & R''_{m,n}(k) = 0, \\ k\geq 2, \quad & R''_{m,n}(k) = \sum_{i:t_i^n<t_k^m} (t_k^m - t_i^n)^2\exp(-\beta_{m,n}\cdot(t_k^m - t_i^n)). \end{aligned}$$

First order derivatives:

$$\frac{\partial}{\partial \nu_m}\log WL^m(\theta \mid \mathcal{T}) = -T + \sum_{\{k:t_k^m<T\}} \frac{w(t_k^m)}{\nu_m + \sum_{j=1}^{M}\alpha_{m,j}R_{m,j}(k)}, \tag{C.2}$$

$$\frac{\partial}{\partial \alpha_{m,n}} \log WL^m(\theta \mid \mathcal{T}) = -\frac{1}{\beta_{m,n}} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ \left(1-\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\right)$$
$$+ \sum_{\{k:t_k^m<T\}} w(t_k^m)\ \frac{R_{m,n}(k)}{\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)}, \tag{C.3}$$

$$\frac{\partial}{\partial \beta_{m,n}} \log WL^m(\theta \mid \mathcal{T}) = \frac{\alpha_{m,n}}{\beta_{m,n}^2} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ \left(1-\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\right)$$
$$- \frac{\alpha_{m,n}}{\beta_{m,n}} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ (T-t_k^n)\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right))$$
$$- \sum_{\{k:t_k^m<T\}} w(t_k^m)\ \frac{\alpha_{m,n} R'_{m,n}(k)}{\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)}. \tag{C.4}$$

Second order derivatives:

$$\frac{\partial^2}{\partial \nu_m^2} \log WL^m(\theta \mid \mathcal{T}) = -\sum_{\{k:t_k^m<T\}} \frac{w(t_k^m)}{\left(\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)\right)^2}, \tag{C.5}$$

$$\frac{\partial^2}{\partial \nu_m \partial \nu_{m'}} \log WL^m(\theta \mid \mathcal{T}) = 0, \qquad m \neq m', \tag{C.6}$$

$$\frac{\partial^2}{\partial \alpha_{m,n}^2} \log WL^m(\theta \mid \mathcal{T}) = -\sum_{\{k:t_k^m<T\}} w(t_k^m)\ \left[\frac{R_{m,n}(k)}{\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)}\right]^2, \tag{C.7}$$

$$\frac{\partial^2}{\partial \alpha_{m,n} \partial \alpha_{m,n'}} \log WL^m(\theta \mid \mathcal{T}) = -\sum_{\{k:t_k^m<T\}} w(t_k^m)\ \frac{R_{m,n}(k) R_{m,n'}(k)}{\left(\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)\right)^2}, \qquad n \neq n', \tag{C.8}$$

$$\frac{\partial^2}{\partial \alpha_{m,n} \partial \alpha_{m',n'}} \log WL^m(\theta \mid \mathcal{T}) = 0, \qquad m \neq m'; n, n' \in [\![1, M]\!], \tag{C.9}$$

$$\frac{\partial^2}{\partial \nu_m \partial \alpha_{m,n}} \log WL^m(\theta \mid \mathcal{T}) = -\sum_{\{k:t_k^m<T\}} w(t_k^m)\ \frac{R_{m,n}(k)}{\left(\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)\right)^2}, \tag{C.10}$$

$$\frac{\partial^2}{\partial \nu_m \partial \alpha_{m',n}} \log WL^m(\theta \mid \mathcal{T}) = 0, \qquad m \neq m', n \in [\![1, M]\!], \tag{C.11}$$

$$\frac{\partial^2}{\partial \beta_{m,n}^2} \log WL^m(\theta \mid \mathcal{T}) = -2\frac{\alpha_{m,n}}{\beta_{m,n}^3} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ \left(1-\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\right)$$
$$+2\frac{\alpha_{m,n}}{\beta_{m,n}^2} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ (T-t_k^n)\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right))$$
$$+\frac{\alpha_{m,n}}{\beta_{m,n}} \sum_{\{k:t_k^n<T\}} w(t_k^n)\ (T-t_k^n)^2\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right))$$
$$+ \sum_{\{k:t_k^m<T\}} w(t_k^m)\ \left(\frac{\alpha_{m,n} R''_{m,n}(k)}{\left(\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)\right)^2} - \left(\frac{\alpha_{m,n} R'_{m,n}(k)}{\nu_m + \sum_{j=1}^M \alpha_{m,j} R_{m,j}(k)}\right)^2\right), \tag{C.12}$$

$$\frac{\partial^2}{\partial\beta_{m,n}\partial\beta_{m,n'}}\log WL^m(\theta\mid\mathcal{T}) = -\sum_{\{k:t_k^m<T\}} w(t_k^m)\,\frac{\alpha_{m,n}R'_{m,n}(k)\alpha_{m,n'}R'_{m,n'}(k)}{\left(\nu_m+\sum_{j=1}^M\alpha_{m,j}R_{m,j}(k)\right)^2}, \qquad n\neq n', \tag{C.13}$$

$$\frac{\partial^2}{\partial\beta_{m,n}\partial\beta_{m',n'}}\log WL^m(\theta\mid\mathcal{T}) = 0, \qquad m\neq m'; n,n'\in[\![1,M]\!], \tag{C.14}$$

$$\frac{\partial^2}{\partial\nu_m\partial\beta_{m,n}}\log WL^m(\theta\mid\mathcal{T}) = \sum_{\{k:t_k^m<T\}} w(t_k^m)\,\frac{\alpha_{m,n}R'_{m,n}(k)}{\left(\nu_m+\sum_{j=1}^M\alpha_{m,j}R_{m,j}(k)\right)^2}, \tag{C.15}$$

$$\frac{\partial^2}{\partial\nu_m\partial\beta_{m',n}}\log WL^m(\theta\mid\mathcal{T}) = 0, \qquad m\neq m'; n\in[\![1,M]\!], \tag{C.16}$$

$$\begin{aligned}\frac{\partial^2}{\partial\beta_{m,n}\partial\alpha_{m,n}}\log WL^m(\theta\mid\mathcal{T}) =& \frac{1}{\beta_{m,n}^2}\sum_{\{k:t_k^n<T\}} w(t_k^n)\,\left(1-\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\right)\\ &-\frac{1}{\beta_{m,n}}\sum_{\{k:t_k^n<T\}} w(t_k^n)\,(T-t_k^n)\exp\left(-\beta_{m,n}\cdot(T-t_k^n)\right)\\ &-\sum_{\{k:t_k^m<T\}} w(t_k^m)\,\frac{R'_{m,n}(k)}{\nu_m+\sum_{j=1}^M\alpha_{m,j}R_{m,j}(k)}\\ &+\sum_{\{k:t_k^m<T\}} w(t_k^m)\,\frac{\alpha_{m,n}R'_{m,n}(k)R_{m,n}(k)}{\left(\nu_m+\sum_{j=1}^M\alpha_{m,j}R_{m,j}(k)\right)^2},\end{aligned} \tag{C.17}$$

$$\frac{\partial^2}{\partial\beta_{m,n}\partial\alpha_{m,n'}}\log WL^m(\theta\mid\mathcal{T}) = \sum_{\{k:t_k^m<T\}} w(t_k^m)\,\frac{\alpha_{m,n}R'_{m,n}(k)R_{m,n'}(k)}{\left(\nu_m+\sum_{j=1}^M\alpha_{m,j}R_{m,j}(k)\right)^2}, \qquad n\neq n', \tag{C.18}$$

$$\frac{\partial^2}{\partial\alpha_{m,n}\partial\beta_{m',n'}}\log WL^m(\theta\mid\mathcal{T}) = 0, \qquad m\neq m'; n,n'\in[\![1,M]\!]. \tag{C.19}$$

C.3. **Gradient and Hessian's Structure.** The derivatives are obviously at least $\mathcal{C}^2$. Thus, by using the Schwartz Theorem, the gradient exists, so does the Hessian and the latter is symmetric. This allows for reducing the number of required computations by half.
Hereinafter we draw a proposition for the structure of the gradient and Hessian, coined respectively by D and H, for an M-variate Hawkes process:

$$D=\left(\frac{\partial}{\partial\nu_1},\cdots,\frac{\partial}{\partial\nu_M},\frac{\partial}{\partial\alpha_{1,1}},\cdots,\frac{\partial}{\partial\alpha_{1,M}},\frac{\partial}{\partial\alpha_{2,1}},\cdots,\frac{\partial}{\partial\alpha_{M,M}},\frac{\partial}{\partial\beta_{1,1}},\cdots,\frac{\partial}{\partial\beta_{1,M}},\frac{\partial}{\partial\beta_{2,1}},\cdots,\frac{\partial}{\partial\beta_{M,M}}\right).$$

By separating the Hessian into nine pieces:

$$H=\begin{pmatrix}H_{1,1}&H_{1,2}&H_{1,3}\\H_{2,1}&H_{2,2}&H_{2,3}\\H_{3,1}&H_{3,2}&H_{3,3}\end{pmatrix},$$

where each piece reads, and we highlighted the symmetrical part:

$$H_{1,1}=\begin{pmatrix}\frac{\partial^2}{\partial\nu_1\partial\nu_1}&0&\cdots&0&0\\0&\frac{\partial^2}{\partial\nu_2\partial\nu_2}&\cdots&0&0\\0&0&\frac{\partial^2}{\partial\nu_3\partial\nu_3}&\cdots&\vdots\\\vdots&\vdots&\vdots&\ddots&0\\0&0&\cdots&0&\frac{\partial^2}{\partial\nu_M\partial\nu_M}\end{pmatrix}$$

$$H_{2,2} =$$

$$\begin{pmatrix}
\frac{\partial^2}{\partial\alpha_{1,1}\partial\alpha_{1,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{1,1}\partial\alpha_{1,M}} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
\frac{\partial^2}{\partial\alpha_{1,M}\partial\alpha_{1,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{1,M}\partial\alpha_{1,M}} & 0 & \cdots & 0 & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\alpha_{2,1}\partial\alpha_{2,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{2,1}\partial\alpha_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \vdots & \ddots & \vdots & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\alpha_{2,M}\partial\alpha_{2,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{2,M}\partial\alpha_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & \frac{\partial^2}{\partial\alpha_{M,M-1}\partial\alpha_{M,M}} & \cdots & \frac{\partial^2}{\partial\alpha_{M,M}\partial\alpha_{M,M}}
\end{pmatrix}$$

$$H_{3,3} =$$

$$\begin{pmatrix}
\frac{\partial^2}{\partial\beta_{1,1}\partial\beta_{1,1}} & \cdots & \frac{\partial^2}{\partial\beta_{1,1}\partial\beta_{1,M}} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
\frac{\partial^2}{\partial\beta_{1,M}\partial\beta_{1,1}} & \cdots & \frac{\partial^2}{\partial\beta_{1,M}\partial\beta_{1,M}} & 0 & \cdots & 0 & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\beta_{2,1}\partial\beta_{2,1}} & \cdots & \frac{\partial^2}{\partial\beta_{2,1}\partial\beta_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \vdots & \ddots & \vdots & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\beta_{2,M}\partial\beta_{2,1}} & \cdots & \frac{\partial^2}{\partial\beta_{2,M}\partial\beta_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & \frac{\partial^2}{\partial\beta_{M,M-1}\partial\beta_{M,M}} & \cdots & \frac{\partial^2}{\partial\beta_{M,M}\partial\beta_{M,M}}
\end{pmatrix}$$

$$H_{2,3} =$$

$$\begin{pmatrix}
\frac{\partial^2}{\partial\alpha_{1,1}\partial\beta_{1,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{1,1}\partial\beta_{1,M}} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
\frac{\partial^2}{\partial\alpha_{1,M}\partial\beta_{1,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{1,M}\partial\beta_{1,M}} & 0 & \cdots & 0 & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\alpha_{2,1}\partial\beta_{2,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{2,1}\partial\beta_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \vdots & \ddots & \vdots & \cdots & \vdots & \cdots & \vdots \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\alpha_{2,M}\partial\beta_{2,1}} & \cdots & \frac{\partial^2}{\partial\alpha_{2,M}\partial\beta_{2,M}} & \cdots & \vdots & \cdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & \frac{\partial^2}{\partial\alpha_{M,M-1}\partial\beta_{M,M}} & \cdots & \frac{\partial^2}{\partial\alpha_{M,M}\partial\beta_{M,M}}
\end{pmatrix}$$

$$H_{1,2} =$$

$$\begin{pmatrix}
\frac{\partial^2}{\partial\nu_1\partial\alpha_{1,1}} & \cdots & \frac{\partial^2}{\partial\nu_1\partial\alpha_{1,M}} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\
0 & \cdots & 0 & \frac{\partial^2}{\partial\nu_2\partial\alpha_{2,1}} & \cdots & \frac{\partial^2}{\partial\nu_2\partial\alpha_{2,M}} & \cdots & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & \frac{\partial^2}{\partial\nu_M\partial\alpha_{M,1}} & \cdots & \frac{\partial^2}{\partial\nu_M\partial\alpha_{M,M}}
\end{pmatrix}$$

$$H_{1,3} =$$

$$\begin{pmatrix} \frac{\partial^2}{\partial\nu_1\partial\beta_{1,1}} & \cdots & \frac{\partial^2}{\partial\nu_1\partial\beta_{1,M}} & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \frac{\partial^2}{\partial\nu_2\partial\beta_{2,1}} & \cdots & \frac{\partial^2}{\partial\nu_2\partial\beta_{2,M}} & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & \frac{\partial^2}{\partial\nu_M\partial\beta_{M,1}} & \cdots & \frac{\partial^2}{\partial\nu_M\partial\beta_{M,M}} \end{pmatrix}$$

C.4. **Implementation and Optimization.** With the expression of the likelihood for a Hawkes process, it is now possible to find the MLE. As mentioned, we use a classic multivariate Newton-Raphson algorithm. However, this optimisation problem is non-convex, and the above procedure might converge to a local maximum instead [24].

First, there is a computational bottleneck due to the computation of the expressions of R, R', R'' and that appears in the computations of the log-likelihood. The usage of the profiler shows that most of the computational time comes from this bottleneck.

Hence, one should rely on the expressions R, R', R'' to make the code clearer and faster. R has a recurrent expression; an idea that we did not find in the literature is the possibility of using memoization on most parts of the computation of the derivatives to make the code run faster. Nonetheless, this part of the code is a computational bottleneck of complexity $\mathcal{O}(n^2)$.

References

[1] F. Abergel and A. Jedidi. "Long-time behavior of a hawkes process–based limit order book". In: *SIAM Journal on Financial Mathematics* 6.1 (2015), pp. 1026–1043.

[2] I. S. Abramson. "On bandwidth variation in kernel estimates - A square root law". In: *Annals Statist.* 10 (1982).

[3] E. S. Ahmed, A. I. Volodin, and A. A. Hussein. "Robust weighted likelihood estimation of exponential parameters". In: *IEEE Transactions on reliability* 54.3 (July 2005), pp. 389–395.

[4] L. B. Allal, A. Lejay, and R. Stoica. "Hawkes point processes based inference applied to seismic data analysis". In: *2020 RING MEETING*. 2020.

[5] E. Anderes and M. Stein. "Local likelihood estimation for nonstationary random fields". In: *Journal of Multivariate Analysis* 102.3 (2011), pp. 506–520. DOI: 10.48550/ARXIV.0911.0047. URL: https://arxiv.org/abs/0911.0047.

[6] E. Bacry, I. Mastromatteo, and J.-F. Muzy. "Hawkes processes in finance". In: *Market Microstructure and Liquidity* 1.01 (2015), p. 1550005. arXiv: 1502.04592 [q-fin.TR].

[7] E. Bacry et al. *Tick: a Python library for statistical learning, with a particular emphasis on time-dependent modelling*. 2018. arXiv: 1707.03003 [stat.ML].

[8] A. Baddeley. "Local composite likelihood for spatial point processes". In: *Spatial Statistics* 22 (2017). Spatio-temporal Statistical Methods in Environmental and Biometrical Problems, pp. 261–295. ISSN: 2211-6753. DOI: https://doi.org/10.1016/j.spasta.2017.03.001. URL: https://www.sciencedirect.com/science/article/pii/S2211675316301099.

[9] P. Brémaud and L. Massoulié. "Stability of Nonlinear Hawkes Processes". In: *The Annals of Probability* 24.3 (1996), pp. 1563–1588. ISSN: 00911798. URL: http://www.jstor.org/stable/2244985.

[10] J. Cao, D. Landriault, and B. Li. "Optimal reinsurance-investment strategy for a dynamic contagion claim model". In: *Insurance: Mathematics and Economics* 93.C (2020), pp. 206–215. DOI: 10.1016/j.insmatheco.2020. URL: https://ideas.repec.org/a/eee/insuma/v93y2020icp206-215.html.

[11] A. Cartea, S. N. Cohen, and S. Labyad. *Gradient-based estimation of linear Hawkes processes with general kernels*. 2021. arXiv Preprint: 2111.10637 (stat.ME).

[12] J. Chen, A. G. Hawkes, and E. Scalas. "A fractional Hawkes process". In: *Nonlocal and Fractional Operators*. Springer, 2021, pp. 121–131. arXiv: 2003.01027 [math.PR].

[13] X. Chen. "Perfect sampling of hawkes processes and queues with hawkes arrivals". In: *Stochastic Systems* 11.13 (2021), pp. 264–283. arXiv: 2002.06369 [math.PR].

[14] Y. Chen. "Likelihood function for multivariate Hawkes processes". In: *Preprint (5 pages) available on https://www.math.fsu.edu/ychen/research/HawkesLikelihood.pdf* (2016).

[15] Y. Chen. "Multivariate Hawkes Processes and Their Simulations". In: *Preprint* (Sept. 2016).

[16] Y. Chen. "Thinning Algorithms for Simulating Point Processes". In: *Preprint* (Sept. 2016).

[17] S. Crowley. "Exponential Hawkes Processes". In: *viXra, Preprint* (Nov. 2015).

[18] S. Crowley. "Point Process Models for Multivariate High-Frequency Irregularly Spaced Data". In: *Citeseer, Preprint* (Dec. 2012).

[19] D. J. Daley and D. Vere-Jones. *An introduction to the theory of point processes: Volume I, Elementary theory and methods*. Springer, 2002.

[20] A. Dassios and H. Zhao. "Exact simulation of Hawkes process with exponentially decaying intensity". In: *Electronic Communications in Probability* 8.62 (2013).

[21] N. Du et al. "Recurrent Marked Temporal Point Processes: Embedding Event History to Vector". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 1555–1564. ISBN: 9781450342322. DOI: 10.1145/2939672.2939875. URL: https://doi.org/10.1145/2939672.2939875.

[22] P. Embrechts, T. Liniger, and L. Lin. "Multivariate Hawkes processes: an application to financial data". In: *Journal of Applied Probability* 48.A (2011), pp. 367–378. DOI: 10.1239/jap/1318940477.

[23] E. Errais, K. Giesecke, and L. R. Goldberg. "Affine point processes and portfolio credit risk". In: *SIAM Journal on Financial Mathematics* 1.1 (June 2010), pp. 642–665. URL: https://ssrn.com/abstract=908045.

[24] V. Filimonov and D. Sornette. "Apparent criticality and calibration issues in the Hawkes self-excited point process model: application to high-frequency financial data". In: *Quantitative Finance* 15.8 (2015), pp. 1293–1314.

[25] V. Filimonov and D. Sornette. "Quantifying reflexivity in financial markets: Toward a prediction of flash crashes". In: *Phys. Rev. E* 85 (5 May 2012), p. 056108. DOI: 10.1103/PhysRevE.85.056108. URL: https://link.aps.org/doi/10.1103/PhysRevE.85.056108.

[26] Y. Guan and Y. Shen. "A weighted estimating equation approach for inhomogeneous spatial point processes". In: *Biometrika* 97.4 (2010), pp. 867–880. ISSN: 00063444, 14643510. URL: http://www.jstor.org/stable/29777142.

[27] S. J. Hardiman, N. Bercot, and J. P. Bouchaud. "Critical reflexivity in financial markets: a Hawkes process analysis". In: *The European Physical Journal B* 86.10 (Oct. 2013). ISSN: 1434-6036. DOI: 10.1140/epjb/e2013-40107-3. URL: http://dx.doi.org/10.1140/epjb/e2013-40107-3.

[28] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[29] A. Hawkes. "Hawkes processes and their applications to finance: a review". In: *Quantitative Finance* 18.2 (2018), pp. 193–198. DOI: 10.1080/14697688.2017.1403131. eprint: https://doi.org/10.1080/14697688.2017.1403131. URL: https://doi.org/10.1080/14697688.2017.1403131.

[30] A. G. Hawkes. "Spectra of Some Self-Exciting and Mutually Exciting Point Processes". In: *Biometrika* 58.1 (Apr. 1971), pp. 83–90.

[31] J. Hwang, S. Lay, and A. Lippman. "Non-parametric Multivariate Density Estimation: A Comparative Study". In: *IEEE Transactions on signal processing* 42.10 (Oct. 1994).

[32] B. Jing, S. Li, and Y. Ma. "Pricing VIX options with volatility clustering". In: *Journal of Futures Markets* 40.6 (June 2020), pp. 928–944. DOI: 10.1002/fut.22092. URL: https://ideas.repec.org/a/wly/jfutmk/v40y2020i6p928-944.html.

[33] M. Kirchner. "An estimation procedure for the Hawkes process". In: *Quantitative Finance* 17.4 (2017), pp. 571–595. arXiv: 1509.02017 [math.PR].

[34] M. Kirchner. "Perspectives on Hawkes Processes". PhD thesis. ETHZ, Jan. 2017.

[35] R. Kobayashi and R. Lambiotte. "Tideh: Time-dependent hawkes process for predicting retweet dynamics". In: *Tenth International AAAI Conference on Web and Social Media*. 2016. eprint: 1603.09449.

[36] P. J. Laub, T. Taimre, and P. K. Pollett. "Hawkes Processes". In: *arXiv preprint arXiv:1507.02822* (July 2015).

[37] E. Lewis and G. Mohler. "A nonparametric EM algorithm for multiscale Hawkes processes". In: *Journal of Nonparametric Statistics* 1.1 (2011), pp. 1–20.

[38] P. A. W. Lewis and G. S. Shedler. "Simulation of nonhomogeneous Poisson processes by thinning". In: *Naval research logistics quarterly* 26.3 (1979), pp. 403–413.
[39] K. W. Lim et al. "Simulation and calibration of a fully Bayesian marked multidimensional Hawkes process with dissimilar decays". In: *Asian Conference on Machine Learning*. PMLR. 2016, pp. 238–253. DOI: 10.48550/ARXIV.1803.04654.
[40] T. Liniger. "Multivariate Hawkes Processes". PhD thesis. ETHZ, Jan. 2009.
[41] Y. Malevergne, D. Sornette, and R. Wei. "A model of financial bubbles and drawdowns with non-local behavioral self-referencing". In: *Swiss Finance Institute Research Paper* 21-96 (2021).
[42] R. Martins and D. Hendricks. *The statistical significance of multivariate Hawkes processes fitted to limit order book data*. 2016. arXiv Preprint: 1604.01824 (q-fin.TR).
[43] L. Massoulié. "Stability results for a general class of interacting point processes dynamics, and applications. This work was partially done while the author was with the Laboratoire des Signaux et Systèmes, CNRS-ESE, Plateau de Moulon, 91192 Gif-sur-Yvette, France." In: *Stochastic Processes and their Applications* 75.1 (1998), pp. 1–30. ISSN: 0304-4149. DOI: https://doi.org/10.1016/S0304-4149(98)00006-4. URL: https://www.sciencedirect.com/science/article/pii/S0304414998000064.
[44] H. Mei and J. Eisner. "The neural hawkes process: A neurally self-modulating multivariate point process". In: *Advances in neural information processing systems* 30 (2017).
[45] J. Møller and J. G. Rasmussen. "Perfect Simulation of Hawkes Processes". In: *Applied Probability Trust in Advances in Applied Probability* 37.3 (Sept. 2005).
[46] M. Morariu-Patrichi. "High-frequency financial data modelling with hybrid marked point processes". PhD thesis. Imperial College, Oct. 2018.
[47] M. Morariu-Patrichi and M. Pakkanen. "State-dependent Hawkes processes and their application to limit order book modelling". In: *Quantitative Finance* (2021), pp. 1–21.
[48] S.A. Murphy and P.K. Sen. "Time-dependent coefficients in a Cox-type regression model". In: *Stochastic Processes and their Applications* 39.1 (1991), pp. 153–180. ISSN: 0304-4149. DOI: https://doi.org/10.1016/0304-4149(91)90039-F. URL: https://www.sciencedirect.com/science/article/pii/030441499190039F.
[49] Y. Ogata. "On Lewis' Simulation Method for Point Processes". In: *IEEE Transactions on signal processing* 27 (1 Jan. 1981).
[50] T. Omi, Y. Hirata, and K. Aihara. "Hawkes process model with a time-dependent background rate and its application to high-frequency financial data". In: *Physical Review E* 96.1 (July 2017). ISSN: 2470-0053. DOI: 10.1103/physreve.96.012303. URL: http://dx.doi.org/10.1103/PhysRevE.96.012303.
[51] T. Omi, N. Ueda, and K. Aihara. "Fully neural network based model for general temporal point processes". In: *Advances in neural information processing systems* 32 (2019). arXiv: 1905.09690 [cs.LG].
[52] T. Ozaki. "Maximum Likelihood Estimation of Hawkes' Self Exciting Point Processes". In: (Sept. 1977).
[53] R. Pasupathy. "Generating Nonhomogeneous Poisson Processes". In: (2011).
[54] M. Rambaldi, P. Pennesi, and F. Lillo. "Modeling foreign exchange market activity around macroeconomic news: Hawkes-process approach". In: *Phys. Rev. E* 91 (1 Jan. 2015), p. 012819. DOI: 10.1103/PhysRevE.91.012819. URL: https://link.aps.org/doi/10.1103/PhysRevE.91.012819.
[55] A. Reinhart. "A review of self-exciting spatio-temporal point processes and their applications". In: *Statistical Science* 33.3 (2018), pp. 299–318.
[56] M. Rizoiu et al. "A Tutorial on Hawkes Processes for Events in Social Media". In: (Oct. 2017).
[57] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Ed. by Chapman & Hall / CRC. 26. Monographs on Statistics and Applied Probability, 1986.
[58] R. Tibshirani and T. Hastie. "Local Likelihood Estimation". In: *Journal of the American Statistical Association* 82.398 (1987), pp. 559–567. ISSN: 01621459. URL: http://www.jstor.org/stable/2289465 (visited on 04/05/2022).
[59] I. M. Toke and F. Pomponio. "Modelling Trades-Through in a Limit Order Book Using Hawkes Processes". In: *Economics* 6.1 (2012), p. 20120022. DOI:

doi:10.5018/economics-ejournal.ja.2012-22. URL: https://doi.org/10.5018/economics-ejournal.ja.2012-22.

[60] P. Van Kerm. "Adaptive kernel density estimation". In: *The Stata Journal* 3.2 (May 2003), pp. 148–156.

[61] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Ed. by Chapman & Hall / CRC. 60. Monographs on Statistics and Applied Probability, 1995.

[62] X. Wang. "Approximating bayesian inference by weighted likelihood". In: *Canadian Journal of Statistics* 34.2 (2006), pp. 279–298. DOI: https://doi.org/10.1002/cjs.5550340206. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cjs.5550340206. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cjs.5550340206.

[63] A. Wehrli. "Financial Point Process modelling, the endo-exo problem, flash crashes and the excess volatility puzzle explained". PhD thesis. ETHZ, July 2021.

[64] A. Wehrli and D. Sornette. "Classification of flash crashes using the Hawkes(p,q) framework". In: *Quantitative Finance* 22.2 (2022), pp. 213–240. DOI: 10.1080/14697688.2021.1941212. eprint: https://doi.org/10.1080/14697688.2021.1941212. URL: https://doi.org/10.1080/14697688.2021.1941212.

[65] A. Wehrli and D. Sornette. "Excess Financial Volatility Explained by Endogenous Excitations Revealed by EM Calibrations of a Generalized Hawkes Point Process". In: *Swiss Finance Institute Research Paper* (Apr. 2021). URL: https://ssrn.com/abstract=3848661.

[66] A. Wehrli, S. Wheatley, and D. Sornette. "Scale-, time- and asset-dependence of Hawkes process estimates on high frequency price changes". In: *Quantitative Finance* 21.5 (2021), pp. 729–752. DOI: 10.1080/14697688.2020.1838602. eprint: https://doi.org/10.1080/14697688.2020.1838602. URL: https://doi.org/10.1080/14697688.2020.1838602.

[67] S. Wheatley, A. Wehrli, and D. Sornette. "The endo–exo problem in high frequency financial price fluctuations and rejecting criticality". In: *Quantitative Finance* 19.7 (2019), pp. 1165–1178.

[68] S. Wheatley, A. Wehrli, and D. Sornette. "The endo–exo problem in high frequency financial price fluctuations and rejecting criticality". In: *Quantitative Finance* 19.7 (2019), pp. 1165–1178. DOI: 10.1080/14697688.2018.1550266. eprint: https://doi.org/10.1080/14697688.2018.1550266. URL: https://doi.org/10.1080/14697688.2018.1550266.

[69] J. Zhuang. "Weighted likelihood estimators for point processes". In: *Spatial Statistics* 14 (2015). Spatio-Temporal Stochastic Modelling of Environmental Hazards, pp. 166–178. ISSN: 2211-6753. DOI: https://doi.org/10.1016/j.spasta.2015.07.009. URL: https://www.sciencedirect.com/science/article/pii/S221167531500072X.

Department of Mathematics, ETH, Zuerich
Email address: niels.carioukotlarek@yahoo.com