Learning Multivariate Hawkes Processes under Quantization Noise

Justin Deschenaux

School of Computer and Communication Sciences
Bachelor Semester Project Report

Supervisor
Wiliam Trouleau
EPFL / INDY2

Supervisor
Prof. Patrick Thiran
EPFL / INDY2

Abstract

Learning the causal relationships between epidemic processes is critical for policy-makers. Think of a disease spreading in a district, being able to predict where a future outbreak is likely going to strike is crucial to plan timely interventions. The challenge is that the collected data is usually aggregated at a coarse time granularity, drowning the signal into noise, and making the state-of-the-art approaches fail. We call the information loss due to aggregation *quantization noise*. A well-known model for such situations, where past events influence the probability of new ones is the *Multivariate Hawkes process* also known as *self-exciting point process*. The goal is to find an estimation procedure for the structure of this model when data is aggregated over fixed-size time intervals. We also consider the case of dimensions that are not synchronously observed, inducing a new source of errors. We call this effect *synchronization noise* and also investigate a possible mitigation against both sources of noise.

1 Introduction

The goal of this project is to find an estimation procedure for the diffusion dynamics of an epidemic spread over a network. The nodes represents arbitrary groups of individuals (e.g. different age categories or geographic locations) while the weight of the directed edges represents how strongly the disease propagates between those groups. A well established model for such situations is the multivariate Hawkes process (MHP). It is used when modeling a collection of random events distributed on the time axis happening in some node of the network. MHPs are also known as self-exciting point processes due to the fact that any event has the potential to generate new ones in the future. It was introduced by Hawkes and Oakes [1974] to model earthquakes and was more recently studied and used in other applications. For example, Trouleau et al. [2019] presents an estimation procedure when each dimension is observed with an unknown delay and Salehi et al. [2019] develops a procedure when only short sequences of events are available. It is used in a broad range of fields such as genome analysis (Reynaud-Bouret and Schbath [2010]), finance (Bacry et al. [2015], Kirchner [2017]) and can even model certain kind of criminal behaviours (Mohler et al. [2011]). MHPs have a wide expressive power.

We study the problem of learning MHPs under observation noise. Most approaches focus on the case where the events arrival times are known. However, it is not always possible to observe them. We rather only have access to a sequence of discrete counts representing the number of events that happened during fixed-size time intervals in each dimension. For instance, in case of an epidemic, hospitals of different cities could report the number of new cases in the past 12 hours. In this project, we investigate potential approaches to tackle this kind of noise in the observation. The solutions presented are as follows: in the case of synced observations, *i.e.* each dimension is aggregated over intervals starting and ending simultaneously, we adapt the procedure from Kirchner [2017], which performs inference on an autoregressive process, considering it as a model of the discrete sequence. In the case of a quantized process observed with a known delay between nodes (e.g. dimension 2 is observed 10 minutes after dimension 1), we use a maximum likelihood approach to estimate parameters of the process under the two kind of noises, *i.e.* quantization and synchronization noises.

This report is organized as follows: Section 2 presents formally the multivariate Hawkes process and the discrete time series representing the available observations. Section 3 focuses on the solution proposed by Kirchner [2017] and how to adapt it to our problem. We present the maximum likelihood estimation for the case of discrete and delayed observations in Section 4.

2 Model

The multivariate Hawkes process $\boldsymbol{N}(t)$ of dimension m models a collection of random events happening at $\{\{t_{1,j}, t_{2,j}, ..., t_{n_j,j}\}_j\}$, $1 \leq j \leq m$ distributed on the time axis and taking place in any node of a network. While the simplest of the point processes, the Poisson process, can only model events happening independently from each other, MHPs are more general in the sense that the instantaneous expected number of events can vary over time, depending on recent events. Given

that this model describes the causal relationships between dimensions, it is naturally described and visualised using graphs.

More formally, a m-dimensional multivariate Hawkes process $\boldsymbol{N}(t)$ is built from m dependent point processes as components, *i.e.* $\boldsymbol{N}(t) := (N_1(t), N_2(t), ..., N_m(t))$. Random points in dimension j are sampled according to the intensity function Λ_j, representing the instantaneous expected number of events in this node. Since the intensity depends on past random events, possibly in other dimensions, we express Λ_j assuming the set of arrival times in each dimension up to time t, *i.e.* $\{\{t_{1,j}, t_{2,j}, ..., t_{n,j}\}_j\}$ is known and referred as $\mathcal{H}_t$. The history of events in dimension j is referred as $\mathcal{H}_t^j$. Consequently, we define the conditional intensity as

$$\Lambda_i\left(t \mid \mathcal{H}_t\right) := \eta_i + \sum_{j=1}^{m} \int_{-\infty}^{t} h_{i \leftarrow j}(t-s) N_j(ds) = \eta_i + \sum_{j=1}^{m} \sum_{\tau \in \mathcal{H}_t^j} h_{i \leftarrow j}(t-\tau) \tag{1}$$

where $\eta_i \geq 0$ is called the baseline intensity and models events drawn independently from $\mathrm{Pois}(\eta_i)$. $h_{i \leftarrow j} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a measurable function called the excitation function, expressing how events in node j influences intensity in node i.

Typical choices for $h_{i \leftarrow j}(t)$ are one-sided exponentials and power laws. In our setup, we assumed that excitation functions are of the form:

$$h_{i \leftarrow j}(t) := \alpha_{i \leftarrow j} \beta \exp(-\beta t) 1_{t>0} \tag{2}$$

where $\alpha_{i \leftarrow j}, \beta$ are real-valued constants. $\alpha_{i \leftarrow j}$ controls the strength of influence from node j to node i while β defines the decay rate of the exponential. Consequently, multivariate Hawkes processes are characterized by a $m \times m$ matrix $A := [\alpha_{i \leftarrow j}]_{i,j}$ and a m-dimensional vector of baseline intensities η. Therefore, estimation is related to these parameters. Existence of non-degenerate MHPs is a subtle matter. See Liniger [2009] for a rigorous reference. The main prerequisite in our setup is that the magnitude of the largest eigenvalue of A (also known as spectral radius) is smaller than 1, to ensure stability, *i.e.* that the expected intensity remains bounded.

Quantization noise In the setting we study herein, we do not observe $\mathcal{H}_t$ exactly. We rather only access a time series of the number of events over some time, also called bins or bin counts, denoted as $\boldsymbol{X}_k \in \mathbb{N}^m$ for the vector of events in interval $((k-1)\Delta, k\Delta]$ or $X_j[k]$ for the number of events in dimension j between $(k-1)\Delta$ (excluded) and $k\Delta$ (included).

3 Learning MHPs under quantization noise

We leverage the procedure from Kirchner [2017]. Although it was designed for computational efficiency in financial engineering by aggregating data on purpose, we use the results presented to solve our problem. Note that the main difference between our setup and the settings of Kirchner [2017] is that he can choose Δ freely and aggregates data himself while we cannot, since it is a intrinsic parameter of the data.

The procedure is justified by the approximation of the expected number of events in the interval $((k-1)\Delta, k\Delta]$, given by

$$\mathbb{E}\left[\boldsymbol{X}_n \mid \mathcal{H}_{(n-1)\Delta}\right] = \int_{(n-1)\Delta}^{n\Delta} \mathbb{E}\left[\Lambda(t) \mid \mathcal{H}_{(n-1)\Delta}\right] \tag{3}$$

$$\overset{(a)}{\approx} \Delta\eta + \Delta \int_{-\infty}^{(n-1)\Delta} h(n\Delta - u)N(du) \tag{4}$$

$$\overset{(b)}{\approx} \Delta\eta + \Delta \int_{(n-p-1)\Delta}^{(n-1)\Delta} h(n\Delta - u)N(du) \tag{5}$$

$$\overset{(c)}{\approx} \Delta\eta + \sum_{k=1}^{p} \Delta h(\Delta k)\boldsymbol{X}_{n-k}. \tag{6}$$

In (a), we assumed that the events in interval $((k-1)\Delta, k\Delta]$ do not influence the intensity until next bin *i.e.* time-span $(k\Delta, (k+1)\Delta]$. Moreover, we also assume that the excitation function $h : \mathbb{R}_{\geq 0}^{m\times m} \to \mathbb{R}_{\geq 0}^{m\times m}$ is piece-wise constant on intervals of size Δ. Consequently, the integral on n-th bin reduces to the integral of a constant value. In step (b), we overlook the influence of events that happened long ago, hence larger values of p reduce the error. Finally, in (c), we assume that all events in time-span of the n-th bin take place at the start of it, *i.e.* arrivals at $(n-1)\Delta$ and use the definition with a sum instead of the one with the integral.

Kirchner [2017] shows that expression (6) converges to an integer-valued autoregressive process (INAR) defined by

$$\boldsymbol{X}_k = \boldsymbol{a}_0 + \sum_{r=1}^{p} A_r \boldsymbol{X}_{k-r} + \boldsymbol{u}_k \tag{7}$$

where $\boldsymbol{X}_k \in \mathbb{N}^m$, $\boldsymbol{a}_0$ is a constant vector, $\boldsymbol{A}_r \in \mathbb{R}^{m\times m}$ are matrices defining combination of past values to obtain $\boldsymbol{X}_n$. Finally, $\boldsymbol{u}_n$ is an integer-valued white-noise sequence. The procedure is based on the following theorem (see Kirchner [2016] for proof)

Theorem 3.1 *Let $N(t)$ be a univariate Hawkes process with baseline intensity $\eta > 0$ and piecewise-continuous excitation function $h : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ such that $\sum_{k=1}^{\infty} h(k\Delta)\Delta < 1$ for all $\Delta \in (0, 1)$. Furthermore, let (X_n) be a univariate INAR(∞) sequence with immigration parameter $\alpha_0 := \Delta\eta$ and reproduction coefficients $\alpha_k := \Delta h(k\Delta)$, $k \in \mathbb{N}$, and define a family of point processes by*

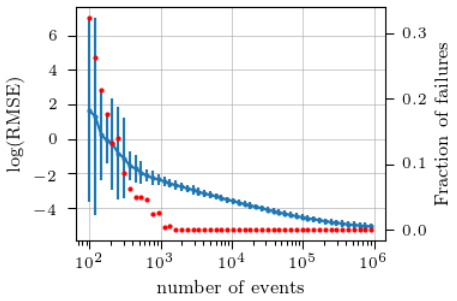
$$N^{(\Delta)}((a, b]) := \sum_{n:n\Delta\in(a,b]} , a < b, b \in (0, 1). \tag{8}$$

Then we have that, for $\Delta \downarrow 0$, the INAR(∞)-based family of point processes $(N^{(\Delta)})$ converges weakly to the Hawkes process N(t).

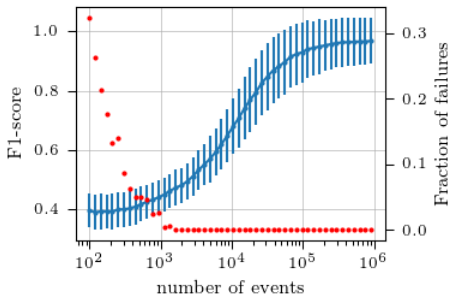
Kirchner [2017] did not work out the multivariate version of Theorem 3.1, but he presented simulations that support the assumption that it still holds in the multivariate case. Consequently, assuming that (6) converges to (7), the proposed solution is to perform conditional least-squares estimation on the data by considering it as a realization of an INAR process to find an estimate of parameters of the Hawkes process. For a m-dimensional integer sequence resulting from aggregating a m-dimensional MHP, we fit it to an INAR(p) model, for large support Δp to obtain estimators of baseline and excitation functions $\hat{\eta} := \frac{1}{\Delta}\hat{\boldsymbol{a}}_0$ and $\hat{H}(k\Delta) := \frac{1}{\Delta}\hat{A}_k$.

For a m-variate integer-valued sequence $(\boldsymbol{X}_k)_{k=1\ldots n}$, where we interpret $\boldsymbol{X}_k$ as a column vector, define $\hat{\theta}^{p,n}$ to be an estimator of the INAR(p) process from the n data bin counts.

$$\hat{\theta}^{p,n} := \boldsymbol{Y}\boldsymbol{Z}^\top(\boldsymbol{Z}\boldsymbol{Z}^\top)^{-1} \tag{9}$$



(a) Logarithm of the the root-mean-square error.



(b) F1-score of network edge detection.

Figure 1: Performance with respect to the number of events. The red dots represent the percentage of realizations for which estimation was not possible. The procedure may likely fail with less than 10^3 events. Note that we show the log(RMSE) instead of using a logscale on the y axis of caption (a) because the latter makes the error bars asymmetric, hence harder to visualize.

where $\boldsymbol{Z} := \begin{pmatrix} \boldsymbol{X}_p & \boldsymbol{X}_{p+1} & ... & \boldsymbol{X}_{n-1} \\ \boldsymbol{X}_{p-1} & \boldsymbol{X}_p & ... & \boldsymbol{X}_{n-2} \\ ... & ... & ... & ... \\ \boldsymbol{X}_1 & \boldsymbol{X}_2 & ... & \boldsymbol{X}_{n-p} \\ 1 & 1 & ... & 1 \end{pmatrix} \in \mathbb{R}^{(mp+1)\times(n-p)}$

and $\boldsymbol{Y} := (\boldsymbol{X}_{p+1}, \boldsymbol{X}_{p+2}, ..., \boldsymbol{X}_n) \in \mathbb{R}^{m\times(n-p)}$

Given those definitions, implementation of the algorithm is straightforward. Note that for some particular instances of $\boldsymbol{X}$, the matrix $\boldsymbol{Z}\boldsymbol{Z}^\top$ is not invertible, hence estimator cannot be computed. This issue does not show up for large n.

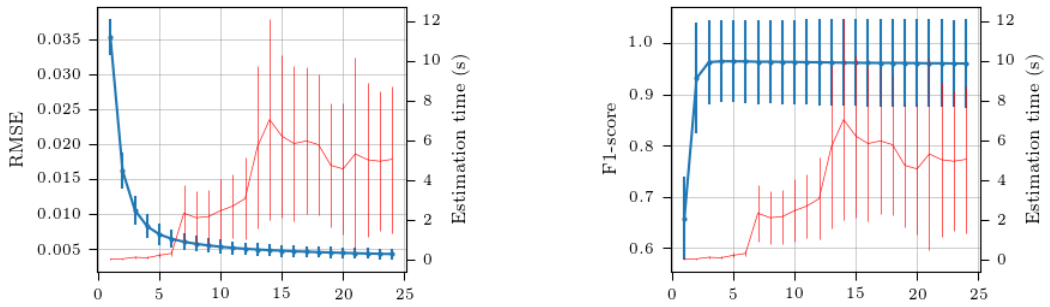
3.1 Experimental results

In order to understand how the procedure from Kirchner [2017] works when we cannot choose Δ, we carried out a few experiments. In order to do so, we first needed to generate synthetic data to work with. The structure of the processes was drawn as *Erdős–Rényi* graphs with edge probability $^{log(m)}/_{m}$. See appendix B for more details on synthetic data generation. The first step was to evaluate the performance of the algorithm on low dimensional MHPs. For a 3-dimensional inference example, see appendix C. After that, we carried 2 larger experiments. In the first one, we evaluated how the procedure is working with varying amount of training data. In the second one, we investigate the influence of the choice of the lag parameter p.

In the following experiments, we first want to evaluate how well the algorithm can recover the existence of edges. We detect edges by considering that there is an influence from j to i when the maximal value among samples $\hat{h}_{i\leftarrow j}$ is larger than a small threshold (0.05) and use the F1-score to quantify the edge detection performance. Secondly, we want to evaluate how well the algorithm can recover the weight of edges. The performance metric used in this case is the root-mean-square error. It is computed between estimated samples $\hat{h}_{i\leftarrow j}(k\Delta)$ and ground-truth samples $h_{i\leftarrow j}(k\Delta)$.

Performance with respect to the number of training events In this experiment, we tried to measure how the amount of available data is affecting the performance. We generated 10 graphs with $m = 10$ dimensions (see appendix B for more implementation details) with 30 realizations per graph. The decay parameter β was chosen such that the 95-th percentile of the kernels $h_{i\leftarrow j}$ falls at 5.5Δ. By choosing such a decay, we ensure that the influence flows over multiple bins *i.e.* the influence of events last long enough to still be visible in quantized data. Results are shown in Figure 1. We see that the procedure produces highly inaccurate results when few data is available, and a non-negligible fraction of experiments cannot be estimated when fewer than 10^3 events are observed.

From figure 1, we conclude that in our simulation setting, the risk of being unable to compute an estimate is high with less than 10^4 events. Moreover, when it is possible, the edge detection is bad.



(a) Root-mean-square error.

(b) F1-score of network edge detection.

Figure 2: The performance with respect to estimation lag p is plotted in blue. The red curve shows the mean computation time with lag p. The procedure is fast, even when we have $500'000$ events and a large lag.

Hence, this procedure is appealing when lots of data is available. It makes sense, as the procedure does few assumptions on what excitation functions $h_{i \leftarrow j}$ are. The amount of training data required for the same performance in higher dimensions is expected to increase by following a quadratic trend since the number of possible edges in the network grows as the square of the number of pairs of nodes, *i.e.* $\mathcal{O}(m^2)$.

Performance with respect to the estimation lag In this experiment, we tried to measure how different estimation lags p affect the performance under the same setup as previously, *i.e.* 10 graphs with 30 realizations each. The decay parameter β was chosen such that the 95-th percentile of $h_{i \leftarrow j}$ falls 5.5Δ after the event occurs to be able to observe influence in quantized data. Since we focus on the lag, we used a large number of events (500'000) to avoid issues induced by lack of data. The blue line represents the performance metric while the red one shows the mean run time in seconds of an estimation with the standard derivation as error bars. In this case, the procedure completed for all realizations, as expected after carrying out the previous experiment. Figure 2 shows the F1-score and root-mean-square error when varying p. We can observe that the RMSE decreases sharply while the F1-score peaks at the same time. These results make sense, since above a certain value of p, events far in the past do not influence the process anymore and are negligible in (6). Moreover, the procedure is fast, even for larger values of p, given how much events there are.

Unbalance between nodes Real-world Hawkes processes likely have a high number of dimensions and are sparse. Consequently, there may exist some imbalance in the number of events per node of the network. When computing an estimate for edges between ordered pairs of nodes where source dimension contains much fewer events than destination one, the procedure is less accurate than between nodes with a balanced number of events. This phenomenon can be observed when aggregating estimates $\hat{h}_{i \leftarrow j}(k\Delta)$ over multiple simulations on the same graph by the higher variance of the result than between dimensions with a similar number of events. This effect can be observed on the plot of Appendix C.

4 Learning MHPs under quantization and desynchronization

In this section, we introduce a second source of noise. Consider a scenario of epidemic spread, where hospitals from different time-zones report every 12 hours how many new cases occurred, but at different times of the day. For instance, hospital 1 may announce its results at 1 am/pm while hospital 2 does at 7 am/pm. We call this effect *synchronization noise* and the procedure from Kirchner [2017] does not work anymore when we introduce it. Indeed, it uses an estimator for synchronous integer sequences. Consequently, we decided to use a more standard technique based on maximum likelihood estimation. The idea is to find an expression for the expected number of events in each bin and then observe that it defines Poisson random variables. A related problem was studied in Trouleau et al. [2019] in order to estimate the causal structure of the Hawkes process when each dimension is

observed with an unknown phase difference, *i.e.* only arrival times local to the node they take place in are known.

Approximate ML for discrete observations Starting from the approximation in (6), we derive an expression for the expected number of events in bin n, given history of the process up to $\Delta(n-1)$, *i.e.* $\mathcal{H}_{\Delta(n-1)}$. For more details on the derivation, see Appendix A.

Suppose $\boldsymbol{N}(t)$ is a m-variate Hawkes process. Define $(\phi_1, \phi_2, ..., \phi_m)$ to be the vector of phases, such that $0 \leq \phi_i < \Delta$ and let $X_i[k] \in \mathbb{N}$ be the number of events during interval $((k-1)\Delta + \phi_i, k\Delta + \phi_i]$ in dimension i. Let $\tilde{\boldsymbol{N}}(t)$ be the process obtained by assuming that all events in bin $X_i[k]$ happened at time $(k-1)\Delta + \phi_i$. Moreover, we assume that these events only start to influence the intensity at the start of the next bin. Finally, let $h_{i \leftarrow j}$ be the excitation functions and $H_{i \leftarrow j}$ be the antiderivative of $h_{i \leftarrow j}$. The expected number of events in bin n of dimension i is given by

$$\mathbb{E}\left[X_i[n] \mid \mathcal{H}_{(n-1)\Delta}\right] = \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \Lambda_i(t)dt \mid \mathcal{H}_{(n-1)\Delta}\right] \tag{10}$$

$$\overset{(a)}{\approx} \Delta\eta_i + \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \sum_{j=1}^{m} \int_{-\infty}^{t} h_{i \leftarrow j}(t-s)\tilde{N}_j(ds)dt \mid \tilde{\mathcal{H}}_{(n-1)\Delta}\right] \tag{11}$$

$$\overset{(b)}{\approx} \Delta\eta_i + \sum_{j=1}^{m}\sum_{k=1}^{\infty} X_j[n-k]\left[H_{i \leftarrow j}(k\Delta + \phi_i - \phi_j) - H_{i \leftarrow j}((k-1)\Delta + \phi_i - \phi_j)\right] \tag{12}$$

$$\overset{(c)}{\approx} \Delta\eta_i + \sum_{j=1}^{m}\sum_{k=1}^{p} X_j[n-k]\left[H_{i \leftarrow j}(k\Delta + \phi_i - \phi_j) - H_{i \leftarrow j}((k-1)\Delta + \phi_i - \phi_j)\right] \tag{13}$$

In (a), we replaced process $\boldsymbol{N}(t)$ by $\tilde{\boldsymbol{N}}(t)$. In (b), we ignore the influence of event in current bins on the intensity function Λ_i *i.e.* events after $(k-1)\Delta$, rewrote the Lebesgue integral as a sum with events taking place at times $k\Delta$ and removed the expected value, since once conditioned on history, the expression is deterministic. We also integrated over t. Finally, for (c) we simply dropped the long term influence for a large p, such that the excitation function is close to zero. Typical values are such that the 95-th percentile of excitation is inside the interval $[0, (p+1)\Delta]$.

Once the history is known, the conditional intensity functions Λ_i are deterministic and define point-processes. By definition, the number of events in a certain interval (conditioned on history) is Poisson distributed, *i.e.*

$$P\left(X_i[k] = x_{i,k} \mid \mathcal{H}_{(n-1)\Delta}\right) = \frac{\lambda_i^{x_{i,k}}[k]}{x_{i,k}!} e^{-\lambda_i[k]} \tag{14}$$

where $\lambda_i[k] := \mathbb{E}\left[X_i[k] \mid \mathcal{H}_{(n-1)\Delta}\right]$. Note that we need to set boundary conditions when $1 \leq k \leq p$. Indeed, we use the number of events in the p previous bins. By default, we assume that it is zero.

Consequently, given a quantized realization of a MHP, by using expression (13), we can estimate the parameters $\alpha_{i \leftarrow j}$ and η_i. By applying the chain rule on the probability distribution, we obtain the likelihood function. Let $\boldsymbol{N}(t)$ be a m-dimensional MHP with n bins observed, taking values $x_{i,k}$. Then,

$$P\left(X_m[n] = x_{m,n}, X_m[n-1] = x_{m,n-1}, ..., X_j[k] = x_{j,k}, ..., X_1[1] = x_{1,1}\right) \tag{15}$$

$$= P\left(X_m[n] = x_{m,n} \mid X_m[n-1] = x_{m,n-1}, ..., X_j[k] = x_{j,k}, ..., X_1[1] = x_{1,1}\right) \cdot P\left(X_m[n-1] = x_{m,n-1}, ..., X_j[k] = x_{j,k}, ..., X_1[1] = x_{1,1}\right) \tag{16}$$

$$= \dots$$

$$= \prod_{i=1}^{m}\prod_{k=1}^{n} \frac{\lambda_i^{x_{i,k}}[k]}{x_{i,k}!} e^{-\lambda_i[k]}. \tag{17}$$

Consequently, the optimization problem is

$$\arg\max_{\alpha_{i\leftarrow j},\eta_i} \prod_{i=1}^{m}\prod_{k=1}^{n} \frac{\lambda_i^{x_{i,k}}[k]}{x_{i,k}!} e^{-\lambda_i[k]} = \arg\min_{\alpha_{i\leftarrow j},\eta_i} -\sum_{i=1}^{m}\sum_{k=1}^{n} x_{i,k}\log(\lambda_i[k]) - \log(x_{i,k}!) - \lambda_i[k] \quad (18)$$

$$\arg\min_{\alpha_{i\leftarrow j},\eta_i} \sum_{i=1}^{m}\sum_{k=1}^{n} \lambda_i[k] - x_{i,k}\log(\lambda_i[k]) \quad (19)$$

where $\lambda_i[k] := \mathbb{E}\left[X_i[k] \mid \mathcal{H}_{(k-1)\Delta}\right]$. We can minimize function (19) using gradient descent. As it is a product of very small values, it is numerically more suitable to optimize the logarithm of expression (17), which is equivalent because the logarithm is a monotonically increasing function. Note that $\lambda_i[k]$ is a function of $\alpha_{i\leftarrow j}$ and η_i parameters. It was not explicitly written to make (19) easier to read.

4.1 Experimental results

Observe that this problem is convex, which is only the case as we fix β a priori (see Bacry et al. [2015]). Therefore, to simplify the problem, we do not consider it as a parameter. Additionally, keeping β fixed makes the problem computationally more efficient. Indeed, when using numerical optimization libraries, we can precompute part of the expression to speed-up inference. For instance, when rewriting (13) using tensors, the weights attributed to previous bin counts, *i.e.* the part with the two $H_{i\leftarrow j}$ functions, only depends on p and β which are hyperparameters. Hence we do not need to recompute it at each optimization step. Finally, in order to implement the law of parsimony and favor simpler models, we introduce a regularization term to (19) corresponding to the $\mathcal{L}^2$ norm of the $(\alpha_{i\leftarrow j})$ and (η_i) parameters.

If we want to perform stochastic gradient descent and consider part of the whole data at each step, we have to decide how large each section is. This is the first of the hyperparameters we have to tune. The other ones are the step size, also known as learning rate, the regularization weight and finally the autoregression lag p and decay β. Note that in the case of epidemics, we can get a rough idea of the last two parameters by taking into account how long the disease makes people contagious. Tuning the hyperparameters is done by splitting the data into a training set and a validation set. The performance with a choice of parameters is defined as the loss function (19) on the validation data after training. Two possible ways of choosing how to split the data are as follows. First, we can take the start or end section of the integer sequence in order to preserve the causality relationship in the training data. The second option, when performing stochastic gradient descent, is to split the time series into k parts and choose any of them as validation data. The $k-1$ remaining parts are then used during training. Ideally, with enough computing power, we can perform k-fold cross-validations to tune hyperparameters.

For the following experiments, we could not explore the space of possible hyperparameters tuples exhaustively nor perform cross-validation due to the limited computing power available. Hence, we expect an overall lower performance and higher variance because we often don't find the best set of hyperparameters.

The graph structure of MHPs was generated exactly as in Section 3. In order to simulate the synchronization noise, we generated phases ϕ_i, $1 \leq i \leq m$ independently from $\mathrm{Uniform}[0, \Delta)$. Consequently, instead of aggregating all dimensions over synced intervals of width Δ, the arrival times were grouped independently on each node according to their phase ϕ_i. We considered the F1-score as the performance evaluation metric, but this time it was computed on the estimated adjacency matrix $\hat{A} := [\hat{\alpha}_{i\leftarrow j}]_{i,j}$, as opposed to function samples $\hat{h}_{i\leftarrow j}(k\Delta)$ in Kirchner [2017].

Performance with respect to the number of training events Given that the procedure is slower to complete, we could not estimate as much realizations as in the experiments on the procedure of Kirchner [2017]. We chose to generate 15 graphs and performed 10 simulations on each of them, one for each number of events considered. We fixed the β parameter to its ground truth value to reduce the hyperparameter space. We can see in Figure 3 that with 10^3 events we already perform better than the method of Kirchner [2017], even though we have a second source of observation noise. The plateau of performance we reach is expected to come from the fixed batch size we chose. This number was fixed after conducting smaller experiments where we performed full batch optimization

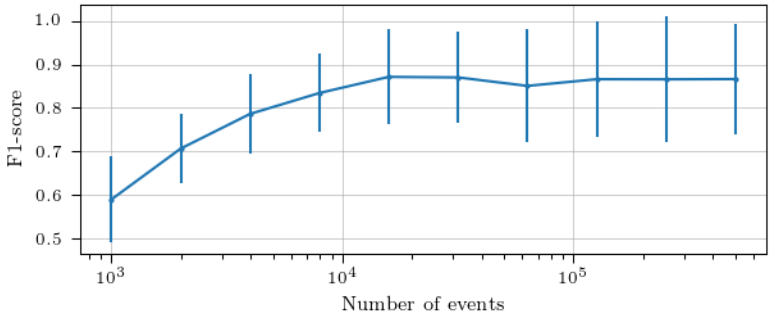


Figure 3: Performance with respect to number of events. The plateau is expected to come from the fixed batch size used during training.

with more than $100'000$ events. In this case, the procedure ended up most of the time with a very inaccurate adjacency matrix, leading to poorer performance than with less events. This is likely a consequence of the small hyperparameters space we used. Since we could not explore it more, as this experiment already required about 30 hours to complete, we chose to use a fixed batch size of $4'000$ bins, when it was possible to do so and a full-batch learning when less data was available. The results from this experiments invites us to work on this procedure further, but it requires more computing power and time than we currently have.

5 Conclusion

After working with estimation procedures for multivariate Hawkes processes, it makes sense to review the pros and cons of each of them. The procedure proposed in Kirchner [2017] is an elegant solution when we can only observe quantized Hawkes processes. It requires a lot of data as it does few assumptions on the excitation functions $h_{i \leftarrow j}$. Hence, it is generic and can work with a large number of excitation functions when there is enough data available. Moreover, it is easily implemented and only requires to choose one hyperparameter *i.e.* the autoregression lag p. It produces estimations for samples $h_{i \leftarrow j}(k\Delta)$ efficiently, as it was its primary goal. However, it is unable to deal with synchronization noise. The second approach, the likelihood-based one is more common. Since few experiments were performed about it, we must remain cautious in our interpretations. The experiment with respect to the number of events may indicate that it could work even when few data is available. However, optimizing (17) relies on numerical technique and has a few hyperparameters to tune. Hence, it is slower than the procedure from Kirchner [2017]. On the other hand, it seems to be a possible candidate to mitigate both quantization and synchronization noise. In order to assess this assumption, we definitely require more in depth experimentations.

There are a few extensions to this project that we could work on. The first one is to try both procedures on a real-world dataset to see what they predict. In the case of the likelihood optimization procedure, there are multiple axes of future research directions. First, it would be interesting to investigate how errors in the choice of the β parameter influences the performance. In the case of an epidemic, we usually have an approximate value for how long someone is contagious, hence can spread a disease. This fact could roughly guide us when choosing the hyperparameters β and p.

Acknowledgement

This project was my first encounter with scientific literature. It was tough and a bit scary at first to delve into a publication such as Kirchner [2017]. I would not have achieved this task without the precious guidance and advice from my supervisor William Trouleau. The project was well organized and he helped me to find new directions when I got stuck. I believe that I learned a lot during our weekly meetings. The first time I read the definition of the multivariate Hawkes processes from Wikipedia, I was worried this project may be too complicated. When looking back, I see that I

enjoyed this project a lot and I am glad that I was given the chance to work on MHPs in the context of epidemic diffusion.

A Appendix: Expected number of events (ML derivation)

Suppose $\boldsymbol{N}(t)$ is a m-variate Hawkes process. Define $(\phi_1, \phi_2, ..., \phi_m)$ to be the vector of phases, such that $0 \leq \phi_i < \Delta$ and let $X_i[k] \in \mathbb{N}$ be the number of events during interval $((k-1)\Delta + \phi_i, k\Delta + \phi_i]$ in dimension i. Let $\tilde{\boldsymbol{N}}(t)$ be the process obtained by assuming that all events in bin $X_i[k]$ happened at time $(k-1)\Delta + \phi_i$. Moreover, we assume that these events only start to influence the intensity at the start of the next bin. Finally, let $h_{i \leftarrow j}$ be the excitation functions and $H_{i \leftarrow j}$ be the antiderivative of $h_{i \leftarrow j}$. The expected number of events in bin n of dimension i is given by

$$\mathbb{E}\left[X_i[n] \mid \mathcal{H}_{(n-1)\Delta}\right]$$

$$\overset{(1)}{=} \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \Lambda_i(t)dt \mid \mathcal{H}_{(n-1)\Delta}\right]$$

$$\overset{(2)}{=} \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \left(\eta_i + \sum_{j=1}^{m} \int_{-\infty}^{t} h_{i \leftarrow j}(t-s) N_j(ds)\right) dt \mid \mathcal{H}_{(n-1)\Delta}\right]$$

$$\overset{(3)}{\approx} \Delta\eta_i + \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \sum_{j=1}^{m} \int_{-\infty}^{t} h_{i \leftarrow j}(t-s) \tilde{N}_j(ds)dt \mid \tilde{\mathcal{H}}_{(n-1)\Delta}\right]$$

$$\overset{(4)}{=} \Delta\eta_i + \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \sum_{j=1}^{m} \sum_{\tau_j \in \tilde{\mathcal{H}}_t^j} h_{i \leftarrow j}(t-\tau_j)dt \mid \tilde{\mathcal{H}}_{(n-1)\Delta}\right]$$

$$\overset{(5)}{\approx} \Delta\eta_i + \mathbb{E}\left[\int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} \sum_{j=1}^{m} \sum_{k=-\infty}^{n-1} X_j[k] h_{i \leftarrow j}(t - \Delta k - \phi_j)dt \mid \tilde{\mathcal{H}}_{(n-1)\Delta}\right]$$

$$\overset{(6)}{=} \Delta\eta_i + \sum_{j=1}^{m} \sum_{k=-\infty}^{n-1} X_j[k] \int_{(n-1)\Delta+\phi_i}^{n\Delta+\phi_i} h_{i \leftarrow j}(t - \Delta k - \phi_j)dt$$

$$\overset{(7)}{=} \Delta\eta_i + \sum_{j=1}^{m} \sum_{k=-\infty}^{n-1} X_j[k] \left[H_{i \leftarrow j}((n-k)\Delta + \phi_i - \phi_j) - H_{i \leftarrow j}((n-k-1)\Delta + \phi_i - \phi_j)\right]$$

$$\overset{(8)}{=} \Delta\eta_i + \sum_{j=1}^{m} \sum_{k=1}^{\infty} X_j[n-k] \left[H_{i \leftarrow j}(k\Delta + \phi_i - \phi_j) - H_{i \leftarrow j}((k-1)\Delta + \phi_i - \phi_j)\right]$$

$$\overset{(9)}{\approx} \Delta\eta_i + \sum_{j=1}^{m} \sum_{k=1}^{p} X_j[n-k] \left[H_{i \leftarrow j}(k\Delta + \phi_i - \phi_j) - H_{i \leftarrow j}((k-1)\Delta + \phi_i - \phi_j)\right]$$

$$\overset{(10)}{=} \Delta\eta_i + \sum_{j=1}^{m} \alpha_{i \leftarrow j} \sum_{k=1}^{p} X_j[n-k] \left[\exp(-\beta((k-1)\Delta + \phi_i - \phi_j)) - \exp(-\beta(k\Delta + \phi_i - \phi_j))\right].$$

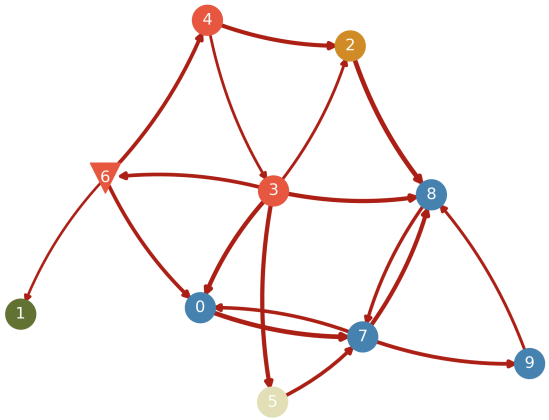


Figure 4: Typical underlying graph structure of synthetic data. Edge thickness between nodes i and j depends on magnitude of $\alpha_{i \leftarrow j}$. Triangular shaped nodes have a self loop. Nodes of the same color form a strongly-connected component.

The steps are:

(1) Since Λ_i describes the instantaneous expected number of events, the expected values of bin counts are obtained through integration.

(2) From definition of Λ_i.

(3) Replace process $\boldsymbol{N}$ by $\tilde{\boldsymbol{N}}$ where events in bin k in dimension j start to influence the intensity only at the start of the next bin. For example, all events in $((k-1)\Delta, k\Delta]$ induce an increase from time $k\Delta$ and like if they all happened at time $(k-1)\Delta$.

(4) Rewrite the Lebesgue integral as sum.

(5) Since we changed $\boldsymbol{N}(t)$ into $\tilde{\boldsymbol{N}}(t)$, events take place at starts of bins, *i.e.* at times of the form $\Delta k + \phi_j$ for dimension j. The added $X_j[k]$ terms represents the number of events that happened in the interval. Moreover, we neglect the influence of events on the bin in which they take place.

(6) Inverting order of integral and sums as well as expected value of a constant term.

(7) Excitation are one-sided exponential functions, hence easy to integrate. $H_{i \leftarrow j}$ is the antiderivative of $h_{i \leftarrow j}$.

(8) Change of variable $\tilde{k} = n - k$. Consequently, since k goes from $-\infty$ to $n-1$, $\tilde{k}$ goes from ∞ to 1. For convenience, $\tilde{k}$ is written as k and the iteration order is inverted.

(9) We neglect the long term influence of events. It makes sense, since we use exponentially decaying functions. Unsurprisingly, larger p induces smaller error (at computational cost).

(10) Rewriting $H_{i \leftarrow j}$ yields the final expression.

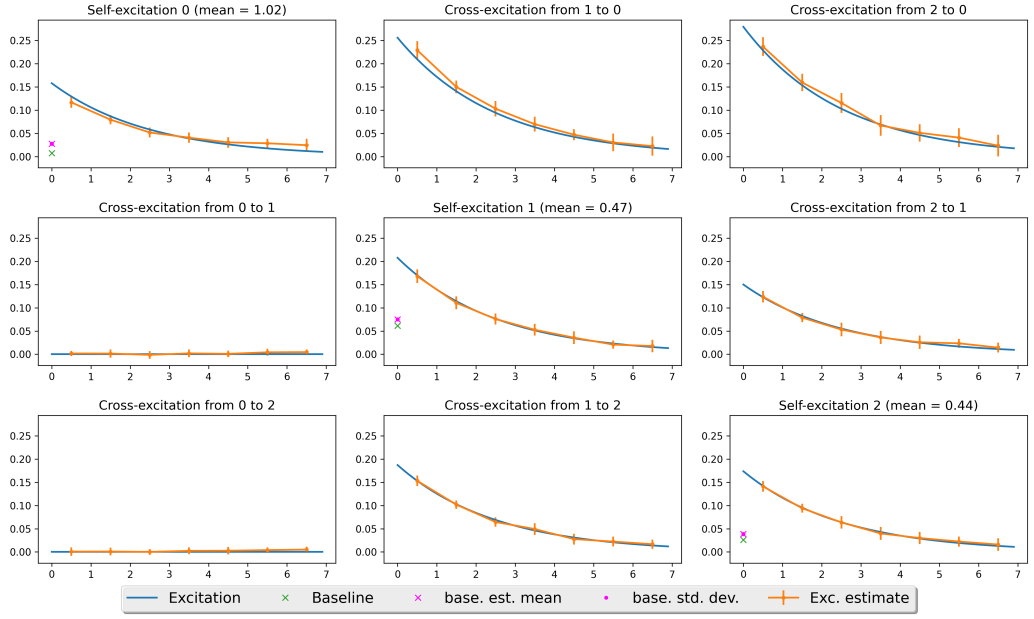
B Synthetic data generation

In this project, we simulate MHPs (with exponential kernels) using the `tick` python module[1]. In order to simulate a m-dimensional process, we provide a $m \times m$ adjacency matrix $A := [\alpha_{i \leftarrow j}]_{i,j}$ whose entries are non-zero with probability $\log(m)/m$, *i.e.* the threshold for asymptotic connectivity of random graphs. The non-zero entries are drawn from $\mathrm{Uniform}[0.5, 1]$, then scaled such that spectral radius has a fixed value (0.9). We also provide decay β such that 95-th percentile of exponential falls in the middle of the p-th bin. This way, the procedures can observe the influence flow on multiple bins. Finally, we provide a m-dimensional vector with entries drawn from $\mathrm{Uniform}[0, 0.1]$ as baseline and a number of events to simulate. Once the simulation is over, we quantize the realization over

[1] find on GitHub `https://github.com/X-DataInitiative/tick`

intervals of width Δ to obtain the time series $\boldsymbol{X}$. See Figure 4 for an example of graph generated by this procedure.

C 3-Dimensional proof of concept



References

Emmanuel Bacry, Iacopo Mastromatteo, and Jean-François Muzy. Hawkes processes in finance. 2015.

Alan Hawkes and D.A. Oakes. A cluster process representation of a self-exciting process. *Journal of Applied Probability*, 11, 09 1974. doi: 10.2307/3212693.

Matthias Kirchner. Hawkes and INAR(∞) processes. *Stochastic Processes and their Applications*, 126, 03 2016. doi: 10.1016/j.spa.2016.02.008.

Matthias Kirchner. An estimation procedure for the hawkes process. 17(4):571 – 595, 2017. ISSN 1469-7688. doi: 10.1080/14697688.2016.1211312. Published online 12 September 2016.

Thomas Josef Liniger. Multivariate hawkes processes. 2009. doi: 10.3929/ethz-a-006037599. Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 18403, 2009.

G. O. Mohler, M. B. Short, P. J. Brantingham, F. P. Schoenberg, and G. E. Tita. Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, 106(493):100–108, 2011. doi: 10.1198/jasa.2011.ap09546. URL `https://doi.org/10.1198/jasa.2011.ap09546`.

Patricia Reynaud-Bouret and Sophie Schbath. Adaptive estimation for hawkes processes; application to genome analysis. *Ann. Statist.*, 38(5):2781–2822, 10 2010. doi: 10.1214/10-AOS806. URL `https://doi.org/10.1214/10-AOS806`.

Farnood Salehi, William Trouleau, Matthias Grossglauser, and Patrick Thiran. Learning hawkes processes from a handful of events. 2019.

William Trouleau, Jalal Etesami, Matthias Grossglauser, Negar Kiyavash, and Patrick Thiran. Learning Hawkes processes under synchronization noise. 97:6325–6334, 09–15 Jun 2019. URL `http://proceedings.mlr.press/v97/trouleau19a.html`.