Fraunhofer ITWM

TU Kaiserslautern

Department of Mathematics

MASTER THESIS

Neural Hawkes Processes with Applications in Finance

Neuronale Hawkes Prozesse mit Anwendungen in der Finanzwirtschaft

Author:
Peter KRETSCHMER

Supervisor:
Prof. Dr. Ralf KORN
Dr. Robert KNOBLOCH

A thesis submitted in fulfilment of the requirements for the degree of
Master of Science

September 2019

Declaration of Authorship

I, PETER KRETSCHMER, hereby declare the following thesis titled 'Neural Hawkes Processes with Applications in Finance' ('Neuronale Hawkes Prozesse mit Anwendungen in der Finanzwirtschaft') to be my own work and I confirm that:

— The thesis I am submitting is entirely my own work except where otherwise indicated.

— It has not been submitted, either partially or in full, for a qualification at this or any other University.

— I have clearly signalled the presence of all material I have quoted from other sources, including any diagrams, charts, tables or graphs.

— I have acknowledged appropriately any assistance I have received.

Signature	Place, Date

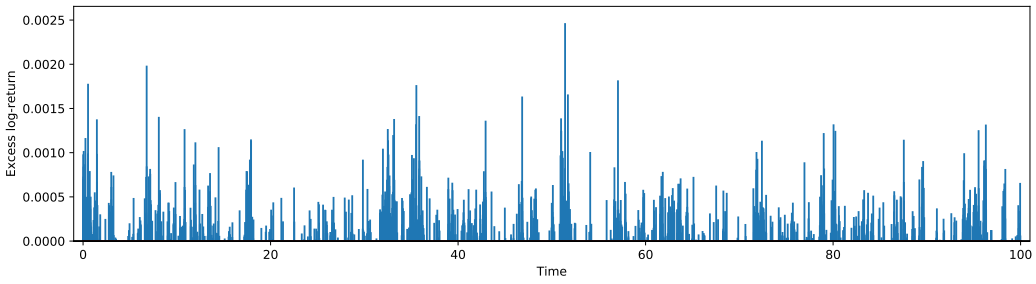
Abstract

In this thesis, we apply point processes based on Recurrent Neural Networks to financial data. The common approach assumes a fixed, parametric form for the dynamics of a process (e.g. Poisson process, Hawkes process). In this way, the imposed structure is often over-simplified and unable to capture the real dynamics of a process. Inspired by the exceptional modelling abilities and the great flexibility of Neural Networks, point processes based on Recurrent Neural Networks promise a more reasonable model. In summary, the main contributions of this thesis are:

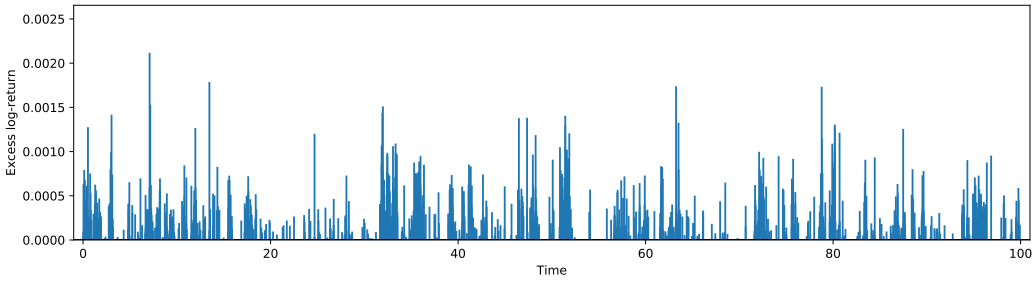
— We extend the Hawkes process and the Neural Hawkes process, a Neural Network based point process, to model multivariate point processes with additional continuous mark information, as these often occur in a financial context.

— We show that the Neural Hawkes process can be seen as a natural extension of already existing parametric point processes.

— We compare the Neural Hawkes process with different parametric models on a real-world dataset consisting of excess returns of the NASDAQ 100 Index. To the best of our knowledge, this thesis is the first to study such a model in this context. It turns out that depending on the task, the Neural Hawkes process yields comparable or even superior performance to the 'best' parametric model.



(a) Threshold excesses of the negative return process.



(b) Threshold excesses of the positive return process.

Figure: Simulated excess returns of a network-based point process model trained on minutely NASDAQ 100 data.

Acknowledgements

I would like to express my gratitude to my supervisors Prof. Dr. Ralf Korn and Dr. Robert Knobloch for giving me the opportunity of writing a thesis, which combines Deep Learning and Quantitative Finance. Especially since the field is still at the very beginning, I am grateful for the supervision and guidance.

I would like to thank my friend Magnus Wiese for the valuable discussions and the constant advice.

After all, this thesis would not have been possible without the continuous support of my family and friends throughout my years of study.

Thank you.

Contents

1 Introduction **1**

2 Point Processes **3**
2.1 Temporal Point Processes . . . 3
2.1.1 Poisson Process . . . 11
2.1.2 Non-homogeneous Poisson Process . . . 17
2.1.3 Hawkes Process . . . 18
2.2 Marked Temporal Point Processes . . . 21
2.2.1 Marked Poisson Process . . . 26
2.2.2 Marked Hawkes Process . . . 27
2.3 Maximum Likelihood Estimation . . . 31
2.4 Goodness-of-Fit Tests . . . 35
2.5 Simulation . . . 40
2.5.1 Simulation of Poisson Processes . . . 40
2.5.2 Simulation of Temporal Point Processes . . . 40
2.5.3 Simulation of Marked Temporal Point Processes . . . 44

3 Neural Point Processes **47**
3.1 Feedforward Neural Networks . . . 47
3.2 Recurrent Neural Networks . . . 49
3.2.1 Backpropagation through Time . . . 53
3.2.2 Gradient Descent algorithms . . . 55
3.2.3 Exploding and vanishing gradients problem . . . 59
3.2.4 Long Short-Term Memory . . . 61
3.2.5 Regularization . . . 64
3.3 Neural Hawkes Process . . . 65

4 Numerical Study **74**
4.1 Bivariate Poisson Process with conditionally i.i.d. marks . . . 77
4.2 Bivariate Hawkes Process with continuous marks . . . 78
4.3 Bivariate Neural Hawkes Process with continuous marks . . . 80
4.4 Numerical Study - Figures . . . 82
4.4.1 Bivariate Poisson Process with conditionally i.i.d. marks . . . 82
4.4.2 Bivariate Hawkes Process with continuous marks . . . 85
4.4.3 Bivariate Neural Hawkes Process with continuous marks . . . 88
4.5 Numerical Study - Tables . . . 91
4.6 Summary . . . 91

5 Conclusion and Outlook **93**

Bibliography **94**

List of Figures

2 Point Processes

2.1 Event times, inter-event times and counting process of a temporal point process. . . . 4
2.2 Conditional intensity function of a homogeneous Poisson process for different intensities λ. . . . 15
2.3 Power law intensity function for $\alpha = 1$ and different values of β. . . . 17
2.4 Conditional intensity function of an exponential Hawkes process with baseline intensity $\mu = 0.5$ and different values for the branching coefficient ϑ and the decay parameter α. . . . 19
2.5 Immigrant-Descendant representation of a Hawkes process with immigrant clusters. . . . 20
2.6 Conditional intensity functions of a bivariate Hawkes process. . . . 29
2.7 Illustration of Ogata's modified thinning algorithm for a univariate Hawkes process. . . . 42

3 Neural Point Processes

3.1 Overview of different activation functions. . . . 48
3.2 FNN with 2 hidden layers, input dimension $N_0 = 3$, output dimension $N_3 = 1$ and hidden dimensions $N_1 = N_2 = 5$. . . . 49
3.3 Computational graph of a recurrent neural network. . . . 51
3.4 Fully connected graphical model of the joint distribution of $x_1, ..., x_N$. . . . 52
3.5 Filtration based graphical model of the joint distribution of $x_1, ..., x_N$. . . . 52
3.6 Calculation of gradients using Backpropagation through time (BPTT). . . . 53
3.7 Vanishing Gradients in Standard RNN. . . . 59
3.8 LSTM Cell. . . . 62
3.9 Gradient Flow in an LSTM. . . . 64
3.10 Early Stopping 64
3.11 Continuous-Time LSTM for $t \in (t_{i-1}, t_i]$. . . . 66
3.12 Scaled softplus function for different scale parameters s together with the $ReLU$ function which it approaches as $s \to 0$. . . . 67
3.13 Illustration of the conditional intensity functions of a bivariate Neural Hawkes process. . . . 72

4 Numerical Study

4.1 Minute-by-minute closing prices of the NASDAQ 100 Index. . . . 75
4.2 Minute-by-minute log-returns of the NASDAQ 100 Index. . . . 75
4.3 Bivariate TPP with absolute threshold excesses. . . . 75

Bivariate Poisson Process with conditionally i.i.d. marks **82**
4.4 Conditional intensity function of the negative and positive return process. 82
4.5 Scale parameter of the mark exponential distribution. 82
4.6 QQ-Plot Exp(1) - Inter-event times of the residual process. 83
4.7 Conditional Uniformity Test of residual process with 95% confidence bands. 83
4.8 Autocorrelation function of the inter-event times of the residual process. . 83
4.9 QQ-Plot Unif$([0,1])$ - Marks of the residual process. 84
4.10 Simulated event sequence with final time $T = 100$. 84

Bivariate Hawkes Process with continuous marks **85**
4.11 Conditional intensity functions of the negative and positive return process. 85
4.12 Scale parameter of the mark exponential distribution. 85
4.13 QQ-Plot Exp(1) - Inter-event times of the residual process. 86
4.14 Conditional Uniformity Test of residual process with 95% confidence bands. 86
4.15 Autocorrelation function of the inter-event times of the residual process. . 86
4.16 QQ-Plot Unif$([0,1])$ - Marks of the residual process. 87
4.17 Simulated event sequence with final time $T = 100$. 87

Bivariate Neural Hawkes Process with continuous marks **88**
4.18 Conditional intensity functions of the negative and positive return process. 88
4.19 Scale parameter of the mark exponential distribution. 88
4.20 QQ-Plot Exp(1) - Inter-event times of the residual process. 89
4.21 Conditional Uniformity Test of residual process with 95% confidence bands. 89
4.22 Autocorrelation function of the inter-event times of the residual process. . 89
4.23 QQ-Plot Unif$([0,1])$ - Marks of the residual process. 90
4.24 Simulated event sequence with final time $T = 100$. 90

List of Tables

4.1 p-values of the Kolmogorov-Smirnov Test for the $\mathrm{Exp}(1)$-distribution of the inter-event times of the residual process. 91
4.2 p-values of the Kolmogorov-Smirnov Test for the $\mathrm{Unif}([0,1])$-distribution of the marks of the residual process. 91
4.3 Log-likelihood per event. 91

Chapter 1

Introduction

Event streams consist of a sequence of events randomly located in time. They are common in many fields including

— seismology, e.g. the occurrence of earthquakes and aftershocks,

— medicine, e.g. the spread of epidemics,

— web traffic, e.g. consumer behaviours,

— insurance, e.g. the occurrence of claims,

— finance, e.g. the occurrence of extreme stock returns.

Event streams can range from just containing temporal information up to including marks providing additional details (e.g. the type) of the occurred events. Generally, these streams can possess several different structural properties. Future events can be independent of past events; they can be excited or inhibited by past events in a simple or highly complex and non-linear manner.

The modelling of event streams is therefore fundamentally different from time series modelling, as events occur randomly and not deterministically in time. In order to understand the dynamics of the underlying process and thereby being able to inferring relevant characteristics of the sequences at hand, the class of *temporal* and *marked temporal point processes* provides a consistent mathematical framework applicable in practise. (Marked) temporal point processes can be seen as a special kind of point processes for which the underlying process is located in time.

The most common technique to specify (marked) temporal point processes is in terms of their *conditional intensity function*. The conditional intensity function is in its essence a stochastic process, which iteratively models the next event conditioned on the past of the process. It can be shown that under certain conditions, the distribution of a (marked) temporal point process is uniquely defined by its conditional intensity function. Moreover, desirable structural properties of a process can easily be incorporated and aspects when dealing with real data including the estimation of model parameters, assessing the model quality by Goodness-of-Fit Tests and simulating event streams can all be performed based on the conditional intensity function.

Amongst the most popular examples of (marked) temporal point processes are the *homogeneous* and *non-homogeneous Poisson process* as well as the *Hawkes process*, which can all be defined in terms of their conditional intensity function. Whereas the homogeneous

Poisson process assumes independence of the events and stationarity in time, the non-homogeneous breaks the latter. Nonetheless, in many real-world applications temporal clusters exist as past events excite the occurrence of future events. In this case, the Hawkes process as proposed in [22, 23] provides a consistent modelling framework often being applied in practise [12, 37, 39]. In recent years, the Hawkes process turned out to be an appropriate model for many applications in finance including high-frequency dynamics, risk management, market impact and orderbook modelling [1, 13, 14].

However, modelling (marked) temporal point processes by assuming a pre-defined, parametric form of the conditional intensity function suffers from model misspecification, which can lead to an oversimplified and incorrect modelling of the real dynamics of a process. Recently, several papers [11, 36, 53] addressing this issue were published. All approaches propose to use a *Recurrent Neural Network* architecture to parametrize the conditional intensity function instead of assuming a fixed parametric form. This is mainly motivated by the exceptional modelling abilities of Recurrent Neural Networks in many sequence-related tasks [10]. Therefore, the proposed approaches are promising in terms of a more realistic model for highly complex and non-linear dynamics of real event streams. In particular, the *Neural Hawkes process* as proposed in [36] provides a consistent and flexible model based on a newly derived *continuous-time Long-Short Term Memory* network.

In this thesis we address this issue and compare common parametric models for (marked) temporal point processes to the newly derived Neural Hawkes process with regard to modelling financial point processes. For this reason, we enrich already existing processes in the literature to model multivariate temporal point processes with additional continuous mark information. Furthermore, we prove that the modelling capabilities of the Neural Hawkes process surpass the ones of common parametric models like the Hawkes process and thereby demonstrate that the Neural Hawkes process can be seen as the natural extension of these models.

The thesis is structured as follows. In Chapter 2, we derive the mathematical framework of temporal and marked temporal point processes. We formally introduce these concepts and derive specifications based on the conditional intensity function. At the end of this chapter, we present important topics when dealing with real data including Maximum Likelihood Estimation (Section 2.3), Goodness-of-Fit Tests (Section 2.4) and Simulation algorithms (Section 2.5). In Chapter 3, we give a rigorous definition of Recurrent Neural Networks (Section 3.2) thematising all relevant topics needed to introduce the Neural Hawkes process in Section 3.3. We conclude the thesis with a numerical study in Chapter 4, comparing different approaches for modelling extreme minute-by-minute excess returns of the NASDAQ 100 Index.

Chapter 2

Point Processes

In this chapter, we introduce the main concepts of temporal point processes and marked temporal point processes. Furthermore, we derive the likelihood for Maximum Likelihood estimation in Section 2.3, Goodness-of-Fit Tests in Section 2.4 as well as Simulation algorithms in Section 2.5. The presented framework mostly stands in line with [8], [43] and [45].

2.1 Temporal Point Processes

A *temporal point process* can be seen as a model for events, which stochastically occur in the time dimension. For different purposes, it is advantageous to have several equivalent definitions at hand. These are based on

— the *event times* $(t_i)_{i\in\mathbb{N}}$,

— the *inter-event times* $(\tau_i)_{i\in\mathbb{N}}$,

— the *counting process* $(N(t))_{t\geq 0}$,

— the *counting measure* $\mathcal{N}$.

A graphical illustration of the concepts is given in Figure 2.1.

Definition 2.1.1 (Temporal Point Process)
A stochastic process of event times $(t_i)_{i\in\mathbb{N}}$ *with*

1. $t_i > 0$ *and* $t_i < t_{i+1}$ *for each* $i \in \mathbb{N}$,
2. $|\{i \in \mathbb{N} : t_i \in A\}| < \infty$ *for each bounded* $A \subset \mathbb{R}_{>0}$,

is called temporal point process (TPP). The history of the process is given by the filtration $(\mathcal{H}_t)_{t\geq 0}$ *with* $\mathcal{H}_t := \sigma(t_i | t_i < t)$ *being the sigma-algebra generated by the events up to, but not including time* t*. Further, the filtration* $(\mathcal{H}_t^+)_{t\geq 0}$ *with* $\mathcal{H}_t^+ := \sigma(t_i | t_i \leq t)$ *is generated by the events up to and including time* t*.*

A temporal point process according to Definition 2.1.1 is *simple*, as $t_i < t_{i+1}$. We could also allow several events to occur at the same time, i.e. $t_i = t_{i+1}$, but this is not necessary for our purposes. The additional assumption 2 ensures that the number of events in every bounded time set is finite, which prevents the process from *exploding*.

Instead of defining the event times, we can equivalently describe a TPP in terms of its *inter-event times*, which correspond to the time durations between two successive events.

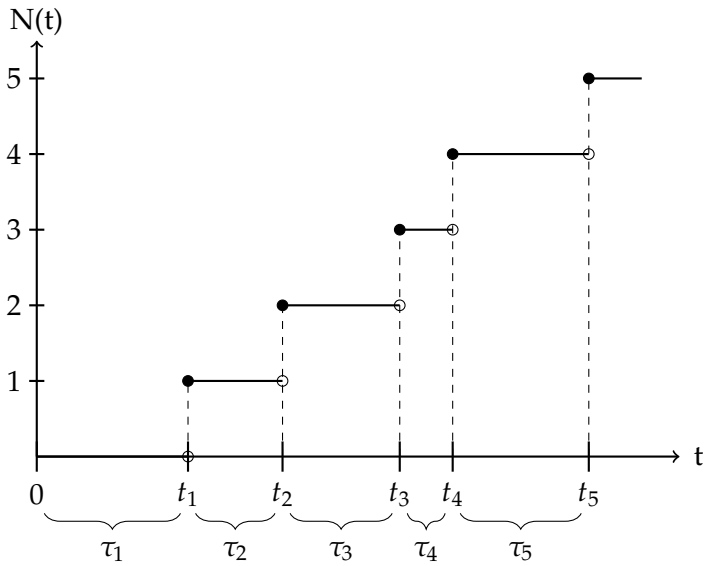


Figure 2.1: Event times, inter-event times and counting process of a temporal point process.

Definition 2.1.2 (Inter-event times)
Consider a TPP $(t_i)_{i\in\mathbb{N}}$. *Define the stochastic process* $(\tau_i)_{i\in\mathbb{N}}$ *by*

$$\tau_i := t_i - t_{i-1}\,,\ \ i \in \mathbb{N}$$

with $t_0 := 0$. *Then,* $(\tau_i)_{i\in\mathbb{N}}$ *is called the process of inter-event times and* τ_i *the i-th inter-event time.*

Vice versa, summing up the inter-event times gives the sequence of the event times. In this case, property 2 of Definition 2.1.1 is not always fulfilled. Based on the Strong Law of Large Numbers, we derive a simple condition on the inter-event times, which ensures this to hold.

Theorem 2.1.3
Consider a process of independent, identically distributed (i.i.d.) inter-event times $(\tau_i)_{i\in\mathbb{N}}$ *with* $\tau_i > 0$. *Assume further that* $0 < \mu := \mathbb{E}[\tau_i] < \infty$. *The process of event times* $(t_i)_{i\in\mathbb{N}}$ *defined by*

$$t_i := \sum_{j=1}^{i} \tau_j$$

is a TPP according to Definition 2.1.1.

Proof. We show the characterizing properties of a TPP.

1. Clear, as $(t_i)_{i\in\mathbb{N}}$ is the process of the cumulative sums of positive random variables.
2. By the Strong Law of Large Numbers (proof given in [45, Theorem 6.4.8]), it holds

$$\mathbb{P}\left(\lim_{i\to\infty} \frac{t_i}{i} = \mu\right) = 1\,.$$

This implies that

$$\mathbb{P}\left(\lim_{i\to\infty} t_i = \infty\right) = 1\,. \tag{2.1}$$

Hence, let $A \subset \mathbb{R}_{>0}$ bounded. There exist $a, b \geq 0$ such that $A \subset (a, b]$. Therefore, we obtain

$$\begin{aligned} |\{i \in \mathbb{N} : t_i \in A\}| &\leq |\{i \in \mathbb{N} : t_i \in (a, b]\,| \\ &= |\{i \in \mathbb{N} : t_i \leq b\}| - |\{i \in \mathbb{N} : t_i \leq a\}| < \infty \end{aligned}$$

using Equation 2.1.

□

Theorem 2.1.3 can be applied to the important class of *renewal processes*, which are TPPs with i.i.d. inter-event times [1]. If we allow for history- or time-dependent distributions of the inter-event times, we need other results. We derive these in the later chapters for special examples of such processes.

A third approach for defining a TPP is via its *counting process*.

Definition 2.1.4 (Counting process)
A stochastic process $N := (N(t))_{t \geq 0}$ is called (simple) counting process, if

1. $N(t) \in \mathbb{N}_0$ [2],
2. *N is right-continuous,*
3. $N(s) \leq N(t)$ *for* $s \leq t$,
4. $N(dt) := N(t + dt) - N(t) \in \{0, 1\}$ *for a sufficiently small, random dt.*

Theorem 2.1.5
Let $(t_i)_{i \in \mathbb{N}}$ be a TPP. The stochastic process $N = (N(t))_{t \geq 0}$ defined by

$$N(t) := \sum_{i \in \mathbb{N}} \mathbb{1}(t_i \leq t)$$

is a counting process, with $\mathbb{1}(A)$ denoting the indicator function of an event A.

Proof. We show that N fufils the properties of Definition 2.1.4.

1. Clear, as $\mathbb{1}(\cdot) \in \{0, 1\}$ and $N(t) = |\{i \in \mathbb{N} : t_i \in (0, t]\}| < \infty$, since $(0, t]$ is bounded.
2. As N is constant between two events, it suffices to consider the process at an event time t_i. Therefore, let $s \in (t_i, t_{i+1})$ be arbitrary. By construction for all $j \in \mathbb{N}$

   $$t_j \leq t_i \iff t_j \leq s\,.$$

   Therefore, it holds

   $$N(s) = \sum_{j \in \mathbb{N}} \mathbb{1}(t_j \leq s) = \sum_{j \in \mathbb{N}} \mathbb{1}(t_j \leq t_i) = N(t_i)$$

   which proves the right-continuity.
3. Let $s < t$. Then, $t_i \leq s$ implies $t_i \leq t$ and as $\mathbb{1}(\cdot) \in \{0, 1\}$ we get

   $$N(s) = \sum_{i \in \mathbb{N}} \mathbb{1}(t_i \leq s) \leq \sum_{i \in \mathbb{N}} \mathbb{1}(t_i \leq t) = N(t)\,.$$

[1] More in [45, Chapter 3].
[2] In particular, $N(t) < \infty$ for each $t \geq 0$.

4. By definition, $(t_i)_{i\in\mathbb{N}}$ fulfils
$$t_i < t_{i+1}\,,\ \ i \in \mathbb{N}\,.$$
Hence, at each time point just one event can occur. As $\mathbb{1}(\cdot) \in \{0,1\}$, this proves the claim.

□

For a TPP, $N(t)$ can be interpreted as the number of events having occurred up to time t. Moreover by right-continuity, $N(t) - N(s)$ gives the number of events in the interval $(s,t]$. In particular, the important relation

$$t_i \leq t \iff N(t) \geq i$$

holds, which we can use to derive a simple proof for the following result.

Theorem 2.1.6
Let $(N(t))_{t\geq 0}$ be a counting process. Define

$$t_i := \inf\{t > 0 | N(t) \geq i\}$$

for $i \in \mathbb{N}$. The resulting process $(t_i)_{i\in\mathbb{N}}$ is a TPP.

Proof. By right-continuity of the counting process N, the process $(t_i)_{i\in\mathbb{N}}$ is well-defined. Further, we prove the defining properties of a TPP according to Definition 2.1.1.

1. For each $i \in \mathbb{N}$, $t_i > 0$ and as N is non-decreasing $t_i \leq t_{i+1}$. Additionally as the differential $N(dt) \in \{0,1\}$, it has to hold that $t_i \neq t_{i+1}$. This shows that $(t_i)_{i\in\mathbb{N}}$ is strictly increasing.

2. For each $t \geq 0$, $N(t) < \infty$. Combining this with analogue arguments as are given in the second part of the proof of Theorem 2.1.3 and using that by construction $N(t) = |\{i \in \mathbb{N} : t_i \leq t\}|$, we obtain the property.

□

We briefly give the definition of two important properties of a counting process, which we later discuss in more detail, when dealing with explicit examples of TPPs.

Definition 2.1.7
Let $N = (N(t))_{t\geq 0}$ be a counting process.

1. *N is said to have independent increments, if for every $n \in \mathbb{N}$ and every sequence $0 \leq t_1 < t_2 < t_3 < ... < t_{n-1} < t_n$ the increments*
$$N(t_2) - N(t_1), N(t_3) - N(t_2),\ ...\ , N(t_n) - N(t_{n-1})$$
are independent.

2. *N is said to possess stationary increments, if the distribution of $N(s+t) - N(s)$ and $N(r+t) - N(r)$ is the same for any $r, s, t \geq 0$.*

For a counting process with independent increments, the number of events occurring in the disjoint time intervals

$$(t_1, t_2], (t_2, t_3],\ ...\ , (t_{n-1}, t_n]$$

are independent of each other. If the process has stationary increments, the distribution of the number of events occurring in some time interval only depends on the length of that interval and not on the exact time.

For notational convenience, we introduce the integral of a function f with respect to a counting process N by

$$\int_0^t f(s)\, N(ds) := \sum_{0<t_i<t} f(t_i)\,.$$

Last, we derive the representation of a TPP using its *counting measure*.

Definition 2.1.8
Let $\mathcal{B}(\mathbb{R}_{>0})$ denote the Borel-σ-algebra on $\mathbb{R}_{>0}$. A mapping $\mathcal{N} : \mathcal{B}(\mathbb{R}_{>0}) \to \mathbb{N}_0 \cup \{\infty\}$ is called counting measure, if

1. *for all pairwise disjoint $A_1, A_2, ... \in \mathcal{B}(\mathbb{R}_{>0})$ it holds that*

$$\mathcal{N}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathcal{N}(A_i)\,.$$

2. *$\mathcal{N}(A) < \infty$ for each bounded $A \in \mathcal{B}(\mathbb{R}_{>0})$.*

Theorem 2.1.9
Let $(t_i)_{i\in\mathbb{N}}$ be a TPP. The mapping $\mathcal{N}$, defined by

$$\mathcal{N}(A) := \sum_{i\in\mathbb{N}} \mathbb{1}(t_i \in A)$$

for $A \in \mathcal{B}(\mathbb{R}_{>0})$, is a counting measure.

Proof. By definition of $\mathcal{N}$, it is clear that $\mathcal{N}(A) \in \mathbb{N}_0 \cup \{\infty\}$. We check the remaining properties.

1. Let $A_1, A_2, ... \in \mathcal{B}(\mathbb{R}_{>0})$ be pairwise disjoint. Hence,

$$\mathbb{1}\left(t_i \in \bigcup_{j=1}^{\infty} A_j\right) = \sum_{j\in\mathbb{N}} \mathbb{1}(t_i \in A_j)\,.$$

   This shows

$$\begin{aligned}\mathcal{N}\left(\bigcup_{j=1}^{\infty} A_j\right) &= \sum_{i\in\mathbb{N}} \mathbb{1}\left(t_i \in \bigcup_{j=1}^{\infty} A_j\right) = \sum_{i\in\mathbb{N}}\sum_{j\in\mathbb{N}} \mathbb{1}(t_i \in A_j) \\ &= \sum_{j\in\mathbb{N}}\sum_{i\in\mathbb{N}} \mathbb{1}(t_i \in A_j) = \sum_{j\in\mathbb{N}} \mathcal{N}(A_j)\,.\end{aligned}$$

2. For each bounded $A \subset \mathbb{R}_{>0}$, we have

$$\mathcal{N}(A) = \sum_{i\in\mathbb{N}} \mathbb{1}(t_i \in A) = |\{i \in \mathbb{N} : t_i \in A\}| < \infty$$

   by the second defining property of a TPP.

□

In fact, the counting process and the counting measure are closely related via

$$\mathcal{N}((0,t]) = N(t)\,.$$

For a TPP on $\mathbb{R}_{>0}$, the counting process uniquely defines the corresponding counting measure. This is a consequence of the uniqueness theorem of measures.

After having introduced the main representations of TPPs, we want to address the topic of modelling and therefore derive the central concept of the *conditional intensity function*. The presented structure is based on [43, Chapter 2] and [9, Chapter 2]. If not stated otherwise, the proofs are extended versions of the proofs given there.

The basic idea for modelling TPPs is to use the evolutionary character of the time meaning that events can depend on past, but not on future events. This motivates a sequential modelling approach for TPPs by specifying the distribution of each event conditioned on the past of the process. Hence, the concept of the history $(\mathcal{H}_t)_{t\geq 0}$ introduced in Definition 2.1.1 plays a crucial role.

The first and most intuitive approach is to specify the *conditional probability density function / conditional cumulative distribution function* of the inter-event times of a TPP.

Definition 2.1.10 (Conditional pdf and cdf)
Consider a TPP $(t_i)_{i\in\mathbb{N}}$ according to Definition 2.1.1. For each $i \in \mathbb{N}$, let $F\big(\cdot\,|\mathcal{H}_{t_i}^+\big)$ be a continuous cdf and $f\big(\cdot\,|\mathcal{H}_{t_i}^+\big) := \frac{d}{dt}F\big(\cdot\,|\mathcal{H}_{t_i}^+\big)$ be the corresponding pdf on (t_i, ∞) [3].

— *The conditional probability density function (conditional pdf) is defined by*

$$f^*(t) := f(t|\mathcal{H}_t) = f\left(t|\mathcal{H}_{t_n}^+\right) ,$$

for t_n being the last event time before t. Roughly speaking, $f^*(t)\,dt$ gives the probability that the next event occurs in the infinitesimal time interval $[t, t+dt)$ conditioned on the history of the process $\mathcal{H}_t$.

— *The conditional cumulative distribution function (conditional cdf) is defined by*

$$F^*(t) := F(t|\mathcal{H}_t) = F\left(t|\mathcal{H}_{t_n}^+\right) = \int_{t_n}^{t} f^*(s)\,ds ,$$

for t_n being the last event time before t. Hence, $F^*(t)$ gives the probability that the next event occurs up to time t given the history $\mathcal{H}_t$.

— *The conditional survival function is defined by*

$$S^*(t) := 1 - F^*(t)$$

and gives the probability that the next event does not occur before time t given the history $\mathcal{H}_t$.

The $*$-notation highlights that these functions are conditioned on the history of the TPP [4].

By the Law of total probability, the joint distribution of a sequence of event times can be written in terms of the conditional pdfs

$$f(t_1, ..., t_n) = \prod_{i=1}^{n} f(t_i|t_1, ..., t_{i-1}) = \prod_{i=1}^{n} f(t_i|\mathcal{H}_{t_i}) = \prod_{i=1}^{n} f^*(t_i) .$$

Nevertheless, in most cases a different modelling approach is used, as the conditional pdf and cdf suffer from several disadvantages as is pointed out in [9, Chapter 2]. Firstly, f^* has to be constructed of proper density function, which therefore have to be non-negative and more importantly integrate to 1. Moreover, predefining a suitable form of

[3] The set of non-differentiable points of F is countable and therefore has Lebesque measure zero. Hence, f can take any values on this set, e.g. the corresponding right-derivatives of F.

[4] $*$-notation adopted from [8].

f^* is difficult in general. In particular, when dealing with events which excite or inhibit the occurrence of future events, it is not at all clear how these structural properties should be incorporated into the conditional pdf/cdf.

That's the reason, why the modelling approach based on the *conditional intensity function* is presented here and used throughout the thesis. The conditional intensity function is also known as *hazard function*.

Definition 2.1.11 (Conditional intensity function)
The conditional intensity function is defined by

$$\lambda^*(t) := \frac{f^*(t)}{1 - F^*(t)}\ ,\ \ t \geq 0$$

with conditional pdf f^* and conditional cdf F^* as introduced in Definition 2.1.10.

The conditional intensity function is given as the quotient of the conditional pdf and the conditional survival function. In particular, the conditional intensity function is an $\mathcal{H}$-adapted, non-negative stochastic process, as it depends on the random past of the process through the conditioning. Moreover by definition, the conditional intensity function is left-continuous at the event times, as it is defined in terms of the history of the process, which contains the events up to, but not including the actual time. Note that this is crucial for the likelihood calculations given in Section 2.3.

Before we continue, we want to build further understanding of the conditional intensity function. Therefore, the following theorem is useful.

Theorem 2.1.12
Consider a TPP with counting process N and let dt be an infinitesimal time length such that $N(dt) \in \{0, 1\}$. The conditional intensity function fulfils

$$\mathbb{E}[N(dt)|\mathcal{H}_t] = \lambda^*(t)\ dt\ ,$$

where $\mathbb{E}$ denotes the expectation.

Proof. We can use

$$\mathbb{P}(A|B \cap C) = \frac{\mathbb{P}(A \cap B|C)}{\mathbb{P}(B|C)}$$

and $N(dt) \in \{0, 1\}$ to obtain

$$\begin{aligned}
\mathbb{E}[N(dt)|\mathcal{H}_t] &= \mathbb{P}(N(dt) = 1|\mathcal{H}_t) \\
&= \mathbb{P}(\text{Next event in [t, t + dt)}|\text{Next event not before t}, \mathcal{H}_t) \\
&= \frac{\mathbb{P}(\text{Next event in [t, t +dt)}, \text{Next event not before t}|\mathcal{H}_t)}{\mathbb{P}(\text{Next event not before t}|\mathcal{H}_t)} \\
&= \frac{\mathbb{P}(\text{Next event in [t, t + dt)}|\mathcal{H}_t)}{\mathbb{P}(\text{Next event not before t}|\mathcal{H}_t)} \\
&= \frac{f^*(t)\ dt}{1 - F^*(t)} \\
&= \lambda^*(t)\ dt.
\end{aligned}$$

□

The conditional intensity function $\lambda^*(t)$ can hence be interpreted as an instantaneous mean rate of events conditional on the past. For example, if $\lambda^*(t) = 50$ we expect 50 events per time unit starting in the time point t.

Modelling a TPP by its conditional intensity function comes with several advantages compared to using the conditional pdf or cdf. Firstly, the conditional intensity function is not composed of density functions, which have to integrate to 1. Additionally as we will see, it is much easier to incorporate structural properties of TPPs like independence of the past or temporal clustering of events in the conditional intensity function than in the conditional pdf. Moreover, topics when dealing with real data including Maximum Likelihood Estimation, Goodness-of-Fit Tests and Simulation Algorithms can all be performed in terms of the conditional intensity function (see Section 2.3, Section 2.4, Section 2.5).

The crucial point is that the distribution of a TPP is uniquely specified by its conditional intensity function. In order to prove this result, we first show some auxiliary relations linking the conditional pdf and cdf with the conditional intensity function (cf. [43, Proposition 2.1]).

Theorem 2.1.13
Consider a TPP $(t_i)_{i\in\mathbb{N}}$ with conditional intensity function λ^*, conditional pdf f^* and conditional cdf F^*. For t_n being the last event before time t, we obtain:

$$f^*(t) = \lambda^*(t) \exp\left(-\int_{t_n}^{t} \lambda^*(s)\, ds\right)$$

$$F^*(t) = 1 - \exp\left(-\int_{t_n}^{t} \lambda^*(s)\, ds\right) .$$

Proof. By definition of the conditional intensity function we get

$$\lambda^*(t) = \frac{f^*(t)}{1 - F^*(t)} = \frac{\frac{d}{dt}F^*(t)}{1 - F^*(t)} = \frac{d}{dt}\left[-\log(1 - F^*(t))\right] .$$

As the TPP is simple, the Fundamental Theorem of Calculus yields

$$\int_{t_n}^{t} \lambda^*(s)\, ds = -\log(1 - F^*(t))$$

by integrating both sides of the above equation. Rearranging this leads to the stated expression for F^*. Using $\frac{d}{dt}F^*(t) = f^*(t)$ and again the Fundamental Theorem of Calculus shows

$$f^*(t) = \lambda^*(t) \exp\left(-\int_{t_n}^{t} \lambda^*(s)\, ds\right) .$$

□

Having these relations, we can prove the following central result, which theoretically justifies defining a TPP in terms of its conditional intensity function. In particular, Theorem 2.1.14 shows that two TPPs coincide if and only if their conditional intensity functions equal.

Theorem 2.1.14
A TPP $(t_i)_{i\in\mathbb{N}}$ is uniquely defined in distribution by its conditional intensity function λ^*, if for any event time t_n and any history $\mathcal{H}_{t_n}^+$

1. *the integral $\int_{t_n}^{t} \lambda^*(s)\, ds$ is well-defined and finite,*
2. $\int_{t_n}^{\infty} \lambda^*(s)\, ds = \infty$.

Proof. By Theorem 2.1.13, the conditional intensity function uniquely specifies the conditional cdf F^*. Hence, it suffices to prove that given the assumptions 1 & 2, the function $F^*(t), t > t_n$ is a valid cdf. We obtain:

— $F^*(t) \in [0,1]$ as $\exp\left(-\int_{t_n}^{t} \lambda^*(s)\, ds\right) \in [0,1]$.

— By assumption 1, F^* is continuous on (t_n, ∞).

— The term $\exp\left(-\int_{t_n}^{t} \lambda^*(s)\, ds\right)$ is non-increasing in t, as $\lambda^* \geq 0$ by definition. This implies that F^* is non-decreasing in t.

— $\lim_{t \searrow t_n} F^*(t) = 1 - \exp(0) = 0$

— $\lim_{t \to \infty} F^*(t) = 1 - \exp\left(-\int_{t_n}^{\infty} \lambda^*(s)\, ds\right) \overset{2.}{=} 1 - \exp(-\infty) = 1$

This proves that F^* is a valid cdf. □

The conditional intensity function can further be used to distinguish two important types of TPPs.

Definition 2.1.15 (Self-excitation / Self-inhibition)
A TPP with conditional intensity function λ^* is called

— *self-exciting, if the conditional intensity function increases whenever an event occurs.*

— *self-inhibiting, if the conditional intensity function decreases whenever an event occurs.*

Hence, for a self-exciting TPP the mean rate of events increases whenever an event occurs and thereby models temporal clusters. For self-inhibiting TPPs it is exactly the opposite, leading to a more regular pattern of event times.

For later purposes we also introduce the notion of the *compensator*, which plays a central role in the Maximum likelihood estimation (Section 2.3) and in the presented Goodness-of-Fit tests (Section 2.4).

Definition 2.1.16 (Compensator)
Let λ^* denote the conditional intensity function of a TPP. The compensator is given by

$$\Lambda^*(t) := \int_0^t \lambda^*(s)\, ds$$

for $t \geq 0$.

Before moving on to marked temporal point processes, we have a look at some important examples of TPPs.

2.1.1 Poisson Process

The Poisson process is the most simple and most important TPP, as the occurrence of an event does neither depend on the history of the process nor on the actual time. The process is used in a wide range of applications including astronomy, biology, seismology, finance and insurance.

Even for arbitrary TPPs, the Poisson process plays a crucial role in the Goodness-of-Fit Tests presented in Section 2.4, as an arbitrary TPP can be transformed into a Poisson process using a random time change. Furthermore, the Simulation algorithms for arbitrary TPPs given in Section 2.5 basically *thin out* a Poisson process. Therefore, we prove some additional results, which are useful in the later sections.

If not stated otherwise, the results and proofs given are based on [45, Chapter 2].

Definition 2.1.17 (Poisson Process)
A counting process $(N(t))_{t\geq 0}$ is called (homogenous) Poisson process with intensity $\lambda > 0$, if

1. $N(0) = 0$
2. *it has independent increments.*
3. *for arbitrary $s, t \geq 0$:*

$$\mathbb{P}(N(s+t) - N(s) = n) = e^{-\lambda t}\frac{(\lambda t)^n}{n!}\,, \quad n \in \mathbb{N}_0$$

meaning that the increment $N(s+t) - N(s)$ is Poisson distributed with intensity λt, i.e. $N(s+t) - N(s) \sim Poi(\lambda t)$.

By definition, the Poisson process has independent increments and as property 3 ensures that the distribution of an increment does not depend on the actual time s, it is clear that:

Corollary 2.1.18
A Poisson process has stationary increments.

Hence, future events are independent of past events (independent increments) and have the same distribution over time (stationary increments). In this sense, a Poisson process can be interpreted as a *memoryless* TPP.

In fact, there are several equivalent ways of defining a Poisson process. We derive some of them here, as each definition concentrates on another important property, which is going to be used in the later sections. First, we show that Definition 2.1.17 is equivalent to defining a Poisson process based on i.i.d. exponentially distributed inter-event times.

Theorem 2.1.19
Consider a Poisson process with intensity λ and inter-event times $(\tau_i)_{i\in\mathbb{N}}$. Then, the $(\tau_i)_{i\in\mathbb{N}}$ are i.i.d. Exp(λ)-distributed, i.e. possess the density

$$f(\tau) = \lambda e^{-\lambda\tau}$$

for $\tau > 0$.

Proof. We first deduce the distribution of τ_1:

$$\mathbb{P}(\tau_1 > t) = \mathbb{P}(N(t) = 0) = e^{-\lambda t}\frac{(\lambda t)^0}{0!} = e^{-\lambda t}$$

which shows $\tau_1 \sim \text{Exp}(\lambda)$. Next, consider the conditional distribution of τ_2 on τ_1:

$$\begin{aligned}
\mathbb{P}(\tau_2 > t|\tau_1 = s) &= \mathbb{P}(N(s+t) - N(s) = 0|N(s) = 1) \\
&= \mathbb{P}(N(s+t) - N(s) = 0) && \text{(independent increments)} \\
&= \mathbb{P}(N(t) = 0) && \text{(stationary increments)} \\
&= e^{-\lambda t}\,.
\end{aligned}$$

Hence, $\tau_2 \sim \text{Exp}(\lambda)$ and independent of τ_1. Repeating these arguments inductively leads the claim. □

In fact, this representation should not come as a surprise. As previously pointed out, the Poisson process can be interpreted as the TPP without memory. We give an intuitive proof of the result stating that the exponential distribution is the corresponding distribution without memory.

Definition 2.1.20 (Memoryless)
A random variable X with $\mathbb{P}(X > 0) = 1$ is called memoryless, if

$$\mathbb{P}(X > s + t | X > s) = \mathbb{P}(X > t)$$

for all $s, t > 0$.

Theorem 2.1.21
A continuous random variable X with $\mathbb{P}(X > 0) = 1$ is memoryless if and only if it is exponentially distributed.

Proof. We show the equivalence by providing both implications.
" $\Leftarrow$ " Let $X \sim \text{Exp}(\lambda)$ and $s, t > 0$:

$$\begin{aligned}\mathbb{P}(X > s + t | X > s) &= \frac{\mathbb{P}(X > s + t, X > s)}{\mathbb{P}(X > s)} = \frac{\mathbb{P}(X > s + t)}{\mathbb{P}(X > s)} \\ &= \frac{e^{-\lambda(s+t)}}{e^{-\lambda s}} = e^{-\lambda t} = \mathbb{P}(X > t).\end{aligned}$$

" $\Rightarrow$ " Let X be a continuous random variable with cdf F and survival function $S := 1 - F$. For arbitrary $s, t > 0$ the memorylessness implies

$$\frac{S(s + t)}{S(s)} = S(t) \, .$$

Considering the function $g := \log \circ S$ yields the relation

$$g(s + t) - g(s) = \log(S(s + t)) - \log(S(s)) = \log\left(\frac{S(s + t)}{S(s)}\right) = \log(S(t)) = g(t) \, .$$

This shows that g is a linear function. Hence, assume $g(t) = a + bt$ for $a, b \in \mathbb{R}$. As $S = \exp \circ g$, we can use the properties of the survival function to obtain necessary conditions on the parameters a and b:

1. $1 = S(0) = \exp(a) \iff a = 0$
2. $0 = \lim_{t\to\infty} S(t) = \lim_{t\to\infty} \exp(bt) \iff b < 0$

Defining $\lambda := -b > 0$ shows

$$S(t) = e^{-\lambda t}$$

and therefore $X \sim \text{Exp}(\lambda)$. □

Indeed, using the memorylessness property and some results on the moment generating function, we can prove that this gives an equivalent definition of the Poisson process.

Theorem 2.1.22
Let $(\tau_i)_{i\in\mathbb{N}} \overset{i.i.d.}{\sim} Exp(\lambda)$. Define

$$t_i = \sum_{j=1}^{i} \tau_j$$

for $i \in \mathbb{N}$. The resulting process $(t_i)_{i\in\mathbb{N}}$ defines event times of a Poisson process with intensity λ.

Proof. We prove the characteristics given in Definition 2.1.17.

1. As $\tau_i > 0$, it is clear that $N(0) = 0$.

2. The inter-event times $(\tau_i)_{i\in\mathbb{N}}$ are independent. Hence, future event times do not depend on past event times and so the process has independent increments.

3. Let $\tau \sim \text{Exp}(\lambda)$. For $t < \lambda$, the moment generating function of τ is given by

$$M_\tau(t) := \mathbb{E}[e^{t\tau}] = \int_0^\infty e^{t\tau}\lambda e^{-\lambda\tau}d\tau = \frac{\lambda}{t-\lambda}e^{(t-\lambda)\tau}\Big|_0^\infty = \frac{\lambda}{\lambda-t}.$$

As the τ_i are i.i.d., the moment generating function of t_n fulfils

$$M_{t_n}(t) = \mathbb{E}\left[e^{t\sum_{i=1}^n \tau_i}\right] = \mathbb{E}\left[\prod_{i=1}^n e^{t\tau_i}\right] = \prod_{i=1}^n \mathbb{E}\left[e^{t\tau_i}\right] = (M_\tau(t))^n = \left(\frac{\lambda}{\lambda-t}\right)^n.$$

In case of existence, the moment generating function uniquely determines the distribution of a random variable (cf. [45, Chapter 1.4]). Hence, t_n is Erlang distributed with parameters λ and n, i.e. it possesses the cdf

$$F_{t_n}(t) = \mathbb{P}(t_n \le t) = 1 - e^{-\lambda t}\sum_{i=0}^{n-1}\frac{(\lambda t)^i}{i!}\,,\quad t \ge 0\,.$$

Using this result, we can calculate the distribution of the counting process N being induced by the event times $(t_i)_{i\in\mathbb{N}}$ as given in Theorem 2.1.5:

$$\begin{aligned}\mathbb{P}(N(t)=n) &= \mathbb{P}(t_n \le t, t_{n+1} > t) = \mathbb{P}(\{t_n \le t\} \cap \{t_{n+1} \le t\}^c)\\ &= \mathbb{P}(t_n \le t) - \mathbb{P}(t_{n+1} \le t)\\ &= \left(1 - e^{-\lambda t}\sum_{i=0}^{n-1}\frac{(\lambda t)^i}{i!}\right) - \left(1 - e^{-\lambda t}\sum_{i=0}^{n}\frac{(\lambda t)^i}{i!}\right)\\ &= e^{-\lambda t}\frac{(\lambda t)^n}{n!}\,.\end{aligned}$$

Thus, $N(t) \sim \text{Poi}(\lambda t)$ and as the exponential distribution is memoryless, this implies that for arbitrary $s, t \ge 0$

$$N(s+t) - N(s) \sim \text{Poi}(\lambda t))\,.$$

□

In particular, this theoretically justifies that a Poisson process can easily be simulated by summing up i.i.d. exponentially distributed inter-event times (see Algorithm 1).

Moreover, we can use this representation to prove that the Poisson process is the unique TPP with constant conditional intensity function. Thus for a Poisson process, the conditional intensity function neither depends on the past of the process nor on the actual time. Note that the definitions given allow for a very simple proof.

Theorem 2.1.23
A Poisson process with intensity λ is the unique TPP with conditional intensity function $\lambda^*(t) = \lambda$.

Proof. The inter-event times $(\tau_i)_{i\in\mathbb{N}}$ of a Poisson process with intensity λ are i.i.d. $\text{Exp}(\lambda)$-distributed. By Definition 2.1.11, the conditional intensity function is given by

$$\lambda^*(t) = \frac{f^*(t)}{1-F^*(t)} = \frac{f(t|\mathcal{H}_t)}{1-F(t|\mathcal{H}_t)} = \frac{\lambda e^{-\lambda(t-t_n)}}{e^{-\lambda(t-t_n)}} = \lambda\,,$$

for t_n being the last event time before t. The uniqueness follows from Theorem 2.1.14 as the assumptions are trivially fulfilled for $\lambda > 0$. □

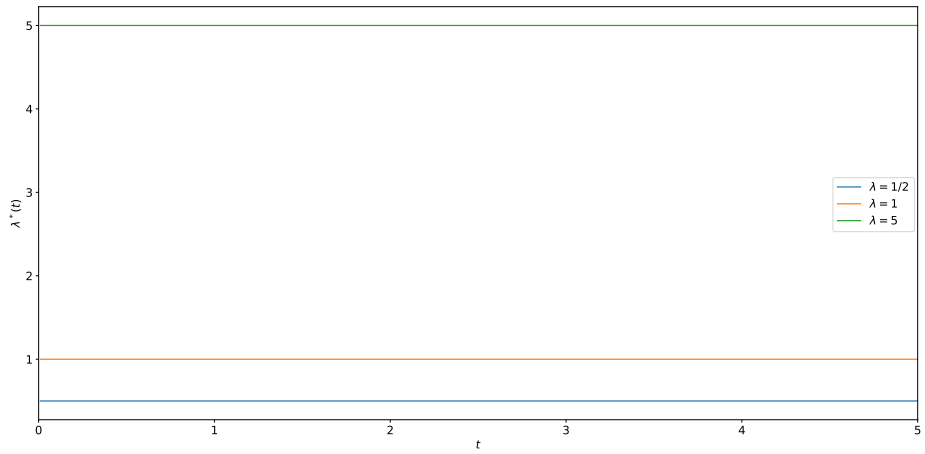


Figure 2.2: Conditional intensity function of a homogeneous Poisson process for different intensities λ.

Lastly, we want to prove a property of a Poisson process, which plays a crucial role in the Conditional-Uniformity Test presented in Section 2.4. Therefore, we first introduce the notion of the *order statistics*.

Definition 2.1.24 (Order statistics)
Let $X_1, ..., X_n$ be a sequence of random variables. $X_{(1)}, ..., X_{(n)}$ are called the order statistics of $X_1, ..., X_n$, if

$$X_{(1)} \leq X_{(2)} \leq ... \leq X_{(n)} .$$

Theorem 2.1.25
Let $X_1, ..., X_n$ be a sequence of i.i.d random variables with density f_X. The joint density of the order statistics $X_{(1)}, ..., X_{(n)}$ is given by

$$f_{X_{(1)},...,X_{(n)}}(x_1, ..., x_n) = n! \prod_{i=1}^{n} f_X(x_i)$$

for $x_1 < ... < x_n$.

Proof. Consider

$$S_n = \{\pi : \{1, ..., n\} \to \{1, ..., n\}\}$$

the set of permutations of the set $\{1, ..., n\}$. Note that $|S_n| = n!$. By definition of the order statistics it suffices to consider $a_1 < b_1 \leq a_2 < b_2 \leq ... \leq a_n < b_n$:

$$\begin{aligned}
\mathbb{P}\left(X_{(1)} \in (a_1, b_1], ..., X_{(n)} \in (a_n, b_n]\right) &= \mathbb{P}\left(\dot{\bigcup}_{\pi \in S_n} \{X_{\pi(1)} \in (a_1, b_1], ..., X_{\pi(n)} \in (a_n, b_n]\}\right) \\
&= \sum_{\pi \in S_n} \mathbb{P}\left(X_{\pi(1)} \in (a_1, b_1], ..., X_{\pi(n)} \in (a_n, b_n]\right) \\
&\overset{i.i.d.}{=} \sum_{\pi \in S_n} \prod_{i=1}^{n} \mathbb{P}(X_1 \in (a_i, b_i]) \\
&= n! \prod_{i=1}^{n} \int_{a_i}^{b_i} f_X(x)\, dx \\
&= \int_{\times_{i=1}^{n}(a_i, b_i]} n! \prod_{i=1}^{n} f_X(x_i)\, d(x_1, ..., x_n) .
\end{aligned}$$

Additionally,

$$\mathbb{P}(X_{(1)} \in (a_1, b_1], ..., X_{(n)} \in (a_n, b_n]) = \int_{\times_{i=1}^{n}(a_i,b_i]} f_{X_{(1)},...,X_{(n)}}(x_1, ..., x_n)\, d(x_1, ..., x_n)\,.$$

Hence, the two integrands have to coincide, which yields the claim. □

We can apply Theorem 2.1.25 to the Poisson process and obtain the following result. The proof is a modified version of [45, Theorem 2.3.1]

Theorem 2.1.26
Consider a Poisson process with intensity $\lambda > 0$. Given that $N(T) = n$, the event times $t_1, ..., t_n$ are distributed as the order statistics of n i.i.d. Unif$([0, T])$-distributed random variables.

Proof. By Theorem 2.1.25, the joint density of the order statistics of n i.i.d. Unif$([0, T])$-distributed random variables equals

$$f(s_1, ..., s_n) = \frac{n!}{T^n}$$

for $s_1 < ... < s_n < T$. Hence, the joint density of $t_1, ..., t_n$ conditioned on $N(T) = n$ is given by

$$\begin{aligned}
&f(t_1 = s_1, ..., t_n = s_n | N(T) = n) \\
&= \frac{f(t_1 = s_1, ..., t_n = s_n, N(T) = n)}{\mathbb{P}(N(T) = n)} \\
&\overset{(1)}{=} \frac{f(\tau_1 = s_1, \tau_2 = s_2 - s_1, ..., \tau_n = s_n - s_{n-1}, \tau_{n+1} > T - s_n)}{\mathbb{P}(N(T) = n)} \\
&\overset{(2)}{=} \frac{f(\tau_1 = s_1) \cdots f(\tau_n = s_n - s_{n-1}) \cdot f(\tau_{n+1} > T - s_n)}{\mathbb{P}(N(T) = n)} \\
&\overset{(3)}{=} \frac{\lambda e^{-\lambda s_1} \cdots \lambda e^{-\lambda(s_n - s_{n-1})} \cdot e^{-\lambda(T - s_n)}}{e^{-\lambda T} \frac{(\lambda T)^n}{n!}} \\
&= \frac{n!}{T^n}\,,
\end{aligned}$$

where we (1) re-wrote the events in terms of their inter-event times, (2) used the independence of the inter-event times and (3) the fact that they are exponentially distributed and $N(T) \sim \text{Poi}(\lambda T)$.

□

Given that we know that n events of a Poisson process have occurred up to time T, the event times are uniformly distributed on $[0, T]$. In particular, we can straightforward formulate a corollary, which will be used to verify one of the simulation algorithms presented in Section 2.5 (see Theorem 2.5.1).

Corollary 2.1.27
Consider a Poisson process N with intensity λ on the finite time interval $[0, T]$. The joint density that exactly n events occur at the times $0 < t_1 < ... < t_n \leq T$ is given by

$$f(t_1, ..., t_n, N(T) = n) = \lambda^n e^{-\lambda T}\,.$$

2.1.2 Non-homogeneous Poisson Process

Depending on the application it might be too simplistic to assume that the TPP neither depends on past events nor on the actual time. Therefore, the Poisson process can be extended to allow for a time-varying conditional intensity function. The resulting process is called the *non-homogeneous Poisson process*.

Definition 2.1.28 (Non-homogenous Poisson Process)
Let $\lambda : [0, \infty) \to (0, \infty)$ be a right-continuous and integrable function. A counting process $(N(t))_{t \geq 0}$ is called non-homogeneous Poisson process with conditional intensity function $\lambda^*(\cdot) = \lambda(\cdot)$, if

1. $N(0) = 0$
2. *the process has independent increments.*
3. *for arbitrary $s, t \geq 0$:*

$$N(s+t) - N(s) \sim Poi\left(\int_s^{s+t} \lambda(r)\, dr\right).$$

In contrast to the (homogeneous) Poisson process, the non-homogeneous Poisson Process does not necessarily have stationary increments, since the distribution of the increments can be time-dependent. The class of intensity functions being applied in practice is of a great variety including periodic, piece-wise constant or monotone functions. An example of the latter is the *Power law intensity function*.

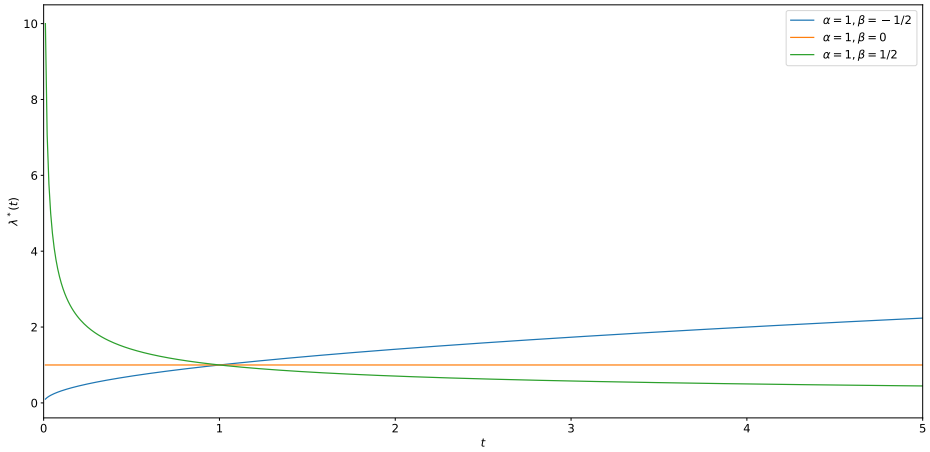


Figure 2.3: Power law intensity function for $\alpha = 1$ and different values of β.

Example 2.1.29 (Power law intensity function)
A non-homogeneous Poisson process with power law intensity function is given by

$$\lambda^*(t) = \alpha t^{-\beta}, \quad t \geq 0$$

for $\alpha > 0$ and $\beta < 1$. The intensity function is constant for $\beta = 0$, decreasing for $0 < \beta < 1$ and increasing for $\beta < 0$. A visualization is given in Figure 2.3.

2.1.3 Hawkes Process

In many real world applications including the modelling of infections, earthquakes or extreme returns in finance, it is not reasonable to assume that future events do not depend on past events, as it is done in the homogeneous and non-homogeneous Poisson process. A suitable class of TPPs, which is able to capture theses contagious types of events is the class of *Hawkes processes* proposed in [22, 23].

In this chapter, we present the univariate Hawkes process and strengthen the analysis with the marked Hawkes process in Section 2.2.2. The details given here are presented in [8, Chapter 7], [30] and [33].

Definition 2.1.30 (Univariate Hawkes Process)
The (univariate) Hawkes process is a TPP $(t_i)_{i\in\mathbb{N}}$ with counting process N given by the conditional intensity function

$$\lambda^*(t) := \mu + \int_0^t \phi(t-s)\, N(ds) = \mu + \sum_{i:t_i<t} \phi(t-t_i)\,, \quad t \geq 0\,.$$

The conditional intensity function is defined using the kernel function $\phi : (0,\infty) \to (0,\infty)$ given by

$$\phi(t) := \vartheta \cdot w(t)$$

and

- *the baseline intensity $\mu > 0$,*
- *the branching coefficient $\vartheta \in (0,1)$,*
- *the decay function $w : (0,\infty) \to (0,\infty)$ fulfilling*

$$\int_0^\infty w(t)\, dt = 1$$

and

$$\int_0^\infty t \cdot w(t)\, dt < \infty\,.$$

In contrast to the homogeneous and non-homogeneous Poisson process, the conditional intensity function of the Hawkes process depends on the actual time and the past of the process as it is modelled by the kernel function ϕ. Each event occurring instantaneously increases the conditional intensity function by the term $\phi(0+) = \vartheta \cdot w(0+) > 0$ and thereby makes future events more probable. Thus, the Hawkes process is a *self-exciting* process and is suitable to model the temporal clustering of events.

The effect of each event on the conditional intensity function strongly depends on the chosen decay function w. For rapidly decreasing w, each event mainly has a local effect, while this can endure longer for slower decaying or even humped functions w.

The normalizing conditions for the decay function w ensure that the kernel function ϕ is uniquely specified and that the resulting Hawkes process is well-defined. In particular, any density on the positive real line with finite expectation can be used as decay function. Moreover, the chosen parametrization has the advantage that the branching coefficient of the underlying branching process directly appears in the conditional intensity function [5].

Further, note the clear advantage of modelling a TPP using its conditional intensity function instead of using the conditional pdf of the inter-event times. We can easily incorporate self-exciting dynamics in the conditional intensity function whereas it would be much more difficult to adapt the conditional pdf correspondingly.

The most prominent Hawkes process uses an exponential decay function.

[5] More details at the end of this section.

Example 2.1.31 (Exponential Hawkes Process)
If the decay function is given by

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

for $\alpha > 0$, the effect of each event on the conditional intensity function decays exponentially with rate α over time. In this case, we call the corresponding Hawkes process exponentially decaying. Note that the exponential decay function obviously fulfils the normalizing conditions of Definition 2.1.30, as it is the density of the Exp(α)-distribution.

An illustration of the conditional intensity function of an exponentially decaying Hawkes process for different parameter values is given in Figure 2.4.

The instantaneous effect each event has on the conditional intensity function ($\phi(0+) = \vartheta\alpha$) is the same for the green and orange process and twice as big as for the blue one. The decay parameter α of the orange intensity function is twice as big as for the other two functions such that the effect of events decreases much faster. Moreover, varying the values of the branching coefficient ϑ illustrates that this parameter has a huge impact on the stability of the process.

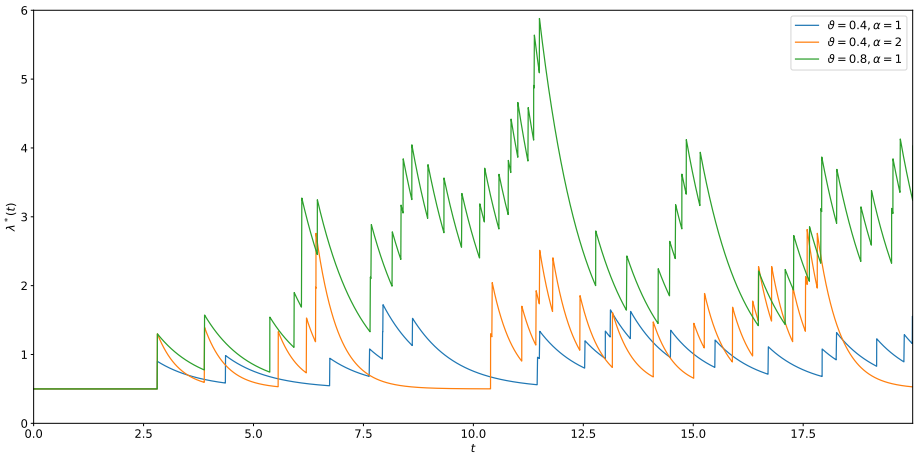


Figure 2.4: Conditional intensity function of an exponential Hawkes process with baseline intensity $\mu = 0.5$ and different values for the branching coefficient ϑ and the decay parameter α.

Motivated by this graphical assessment, we want to derive conditions on the branching coefficient ϑ, which ensure the corresponding Hawkes process to be stationary in the weak sense (cf. [23]). For this reason, consider a stationary Hawkes process with counting process N. Hence, there exists a mean intensity $\lambda > 0$ such that $\lambda = \mathbb{E}[\lambda^*(t)]$ for all $t > 0$. Applying this to Theorem 2.1.12 yields

$$\mathbb{E}[N(dt)] = \mathbb{E}[\mathbb{E}[N(dt)|\mathcal{H}_t]] = \mathbb{E}[\lambda^*(t)]\, dt = \lambda\, dt\,.$$

For the special case of the Hawkes process, this leads to

$$\begin{aligned}\lambda = \mathbb{E}[\lambda^*(t)] &= \mathbb{E}\left[\mu + \int_0^t \phi(t-s)\, N(ds)\right] \\ &= \mu + \int_0^t \phi(t-s)\, \mathbb{E}\left[N(ds)\right] = \mu + \lambda \int_0^t \phi(t-s)\, ds\end{aligned}$$

using Fubini's theorem. The specific form of the kernel function ϕ then gives the following representation of the mean intensity:

$$\lambda = \frac{\mu}{1 - \int_0^\infty \phi(r)\, dr} = \frac{\mu}{1 - \vartheta \int_0^\infty w(r)\, dr} = \frac{\mu}{1 - \vartheta}\,.$$

Hence, the branching coefficient must fulfil $\vartheta < 1$, which justifies Definition 2.1.30.

Another useful interpretation of the Hawkes process derives from the *Immigrant-Descendant representation* originally proposed in [24]. Each event of a Hawkes process can either be seen as an *immigrant*, i.e. an event of the homogeneous Poisson process with intensity μ (corresponding to the baseline intensity of the Hawkes process) or a *descendant* of such an immigrant event. An illustration is given in Figure 2.5. Each immigrant event (black square) produces either zero or more descendants. Direct descendants of immigrants are called *first generation descendants* (black circles); direct descendants of the first generation descendants are called *second generation descendants* (black diamonds) and so forth. Each immigrant and its descendants form together a Cluster. In this sense, the Hawkes process is often called a *Poisson Cluster process* or *Branching process*.

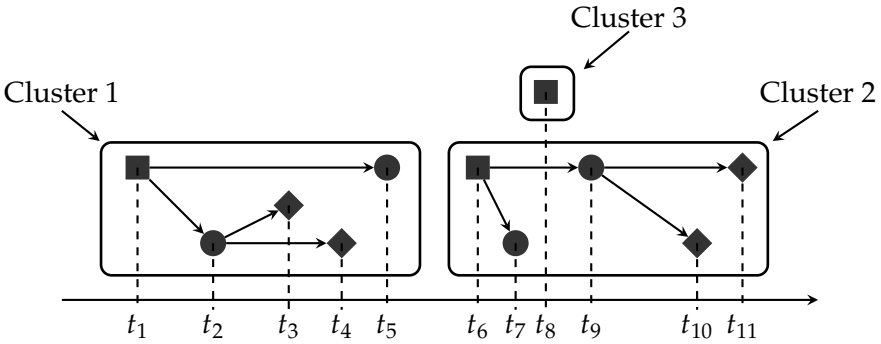


Figure 2.5: Immigrant-Descendant representation of a Hawkes process with immigrant clusters (based on [30, Figure 4]). Immigrant events are depicted with squares, first generation descendants with circles and second generation descendants with diamonds.

The Immigrant-Descendant representation can further be used to derive conditions on the Hawkes process to be well-defined and to provide additional understanding of the branching coefficient ϑ (cf. [24], [20, Section 5.4], [30]). For this reason consider an event cluster of an immigrant event and its associated descendants. For $n \in \mathbb{N}$, denote by

$$G_n := \text{Number of n-th generation descendants}$$

and define $G_0 := 1$, as each cluster contains exactly one immigrant event. An event occurring at time t_i produces direct descendants (i.e. of the next generation) according to a non-homogeneous Poisson process with intensity function $\phi(t - t_i)$, $t > t_i$. By Definition 2.1.28, $G_1 \sim \text{Poi}(\vartheta)$ as

$$\int_{t_i}^\infty \phi(t - t_i)\, dt = \int_0^\infty \phi(s)\, ds = \vartheta \underbrace{\int_0^\infty w(s)\, ds}_{=1} = \vartheta.$$

Hence, for $n \in \mathbb{N}$

$$\mathbb{E}[G_n] = \vartheta^n$$

using [20, Section 5.4, Lemma 2], as each event produces in the mean ϑ direct descendants. The expected cluster size is thus given by

$$\mathbb{E}\left[\sum_{n=0}^{\infty} G_n\right] = \sum_{n=0}^{\infty} \mathbb{E}[G_n] = \sum_{n=0}^{\infty} \vartheta^n = \begin{cases} \frac{1}{1-\vartheta}\,, & \vartheta < 1 \\ \infty\,, & \text{else.} \end{cases}$$

Hence for the clusters to have finite expected size and the Hawkes process to be well-defined, the branching coefficient has to fulfil $\vartheta < 1$. In this case, the branching coefficient has the additional interpretation of being the ratio of the expected number of descendants and the expected cluster size (i.e. including the immigrant) as

$$\frac{\mathbb{E}\left[\sum_{n=1}^{\infty} G_n\right]}{\mathbb{E}\left[\sum_{n=0}^{\infty} G_n\right]} = \frac{\sum_{n=1}^{\infty} \mathbb{E}[G_n]}{\sum_{n=0}^{\infty} \mathbb{E}[G_n]} = \frac{\sum_{n=1}^{\infty} \vartheta^n}{\sum_{n=0}^{\infty} \vartheta^n} = \frac{\frac{\vartheta}{1-\vartheta}}{\frac{1}{1-\vartheta}} = \vartheta\,.$$

Each event of a Hawkes process is therefore either a descendant with probability ϑ or an immigrant with probability $1-\vartheta$.

2.2 Marked Temporal Point Processes

In many real world applications, there is additional information available providing more details on each event. These information can either be discrete specifying different event types (e.g. Extreme negative and extreme positive stock returns), continuous (e.g. the actual value of the extreme return) or bivariate consisting of the discrete event type together with an additional continuous mark value (e.g. Extreme negative and extreme positive returns with the actual return value).

Therefore, we extend the concept of a TPP to capture such information. The resulting model is called *marked temporal point process*. We enhance the framework given in [8, Chapter 6.4, Chapter 7.3] and [43, Chapter 2.4] to nicely extend the setting of the TPP, formulate additional theorems and derive the proofs for all results in this section.

Definition 2.2.1 (Marked Temporal Point Process)
A stochastic process $(t_i, m_i)_{i\in\mathbb{N}}$ is called a marked temporal point process (MTPP) [6] *on $(0,\infty)\times\mathcal{M}$, if*

— *the ground process $(t_i)_{i\in\mathbb{N}}$ is a TPP,*

— *the mark process $(m_i)_{i\in\mathbb{N}}$ is a stochastic process with values in the mark space $\mathcal{M}\neq\emptyset$.*

We denote by

— *t_i the i-th event time,*

— *m_i the i-th event mark,*

— *$(\mathcal{H}_t)_{t\geq 0}$ given by $\mathcal{H}_t := \sigma((t_i,m_i)|t_i<t)$ the history of the MTPP,*

— *$(\mathcal{H}^g_t)_{t\geq 0}$ given by $\mathcal{H}^g_t := \sigma(t_i|t_i<t)$ the history of the ground process,*

— *$\mathcal{N}$ the counting measure given by* [7]*.*

$$\mathcal{N} : \mathcal{B}((0,\infty))\otimes\mathcal{M}^\sigma \to \mathbb{N}_0\cup\{\infty\}$$
$$A \mapsto \sum_{i\in\mathbb{N}} \mathbb{1}((t_i,m_i)\in A)$$

for a sigma-algebra $\mathcal{M}^\sigma$ on $\mathcal{M}$,

[6] In fact, an MTPP according to Definition 2.2.1 is simple, as we defined TPPs to be simple.
[7] $\otimes$ denotes the product sigma-algebra.

— $N_g := (N_g(t))_{t\geq 0}$ *defined by*

$$N_g(t) := \sum_{i\in\mathbb{N}} \mathbb{1}(t_i \leq t)$$

the counting process of the ground process.

An MTPP can therefore be interpreted as a TPP $(t_i)_{i\in\mathbb{N}}$ with additional mark information given by the mark process $(m_i)_{i\in\mathbb{N}}$. Depending on the mark space $\mathcal{M}$, we distinguish different types of MTPPs.

Definition 2.2.2 (Multivariate Temporal Point Process (with continuous marks))
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP. The process is called a

— *multivariate temporal point process (multivariate TPP), if the mark space is given by $\mathcal{M} = \{1, ..., M\}$ for some finite $M \in \mathbb{N}$. The counting process $N := (N(t))_{t\geq 0}$ is then given by*

$$N(t) := (N_1(t), ..., N_M(t)) ,$$

where the counting process of the m-th component process is defined by

$$N_m(t) := \mathcal{N}((0, t] \times \{m\}) , \quad m = 1, ..., M .$$

— *multivariate temporal point process with continuous marks (multivariate TPP with continuous marks), if the mark space is given by $\mathcal{M} = \{1, ..., M\} \times \mathcal{R}$ for some $M \in \mathbb{N}$, $\mathcal{R} \subset \mathbb{R}$ and the event marks take the form*

$$m_i = \left(m_i^{(T)}, m_i^{(V)}\right) , \quad i \in \mathbb{N} .$$

We call $m_i^{(T)}$ the mark type and $m_i^{(V)}$ the mark value associated to the event occurring at time t_i. The counting process of the m-th component process is correspondingly given by

$$N_m(t) := \mathcal{N}((0, t] \times \{m\} \times \mathcal{R}) .$$

Note that in both cases $N_g(t) = \sum_{m=1}^{M} N_m(t)$.

The marks of a multivariate TPP take values in a finite mark space and can hence be interpreted as the type of the events. Additionally, a multivariate TPP with continuous marks provides an extra information on the mark value. This is useful in many practical applications, as one often not only knows the type of an event, but also some more details specifying it.

As for TPPs, we model MTPPs by the conditional intensity function. This is straightforward as the evolutionary character of the time stays valid, since the ground process of an MTPP is a TPP.

Definition 2.2.3 (Conditional intensity function)
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP. Define the conditional intensity function by

$$\lambda^*(t, m) := \frac{f^*(t, m)}{1 - F^*(t)} = \frac{f(t, m|\mathcal{H}_t)}{1 - F(t|\mathcal{H}_t)}$$

for $t > 0$ and $m \in \mathcal{M}$. Again, the $*$-notation is used to highlight the dependence on the history of the process.

The conditional intensity function of an MTPP is therefore given by the fraction of the joint density of the time and mark component and the survival function. Note that the distributions are conditioned on the history $\mathcal{H}_t$ of the MTPP (and not on $\mathcal{H}_t^g$), which contains information on event times and event marks. Future event times and marks can therefore not only depend on past event times, but also on past event marks. The term density is used loosely to refer to the pdf in the continuous case and the probability mass function in the discrete case. For clarity, we give a heuristic interpretation of the conditional intensity function and provide the proof.

Theorem 2.2.4
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP on $(0,\infty) \times \mathcal{M}$ with conditional intensity function λ^*.

1. *If $\mathcal{M} = \{1, ..., M\}$ for some $M \in \mathbb{N}$:*

$$\lambda^*(t, m)\, dt = \mathbb{E}[N(dt \times m)|\mathcal{H}_t]$$

   with differential $N(dt \times m) := \mathcal{N}((t, t + dt] \times \{m\})$.

2. *If $\mathcal{M} = \mathcal{R}$ for some $\mathcal{R} \subset \mathbb{R}$:*

$$\lambda^*(t, m)\, dt\, dm = \mathbb{E}[N(dt \times dm)|\mathcal{H}_t]$$

   with differential $N(dt \times dm) := \mathcal{N}((t, t + dt] \times (m, m + dm])$.

Proof. Distinguish the two cases:

1. Analogue to Theorem 2.1.12 for fixed $m \in \mathcal{M}$.

2. For infinitesimal dt and dm, we have $N(dt \times dm) \in \{0, 1\}$. Hence,

$$\begin{aligned}
&\mathbb{E}[N(dt \times dm|\mathcal{H}_t)] \\
&= \mathbb{P}(N(dt \times dm) = 1|\mathcal{H}_t) \\
&= \mathbb{P}(\text{Next event in [t,t+dt), Next mark in [m,m+dm)}|\text{Next event not before t}, \mathcal{H}_t) \\
&= \frac{\mathbb{P}(\text{Next event in [t,t+dt), Next event not before t, Next mark in [m,m+dm)}|\mathcal{H}_t)}{\mathbb{P}(\text{Next event not before t}|\mathcal{H}_t)} \\
&= \frac{\mathbb{P}(\text{Next event in [t,t+dt), Next mark in [m,m+dm)}|\mathcal{H}_t)}{\mathbb{P}(\text{Next event not before t}|\mathcal{H}_t)} \\
&= \frac{f(t, m|\mathcal{H}_t)\, dt\, dm}{1 - F(t|\mathcal{H}_t)} \\
&= \lambda^*(t, m)\, dt\, dm
\end{aligned}$$

□

This shows that analogue to the case of TPPs, the conditional intensity function $\lambda^*(t, m)$ of an MTPP can be interpreted as an instantaneous, mean rate of events conditioned on the past of the process. We can easily deduce another useful representation of the conditional intensity function.

Theorem 2.2.5
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP. The conditional intensity function λ^* can be written as

$$\lambda^*(t, m) = \lambda_g^*(t) \cdot f^*(m|t)$$

where

— *λ_g^* denotes the conditional intensity function of the ground process $(t_i)_{i\in\mathbb{N}}$ given by*

$$\lambda_g^*(t) = \lambda_g(t|\mathcal{H}_t) = \frac{f(t|\mathcal{H}_t)}{1 - F(t|\mathcal{H}_t)} .$$

Note that λ_g^* is conditioned on $\mathcal{H}_t$ (and not on $\mathcal{H}_t^g$), which contains time and mark information.

— *$f^*(m|t)$ denotes the pdf of the conditional mark distribution given the event time t and the history $\mathcal{H}_t$, i.e.*

$$f^*(m|t) = f(m|t, \mathcal{H}_t) .$$

Proof. By Definition 2.2.3,

$$\begin{aligned}\lambda^*(t,m) &= \frac{f^*(t,m)}{1 - F^*(t)} = \frac{f(t,m|\mathcal{H}_t)}{1 - F(t|\mathcal{H}_t)} = \frac{f(t|\mathcal{H}_t) \cdot f(m|t,\mathcal{H}_t)}{1 - F(t|\mathcal{H}_t)} \\ &= \lambda_g(t|\mathcal{H}_t) \cdot f(m|t,\mathcal{H}_t) = \lambda_g^*(t) \cdot f^*(m|t) .\end{aligned}$$

□

The conditional intensity function of an MTPP factorizes into the conditional intensity function of the ground process, which in contrast to the case of a TPP also depends on mark information, and the conditional mark distribution given the event time. Note the similarity to the general relation of the joint, marginal and conditional distribution

$$f(x,y) = f(x) \cdot f(y|x) .$$

Using this representation, we can can prove the following relation:

Theorem 2.2.6
Consider an MTPP with conditional intensity function λ^* and conditional intensity function of the ground process λ_g^*. It holds

$$\lambda_g^*(t) = \int_{\mathcal{M}} \lambda^*(t,m)\, dl_{\mathcal{M}}(m) ,$$

where $l_{\mathcal{M}}$ denotes the reference measure of the conditional mark density $f^*(m|t)$ on $(\mathcal{M}, \mathcal{M}^\sigma)$ (typically, $l_{\mathcal{M}}$ is the Lebesgue measure in the continuous case and the counting measure in the discrete case).

Proof. Simply use the factorization of the conditional intensity function of an MTPP according to Theorem 2.2.5 and the fact that $f^*(m|t)$ is a density with respect to the reference measure $l_{\mathcal{M}}$:

$$\int_{\mathcal{M}} \lambda^*(t,m)\, dl_{\mathcal{M}}(m) = \int_{\mathcal{M}} \lambda_g^*(t) \cdot f^*(m|t)\, dl_{\mathcal{M}}(m) = \lambda_g^*(t) \cdot \underbrace{\int_{\mathcal{M}} f^*(m|t)\, dl_{\mathcal{M}}(m)}_{=1} = \lambda_g^*(t) .$$

□

This is reasonable as λ_g^* gives the intensity of the ground process and hence we can simply integrate $\lambda^*(t,m)$ over all possible states in the mark space $\mathcal{M}$ to obtain λ_g^*. As an example, we derive the form of the conditional intensity function of a multivariate TPP with continuous marks, as this will be needed to calculate the likelihood of the corresponding process in Section 2.3.

Example 2.2.7 (Cond. intensity function of a multivariate TPP with continuous marks) *Consider an M-variate TPP with continuous marks* $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i\in\mathbb{N}}$ *and mark space* $\mathcal{M} = \{1, ..., M\} \times \mathcal{R}$ *for* $\mathcal{R} \subset \mathbb{R}$. *The conditional intensity function is given by*

$$\begin{aligned}\lambda^*\left(t, m_t^{(T)}, m_t^{(V)}\right) &= \lambda_g^*(t) \cdot f^*\left(m_t^{(T)}, m_t^{(V)}|t\right) \\ &= \lambda_g^*(t) \cdot f^*\left(m_t^{(T)}|t\right) \cdot f^*\left(m_t^{(V)}|t, m_t^{(T)}\right) \\ &= \lambda^*\left(t, m_t^{(T)}\right) \cdot f^*\left(m_t^{(V)}|t, m_t^{(T)}\right) \\ &=: \lambda^*_{m_t^{(T)}}(t) \cdot f^*\left(m_t^{(V)}|t, m_t^{(T)}\right) ,\end{aligned}$$

where we define $\lambda^*_{m_t^{(T)}}(t) := \lambda^*\left(t, m_t^{(T)}\right)$ *for notational reasons. In particular as*

$$\lambda_g^*(t) \cdot f^*\left(m_t^{(T)}|t\right) = \lambda^*_{m_t^{(T)}}(t) ,$$

we can apply Theorem 2.2.6 to obtain the conditional intensity function of the ground process

$$\lambda_g^*(t) = \sum_{m=1}^{M} \lambda_m^*(t) .$$

In order to theoretically justify this modelling approach, we prove that under certain conditions an MTPP is uniquely specified by its conditional intensity function. Therefore, we first show that the conditional pdf and cdf of the inter-event times can be written in terms of the conditional intensity function of the ground process.

Theorem 2.2.8
Consider an MTPP $(t_i, m_i)_{i\in\mathbb{N}}$ *with conditional intensity function of the ground process* λ_g^*, *conditional pdf* f^* *and conditional cdf* F^*. *It holds true that*

$$f^*(t) = \lambda_g^*(t) \exp\left(-\int_{t_n}^{t} \lambda_g^*(s)\, ds\right)$$

$$F^*(t) = 1 - \exp\left(-\int_{t_n}^{t} \lambda_g^*(s)\, ds\right) ,$$

for t_n *being the time of the last event before* t.

Proof. By Theorem 2.2.5, $\lambda_g^*(t) = \frac{f^*(t)}{1-F^*(t)}$. Hence, applying Theorem 2.1.13 to λ_g^* yields the claim. □

Theorem 2.2.9
Consider an MTPP $(t_i, m_i)_{i\in\mathbb{N}}$ *with conditional intensity function* λ^* *and conditional intensity function of the ground process* λ_g^*. *If for any event time* t_n *and any history* $\mathcal{H}_{t_n}^+$

1. *the integral* $\int_{t_n}^{t} \lambda_g^*(s)\, ds$ *is well-defined and finite,*
2. $\int_{t_n}^{\infty} \lambda_g^*(s)\, ds = \infty$,

the MTPP is uniquely defined in distribution by λ^*.

Proof. We first prove that the conditional intensity function λ^* of an MTPP is uniquely specified by the conditional intensity function of the ground process λ_g^* and the conditonal mark distribution $f^*(m|t)$. Therefore assume that for each $t \geq 0$ and $m \in \mathcal{M}$

$$\lambda_{g,1}^*(t) \cdot f_1^*(m|t) = \lambda^*(t,m) = \lambda_{g,2}^*(t) \cdot f_2^*(m|t) \,.$$

Integrating both sides yields

$$\lambda_{g,1}^*(t) = \lambda_{g,1}^*(t) \underbrace{\int_{\mathcal{M}} f_1^*(m|t)\, dl_{\mathcal{M}}(m)}_{=1} = \lambda_{g,2}^*(t) \underbrace{\int_{\mathcal{M}} f_2^*(m|t)\, dl_{\mathcal{M}}(m)}_{=1} = \lambda_{g,2}^*(t) \,,$$

which shows the uniqueness of the representation, as $f^*(m|t)$ is a density with respect to $l_{\mathcal{M}}$.

What remains to prove is that under the assumptions 1&2 $f^*(t,m)$ is a valid density. By Theorem 2.2.8, the conditional pdf of the inter-event times $f^*(t)$ is specified by the conditional intensity function of the ground process λ_g^*. Given the assumptions, we can apply Theorem 2.1.14 to the ground process, which yields that $f^*(t)$ is indeed a valid density. As by definition $f^*(m|t)$ is a density, we obtain that

$$f^*(t,m) = f^*(t) \cdot f^*(m|t)$$

is properly defined. □

Based on the interaction between the time and the mark process, we distinguish different types of MTPPs and therefore introduce the notion of *independent* and *unpredictable marks*.

Definition 2.2.10 (Independent/Unpredictable marks)
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP with conditional intensity function

$$\lambda^*(t,m) = \lambda_g^*(t) \cdot f^*(m|t) \,.$$

The process is said to have

- *unpredictable marks, if $f^*(m|t) = f(m|t)$ for a density $f(\cdot|t)$ independent of the history $\mathcal{H}_t$,*
- *independent marks, if it has unpredictable marks and the conditional intensity function of the ground process λ_g^* is independent of the marks, i.e. $\mathcal{H}^g$-measurable.*

In both cases, the mark distribution only depends on the current time t and not on the past of the process. In contrast to an MTPP with independent marks, in an MTPP with unpredictable marks future event times may not only depend on past event times, but also on past event marks.

For an illustration of these concepts, we have a look at two central examples of MTPPs.

2.2.1 Marked Poisson Process

The most simple multivariate TPP is the multivariate Poisson process, which assumes that the component processes are independent Poisson processes.

Definition 2.2.11 (Multivariate Poisson process (with continuous marks))
Let $N = (N(t))_{t\geq 0}$ with $N(t) = (N_1(t), ..., N_M(t))$ be the counting process of an M-variate TPP. N is called multivariate Poisson process, if the component processes N_m are independent Poisson processes with intensity λ_m.

If additionally to the mark type $m^{(T)} \in \{1, ..., M\}$ the mark value $m^{(V)}$ is available and we assume that for each mark type $m^{(T)}$ the mark values $m^{(V)}$ are i.i.d., i.e.

$$m^{(V)} \mid m^{(T)} \overset{i.i.d.}{\sim} f_{\theta_{m^{(T)}}}$$

for a corresponding density $f_{\theta_{m^{(T)}}}$ being dependent on the mark type $m^{(T)}$, we call the resulting process a multivariate Poisson process with conditionally i.i.d. marks.

The multivariate Poisson process with conditionally i.i.d. marks is the most simple model to reasonably capture a multivariate TPP with continuous marks. In particular, this gives an example of a process with independent marks.

2.2.2 Marked Hawkes Process

The univariate Hawkes process as introduced in Section 2.1.3 can be extended to model MTPPs. To demonstrate the diversity achievable, we present Hawkes process models for MTPPs with $\mathcal{M} = \mathcal{R}$ for $\mathcal{R} \subset \mathbb{R}$, for multivariate TPPs and multivariate TPPs with continuous marks. We begin with the first.

Definition 2.2.12 (Univariate Hawkes Process with unpredictable/independent marks, cf. [8, Example 7.3(b)])
Let $(t_i, m_i)_{i \in \mathbb{N}}$ be an MTPP with mark space $\mathcal{M} = \mathcal{R}$ for $\mathcal{R} \subset \mathbb{R}$ and denote the counting process with N. Further, stick to the notation of the univariate Hawkes process in Definition 2.1.30 and assume for simplicity that $f^*(m|t) = f(m)$ for some density f on $\mathcal{R}$. A univariate Hawkes process with

— *unpredictable marks is given by the conditional intensity function of the ground process*

$$\lambda_g^*(t) = \left[\mu + \int_{(0,t)\times\mathcal{R}} \phi(t-s) \cdot \xi(m)\ N(ds \times dm) \right]$$

with impact function $\xi : \mathcal{R} \to (0, \infty)$ fulfilling

$$\int_{\mathcal{R}} f(m) \cdot \xi(m)\ dm = 1\ . \tag{2.2}$$

— *independent marks is given by the conditional intensity function of the ground process*

$$\lambda_g^*(t) = \left[\mu + \int_0^t \phi(t-s)\ N(ds) \right]\ .$$

In both cases, the ground intensity has the structure of a univariate Hawkes process and for simplicity the mark distribution is given by a density $f(m)$ independent of the history of the process and the current time t. In contrast to the Hawkes process with independent marks, the ground intensity of the Hawkes process with unpredictable marks depends further on the marks through the impact function ξ. Therefore, the resulting process has unpredictable, but not independent marks.

A famous example of a Hawkes process with unpredictable marks is the *ETAS (epidemic type after-shock sequence) model* used for earthquake modelling. The model usage is justified by the fact that earthquakes produce aftershocks (increased intensity) and earthquakes with high magnitude typically have a stronger effect (impact function). For more details on the ETAS model, see [39].

The normalizing condition given in Equation 2.2 ensures the resulting process to be well-defined and uniquely parametrized. In particular, it shows that the impact function is linked to the mark distribution and has to be chosen suitably.

Example 2.2.13 (Impact function)
Assume a univariate Hawkes process with unpredictable marks as given in Definition 2.2.12. Examples of suitably normalized impact functions are

— *the polynomial impact function:*

$$\xi(m) = \frac{\sum_{i=1}^{p} \epsilon_i m^i}{\sum_{i=1}^{p} \epsilon_i \mathbb{E}\left[m^i\right]}$$

for $\epsilon_i > 0$.

— *the exponential impact function:*

$$\xi(m) = \frac{\exp(\epsilon m)}{\mathbb{E}\left[\exp(\epsilon m)\right]}$$

for $\epsilon > 0$.

— *the void impact function:*

$$\xi(m) = 1.$$

For more details, see [33, Section 1.3].

Next, we introduce the multivariate Hawkes process, which was originally proposed in [23]. The main idea is to incorporate all possible kinds of self- and mutual excitement between the component processes into the conditional intensity function.

Definition 2.2.14 (Multivariate Hawkes Process)
An M-variate TPP $(t_i, m_i)_{i\in\mathbb{N}}$ *with counting process* $N = (N(t))_{t\geq 0} = (N_1(t), ..., N_M(t))_{t\geq 0}$ *is called M-variate Hawkes process, if the conditional intensity function of the m-th component process* N_m *is given by*

$$\begin{aligned}\lambda_m^*(t) :&= \mu_m + \sum_{k=1}^{M} \int_0^t \phi_{m,k}(t-s)\, N_k(ds) \\ &= \mu_m + \sum_{k=1}^{M} \sum_{\substack{i:t_i<t \\ m_i=k}} \phi_{m,k}(t-t_i)\end{aligned}$$

for $t \geq 0$. *The conditional intensity functions of the ground processes are defined using the kernel function* $\phi : (0,\infty) \to (0,\infty)^{M\times M}$ *given by*

$$\phi_{m,k}(t) := \vartheta_{m,k} \cdot w_{m,k}(t)$$

for each $m, k \in \{1, ..., M\}$ *and*

— *the baseline intensity* $\mu := (\mu_1, ..., \mu_M)^T \in (0,\infty)^M$,

— *the branching matrix* $\vartheta := (\vartheta_{m,k})_{m,k=1,...,M} \in (0,\infty)^{M\times M}$,

— *the decay function* $w : (0,\infty) \to (0,\infty)^{M\times M}$ *with*

$$\int_0^\infty w_{m,k}(t)\, dt = 1$$

and

$$\int_0^\infty t \cdot w_{m,k}(t)\, dt < \infty$$

for each $m, k \in \{1, ..., M\}$.

In vector notation, we can write the conditional intensity function $\lambda^*(t) = (\lambda_1^*(t), ..., \lambda_M^*(t))^T$ *as*

$$\lambda^*(t) = \mu + \int_0^t \phi(t-s)\, N(ds)\,.$$

As for the univariate Hawkes process, the conditional intensity function λ_m^* is given by a constant baseline intensity μ_m and a term capturing the effect of the history of the process. The main difference is that the interaction between two component processes or one component process itself is governed by a different kernel function $\phi_{m,k}$. In this way, all kinds and strengths of self - and mutual excitement can be captured.

Note that an event of type k instantaneously increases the conditional intensity function λ_m^* of events with type m by $\phi_{m,k}(0+) = \vartheta_{m,k} \cdot w_{m,k}(0+) > 0$. As this effect is always positive, the multivariate Hawkes process can just capture effects of excitement and not of inhibition.

Analogue to the univariate case, the normalizing conditions of the decay function ensure the resulting process to be uniquely parametrised. The most prominent example of a decay function is the *exponential decay function*.

Example 2.2.15 (Multivariate Hawkes Process with exponential decay)
An M-variate Hawkes process $(t_i, m_i)_{i\in\mathbb{N}}$ *is called exponentially decaying, if the decay function is given by*

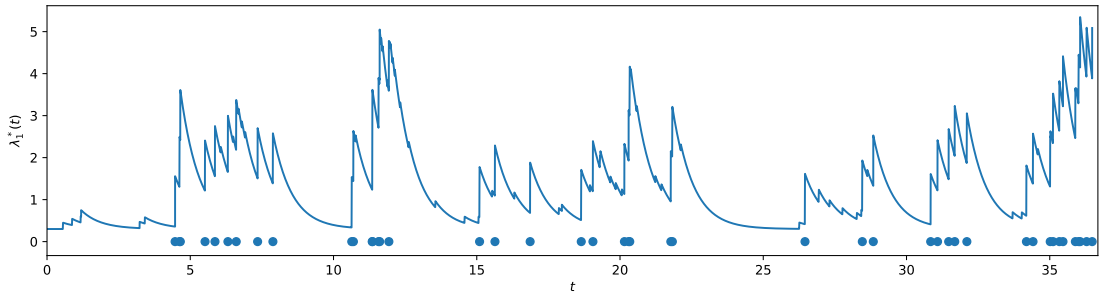
$$w_{m,k}(t) = \alpha_{m,k} e^{-\alpha_{m,k} t}$$

for $\alpha_{m,k} > 0$ *and* $m, k \in \{1, ..., M\}$.

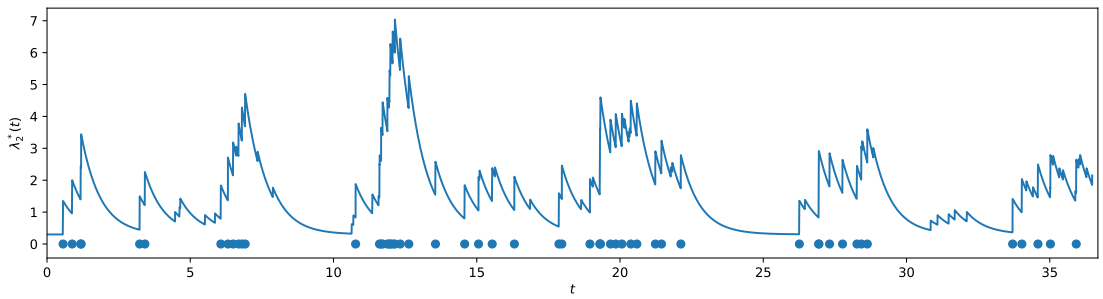
An example of a bivariate Hawkes process with exponential decay and parameters

$$\mu = \begin{bmatrix}\mu_1\\ \mu_2\end{bmatrix} = \begin{bmatrix}0.3\\ 0.3\end{bmatrix},\quad \vartheta = \begin{bmatrix}\vartheta_{1,1} & \vartheta_{1,2}\\ \vartheta_{2,1} & \vartheta_{2,2}\end{bmatrix} = \begin{bmatrix}0.8 & 0.1\\ 0.2 & 0.7\end{bmatrix},\quad \alpha = \begin{bmatrix}\alpha_{1,1} & \alpha_{1,2}\\ \alpha_{2,1} & \alpha_{2,2}\end{bmatrix} = \begin{bmatrix}1.5 & 1.5\\ 1.5 & 1.5\end{bmatrix}$$

is given in Figure 2.6.



(a) First component process



(b) Second component process

Figure 2.6: Conditional intensity functions of a bivariate Hawkes process. Events are denoted with dots.

The two component processes share the baseline intensity as well as the decay parameters. As can be seen, each event of type 1 effects the conditional intensity function of events of type 2 and the other way around. This is the case since $\vartheta_{1,2}, \vartheta_{2,1} > 0$. Note further that as $\vartheta_{1,1} > \vartheta_{1,2}$ and $\vartheta_{2,2} > \vartheta_{2,1}$, the effect of mutual excitement is much smaller than the effect of self-excitement. Hence, each event occurring has a bigger effect on events of the same type than of the other type. In particular, Figure 2.6 nicely illustrates that Hawkes processes are able to model temporal clusters of events and dependencies between events of different types.

Analogue to the univariate case, we are interested in deriving conditions, which ensure the underlying process to be stationary (cf. [4, 23]). Therefore, we first need a definition.

Definition 2.2.16 (Spectral radius)
Let A denote a matrix. The spectral radius $\rho(A)$ is defined by the maximum of the absolute values of its eigenvalues.

Assume the M-variate Hawkes process N to be stationary. Hence, there exists a mean intensity $\lambda \in (0, \infty)^M$ such that for any $t \geq 0$

$$\begin{aligned}\lambda = \mathbb{E}[\lambda^*(t)] &= \mathbb{E}\left[\mu + \int_0^t \phi(t-s)\, N(ds)\right] \\ &= \mu + \int_0^t \phi(t-s)\, \mathbb{E}[N(ds)] = \mu + \lambda \int_0^t \phi(t-s)\, ds\,.\end{aligned}$$

in vector notation. Using that the decay function integrates to one, we obtain

$$\int_0^t \phi(t-s)\, ds = \vartheta \underbrace{\int_0^t w(s)\, ds}_{=I_M} = \vartheta$$

where I_M denotes the $M \times M$ - unit matrix. Thus, the mean intensity fulfils $\lambda(I_M - \vartheta) = \mu$, which holds iff

$$\lambda = (I_M - \vartheta)^{-1}\mu$$

in the case of the matrix $I_M - \vartheta$ being invertible. By the Convergence of the Neumann Series (cf. [47, Theorem 1.5])), a necessary condition for the stationarity of the process is therefore given by

$$\rho(\vartheta) < 1\,.$$

In a similar manner, the *multivariate Hawkes process with continuous marks* can be defined. In this case additionally to the mark type, a continuous mark value is available. The effect of the mark value on the conditional intensity function is modelled by a suitable impact function. Moreover to fully specify a multivariate TPP with continuous marks, the conditional distribution of the mark values has to be specified. Therefore, the model presented here extends the approaches given in [13] and [8, Example 7.3 (c)]. The main idea is to assume that the conditional intensity functions as well as the conditional mark distribution are controlled by an underlying *hidden process*, which incorporates past event times and marks. In this manner, not only do future event times, but also future marks depend on the history of the process.

Definition 2.2.17 (Multivariate Hawkes Process with continuous marks)
Let $(t_i, m_i^{(T)}, m_i^{(V)})_{i\in\mathbb{N}}$ be a multivariate TPP with continuous marks given by the mark space

$\mathcal{M} = \{1, ..., M\} \times \mathcal{R}$ for $\mathcal{R} \subset \mathbb{R}$ and denote by $N = (N(t))_{t \geq 0}$ the corresponding counting process. Define the hidden process $h : (0, \infty) \to (0, \infty)^{M \times M}$ for $m, k \in \{1, ..., M\}$ by

$$\begin{aligned} h_{m,k}(t) : &= \int_{(0,t) \times \mathcal{R}} w_{m,k}(t - s) \cdot \xi_{m,k}\left(m^{(V)}\right) \; N_k\left(ds \times dm^{(V)}\right) \\ &= \sum_{\substack{i: t_i < t \\ m_i^{(T)} = k}} w_{m,k}(t - t_i) \cdot \xi_{m,k}\left(m_i^{(V)}\right) \; , \end{aligned}$$

where $\xi : \mathcal{R} \to (0, \infty)^{M \times M}$ denotes the impact function and assume that the decay function $w : (0, \infty) \to (0, \infty)^{M \times M}$ fulfils the assumptions given in Definition 2.2.14.Further, define the conditional intensity function of the m-th component process by

$$\lambda_m^*(t) := \mu_m + \sum_{k=1}^{M} \vartheta_{m,k} \cdot h_{m,k}(t)$$

using the baseline intensity $\mu := (\mu_1, ..., \mu_M)^T \in (0, \infty)^M$ and the branching matrix $\vartheta := (\vartheta_{m,k})_{m,k=1,...,M} \in (0, \infty)^{M \times M}$.

Suppose further that the conditional distribution of the mark value $m_t^{(V)}$ of an event of type $m_t^{(T)} = m$ occurring at time t is given by

$$f^*\left(m_t^{(V)} | t, m_t^{(T)} = m\right) = f_{\beta_m(t)}\left(m_t^{(V)}\right)$$

for a density $f_{\beta_m(t)}$ with distributional parameter $\beta_m(t)$, The distributional parameter is defined using $a \in \mathbb{R}^M$ and $b := (b_{m,k})_{m,k=1,...,M} \in \mathbb{R}^{M \times M}$ by

$$\beta_m(t) := a_m + \sum_{k=1}^{M} b_{m,k} \cdot h_{m,k}(t)$$

fulfilling

$$\int_{\mathcal{R}} f_{\beta_k(t)}\left(m^{(V)}\right) \cdot \xi_{m,k}\left(m^{(V)}\right) \; dm^{(V)} = 1 \tag{2.3}$$

for $m, k \in \{1, ..., M\}$. In this case, the resulting process is called a Multivariate Hawkes process with continuous marks.

Compared to the multivariate Hawkes process, the conditional intensity functions of the component processes of the multivariate Hawkes process with continuous marks just differ by the impact function, which models the effect of the mark value. The hidden process h can be seen as a summary statistic of the past of the process and is solemnly introduced to capture that both, the future event times and future event marks, depend on past event times and marks. Using this notation, the conditional intensity function as well as the conditional mark value distribution can be defined by a linear transformation of the hidden process.

2.3 Maximum Likelihood Estimation

When applying a TPP or MTPP to model real data, the process' parameters have to be inferred from the data. We can apply Maximum Likelihood Estimation, as the likelihood function is expressible in terms of the conditional intensity function. This is another clear benefit of the modelling approach chosen. The proof of the following result for TPPs is an extended version of the one given in [43, Proposition 3.1].

Theorem 2.3.1 ((Log) Likelihood of TPPs, cf. [8, Proposition 7.2.III.])
Let $(t_i)_{i\in\mathbb{N}}$ be a TPP with counting process N and conditional intensity function λ^*. For a finite $T > 0$, let $t_1, ..., t_{N(T)}$ denote the realizations of the TPP on $[0, T]$. The likelihood is given by

$$L = \left[\prod_{i=1}^{N(T)} \lambda^*(t_i)\right] \exp\left(-\int_0^T \lambda^*(s)\, ds\right)$$

and hence the log-likelihood takes the form

$$\log L = \sum_{i=1}^{N(T)} \log(\lambda^*(t_i)) - \int_0^T \lambda^*(s)\, ds\,.$$

Proof. Consider the time interval $[0, t_{N(T)}]$. The likelihood is given by the joint density of the event times:

$$\begin{aligned}
L &= f\left(t_1, ..., t_{N(T)}\right) \\
&= \prod_{i=1}^{N(T)} f(t_i | \underbrace{t_1, ..., t_{i-1}}_{=\mathcal{H}_{t_i}}) && \text{(Chain rule)} \\
&= \prod_{i=1}^{N(T)} f^*(t_i) \\
&= \prod_{i=1}^{N(T)} \lambda^*(t_i) \cdot \exp\left(-\int_{t_{i-1}}^{t_i} \lambda^*(s)\, ds\right) && \text{(Theorem 2.1.13)} \\
&= \left[\prod_{i=1}^{N(T)} \lambda^*(t_i)\right] \exp\left(-\sum_{i=1}^{N(T)} \int_{t_{i-1}}^{t_i} \lambda^*(s)\, ds\right) \\
&= \left[\prod_{i=1}^{N(T)} \lambda^*(t_i)\right] \exp\left(-\int_0^{t_{N(T)}} \lambda^*(s)\, ds\right) && (t_0 := 0)
\end{aligned}$$

If $[0, t_{N(T)}] \subsetneq [0, T]$, we have to correct for no event occurring in $(t_{N(T)}, T]$. As this probability is given by $1 - F^*(T)$, we simply have to adjust the likelihood by this term:

$$\begin{aligned}
L &= \left[\prod_{i=1}^{N(T)} f^*(t_i)\right] \cdot (1 - F^*(T)) \\
&= \left[\prod_{i=1}^{N(T)} \lambda^*(t_i)\right] \exp\left(-\int_0^{t_{N(T)}} \lambda^*(s)\, ds\right) \cdot \exp\left(-\int_{t_{N(T)}}^{T} \lambda^*(s)\, ds\right) && \text{(Theorem 2.1.13)} \\
&= \left[\prod_{i=1}^{N(T)} \lambda^*(t_i)\right] \exp\left(-\int_0^{T} \lambda^*(s)\, ds\right)
\end{aligned}$$

Applying the log yields the required form of the log-likelihood. □

Analogue, we obtain the likelihood for MTPPs and derive a proof, which nicely extends the idea used for the TPPs.

Theorem 2.3.2 ((Log) Likelihood of MTPPs, cf. [8, Proposition 7.3 III])
Let $(t_i, m_i)_{i\in\mathbb{N}}$ be an MTPP with counting process of the ground process N_g and conditional intensity function λ^*. For a finite $T > 0$, let $(t_1, m_1), ..., (t_{N_g(T)}, m_{N_g(T)})$ denote the realizations

of the MTPP on $[0, T]$. *The likelihood is given by*

$$\begin{aligned} L &= \left[\prod_{i=1}^{N_g(T)} \lambda^*(t_i, m_i)\right] \exp\left(-\int_0^T \int_{\mathcal{M}} \lambda^*(s, m)\, dl_{\mathcal{M}}(m)\, ds\right) \\ &= \left[\prod_{i=1}^{N_g(T)} \lambda_g^*(t_i)\right] \left[\prod_{i=1}^{N_g(T)} f^*(m_i|t_i)\right] \exp\left(-\int_0^T \lambda_g^*(s)\, ds\right) \end{aligned}$$

for the reference measure $l_{\mathcal{M}}$ *on* $(\mathcal{M}, \mathcal{M}^\sigma)$. *In particular, the log-likelihood has the form*

$$\begin{aligned} \log L &= \sum_{i=1}^{N_g(T)} \log(\lambda^*(t_i, m_i)) - \int_0^T \int_{\mathcal{M}} \lambda^*(s, m)\, dl_{\mathcal{M}}(m)\, ds \\ &= \left[\sum_{i=1}^{N_g(T)} \log\left(\lambda_g^*(t_i)\right)\right] + \left[\sum_{i=1}^{N_g(T)} \log(f^*(m_i|t_i))\right] - \int_0^T \lambda_g^*(s)\, ds\,. \end{aligned}$$

Proof. Consider the likelihood on $[0, t_{N_g(T)}]$.

$$\begin{aligned} L &= f\left((t_1, m_1), ..., (t_{N_g(T)}, m_{N_g(T)})\right) \\ &= \prod_{i=1}^{N_g(T)} f(t_i, m_i | \underbrace{(t_1, m_1), ..., (t_{i-1}, m_{i-1})}_{=\mathcal{H}_{t_i}}) && \text{(Chain rule)} \\ &= \prod_{i=1}^{N_g(T)} f^*(t_i, m_i) \\ &= \prod_{i=1}^{N_g(T)} f^*(t_i) \cdot f^*(m_i|t_i) \\ &= \prod_{i=1}^{N_g(T)} \lambda_g^*(t_i) \cdot \exp\left(-\int_{t_{i-1}}^{t_i} \lambda_g^*(s)\, ds\right) \cdot f^*(m_i|t_i) && \text{(Theorem 2.2.8)} \\ &= \left[\prod_{i=1}^{N_g(T)} \lambda_g^*(t_i) \cdot f^*(m_i|t_i)\right] \exp\left(-\sum_{i=1}^{N_g(T)} \int_{t_{i-1}}^{t_i} \lambda_g^*(s)\, ds\right) \\ &= \left[\prod_{i=1}^{N_g(T)} \lambda^*(t_i, m_i)\right] \exp\left(-\int_0^{t_{N_g(T)}} \lambda_g^*(s)\, ds\right) && \text{(Theorem 2.2.5)} \\ &= \left[\prod_{i=1}^{N_g(T)} \lambda^*(t_i, m_i)\right] \exp\left(-\int_0^{t_{N_g(T)}} \int_{\mathcal{M}} \lambda^*(s, m)\, dl_{\mathcal{M}}(m)\, ds\right) && \text{(Theorem 2.2.6)} \end{aligned}$$

As for the TPP, if $[0, t_{N(T)}] \subsetneq [0, T]$, we have to correct for no event occurring in $(t_{N(T))}, T]$. By Theorem 2.2.8, the probability of this event is given by

$$1 - F^*(T) = \exp\left(-\int_{t_{N_g(T)}}^T \lambda_g^*(s)\, ds\right)\,.$$

Hence, the likelihood has the stated form

$$L = \left[\prod_{i=1}^{N_g(T)} f^*(t_i, m_i)\right] (1 - F^*(T))\,.$$

Applying the log yields the form of the log-likelihood. □

We can use Theorem 2.3.2 to obtain the likelihood for some important examples of MTPPs, which we have discussed in the previous sections.

Corollary 2.3.3
Consider the setting of Theorem 2.3.2 in which the MTPP is a

1. *Multivariate TPP ($\mathcal{M} = \{1, ..., M\}$): The log-likelihood is given by*

$$\begin{aligned}\log L &= \sum_{i=1}^{N_g(T))} \log(\lambda^*(t_i, m_i)) - \sum_{m=1}^{M} \Lambda_m^*(T) \\ &= \sum_{i=1}^{N_g(T))} \log\left(\lambda_{m_i}^*(t_i)\right) - \sum_{m=1}^{M} \Lambda_m^*(T)\end{aligned}$$

with $\lambda_m^*(t) := \lambda^*(t, m)$ for $m \in \{1, ..., M\}$ and compensator

$$\Lambda_m^*(T) := \int_0^T \lambda_m^*(s)\, ds\,.$$

2. *Multivariate TPP with continuous marks ($\mathcal{M} = \{1, ..., M\} \times \mathcal{R}$): The log-likelihood is given by*

$$\begin{aligned}\log L &= \sum_{i=1}^{N_g(T))} \log\left(\lambda^*\left(t_i, m_i^{(T)}\right)\right) + \sum_{i=1}^{N_g(T)} \log\left(f^*\left(m_i^{(V)} | t_i, m_i^{(T)}\right)\right) - \sum_{m=1}^{M} \Lambda_m^*(T) \\ &= \sum_{i=1}^{N_g(T))} \log\left(\lambda_{m_i^{(T)}}^*(t_i)\right) + \sum_{i=1}^{N_g(T)} \log\left(f^*\left(m_i^{(V)} | t_i, m_i^{(T)}\right)\right) - \sum_{m=1}^{M} \Lambda_m^*(T)\end{aligned}$$

with $\lambda_{m_i^{(T)}}^*(t_i) := \lambda^*\left(t_i, m_i^{(T)}\right)$ for $m_i^{(T)} \in \{1, ..., M\}$ and the compensator as in 1.

Although an explicit expression for the log-likelihood exists, in most of the cases this can not be solved analytically to obtain the maximum likelihood estimates. We want to look at one example, in which we can derive an analytic solution for the MLE.

Example 2.3.4 (MLE of multivariate Poisson Process with conditionally i.i.d. marks)
Let $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i \in \mathbb{N}}$ be an M-variate Poisson process with conditionally i.i.d. marks as defined in Definition 2.2.11. By Corollary 2.3.3, the log-likelihood is given by

$$\begin{aligned}\log L &= \sum_{i=1}^{N_g(T))} \log\left(\lambda^*\left(t_i, m_i^{(T)}\right)\right) + \sum_{i=1}^{N_g(T)} \log\left(f^*\left(m_i^{(V)} | t_i, m_i^{(T)}\right)\right) - \sum_{m=1}^{M} \Lambda_m^*(T) \\ &= \sum_{m=1}^{M} N_m(T) \cdot \log(\lambda_m) + \sum_{i=1}^{N_g(T)} \log\left(f_{\theta_{m_i^{(T)}}}\left(m_i^{(V)}\right)\right) - \left(\sum_{m=1}^{M} \lambda_m\right) T\,,\end{aligned}$$

where N_m denotes the counting process of events of type m. To calculate the maximum likelihood estimates, we derive the partial derivatives of the log-likelihood with respect to the model parameters [8]:

$$\frac{\partial \log L}{\partial \lambda_m} = \frac{N_m(T)}{\lambda_m} - T \stackrel{!}{=} 0 \implies \hat{\lambda}_m = \frac{N_m(T)}{T}$$

$$\frac{\partial \log L}{\partial \theta_m} = \frac{\partial}{\partial \theta_m} \sum_{i=1}^{N_g(T)} \log\left(f_{\theta_{m_i^{(T)}}}\left(m_i^{(V)}\right)\right) \stackrel{!}{=} 0 \implies \hat{\theta}_m$$

In particular, $\hat{\theta}_m$ is the classic maximum likelihood estimate of the distribution f_{θ_m}.

In nearly all the other cases numerical methods have to be applied to find the model parameters which maximize the log-likelihood.

[8] The second order derivatives ensure that the estimates indeed maximize the log-likelihood.

2.4 Goodness-of-Fit Tests

Having estimated the parameters of a TPP or MTPP, we need to evaluate the goodness-of-fit of the model. Hence in this section, we derive several appropriate methods, which are a mixture of graphical and statistical tests and are mainly based on [39], [8, Chapter 7.4] and [33, Chapter 1]. For better understanding, we present a theoretical justification for each test.

Test 1 (Log-likelihood)
Compare the log-likelihood of the different models. The higher the log-likelihood, the better the model describes the data.

Besides this measure, all the methods presented in the following are based on the so-called *Random Time Change Theorem*, which states that testing for an arbitrary TPP can be reduced to testing for a homogeneous Poisson process. Due to the structural properties of the Poisson process derived in Section 2.1.1, testing for a Poisson process is much simpler and hence reduces the complexity of the problem at hand enormously.

Therefore, we first introduce the concept of the *residual process*. The Goodness-of-Fit Tests based on the residual process are often called *point process residual analysis* as proposed in [39]. The compensator Λ^* introduced in Definition 2.1.16 as

$$\Lambda^*(t) = \int_0^t \lambda^*(s)\, ds$$

plays an essential role.

Definition 2.4.1 (Residual process / Random Time Change, cf. [33, Definition 1.19])
Assume the conditional intensity function fulfils $\lambda^* > 0$ and the compensator $\Lambda^*(t) \overset{t \to \infty}{\longrightarrow} \infty$ (in the case of multivariate TPPs accordingly for the component processes).

— *Let $(t_i)_{i\in\mathbb{N}}$ be a TPP with compensator Λ^*. The residual process $(\tilde{t}_i)_{i\in\mathbb{N}}$ is defined by*

$$\tilde{t}_i := \Lambda^*(t_i)\ .$$

— *Let $(t_i, m_i)_{i\in\mathbb{N}}$ denote an M-variate TPP and Λ^*_m the compensator of the m-th component process. The M-variate residual process $(\tilde{t}_i, m_i)_{i\in\mathbb{N}}$ is given by*

$$\tilde{t}_i := \Lambda^*_{m_i}(t_i)\ .$$

— *Let $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i\in\mathbb{N}}$ be an M-variate TPP with continuous marks. For each mark type $m \in \{1, ..., M\}$ let Λ^*_m denote the corresponding compensator and $F^*\left(m^{(V)}|t, m^{(T)}\right)$ the conditional distribution function of the mark value $m^{(V)}$. Then, the M-variate residual process with continuous marks $\left(\tilde{t}_i, m_i^{(T)}, \tilde{m}_i^{(V)}\right)_{i\in\mathbb{N}}$ is given by*

$$\begin{aligned}\tilde{t}_i &:= \Lambda^*_{m_i^{(T)}}(t_i)\\ \tilde{m}_i^{(V)} &:= F^*\left(m_i^{(V)}|t_i, m_i^{(T)}\right)\ .\end{aligned}$$

The idea of the residual process is to transform the original sequence of event times by applying the compensator of the underlying model. This transformation is called *Random Time Change*. In the case of multivariate TPPs with continuous marks, the mark values

are additionally transformed using the conditional distribution function given the event time and the mark type.

For the residual process of a TPP or MTPP, we derive the proofs of the central *Random Time Change Theorem*, which can be seen as an extension of the following famous result.

Theorem 2.4.2
Let $X \sim F$ for some continuous and strictly increasing distribution function F. The transformed random variable $F(X)$ is Unif$([0,1])$ - distributed.

Proof. As F is continuous and strictly increasing, it is invertible. Hence for any $u \in [0,1]$ we obtain

$$\mathbb{P}(F(X) \leq u) = \mathbb{P}\left(X \leq F^{-1}(u)\right) = F\left(F^{-1}(u)\right) = u\ .$$

□

Theorem 2.4.3 (Random Time Change Theorem for TPPs)
Let $(t_i)_{i\in\mathbb{N}}$ be a TPP with residual process $(\tilde{t}_i)_{i\in\mathbb{N}}$. The residual process is a unit-rate Poisson process.

Proof. Let λ^* be the conditional intensity function and Λ^* the compensator of the TPP $(t_i)_{i\in\mathbb{N}}$. Further, define the inter-event times of the residual process $(\tilde{\tau}_i)_{i\in\mathbb{N}}$ by

$$\tilde{\tau}_i := \tilde{t}_i - \tilde{t}_{i-1}$$

and $\tilde{t}_0 := 0$. We prove $(\tilde{\tau}_i)_{i\in\mathbb{N}} \overset{i.i.d.}{\sim} \text{Exp}(1)$. Therefore, consider an arbitrary $n \in \mathbb{N}$. First, note that by Theorem 2.1.13

$$F^*(t_{n+1}) = 1 - \exp\left(-\int_{t_n}^{t_{n+1}} \lambda^*(s)\, ds\right) = 1 - \exp\left(-[\Lambda^*(t_{n+1}) - \Lambda^*(t_n)]\right)\ .$$

Using this relation, we obtain the distribution function

$$\begin{aligned} F_{\tilde{\tau}_{n+1}}(\tau|\mathcal{H}_{t_{n+1}}) &= \mathbb{P}(\tilde{\tau}_{n+1} \leq \tau|\mathcal{H}_{t_{n+1}}) = \mathbb{P}\left(\tilde{t}_{n+1} - \tilde{t}_n \leq \tau|\mathcal{H}_{t_{n+1}}\right) \\ &= \mathbb{P}\left(\Lambda^*(t_{n+1}) - \Lambda^*(t_n) \leq \tau|\mathcal{H}_{t_{n+1}}\right) \\ &= \mathbb{P}\left(F^*(t_{n+1}) \leq 1 - \exp(-\tau)|\mathcal{H}_{t_{n+1}}\right) \\ &= 1 - \exp(-\tau)\ , \end{aligned}$$

where we applied Theorem 2.4.2 in the last equality. This yields the claim. □

Theorem 2.4.4 (Random Time Change Theorem for multivariate TPPs, cf. [33, Theorem 1.20])
It is valid:

1. *Consider the M-variate TPP $(t_i, m_i)_{i\in\mathbb{N}}$. The corresponding residual process $(\tilde{t}_i, m_i)_{i\in\mathbb{N}}$ defines an M-variate Poisson process with unit rates and independent components.*
2. *Consider the M-variate TPP with continuous marks $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i\in\mathbb{N}}$ with M-variate residual process with continuous marks $\left(\tilde{t}_i, m_i^{(T)}, \tilde{m}_i^{(V)}\right)_{i\in\mathbb{N}}$. The ground processes of the residual process define an M-variate Poisson process with unit rates and independent components. Further, the mark values of the residual process are i.i.d. Unif$([0,1])$-distributed random variables.*

Proof. Apply Theorem 2.4.3 to the component processes to obtain 1 and additionally Theorem 2.4.2 to the conditional mark value distribution for 2. □

Essentially, the Random Time Change Theorem shows that if a TPP is transformed using its true conditional intensity function, the resulting residual process defines a Poisson process with unit rate. If in contrast the transformation assumes a wrong conditional intensity function, the residual process will exhibit some systematic deviations from the Poisson process. Hence, testing for a possibly complex conditional intensity function can be boiled down to testing for a unit rate Poisson process after the Random Time Change, which is a much simpler and well-studied problem.

When working with real data, we are hence in the following setting. Let $t_1, ..., t_n$ denote the realizations of a TPP (analogue in the multivariate case for the component processes) on the finite time interval $[0, T]$. Further, assume that a conditional intensity function λ^* has been fitted to the data. Use this conditional intensity function to calculate the residual process $\tilde{t}_1, ..., \tilde{t}_n$ as well as the inter-event times of the residual process $\tilde{\tau}_1, ..., \tilde{\tau}_n$ given by

$$\tilde{\tau}_i := \tilde{t}_i - \tilde{t}_{i-1}$$

with $\tilde{t}_0 := 0$.

Based on the theoretical properties of the Poisson process derived in Section 2.1.1, we construct several goodness-of-fit tests each of them being followed by the respective theoretical foundation.

Test 2 (QQ Plot - Exp(1) distribution)
Let F denote the distribution function of the $Exp(1)$-distribution. Consider the theoretical quantiles given by

$$q_{\frac{i}{n}} := F^{-1}\left(\frac{i}{n}\right)$$

for $i = 1, ..., n-1$, where the index i only runs until $n-1$ as $F^{-1}(1) = \infty$.

Further, denote by $\tilde{\tau}_{(1)}, ..., \tilde{\tau}_{(n)}$ the order statistics of the inter-event times of the residual process. Plot the points

$$\left(q_{\frac{i}{n}}, \tilde{\tau}_{(i)}\right)$$

for $i = 1, ..., n-1$ and a regression line through these points. For the true model, the points should roughly lie on the regression line.

Theoretical Foundation
Theorem 2.1.19 shows that the inter-event times of a Poisson process with unit rate are i.i.d. $Exp(1)$-distributed. Furthermore, the famous Theorem of Glivenko-Cantelli proves the uniform convergence of the empirical cdf to the corresponding theoretical cdf. For completeness, we state the Theorem here.

Theorem 2.4.5 (Glivenko-Cantelli Theorem, cf. [48, Theorem 9.7])
Let $(X_i)_{i\in\mathbb{N}}$ be a sequence of i.i.d. random variables with distribution function F. The empirical distribution function defined for each $n \in \mathbb{N}$ by

$$\hat{F}_n(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(X_i \leq x)\,, \quad x \in \mathbb{R}$$

fulfils

$$\sup_{x\in\mathbb{R}} |\hat{F}_n(x) - F(x)| \overset{n\to\infty}{\longrightarrow} 0$$

$\mathbb{P}$-almost surely.

In addition to using a graphical method for checking the Exponential distribution of the inter-event times of the residual process, we also apply a statistical test.

Test 3 (Kolmogorov-Smirnov Test Exp(1)-distribution)
First, choose a significance level α. Perform the Kolmogorov- Smirnov Test on $\tilde{\tau}_1, ..., \tilde{\tau}_n$ against the theoretical distribution $F_0 = Exp(1)$. Reject the null hypothesis that $\tilde{\tau}_1, ..., \tilde{\tau}_n \sim F_0$, if the p-value is smaller than α.

Theoretical Foundation
In general, let $X_1, ..., X_n \overset{i.i.d.}{\sim} F$ with continuous distribution function F. Consider the Test

$$H_0 : F = F_0 \text{ against } H_1 : F \neq F_0$$

for some continuous distribution function F_0. As test statistic the Kolmogorov-Smirnov Test (KS-Test) uses

$$D_n = \sup_{x \in \mathbb{R}} |\hat{F}_n(x) - F_0(x)| ,$$

i.e. the maximal difference between the empirical and hypothetical distribution function, which converges uniformly to zero by the Glivenko-Cantelli Theorem.

It can be shown that under the null hypothesis, a rescaled version of the test statistic converges in distribution to the so-called Kolmogorov distribution irrespective of the true underlying distribution function F_0.

Theorem 2.4.6 (Kolmogorov distribution, cf. [44, Theorem 6.4.4, Remark 6.4.5])
For $(X_i)_{i \in \mathbb{N}} \overset{i.i.d.}{\sim} F_0$ with continuous F_0 it holds

$$\sqrt{n} D_n \xrightarrow{\mathcal{L}} F_K ,$$

where F_K denotes the distribution function of the Kolmogorov distribution given by

$$F_K(x) = 1 - 2 \sum_{k=1}^{\infty} (-1)^{k-1} e^{-2k^2 x^2} .$$

Hence, the KS-Test defines an approximate level α test, if the null hypothesis gets rejected whenever

$$\sqrt{n} D_n > K_{1-\alpha} ,$$

where $K_{1-\alpha}$ denotes the $(1-\alpha)$-quantile of the Kolmogorov distirbution or equivalently if the p-value is smaller than the significance level.

In particular, the KS-Test can be applied in our setting, as the exponential distribution is continuous and its rate parameter one is fix and does not have to be estimated from the data.

Test 4 (Conditional Uniformity Test)
First, choose a significance level α. Next, plot the empirical distribution function of $\frac{\tilde{t}_1}{\Lambda^*(T)}, ..., \frac{\tilde{t}_n}{\Lambda^*(T)}$, i.e. the step function going through the points

$$\left(\frac{\tilde{t}_i}{\Lambda^*(T)}, \frac{i}{n} \right)$$

for $i = 1, ..., n$. Moreover, plot the confidence band

$$y = x \pm \frac{K_{1-\alpha}}{\sqrt{n}}$$

on $x, y \in [0, 1]$ where $K_{1-\alpha}$ denotes the $(1-\alpha)$ - quantile of the Kolmogorov distribution (see Theorem 2.4.6). An approximate level α test is given, if we reject the Poisson process assumption given that the empirical distribution function crosses the confidence band at any point.

Theoretical Foundation
The Conditional Uniformity Test is basically a graphical KS-Test. Its theoretical justification is given in Theorem 2.1.26 which shows that conditioned on the number of events n on $[0, \Lambda^*(T)]$, *the event times* $\tilde{t}_1, ..., \tilde{t}_n$ *of a Poisson process are distributed as the order statistics of n i.i.d. Unif*$([0, \Lambda^*(T)])$ *distributed random variables. Hence, the normalized event times* $\frac{\tilde{t}_1}{\Lambda^*(T)}, ..., \frac{\tilde{t}_n}{\Lambda^*(T)}$ *are distributed as the order statistics of n i.i.d. Unif*$([0,1])$*- distributed random variables. As for the Uniform distribution on* $[0,1]$ *the distribution function equals the identity function*

$$F_0(x) = x \quad \text{for } x \in [0,1] \,,$$

an approximate level α KS-Test is given if the null hypothesis is rejected in the case of

$$\begin{aligned} & \sqrt{n} D_n > K_{1-\alpha} \\ \iff & D_n > \frac{K_{1-\alpha}}{\sqrt{n}} \\ \iff & \sup_{x \in [0,1]} |\hat{F}_n(x) - F_0(x)| > \frac{K_{1-\alpha}}{\sqrt{n}} \\ \iff & \sup_{x \in [0,1]} |\hat{F}_n(x) - x| > \frac{K_{1-\alpha}}{\sqrt{n}} \end{aligned}$$

where $\hat{F}_n$ *denotes the empirical distribution function of the* $\frac{\tilde{t}_i}{\Lambda^*(T)}$. *This proves the proposed form of the confidence band.*

Next, we present a method to check if the inter-event times of the residual process are independent.

Test 5 (Autocorrelation function)
Plot the sample autocorrelation function of $\tilde{\tau}_1, ..., \tilde{\tau}_n$. *For the true model, there should be, except for lag 0, no significant deviation from zero.*

Theoretical Foundation
By Theorem 2.1.19, the inter-event times of a Poisson process are independent. Note that no autocorrelation does not imply independence, but the existence of significant autocorrelation disproves independence.

More tests focusing on other properties of the Poisson process can be derived in a similar manner (see e.g. [31]).

In the case of a multivariate TPP with continuous marks, we additionally want to test the Goodness-of-Fit of the modelled mark distribution. Recall that we have proven in Theorem 2.4.4 that the mark values of the residual process are i.i.d. Unif$([0,1])$-distributed. Therefore, we additionally consider the following two tests.

Mark Test 1 (QQ Plot - Unif$([0,1])$ distribution)
Analogue to Test 2, construct for each mark type $m^{(T)} \in \{1, ..., M\}$ *a QQ-Plot of the mark values of the residual process* $\tilde{m}^{(V)}$ *against the Unif*$([0,1])$ *distribution.*

Mark Test 2 (Kolmogorov-Smirnov Test - Unif$([0,1])$ distribution)
Analogue to Test 3, perform for each mark type $m^{(T)} \in \{1, ..., M\}$ *a KS-Test on the mark values of the residual process* $\tilde{m}^{(V)}$ *against the Unif*$([0,1])$*-distribution.*

2.5 Simulation

There exist many reasons, why one should be interested in algorithms for simulating point processes (cf. [43, Chapter 4.3]). A consistent method for simulation can be used

— as a constructive proof that the point process is well-defined.

— as a tool for Goodness-of-Fit testing. Assume that a point process with conditional intensity function λ^* has been fitted to some data. Simulating the point process allows to compare real and simulated event sequences. Hence, one can get a practical intuition of whether the model is suitable for the corresponding data or not, depending on whether the simulated data shows the same patterns as the original data.

— to provide valuable understanding of the process and the effect of different parameters on the model.

— as a generative method to make predictions about the future.

— to estimate characteristics of the process for which no closed formula exist.

We first describe a simple method to simulate a homogeneous Poisson process. Using this, we can formulate the famous *thinning algorithm* for simulating arbitrary TPPs given by their conditional intensity function. We conclude this chapter with an extension to the case of MTPPs and in particular multivariate TPPs with continuous marks.

2.5.1 Simulation of Poisson Processes

Simulating a homogeneous Poisson process is simple as by Theorem 2.1.22 the event times are the sums of successive i.i.d. $\text{Exp}(\lambda)$-distributed inter-event times. An implementation is given in Algorithm 1.

Algorithm 1: Homogeneous Poisson Process

Input: Intensity λ, finite time interval $[0, T]$
Output: Set of event times $\{t_1, ..., t_n\}$

$t := 0$
$i := 1$
while *True* **do**
  Generate $\tau \sim \text{Exp}(\lambda)$.
  $t = t + \tau$
  if $t \leq T$ **then**
    $t_i := t$
    $i = i + 1$
  else
    break
  end
end

2.5.2 Simulation of Temporal Point Processes

For a general TPP specified by its conditional intensity function λ^*, in particular non-homogeneous Poisson processes and Hawkes processes, there are two main methods for simulation:

— *Inverse Method*

— *Ogata's modified thinning algorithm* proposed in [38]

The presented theory is based on [8, Chapter 7.5] and [43, Chapter 4].

The Inverse Method uses that a unit-rate Poisson process can be transformed into a TPP with conditional intensity function λ^* by applying the inverse of the compensator $(\Lambda^*)^{-1}$ to its event times. Basically, this is just the opposite of the Random Time Change Theorem described in Theorem 2.4.3. A detailed proof is given in [43, Proposition 4.1].

The main problem with this approach is that the inverse of the compensator has to be calculated. In most of the cases, no explicit solution of this inverse exists such that a numerical approximation has to be used.

This is the reason why the Inverse Method is typically not applied in practise. Therefore, we do not discuss this approach further and instead concentrate on Ogata's modified thinning algorithm, which does not require this inversion.

The main idea of Ogata's modified thinning algorithm is to simulate a homogeneous Poisson process with an intensity, which is an upper bound of the conditional intensity function of the TPP to be simulated and thin out the process of event times using an acceptance-rejection approach. The algorithm is a modification of *Lewis' thinning algorithm* (cf. [32]), which was originally proposed as a method to simulate non-homogeneous Poisson processes. Whereas the algorithm of Lewis' requires a global bound for the conditional intensity function regardless of time and history, Ogata's algorithm just needs local bounds, which are valid in between two events. This makes the approach applicable for a wider class of TPPs. An implementation is given in Algorithm 2.

Algorithm 2: Ogata's modified thinning algorithm (cf. [8, Algorithm 7.5.IV], [43, Algorithm 4.2])

Input: Conditional intensity function λ^*, finite time interval $[0, T]$
Output: Set of event times $\{t_1, ..., t_n\}$

$t := 0$
$i := 1$
while $t \leq T$ **do**
  Compute constants $l(t)$ and $\bar{\lambda}(t)$ such that $\bar{\lambda}(t) \geq \sup_{s \in [t, t+l(t)]} \lambda^*(s)$.
  Generate $\tau \sim \text{Exp}\left(\bar{\lambda}(t)\right)$.
  Generate $U \sim \text{Unif}([0, 1])$.
  if $\tau > l(t)$ **then**
    $t = t + l(t)$
  else if $t + \tau > T$ **or** $U > \frac{\lambda^*(t+\tau)}{\bar{\lambda}(t)}$ **then**
    $t = t + \tau$
  else
    $t = t + \tau$
    $t_i := t$
    $i = i + 1$
  end
end

Assume that the last event time of a TPP is given by t_{i-1}. How does the algorithm simulate the time of the next event t_i?

The idea is to find a local upper bound $\bar{\lambda}(t_{i-1})$ for the conditional intensity function λ^* being valid on the time interval $[t_{i-1}, t_{i-1} + l(t_{i-1})]$ for a suitable distance $l(t_{i-1})$, i.e.

$$\bar{\lambda}(t_{i-1}) \geq \sup_{s \in [t_{i-1}, t_{i-1}+l(t_{i-1}))} \lambda^*(s) \,.$$

According to this bound, a new candidate inter-event time $\tau \sim \text{Exp}\left(\bar{\lambda}(t_{i-1})\right)$ is sampled, as it is done for the homogeneous Poisson process. Assuming that $t := t_{i-1} + \tau \leq T$ and $\tau \leq l(t_{i-1})$, the value t is accepted as the next event time t_i with probability

$$\mathbb{P}\left(U \leq \frac{\lambda^*(t)}{\bar{\lambda}(t_{i-1})}\right) = \frac{\lambda^*(t)}{\bar{\lambda}(t_{i-1})}$$

as $U \sim \text{Unif}([0,1])$. The procedure then starts again from the beginning with t taking the place of t_{i-1}. An illustration is given in Figure 2.7.

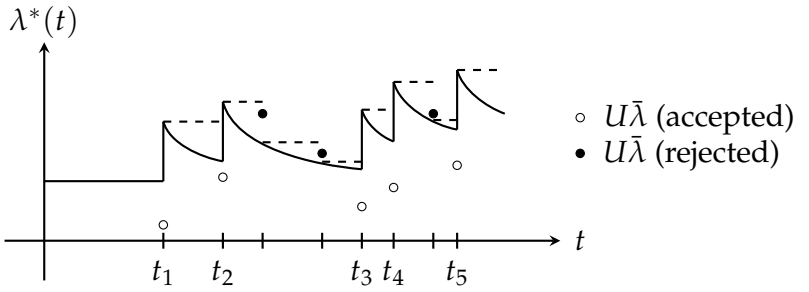


Figure 2.7: Illustration of Ogata's modified thinning algorithm for a univariate Hawkes process (similar in [5, Figure 3]). The upper bounds $\bar{\lambda}$ are given by dashed lines. Accepted (rejected) event times are indicated by white (black) circles. Times get accepted as event times whenever $U\bar{\lambda} \leq \lambda^*$.

What remains to prove is that Algorithm 2 indeed produces realizations of a TPP with conditional intensity function λ^*. The corresponding proof is split in two parts. In Theorem 2.5.1, we show the correctness of the algorithm for non-homogeneous Poisson processes, which we then extend to arbitrary TPPs in Theorem 2.5.2. The reason therefore is that in between two events the conditional intensity function is deterministic and hence simulating an arbitrary TPP is equivalent to iteratively simulating event by event of different non-homogeneous Poisson processes.

Theorem 2.5.1 (Cf. [5, Theorem 4.2])
Let $\bar{t}_1, ..., \bar{t}_{\bar{N}(T)}$ denote the event times of a homogeneous Poisson process $\bar{N}$ with intensity $\bar{\lambda}$ on the finite time interval $[0, T]$. Further, let N be a non-homogeneous Poisson process with intensity function λ such that

$$\bar{\lambda} \geq \sup_{t \in [0,T]} \lambda(t) \,.$$

The event sequence given by accepting each $\bar{t}_i$ with probability $\frac{\lambda(\bar{t}_i)}{\bar{\lambda}}$ is a realization of the non-homogeneous Poisson process N.

Proof. First note that the event sequence obtained by this acceptance-rejection approach has independent increments, as $\bar{N}$ has independent increments and the time points get accepted independently of each other. Further, as the conditional intensity function is positive and

$$\bar{\lambda} \geq \sup_{t \in [0,T]} \lambda(t) \,,$$

a valid probability is given by $\frac{\lambda(t)}{\bar{\lambda}}$ for each time point $t \in [0, T]$.

Moreover, let $a, b \in [0, T]$ such that $a < b$ and denote by $\bar{\mathcal{N}}$ and $\mathcal{N}$ the counting measure of the homogeneous Poisson process and of the obtained event sequence, respectively. We can apply Corollary 2.1.27 and obtain the following probability

$$\begin{aligned}
\mathbb{P}\left(\mathcal{N}((a,b]) = 0, \bar{\mathcal{N}}((a,b]) = n\right) &= \int_a^b \int_{t_1}^b \cdots \int_{t_{n-1}}^b \bar{\lambda}^n e^{-\bar{\lambda}(b-a)} \prod_{i=1}^n \left(1 - \frac{\lambda(t_i)}{\bar{\lambda}}\right) dt_n \cdots dt_2 \, dt_1 \\
&= \frac{e^{-\bar{\lambda}(b-a)}}{n!} \int_a^b \cdots \int_a^b \prod_{i=1}^n (\bar{\lambda} - \lambda(s_i)) \, ds_n \cdots ds_1 \\
&= \frac{e^{-\bar{\lambda}(b-a)}}{n!} \left(\int_a^b \bar{\lambda} - \lambda(s) \, ds\right)^n \\
&= \frac{e^{-\bar{\lambda}(b-a)}}{n!} \left(\bar{\lambda}(b-a) - \int_a^b \lambda(s) \, ds\right)^n ,
\end{aligned}$$

where we used an arbitrary permutation $s_1, ..., s_n$ of $t_1, ..., t_n$. Hence, by the law of total probability this finalizes to

$$\begin{aligned}
\mathbb{P}(\mathcal{N}((a,b]) = 0) &= \sum_{n=0}^{\infty} \mathbb{P}\left(\mathcal{N}((a,b]) = 0, \, \bar{\mathcal{N}}((a,b]) = n\right) \\
&= \sum_{n=0}^{\infty} \frac{e^{-\bar{\lambda}(b-a)}}{n!} \left(\bar{\lambda}(b-a) - \int_a^b \lambda(s) \, ds\right)^n \\
&= e^{-\bar{\lambda}(b-a)} \sum_{n=0}^{\infty} \frac{\left(\bar{\lambda}(b-a) - \int_a^b \lambda(s) \, ds\right)^n}{n!} \\
&= e^{-\bar{\lambda}(b-a)} e^{\bar{\lambda}(b-a) - \int_a^b \lambda(s) \, ds} \\
&= e^{-\int_a^b \lambda(s) \, ds} .
\end{aligned}$$

This specifies the distribution of the next event time and as TPPs can be constructed sequentially, the obtained event times are realizations of a non-homogeneous Poisson Process with intensity function λ. □

Theorem 2.5.2
Event sequences constructed by Algorithm 2 are realizations of a TPP with conditional intensity function λ^*.

Proof. By definition, the conditional intensity function λ^* is a deterministic function of time in between two events (which are random). Hence, simulating the next event time of an arbitrary TPP is equivalent to simulating the next event time of a non-homogeneous Poisson process with deterministic (conditioned on the history) intensity function $\lambda(t) = \lambda^*(t)$. As Theorem 2.5.1 shows the validity of the algorithm for non-homogeneous Poisson processes, it remains to prove that the upper bound $\bar{\lambda}$ of the conditional intensity function can be chosen iteratively. This is a consequence of the independent increments of the homogeneous Poisson process $\bar{\mathcal{N}}$ as

$$\mathbb{P}(\bar{\mathcal{N}}((a,b]) = 0 \mid \bar{\mathcal{N}}((0,a]) = 0) = \mathbb{P}(\bar{\mathcal{N}}((a,b]) = 0) .$$

□

Recall that new candidate event times are generated according to a homogeneous Poisson process with rate $\bar{\lambda}$ and that these get accepted as event times with probability

$\frac{\lambda^*(\cdot)}{\bar{\lambda}}$. Hence, for large $\bar{\lambda}$ more points are generated within a period of time, but the rejection rate is high too. Therefore in terms of efficiency, the upper bound $\bar{\lambda}$ should be chosen as small as possible.

The difficulty of finding local bounds for the conditional intensity function strongly depends on the TPP. For a univariate Hawkes process this is simple.

Example 2.5.3 (Simulation of a univariate Hawkes Process)
The conditional intensity function of a univariate Hawkes process decreases monotonically in between two events. Hence, we can always choose

$$l(t) := \infty\,, \quad m(t) := \lambda^*(t+) := \lim_{s \searrow t} \lambda^*(s)$$

and apply Algorithm 2. Note that if the algorithm rejects some t as event time, the smaller bound $m(t) := \lambda^*(t+) = \lambda^*(t)$ can be used in the next iteration step. This makes the algorithm more efficient and is sometimes called adaptive thinning. An illustration is given in Figure 2.7.

For some special cases of TPPs there exist other, more efficient simulation algorithms. For the purpose of generality, we stick to the thinning algorithm presented here.

2.5.3 Simulation of Marked Temporal Point Processes

The extension of Ogata's modified thinning algorithm to the case of MTPPs is straightforward, as the evolutionary character of the ground process stays valid and the mark distribution is specified conditional on the history of the process.

First, consider an MTPP with mark space $\mathcal{M} = \mathcal{R}$ for some $\mathcal{R} \subset \mathbb{R}$. Analogue to the case of TPPs, the next event time is simulated and using the conditional mark distribution a corresponding mark is generated (see Algorithm 3).

Algorithm 3: Ogata's modified thinning algorithm for MTPPs with $\mathcal{M} \subset \mathbb{R}$

Input: Conditional intensity function of the ground process λ_g^*, conditional mark distribution $f^*(m|t)$, finite time interval $[0, T]$
Output: Time-mark event sequence $\{(t_1, m_1), ..., (t_n, m_n)\}$

$t := 0$
$i := 1$
while $t \leq T$ **do**
  Compute constants $l(t)$ and $\bar{\lambda}(t)$ such that $\bar{\lambda}(t) \geq \sup_{s \in [t, t+l(t)]} \lambda_g^*(s)$.
  Generate $\tau \sim \text{Exp}\left(\bar{\lambda}(t)\right)$.
  Generate $U \sim \text{Unif}([0, 1])$.
  if $\tau > l(t)$ **then**
    $t = t + l(t)$
  else if $t + \tau > T$ **or** $U > \frac{\lambda_g^*(t+\tau)}{\bar{\lambda}(t)}$ **then**
    $t = t + \tau$
  else
    $t = t + \tau$
    $t_i := t$
    Generate $m_i \sim f^*(m|t_i)$.
    $i = i + 1$
  end
end

Algorithm 3 is basically identical to the Thinning Algorithm for TPPs (Algorithm 2) except that in this case a local bound for the ground intensity is calculated, as this controls the time component of the process and additionally a corresponding mark is generated according to the conditional mark distribution.

For the simulation of multivariate TPPs with continuous marks, we provide a slight reformulation of Algorithm 3 and give a comprehensive description. The algorithm is based on [33, Chapter 1] and an implementation is given in Algorithm 4.

Algorithm 4: Ogata's modified thinning algorithm for an M-variate TPP with continuous marks (cf. [33, Algorithm 1.21])

Input: Conditional intensity functions of the component processes λ_m^*, conditional mark value distributions $f^*\left(m^{(V)} \mid t, m^{(T)}\right)$, finite time interval $[0, T]$

Output: Time-mark event sequence $\left\{\left(t_1, m_1^{(T)}, m_1^{(V)}\right), ..., \left(t_n, m_n^{(T)}, m_n^{(V)}\right)\right\}$

$t := 0$
$i := 1$
while *True* **do**
  for $m = 1$ **to** M **do**
    Compute constant $\bar{\lambda}_m(t)$ such that $\bar{\lambda}_m(t) \geq \sup_{s \in [t,\infty)} \lambda_m^*(s)$.
    $t_m := t$
    while *True* **do**
      Generate $\tau \sim \text{Exp}\left(\bar{\lambda}_m(t)\right)$.
      Generate $U \sim \text{Unif}([0,1])$.
      $t_m = t_m + \tau$
      if $U \leq \frac{\lambda_m^*(t_m)}{\bar{\lambda}_m(t)}$ **and** $t_m \leq T$ **then**
        $t_{i,m} = t_m$
        break
      else if $t_m > T$ **then**
        $t_{i,m} = \text{NULL}$
        break
      end
    end
  end
  if $\{t_{i,m} \mid m \in \{1, ..., M\}\} \neq \emptyset$ **then**
    $t_i := \min_{m \in \{1,...,M\}} t_{i,m}$
    $m_i^{(T)} := \operatorname{argmin}_{m \in \{1,...,M\}} t_{i,m}$
    Generate $m_i^{(V)} \sim f^*\left(m^{(V)} \mid t_i, m_i^{(T)}\right)$.
    $t = t_i$
    $i = i + 1$
  else
    break
  end
end

Consider here the case that the local bounds $\bar{\lambda}_m(\cdot)$ are valid for the whole time period after an event has occurred, i.e. $l_m(t) = \infty$. In order to give an understanding of the algorithm, assume that the last event is given by the tuple $\left(t_{i-1}, m_{i-1}^{(T)}, m_{i-1}^{(V)}\right)$. How does the algorithm simulate the time, the mark type and the mark value of the next event?

Analogue to the univariate case, for each mark type $m^{(T)} \in \{1, ..., M\}$ a candidate event time $t_{i,m^{(T)}}$ is generated. Amongst this set of candidate times, the smallest one is picked as the next event time t_i together with its mark type $m_i^{(T)}$. The mark value $m_i^{(V)}$ is given by a realization of the conditional mark value distribution $f^*\left(m^{(V)}|t_i, m_i^{(T)}\right)$. All the other candidate event times are discarded. This procedure is then repeated iteratively until the end time T.

The question remains why this provides a realization of the corresponding point process. Let us therefore consider an arbitrary mark type $m^{(T)} \in \{1, ..., M\}$. The corresponding candidate event time $t_{i,m^{(T)}}$ is generated by implicitly assuming that no event of the same mark type $m^{(T)}$ or of another type $\bar{m}^{(T)} \neq m^{(T)}$ has occurred before time $t_{i,m^{(T)}}$. For the event

$$t_i = \min_{m=1,...,M} t_{i,m}$$

of mark type

$$m_i^{(T)} := \operatorname*{argmin}_{m \in \{1,...,M\}} t_{i,m} \,,$$

this is indeed a valid assumption. In contrast, for all the other candidate event times $t_{i,\bar{m}^{(T)}}$ with $\bar{m}^{(T)} \neq m_i^{(T)}$ this assumption is wrong. Their conditional intensity function is affected by the event occurring at time t_i and is therefore different to the one used to generate $t_{i,\bar{m}^{(T)}}$. Hence, all the other candidate event times have to be discarded. Note that to know for which mark type the assumption is valid, we of course first have to generate the candidate event times of all mark types.

Chapter 3

Neural Point Processes

Neural networks originate from neuroscience and the signal processing of neurons in the brain (see e.g. [34]). They were developed to artificially duplicate the structure of the brain to obtain an 'intelligent' system, as well as to get a better understanding of the functionality of the brain in general. Over time, neural networks untied this analogy and evolved to a powerful tool for many real world applications including object and speech recognition, machine translation and data generation. For an overview of the diverse range of applications see e.g. [10].

Inspired by this usefulness, we present in this chapter the *Neural Hawkes process* as proposed in [36], which is a point process whose conditional intensity function is modelled by a suitable *Recurrent Neural Network*. Before describing this approach in detail, we first give some background information on *Feedforward Neural Networks* and *Recurrent Neural Networks*.

3.1 Feedforward Neural Networks

In this section, we shortly discuss the main concepts of *Feedforward Neural Networks (FNNs)*, as will be necessary to understand the topics of the following chapters. The presented framework is based on [50].

A *Feedforward Neural Network (FNN)* can be seen as a universal function approximator, which is composed of several simple functions, so-called *layers*. Each of these layers consists of an affine transformation followed by a non-linear *activation function* and provides a new representation of the input data. Activation functions are essential for Neural Networks, as these make the model non-linear.

Definition 3.1.1 (Activation Function)
A Lipschitz continuous and monotonic function $\phi : \mathbb{R} \to \mathbb{R}$ is called activation function.

The Lipschitz continuity ensures the activation function to have bounded gradients, which is crucial for gradient descent training (more details in Section 3.2.2) and monotonicity benefits convergence. Prominent representatives of the class of activation functions are:

— *Rectified Linear Unit*:

$$\mathrm{ReLU} : \mathbb{R} \to \mathbb{R}_{\geq 0} : x \mapsto \max(x, 0)$$

— *Sigmoid function*:

$$\sigma : \mathbb{R} \to [0, 1] : x \mapsto \frac{1}{1 + e^{-x}}$$

— *Hyperbolic tangent*:

$$\tanh : \mathbb{R} \to [-1,1] : x \mapsto \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

A visualization of these three activation functions is given in Figure 3.1. The sigmoid and the tanh function suffer from saturating gradients for $|x| \to \infty$, which causes problems in gradient descent training. Therefore, mostly the ReLU and slight modifications of it are used as activation function for FNNs. Nevertheless, the sigmoid and the tanh function play a crucial role in the *Long-Short Term Memory (LSTM)* in Section 3.2.4 and the *Neural Hawkes process* in Section 3.3.

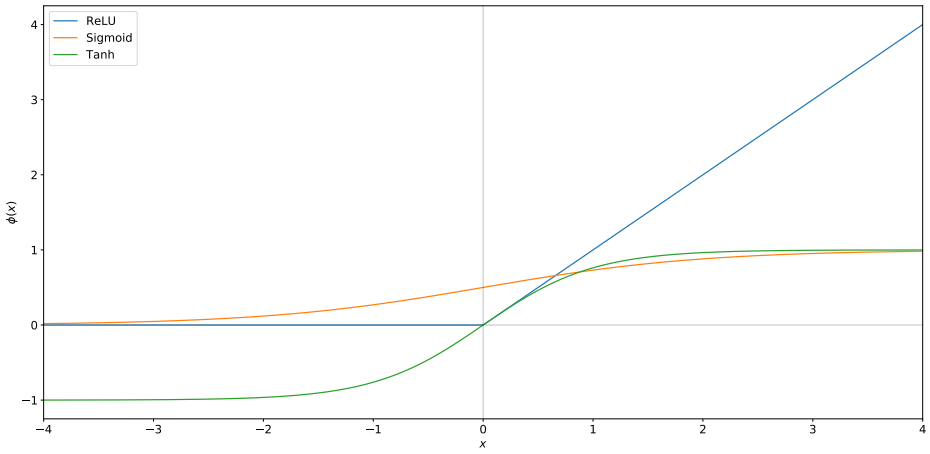


Figure 3.1: Overview of different activation functions.

Definition 3.1.2 (Feedforward Neural Network)
Let $L, N_0, \dots, N_{L+1} \in \mathbb{N}$, ϕ an activation function, Θ an Euclidean vector space and for $l \in \{1, \dots, L+1\}$ let $a_l : \mathbb{R}^{N_{l-1}} \to \mathbb{R}^{N_l}$ be an affine function. A mapping $f : \mathbb{R}^{N_0} \times \Theta \to \mathbb{R}^{N_{L+1}}$ defined by

$$f(x, \theta) = a_{L+1} \circ f_L \circ \cdots \circ f_1(x) \,,$$

where

$$f_l = \phi \circ a_l \,, \quad l \in \{1, \dots, L\}$$

for ϕ being applied component-wise and $\circ$ denoting the composition of two functions, is called a Feedforward Neural Network (FNN) with L hidden layers. In this setting, N_0 and N_{L+1} represent the input and output dimension, $N_1, \dots, N_L$ the hidden dimensions and a_{L+1} the output layer. Furthermore, for any $l \in \{1, \dots, L+1\}$ the function a_l is represented by $a_l : x \mapsto W^{(l)}x + b^{(l)}$, where $W^{(l)} \in \mathbb{R}^{N_l \times N_{l-1}}$ denotes the weight matrix and $b^{(l)} \in \mathbb{R}^{N_l}$ the bias. Using this notation the MLP's parameters are given by

$$\theta = \left(W^{(1)}, \dots, W^{(L+1)}, b^{(1)}, \dots, b^{(L+1)}\right) \in \Theta.$$

Further, we denote the class of FNNs with L hidden layers mapping from $\mathbb{R}^{d_0}$ to $\mathbb{R}^{d_1}$ as $FNN_{d_0,d_1,L}$. If $L >> 1$, we call the FNN deep.

The network is called feedforward as information is processed in one-way, from the input through the hidden layers to the output. The computational graph is therefore directed and acylic as can be seen in Figure 3.2. If we allow the model to contain feedback loops, we obtain *Recurrent Neural Networks (RNNs)* which are the topic of Section 3.2.

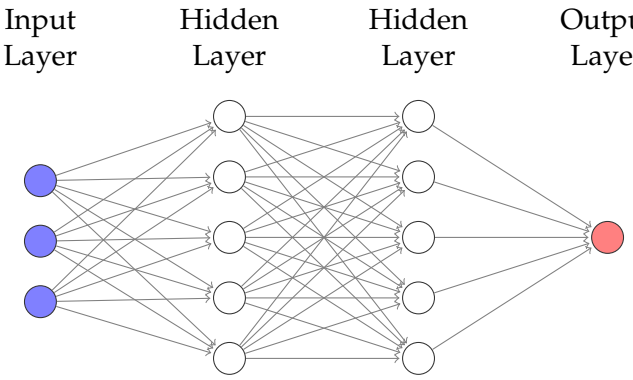


Figure 3.2: FNN with 2 hidden layers, input dimension $N_0 = 3$, output dimension $N_3 = 1$ and hidden dimensions $N_1 = N_2 = 5$.

Note that FNNs are often called *fully connected Neural Network* or *Multilayer Perceptron (MLP)*.

Famous theoretical results have been proven to demonstrate the modelling abilities of Feedforward Neural Networks. The most popular, the so-called *Universal Approximation Theorem*, shows that even simple Feedforward Neural Networks with one hidden layer and a sufficiently large hidden dimension can approximate any continuous function on a compact interval arbitrarily well. For the sake of completeness, we state Theorem 1 and 2 from [26] in the form as is given in [3, Theorem 4.2].

Theorem 3.1.3 (Universal Approximation Theorem, cf. [3, Theorem 4.2])
Assume the activation function ϕ to be bounded and non-constant. The following statements hold:

— *For any finite measure μ on $\left(\mathbb{R}^{d_0}, \mathcal{B}\left(\mathbb{R}^{d_0}\right)\right)$ and $1 \leq p < \infty$, the set $FNN_{d_0,1,1}$ is dense in $L^p\left(\mathbb{R}^{d_0}, \mu\right)$.*

— *If ϕ is additionally continuous, the set $FNN_{N_0,1,1}$ is dense in $C\left(\mathbb{R}^{d_0}\right)$ with respect to the topology of uniform convergence on compact sets.*

Note that this carries over to the case of output dimension $d_1 > 1$ and more than one hidden layer. For more details on FNNs we refer the reader to [17, Chapter 6].

3.2 Recurrent Neural Networks

The need for an architecture to process sequential data led to the development of Recurrent Neural Networks (RNNs), see e.g. [46]. A model for sequential data should be able to process sequences of variable length and capture the dependencies of different sequence elements. RNNs solve this by using a recurrent structure and simply define the models transition from one step to the next, thereby enabling to store information by internal feedback-loops. In practise, RNNs have shown to be extremely powerful in many real world applications including hand-writing and speech recognition, music generation and machine translation.

We first give some basic information on RNNs and their structure. Afterwards, we discuss how RNNs can be trained to fit some data and we end this section with a more detailed look on some special RNN architectures, which are relevant for the introduction of the Neural Hawkes process in Section 3.3. Throughout this section, we mainly follow the great textbooks [17] and [18]. In particular, we put the theory into a more mathematical context.

Definition 3.2.1 (Recurrent Neural Network)
Let $\mathcal{X}, \mathcal{H}, \mathcal{Y}$ be real vector spaces, $(x_t)_{t=1,...,N} \in \mathcal{X}^N$ an input sequence of finite length N and ϕ_h, ϕ_y two activation functions. A Recurrent Neural Network (RNN) is a mapping

$$\begin{aligned} f : \mathcal{X}^N \times \Theta &\to \mathcal{Y}^N \\ ((x_t)_{t=1,...,N}, \theta) &\mapsto (y_t)_{t=1,...,N} \end{aligned}$$

defined recursively by

$$\begin{aligned} y_t &= f_{\theta_y}^{(y)}(h_t) \\ h_t &= f_{\theta_h}^{(h)}(x_t, h_{t-1}) \end{aligned}$$

for $\theta = (\theta_h, \theta_y) \in \Theta = \Theta_h \times \Theta_y$ and $h_0 := 0$. The functions

$$\begin{aligned} f^{(y)} : \mathcal{H} \times \Theta_y &\to \mathcal{Y} \\ (h, \theta_y) &\mapsto \phi_y \circ a_{\theta_y}^{(y)}(h) \\ f^{(h)} : \mathcal{X} \times \mathcal{H} \times \Theta_h &\to \mathcal{H} \\ (x, h, \theta_h) &\mapsto \phi_h \circ a_{\theta_h}^{(h)}(x, h) \end{aligned}$$

are given by affine mappings

$$\begin{aligned} a_{\theta_y}^{(y)} &: h \mapsto V_y h + b_y \\ a_{\theta_h}^{(h)} &: (x, h) \mapsto W_h x + V_h h + b_h \,, \end{aligned}$$

which are parametrized using the weight matrices W_h, V_h, V_y and biases b_h, b_y. We further call Θ the parameter space, $f_{\theta_y}^{(y)}$ the hidden-to-output mapping, $f_{\theta_h}^{(h)}$ the hidden mapping as well as x_t the input, h_t the hidden state and y_t the output at time t. The RNN parameters are hence given by

$$\theta = (\theta_h, \theta_y) = (W_h, V_h, b_h, V_y, b_y) \in \Theta \,.$$

A Recurrent Neural Network can be seen as a function mapping an input sequence to an output sequence using a recursive structure. For each $t \in \{1, ..., N\}$, the hidden state h_t depends on the input at the current time step x_t as well as the hidden state at the previous time step h_{t-1}. Based on the hidden state, the output at the current time step y_t is calculated.

Similarly to FNNs, the hidden-to-output mapping $f_{\theta_y}^{(y)}$ and the hidden mapping $f_{\theta_h}^{(h)}$ are constructed by composing a linear layer with an activation function. In this sense, RNNs can be seen as a straightforward extension of FNNs to the sequential case.

In particular, each time step uses the same hidden mapping and the same hidden-to-output mapping. This is called *parameter sharing* and represents an essential characteristic of RNNs. It is therefore enough to learn the parameters θ_h and θ_y once and apply it in each time step. Hence, the number of parameters is in $\mathcal{O}(1)$ as a function of the sequence length N. Note that this is far more efficient than learning individual mappings for each time step. This would have the additional disadvantage that the model would not be able to process sequences of lengths not being trained on.

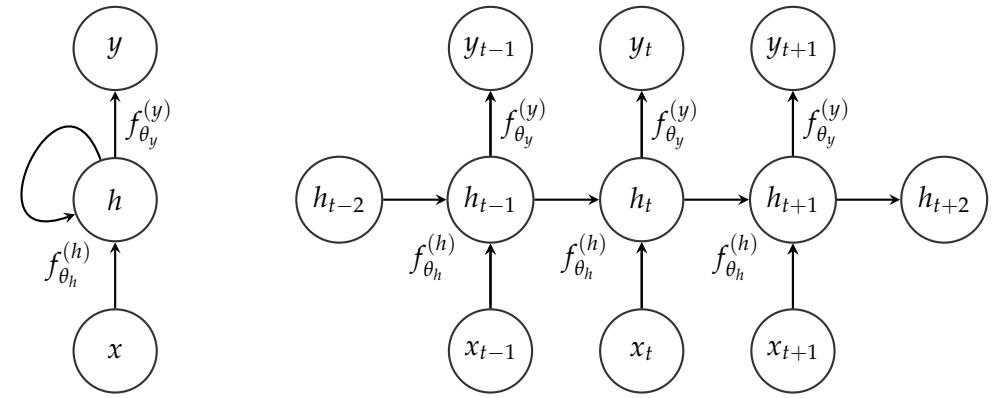
Note that the RNN according to Definition 3.2.1 has one hidden layer h. In practise, performance can often be boosted by considering *deep* RNNs with more than one hidden layer, which typically are more difficult to optimize (see e.g. [19]). This can easily be

incorporated in the definition given by including additional hidden layers before calculating the output. For example, the recursions of an RNN with 2 hidden layers are given by

$$\begin{aligned} y_t &= f_{\theta_y}^{(y)}\left(h_t^{(2)}\right) \\ h_t^{(2)} &= f_{\theta_h^{(2)}}^{(2)}\left(h_t^{(1)}, h_{t-1}^{(2)}\right) \\ h_t^{(1)} &= f_{\theta_h^{(1)}}^{(1)}\left(x_t, h_{t-1}^{(1)}\right). \end{aligned}$$

In the following, we stick to the case of an RNN with one hidden layer, as this suffices for our purposes.

There are in general two ways of visualizing RNNs: using a *cyclic* or *unfolded computational graph*. The two methods are displayed in Figure 3.3.



(a) Cyclic graph (b) Unfolded graph

Figure 3.3: Computational graph of a recurrent neural network.

The cyclic graph represents the recurrent structure of the RNN, as it is given in Definition 3.2.1. The unfolded graph results by solving the corresponding recursion

$$\begin{aligned} h_{t-1} &= f_{\theta_h}^{(h)}(x_{t-1}, h_{t-2}) \\ h_t &= f_{\theta_h}^{(h)}(x_t, h_{t-1}) = f_{\theta_h}^{(h)}\left(x_t, f_{\theta_h}^{(h)}(x_{t-1}, h_{t-2})\right) \\ h_{t+1} &= f_{\theta_h}^{(h)}(x_{t+1}, h_t) = f_{\theta_h}^{(h)}\left(x_{t+1}, f_{\theta_h}^{(h)}\left(x_t, f_{\theta_h}^{(h)}(x_{t-1}, h_{t-2})\right)\right). \end{aligned}$$

Hence, the unfolded graph is directed and acyclic, which stands in line with the graph representation of FNNs. Furthermore, the unfolded graph has the advantage that one can immediately read off the way information is processed (*forward pass*) and how gradients can be calculated backward inductively (*backward pass*).

Before continuing with the optimization of RNNs, we want to give a more thorough understanding of the RNN structure and in particular of the hidden state (cf. [17, Chapter 10.2.3]). Therefore, consider an input sequence $x_1, ..., x_N$ without corresponding output sequence. We can write the joint distribution of the inputs as a product of the conditional distributions

$$\mathbb{P}(x_1, ..., x_N) = \prod_{t=1}^{N} \mathbb{P}(x_t | x_1, ..., x_{t-1}).$$

Figure 3.4 presents this as a graphical model, in which a directed edge from x_i to x_j is drawn, if the conditional distribution of x_j depends on x_i in this factorization.

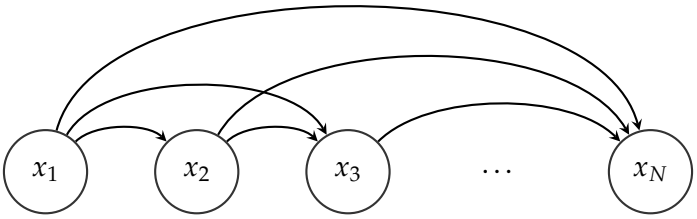


Figure 3.4: Fully connected graphical model of the joint distribution of $x_1, ..., x_N$.

Parametrizing the joint distribution in this way is very inefficient as the number of parameters grows like $\mathcal{O}(N^2)$. In contrast, the RNN uses a more efficient approach. Consider the natural filtration of the input sequence[1]

$$\mathcal{H}_t^+ = \sigma(x_s | s \leq t) = \sigma(x_1, ..., x_t)$$

for $t = 1, ..., N$. Using the filtration, the joint distribution can be written as

$$\mathbb{P}(x_1, ..., x_N) = \prod_{t=1}^{N} \mathbb{P}(x_t | \mathcal{H}_{t-1}^+) \, ,$$

which has the graphical representation given in Figure 3.5.

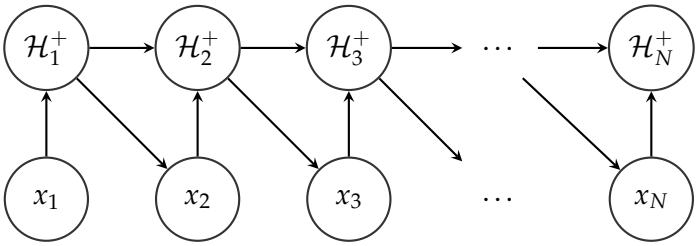


Figure 3.5: Filtration based graphical model of the joint distribution of $x_1, ..., x_N$.

In contrast to the fully connected model, there is no edge from each node to all of the following nodes in the sequence. Instead, the filtration incorporates the information of the past and gets updated with the new input at each time step. In the RNN, the hidden state aims the role of a task-specific, sufficient statistic of the filtration. Instead of conditioning on the filtration, we can therefore condition on the hidden state to parametrize the joint distribution

$$\mathbb{P}(x_1, ..., x_N) = \prod_{t=1}^{N} \mathbb{P}(x_t | \mathcal{H}_{t-1}^+) \cong \prod_{t=1}^{N} \mathbb{P}(x_t | h_{t-1}) \, .$$

In the case of an output sequence being associated to the input sequence, the hidden state h_t incorporates the relevant information of the past input sequence $x_1, ..., x_t$ for the specific task to output an estimate of y_t. Note that this implicitly assumes that future outputs do not depend on past outputs.

Analogue to the Universal Approximation Theorem of FNNs, there exist results for RNNs, which theoretically justify their application for sequence modelling. For completeness, we state an appropriate result here.

[1] We use the superscript $+$ to be consistent with the notation of the the previous sections.

Theorem 3.2.2 (cf. [21, Theorem 3])
Let $\mathcal{X}, \mathcal{Y}$ be real vector spaces and $N \in \mathbb{N}$. Any measurable sequence-to-sequence mapping $g : \mathcal{X}^N \to \mathcal{Y}^N$ can be approximated arbitrarily well in probability by an RNN $f : \mathcal{X}^N \times \Theta \to \mathcal{Y}^N$ with one hidden layer and a sufficiently large hidden dimension.

3.2.1 Backpropagation through Time

In order to learn the parameters of an RNN, a *loss function* has to be specified which describes how good the model describes some data. The larger the loss, the worse the fit of the model. Hence, we are looking for parameters, which minimize the loss function. Typically, *gradient descent algorithms* are applied for this (more in Section 3.2.2). Therefore, the gradients of the loss function with respect to the model parameters have to be derived. The corresponding algorithm for RNNs is called *Backpropagation through time (BPTT)*, see e.g. [51]. We derive here the equations for our setting.

The main idea of the BPTT algorithm is to use the chain rule to calculate gradients efficiently. The BPTT algorithm can be seen as the extension of the classic Backpropagation algorithm used to learn the parameters of FNNs incorporating the recursive structure of the RNN. BPTT can also be understood as applying Backpropagation to the unrolled graph, as can be seen in Figure 3.6. For more details on the Backpropagation algorithm for FNNs, we refer the reader to [17, Chapter 6].

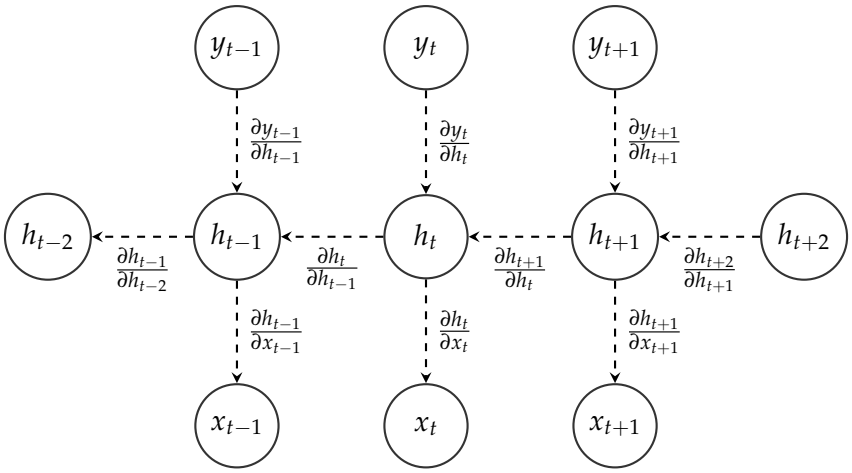


Figure 3.6: Calculation of gradients using Backpropagation through time (BPTT).

As already pointed out, we first have to decide for a *loss function*. In the setting of RNNs, the loss is typically given as the sum of the individual losses at each time step t, i.e.

$$L(\theta) = \sum_{t=1}^{N} L_t(\theta) \, .$$

Depending on the task, the loss can attain several different forms. Often the negative log-likelihood is used.

In the BPTT algorithm, we calculate gradients by traversing the unrolled graph backward recursively starting at the end of the sequence x_N and going to the start x_1. First, we calculate the derivative of the loss with respect to the individual loss at each time step $t \in \{1, ..., N\}$

$$\frac{\partial L(\theta)}{\partial L_t(\theta)} = 1 \, .$$

The output y_t at each time step t only contributes to the loss $L(\theta)$ through its effect on the

loss at the same time $L_t(\theta)$. Therefore, using the chain rule we obtain

$$\nabla_{y_t} L(\theta) = \frac{\partial L(\theta)}{\partial L_t(\theta)} \frac{\partial L_t(\theta)}{\partial y_t} = \frac{\partial L_t(\theta)}{\partial y_t} ,$$

which depends on the chosen form of the loss $L_t(\theta)$.

Next, the gradients of the loss with respect to the hidden states have to be calculated. The hidden state h_N corresponding to the last input in the sequence x_N just affects the output y_N at the same time step. Hence,

$$\nabla_{h_N} L(\theta) = \left(\frac{\partial y_N}{\partial h_N} \right)^T \nabla_{y_N} L(\theta) = V_y^T \operatorname{diag} \left(\phi_y' \left(a_{\theta_y}^{(y)}(h_N) \right) \right) \nabla_{y_N} L(\theta) ,$$

as the diagonal matrix $\operatorname{diag}(\phi_y'(\cdot))$ is the Jacobian of the point-wise applied activation function ϕ_y.

This is different for all the other time steps $t \in \{1, ..., N-1\}$, as in this case the hidden state h_t has as direct successors in the unrolled graph the output at the same time step y_t, but also the hidden state h_{t+1} at the next time step $t+1$. Note that therefore h_t has an effect on all the future losses $L_s(\theta)$, $s \geq t$. We need to take this into account when calculating gradients. Hence, we iterate backward through time starting at $t = N-1$ until time step $t = 1$ and obtain again using the chain rule:

$$\begin{aligned} \nabla_{h_t} L(\theta) &= \left(\frac{\partial h_{t+1}}{\partial h_t} \right)^T \nabla_{h_{t+1}} L(\theta) + \left(\frac{\partial y_t}{\partial h_t} \right)^T \nabla_{y_t} L(\theta) \\ &= V_h^T \operatorname{diag} \left(\phi_h' \left(a_{\theta_h}^{(h)}(x_{t+1}, h_t) \right) \right) \nabla_{h_{t+1}} L(\theta) + V_y^T \operatorname{diag} \left(\phi_y' \left(a_{\theta_y}^{(y)}(h_t) \right) \right) \nabla_{y_t} L(\theta) . \end{aligned}$$

Recall that by iterating backward through time, the gradient of the successive time step $\nabla_{h_{t+1}} L(\theta)$ is already known.

Having the gradients of the loss with respect to the outputs y_t and the hidden states h_t, the gradients corresponding to the model parameters V_y, b_y, W_h, V_h, b_h can be obtained. For the parameters of the hidden-to-output mapping, this is again straightforward:

$$\nabla_{V_y} L(\theta) = \sum_{t=1}^{N} \left(\nabla_{V_y} y_t \right)^T \nabla_{y_t} L(\theta) = \sum_{t=1}^{N} h_t^T \operatorname{diag} \left(\phi_y' \left(a_{\theta_y}^{(y)}(h_t) \right) \right) \nabla_{y_t} L(\theta)$$

$$\nabla_{b_y} L(\theta) = \sum_{t=1}^{N} \left(\frac{\partial y_t}{\partial b_y} \right)^T \nabla_{y_t} L(\theta) = \sum_{t=1}^{N} \operatorname{diag} \left(\phi_y' \left(a_{\theta_y}^{(y)}(h_t) \right) \right) \nabla_{y_t} L(\theta)$$

As already pointed out for the hidden state, we have to be more careful when calculating the gradients with respect to the parameters of the hidden mapping, as by parameter sharing these effect the loss in all time steps. Thus, for each parameter of the hidden mapping W_h, V_h, b_h, we use the superscript (t) to split the effect to the individual time steps. By doing so, we obtain

$$\nabla_{W_h} L(\theta) = \sum_{t=1}^{N} \left(\nabla_{W_h^{(t)}} h_t \right)^T \nabla_{h_t} L(\theta) = \sum_{t=1}^{N} x_t^T \operatorname{diag} \left(\phi_h' \left(a_{\theta_h}^{(h)}(x_t, h_{t-1}) \right) \right) \nabla_{h_t} L(\theta)$$

$$\nabla_{V_h} L(\theta) = \sum_{t=1}^{N} \left(\nabla_{V_h^{(t)}} h_t \right)^T \nabla_{h_t} L(\theta) = \sum_{t=1}^{N} h_{t-1}^T \operatorname{diag} \left(\phi_h' \left(a_{\theta_h}^{(h)}(x_t, h_{t-1}) \right) \right) \nabla_{h_t} L(\theta)$$

$$\nabla_{b_h} L(\theta) = \sum_{t=1}^{N} \left(\frac{\partial h_t}{\partial b_h^{(t)}} \right)^T \nabla_{h_t} L(\theta) = \sum_{t=1}^{N} \operatorname{diag} \left(\phi_h' \left(a_{\theta_h}^{(h)}(x_t, h_{t-1}) \right) \right) \nabla_{h_t} L(\theta)$$

3.2.2 Gradient Descent algorithms

Having the gradients of the loss function with respect to the model parameters, any gradient based optimization routine can be used to learn them. The basic idea of gradient descent algorithms is nicely described in [7, Chapter 8].

Assume we are interested in minimizing the loss function $L(\theta)$ for $\theta \in \Theta$. Further, suppose the algorithm currently halts at θ_0. We want to find a $\theta \in \Theta$ such that

$$L(\theta) < L(\theta_0) \ .$$

Therefore, consider a direction $d \neq 0$ with $\|d\| = 1$ and define $\theta := \theta_0 + d$. Applying a first-order Taylor approximation yields

$$L(\theta) \approx L(\theta_0) + \nabla L(\theta_0)^T d \ .$$

Hence, the direction d has to be chosen to minimize the term $\nabla L(\theta_0)^T d$. Using the Cauchy-Schwarz inequality we obtain

$$\nabla L(\theta_0)^T d \leq \|\nabla L(\theta_0)\| \cdot \|d\| = \|\nabla L(\theta_0)\| \ .$$

Choosing $d_{max} := \frac{\nabla L(\theta_0)}{\|\nabla L(\theta_0)\|}$ leads to

$$\nabla L(\theta_0)^T d_{max} = \frac{\|\nabla L(\theta_0)\|^2}{\|\nabla L(\theta_0)\|} = \|\nabla L(\theta_0)\| \ .$$

Hence, the gradient gives the direction of the strongest ascent of the loss function L in θ_0. Analogue, $d_{min} := -d_{max} = -\frac{\nabla L(\theta_0)}{\|\nabla L(\theta_0)\|}$, i.e. the negative gradient, gives the direction of the strongest descent of the loss function. As a result, we define $\theta := \theta_0 + d_{min} = \theta_0 - \frac{\nabla L(\theta_0)}{\|\nabla L(\theta_0)\|}$ as the next point to evaluate the loss function at.

By iteratively applying a step in the direction of the negative gradient, we obtain an algorithm to minimize the loss function L. This procedure is called *gradient descent*. A simple implementation is given in Algorithm 5, where we additionally use the so-called *learning rate* η to control the size of the gradient descent step.

Algorithm 5: Gradient Descent

Input: Training data $\mathcal{D}$, loss function L, learning rate η, initial parameter θ
Output: Optimized parameter θ

1 **while** *not stopped* **do**
2 Perform the gradient descent step

$$\theta = \theta - \eta \nabla L(\theta, \mathcal{D}) \ .$$

3 **end**

For Neural Network training, typically slightly modified versions of the base Gradient Descent algorithm are applied. In the following, we explain two of the most common based on [17, Chapter 8].

Stochastic Gradient Descent

Neural Network training is typically done *batch-wise*. Instead of passing the whole data set at once through the RNN, only a small subset (*batch*) is used, the loss calculated accordingly and the error backpropagated using the gradient information. The elements

forming a batch are selected randomly to break any symmetry possibly present in the initial data ordering.

Batch-wise training is applied since processing the whole data set at once is typically to consuming or even infeasible in terms of memory or computational time. Calculations based on batches are much faster and computationally more efficient. In particular, batch-wise training has the further advantages that the computational time of an update step does not depend on the actual size of the underlying data and gradient calculations can be performed simultaneously (*Parallelization*), which can speed up learning tremendously.

The *batch-size*, i.e. the number of data elements grouped to one batch, has to be chosen large enough to give a reliable estimate of the loss and the corresponding gradient, but small enough to be processable. As some hardware architectures benefit from, the batch-size is typically chosen of the size 2^n.

Applying gradient descent batch-wise is typically called *Stochastic gradient descent (SGD)* or *Batch gradient descent*. A simple implementation is given in Algorithm 6.

Algorithm 6: Stochastic Gradient Descent (cf. [17, Algorithm 8.1])

Input: Training data $\mathcal{D}$, loss function L, batch size n_B, learning rate scheme $(\eta_i)_{i\in\mathbb{N}}$, initial parameter θ

Output: Optimized parameter θ

$i := 1$

while *not stopped* **do**

Sample a batch $\mathcal{B} = \{x^{(1)}, ..., x^{(n_B)}\}$ of n_B randomly selected elements from the training set $\mathcal{D}$.

Calculate the gradient of the loss with respect to θ on the batch $\mathcal{B}$

$$\hat{\nabla}_\theta = \frac{1}{n_B} \sum_{i=1}^{n_B} \nabla_\theta L\left(\theta, x^{(i)}\right).$$

Perform the gradient descent step

$$\theta = \theta - \eta_i \hat{\nabla}_\theta.$$

Adjust the iteration index: $i = i + 1$.

end

Often a learning-rate scheme is used as this leads to a better convergence behaviour than a constant learning-rate. Typically, the scheme is specified such that the learning-rate decreases until some iteration and than stays constant. A simple decreasing scheme is for example given by

$$\eta_i = \begin{cases} \left(1 - \frac{i}{I}\right)\eta_0 + \frac{i}{I}\eta_I\,, & i \leq I \\ \eta_I & , \text{ else} \end{cases}$$

for some predefined period I, as well as initial and target learning-rates η_0 and η_I. The learning rate usually attains values in the order of $1e-2$ or $1e-3$.

Adam algorithm

In the SGD algorithm, parameters are updated according to the product of the learning rate and the gradient. Hence, the learning rate is a crucial hyperparameter for successful

Neural Network training, as it controls the size of the update step. Choosing a suitable learning rate is not at all an easy task. If the learning rate is too small, learning may run forever and if it is too large, it may even prevent convergence, as parameter update steps are too large for the gradient to provide valid guide. Therefore, instead of predefining a suitable learning rate or even thinking of learning rate schemes, algorithms based on *adaptive learning rates* learn the rates themselves depending on the context. In this way, these algorithms typically outperform standard SGD. One of the most prominent gradient descent algorithm with adaptive learning rate is the *Adam* algorithm. As we use this algorithm in the numerical study, we have a more detailed look here.

The Adam algorithm, short for *adaptive moments*, was proposed in [29] and adapts the learning rate based on unbiased estimates of the first and second moment of the gradient. Moreover, the algorithm incorporates current and past gradients in each update step (*Momentum*). Depending on the application, this can fasten computations enormously. An implementation is given in Algorithm 7.

Analogue to the SGD algorithm, the Adam algorithm first calculates the gradient of the loss batch-wise. Instead of using a simple Gradient descent step for the parameter update, an exponential moving average of past and current gradients is used to estimate the first and second moment of the gradient, see m_1 and m_2. The hyperparameters β_1 and β_2 control the decay rate of the past gradients. The moment estimates m_1 and m_2 are biased due to the initialization $m_1 = m_2 = 0$. It can easily be shown that we can correct for this and obtain with $\hat{m}_1$ and $\hat{m}_2$ unbiased estimates.

We prove the corresponding result for the first moment, for the second moment this follows analogue. We denote by the superscript (i) the estimates of iteration i. Using the recursive structure of the first moment estimate, it can easily be seen that

$$m_1^{(i)} = \beta_1 m_1^{(i-1)} + (1-\beta_1)\hat{\nabla}_\theta^{(i)} = (1-\beta_1)\sum_{j=1}^{i} \beta_1^{i-j}\hat{\nabla}_\theta^{(j)}.$$

Therefore assuming $\mathbb{E}\left[\hat{\nabla}_\theta^{(j)}\right] = \mathbb{E}\left[\nabla_\theta\right]$ for all j, we obtain

$$\begin{aligned}
\mathbb{E}\left[m_1^{(i)}\right] &= (1-\beta_1)\sum_{j=1}^{i} \beta_1^{i-j}\mathbb{E}\left[\hat{\nabla}_\theta^{(j)}\right] \\
&= \mathbb{E}\left[\nabla_\theta\right](1-\beta_1)\sum_{j=1}^{i} \beta_1^{i-j} \\
&= \mathbb{E}\left[\nabla_\theta\right]\left(1-\beta_1^i\right)
\end{aligned}$$

with the following result for the geometric series ($\beta_1 \in (0,1)$)

$$\sum_{j=1}^{i} \beta_1^{i-j} = \frac{1-\beta_1^i}{1-\beta_1}\,.$$

An unbiased estimator of the first moment of the gradient is thus given by

$$\hat{m}_1^{(i)} = \frac{m_1^{(i)}}{1-\beta_1^i}\,.$$

Based on the unbiased moment estimates $\hat{m}_1$ and $\hat{m}_2$, the parameter update step is performed. In particular, the size of the update step is not equal to the learning rate times

the norm of the gradient as for SGD. Instead, it depends on the norm and the direction of past and current gradients. An approximate bound of the effective step size is given by the step size parameter α as

$$\left|\left|\alpha \cdot \frac{\hat{m}_1}{\sqrt{\hat{m}_2} + \epsilon}\right|\right| \lessapprox \alpha$$

(see [29, Section 2.1]).

Algorithm 7: Adam algorithm (cf. [29, Algorithm 1])

Input: Training set $\mathcal{D}$, loss function L, batch size n_B, stepsize α (default: 0.001), exponential decay rates for the moment estimates $\beta_1, \beta_2 \in (0, 1)$ (default: $\beta_1 = 0.9, \beta_2 = 0.999$), numerical stabilizer ϵ (default: 1e-8), initial parameter θ

Output: Optimized parameter θ

Initialize the first and second moment variable: $m_1 := 0, m_2 := 0$

Initialize the iteration index: $i := 1$

while *not stopped* **do**

Sample a batch $\mathcal{B} = \{x^{(1)}, ..., x^{(n_B)}\}$ of n_B randomly selected elements from the training set $\mathcal{D}$.

Calculate the gradient of the loss w.r.t. θ on the batch $\mathcal{B}$

$$\hat{\nabla}_\theta = \frac{1}{n_B} \sum_{i=1}^{n_B} \nabla_\theta L\left(\theta, x^{(i)}\right).$$

Update the biased first moment estimate

$$m_1 = \beta_1 m_1 + (1 - \beta_1)\hat{\nabla}_\theta.$$

Update the biased second moment estimate

$$m_2 = \beta_2 m_2 + (1 - \beta_2)\hat{\nabla}_\theta \odot \hat{\nabla}_\theta.$$

Correct for the bias in the first moment estimate

$$\hat{m}_1 = \frac{m_1}{1 - \beta_1^i}.$$

Correct for the bias in the second moment estimate

$$\hat{m}_2 = \frac{m_2}{1 - \beta_2^i}.$$

Update the parameter

$$\theta = \theta - \alpha \frac{\hat{m}_1}{\sqrt{\hat{m}_2} + \epsilon}.$$

Adjust the iteration index: $i = i + 1$

end

3.2.3 Exploding and vanishing gradients problem

One of the main tasks of RNNs when being applied to sequence modelling is to learn short- and long-range dependencies. Long-range dependencies are present, if the output at some time t depends on the input at some time $s << t$. For example, when using RNNs to model temporal point processes, the RNN should be able to capture if and how past events effect future events. Especially the task of learning long-range dependencies is difficult due to the *exploding and vanishing gradients problem* (see e.g. [2]), meaning that the norm of the gradients either tend to increase (rarely) or decrease (often) exponentially as a function of the time lag. By reason of this exponential behaviour, it is much more difficult to learn long-range than short-range dependencies.

Note that this problem makes RNN training difficult as parameters get updated according to their gradient. Hence if the gradient vanishes, there is no signal left for adjusting the parameters and therefore learning becomes impossible. On the other side, for exploding gradients learning becomes unstable and irregular as the gradient only gives local information on the loss structure.

The main reason for the exploding and vanishing gradients problem is the recursive structure of the RNN as the same parameters are applied in each time step. Roughly speaking, the same way as multiplying an $x \geq 0$ again and again by itself either decreases to 0 (for $x \in [0, 1)$) or increases to ∞ (for $x > 1$), the norm of a product of Jacobians attains the same behaviour.

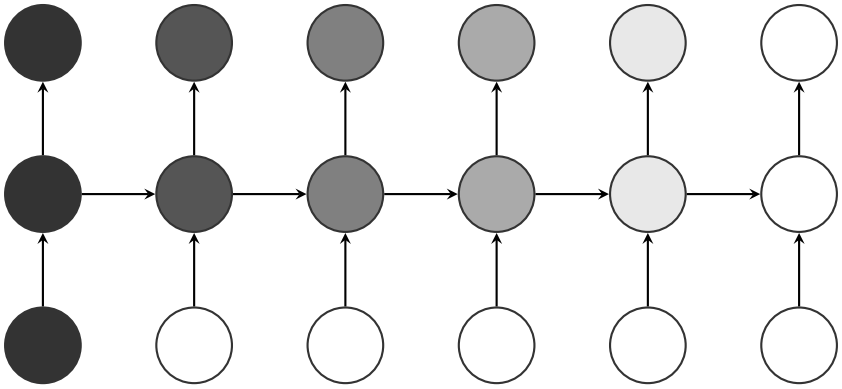


Figure 3.7: Vanishing Gradients in Standard RNN (reproduced from [18, Figure 4.1]).

An illustration of the vanishing gradients problem is given in Figure 3.7. The darker the color of a node the higher the sensitivity with respect to the input of the first time step. As the time lag increases, the sensitivity decreases i.e. gradients vanish.

For a more theoretical perspective, the authors of [41] proposed conditions on the activation functions and weight matrices of an RNN specifying the occurrence of exploding and vanishing gradients. To prove these results in our setting, we first give a definition.

Definition 3.2.3 (Spectral norm)
Let A denote a matrix. The Spectral norm $\|A\|_2$ is defined as the induced matrix norm of the L^2 norm, i.e.

$$\|A\|_2 := \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} .$$

Note that for any induced matrix norm $\|\cdot\|$ it holds ([47, Theorem 1.3])

$$\rho(A) \leq \|A\| ,$$

where $\rho(A)$ denotes the spectral radius of A (see Definition 2.2.16). This allows us to prove the following theorem.

Theorem 3.2.4 (Exploding and Vanishing Gradients)
Let f_θ be an RNN according to Definition 3.2.1 and assume

$$\left\|\phi_h^{'}\right\|_\infty := \sup_{x\in\mathbb{R}} |\phi_h^{'}(x)| \leq c$$

for some $c \in (0, \infty)$. It is valid:

1. *If $\|V_h\|_2 < \frac{1}{c}$, the norm of the gradients decays exponentially fast as a function of the time lag.*
2. *$\rho(V_h) > \frac{1}{c}$ is a necessary condition for the norm of the gradients to increase exponentially as a function of the time lag.*

Proof. First, note that $\left\|\phi_h^{'}\right\|_\infty \leq c$ implies $\left\|\mathrm{diag}(\phi_h^{'}(\cdot))\right\|_2 \leq c$.

1. The gradient of the hidden states with respect to successive time steps is given by

$$\frac{\partial h_{t+1}}{\partial h_t} = V_h^T \,\mathrm{diag}\left(\phi_h^{'}\left(a_{\theta_h}^{(h)}(x_{t+1}, h_t)\right)\right).$$

   Hence, taking the spectral norm and using the sub-multiplicativity we obtain

$$\left\|\frac{\partial h_{t+1}}{\partial h_t}\right\|_2 \leq \left\|V_h^T\right\|_2 \left\|\mathrm{diag}\left(\phi_h^{'}\left(a_{\theta_h}^{(h)}(x_{t+1}, h_t)\right)\right)\right\|_2 < \frac{1}{c}\cdot c = 1\,.$$

   In particular, there exists a global $\eta \in (0,1)$ such that for all $t \in \mathbb{N}_0$

$$\left\|\frac{\partial h_{t+1}}{\partial h_t}\right\|_2 \leq \eta < 1\,.$$

   Consider further the derivative of the loss at time t w.r.t. the hidden state at time s for $s \leq t$

$$\begin{aligned}\left\|\frac{\partial L_t(\theta)}{\partial h_s}\right\|_2 &= \left\|\frac{\partial L_t(\theta)}{\partial h_t}\cdot\left(\prod_{r=s}^{t-1}\frac{\partial h_{r+1}}{\partial h_r}\right)\right\|_2 \\ &\leq \left(\prod_{r=s}^{t-1}\left\|\frac{\partial h_{r+1}}{\partial h_r}\right\|_2\right)\left\|\frac{\partial L_t(\theta)}{\partial h_t}\right\|_2 \\ &\leq \eta^{t-s}\left\|\frac{\partial L_t(\theta)}{\partial h_t}\right\|_2.\end{aligned}$$

   As $\eta \in (0,1)$, this proves that the gradient norm decreases exponentially fast as a function of the time lag $t-s$.

2. Follows directly as $\frac{1}{c} < \rho(V_h) \leq \|V_h\|_2$ holds and otherwise by the first part of the theorem the gradient would decrease exponentially.

□

Many approaches have been proposed to cure the exploding and vanishing gradients problem (more in [17, Chapter 10.8 - 10.11]). Among those are:

— Add direct connections between hidden states with larger time lags instead of just connecting neighbouring states (*skip connections*).

— Replace the RNN architecture by Temporal Convolutional Networks (TCNs), a sequential analogue to Convolutional Neural Networks (CNNs). For more details on TCNs see [50].

— For the exploding gradient problem, *gradient clipping* as introduced in [41] constitutes a very efficient method often applied in practise. For a gradient ∇_θ, simply include the step

$$\text{if } \|\nabla_\theta\|_2 > b : \nabla_\theta = b\frac{\nabla_\theta}{\|\nabla_\theta\|_2}$$

before updating the parameter θ. By doing so, the direction of the gradient stays the same but the norm is bounded.

At the moment, the most widely used approach to cure the problem are *gated* RNNs. The main idea of gated RNNs is to use *gates* to control the flow of information in a way that gradients do not explode nor vanish. The class of gated RNNs includes the *Long-Short Term Memory (LSTM)* [25] and the *Gated Recurrent Unit (GRU)* [6].

In the following section, we take a closer look at the LSTM as this is currently one of the best performing models in many sequence related tasks like speech recognition and machine translation. In particular, the *Neural Hawkes process* introduced in Section 3.3 will be based on a continuous-time extension of the LSTM. Therefore, a profound understanding of the base model is helpful.

3.2.4 Long Short-Term Memory

The *Long-Short Term Memory (LSTM)* is a Neural Network architecture that was originally proposed by Sepp Hochreiter and Jürgen Schmidhuber in [25] and got adjusted by many contributors to its present form. It was designed the cure the exploding and vanishing gradients problem using *gates* to control the inflow of new inputs, the handling of past information and the output at each time step. Most importantly, the way the LSTM handles this is not predefined. Instead depending on the context, the model learns an optimal behaviour by itself. Furthermore, the LSTM can be trained using simple gradient descent methods like SGD or the Adam optimizer (see Section 3.2.2). Before going more into detail, we first give a mathematical definition.

Definition 3.2.5 (Long-Short Term Memory)
Let $\mathcal{X}, \mathcal{H}, \mathcal{C}, \mathcal{Y}$ be real vector spaces with $\dim(\mathcal{H}) = \dim(\mathcal{C}) = D$, ϕ_y an activation function and $(x_t)_{t=1,\dots,N} \in \mathcal{X}^N$ an input sequence of finite length N. The Long-Short Term Memory (LSTM) is a mapping

$$\begin{aligned} f : \mathcal{X}^N \times \Theta &\to \mathcal{Y}^N \\ ((x_t)_{t=1,\dots,N}, \theta) &\mapsto (y_t)_{t=1,\dots,N} \end{aligned}$$

defined recursively by

$$\begin{aligned} y_t &= f^{(y)}_{\theta_y}(h_t) \\ (h_t, c_t) &= f^{(h,c)}_{\theta_{h,c}}(x_t, h_{t-1}, c_{t-1}) \end{aligned}$$

for $\theta = (\theta_{h,c}, \theta_y) \in \Theta = \Theta_{h,c} \times \Theta_y$ and $h_0, c_0 := 0$. The hidden-to-output mapping is given by

$$\begin{aligned} f^{(y)} : \mathcal{H} \times \Theta_y &\to \mathcal{Y} \\ (h, \theta_y) &\mapsto \phi_y \circ a^{(y)}_{\theta_y}(h) \end{aligned}$$

for an affine function $a_{\theta_y}^{(y)} : h \mapsto V_y \cdot h + b_y$ *and the hidden mapping*

$$f^{(h,c)} : \mathcal{X} \times \mathcal{H} \times \mathcal{C} \times \Theta_{h,c} \to \mathcal{H} \times \mathcal{C}$$

is defined by the recursion

$$\begin{aligned}
f_t &= \sigma\left(W_f x_t + V_f h_{t-1} + b_f\right) \\
i_t &= \sigma(W_i x_t + V_i h_{t-1} + b_i) \\
\tilde{c}_t &= \tanh(W_c x_t + V_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
o_t &= \sigma(W_o x_t + V_o h_{t-1} + b_o) \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}$$

for σ *denoting the sigmoid function and* tanh *the hyperbolic tangent.*

We call Θ *the parameter space containing the weight matrices* W, V *and bias terms* b *and the hidden mapping* $f^{(h,c)}$ *the LSTM cell. Further, denote by* x_t *the input,* f_t *the forget gate,* i_t *the input gate,* c_t *the cell state,* o_t *the output gate,* h_t *the hidden state and* y_t *the output at time* t*.*

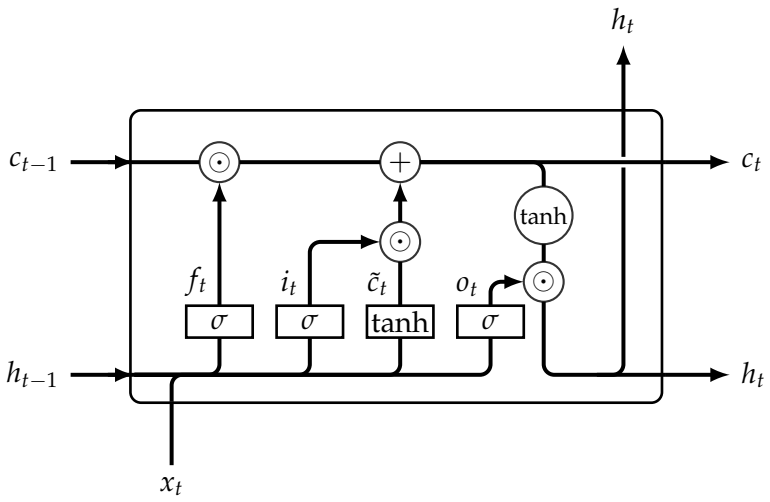


Figure 3.8: LSTM Cell (Layout taken from [40]).

Basically, the setup of an LSTM is the same as for a standard RNN. In each time step t, the input x_t and the past state information, for an LSTM c_{t-1} and h_{t-1}, are put into the hidden mapping. But in the LSTM the hidden mapping is more than just a linear layer followed by an activation function. Instead, it uses a set of gates and activations to determine the flow of information. A visualization of the architecture is given in Figure 3.8 in which rectangular blocks denote layers and nodes point wise operations.

For a more detailed description of the LSTM cell, we stick to the great blog article [40]. What we call *gate* is just a simple sigmoid layer. Depending on the input, a gate will at its extremes be either *open*, if the sigmoid outputs 1, or *closed*, if the sigmoid outputs 0. Hence, if another variable gets multiplied by such a gate, the model lets either pass the variables information (gate open) or stops the flow (gate closed). The LSTM has three such gates: the forget, the input and the output gate.

Moreover contrary to the standard RNN, the LSTM contains two state variables: the cell state and the hidden state. The cell state can be interpreted as a summary statistic of past and current information and the hidden state can be seen as a filtered version of the

cell state capturing the sufficient information for the output.

We will now have a closer look at each of the steps within an LSTM cell.

1. **Forget**
   First, the LSTM determines which part of the past information contained in the cell state c_{t-1} can be deleted. This might be useful when context changes and past information is no longer a valid predictor for future tasks.

   Therefore, the forget gate f_t is calculated on the basis of the past hidden state h_{t-1} and the current input x_t

   $$f_t = \sigma\left(W_f x_t + V_f h_{t-1} + b_f\right) \in (0,1)^D.$$

2. **Input**
   Second, the LSTM identifies new information that should be incorporated in the cell state c_t. Analogue to the forget gate f_t, the input gate i_t gets calculated

   $$i_t = \sigma(W_i x_t + V_i h_{t-1} + b_i) \in (0,1)^D$$

   to decide to which components of the cell state to add the new information $\tilde{c}_t$ given by a tanh layer

   $$\tilde{c}_t = \tanh(W_c x_t + V_c h_{t-1} + b_c) \in (-1,1)^D.$$

3. **Update cell state**
   Combine Step 1 and Step 2 to obtain the new cell state

   $$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \in \mathbb{R}^D$$

   giving a summary statistic of relevant past and present information.

4. **Update hidden state**
   Determine which parts of the cell state are sufficient for the output by calculating the output gate

   $$o_t = \sigma(W_o x_t + V_o h_{t-1} + b_o) \in (0,1)^D$$

   and use this to determine the hidden state h_t as a filtered version of the cell state

   $$h_t = o_t \odot \tanh(c_t) \in (-1,1)^D.$$

Looking at the LSTM one might ask the question why this special structure prevents gradients from vanishing? We want to give a vivid explanation by comparing Figure 3.9 with Figure 3.7 (cf. [18, Chapter 4.1]). Again the darker the colour, the more sensitive the respective node with respect to the input at the first time step. Furthermore, the input gate, the forget gate and the output gate are illustrated below, to the left and above each hidden node with a small circle indicating an open and a small square a closed gate. For simplicity we only consider the case of completely open or closed gates.

Note that the sensitivity stays high as long as the input gate is closed and the forget gate is open, i.e. the cell state stays constant over time. Moreover, the outputs corresponding to open output gates are completely sensitive to the input. Hence, this gated structure enables information to be preserved and gradients to flow for large durations, thereby mitigating the problem of vanishing gradients.

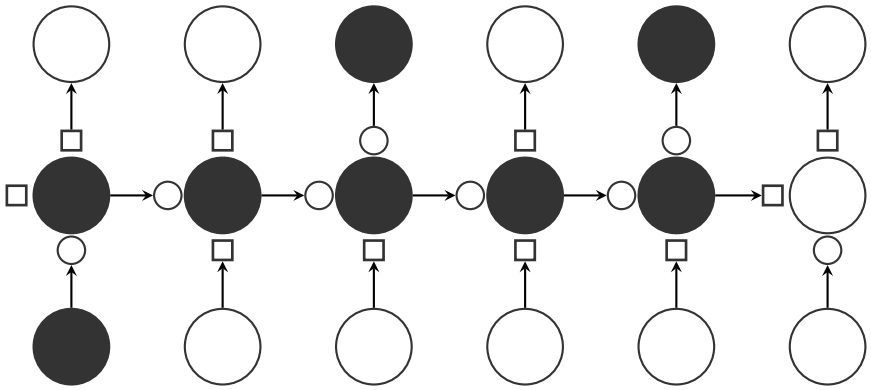


Figure 3.9: Gradient Flow in an LSTM (reproduced from [18, Figure 4.4]). The input gate, the forget gate and the output gate are illustrated below, to the left and above each hidden node. A small circle indicates an open and a small square a closed gate.

3.2.5 Regularization

The aim of Neural Network training consists in optimizing the model such that it performs best on out-of-sample data (*generalization*). Due to the possible high capacity, Neural Network models are prone to *overfitting*, i.e. being trained on some *training set*, these models simply memorize the input data and adopt to random errors present. Of course, when being applied to out-of-sample data, the model will perform worse.

To avoid this, *regularization* is applied to the Neural Network during training which typically slightly increases the loss of the model on the training data, but improves the modelling on out-of-sample data (*test set*). In many deep learning applications, the best performing model has a high capacity and is regularized suitably.

We shortly introduce two regularization techniques used in the Numerical Study in Chapter 4. The sections are a summary of [17, Chapter 7] and [18, Chapter 3.3.2].

Early Stopping

Early stopping is one of the most widely applied regularization techniques, as it effectively prevents overfitting without making changes to the model or the loss function. What is necessary for Early Stopping is that a small subset of the training data, the *validation set*, is used to evaluate the out-of-sample performance of the model. The technique can most easily be explained by looking at Figure 3.10.

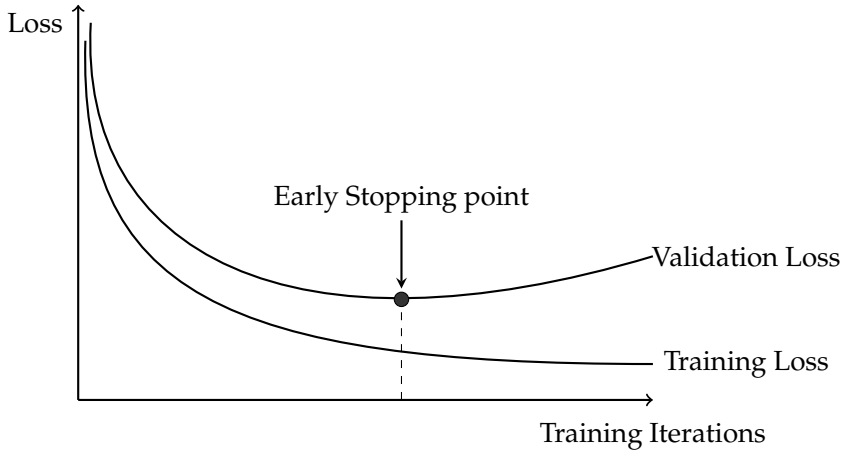


Figure 3.10: Early Stopping

The graphic displays the training and validation loss as a function of the training iterations. While the training loss decreases fairly over time, the validation loss typically exhibits an U-shaped form. At some point in time, the *early stopping point*, the model will start overfitting the training set, which can be detected by an increase of the validation loss. As the model should have a low generalization error, using early stopping one halts training at the early stopping point and returns the corresponding model parameters.

Parameter Norm Penalties

The idea of *Parameter Norm Penalties* is to prevent parameter norms from increasing too much as this typically accompanies overfitting. Therefore, a norm penalty $R(\theta)$ is added to the Loss function

$$\tilde{L}(\theta) = L(\theta) + \alpha R(\theta)$$

with some hyperparameter $\alpha \geq 0$. The larger α, the stronger the regularization. Typically, only the weight matrices (in the following for simplicity just W) are regularized, as a suitable bias term can be crucial for the model to operate on the right scale. The most common Parameter Norm Penalty uses the L^2-norm of the weight matrices

$$\tilde{L}(\theta) = L(\theta) + \frac{\alpha}{2} \|W\|_2^2 \ .$$

The corresponding method is called L^2-*Regularization* and results in weight matrices which values near 0.

In general, there exist many more norm based regularization approaches. For example, the L^1-norm of the weight matrices, i.e. $R(\theta) = \|W\|_1$, is also commonly applied and leads to more sparsity.

3.3 Neural Hawkes Process

In this section, we unite the theory of (marked) TPPs and Recurrent Neural Networks. Parametric models for (marked) TPPs as described in Chapter 2 share the disadvantage that one assumes a fixed and highly simplified form of the conditional intensity function and is thereby often unable to capture the real dynamics of a process. A new approach to more realistically model the possibly complex and non-linear effect of past events on future events is presented in [36].

Motivated by the exceptional modelling abilities of RNNs in many sequence related tasks, a new class of temporal point processes, the so-called *Neural Hawkes process*, is introduced. The Neural Hawkes process defines the conditional intensity function using the hidden-state of a new Recurrent Neural Network architecture. The recurrent architecture used is a specially designed continuous-time version of the LSTM (cf. Definition 3.2.5).

The main rationale supporting this approach is the fact that the hidden-state of an RNN can be interpreted as a summary statistic of the history of a process being able to capture highly non-linear interactions of event times, durations, mark types and mark values. In this way, the Neural Hawkes process does not limit the process' properties in the same way parametric point process models do. To justify this theoretically, we prove some results showing that the Neural Hawkes process can capture effects, already existing parametric models fail for in general.

First, we give a mathematical definition. An illustration of the architecture is given in Figure 3.11.

Definition 3.3.1 (Neural Hawkes Process)
Consider a multivariate TPP $(t_i, m_i)_{i=1,...,n}$ with mark type $m \in \{1, ..., M\}$. Define for each $i = 1, ..., n$ the mark representation $\tilde{m}_i \in \{0, 1\}^M$ by

$$\tilde{m}_i^{(j)} = \begin{cases} 1\,, & j = m_i \\ 0\,, & else \end{cases} \qquad for\ j = 1, ..., M\,.$$

The process $(t_i, m_i)_{i=1,...,n}$ is said to be a Neural Hawkes process, if for each mark type $m \in \{1, ..., M\}$ the conditional intensity function is of the form

$$\lambda_m^*(t) = f_m\left(w_m^T h(t)\right) \quad for \quad t \geq 0\,,$$

where f_m denotes the scaled softplus function $f_m(x) = s_m \cdot \log(1 + e^{x/s_m})$, $w_m \in \mathbb{R}^D$ a weight vector and $h(t) \in (-1, 1)^D$ the hidden state of a D-dimensional Continuous Time Long-Short Term Memory (CTLSTM) given by [2]

$$h(t) = o_i \odot \tanh(c(t)) \quad for \quad t \in (t_{i-1}, t_i]\,.$$

The cell state $c(t) \in \mathbb{R}^D$ and the output gate $o_i \in (0, 1)^D$ are defined using the discrete-time update steps

$$\begin{aligned} i_i &= \sigma(W_i \tilde{m}_{i-1} + V_i h(t_{i-1}) + b_i) \\ \bar{i}_i &= \sigma(W_{\bar{i}} \tilde{m}_{i-1} + V_{\bar{i}} h(t_{i-1}) + b_{\bar{i}}) \\ f_i &= \sigma(W_f \tilde{m}_{i-1} + V_f h(t_{i-1}) + b_f) \\ \bar{f}_i &= \sigma(W_{\bar{f}} \tilde{m}_{i-1} + V_{\bar{f}} h(t_{i-1}) + b_{\bar{f}}) \\ z_i &= \tanh(W_z \tilde{m}_{i-1} + V_z h(t_{i-1}) + b_z) \end{aligned} \qquad \begin{aligned} c_i &= f_i \odot c(t_{i-1}) + i_i \odot z_i \\ \bar{c}_i &= \bar{f}_i \odot \bar{c}_{i-1} + \bar{i}_i \odot z_i \\ \delta_i &= f(W_\delta \tilde{m}_{i-1} + V_\delta h(t_{i-1}) + b_\delta) \\ o_i &= \sigma(W_o \tilde{m}_{i-1} + V_o h(t_{i-1}) + b_o) \end{aligned} \tag{3.1}$$

and

$$c(t) = \bar{c}_i + (c_i - \bar{c}_i) e^{-\delta_i (t - t_{i-1})} \quad for \quad t \in (t_{i-1}, t_i] \tag{CA}$$

where f denotes the un-scaled softplus function $f(x) = \log(1 + e^x)$.

For notational convenience, we call $i_i, \bar{i}_i \in (0, 1)^D$ input gates, $f_i, \bar{f}_i \in (0, 1)^D$ forget gates, $z_i \in (-1, 1)^D$ value update, $c_i \in \mathbb{R}^D$ initial cell state, $\bar{c}_i \in \mathbb{R}^D$ limit cell state and $\delta_i \in (0, \infty)^D$ decay rate.

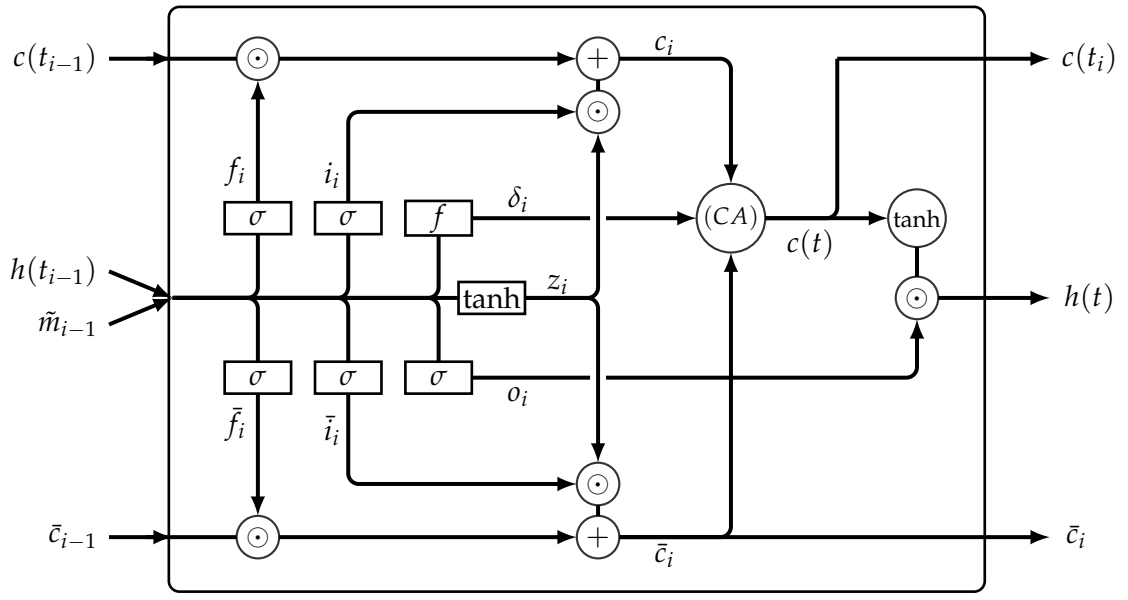


Figure 3.11: Continuous-Time LSTM for $t \in (t_{i-1}, t_i]$.

[2] $\odot$ denotes the Hadamard product.

In the Neural Hawkes process, the conditional intensity function is given by a linear transformation of the hidden state of a CTLSTM followed by the scaled softplus function to ensure positivity (see Figure 3.12). Hence, the conditional intensity function is a deterministic function of the hidden state. As it is always the case, the stochasticity of the process derives from the randomness of the time and the mark of the next event.

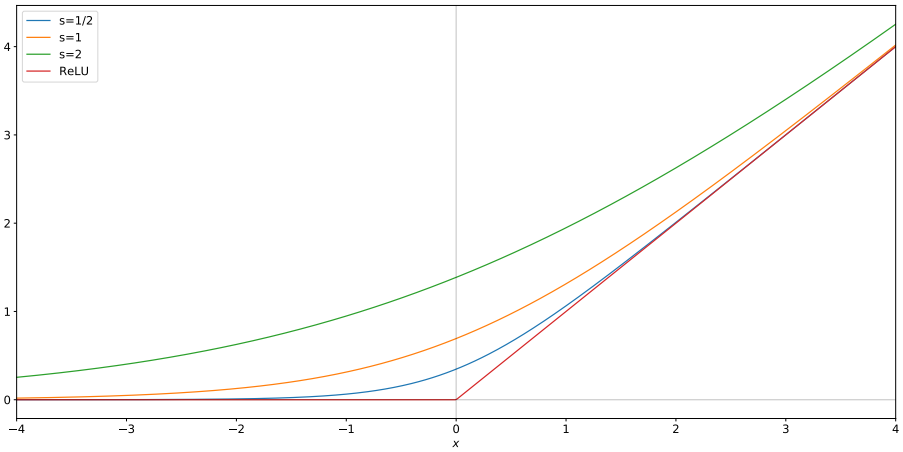


Figure 3.12: Scaled softplus function for different scale parameters s together with the $ReLU$ function which it approaches as $s \to 0$.

The CTLSTM described in Definition 3.3.1 is closely related to the (discrete-time) LSTM presented in Definition 3.2.5, but makes an additional continuity assumption (CA) which allows to model the cell state $c(t)$ in between two occurred events. Note that the presented CTLSTM consists of one hidden layer with hidden dimension D. One could also consider a deep CTLSTM with more than one hidden layer. As this is not necessary for our purposes, we will consider the one layer case.

But how exactly is the hidden state $h(t)$ modelled in the CTLSTM? The basic structure of the CTLSTM is the same as for the (discrete-time) LSTM. An event occurring at time t_{i-1} leads to an update step of the CTLSTM as is given in Equation 3.1. For the input gate, forget gate, value update and output gate a linear layer is applied to the mark representation $\tilde{m}_{i-1}$ and the (decayed) value of the hidden state $h(t_{i-1})$ and passed through a non-linear activation function. In contrast to the LSTM, the update step is not calculated on the basis of the hidden state at the previous event time t_{i-2}. Instead, it uses the value of the hidden state after it has decayed from time t_{i-2} to t_{i-1}. Based on these values, the initial cell state c_i of the interval $(t_{i-1}, t_i]$ is calculated again using the cell state at time t_{i-1}, after the passed duration $t_{i-1} - t_{i-2}$.

What makes the CTLSTM fundamentally different from the LSTM is the fact that the CTLSTM allows for a continuous modelling of the cell - and hidden state. In order to enable this, the continuity assumption (CA) defines that in between two events the cell state $c(t)$ converges with rate δ_i (component-wise) towards the so-called limit cell state $\bar{c}_i$. This is a reasonable assumption, as can be seen in the following Theorem 3.3.6 several behaviours of the conditional intensity function can be modelled in this way. In particular, assumption (CA) captures the time component of each event, which therefore does not have to be embedded in the update step given by Equation 3.1.

As in the LSTM, the hidden state $h(t)$ can then be seen as a filtered version of the cell state $c(t)$, obtained by passing the cell state through a tanh non-linearity and multiplying it component wise with the output gate.

The Neural Hawkes process works by definition for multivariate TPPs, but not for multivariate TPPs with continuous marks, as a modelling of the mark distribution is missing. Therefore, we want to take this idea one step further and introduce the class of *Neural Hawkes processes with continuous marks*.

The main idea is that not only future event times, but also future event marks can be highly dependent on past event times and marks. Hence, the conditional distribution of the mark values is parametrized in the same way, as it is done for the conditional intensity function: by conditioning on the hidden state of a CTLSTM. The rationale for this modelling approach is that if the hidden state of a CTLSTM can be seen as a sufficient summary statistic of the event history, we can condition on this information to get a better fit of the mark distribution.

Note that this nicely extends already existing approaches in the literature. In [8, Example 7.3(c)] the authors propose to parametrize the conditional intensity function and the conditional mark distribution in terms of an underlying Markovian process, which captures past event times and marks. This closely resembles our approach, in which we assume the underlying process to be the hidden state of a CTLSTM. The presented model can also be seen as an extension of the multivariate Hawkes process with continuous marks introduced in Definition 2.2.17.

Definition 3.3.2 (Neural Hawkes Process with continuous marks)
Consider a multivariate TPP with continuous marks $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i=1,\dots,n}$ *where* $m_i^{(V)} \in \mathbb{R}$ *denotes the mark value of an event of type* $m_i^{(T)} \in \{1, \dots, M\}$ *occurring at time* t_i*. Define the mark representation* $\tilde{m}_i \in \mathbb{R}^M$ *by*

$$\tilde{m}_i^{(j)} = \begin{cases} m_i^{(V)} \text{ , } j = m_i^{(T)} \\ 0 \quad \text{ , else} \end{cases} \qquad \text{for} \quad j = 1, \dots, M \,.$$

Assume that the component processes of $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i=1,\dots,n}$ *are given by a Neural Hawkes process according to Definition 3.3.1, calculated on the basis of the mark representation* $\tilde{m}_i$ *and denote by* $h(t)$ *the hidden state of the respective CTLSTM.*

Moreover, suppose that the conditional distribution of the mark value $m_t^{(V)}$ *of an event of type* $m_t^{(T)}$ *occurring at time* t *is given by*

$$f^*\left(m_t^{(V)} | t, m_t^{(T)} = m\right) = f_{\beta_m(t)}\left(m_t^{(V)}\right)$$

for a density $f_{\beta_m(t)}$ *with distributional parameter*

$$\beta_m(t) = \tilde{f}\left(\tilde{w}_m^T h(t)\right),$$

weight vector $\tilde{w}_m \in \mathbb{R}^D$ *and a suitable transfer function* $\tilde{f}$*. Then,* $\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i=1,\dots,n}$ *is called a Neural Hawkes process with continuous marks.*

We originally motivated the introduction of the Neural Hawkes process by the need for a model, which captures complex dependencies of event times, durations and marks in real event sequences. For a theoretical justification, we formulate an appropriate result showing that the Neural Hawkes process indeed offers greater flexibility than the classic Hawkes process. Recall that the Hawkes process makes the following assumptions on the underlying process:

1. The baseline intensity is assumed to be constant.
2. The branching coefficients fulfil $\vartheta > 0$. Hence, effects of excitation, but not of inhibition can be modelled.

3. The conditional intensity function decays exponentially to the baseline intensity.

4. Each event has an additive effect on the intensity function.

We first give a simplifying notation and then prove some Lemmas, which will be useful for the main theorem. We say that there is *no universal order* between two variables x and y, if $x < y$, $x > y$ or $x = y$ can occur.

Lemma 3.3.3
There is no universal order between c_i and $\bar{c}_i$, i.e. depending on the weight matrices and biases for an arbitrary $d \in \{1, ..., D\}$ $c_i^{(d)} < \bar{c}_i^{(d)}$, $c_i^{(d)} > \bar{c}_i^{(d)}$ as well as $c_i^{(d)} = \bar{c}_i^{(d)}$ might be the case.

Proof. Clear, as by construction c_i and $\bar{c}_i$ depend on different weight matrices (e.g. $W_i, W_{\bar{i}}$) and biases (e.g. $b_i, b_{\bar{i}}$). □

Lemma 3.3.4
For each $d \in \{1, ..., D\}$ and $t \in (t_{i-1}, t_i]$ the cell state fulfils

$$c_d(t) \in \left(\min\left(c_i^{(d)}, \bar{c}_i^{(d)}\right), \max\left(c_i^{(d)}, \bar{c}_i^{(d)}\right)\right)$$

if $c_i^{(d)} \neq \bar{c}_i^{(d)}$ and

$$c_d(t) = c_i^{(d)} = \bar{c}_i^{(d)}$$

if $c_i^{(d)} = \bar{c}_i^{(d)}$.

Proof. W.l.o.g.: $D = 1$. For $D > 1$ component-wise. If $c_i = \bar{c}_i$ the statement is clear by definition (CA). Consider the case $\bar{c}_i < c_i$:

$$\bar{c}_i < \bar{c}_i + \underbrace{(c_i - \bar{c}_i)}_{>0} \underbrace{e^{-\delta_i(t-t_{i-1})}}_{\in (0,1)} < \bar{c}_i + (c_i - \bar{c}_i) = c_i \quad \Longrightarrow \quad \bar{c}_i < c(t) < c_i$$

Analogue we get for $\bar{c}_i > c_i$

$$c_i = \bar{c}_i + (c_i - \bar{c}_i) < \bar{c}_i + \underbrace{(c_i - \bar{c}_i)}_{<0} \underbrace{e^{-\delta_i(t-t_{i-1})}}_{\in (0,1)} < \bar{c}_i \quad \Longrightarrow \quad c_i < c(t) < \bar{c}_i$$

□

Lemma 3.3.5
The cell state fulfils the boundary condition

$$c(t_{i-1}^+) := \lim_{t \searrow t_{i-1}} c(t) = c_i$$

and assuming that there is no further event after the event occurring at time t_{i-1}, it is valid

$$\lim_{t \to \infty} c(t) = \bar{c}_i \,.$$

Proof. Both equations follow straightforward from the definition (CA) of the cell state as

$$\lim_{x \searrow 0} e^{-x} = 1 \quad \text{and} \quad \lim_{x \to \infty} e^{-x} = 0.$$

□

We can now give our main theorem proving that the modelling abilities of the Neural Hawkes process extend those of the classic Hawkes process.

Theorem 3.3.6
Consider an M-variate Neural Hawkes process $(t_i, m_i)_{i=1,...,n}$.

1. *On each subinterval* $(t_{i-1}, t_i]$ *the baseline intensity of mark type* $m \in \{1, ..., M\}$ *is given by*
$$\mu^{(m)}_{t_{i-1},t_i} := f_m \left(w_m^T (o_i \odot \tanh(\bar{c}_i)) \right).$$
In particular, the baseline intensity is not constant on $(0, t_n]$.

2. *There is no universal order between* $\lambda^*_m(t_{i-1})$ *and* $\lambda^*_m(t^+_{i-1})$, *i.e. depending on the weight matrices and biases* $\lambda^*_m(t_{i-1}) < \lambda^*_m(t^+_{i-1})$, $\lambda^*_m(t_{i-1}) > \lambda^*_m(t^+_{i-1})$ *as well as* $\lambda_m(t_{i-1}) = \lambda^*_m(t^+_{i-1})$ *can occur. Hence, the Neural Hawkes process can capture effects of excitation and inhibition. Moreover, the level of excitation or inhibition depends on past event times, event durations, mark types and mark values.*

3. *If* $c_i = \bar{c}_i$, *the conditional intensity function* $\lambda^*_m(\cdot)$ *is constant on* $(t_{i-1}, t_i]$ *for each mark type* $m \in \{1, ..., M\}$. *Otherwise, there exist weight matrices and biases such that*
$$\lambda^*_m(t^+_{i-1}) \leq \mu^{(m)}_{t_{i-1},t_i}$$
or
$$\lambda^*_m(t^+_{i-1}) \geq \mu^{(m)}_{t_{i-1},t_i}.$$
Hence after an event has occurred, the conditional intensity function can converge to a limit which lies above or below the current intensity level. In particular, this convergence does not have to be monotone.

4. *In general, the occurrence of an event has no additive effect.*

Proof. All claims can be proven using Definition 3.3.1.

1. Assuming that there is no further event after the event at time t_{i-1}, Lemma 3.3.5 proves
$$\lim_{t\to\infty} c(t) = \bar{c}_i .$$
Due to the continuity of the tanh activation, the limit of the hidden state is given by
$$\lim_{t\to\infty} h(t) = o_i \odot \tanh\left(\lim_{t\to\infty} c(t) \right) = o_i \odot \tanh(\bar{c}_i) .$$
Again using the continuity of the scaled softplus function f_m leads to the baseline intensity
$$\mu^{(m)}_{t_{i-1},t_i} := \lim_{t\to\infty} \lambda^*_m(t) = f_m \left(w_m^T \left(\lim_{t\to\infty} h(t) \right) \right) = f_m \left(w_m^T (o_i \odot \tanh(\bar{c}_i)) \right)$$
for an arbitrary mark type $m \in \{1, ..., M\}$. As $\mu^{(m)}_{t_{i-1},t_i}$ depends on the output gate o_i and the limit cell state $\bar{c}_i$, the baseline intensity is different for various event time intervals $(t_{i-1}, t_i]$.

2. W.l.o.g.: D=1. By Lemma 3.3.5 and the definition of the cell state,
$$c(t^+_{i-1}) := \lim_{t \searrow t_{i-1}} c(t) = c_i = f_i \cdot c(t_{i-1}) + i_i \cdot z_i .$$

Since $f_i, i_i \in (0,1)$ and $z_i \in (-1,1)$, depending on the attained values all three cases $c(t_{i-1}^+) < c(t_{i-1})$, $c(t_{i-1}^+) > c(t_{i-1})$ and $c(t_{i-1}^+) = c(t_{i-1})$ are potential states. As the same holds true for the output gates o_{i-1}, o_i and $w_m \in \mathbb{R}$, this is also the case for

$$w_m \cdot (o_{i-1} \cdot \tanh(c(t_{i-1}))) \lesseqgtr w_m \cdot (o_i \cdot \tanh(c(t_{i-1}^+))) .$$

The scaled softplus function f_m is strictly increasing. Therefore, this relation carries over to the conditional intensity function

$$\lambda_m^*(t_{i-1}) \lesseqgtr \lambda_m^*(t_{i-1}^+) .$$

The instantaneous effect of an event occurring at time t_{i-1} on the intensity λ_m^* is given by

$$\lambda_m^*(t_{i-1}^+) - \lambda_m^*(t_{i-1}) = f(w_m \cdot o_i \cdot \tanh(c(t_{i-1}^+))) - f(w_m \cdot o_{i-1} \cdot \tanh(c(t_{i-1}))) .$$

This term is non-constant and depends on past event times, mark types and mark values.

3. Consider the case that $c_i = \bar{c}_i = c$. By Lemma 3.3.4, the cell state is constant on $(t_{i-1}, t_i]$. Hence by definition, the conditional intensity function is for each $t \in (t_{i-1}, t_i]$ given by

$$\lambda_m^*(t) = f_m\left(w_m^T h(t)\right) = f_m\left(w_m^T (o_i \odot \tanh(c(t)))\right) = f_m\left(w_m^T (o_i \odot \tanh(c))\right),$$

which is constant.
Consider the case $\bar{c}_i > c_i$ and assume w.l.o.g. $D = 1$. As $c(t_{i-1}^+) = c_i < \bar{c}_i$ and the tanh is strictly increasing

$$\tanh(c(t_{i-1}^+)) < \tanh(\bar{c}_i)$$

holds true. Assume $w_m > 0$ (works analogue for $w_m < 0$). Since $o_i \in (0,1)$ we have

$$w_m \cdot o_i \cdot \tanh(c(t_{i-1}^+)) < w_m \cdot o_i \cdot \tanh(\bar{c}_i) .$$

Moreover, as the softplus function f_m is strictly increasing, this relation carries over to the conditional intensity function

$$\lambda_m^*(t_{i-1}^+) = f_m\left(w_m \cdot o_i \cdot \tanh\left(c\left(t_{i-1}^+\right)\right)\right) < f_m(w_m \cdot o_i \cdot \tanh(\bar{c}_i)) = \mu_{t_{i-1},t_i}^{(m)} .$$

This proves the first part of the statement. Regarding the convergence behaviour, let D again be arbitrary. The j-th component of $c(t)$ converges exponentially with rate $\delta_i^{(j)}$ on $(t_{i-1}, t_i]$ towards the baseline intensity. As $o_i \in (0,1)^D$ and the tanh is strictly increasing, the component-wise monotonicity stays valid for the hidden state $h(t)$. Depending on the weight vector $w_m \in \mathbb{R}^D$, the component-wise monotonicity and the corresponding convergence rates, the intensity function $\lambda_m^*(\cdot)$ can exhibit non-monotonous convergence behaviour.

4. The occurrence of a new event leads to an update step of the CTLSTM. Passed through several non-linearities, the effect on the conditional intensity function is non-additive.

□

A simplified illustration of the conditional intensity function of a bivariate Neural Hawkes process is given in Figure 3.13. Events of type 1 instantaneously excite the conditional intensity function of events of type 1 and inhibit events of type 2. The strength of this instantaneous effect, as well as the evolution over time strongly depends on the past of the process and differs from event to event. In particular, the conditional intensity function exhibits non-monotonic behaviour. Analogue statements hold for events of type 2.

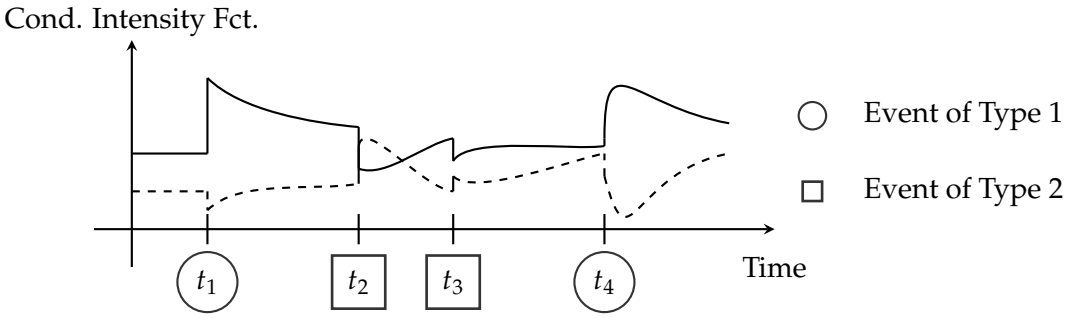


Figure 3.13: Illustration of the conditional intensity functions of a bivariate Neural Hawkes process. The conditional intensity function of events of type 1/2 is printed non-dashed/dashed (reproduced from [36, Figure 1]).

Next, we shortly discuss how the estimation of the model parameters, Goodness-of-Fit Tests and Simulations can be performed in the context of Neural Hawkes processes (with continuous marks). We can simply apply the methods derived in Section 2.3, Section 2.4 and Section 2.5, as the Neural Hawkes process is defined in terms of its conditional intensity function.

Maximum Likelihood Estimation of the model parameters

In order to fit an M-variate Neural Hawkes process (with continous marks) to some data, the process' parameters have to be inferred. Recall that the conditional intensity function uniquely defines the likelihood function (see Section 2.3). By Corollary 2.3.3, the log-likelihood of a Neural Hawkes process with continuous marks is given by

$$\log L = \sum_{i:t_i \leq T} \log\left(\lambda^*_{m_i^{(T)}}(t_i)\right) - \int_0^T \lambda^*(t)\, dt + \sum_{i:t_i \leq T} \log\left(f^*\left(m_i^{(V)} | t_i, m_i^{(T)}\right)\right)$$

for the summed conditional intensity function

$$\lambda^*(t) = \sum_{m=1}^{M} \lambda^*_m(t)$$

and without the third term for a plain Neural Hawkes process. Hence, we can use the negative log-likelihood as loss function for training and thereby obtain the maximum likelihood estimates of the parameters. For the integral term

$$\int_0^T \lambda^*(t)\, dt$$

contained in the log-likelihood, we use a simple Monte Carlo estimator (see Algorithm 8), as it can not be calculated explicitly (cf. [36, Chapter B.2]).

For theoretical justification, let $s \sim \mathrm{Unif}([0,T])$ and consider the random variable $X := T\lambda^*(s)$. Computing the expectation yields

$$\mathbb{E}[X] = \int_{\mathbb{R}} T\, \lambda^*(r)\, \mathbb{1}_{[0,T]}(t)\, \frac{1}{T}\, dt = \int_0^T \lambda^*(t)\, dt\,.$$

Hence, the Monte Carlo estimator is given by

$$\int_0^T \lambda^*(t)\, dt \approx \frac{1}{N}\sum_{i=1}^{N} T\lambda^*(s_i) = \frac{T}{N}\sum_{i=1}^{N}\lambda^*(s_i)$$

for $s_1, ..., s_N \overset{i.i.d}{\sim} \text{Unif}([0,T])$. Note that increasing N leads to a more accurate loss function at the cost of higher computational time.

Algorithm 8: Monte Carlo Integration

Input: End time T, summed intensity function $\lambda^*(t) = \sum_{m=1}^{M}\lambda_m^*(t)$, number of Monte Carlo realizations N
Output: Integral estimator $\hat{\Lambda}(T)$
1 Generate $s_1, ..., s_N \overset{i.i.d}{\sim} \text{Unif}([0,T])$.
2 Calculate Monte Carlo estimator: $\hat{\Lambda}(T) = \frac{T}{N}\sum_{i=1}^{N}\lambda^*(s_i)$

To minimize the loss, any gradient descent approach given in Section 3.2.2 can be applied with gradients obtained by backpropagation through time (see Section 3.2.1).

Goodness-of-Fit Tests

We can simply apply the methods derived in Section 2.4.

Simulation

In order to simulate a Neural Hawkes process (with continuous marks), Ogata's extended modified thinning algorithm as it is derived in Section 2.5.3 can be used (cf. [36, Chapter B.3]). There also exist more efficient methods, but we stick to the one described as it can be applied to a broad class of temporal point processes.

Recall that in the corresponding algorithm, we need to construct an upper bound $\bar{\lambda}_m$ for the conditional intensity function of each mark type $m \in \{1, ..., M\}$. Therefore, let $t \in (t_{i-1}, t_i]$. By definition, the conditional intensity function of the Neural Hawkes process is of the form

$$\lambda_m^*(t) = f_m\left(w_m^T h(t)\right) = f_m\left(\sum_{d=1}^{D} w_{m,d} h_d(t)\right).$$

As by Lemma 3.3.4 the cell state fulfils

$$c_d(t) \in \left(\min\left(c_i^{(d)}, \bar{c}_i^{(d)}\right), \max\left(c_i^{(d)}, \bar{c}_i^{(d)}\right)\right) \quad \text{for } d \in \{1, ..., D\},$$

we obtain an upper bound for each addend by

$$w_{m,d}\cdot h_d(t) = w_{m,d}\cdot o_{i,d}\cdot \tanh\left(c_d(t)\right) \leq \max_{c_d \in \left\{c_i^{(d)}, \bar{c}_i^{(d)}\right\}} w_{m,d}\cdot o_{i,d}\cdot \tanh\left(c_d\right) .$$

Using that the scaled softplus function is strictly increasing, the required upper bound for the conditional intensity function is given by

$$\lambda_m^*(t) \leq f_m\left(\sum_{d=1}^{D}\max_{c_d \in \left\{c_i^{(d)}, \bar{c}_i^{(d)}\right\}} w_{m,d}\cdot o_{i,d}\cdot \tanh\left(c_d\right)\right) =: \bar{\lambda}_m .$$

Chapter 4

Numerical Study

In this chapter, we compare different approaches for modelling extreme returns of a stock index. Therefore, we consider minute-by-minute closing prices $\{s_t\}_{t\in\mathbb{T}_{\text{Price}}}$ of the NASDAQ 100 Index from July 26, 2016 to April 28, 2017 in total 191 trading days (see Figure 4.1; one time unit corresponds to one trading day). Each trading day consists of 390 observations from the opening to the closing of the market. The corresponding dataset is freely available thanks to the authors of [42].
Based on the closing prices, we calculate minute-by-minute log-returns

$$r_t = \log\left(\frac{s_t}{s_{t-1}}\right) .$$

To exclude overnight effects, the first return of each day is calculated using the closing price of the first and second minute. Hence, we obtain a time series of log-returns $\{r_t\}_{t\in\mathbb{T}_{\text{Return}}}$ consisting of 389 observations for each trading day (see Figure 4.2).

Justified by the Extreme Value Theory (cf. [13], [35, Section 7.4]), we make the simplifying assumption that small price changes are caused by endogenous price mechanisms of the market and only extreme negative and positive returns are the result of relevant market information. Therefore, we thin out the log-return process $\{r_t\}_{t\in\mathbb{T}_{\text{Return}}}$ by just considering the returns, which lie below the 2.5% quantile $q_{2.5}$ or above the 97.5% quantile $q_{97.5}$ of the log-return distribution. In this way, events (extreme returns) occur stochastically over time. We have hence transformed the time series $\{r_t\}_{t\in\mathbb{T}_{\text{Return}}}$ into a point process with time index set $\mathbb{T}$ given by

$$\forall t \in \mathbb{T}_{\text{Return}} : t \in \mathbb{T} \iff r_t \leq q_{2.5} \vee r_t \geq q_{97.5} .$$

As mark information, the mark type $m^{(T)} \in \{N, P\}$ indicating extreme negative and positive returns and the mark value $m^{(V)}$ given by the absolute value of the threshold excess are used, i.e.

$$\forall t \in \mathbb{T} : r_t \leq q_{2.5} \Longrightarrow m_t^{(T)} := N,\ m_t^{(V)} := |r_t - q_{2.5}|$$

$$\forall t \in \mathbb{T} : r_t \geq q_{97.5} \Longrightarrow m_t^{(T)} := P,\ m_t^{(V)} := |r_t - q_{97.5}| .$$

All-in-all, we obtain a bivariate temporal point process with continuous marks

$$\left(t, m_t^{(T)}, m_t^{(V)}\right)_{t\in\mathbb{T}}$$

depicted in Figure 4.3, which we aim at modelling using different approaches described in the previous chapters.

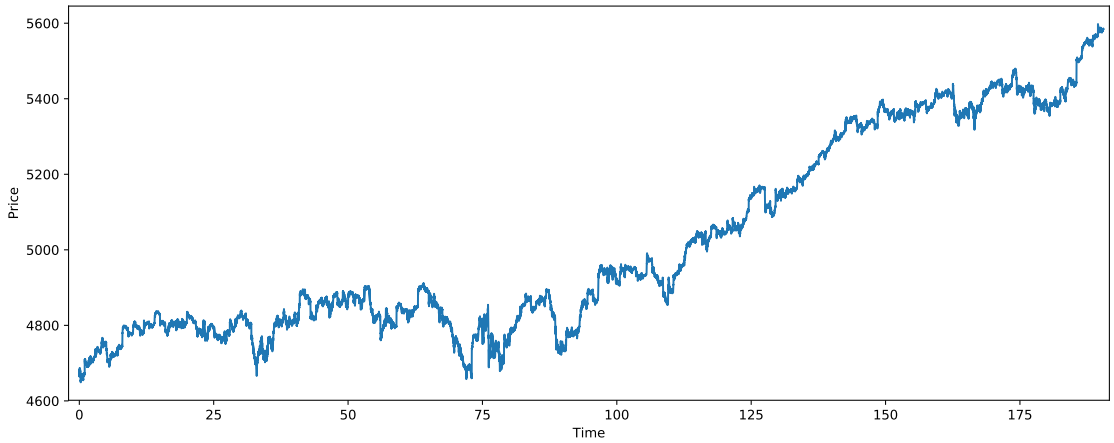


Figure 4.1: Minute-by-minute closing prices of the NASDAQ 100 Index.

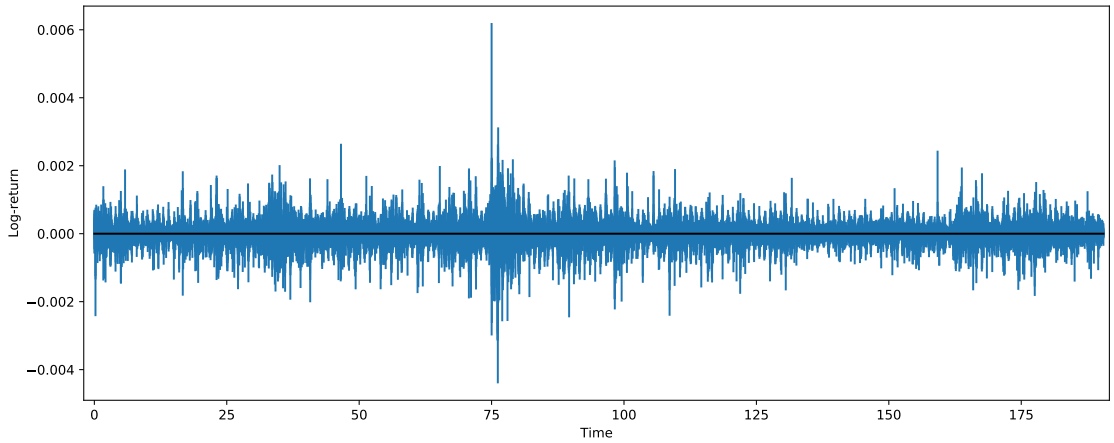
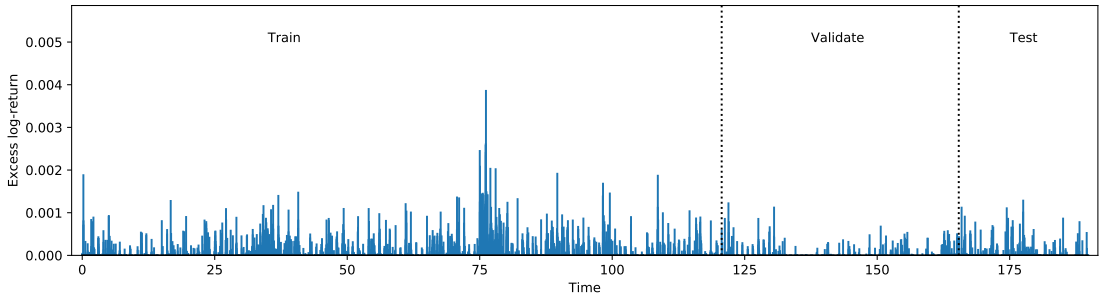
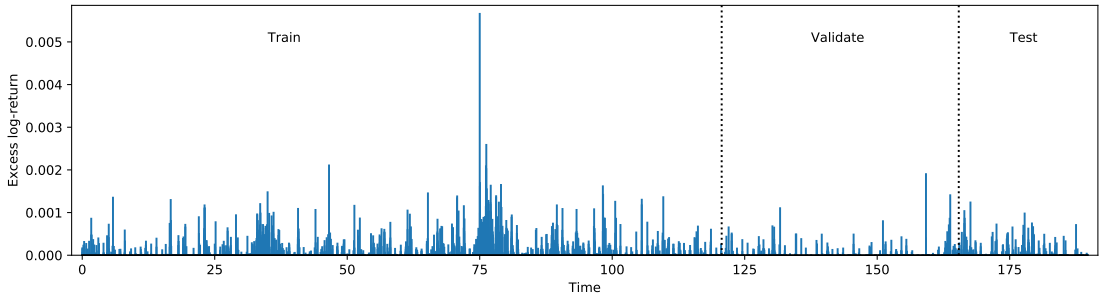


Figure 4.2: Minute-by-minute log-returns of the NASDAQ 100 Index.



(a) Negative returns



(b) Positive returns

Figure 4.3: Bivariate TPP with absolute threshold excesses.

Models

By Theorem 2.2.9, the bivariate TPP with continuous marks is uniquely specified by its conditional intensity function which takes the form

$$\lambda^*\left(t, m_t^{(T)}, m_t^{(V)}\right) = \lambda^*_{m_t^{(T)}}(t) \cdot f^*\left(m_t^{(V)} | t, m_t^{(T)}\right) ,$$

as it is derived in Example 2.2.7. Hence, a complete model for the process at hand has to specify

— the conditional intensity function of the negative and positive return process

$$\lambda_N^*(\cdot),\ \lambda_P^*(\cdot)$$

— the conditional distribution of the mark value $m_t^{(V)}$ of an event of type $m_t^{(T)}$ occurring at time t

$$f^*\left(m_t^{(V)} \mid t, m_t^{(T)} = N\right),\ f^*\left(m_t^{(V)} \mid t, m_t^{(T)} = P\right) .$$

We consider the following three models with an increasing degree of complexity, beginning with the most simple one and ending with a newly derived model first ever to be applied in this setting:

— Bivariate Poisson process with conditionally i.i.d. marks

— Bivariate Hawkes process with continuous marks

— Bivariate Neural Hawkes process with continuous marks

For all three models we assume that the threshold excesses $m^{(V)}$ follow an exponential distribution with scale parameter β. The approaches only differ in the way, the scale parameter is being modelled. The use of the exponential distribution is mainly motivated by the Extreme Value Theory, where it appears as a special case of the *Generalized Pareto Distribution* typically applied for modelling threshold exceedances (cf. [35, Section 7.2]). The parametrization based on the scale parameter β, and not on $\frac{1}{\beta}$ as it is done in the previous chapters, is chosen due to reasons of computational efficiency.

Goodness-of-Fit Tests

We train each model on the first 80% of the available data. For the Bivariate Hawkes process with continuous marks and the Bivariate Neural Hawkes process with continuous marks, we use the next 10% of the data for validation, i.e. we compare fitted models according to different hyperparameter settings on their out-of-sample performance and choose the best. For the Bivariate Poisson process with conditionally i.i.d. marks this is not done, as there are no hyperparameters. To finally evaluate the three best performing models with one for each model class, we use the last 10% of the data as test set and perform the Goodness-of-Fit Tests described in Section 2.4. Hence for each model, we present for the intensity modelling

— the estimated conditional intensity function of the negative and positive return process,

— the QQ-Plot for the $\mathrm{Exp}(1)$-distribution of the inter-event times of the residual process (Test 2),

— the Conditional Uniformity Test of the residual process with 95% confidence bands (Test 4),

— the Autocorrelation function of the inter-event times of the residual process (Test 5),

— the p-values of the Kolmogorov-Smirnov Test for the Exp(1)-distribution of the inter-event times of the residual process (Test 3),

— the log-likelihood of the intensity modelling per event (Test 1),

and for the mark modelling

— the estimated scale parameter of the marks exponential distribution,

— the QQ-Plot for the Unif($[0,1]$)-distribution of the marks of the residual process (Mark Test 1),

— the p-values of the Kolmogorov-Smirnov Test for the Unif($[0,1]$)-distribution of the marks of the residual process (Mark Test 2),

— the log-likelihood of the mark modelling per event (Test 1),

as well as a simulated event sequence (see Introduction to Section 2.5). The corresponding Figures and Tables can be found in Section 4.4 and Section 4.5, respectively. A summary of the main results is given in Section 4.6.

4.1 Bivariate Poisson Process with conditionally i.i.d. marks

Assume a bivariate Poisson process with conditionally i.i.d. marks as described in Definition 2.2.11 with the additional assumption that the mark value distribution is given by an exponential distribution dependent on the mark type. Note that this is the typical *Peaks-over-Threshold (POT) model* often applied in practice (cf. [35, Section 7.4.2]).

Hence, we suppose that the conditional intensity functions of the negative and positive return process are given by

$$\lambda_N^*(\cdot) = \lambda_N,\ \lambda_P^*(\cdot) = \lambda_P$$

for constants $\lambda_N, \lambda_P > 0$. We further assume that for each mark type $m \in \{N, P\}$ the conditional mark value distribution is of the form

$$f^*\left(m_t^{(V)} | t, m_t^{(T)} = m\right) = f\left(m_t^{(V)} | m_t^{(T)} = m\right) = f_{\beta_m}\left(m_t^{(V)}\right)\ ,$$

with f_{β_m} being the density of the Exp$\left(\frac{1}{\beta_m}\right)$-distribution for $\beta_m > 0$.

As is derived in Example 2.3.4, the maximum likelihood estimates of the process' intensities are explicitly given by

$$\hat{\lambda}_N = \frac{N_N(T)}{T},\ \hat{\lambda}_P = \frac{N_P(T)}{T}\ ,$$

where N_N and N_P denote the counting processes of the negative and positive return process and T the final time point. Moreover in this model, the maximum likelihood estimates of the scale parameter of the mark value distributions coincide with the classic maximum likelihood estimates of i.i.d. exponentially distributed data and are hence given by

$$\hat{\beta}_N = \frac{\sum_{t:m_t^{(T)}=N} m_t^{(V)}}{N_N(T)}\ ,\ \hat{\beta}_P = \frac{\sum_{t:m_t^{(T)}=P} m_t^{(V)}}{N_P(T)}\ .$$

In our case, we obtain the estimates

$$\hat{\lambda}_N = 10.062743\ ,\ \hat{\lambda}_P = 10.105074$$

and

$$\hat{\beta}_N = 0.000276\ ,\ \hat{\beta}_P = 0.000251\ .$$

Hence, the intensity of the positive return process is slightly higher than for the negative return process and we expect higher excess returns for the negative than for the positive return process. In particular, the intensities and scale parameters stay constant over time, as it is depicted in Figure 4.4 and Figure 4.5.

The test results suggest that neither the modelling of the intensity nor of the mark distribution captures the real dynamics of the process. The QQ-Plot for the $\text{Exp}(1)$-distribution of the inter-event times of the residual process (Figure 4.6) and the QQ-Plot for the $\text{Unif}([0,1])$-distribution of the marks of the residual process (Figure 4.9) indicate a systematic misfit. This analysis is supported by the Conditional Uniformity Test of the residual process (Figure 4.7) and the KS-Tests for the intensity and the mark modelling (Table 4.1, Table 4.2), which all reject the model for the negative and positive return process on a 5% level.

The Autocorrelation functions of the inter-event times of the residual process (Figure 4.8) show significant autocorrelations on a 5% level. Hence, inter-event times of successive events are correlated, which should not be the case for the correct model. Moreover, the log-likelihood of the intensity and mark modelling is for the negative and positive return process the smallest amongst all the models considered (Table 4.3).

Lastly, a simulated event sequence depicted in Figure 4.10 indicates significant structural differences to the true event sequence given in Figure 4.3. In contrast to the true event sequence, the negative and positive return process are independent of each other. Moreover, the simulated sequence shows no temporal clustering of events. Instead, event times are roughly uniformly distributed over time. In the same way, the marks are much more regular, which is a consequence of the i.i.d. modelling and therefore do not mirror epochs of high and low threshold exceedances as in reality.

4.2 Bivariate Hawkes Process with continuous marks

Consider a Bivariate Hawkes process with continuous marks as proposed in Definition 2.2.17. In this case, the conditional intensity function is for $m \in \{N, P\}$ defined by

$$\lambda_m^*(t) := \mu_m + \sum_{k\in\{N,P\}} \vartheta_{m,k} \cdot h_{m,k}(t)\ ,$$

using the hidden process

$$h_{m,k}(t) := \int_{(0,t)\times(0,\infty)} w_m(t-s) \cdot \xi_k\left(m^{(V)}\right)\ N_k\left(ds \times dm^{(V)}\right)\ .$$

Note that for simplicity, the decay function w_m and the impact function ξ_k only depend on one of the two component processes. Further, assume that the decay function is of the exponential type

$$w_m(t) := \alpha_m \cdot e^{-\alpha_m t}$$

and suppose a normalized, linear impact function given by

$$\xi_k\left(m^{(V)}\right) := \frac{1+\gamma_k \cdot m^{(V)}}{1+\gamma_k \cdot \mathbb{E}\left[m^{(V)}\right]}$$

for $\gamma_k \geq 0$ (cf. [13]). The chosen impact function fulfils Equation 2.3 and is by definition uniquely parametrized.
The conditional distribution of the mark value $m_t^{(V)}$ is given by

$$f^*\left(m_t^{(V)} | t, m_t^{(T)} = m\right) = f_{\beta_m(t)}\left(m_t^{(V)}\right) ,$$

where $f_{\beta_m(t)}$ denotes the density of the exponential distribution with scale parameter

$$\beta_m(t) := a_m + \sum_{k \in \{N,P\}} b_{m,k} \cdot h_{m,k}(t)$$

for $a \in (0,\infty)^2$ and $b \in (0,\infty)^{2\times 2}$. Assuming this parametrization, the Maximum Likelihood estimates can not be calculated explicitly. Instead, a numeric optimizer is used to obtain the following estimates:

$$\hat{\mu} = \begin{bmatrix} \hat{\mu}_N \\ \hat{\mu}_P \end{bmatrix} = \begin{bmatrix} 1.584216 \\ 1.190510 \end{bmatrix}, \quad \hat{\alpha} = \begin{bmatrix} \hat{\alpha}_N \\ \hat{\alpha}_P \end{bmatrix} = \begin{bmatrix} 19.530957 \\ 19.387536 \end{bmatrix}, \quad \hat{\gamma} = \begin{bmatrix} \hat{\gamma}_N \\ \hat{\gamma}_P \end{bmatrix} = \begin{bmatrix} 969.733958 \\ 106.831066 \end{bmatrix}$$

$$\hat{\vartheta} = \begin{bmatrix} \hat{\vartheta}_{N,N} & \hat{\vartheta}_{N,P} \\ \hat{\vartheta}_{P,N} & \hat{\vartheta}_{P,P} \end{bmatrix} = \begin{bmatrix} 0.551931 & 0.290369 \\ 0.471295 & 0.392940 \end{bmatrix}$$

$$\hat{a} = \begin{bmatrix} \hat{a}_N \\ \hat{a}_P \end{bmatrix} = \begin{bmatrix} 0.000191 \\ 0.000161 \end{bmatrix}, \quad \hat{b} = \begin{bmatrix} \hat{b}_{N,N} & \hat{b}_{N,P} \\ \hat{b}_{P,N} & \hat{b}_{P,P} \end{bmatrix} = \begin{bmatrix} 2.386969\text{e-}06 & 1.396692\text{e-}07 \\ 1.228131\text{e-}06 & 1.381718\text{e-}06 \end{bmatrix}$$

The resulting Hawkes process is well-defined as $\rho(\hat{\vartheta}) = 0.850813 < 1$. In particular, as all components of $\hat{\vartheta}$ significantly deviate from zero, all possible types of self- and mutual-excitement are present. Moreover, events of the negative return process have a higher impact on the conditional intensity function than events of the positive return process ($\hat{\vartheta}_{\cdot,N} > \hat{\vartheta}_{\cdot,P}$). The same is true for the threshold excesses. As $\hat{\gamma}_N >> \hat{\gamma}_P$, the effect of threshold excesses of the negative return process on the conditional intensity function is much higher than for the threshold excesses of the positive return process. The estimates $\hat{a}$ and $\hat{b}$ are of small order, as they define a linear transformation of the hidden process h such that the scale parameter β operates on the right scale. In particular as $\hat{b}_{N,N} >> \hat{b}_{N,P} \approx 0$, events of the positive return process have a negligible effect on the scale parameter of the negative return process.

The conditional intensity function and the estimated scale parameter of the negative and positive return process are given in Figure 4.11 and Figure 4.12. The conditional intensity functions are very similar for the negative and positive return process. Moreover, the effects of self- and mutual excitement are strong and clearly visible in both components.

The estimated scale parameter is very similar to the conditional intensity function. This should not come as a surprise as both are defined as a linear transformation of the hidden process h. The transformation leading to the scale parameter ensures the process to operate on the right scale. Most importantly, it shows the desired effect that the expected threshold exceedances are higher in a temporal cluster of events.

According to the results of the Goodness-of-Fit Tests, the Bivariate Hawkes process with continuous marks is a good model for the time and mark component of the data. The QQ-Plots for the Exp(1)-distribution of the inter-event times of the residual process (Figure 4.13) show a very good fit. In particular for the positive return process, it is

nearly perfect. The QQ-Plots for the Unif($[0,1]$)-distribution of the marks of the residual process (Figure 4.16) indicate the modelling to be comparable to the Poisson process for the negative return process and better for the positive process.

The presented statistical tests, i.e. the Conditional Uniformity test (Figure 4.14) as well as the two Kolmogorov-Smirnov Tests (Table 4.1 Table 4.2), accept the modelling of the time and the mark component for the negative and positive return process on a 5% level. Note that the p-values of the KS-Test for the Exp(1)-distribution are extremely high. As for the Poisson model, the ACFs of the inter-event times of the residual process (Figure 4.15) show significant autocorrelation for several lags disagreeing the correctness of the model.

A simulated event sequence with final time $T = 100$ is displayed in Figure 4.17. In contrast to the simulated sequence of the Poisson model (Figure 4.10), the sequence exhibits the patterns present in the real event sequence. One clearly observes the temporal clustering of the events and the dependence between the positive and negative return process. Moreover as desired, the threshold excesses are higher in the temporal clusters.

4.3 Bivariate Neural Hawkes Process with continuous marks

Assume the data can be modelled using a bivariate Neural Hawkes process as given in Definition 3.3.2. Hence, using a continuous-time LSTM we model a hidden-state $(h(t))_{t\in[0,T]}$ over time. Based on this state, we assume that for each mark type $m \in \{N, P\}$ the conditional intensity function is of the form

$$\lambda_m^*(t) = f_m\left(w_m^T h(t)\right) \quad \text{for} \quad t \geq 0\,,$$

where f_m denotes the scaled softplus function

$$f_m(x) = s_m \cdot \log(1 + e^{x/s_m})$$

used to ensure positivity. Further, assume that the conditional distribution of the mark value $m_t^{(V)}$ of an event of type $m_t^{(T)}$ occurring at time t is given by

$$f^*\left(m_t^{(V)} | t, m_t^{(T)} = m\right) = f_{\beta_m(t)}\left(m_t^{(V)}\right)$$

for the exponential distribution $f_{\beta_m(t)}$ with scale parameter

$$\beta_m(t) = \tilde{f}_m\left(\tilde{w}_m^T h(t)\right),$$

where again the scaled softplus function $\tilde{f}_m(x) = \tilde{s}_m \cdot \log\left(1 + e^{x/\tilde{s}_m}\right)$ is used to ensure positivity.

We train the model on subsequences of length 50 obtained by applying a rolling window of length 50 and stride 1 to the training set. Hence, we use

$$\left(t_i, m_i^{(T)}, m_i^{(V)}\right)_{i=n,\ldots,(n+50-1)}$$

for training, with $n = 1, \ldots, n_{\text{Train}} - 50 + 1$ and the size of the training set n_{Train}. Note that in this way, the obtained subsequences are not disjoint. Nevertheless applying a rolling window is useful, as the advantage of having a sufficient amount of data to train on outweighs the disadvantage of a biased estimate of the gradient. The length of the rolling window was chosen as a good compromise between learning long-range dependencies

and a reasonable computational time, but in general the model performs quite robust against this hyperparameter.

As is derived in Section 3.3, we use the negative log-likelihood as loss function for training. We fitted the model for the hidden-sizes $64, 128, 256, 512$. Again, the model is quite robust against this hyperparameter. To avoid overfitting, we apply early stopping (Section 3.2.5) based on the log-likelihood of the model on the validation set. L2-regularization as described in Section 3.2.5 shows no beneficial effect. Instead, it deteriorates the model trough its impact on the loss function.

The estimated conditional intensity functions and the estimated scale parameters of the mark distribution are displayed in Figure 4.18 and Figure 4.19. Both exhibit reasonable patterns and are quite similar to the ones obtained for the bivariate Hawkes process with continuous marks, indicating the assumptions being made by the Hawkes process to be valid in this case.

The conditional intensity functions are fairly similar for the negative and positive return process, as one would expect when looking at the raw data. In particular, each event results in an instantaneous increase of the conditional intensity function. Hence, the process modelled is also self- and mutually exciting and thereby captures the temporal clustering of events.

The modelling of the scale parameter differs slightly. Each event instantaneously increases the scale parameter, but this effect vanishes quickly, as the scale parameter decreases sharply to its baseline value. Hence, only if within a small period of time a high number of events occur, the estimated scale parameter is much higher, indicating higher expected threshold excess. Note that the structural difference in the intensity and scale parameter modelling is possible, as, in contrast to the Hawkes model, both are obtained by non-linear transformations of the hidden state.

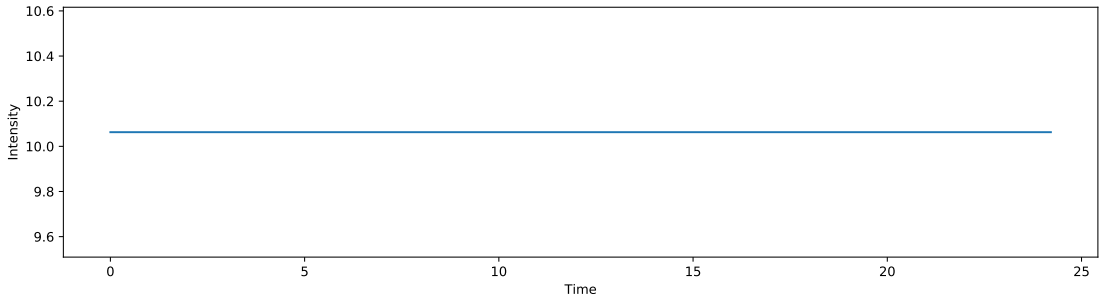
The Goodness-of-Fit Tests indicate the data to be well modelled by the bivariate Neural Hawkes Process with continuous marks. The QQ-Plots for the $\text{Exp}(1)$-distribution (Figure 4.20) show that the inter-event times of the residual process fit the exponential distribution very well. Only the extreme quantiles differ from their theoretical counterparts significantly. The QQ-Plots for the $\text{Unif}([0,1])$-distribution of the marks of the residual process (Figure 4.23) are better, in particular for the positive return process, compared to the ones under the i.i.d assumption, but still not perfect as should not come as a surprise.

The intensity and mark modelling are not rejected on a 5% level by the Conditional Uniformity Test (Figure 4.21) and both KS-Tests (Table 4.1, Table 4.2). Further Table 4.3 shows that the log-likelihood is amongst all the processes considered the highest for the mark modelling and a bit worse compared to the Hawkes process for the intensity modelling. The only result which again rejects the Neural Hawkes model is the ACF showing significant autocorrelation for the positive and negative return process (Figure 4.22)

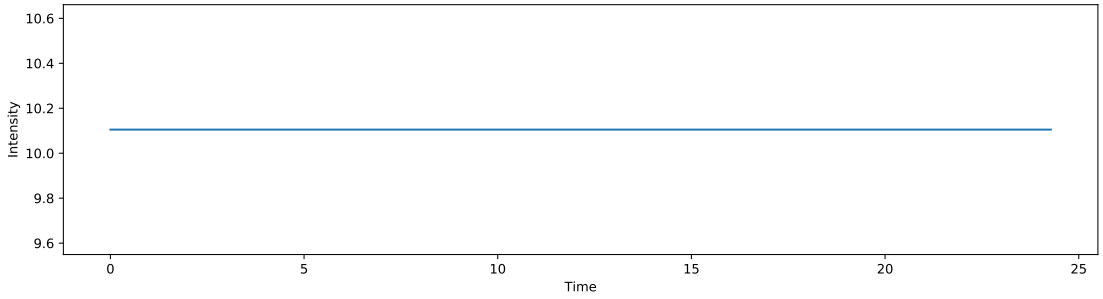
A simulated event sequence of the underlying Neural Hawkes process is depicted in Figure 4.24. It shows all the essential structural properties of the real event sequence including the temporal clustering of events together with increased mean excess returns, as well as high dependence between the component processes. This strongly supports the supposed model.

4.4 Numerical Study - Figures

4.4.1 Bivariate Poisson Process with conditionally i.i.d. marks

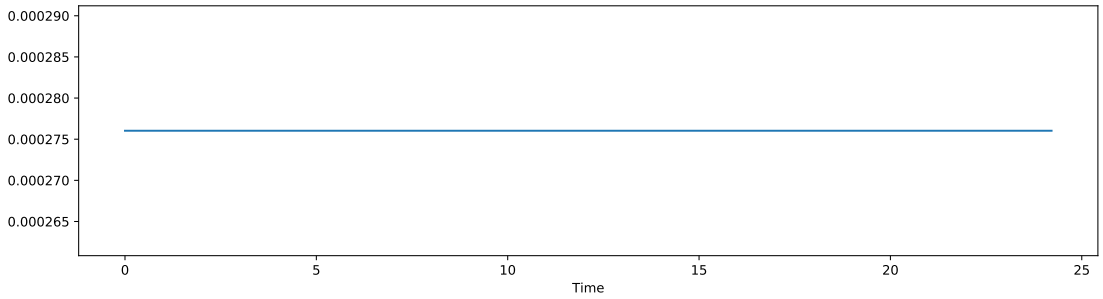


(a) Negative returns

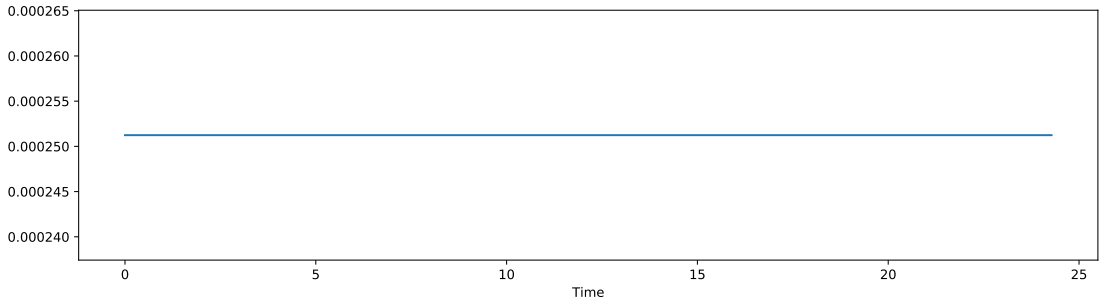


(b) Positive returns

Figure 4.4: Conditional intensity function of the negative and positive return process.

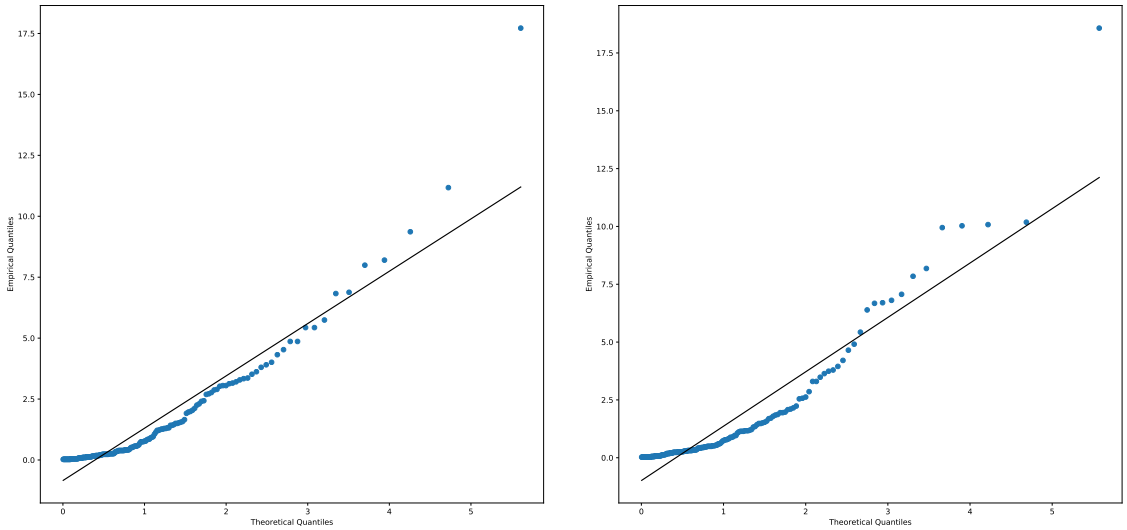


(a) Negative returns



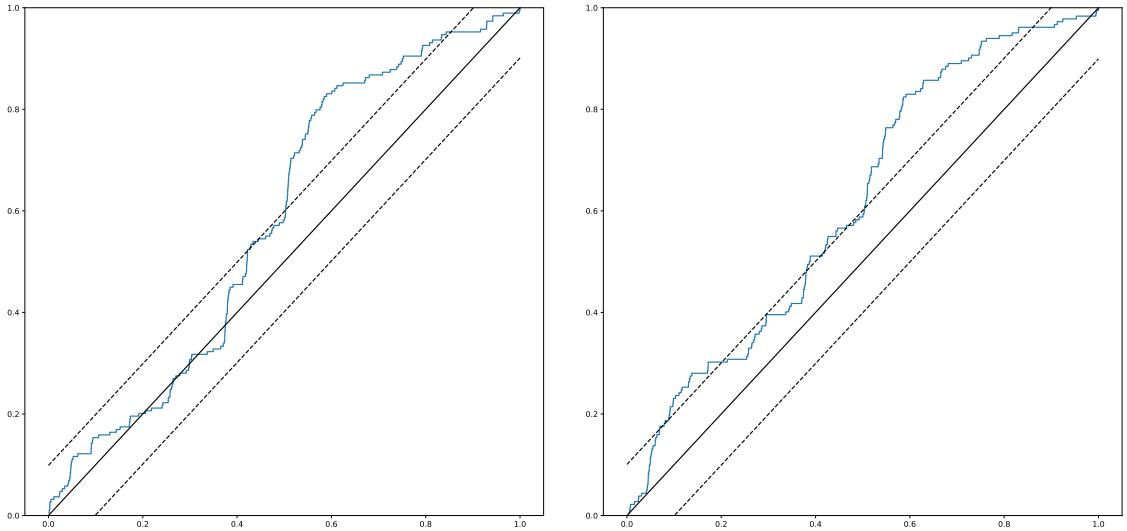
(b) Positive returns

Figure 4.5: Scale parameter of the mark exponential distribution.



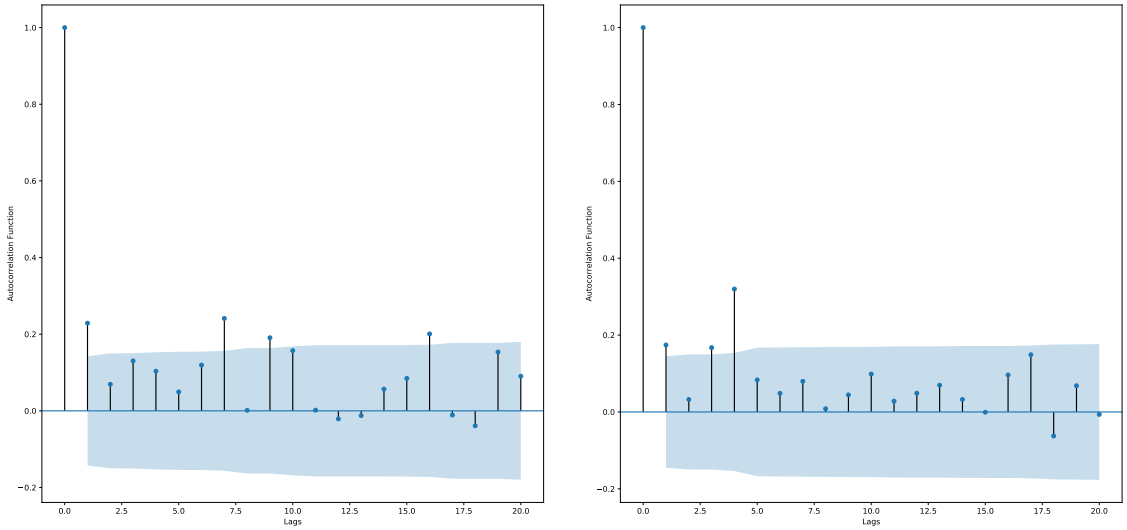
(a) Negative returns (b) Positive returns

Figure 4.6: QQ-Plot Exp(1) - Inter-event times of the residual process.



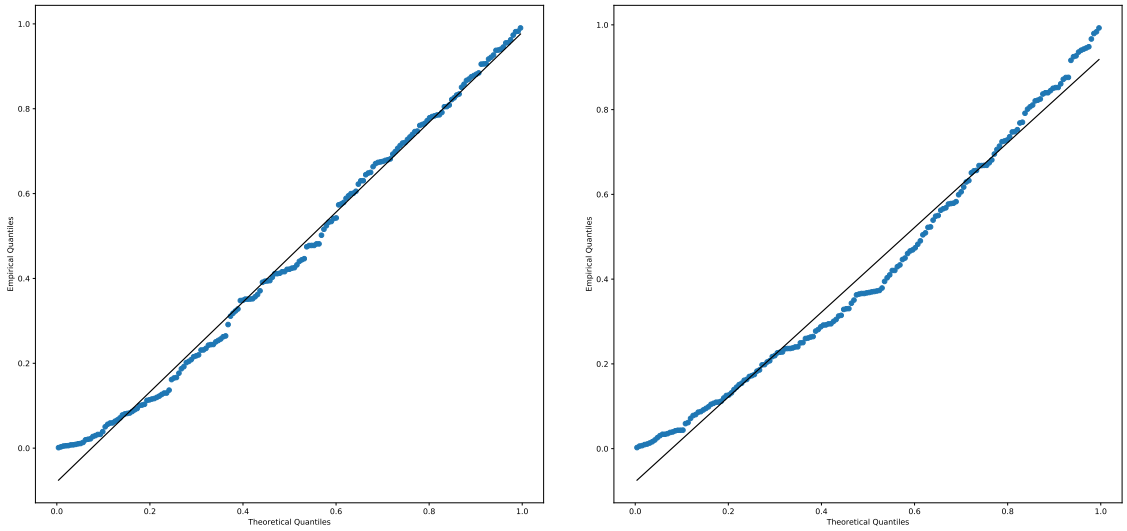
(a) Negative returns (b) Positive returns

Figure 4.7: Conditional Uniformity Test of residual process with 95% confidence bands.



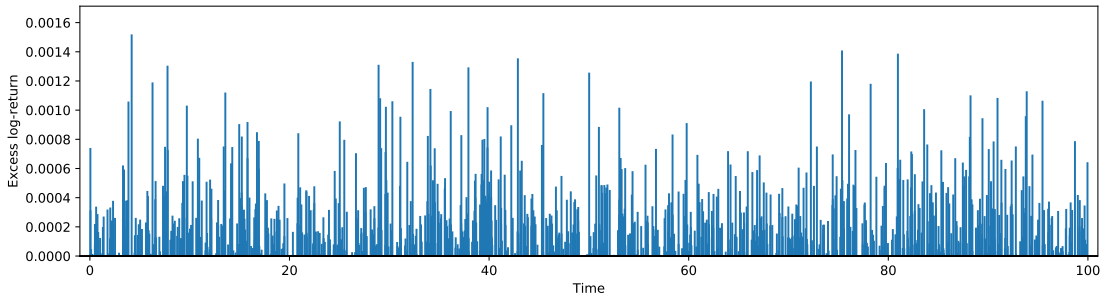
(a) Negative returns (b) Positive returns

Figure 4.8: Autocorrelation function of the inter-event times of the residual process.

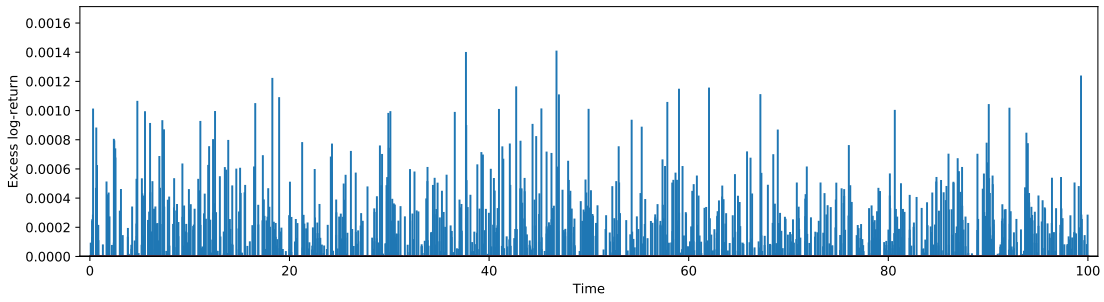


(a) Negative returns (b) Positive returns

Figure 4.9: QQ-Plot Unif$([0,1])$ - Marks of the residual process.



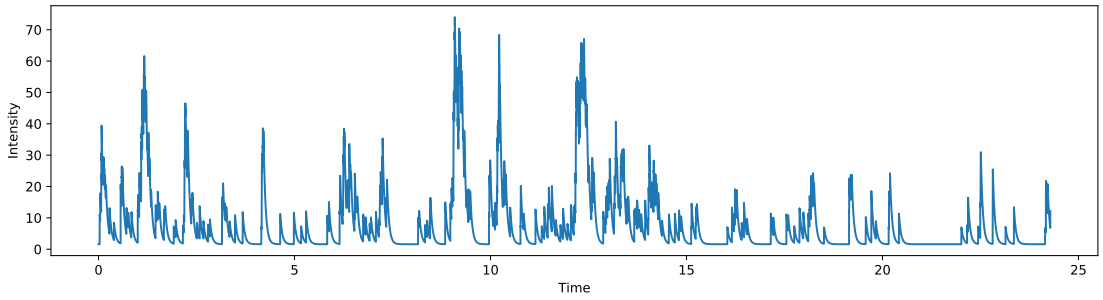
(a) Negative returns



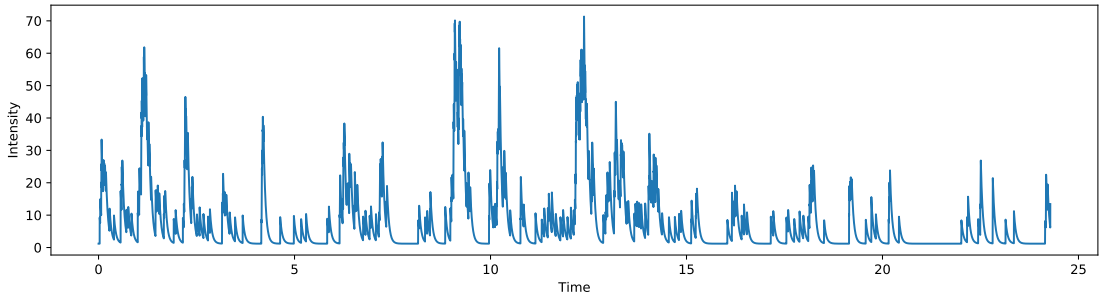
(b) Positive returns

Figure 4.10: Simulated event sequence with final time $T = 100$.

4.4.2 Bivariate Hawkes Process with continuous marks

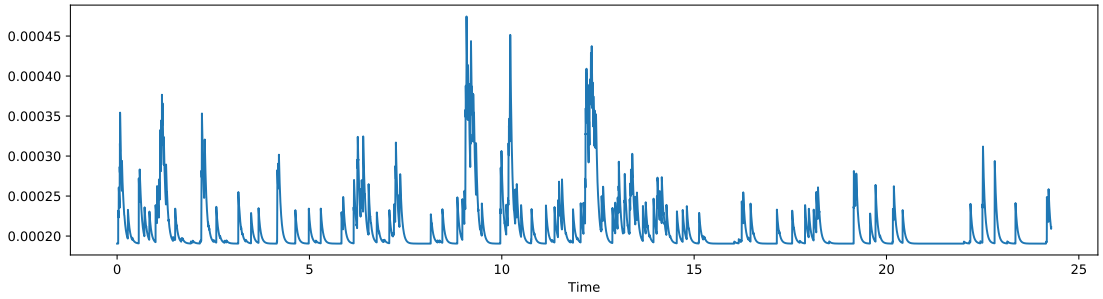


(a) Negative returns

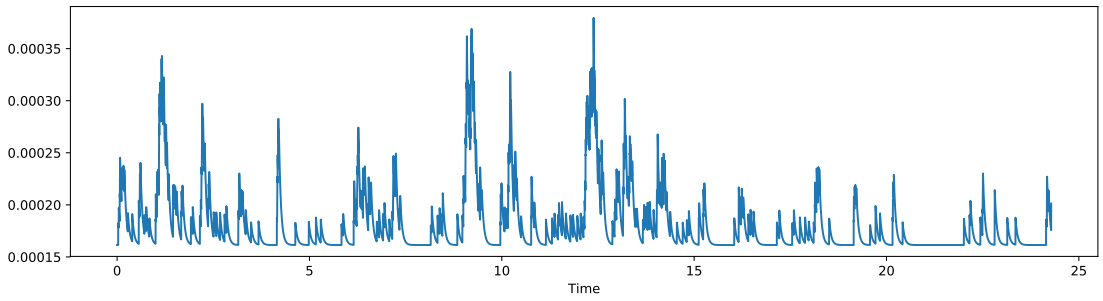


(b) Positive returns

Figure 4.11: Conditional intensity functions of the negative and positive return process.

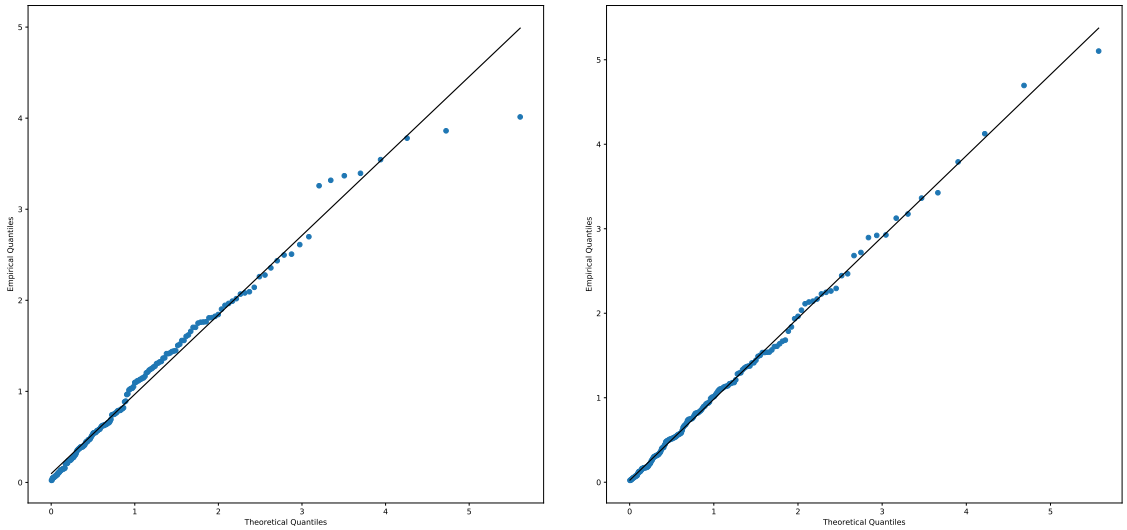


(a) Negative returns



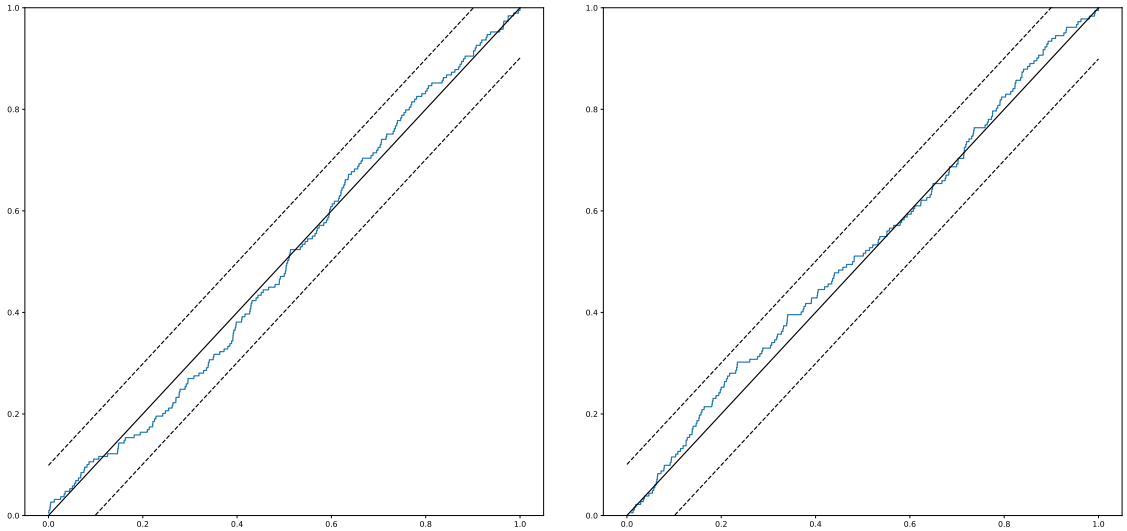
(b) Positive returns

Figure 4.12: Scale parameter of the mark exponential distribution.



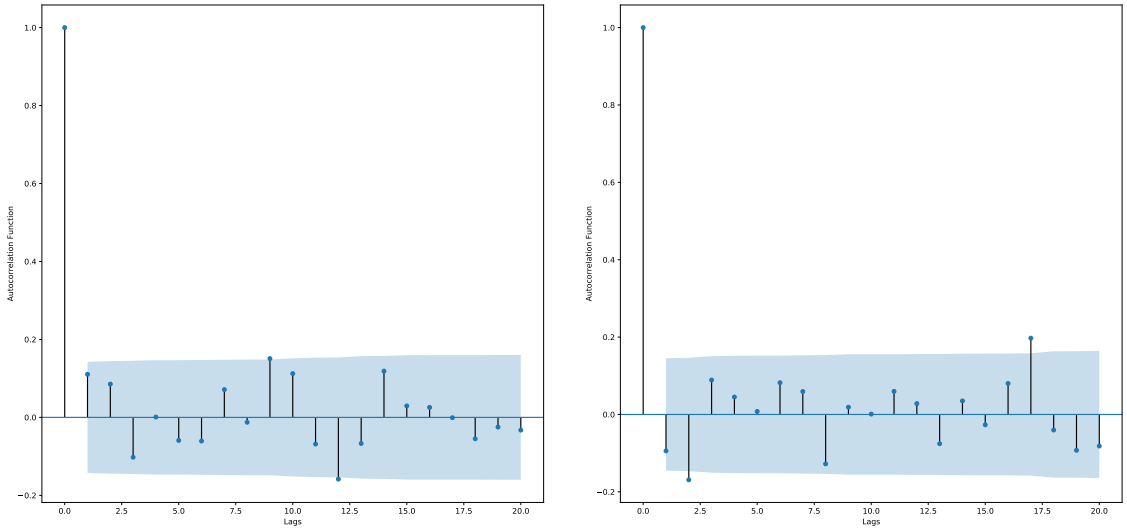
(a) Negative returns (b) Positive returns

Figure 4.13: QQ-Plot Exp(1) - Inter-event times of the residual process.



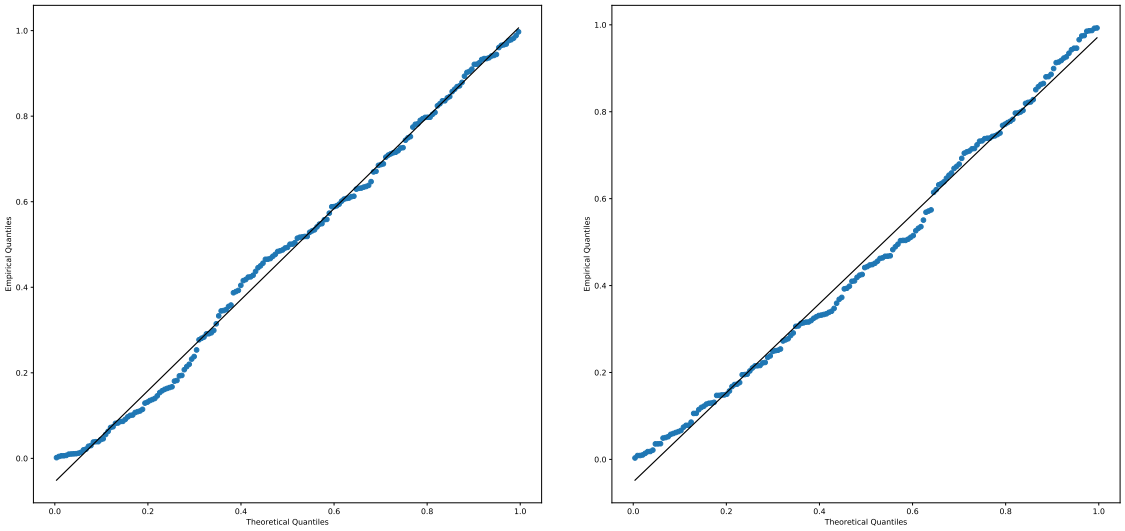
(a) Negative returns (b) Positive returns

Figure 4.14: Conditional Uniformity Test of residual process with 95% confidence bands.



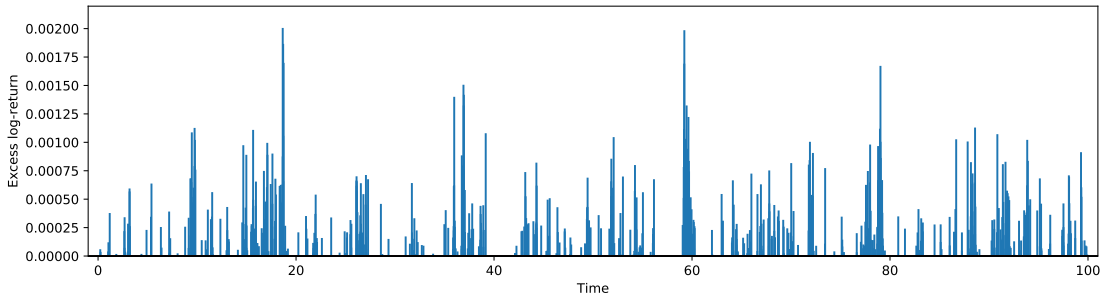
(a) Negative returns (b) Positive returns

Figure 4.15: Autocorrelation function of the inter-event times of the residual process.

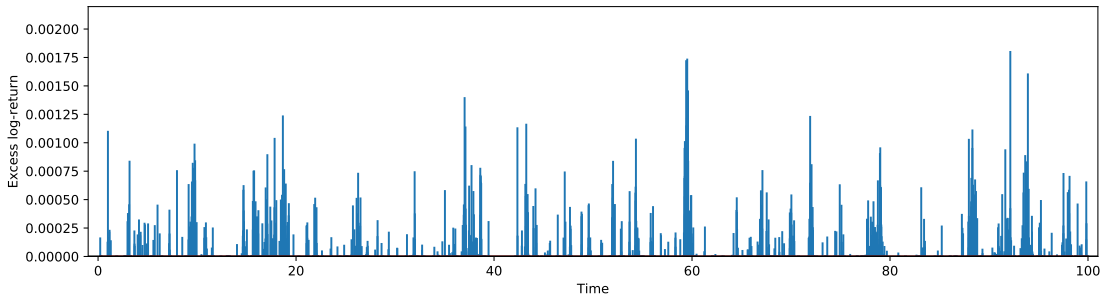


(a) Negative returns (b) Positive returns

Figure 4.16: QQ-Plot Unif$([0,1])$ - Marks of the residual process.



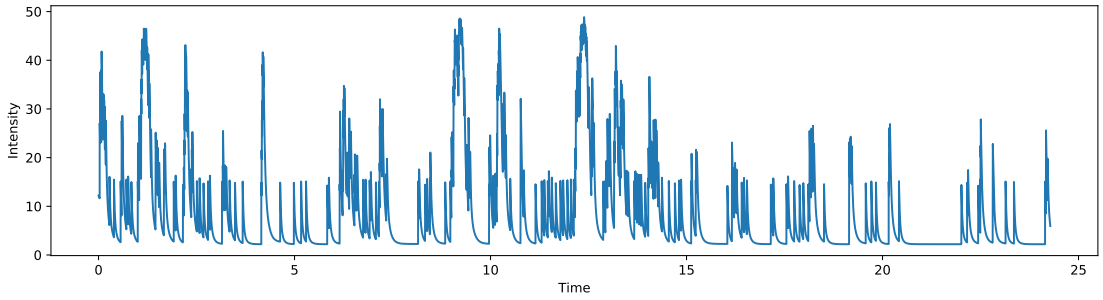
(a) Negative returns



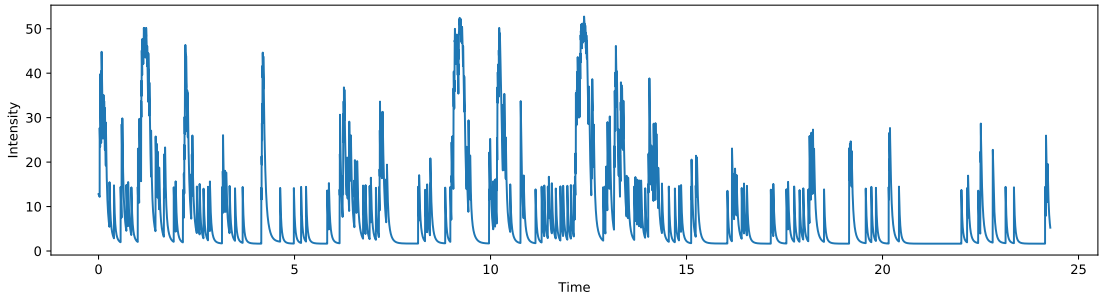
(b) Positive returns

Figure 4.17: Simulated event sequence with final time $T = 100$.

4.4.3 Bivariate Neural Hawkes Process with continuous marks

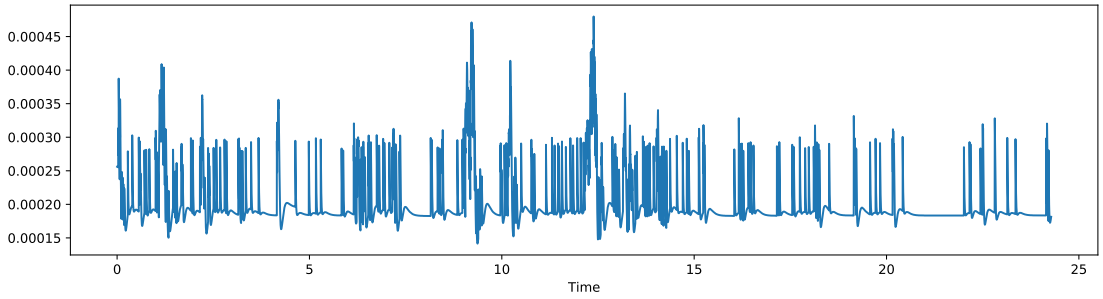


(a) Negative returns

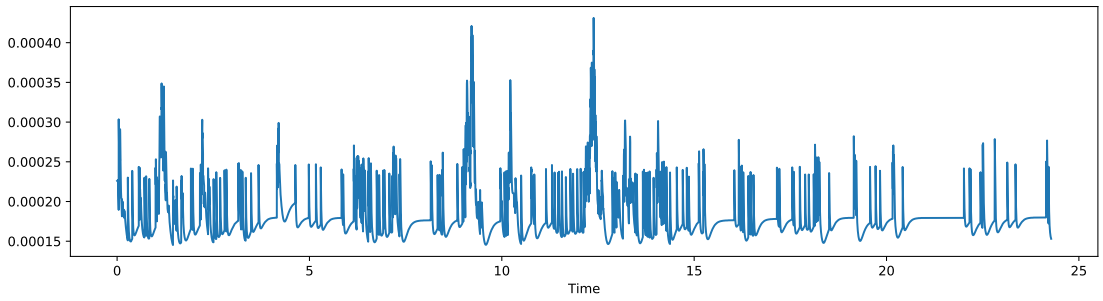


(b) Positive returns

Figure 4.18: Conditional intensity functions of the negative and positive return process.

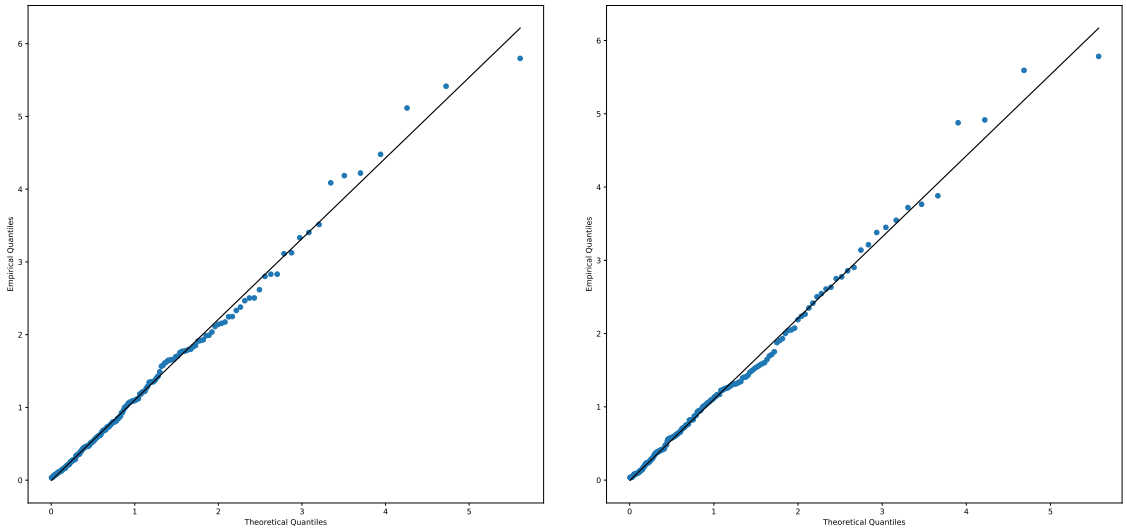


(a) Negative returns



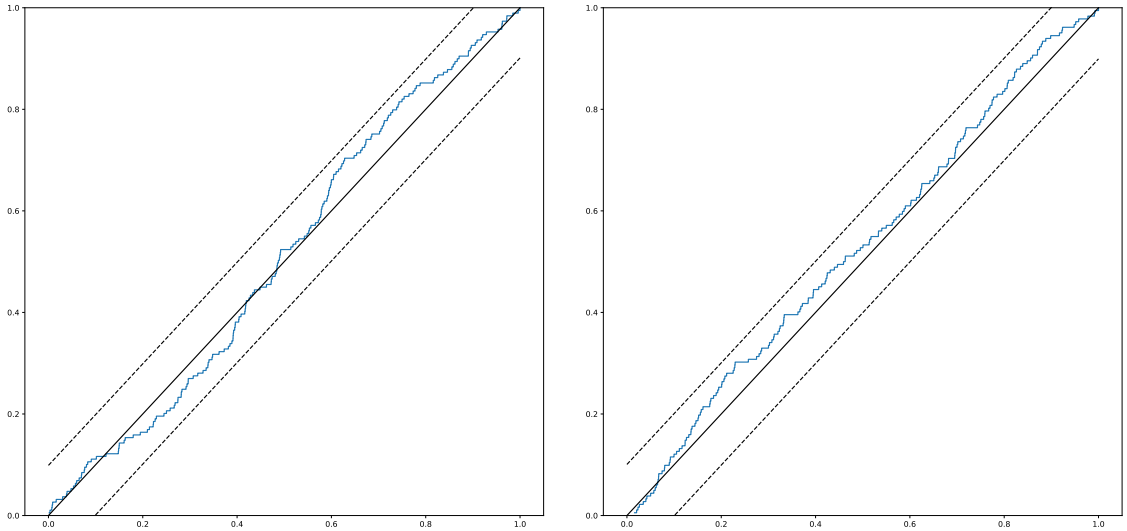
(b) Positive returns

Figure 4.19: Scale parameter of the mark exponential distribution.



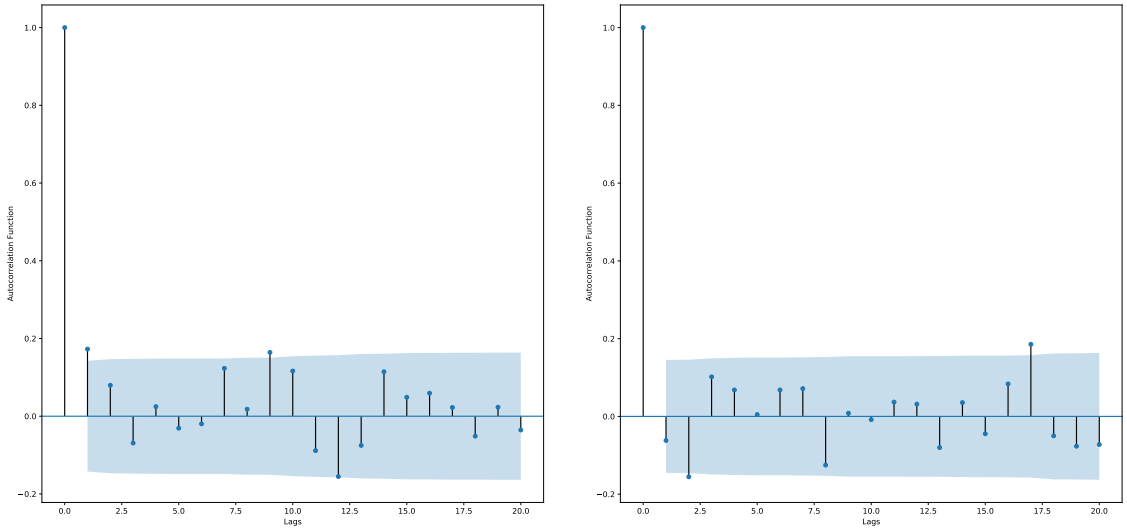
(a) Negative returns (b) Positive returns

Figure 4.20: QQ-Plot Exp(1) - Inter-event times of the residual process.



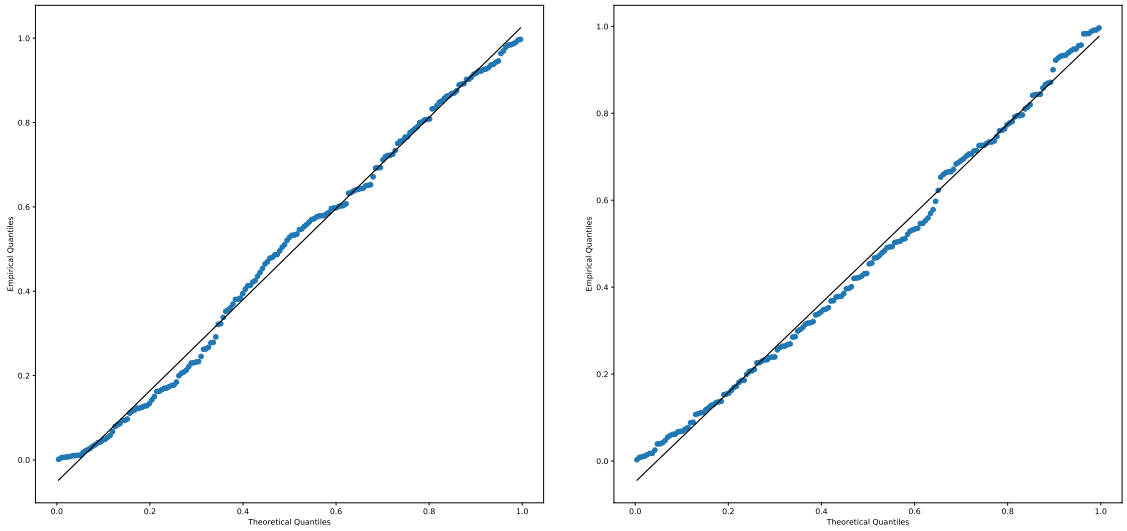
(a) Negative returns (b) Positive returns

Figure 4.21: Conditional Uniformity Test of residual process with 95% confidence bands.



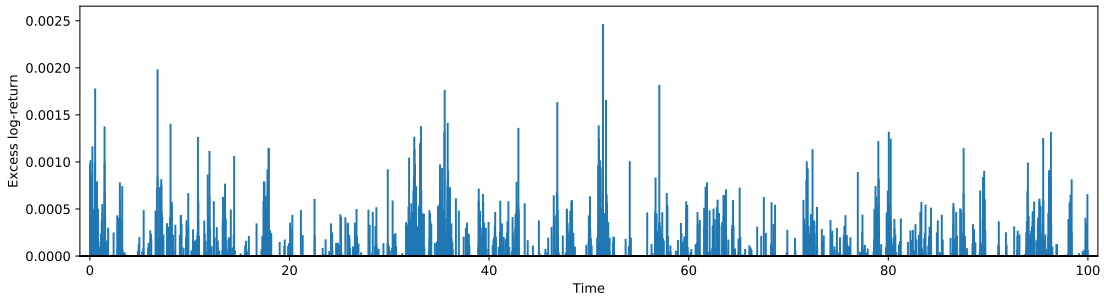
(a) Negative returns (b) Positive returns

Figure 4.22: Autocorrelation function of the inter-event times of the residual process.

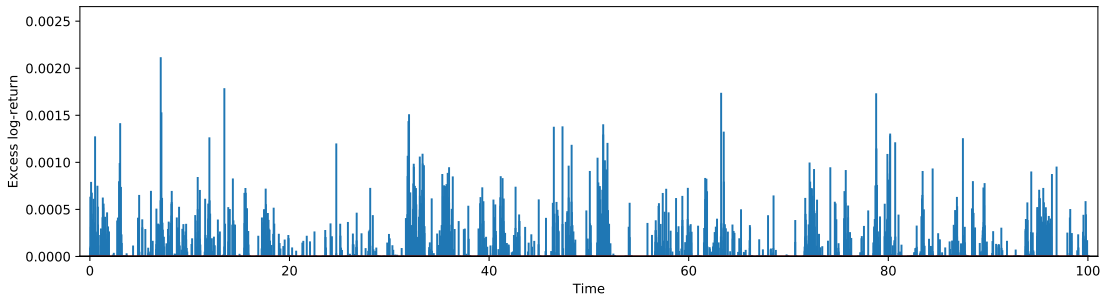


(a) Negative returns (b) Positive returns

Figure 4.23: QQ-Plot Unif$([0,1])$ - Marks of the residual process.



(a) Negative returns



(b) Positive returns

Figure 4.24: Simulated event sequence with final time $T = 100$.

4.5 Numerical Study - Tables

	Bivariate Poisson process with conditionally i.i.d marks	Bivariate Hawkes process with continuous marks	Bivariate Neural Hawkes process with continuous marks
Negative return process	7.056861e-10	0.833719	0.587654
Positive return process	5.155624e-07	0.928023	0.525080

Table 4.1: p-values of the Kolmogorov-Smirnov Test for the $\mathrm{Exp}(1)$-distribution of the inter-event times of the residual process.

	Bivariate Poisson process with conditionally i.i.d marks	Bivariate Hawkes process with continuous marks	Bivariate Neural Hawkes process with continuous marks
Negative return process	0.022227	0.110275	0.207601
Positive return process	0.000305	0.102878	0.254838

Table 4.2: p-values of the Kolmogorov-Smirnov Test for the $\mathrm{Unif}([0,1])$-distribution of the marks of the residual process.

	Bivariate Poisson process with conditionally i.i.d marks	Bivariate Hawkes process with continuous marks	Bivariate Neural Hawkes process with continuous marks
log-likelihood (intensity)	0.991010	1.481255	1.474743
log-likelihood (mark)	7.411477	7.452119	7.456583

Table 4.3: Log-likelihood per event.

4.6 Summary

In this section, we compared the multivariate Neural Hawkes process with continuous marks to the multivariate Poisson process with conditionally i.i.d. marks and the multivariate Hawkes process with continuous marks regarding the task of modelling minutely excess returns (events) of the NASDAQ 100 as an example of a financial point process. The main empirical properties of the considered dataset are:

Prop. 1: Events do not occur uniformly over time. Instead, there are periods with a high number of excess returns (due to relevant market information) called *temporal clusters*, as well as periods of *quietness*, in which most of the returns are not extreme in the sense considered (due to the lack of relevant market information).

Prop. 2: The size of the excess return is irregular and depends on the time component of the process. Temporal clusters are typically accompanied by high excess returns and periods of quietness by small excesses.

Prop. 3: The negative and positive return process are highly dependent. Temporal clusters and periods of quietness occur simultaneously in both processes.

The modelling results of the three different processes can be summarized as follows:

— **Bivariate Poisson process with conditionally i.i.d. marks:**
The Poisson model is the most simple, meaningful MTPP, as the conditional intensity function and the conditional mark distribution are assumed to be history and time independent. It performed the worst among all the models considered and was rejected by all Goodness-of-Fit tests. In particular, it yield by far the lowest log-likelihood for the time and mark modelling.

This should not come as a surprise, as the structure imposed by the model is way to simplistic and does not stand in line with the empirical properties of the dataset at hand. The Poisson model assumes the inter-event times to be i.i.d. (Contradiction to Prop. 1), the size of the excess returns to be i.i.d. (Contradiction to Prop. 2) and the negative and positive return process to be independent (Contradiction to Prop. 3). The Poisson model is hence for comparison only and motivates the more complex models.

— **Bivariate Hawkes process with continuous marks:**
The Hawkes model is at the moment the state-of-the art parametric approach for modelling self - and mutually exciting point processes. The conditional intensity function and the conditional mark distribution are designed to capture the effect of the history of the process and the actual time, but in a fixed, simplified form. All considered Goodness-of-Fit Tests accept the model and hence indicate it to yield a reasonable fit for the data. Further, the Hawkes model achieves the highest log-likelihood for the time component and the second highest log-likelihood for the mark modelling amongst all models considered.

Note that this can be explained by looking at the imposed structure of the Hawkes model and the empirical properties of the dataset. By definition of the conditional intensity function, each event excites the occurrence of future events in the positive and negative return process, which allows the model to capture temporal clusters and explains the high dependence between the component processes (Prop. 1 & 3). Further, the conditional mark distribution is designed to depend on the history too, which allows the modelling of different mean excesses in temporal clusters and periods of quietness (Prop. 2).

Nevertheless, one should keep in mind that if the structural assumptions of the Hawkes model do not mirror the reality, e.g. if events inhibit the occurrence of future events or if the dependence between past and future events is more complex and highly non-linear, the model bias will be high and the real dynamics will not be captured adequately. In this case, one can either manually adjust the Hawkes model in the hope of finding a reasonable form or switch to a *non-parametric* model (e.g. the Neural Hawkes model), which allows for more flexibility and directly adapts to the data.

— **Bivariate Neural Hawkes process with continuous marks:**
The Neural Hawkes model is a novel approach, which uses a continuous-time Recurrent Neural Network architecture to capture the dependence between past and future events. In contrast to the Hawkes model, the construction of the Neural Hawkes model allows for high flexibility, thereby reduces the model bias and enables it to directly learn the real dynamics from the data. Therefore, the model is called non-parametric.

The results suggest the Neural Hawkes to be reasonable for the modelling of the dataset, as it is accepted by all Goodness-of-Fit tests. Furthermore, the model yields

a comparable, but slightly worse log-likelihood for the time modelling than the Hawkes model and the best log-likelihood for the mark modelling. The Neural Hawkes model learned from the data that each event increases the occurrence of future events in the positive and negative return process (Prop. 1 & Prop. 3). Further, it captured a high mean excess return in temporal clusters and a low excess return in periods of quietness (Prop. 2).

Recall that different to the Hawkes model, this behaviour is not predefined. Instead, it is learned from the data itself. In fact for the example considered here, this supports the modelling assumptions of the Hawkes model. Nevertheless when dealing with data with more complex or diffuse dynamics such that the simplifying assumptions, e.g. of the Hawkes model, are no longer valid, the Neural Hawkes model is expected to benefit more from its general formulation and to outperform the classic, parametric models. In particular, the performance of the Neural Hawkes model has shown to be comparable or even superior to the 'best' parametric models and therefore represents a model to bear in mind.

Chapter 5

Conclusion and Outlook

In this thesis we have shown that Recurrent Neural Network architectures can successfully be applied in the context of financial point processes. Besides time series, point processes form a second fundamental class of stochastic processes in finance, thereby allowing the presented models to be applied on a wide range of applications. In particular, if the assumptions imposed by typical parametric models like the Poisson or Hawkes process do not capture the real data-generating mechanism, Neural Network based point process models yield a promising alternative. In our view, future work should concentrate on two main tasks.

Firstly, Neural Network based point processes should be extended to allow the modelling of more general types of processes, as such with two or more events at the same time. Further, different model setups should be considered. This can include the underlying network architecture, e.g. not the LSTM as for the Neural Hawkes process, or novel approaches to modify the baseline model to be continuously defined in time.

Secondly, high-frequency data represents another interesting area of finance for these models to being applied. When working on the finest time scales, changes in the bid/ask price or more general in the order book occur by nature randomly in time and can therefore be modelled as a point process. Hence, Neural Network based processes should be tested on their suitability as models for market microstructure. In particular, modelling the complex interactions between different types of orders in the order book appears promising.

Bibliography

[1] E. Bacry, I. Mastromatteo, and J.-F. Muzy. Hawkes processes in finance. *arXiv:1502.04592*, 2015.

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[3] H. Buhler, L. Gonon, J. Teichmann, and B. Wood. Deep Hedging. *Quantitative Finance*, pages 1–21, 02 2019.

[4] Y. Chen. Multivariate Hawkes Processes and Their Simulations. Florida State University, 2016.

[5] Y. Chen. Thinning Algorithms for Simulating Point Processes. Florida State University, 2016.

[6] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

[7] E. K. Chong and S. H. Zak. *An Introduction to Optimization*, volume 4. Wiley, 2013.

[8] D. J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes. Vol. I.* Probability and its Applications (New York). Springer, second edition, 2003.

[9] A. De, U. Upadhyay, and M. Gomez-Rodriguez. Notes for Human-Centered ML: Temporal Point Processes. Saarland University, 2018.

[10] L. Deng and D. Yu. Deep Learning: Methods and Applications. *Foundations and Trends in Signal Processing*, 7(3-4):197–387, 2014.

[11] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song. 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, pages 1555–1564, 2016.

[12] N. Du, M. Farajtabar, A. Ahmed, A. J. Smola, and L. Song. Dirichlet-Hawkes Processes with Applications to Clustering Continuous-Time Document Streams. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 219–228. ACM, 2015.

[13] P. Embrechts, T. Liniger, and L. Lin. Multivariate Hawkes processes: an application to financial data. 48A:367–378, 08 2011.

[14] E. Errais, K. Giesecke, and L. Goldberg. Affine Point Processes and Portfolio Credit Risk. *SIAM J. Financial Math.*, 1:642–665, 06 2010.

[15] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 1968.

[16] R. Gallager. Discrete Stochastic Processes: Chapter 2 - Poisson Processes. MIT, 2011.

[17] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[18] A. Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385. 2012.

[19] A. Graves, A. Mohamed, and G. E. Hinton. Speech Recognition with Deep Recurrent Neural Networks. *CoRR*, abs/1303.5778, 2013.

[20] G. Grimmett and D. Stirzaker. *Probability and random processes*. Oxford university press, 2001.

[21] B. Hammer. On the Approximation Capability of Recurrent Neural Networks. *Neurocomputing*, 31, 10 2001.

[22] A. G. Hawkes. Point Spectra of Some Mutually Exciting Point Processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 33(3):438–443, 1971.

[23] A. G. Hawkes. Spectra of Some Self-Exciting and Mutually Exciting Point Processes. *Biometrika*, 58(1):83–90, 1971.

[24] A. G. Hawkes and D. Oakes. A cluster process representation of a self-exciting process. *Journal of Applied Probability*, 11(3):493–503, 1974.

[25] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735-1780, 1997.

[26] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257, 1991.

[27] M. Jacobsen. *Point Process Theory and Applications: Marked Point and Piecewise Deterministic Processes*. 2006.

[28] S.-H. Kim and W. Whitt. The Power of Alternative Kolmogorov-Smirnov Tests Based on Transformations of the Data. *ACM Trans. Model. Comput. Simul.*, 25:24:1–24:22, 2015.

[29] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2014.

[30] P. J. Laub, T. Taimre, and P. K. Pollett. Hawkes Processes. *arXiv:1507.02822*, 2015.

[31] P. A. W. Lewis. Some Results on Tests for Poisson Processes. *Biometrika*, 52(1/2):67–77, 1965.

[32] P. A. W. Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413, 1979.

[33] T. Liniger. Multivariate Hawkes Processes. *Doctoral thesis*, 2009.

[34] W. S. McCulloch and W. Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *MIT Press*, pages 15–27, 1988.

[35] A. J. McNeil, R. Frey, and P. Embrechts. *Quantitative Risk Management: Concepts, Techniques and Tools*. Princeton University Press, Princeton, NJ, USA, 2015.

[36] H. Mei and J. Eisner. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process. *CoRR*, abs/1612.09328, 2016.

[37] 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.

[38] Y. Ogata. On Lewis' simulation method for point processes. *IEEE Trans. Information Theory*, 27:23–30, 1981.

[39] Y. Ogata. Statistical Models for Earthquake Occurrences and Residual Analysis for Point Processes. *Journal of the American Statistical Association*, 83(401):9–27, 1988.

[40] C. Olah. Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2019-06-20.

[41] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1310–1318, 2013.

[42] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. *CoRR*, abs/1704.02971, 2017.

[43] J. G. Rasmussen. Temporal point processes and the Conditional Intensity Function. Aalborg University, 2018.

[44] C. Redenbach. Mathematical Statistics. TU Kaiserslautern, 2017.

[45] S. M. Ross. *Stochastic processes*. Wiley series in probability and statistics: Probability and statistics. Wiley, 1996.

[46] D. Rumelhart, G. Hinton, and R. Williams. Learning Representations by Back Propagating Errors. *Nature*, 323:533–536, 1986.

[47] G. Steidl. Numerics. TU Kaiserslautern, 2016.

[48] H. von Weizsaecker. Probability Theory 1. TU Kaiserslautern, 2013.

[49] M. Wiese. Deep Generative Modeling of Financial Time Series. *Master thesis*, 2019.

[50] M. Wiese, R. Knobloch, R. Korn, and P. Kretschmer. Quant GANs: Deep Generation of Financial Time Series. *arXiv:1907.06673*, 2019.

[51] R. Williams and D. Zipser. Gradient-Based Learning Algorithms for Recurrent Networks and Their Computational Complexity. *Developments in connectionist theory. Backpropagation: Theory, architectures, and applications*, pages 433–486, 1998.

[52] S. Xiao, M. Farajtabar, X. Ye, J. Yan, L. Song, and H. Zha. Wasserstein Learning of Deep Generative Point Process Models. *CoRR*, abs/1705.08051, 2017.

[53] S. Xiao, J. Yan, S. M. Chu, X. Yang, and H. Zha. Modeling The Intensity Function Of Point Process Via Recurrent Neural Networks. *CoRR*, abs/1705.08982, 2017.