THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Hawkes Processes: Simulation, Estimation, and Validation

Patrick Laub, BE(Software, Hons. I)/BSc(Mathematics)
Supervised by Prof. Phil Pollett

A thesis submitted in partial fulfillment of the degree of
Bachelor of Science (Honours) in 2014
School of Mathematics and Physics

To Tiffles.
The wind rises... We must try to live!

Acknowledgements

Firstly, I'd like to thank my supervisor and mentor Phil Pollett. Your influence has been wise, reassuring, and monumental. If it weren't for you I would not be in the probability game at all! Also, I need to thank Thomas Taimre. Your generous imparting of knowledge ranged from Stochastic Loewner Evolution to the correct font selection for differentials.

To my family, I apologise for the lack of frequent contact, and thank you for your unerring support of my academic venture. In particular my aunt Mrs. Chris Brown was pivotal to the creation of this thesis. Your willingness to read draft after draft was a phenomenal help in motivating the writing/editing process, and I thank you for it.

Finally, I'd like to thank all of my friends. Fig. 6.1a can be re-purposed to describe the rate at which I've accepted your invitations to events around semester crunch periods. And thanks to my fellow honours students, who made the year very enjoyable. I'll end with the words of Ernest Rhys applied to our little dungeon in the maths building:

"Strange things pass nightly in this little room,
All dreary as it looks by light of day;
Enchantment reigns here when at evening play
Red fire-light glimpses through the pallid gloom."

Contents

List of Symbols

$\mathbb{R}^+$ Positive reals $\{x : x \in \mathbb{R}, x > 0\}$

$\mathbb{1}(\cdot)$ Indicator function

$\Phi(x)$ Standard normal cumulative distribution function (c.d.f.)

$\text{Exp}(r)$ Exponential distribution (with rate r)

$\text{Poi}(r)$ Poisson distribution (with rate r)

$\text{Unif}(a, b)$ Uniform distribution over $[a, b]$

$a := b$ Reads 'a is defined as b'

$N(t)$ The number of arrivals before time t (a counting process), page 3

$\mathcal{H}(t)$ The history of arrivals up to time u (a filtration), page 3

$\boldsymbol{T} = \{T_1, T_2, \dots\}$ The random time of arrivals (a point process), page 3

$F^*(t)$ The cumulative distribution function (c.d.f.) of the next arrival given the previous arrival history, page 5

$f^*(t)$ The conditional probability density function (p.d.f.) of the next arrival given the previous arrival history, page 5

$\lambda^*(t)$ The conditional intensity function, page 6

$\Lambda(t)$ The compensator (or integrated conditional intensity function), page 7

λ The background intensity of the Hawkes process, page 9

$\mu(s)$ The excitation function for the Hawkes process, page 9

$\{t_1, t_2, \dots, t_k\}$ An observed sequence of arrival times, page 10

α Jump in intensity after arrival (exponential decay), page 11

β Intensity decay rate after arrival (exponential decay), page 11

n Branching ratio, page 12

List of Acronyms

a.s. almost surely

c.d.f. cumulative distribution function

i.i.d. independent and identically distributed

p.d.f. probability density function

w.p. with probability

SDE stochastic differential equation

Q–Q quantile–quantile (plot)

Chapter 1

Introduction

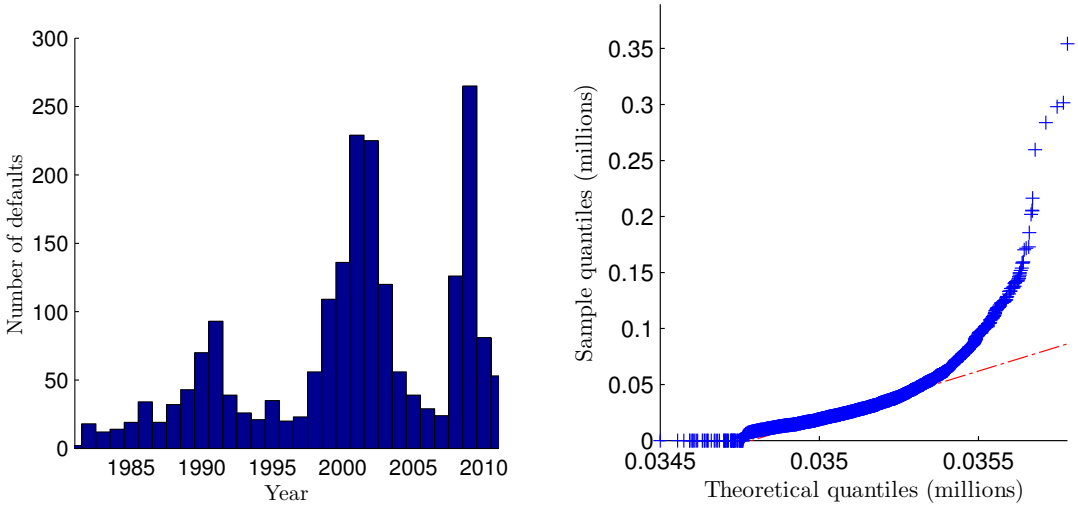
It is expected that some types of events that are observed will naturally cluster in time. An earthquake typically increases the geological tension of the region in which it occurs, and aftershocks will likely follow (Ogata 1988). A fight between rival gangs might ignite a spate of criminal retaliations (Mohler et al. 2011). Selling a significant quantity of a stock could precipitate a trading flurry or, on a larger scale, the collapse of a Wall Street investment bank could send shockwaves through the world's financial centres (Azizpour et al. 2010).

A mathematical model for these so-called 'self-exciting' processes is the *Hawkes process* (Hawkes 1971a). The Hawkes process is a counting process that models a sequence of 'arrivals' of some type over time, e.g. earthquakes, gang violence, trade orders, or bank defaults. Each arrival *excites* the process in that the likelihood of a subsequent arrival is increased for some time period after the initial arrival. As such, it is a non-Markovian extension of the Poisson process.

As the Hawkes process literature in financial fields is particularly well developed, applications in these areas are chiefly considered. Some datasets seem intuitively to come from a self-exciting process, such as the number of companies defaulting on loans each year (seen in Fig. 1.1a). Similarly, using a basic Poisson process to model the arrival of trade orders to a stock is a highly unrealistic notion (an example Q–Q plot failing this hypothesis is shown in Fig. 1.1b). This is because participants in equity markets exhibit a herding behaviour, a standard example of economic reflexivity (Filimonov and Sornette 2012).

Though the Hawkes process has been widely used in many fields it is often difficult to fit to a dataset. Current methods are computationally inefficient, biased, convoluted, or rely on simplifying assumptions. The process of generating, model fitting, and testing the goodness of fit of Hawkes processes is examined in this thesis.

The remainder of this thesis is structured as follows. Some background to Hawkes processes is given in Chapter 2, and then the history of Hawkes processes up to the



(a) (b)

Figure 1.1: (a) Global corporate defaults 1981–2011. (b) Q–Q plot testing daily traded volume of code ASX:ILC to be a Poisson random variables.

latest developments and applications in Chapter 3. Chapter 4 shows how one would fit a Hawkes process to data and Chapter 5 describes how to test such a fit. Methods for simulating realisations of Hawkes processes are outlined in Chapter 6 with MATLAB implementations included in the appendices. Finally, Chapter 7 summarises the key insights into Hawkes processes examined in this thesis.

Sections 3.4, 3.6, and subsection 5.2.4 have been marked with a †. These sections require background knowledge, for example in spectral analysis and financial mathematics (such as in Tankov 2003). These sections can be avoided without loss of intelligibility. In a similar fashion, overly technical definitions and comments are footnoted and lengthy segments of proofs are placed in Appendix A.

Chapter 2

Background

2.1 Counting and point processes

In order to precisely define the Hawkes processes, basic rules of point processes and counting processes are developed. This chapter begins by introducing the definitions of these two fundamental processes, building up to the Poisson process. Discussion of the Poisson process is necessary as one can view the Hawkes process as a generalisation of the (time-)inhomogeneous Poisson process. Note that only definitions for the one-dimensional case are given, though many of these processes have a natural extension to higher dimensions. To begin, consider:

Definition 1. *Counting process*
A *counting process* is a stochastic process $(N(t) : t \geq 0)$ taking values in $\mathbb{N}_0$ that satisfies $N(0) = 0$, is finite, and is a right-continuous step function with jumps of size +1. Say that $(\mathcal{H}(u) : u \geq 0)$ is the history[1] of the arrivals up to time u.

A counting process can be viewed as a cumulative count of the number of 'arrivals' into a system up to the current time. Another way to characterise such a process is to consider the sequence of random arrival times $\boldsymbol{T} = \{T_1, T_2, \dots\}$ at which the counting process $N(\cdot)$ has jumped. The process defined as these arrival times is called a point process, described in Definition 2 (adapted from Carstensen 2010); see Fig. 2.1 for an example point process and its associated counting process.

Definition 2. *Point process*
If a sequence of random variables $\boldsymbol{T} = \{T_1, T_2, \dots\}$, taking values in $\mathbb{R}^+ \cup \{\infty\}$, has: $\mathbb{P}(0 < T_0 \leq T_1 \leq T_2 \leq \dots) = 1$, $\mathbb{P}(T_i < T_{i+1}, T_i < \infty) = \mathbb{P}(T_i < \infty)$ for $i \geq 1$, and the number of points in a bounded region is finite almost surely (a.s.), then $\boldsymbol{T}$ is a *(simple) point process*.

[1] Strictly speaking $\mathcal{H}(\cdot)$ is a filtration, i.e., an increasing sequence of σ-algebras.

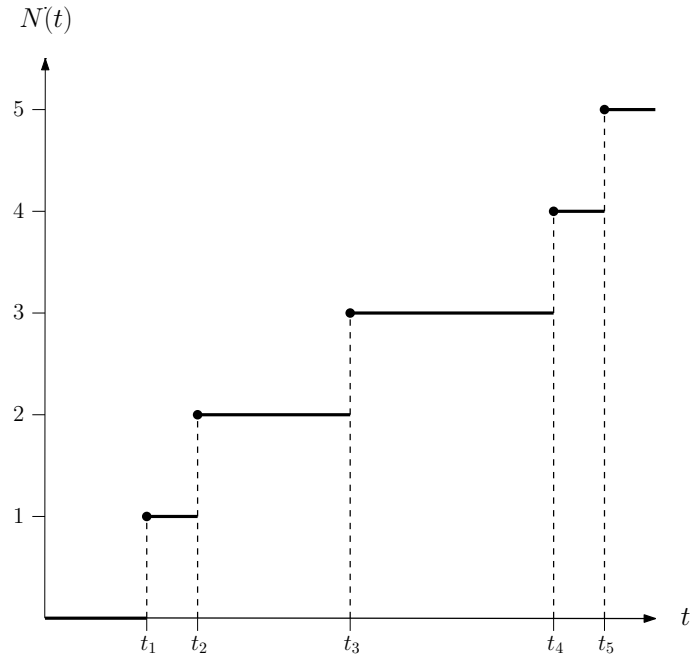


Figure 2.1: An example point process realisation $\{t_1, t_2, \dots\}$ and corresponding counting process $N(t)$.

Note that point processes need not be restricted to the half-line. An equivalent definition to Definition 2 is the *random counting measure*[2]. The definition is included for completeness:

Definition 3. *Random counting measure*
Let (E, Σ) be a measurable space (typically $E \subset \mathbb{R}^d$ and Σ a σ-algebra over E) and $T_1, \dots, T_n \in E$ be a collection of random points. Then $N : \Sigma \to \mathbb{N}$ defined by

$$N(A) = \sum_{i=1}^{n} \mathbb{1}(T_i \in A)$$

is a *random counting measure* over E.

The point and counting process terminology is often interchangeable. For example, if one refers to a Poisson process or a Hawkes process then, the reader must infer from the context whether the counting process $N(\cdot)$ or the point process of times $\boldsymbol{T}$ is being discussed.

[2]See Kallenberg 1976 for a complete treatment.

A way to characterise a particular point process is to give the distribution function of the next arrival time conditioned on the past. Given the history up until the last arrival u, written as $\mathcal{H}(u)$, define (as per Ozaki 1979) the conditional c.d.f. (and p.d.f.) of the next arrival time T_{k+1} as

$$F^*(t \,|\, \mathcal{H}(u)) = \int_u^t \mathbb{P}(T_{k+1} \in [s + \mathrm{d}s] \,|\, \mathcal{H}(u))\, \mathrm{d}s = \int_u^t f^*(s \,|\, \mathcal{H}(u))\, \mathrm{d}s\,.$$

The joint density for a realisation $\{t_1, t_2, \ldots, t_k\}$ is then, by the chain rule,

$$f(t_1, t_2, \ldots, t_k) = \prod_{i=1}^{k} f^*(t_i \,|\, \mathcal{H}(t_{i-1}))\,. \tag{2.1}$$

In the literature the notation rarely conditions on these histories $\mathcal{H}(\cdot)$ explicitly, but instead places a superscript star over the function (e.g. see Daley and Vere-Jones 2003). As such these functions are instead written as $F^*(t)$ and $f^*(t)$. This characterisation provides a definition for various classes of point processes. If a point process has a distribution $f^*(t)$ which is independent of $\mathcal{H}(t)$ then the process is called a *renewal process*. Another way to state this is that $f^*(t) = g(t - t_k)$ for some p.d.f. $g : \mathbb{R}^+ \to \mathbb{R}^+$, so the interarrival times are independent and identically distributed (i.i.d.) random variables. For example, if these i.i.d. interarrival times are exponentially distributed then the renewal process is called a *homogeneous Poisson process*.

2.2 Poisson processes

Of particular interest to the study of Hawkes processes is the *inhomogeneous Poisson process*. Definition 4 describes how these processes are characterised. Again remember that this is the one-dimensional definition, not the most general definition of Poisson processes (see Kroese and Botev 2014 for the random measure definition used in higher dimensions and for simulation techniques).

Definition 4. *Poisson process*
Say a process $(N(t) : t \geq 0)$ is a counting process and that it satisfies $\forall s < t$ that $N(t) - N(s)$ is independent of $N(s)$ and that

$$\mathbb{P}(N(t+h) - N(t) = m \,|\, N(t)) = \begin{cases} \lambda(t)\, h\,, & m = 1 \\ \mathrm{o}(h)\,, & m > 1 \\ 1 - \lambda(t)\, h + \mathrm{o}(h)\,, & m = 0 \end{cases}$$

then $N(t)$ is called a *inhomogeneous Poisson process* with $\lambda : \mathbb{R}^+ \to \mathbb{R}^+$ called the *intensity function*; though if $\lambda(t) = \lambda$ for all $t \geq 0$, $N(t)$ is a *homogeneous Poisson process*.

2.3 Conditional intensity functions

Often it is difficult to work with the conditional arrival distribution. Instead another characterisation of point processes is used, the conditional intensity function[3]. Originally this function was called the hazard function (Cox 1955) and was defined as

$$\lambda^*(t) = \frac{f^*(t)}{1 - F^*(t)}\,. \tag{2.2}$$

Although this definition is still valid, Definition 5 gives a more intuitive representation of the conditional intensity function as the expected rate of arrivals conditioned on $\mathcal{H}(t)$.

> **Definition 5.** *Conditional intensity function*
> Consider a counting process $N(\cdot)$ with associated histories $\mathcal{H}(\cdot)$. If a function $\lambda^*(t)$ exists such that
>
> $$\lambda^*(t) = \lim_{h \downarrow 0} \frac{\mathbb{E}[N(t+h) - N(t) \,|\, \mathcal{H}(t)]}{h}$$
>
> that only relies on information of $N(\cdot)$ in the past[4], then it is called the *conditional intensity function* of $N(\cdot)$.

The terms 'self-exciting' and 'self-regulating' can be defined in terms of the conditional intensity function. If an arrival causes the conditional intensity function to rise then the process is said to be *self-exciting*. This behaviour causes temporal clustering of $\boldsymbol{T}$. In this setting $\lambda^*(t)$ must be chosen to avoid explosion (defined as $N(t) = \infty$ for finite t with non-zero probability). See Fig. 2.2 for an example realisation of such a $\lambda^*(t)$.

The opposite behaviour could be observed, in that the conditional intensity function drops after an arrival. This type of process is called *self-regulating* and the arrival times appear quite temporally regular. Such processes are not examined in this document, though a simple example would be the arrival of speeding tickets to a driver over time (assuming each arrival causes a period of heightened caution when driving).

[3] Indeed if the conditional intensity function exists it uniquely characterises the finite-dimensional distributions of the point process (see Proposition 7.2.IV Daley and Vere-Jones 2003).

[4] I.e. $\lambda^*(t)$ is $\mathcal{H}(t)$–measurable.

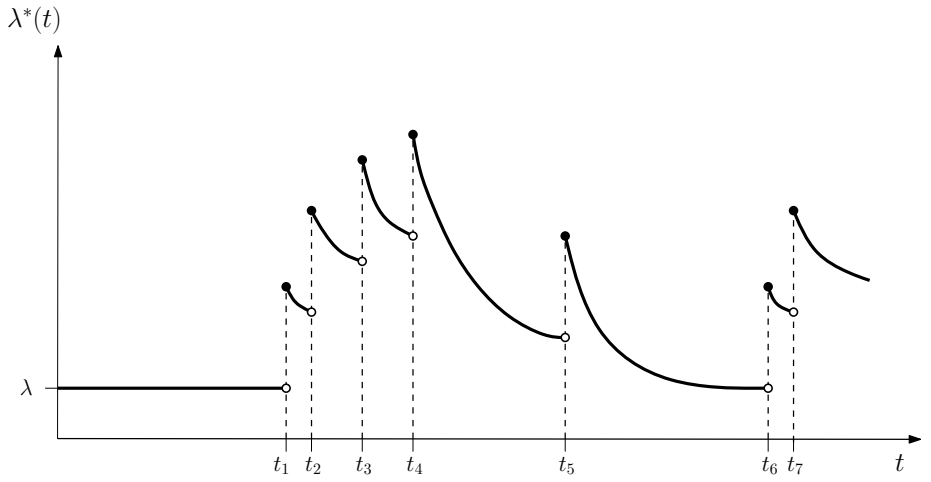


Figure 2.2: An example $\lambda^*(t)$ for a self-exciting process.

2.4 Compensators

Frequently the integrated conditional intensity function is needed (e.g. in parameter estimation and goodness of fit testing), hence it has been named:

Definition 6. *Compensator*
For a counting process $N(\cdot)$ the non-decreasing function

$$\Lambda(t) = \int_0^t \lambda^*(s)\, \mathrm{d}s$$

is called the *compensator* of the counting process.

In fact a compensator is typically defined more subtly[5] and exists[6] even when $\lambda^*(\cdot)$ does not exist. However for Hawkes processes $\lambda^*(\cdot)$ always exists (in fact, a Hawkes process is defined by this function) and therefore Definition 6 is sufficient.

[5]Technically $\Lambda(t)$ is the unique $\mathcal{H}(t)$ predictable function, with $\Lambda(0) = 0$, and is non-decreasing, such that $N(t) = M(t) + \Lambda(t)$ almost surely for $t \geq 0$ and where $M(t)$ is an $\mathcal{H}(t)$ local martingale.

[6]Existence guaranteed by the Doob–Meyer decomposition theorem.

Chapter 3

Literature Review

3.1 Hawkes process definition

Point processes gained a significant amount of attention in the field of statistics during the 1950s and 1960s. First Cox (1955) introduced the notion of a doubly stochastic Poisson process (now called the Cox process) and Bartlett (1963a,b, 1964) investigated statistical methods for point processes based on their power spectral densities. At IBM Research Laboratories Lewis (1964) formulated a point process model (for computer failure patterns) which was a step in the direction of the Hawkes process. The activity culminated in the significant monograph by Cox and Lewis (1966) on time series analysis; modern researchers appreciate this text as an important development of point process theory since it canvassed their wide range of applications (Daley and Vere-Jones 2003, p. 16).

It was in this context that Hawkes (1971a) set out to bring Bartlett's spectral analysis approach to a new process, a self-exciting point process. The process Hawkes described was a one-dimensional point process (though defined on $t \in \mathbb{R}$ as opposed to Poisson processes defined for $t \in \mathbb{R}^+$), and is characterised by:

Definition 7. *Hawkes process*
Consider $(N(t) : t \in \mathbb{R})$ a counting process, with associated history $(\mathcal{H}(t) : t \in \mathbb{R})$, that satisfies

$$\mathbb{P}(N(t+h) - N(t) = m \,|\, \mathcal{H}(t)) = \begin{cases} \lambda^*(t)\, h + o(h)\,, & m = 1 \\ o(h)\,, & m > 1 \\ 1 - \lambda^*(t)\, h + o(h)\,, & m = 0 \end{cases}.$$

Suppose the process' conditional intensity function is of the form

$$\lambda^*(t) = \lambda + \int_{-\infty}^{t} \mu(t-u)\, \mathrm{d}N(u) \tag{3.1}$$

for some $\lambda \in \mathbb{R}^+$ and $\mu : \mathbb{R}^+ \to \mathbb{R}^+ \cup \{0\}$[1] which are called the *background intensity* and *excitation function* respectively. Such a process $N(\cdot)$ is a *Hawkes process*.

[1] Assume that $\mu(\cdot) \neq 0$ as the contrary would entail a homogeneous Poisson process.

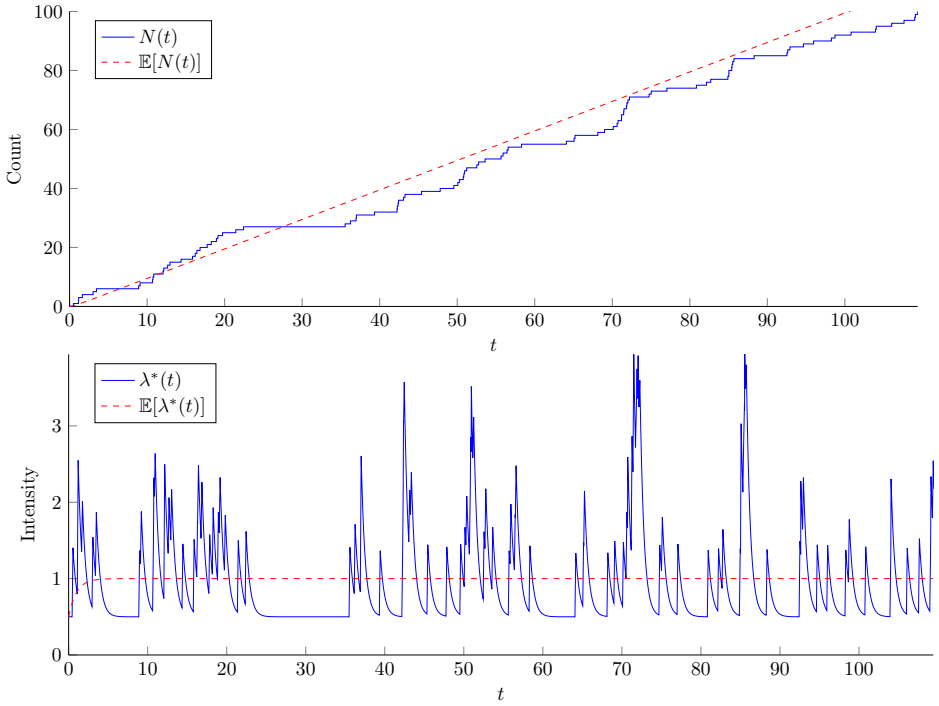


Figure 3.1: A typical Hawkes process realisation $N(t)$ and associated $\lambda^*(t)$ plotted against their expected values.

In modern terminology, Definition 7 describes a linear Hawkes process (the non-linear definition is given later in Definition 8). A realisation of a Hawkes process is shown in Fig. 3.1 with the associated path of the conditional intensity process. Hawkes (1971b) soon extended this single point process into a collection of self- and mutually-exciting point processes, which is discussed after elaborating upon this one-dimensional process.

3.2 Hawkes conditional intensity function

The form of the Hawkes conditional intensity function in Eq. (3.1) is consistent with the literature though it somewhat obscures the intuition. Using $\{t_1, t_2, \ldots, t_k\}$ to denote the observed sequence of past arrival times of the point process up to time t, the Hawkes conditional intensity is

$$\lambda^*(t) = \lambda + \sum_{t_i < t} \mu(t - t_i)\,.$$

The structure of this $\lambda^*(\cdot)$ is quite flexible and requires specification of the excitation function. A common choice for the excitation function is one of exponential decay; Hawkes (1971a) originally used this form as it simplified his theoretical derivations (Hautsch 2011). In this case $\mu(\cdot)$ is specified by constants $\alpha, \beta \in \mathbb{R}^+$ such that

$$\begin{aligned} \lambda^*(t) &= \lambda + \int_{-\infty}^{t} \alpha e^{-\beta(t-s)} \, dN(s) \\ &= \lambda + \sum_{t_i<t} \alpha e^{-\beta(t-t_i)} \, . \end{aligned} \tag{3.2}$$

The constants have the following intuition: each arrival in the system instantaneously increases the arrival intensity by α, then over time this arrival's influence decays at rate β. As noted earlier λ is the background intensity of the process.

Another frequent choice is a power law function, such as

$$\begin{aligned} \lambda^*(t) &= \lambda + \int_{-\infty}^{t} \frac{k}{(c + (t-s))^p} \, dN(s) \\ &= \lambda + \sum_{t_i<t} \frac{k}{(c + (t-t_i))^p} \end{aligned}$$

with some positive scalars c, k, and p. The power law form was popularised by the geological model called Omori's law, used to predict the rate of aftershocks caused by an earthquake (Ogata 1999). More computationally efficient than either of these excitation functions is a piecewise linear function as in Chatalbashev et al. (2007). However, the following discussion will focus on the exponential form of the excitation function.

One practical problem to consider is that it is not possible to observe processes in nature from time minus infinity. If the Hawkes process is restricted to $\mathbb{R}^+$ with some initial condition $\lambda^*(0) = \lambda_0$ then the conditional intensity process satisfies the stochastic differential equation (SDE)

$$d\lambda^*(t) = \beta(\lambda - \lambda^*(t)) \, dt + \alpha \, dN(t) \, , \quad t \geq 0 \, .$$

Applying stochastic calculus to yield the general solution of

$$\lambda^*(t) = e^{-\beta t}(\lambda_0 - \lambda) + \lambda + \int_0^t \alpha e^{\beta(t-s)} \, dN(s) \, , \quad t \geq 0 \, ,$$

which is a natural extension of Eq. (3.2) (Da Fonseca and Zaatour 2014).

3.3 Immigration–birth representation

Stability properties of the Hawkes process are often simpler to divine if it is viewed as a branching process. Imagine counting the population in a country where people arrive either via *immigration* or by *birth*. Say that the stream of immigrants to the country form a homogeneous Poisson process at rate λ. Each person then produces zero or more children in an i.i.d. fashion, and the arrival of births form an inhomogeneous Poisson process.

An illustration of this interpretation can be seen in Fig. 3.2. In branching theory terminology, this *immigration–birth representation* describes a Galton–Watson process with a modified time dimension. Hawkes and Oakes (1974) used the representation to derive asymptotic characteristics of the process, such as Theorem 1. More modern work uses the representation for applying Bayesian techniques (see Rasmussen 2013).

Theorem 1. *Hawkes process asymptotic normality*
If

$$0 < n := \int_0^\infty \mu(s)\, \mathrm{d}s < 1$$

and

$$\int_0^\infty s\mu(s)\, \mathrm{d}s < \infty$$

then the number of Hawkes process arrivals in $(0, t]$ as $t \to \infty$ is normally distributed. Using the notation that $N(0, t] = N(t) - N(0)$ then

$$\mathbb{P}\left(\frac{N(0,t] - \lambda t/(1-n)}{\sqrt{\lambda t/(1-n)^3}} \leq y \right) \to \Phi(y)\,.$$

For a person who enters the system at time $t_i \in \mathbb{R}$, the rate at which they produce offspring at future times $t > t_i$ is $\mu(t - t_i)$. Say that the direct offspring of this person are the *first-generation*, and their offspring are the *second-generation*, and so on; members of the union of all these generations are called the *descendants* of this t_i arrival.

Using the notation from Grimmett and Stirzaker (2001) Section 5.4, define Z_i to be the random number of offspring in ith generation (with $Z_0 = 1$). As the first-generation offspring arrived from a Poisson process $Z_1 \sim \mathrm{Poi}(n)$ where the mean n is

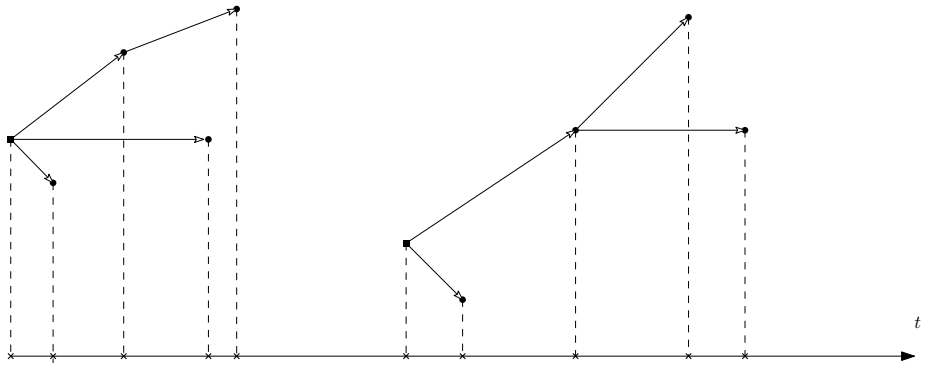


Figure 3.2: Hawkes process represented as a collection of family trees (immigration–birth representation). Squares indicate immigrants, circles are offspring/descendants, and the crosses denote the generated point process.

known as the *branching ratio*. This branching ratio (which can take values in $(0, \infty]$) is defined in Theorem 1 and in the case of an exponentially decaying intensity is

$$n = \int_0^\infty \alpha e^{-\beta s} \, \mathrm{d}s = \frac{\alpha}{\beta} \,. \tag{3.3}$$

Knowledge of the branching ratio can inform development of simulation algorithms. For each immigrant i, the times of the first-generation offspring arrivals — conditioned on knowing the total number of them Z_1 — are each i.i.d. with density $\mu(t - t_i)/n$. Section 6 explores Hawkes process simulation methods inspired by the immigration–birth representation in more detail.

The value of n also determines whether or not the Hawkes process *explodes*[2]. To see this, let $g(t) = \mathbb{E}[\lambda^*(t)]$. A renewal–type equation will be constructed for g and then its limiting value will be determined. Conditioning on the time of the first jump,

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

[2]Using the standard definition as the event that $N(t) - N(s) = \infty$ for $t - s < \infty$.

In order to calculate this expected value, start with

$$\lambda^*(s) = \lim_{h \downarrow 0} \frac{\mathbb{E}[N(s+h) - N(s) \,|\, \mathcal{H}(s)]}{h} = \frac{\mathbb{E}[\mathrm{d}N(s) \,|\, \mathcal{H}(s)]}{\mathrm{d}s}$$

and take expectations (and apply the tower property)

$$g(s) = \mathbb{E}[\lambda^*(s)] = \frac{\mathbb{E}[\mathbb{E}[\mathrm{d}N(s) \,|\, \mathcal{H}(s)]]}{\mathrm{d}s} = \frac{\mathbb{E}[\mathrm{d}N(s)]}{\mathrm{d}s}$$

to see that

$$\mathbb{E}[\mathrm{d}N(s)] = g(s)\,\mathrm{d}s\,.$$

Therefore

$$\begin{aligned} g(t) &= \lambda + \int_0^t \mu(t-s)\, g(s)\,\mathrm{d}s \\ &= \lambda + \int_0^t g(t-s)\, \mu(s)\,\mathrm{d}s\,. \end{aligned}$$

This renewal–type equation (in convolution notation is $g = \lambda + g \star \mu$) then has different solutions according to the value of n. Asmussen (2003) splits the cases into: the *defective* case ($n < 1$), the *proper* case ($n = 1$), and the *excessive* case ($n > 1$). Proposition 7.4 states that for the defective case

$$g(t) = \mathbb{E}[\lambda^*(t)] \to \frac{\lambda}{1-n}\,. \tag{3.4}$$

However in the excessive case then $\lambda^*(t) \to \infty$ exponentially quickly, and hence $N(\cdot)$ eventually explodes a.s.

Explosion for $n > 1$ is supported by viewing the arrivals as a branching process. Since $\mathbb{E}[Z_i] = n^i$ (see Section 5.4 Lemma 2 of Grimmett and Stirzaker 2001) the expected number of descendants for one person is

$$\mathbb{E}\left[\sum_{i=1}^{\infty} Z_i\right] = \sum_{i=1}^{\infty} \mathbb{E}[Z_i] = \sum_{i=1}^{\infty} n^i = \begin{cases} \frac{n}{1-n}, & n < 1 \\ \infty, & n \geq 1 \end{cases}.$$

Therefore $n \geq 1$ means that one immigrant would generate infinitely many descendants on average.

When $n \in (0,1)$ the branching ratio is a probability and is intuitively understood as the ratio of the number of total offspring to the size of the entire family (i.e., the total offspring plus the original immigrant); that is

$$\frac{\mathbb{E}\left[\sum_{i=1}^{\infty} Z_i\right]}{1+\mathbb{E}\left[\sum_{i=1}^{\infty} Z_i\right]} = \frac{\frac{n}{1-n}}{1+\frac{n}{1-n}} = \frac{\frac{n}{1-n}}{\frac{1}{1-n}} = n\,.$$

Therefore, any Hawkes process arrival selected at random was generated *endogenously* (i.e., a child) with probability (w.p.) n or *exogenously* (i.e., an immigrant) w.p. $1-n$. Most properties of the Hawkes process rely on the process being *stationary* which is another way to insist that $n \in (0,1)$ (a rigorous definition is given in Section 3.4), so this is assumed hereinafter.

3.4† Covariance and power spectral densities

Hawkes processes originated from the spectral analysis of general stationary point processes. Finding the power spectral density of the Hawkes process gives access to many techniques from the field; for example, model fitting can be achieved by using the observed periodogram of a realisation. The power spectral density is defined in terms of the covariance density. Once again the exposition is simplified by using the shorthand that

$$\mathrm{d}N(t) = \lim_{h \downarrow 0} N(t+h) - N(t)\,.$$

Unfortunately the term "stationary" has many different meanings in probability theory. In this context the Hawkes process is stationary when the jump process $(\mathrm{d}N(t) : t \in \mathbb{R})$ — which takes values in $\{0,1\}$ — is *weakly stationary*. This means that $\mathbb{E}[\mathrm{d}N(t)]$ and $\mathbb{C}\mathrm{ov}(\mathrm{d}N(t), \mathrm{d}N(t+s))$ do not depend on t. Stationarity in this sense does not imply stationarity of $N(\cdot)$ or stationarity of the inter-arrival times (Lewis 1970). One consequence of stationarity is that $\lambda^*(\cdot)$ will have a long term mean (as given by Eq. (3.4))

$$\overline{\lambda^*} := \mathbb{E}[\lambda^*(t)] = \frac{\mathbb{E}[\mathrm{d}N(t)]}{\mathrm{d}t} = \frac{\lambda}{1-n}\,. \tag{3.5}$$

The *(auto)covariance density* is defined, for $\tau > 0$, to be

$$R(\tau) = \mathbb{C}\mathrm{ov}\left(\frac{\mathrm{d}N(t)}{\mathrm{d}t}, \frac{\mathrm{d}N(t+\tau)}{\mathrm{d}\tau}\right).$$

Due to the symmetry of covariance it holds that $R(-\tau) = R(\tau)$, however $R(\cdot)$ cannot be extended to all of $\mathbb{R}$ due to an atom at the origin. For simple point processes $\mathbb{E}[(\mathrm{d}N(t))^2] = \mathbb{E}[\mathrm{d}N(t)]$ (as $\mathrm{d}N(t) \in \{0, 1\}$) therefore for $\tau = 0$

$$\mathbb{E}[(\mathrm{d}N(t))^2] = \mathbb{E}[\mathrm{d}N(t)] = \overline{\lambda^*}\,\mathrm{d}t\,.$$

Therefore the *complete covariance density* (complete in that its domain is all of $\mathbb{R}$) is defined as

$$R^{(c)}(\tau) = \overline{\lambda^*}\delta(\tau) + R(\tau) \tag{3.6}$$

where $\delta(\cdot)$ is the Dirac delta function[3]. The *power spectral density function*

$$S(\omega) := \frac{1}{2\pi}\int_{-\infty}^{\infty} \mathrm{e}^{-i\tau\omega} R^{(c)}(\tau)\,\mathrm{d}\tau = \frac{1}{2\pi}\left[\overline{\lambda^*} + \int_{-\infty}^{\infty} \mathrm{e}^{-i\tau\omega} R(\tau)\,\mathrm{d}\tau\right]. \tag{3.7}$$

Up until this point the discussion (excluding the final value of Eq. (3.5)) has considered general stationary point processes. To apply the theory specifically to Hawkes processes then consider:

Theorem 2. *Hawkes process power spectral density*
Consider a Hawkes process with an exponentially decaying intensity with $\alpha < \beta$. The intensity process then has covariance density, for $\tau > 0$,

$$R(\tau) = \frac{\alpha\beta\lambda(2\beta - \alpha)}{2(\beta - \alpha)^2}\mathrm{e}^{-(\beta-\alpha)\tau}\,.$$

Hence, its power spectral density is, $\forall\,\omega \in \mathbb{R}$,

$$S(\omega) = \frac{\lambda\beta}{2\pi(\beta - \alpha)}\left(1 + \frac{\alpha(2\beta - \alpha)}{(\beta - \alpha)^2 + \omega^2}\right).$$

Proof (adapted from Hawkes 1971a): Consider the covariance density for $\tau \in \mathbb{R} \setminus \{0\}$:

$$R(\tau) = \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\frac{\mathrm{d}N(t+\tau)}{\mathrm{d}\tau}\right] - \overline{\lambda^*}^2\,. \tag{3.8}$$

[3]Typically $R(0)$ is defined such that $R^c(\cdot)$ is everywhere continuous. Lewis (1970, p. 357) states that strictly speaking $R^c(\cdot)$ "does not have a 'value' at $\tau = 0$". See Bartlett (1963b), Cox and Lewis (1966), and Hawkes (1971a) for further details.

Firstly note that, via the tower property,

$$\begin{aligned}\mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\frac{\mathrm{d}N(t+\tau)}{\mathrm{d}\tau}\right] &= \mathbb{E}\left[\mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\frac{\mathrm{d}N(t+\tau)}{\mathrm{d}\tau}\,\middle|\,\mathcal{H}(t+\tau)\right]\right]\\ &= \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\mathbb{E}\left[\frac{\mathrm{d}N(t+\tau)}{\mathrm{d}\tau}\,\middle|\,\mathcal{H}(t+\tau)\right]\right]\\ &= \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\lambda^*(t+\tau)\right].\end{aligned}$$

Hence Eq. (3.8) can be combined with Eq. (3.1) to achieve

$$R(\tau) = \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\left(\lambda + \int_{-\infty}^{t+\tau}\mu(t+\tau-s)\,\mathrm{d}N(s)\right)\right] - \overline{\lambda^*}^2,$$

which yields[4]

$$\begin{aligned}R(\tau) &= \overline{\lambda^*}\mu(\tau) + \int_{-\infty}^{\tau}\mu(\tau-v)R(\tau)\,\mathrm{d}v\\ &= \overline{\lambda^*}\mu(\tau) + \int_0^{\infty}\mu(\tau+v)R(v)\,\mathrm{d}v + \int_0^{\tau}\mu(\tau-v)R(v)\,\mathrm{d}v\,.\end{aligned} \tag{3.9}$$

Taking the Laplace transform of Eq. (3.9) gives[5]

$$\mathscr{L}\left\{R(\tau)\right\}(s) = \frac{\alpha\overline{\lambda^*}(2\beta-\alpha)}{2(\beta-\alpha)(s+\beta-\alpha)}\,. \tag{3.10}$$

Note that Eq. (3.3) and Eq. (3.5) supply

$$\overline{\lambda^*} = \frac{\beta\lambda}{\beta-\alpha} \Rightarrow \mathscr{L}\left\{R(\tau)\right\}(s) = \frac{\alpha\beta\lambda(2\beta-\alpha)}{2(\beta-\alpha)^2(s+\beta-\alpha)}\,, \tag{3.11}$$

$$\therefore R(\tau) = \mathscr{L}^{-1}\left\{\frac{\alpha\beta\lambda(2\beta-\alpha)}{2(\beta-\alpha)^2(s+\beta-\alpha)}\right\} = \frac{\alpha\beta\lambda(2\beta-\alpha)}{2(\beta-\alpha)^2}\mathrm{e}^{-(\beta-\alpha)\tau}\,.$$

The value of $\overline{\lambda^*}$ from Eq. (3.11) and Eq. (3.10) can be substituted into the definition

[4]Refer to Appendix A.1 for details. This is a Wiener–Hopf-type integral equation.
[5]Refer to Appendix A.2 for details.

given in Eq. (3.7):

$$\begin{aligned}
S(\omega) &= \frac{1}{2\pi}\left[\overline{\lambda^*} + \int_{-\infty}^{\infty} \mathrm{e}^{-i\tau\omega} R(\tau)\,\mathrm{d}\tau\right] \\
&= \frac{1}{2\pi}\left[\overline{\lambda^*} + \int_{0}^{\infty} \mathrm{e}^{-i\tau\omega} R(\tau)\,\mathrm{d}\tau + \int_{0}^{\infty} \mathrm{e}^{i\tau\omega} R(\tau)\,\mathrm{d}\tau\right] \\
&= \frac{1}{2\pi}\left[\overline{\lambda^*} + \mathscr{L}\left\{R(\tau)\right\}(i\omega) + \mathscr{L}\left\{R(\tau)\right\}(-i\omega)\right] \\
&= \frac{1}{2\pi}\left[\overline{\lambda^*} + \frac{\alpha\overline{\lambda^*}(2\beta-\alpha)}{2(\beta-\alpha)(i\omega+\beta-\alpha)} + \frac{\alpha\overline{\lambda^*}(2\beta-\alpha)}{2(\beta-\alpha)(-i\omega+\beta-\alpha)}\right] \\
&= \frac{\lambda\beta}{2\pi(\beta-\alpha)}\left[1 + \frac{\alpha(2\beta-\alpha)}{(\beta-\alpha)^2+\omega^2}\right].
\end{aligned}$$

□

As $R(\cdot)$ is a real-valued symmetric function, its Fourier transform $S(\cdot)$ is also real-valued and symmetric, i.e.,

$$S(\omega) = \frac{1}{2\pi}\left[\overline{\lambda^*} + \int_{-\infty}^{\infty} \mathrm{e}^{-i\tau\omega} R(\tau)\,\mathrm{d}\tau\right] = \frac{1}{2\pi}\left[\overline{\lambda^*} + \int_{-\infty}^{\infty} \cos(\tau\omega) R(\tau)\,\mathrm{d}\tau\right], \text{ and}$$

$$S_+(\omega) := S(-\omega) + S(\omega) = 2S(\omega).$$

It is common that $S_+(\cdot)$ is plotted instead of $S(\cdot)$; this is equivalent to wrapping the negative frequencies over to the positive half-line (Cox and Lewis 1966, Section 4.5). Fig. 3.3 shows an example spectral density.

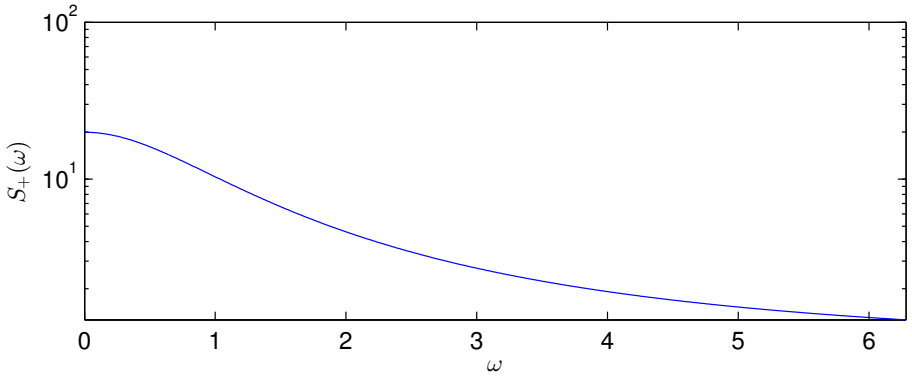


Figure 3.3: Theoretical power density spectrum for a Hawkes process specified by $(\lambda, \alpha, \beta) = (0.5, 4, 5)$; c.f. Fig. 1 of Ozaki (1979).

3.5 Generalisations of Hawkes processes

The immigration–birth representation is useful both theoretically and practically. However it can only be used to describe *linear* Hawkes processes. Brémaud and Massoulié (1996) generalised the Hawkes process to its nonlinear form:

Definition 8. *Nonlinear Hawkes process*
Consider a counting process with conditional intensity function of the form

$$\lambda^*(t) = \Psi\left(\int_{-\infty}^{t} \mu(t-s)\, N(\mathrm{d}s)\right)$$

where $\Psi : \mathbb{R} \to \mathbb{R}^+$, $\mu : \mathbb{R}^+ \to \mathbb{R}$. Then $N(\cdot)$ is a *nonlinear Hawkes process*. Note that selecting $\Psi(x) = \lambda + x$ reduces $N(\cdot)$ to the linear Hawkes process of Definition 7.

Modern work on nonlinear Hawkes processes is much rarer than the original linear case (for simulation see pp. 96–116 of Carstensen 2010, and associated theory in Zhu 2013). This is due to a combination of factors; firstly, the generalisation was introduced relatively recently, and secondly, the increased complexity frustrates even simple investigations.

Now to return to the extension mentioned earlier, that of a collection of self- and *mutually-exciting* Hawkes processes. The processes being examined are collections of one-dimensional Hawkes processes which 'excite' themselves and each other. There are models for Hawkes processes where the points themselves are multi-dimensional, e.g., spatial Hawkes processes or temporo-spatial Hawkes processes (Mohler et al. 2011), which are not examined in this thesis.

Definition 9. *Mutually exciting Hawkes process*
Consider a collection of m counting processes $\{N_1(\cdot), \ldots, N_m(\cdot)\}$ denoted $\boldsymbol{N}$. Say $\{T_{i,j} : i \in \{1, \ldots, m\}, j \in \mathbb{N}\}$ are the random arrival times for each counting process (and $t_{i,j}$ for observed arrivals). If for each $i = 1, \ldots, m$ then $N_i(\cdot)$ has conditional intensity of the form

$$\lambda_i^*(t) = \lambda_i + \sum_{j=1}^{m} \int_{-\infty}^{t} \mu_j(t-u)\, \mathrm{d}N_j(u) \tag{3.12}$$

for some $\lambda_i \in \mathbb{R}^+$ and $\mu_i : \mathbb{R}^+ \to \mathbb{R}^+ \cup \{0\}$ $(\mu_i(\cdot) \not\equiv 0)$, then $\boldsymbol{N}$ is called a *mutually exciting Hawkes process*. When the excitation functions are set to

be exponentially decaying, Eq. (3.12) can be written as

$$\lambda_i^*(t) = \lambda_i + \sum_{j=1}^{m} \int_{-\infty}^{t} \alpha_{i,j} \mathrm{e}^{-\beta_{i,j}(t-s)} \, \mathrm{d}N_j(s) \tag{3.13}$$
$$= \lambda_i + \sum_{j=1}^{m} \sum_{t_{j,k}<t} \alpha_{i,j} \mathrm{e}^{-\beta_{i,j}(t-t_{j,k})}$$

for non-negative constants $\{\alpha_{i,j}, \beta_{i,j} : i, j = 1, \ldots, m\}$.

3.6† Financial applications

This section primarily reviews the work of Aït-Sahalia et al. (2010) and Filimonov and Sornette (2012). It assumes the reader has been introduced to mathematical finance using SDEs; following chapters do not make this assumption and can be skipped to without loss of intelligibility.

3.6.1† Financial contagion

With these definitions the discussion can turn to the latest applications of Hawkes processes. A major domain for self- and mutually-exciting processes is financial analysis. Frequently it is seen that large movements in a major stock market propagate in foreign markets as a process called *financial contagion*. Examples of this phenomenon are clearly visible in historical series of asset prices; Fig. 3.4 shows one such case.

The 'Hawkes diffusion model' introduced by Aït-Sahalia et al. (2010) is an attempt to extend previous models of stock prices to include financial contagion. Modern models for stock prices are typically built upon the model popularised by Black and Scholes (1973) where the log returns on the stock follow geometric Brownian motion. Whilst this seminal paper was lauded by the economics community, the model inadequately captured the 'fat tails' of the return distribution and so was not commonly used by traders (Haug and Taleb 2014). Merton (1976) attempted to incorporate heavy tails by including a Poisson jump process to model booms and crashes in the stock returns; this model is often called Merton diffusion model. The Hawkes diffusion model extends this model by replacing the Poisson jump process with a mutually-exciting Hawkes process, so that crashes can self-excite and propagate in a market and between global markets.

The basic Hawkes diffusion model describes the log returns of m assets $\{X_1(\cdot), \dots, X_m(\cdot)\}$ where each asset $i = 1, \dots, m$ has associated expected return $\mu_i \in \mathbb{R}$, constant volatility $\sigma_i \in \mathbb{R}^+$, and standard Brownian motion $(W_i^X(t) : t \geq 0)$. The Brownian motions have constant correlation coefficients $\{\rho_{i,j} : i, j = 1, \dots, m\}$. Jumps are added by a self- and mutually-exciting Hawkes process (as per Definition 9 with some selection of constants $\alpha_{\cdot,\cdot}$ and $\beta_{\cdot,\cdot}$) with stochastic jump sizes $(Z_i(t) : t \geq 0)$. The asset dynamics are then assumed to satisfy the SDE

$$\mathrm{d}X_i(t) = \mu_i \,\mathrm{d}t + \sigma_i \,\mathrm{d}W_i^X(t) + Z_i(t) \,\mathrm{d}N_i(t) \,.$$

The general Hawkes diffusion model replaces the constant volatilities with stochastic volatilities $\{V_1(\cdot), \dots, V_m(\cdot)\}$ specified by the Heston model. Each asset $i = 1, \dots, m$ has a: long-term mean volatility $\theta_i \in \mathbb{R}^+$, rate of returning to this mean $\kappa_i \in \mathbb{R}^+$, volatility of the volatility $\nu_i \in \mathbb{R}^+$, and standard Brownian motion $(W_i^V(t) : t \geq 0)$[6]. Then the full dynamics are captured by

$$\mathrm{d}X_i(t) = \mu_i \,\mathrm{d}t + \sqrt{V_i(t)} \,\mathrm{d}W_i^X(t) + Z_i(t) \,\mathrm{d}N_i(t) \,,$$

$$\mathrm{d}V_i(t) = \kappa_i(\theta_i - V_i(t)) \,\mathrm{d}t + \nu_i \sqrt{V_i(t)} \,\mathrm{d}W_i^V(t) \,.$$

However the added realism of the Hawkes diffusion model comes at a high price. The constant volatility model requires $5m + 3m^2$ parameters to be fit (assuming $Z_i(\cdot)$ is characterised by two parameters) and the stochastic volatility extension requires an extra $3m$ parameters (assuming $\forall i, j = 1, \dots, m$ that $\mathbb{E}[W_i(\cdot)^V W_j(\cdot)^V] = 0$). In Aït-Sahalia et al. (2010) hypothesis tests reject the Merton diffusion model in favour of the Hawkes diffusion model, however there are no tests for overfitting the data (e.g., Akaike or Bayesian information criterion comparisons). Remember that John Von Neumann (reputedly) claimed that "with four parameters I can fit an elephant" (Dyson 2004).

Simply for computational necessity the authors made a number of simplifying assumptions to reduce the number of parameters to fit (e.g., that the background intensity of crashes is the same for all markets). Even so, the Hawkes diffusion model was only able to be fitted for pairs of markets ($m = 2$) instead of for the globe as a whole. Since the model was calibrated to daily returns of market indices, historical data was easily available (e.g., from Google or Yahoo! finance); care had to be taken to convert timezones and handle the different market opening and closing times. The

[6] Correlation between the $W_\cdot^X(\cdot)$'s is optional, yet the effect would be dominated by the jump component.

parameter estimation method used by Aït-Sahalia et al. (2010) was the generalised method of moments, however the theoretical moments derived satisfy long and convoluted equations.

3.6.2† Mid-price changes and high-frequency trading

A simpler system to model is a single stock's price over time, though there are many different prices to consider. For each stock one could use: the last transaction price, the best ask price, the best bid price, or the mid-price (defined as the average of best ask and best bid prices). The last transaction price includes inherent microstructure noise (e.g. the bid–ask bounce), and the best ask and bid prices fail to represent the actions of both buyers and sellers in the market.

Filimonov and Sornette (2012) model the mid-price changes over time as a Hawkes process. In particular they look at long-term trends of the (estimated) branching ratio. In this context, n shows the proportion of price moves that are not due to external market information but simply reactions to other market participants. This ratio can be seen as the quantification of the principle of economic reflexivity. The authors conclude that the branching ratio has increased dramatically from 30% in 1998 to 70% in 2007.

Later that year Lorenzen (2012) critiqued the test procedure used in this analysis. Filimonov and Sornette (2012) had worked with a dataset with timestamps accurate to a second, and this often led to multiple arrivals nominally at the same time (which is an impossible event for simple point processes). Fake precision was achieved by adding $\text{Unif}(0, 1)$ random fractions of seconds to all timestamps, a technique also used by Bowsher 2007. Lorenzen found that this method added an element of smoothing to the data which gave it a better fit to the model than the actual millisecond precision data. The randomisation also introduced bias to the Hawkes process parameter estimates, particularly of α and β. Lorenzen formed a crude measure of high-frequency trading activity leading to an interesting correlation between this activity and n over the observed period.

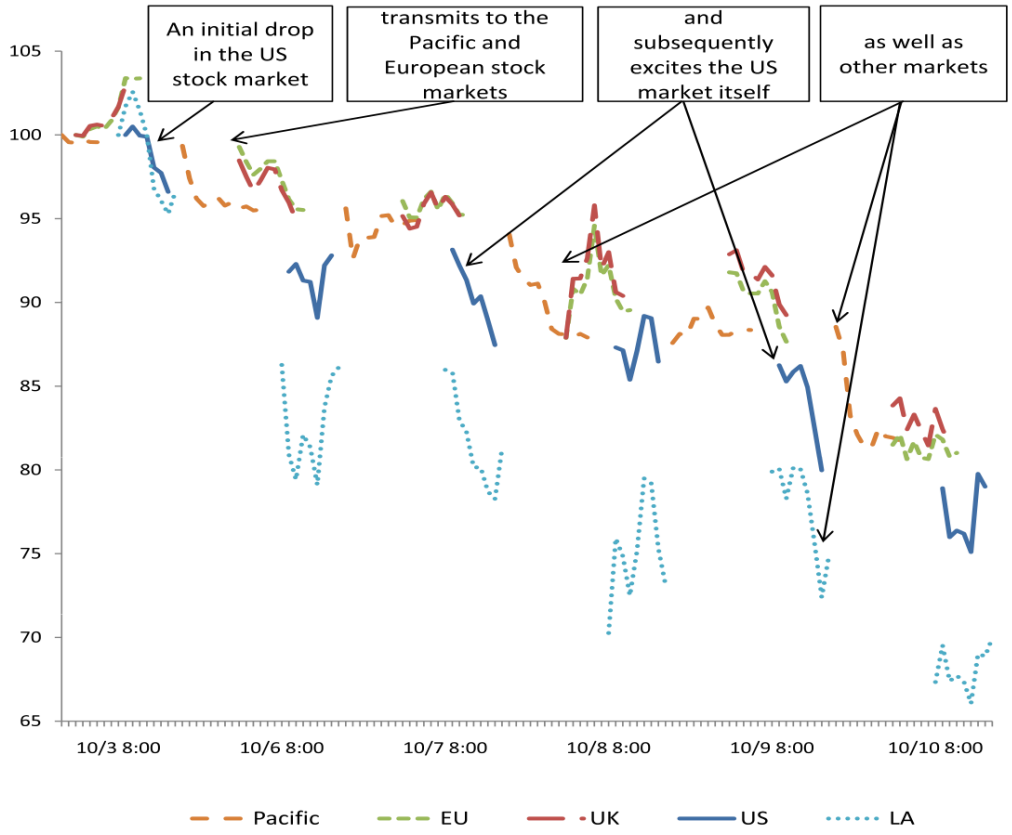


Figure 3.4: Example of mutual excitation in global markets. This figure plots the cascade of declines in international equity markets experienced between October 3, 2008 and October 10, 2008 in the US; Latin America (LA); UK; Developed European countries (EU); and Developed countries in the Pacific. Data are hourly. The first observation of each price index series is normalised to 100 and the following observations are normalised by the same factor. Source: MSCI MXRT international equity indices on Bloomberg (direct copy from Aït-Sahalia et al. 2010).

Chapter 4

Parameter Estimation

This chapter investigates the problem of generating parameters estimates $\widehat{\boldsymbol{\theta}} = (\widehat{\lambda}, \widehat{\alpha}, \widehat{\beta})$ given some finite set of arrival times $\boldsymbol{t} = \{t_1, t_2, \ldots, t_k\}$ presumed to be from a Hawkes process[1]. The estimators are tested over simulated data, for the sake of simplicity and lack of relevant data. Unfortunately this method bypasses the many significant challenges raised by real datasets, challenges that caused Filimonov and Sornette (2013) to state that:

> *"Our overall conclusion is that calibrating the Hawkes process is akin to an excursion within a minefield that requires expert and careful testing before any conclusive step can be taken."*

The method considered is *maximum likelihood estimation*, which begins by finding the *likelihood function*, and estimates the model parameters as the inputs which maximise this function.

4.1 Likelihood function derivation

Daley and Vere-Jones (2003, Proposition 7.2.III) give the following result:

> **Theorem 3.** *Hawkes process likelihood*
> Let $N(\cdot)$ be a regular point process on $[0, T]$ for some finite positive T, and let $t_1, \ldots, t_k$ denote a realisation of $N(\cdot)$ over $[0, T]$. Then, the likelihood L of $N(\cdot)$ is expressible in the form
>
> $$L = \left[\prod_{i=1}^{k} \lambda^*(t_i)\right] \exp\left(-\int_0^T \lambda^*(u)\,\mathrm{d}u\right).$$

[1]Note: The notation will omit the $\widehat{\boldsymbol{\theta}}$ and $\boldsymbol{t}$ arguments from functions, i.e. $L = L(\widehat{\boldsymbol{\theta}}; \boldsymbol{t})$, $l = l(\widehat{\boldsymbol{\theta}}; \boldsymbol{t})$, $\lambda^*(t) = \lambda^*(t; \boldsymbol{t}, \widehat{\boldsymbol{\theta}})$, and $\Lambda(t) = \Lambda(t; \boldsymbol{t}, \widehat{\boldsymbol{\theta}})$.

Proof: The joint density function from Eq. (2.1) is:

$$L = f(t_1, t_2, \ldots, t_k) = \prod_{i=1}^{k} f^*(t_i)\,.$$

This function can be written in terms of the conditional intensity function. Rearrange Eq. (2.2) to find of $f^*(t)$ in terms of $\lambda^*(t)$ (as per Rasmussen 2009):

$$\begin{aligned}\lambda^*(t) &= \frac{f^*(t)}{1 - F^*(t)} \\ &= \frac{\frac{\mathrm{d}}{\mathrm{d}t}F^*(t)}{1 - F^*(t)} \\ &= -\frac{\mathrm{d}\log(1 - F^*(t))}{\mathrm{d}t}\,.\end{aligned}$$

Integrate both sides over the interval (t_k, t):

$$-\int_{t_k}^{t} \lambda^*(u)\,\mathrm{d}u = \log(1 - F^*(t)) - \log(1 - F^*(t_k))\,.$$

The Hawkes process is a *simple* point process, meaning that multiple arrivals cannot occur at the same time. Hence $F^*(t_k) = 0$ as $T_{k+1} > t_k$ so

$$-\int_{t_k}^{t} \lambda^*(u)\,\mathrm{d}u = \log(1 - F^*(t))\,. \tag{4.1}$$

Further rearranging yields

$$\begin{aligned}F^*(t) &= 1 - \exp\left(-\int_{t_k}^{t} \lambda^*(u)\,\mathrm{d}u\right) \\ f^*(t) &= \lambda^*(t)\exp\left(-\int_{t_k}^{t} \lambda^*(u)\,\mathrm{d}u\right)\,.\end{aligned}$$

Thus the likelihood becomes

$$\begin{aligned}L &= \prod_{i=1}^{k} f^*(t_i) \\ &= \prod_{i=1}^{k} \lambda^*(t_i)\exp\left(-\int_{t_{i-1}}^{t_i} \lambda^*(u)\,\mathrm{d}u\right) \\ &= \left[\prod_{i=1}^{k} \lambda^*(t_i)\right]\exp\left(-\int_{0}^{t_k} \lambda^*(u)\,\mathrm{d}u\right)\,.\end{aligned} \tag{4.2}$$

This likelihood is defined for observing a process until the time of the kth arrival. When the process is observed over some time period $[0, T] \supset [0, t_k]$, the likelihood should include the probability of seeing no arrivals in the time interval $(t_k, T]$:

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

Using the same formulation of $F^*(t)$ then

$$\therefore L = \left[\prod_{i=1}^{k} \lambda^*(t_i)\right] \exp\left(-\int_0^T \lambda^*(u)\,\mathrm{d}u\right) .$$

□

4.2 Simplifications for exponential decay

With the likelihood function from Eq. (4.2), the log-likelihood for the interval $[0, t_k]$ can be derived as

$$l = \sum_{i=1}^{k} \log(\lambda^*(t_i)) - \int_0^{t_k} \lambda^*(u)\,\mathrm{d}u = \sum_{i=1}^{k} \log(\lambda^*(t_i)) - \Lambda(t_k) . \tag{4.3}$$

Note that the integral over $[0, t_k]$ can be broken up into the segments $[0, t_1]$, $[t_1, t_2]$, $\ldots$, $[t_{k-1}, t_k]$, and therefore

$$\Lambda(t_k) = \int_0^{t_k} \lambda^*(u)\,\mathrm{d}u = \int_0^{t_1} \lambda^*(u)\,\mathrm{d}u + \sum_{i=1}^{k-1} \int_{t_i}^{t_{i+1}} \lambda^*(u)\,\mathrm{d}u .$$

This can be simplified in the case where $\lambda^*(\cdot)$ decays exponentially:

$$\begin{aligned}
\Lambda(t_k) &= \int_0^{t_1} \lambda\,\mathrm{d}u + \sum_{i=1}^{k-1}\left[\int_{t_i}^{t_{i+1}} \lambda + \sum_{t_j<u} \alpha \mathrm{e}^{-\beta(u-t_j)}\,\mathrm{d}u\right] \\
&= \lambda t_k + \alpha \sum_{i=1}^{k-1} \int_{t_i}^{t_{i+1}} \sum_{j=1}^{i} \mathrm{e}^{-\beta(u-t_j)}\,\mathrm{d}u \\
&= \lambda t_k + \alpha \sum_{i=1}^{k-1} \sum_{j=1}^{i} \int_{t_i}^{t_{i+1}} \mathrm{e}^{-\beta(u-t_j)}\,\mathrm{d}u \\
&= \lambda t_k - \frac{\alpha}{\beta} \sum_{i=1}^{k-1} \sum_{j=1}^{i} \left[\mathrm{e}^{-\beta(t_{i+1}-t_j)} - \mathrm{e}^{-\beta(t_i-t_j)}\right] .
\end{aligned}$$

Finally, many of the terms of this double summation cancel out leaving[2]

$$\begin{aligned}\Lambda(t_k) &= \lambda t_k - \frac{\alpha}{\beta}\sum_{i=1}^{k-1}\left[\mathrm{e}^{-\beta(t_k-t_i)} - \mathrm{e}^{-\beta(t_i-t_i)}\right] \\ &= \lambda t_k - \frac{\alpha}{\beta}\sum_{i=1}^{k}\left[\mathrm{e}^{-\beta(t_k-t_i)} - 1\right]. \end{aligned} \tag{4.4}$$

Substituting $\lambda^*(\cdot)$ and $\Lambda(\cdot)$ into Eq. (4.3) gives

$$l = \sum_{i=1}^{k}\log\left[\lambda + \alpha\sum_{j=1}^{i-1}\mathrm{e}^{-\beta(t_i-t_j)}\right] - \lambda t_k + \frac{\alpha}{\beta}\sum_{i=1}^{k}\left[\mathrm{e}^{-\beta(t_k-t_i)} - 1\right]. \tag{4.5}$$

This direct approach is computationally infeasible as the first term's double summation implies $\mathcal{O}(k^2)$ complexity. Fortunately the similar structure of the inner summations allows l to be computed with $\mathcal{O}(k)$ complexity (Ogata 1978, Crowley 2013). Denote the inner summation, for some value of $i \in \{2, \ldots, k\}$, as

$$A(i) = \sum_{j=1}^{i-1}\mathrm{e}^{-\beta(t_i-t_j)}. \tag{4.6}$$

This can be defined recursively in terms of $A(i-1)$ as follows:

$$\begin{aligned} A(i) &= \sum_{j=1}^{i-1}\mathrm{e}^{-\beta t_i+\beta t_j} \\ &= \mathrm{e}^{-\beta t_i+\beta t_{i-1}}\mathrm{e}^{\beta t_i-\beta t_{i-1}}\sum_{j=1}^{i-1}\mathrm{e}^{-\beta t_i+\beta t_j} \\ &= \mathrm{e}^{-\beta t_i+\beta t_{i-1}}\sum_{j=1}^{i-1}\mathrm{e}^{-\beta t_{i-1}+\beta t_j} \\ &= \mathrm{e}^{-\beta(t_i-t_{i-1})}\left(1 + \sum_{j=1}^{i-2}\mathrm{e}^{-\beta(t_{i-1}-t_j)}\right) \\ &= \mathrm{e}^{-\beta(t_i-t_{i-1})}(1 + A(i-1)). \end{aligned}$$

With the added base case of $A(1) = 0$, l can be rewritten as

$$l = \sum_{i=1}^{k}\log(\lambda + \alpha A(i)) - \lambda t_k + \frac{\alpha}{\beta}\sum_{i=1}^{k}\left[\mathrm{e}^{-\beta(t_k-t_i)} - 1\right]. \tag{4.7}$$

[2]Here the final summand is unnecessary, though it is often included, see Lorenzen (2012).

Ozaki (1979) also gives the partial derivatives and the Hessian for this log-likelihood function. Of particular note is that each derivative calculation can be achieved in order $\mathcal{O}(k)$ complexity when a recursive approach (similar to Eq. (4.6)) is taken (Ogata 1981). The recursion implies that the joint process $(N(t), \lambda^*(t))$ is Markovian (see Remark 1.22 of Liniger 2009).

4.3 Discussion

The understanding of the maximum likelihood estimation method for the Hawkes process has changed significantly over time. The general form of the log-likelihood function, Eq. (4.3), was known by Rubin (1972). It was applied to the Hawkes process by Ozaki (1979) who derived Eq. (4.5) and the improved recursive form Eq. (4.7). Ozaki also found (as noted earlier) an efficient method for calculating the derivatives and the Hessian matrix. Consistency, asymptotic normality and efficiency of the estimator were proved by Ogata (1978).

Therefore the maximum likelihood estimator should be a very effective technology for model fitting. However, Filimonov and Sornette (2012) find that for finite sample sizes the estimator produces significant bias, encounters many local optima, and is highly sensitive to the selection of excitation function.

Also, the $\mathcal{O}(k)$ complexity swiftly becomes unusable when sample sizes become large; remember any iterative optimisation routine would calculate the likelihood function perhaps thousands of times. The R 'hawkes' package thus implements this routine in C++ in an attempt to mitigate the performance issues (Appendix B.1 shows an optimised revision of this routine).

This performance bottleneck is the cause of the latest trend of using the generalised method of moments to perform parameter estimation. Da Fonseca and Zaatour (2014) state that the procedure is "instantaneous" on their test sets. The method uses sample moments and the sample autocorrelation function which are smoothed via (rather arbitrary) user-selected procedure.

Chapter 5

Goodness of Fit

5.1 Transformation to a Poisson process

Assessing the goodness of fit for some point data to a Hawkes model is an important practical consideration. In performing this assessment the point process' compensator is essential, as is the random time change theorem (here adapted from Brown et al. 2002):

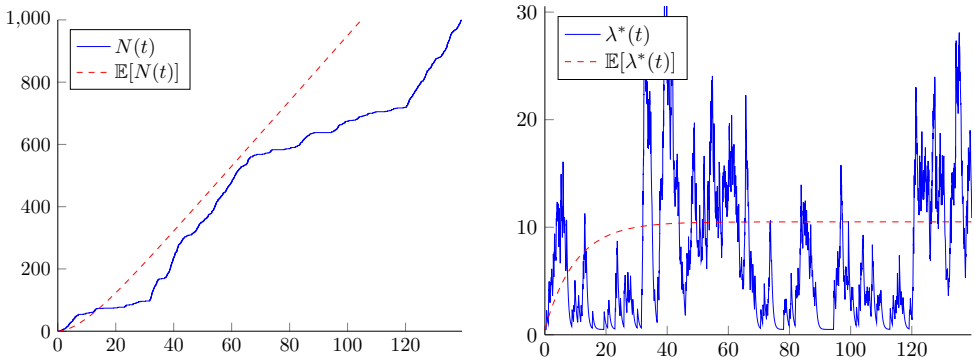
> **Theorem 4.** *Random time change theorem*
> Say $\{t_1, t_2, \ldots, t_k\}$ is a realisation over time $[0, T]$ from a point process with conditional intensity function $\lambda^*(\cdot)$. If $\lambda^*(\cdot)$ is positive over $[0, T]$ and $\Lambda(T) < \infty$ a.s. then the transformed points $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_k)\}$ form a Poisson process with unit rate.

The random time change theorem is fundamental to the model fitting procedure called *(point process) residual analysis*. Daley and Vere-Jones (2003) Proposition 7.4.IV rewords and extends the theorem as such[1]:

> **Theorem 5.** *Residual analysis*
> Consider an unbounded, increasing sequence of time points $\{t_1, t_2, \ldots\}$ in the half-line $(0, \infty)$, and a monotonic, continuous compensator $\Lambda(\cdot)$ such that $\lim_{t\to\infty} \Lambda(t) = \infty$ a.s. The transformed sequence $\{t_1^*, t_2^*, \ldots\} = \{\Lambda(t_1), \Lambda(t_2), \ldots\}$ is a realisation of a unit rate Poisson process if and only if the original sequence $\{t_1, t_2, \ldots\}$ is a realisation from the point process defined by $\Lambda(\cdot)$.

Hence, equipped with a closed form of the compensator from Eq. (4.4), the quality of the statistical inference can be ascertained using standard fitness tests for Poisson processes. Fig. 5.1 shows a realisation of a Hawkes process and the corresponding transformed process

[1] Original work on residual analysis goes back to Meyer (1971), Papangelou (1972) and Watanabe (1964) (Embrechts et al. 2011).



(a) Hawkes process realisation

(b) Conditional intensity

(c) Compensator

(d) Transformed process

Figure 5.1: An example of using the random time change theorem to transform a Hawkes process into a unit rate Poisson process.

5.2 Tests for Poisson process

5.2.1 Basic tests

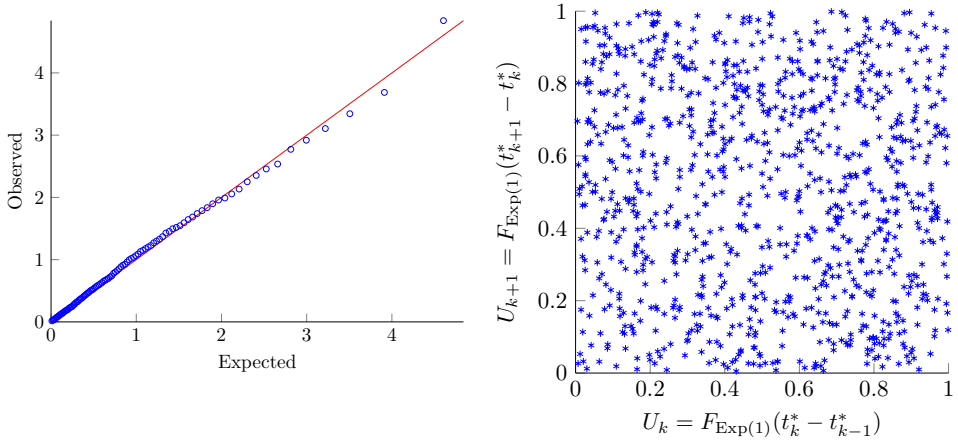
There are many procedures for testing whether a series of points form a Poisson process (see Cox and Lewis 1966 for an extensive treatment). As a first test, one can run a hypothesis test to check $\sum_i \mathbb{1}(t_i^* < t) \sim \text{Poi}(t)$. After this succeeds then the interarrival times,

$$\{\tau_1, \tau_2, \tau_3, \dots\} = \{t_1^*, t_2^* - t_1^*, t_3^* - t_2^*, \dots\},$$

should be tested to ensure $\tau_i \overset{\text{i.i.d.}}{\sim} \text{Exp}(1)$. A qualitative approach is to create a Q–Q plot for τ_i using the exponential distribution (see e.g. Fig. 5.2a). Otherwise a quantitative alternative is to run Kolmogorov–Smirnov (or perhaps Anderson–Darling) tests.

5.2.2 Test for independence

The next test, after confirming there is reason to believe that the τ_i are exponentially distributed, is to check their independence. This can be done by looking for autocorrelation in the τ_i sequence[2]. A visual examination can be performed by plotting the points (U_{i+1}, U_i). If there are noticeable patterns then the τ_i are autocorrelated; otherwise the points should look evenly scattered, see e.g. Fig. 5.2b. Quantitative extensions exist; for example see Section 3.3.3 of Knuth (2014), or serial correlation tests in Kroese et al. (2011).



(a) (b)

Figure 5.2: (a) Q–Q testing for i.i.d. Exp(1) interarrival times. (b) A qualitative autocorrelation test.

5.2.3 Lewis test

A statistical test with more power is the Lewis test as described by Kim and Whitt (2013). Firstly, it relies on the fact that if $\{t_1^*, t_2^*, \ldots, t_N^*\}$ are arrival times for a unit rate Poisson process then $\{t_1^*/t_N^*, t_2^*/t_N^*, \ldots, t_{N-1}^*/t_N^*\}$ are distributed as the order statistics of a uniform $[0, 1]$ distribution. This observation is called conditional uniformity, and forms the basis for a test itself. Lewis' test relies on applying Durbin's modification[3]. See the end of Appendix C.1.1 for an implementation of the test.

[2] Obviously zero autocorrelation does not imply independence, but a non-zero amount would certainly imply a non-Poisson model.

[3] The modification was introduced in Durbin (1961) with a widely applicable treatment by Lewis (1965).

5.2.4† Brownian motion approximation test

An approximate Kolmogorov–Smirnov-type test, Algorithm 7.4.V from Daley and Vere-Jones (2003), is as follows:

1. Given $\{t_1^*, \ldots, t_{N(T)}^*\}$, plot the cumulative step-function $Y(x)$ through the points $(x_i, y_i) = (t_i^*/T, i/N(T))$ in the unit square.
2. Also plot confidence lines $y = x \pm Z_{1-\alpha/2}/\sqrt{T}$ where $\Phi(Z_{1-\alpha/2}) = 1 - \alpha/2$.
3. Finally, accept the hypothesis that $\{t_i^*\}$ come from a unit rate Poisson process if $Y(x)$ stays within the confidence lines (with $100(1-\alpha)\%$ certainty).

An example realisation of this test is shown in Fig. 5.3.

The authors stressed that the test is approximate in two senses. Firstly, it uses the Brownian motion approximation to the Poisson process, and therefore it is a large sample test. Secondly, Kolmogorov–Smirnov-type tests introduce bias when the data being tested is also the data used to estimate parameters. Schoenberg (2002) shows the interesting way in which this bias exhibits itself for small sample sizes.

However the test does not make sense statistically, nor does it perform well. Fig. 5.4a shows how the test performs drastically inaccurately even when using input that is from a unit rate Poisson process with a large number of observations. Possibly the $\sqrt{T}$ term is a typographic error. An alternative test based on the Brownian motion approximation is suggested.

Say that $N(t)$ is a Poisson process of rate T. Define $M(t) = (N(t) - tT)/\sqrt{T}$ for $t \in [0,1]$. Donsker's invariance principle implies that as $T \to \infty$ then $(M(t) : t \in [0,1])$ converges in distribution to standard Brownian motion $(B(t) : t \in [0,1])$. Fig. 5.5 shows example realisations of $M(t)$ for various T that, at least qualitatively, are decent approximations to standard Brownian motion.

An alternative test it to utilise the first arcsine law for Brownian motion. That states that the random time $M^* \in [0,1]$, given by

$$M^* = \arg\max_{s \in [0,1]} B(s)\,,$$

is arcsine distributed (i.e. $M^* \sim \text{Beta}(1/2, 1/2)$).

Therefore the test takes a sequence of arrivals observed over $[0,T]$ and:

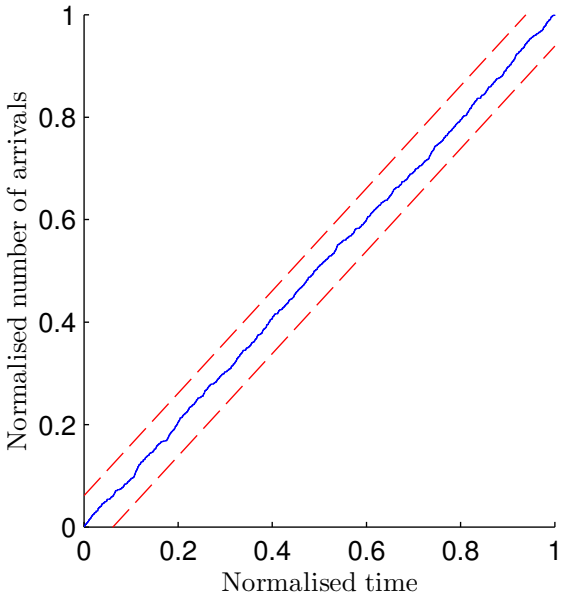


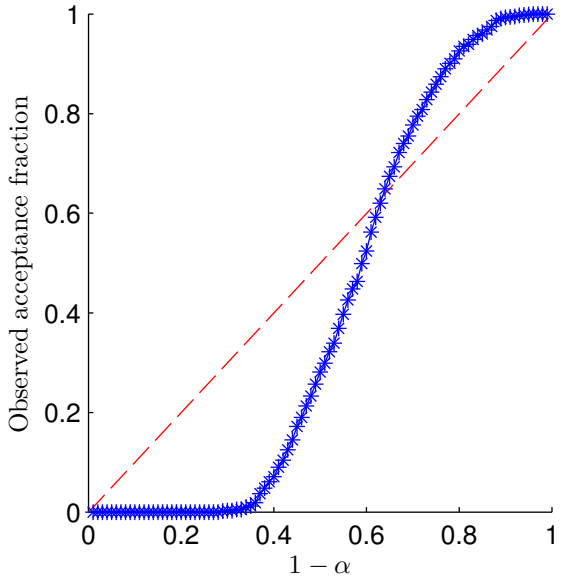
Figure 5.3: An example realisation of Algorithm 7.4.V of Daley and Vere-Jones (2003). The blue step function gives $Y(x)$, whereas the red dashed lines are boundaries of 95% confidence. The data source is 10^3 Hawkes process arrivals with $(\lambda, \alpha, \beta) = (0.5, 2, 2.1)$.

1. transforms the arrivals to $\{t_1^*/T, t_2^*/T, \ldots, t_k^*/T\}$ which should be a Poisson process of rate T over $[0, 1]$,

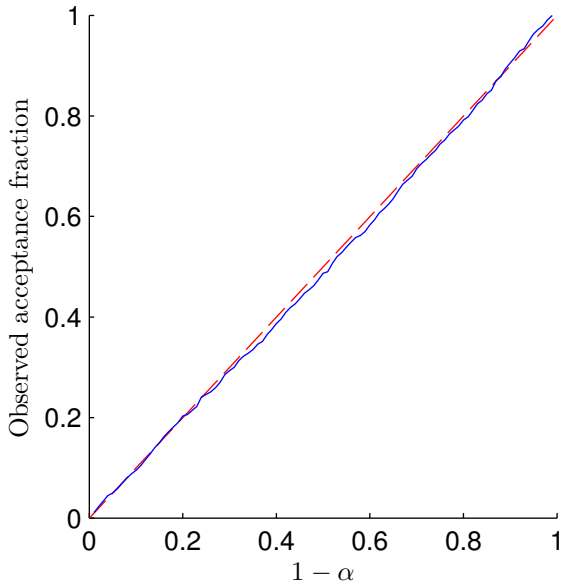
2. constructs the Brownian motion approximation $M(t)$ as above, finds the maximiser M^*, and

3. accepts the 'unit-rate Poisson process' hypothesis if M^* lies within the $(\alpha/2, 1-\alpha/2)$ quantiles of the Beta$(1/2, 1/2)$ distribution, otherwise reject it.

Fig. 5.4b shows that this algorithm performs as expected. As a final note, many other tests can be done based off properties of Brownian motion. For example, the test could simply note that $M(1) \sim \mathrm{N}(0, 1)$ and therefore accept if $M(1) \in [Z_{\alpha/2}, Z_{1-\alpha/2}]$ and reject otherwise.

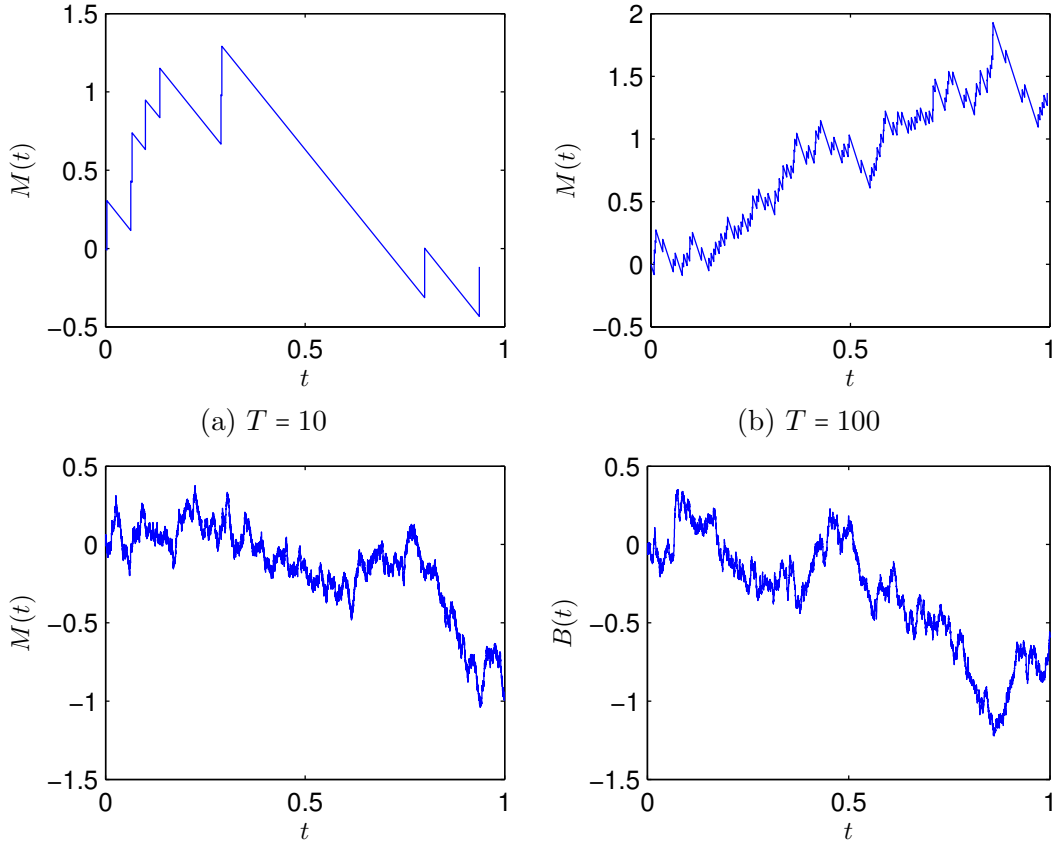


(a)



(b)

Figure 5.4: (a) Testing Algorithm 7.4.V of Daley and Vere-Jones (2003) and (b) testing proposed alternative algorithm. The algorithms were run over 10^3 simulated unit rate Poisson processes over time horizon $[0, 10^3]$. The blue crosses show the observed acceptance rate of each algorithm for various significance levels. The red dashed line shows the theoretical number of tests that should have been accepted at each threshold.



(a) $T = 10$ (b) $T = 100$

(c) $T = 10,0000$ (d) Actual Brownian motion

Figure 5.5: Realisations of Poisson process approximations to Brownian motion. Each approximation uses T, except (d) which is a direct simulation of Brownian motion for comparison.

Chapter 6

Simulation Methods

6.1 Transformation methods

For general point processes a simulation algorithm arises from the converse of the random time change theorem (given in Section 5.1). In essence, a unit rate Poisson process $\{t_1^*, t_2^*, \dots\}$ is transformed by the inverse compensator $\Lambda(\cdot)^{-1}$ into any general point process defined by that compensator. The method, sometimes called the *inverse compensator method*, iteratively solves the equations

$$t_1^* = \int_0^{t_1} \lambda^*(s)\,\mathrm{d}s, \quad t_{k+1}^* - t_k^* = \int_{t_k}^{t_{k+1}} \lambda^*(s)\,\mathrm{d}s$$

for $\{t_1, t_2, \dots\}$, the desired point process (see Giesecke and Tomecek 2005 and Algorithm 7.4.III of Daley and Vere-Jones 2003).

For Hawkes processes the algorithm was first suggested by Ozaki (1979) and did not explicitly state any relation to time changes. It instead focused on Eq. (4.1),

$$\int_{t_k}^{t} \lambda^*(u)\,\mathrm{d}u = -\log(1 - F^*(t))\,,$$

which relates the conditional c.d.f. of the next arrival to the previous history of arrivals $\{t_1, t_2, \dots, t_k\}$ and the specified $\lambda^*(t)$. This relation means the next arrival time T_{k+1} can easily be generated by the inverse transform method, i.e. draw $U \sim \mathrm{Unif}[0,1]$ then t_{k+1} is found by solving

$$\int_{t_k}^{t_{k+1}} \lambda^*(u)\,\mathrm{d}u = -\log(U)\,.$$

For an exponentially decaying intensity the equation becomes

$$\log(U) + \lambda(t_{k+1} - t_k) - \frac{\alpha}{\beta}\left(\sum_{i=1}^{k} \mathrm{e}^{\beta(t_{k-1}-t_i)} - \sum_{i=1}^{k} \mathrm{e}^{-\beta(t_k - t_i)}\right) = 0\,.$$

Solving for t_{k+1} can be done in linear time using the recursion of Eq. (4.6). However if a different excitation function is used then this equation must be solved numerically, e.g. using Newton's method (Ogata 1981), which entails a significant computational effort.

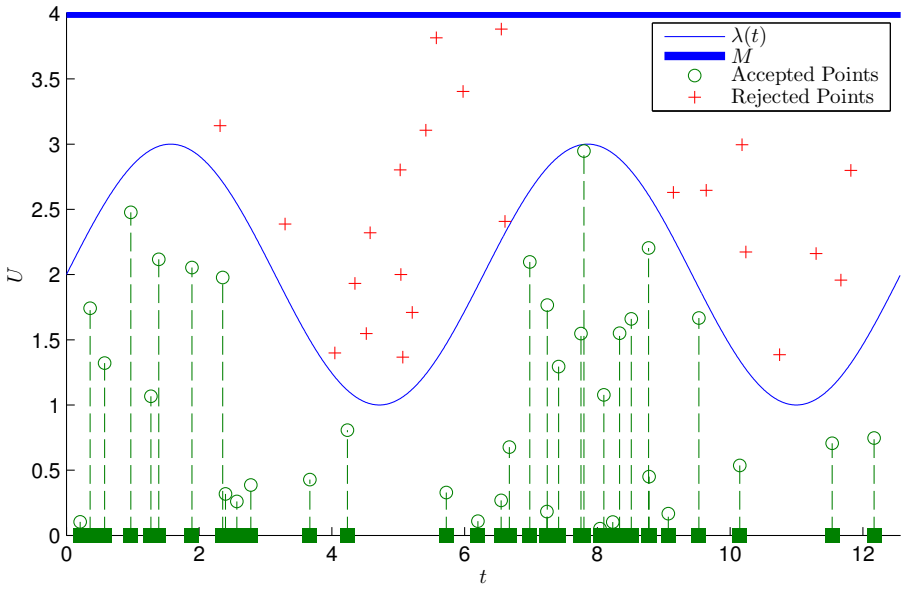
6.2 Ogata's modified thinning algorithm

Hawkes process generation is a similar problem to inhomogeneous Poisson process generation. The standard way to generate a inhomogeneous Poisson process which is driven by intensity function $\lambda(\cdot)$ is via thinning. Formally the process is described by Algorithm 1 (Lewis and Shedler 1979). The intuition is to generate a "faster" homogeneous Poisson process, and remove points probabilistically so that the remaining points satisfy the time-varying intensity $\lambda(\cdot)$. The first process' rate M cannot be less than $\lambda(\cdot)$ over $[0, T]$.

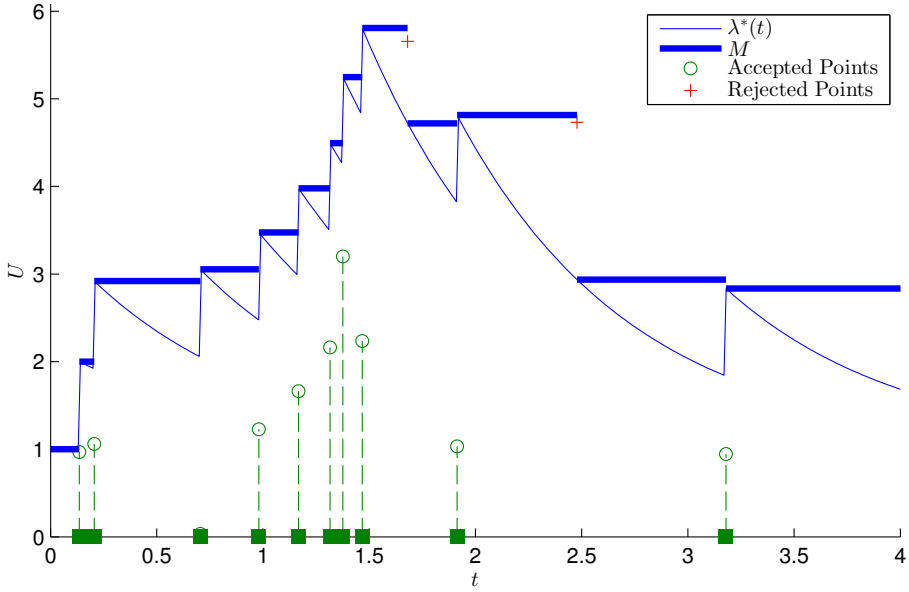
A similar approach can be used for the Hawkes process, called *Ogata's modified thinning algorithm* (Ogata 1981, Liniger 2009). The conditional intensity $\lambda^*(\cdot)$ does not have an a.s. asymptotic upper bound, however it is common for the intensity to be non-increasing in periods without any arrivals. This implies that for $t \in (T_i, T_{i+1}]$, $\lambda^*(t) \leq \lambda^*(T_i^+)$ (i.e. the time just after T_i, when that arrival has been registered). So the M value can be updated during each simulation. Algorithm 2 describes the process and Fig. 6.1 shows an example of each thinning procedure.

Algorithm 1 Generate an inhomogeneous Poisson process by thinning

procedure PoissonByThinning(T, $\lambda(\cdot)$, M)
  require: $\lambda(\cdot) \leq M$ on $[0, T]$
  $P \leftarrow []$, $t \leftarrow 0$.
  while $t < T$ **do**
    $E \leftarrow \text{Exp}(M)$.
    $t \leftarrow t + E$.
    $U \leftarrow \text{Unif}(0, M)$.
    if $t < T$ **and** $U \leq \lambda(t)$ **then**
      $P \leftarrow [P, t]$.
    end if
  end while
  return P
end procedure



(a) Poisson process



(b) Hawkes process

Figure 6.1: Processes generated by thinning. (a) A Poisson process with intensity $\lambda(t) = 2 + \sin(t)$, bounded above by $M = 4$. (b) A Hawkes process with $(\lambda, \alpha, \beta) = (1, 1, 1.1)$. Each (t, U) point describes a suggested arrival at time t whose U value is given in Algorithm 1 and Algorithm 2. Plus signs signs indicate rejected points, circles accepted, and green squares the resulting point processes.

Algorithm 2 Generate an Hawkes process by thinning

procedure HawkesByThinning(T, $\lambda^*(\cdot)$)
  require: $\lambda^*(\cdot)$ non-increasing in periods of no arrivals.
  $\varepsilon \leftarrow 10^{-10}$ (some tiny value > 0).
  $P \leftarrow []$, $t \leftarrow 0$.
  while $t < T$ **do**
    Find new upper bound: $M \leftarrow \lambda^*(t + \varepsilon)$.
    Generate next candidate point: $E \leftarrow \text{Exp}(M)$, $t \leftarrow t + E$.
    Keep it with some probability: $U \leftarrow \text{Unif}(0, M)$.
    if $t < T$ **and** $U \leq \lambda^*(t)$ **then**
      $P \leftarrow [P, t]$.
    end if
  end while
  return P
end procedure

6.3 Superposition of Poisson processes

The immigration–birth representation gives rise to a simple simulation procedure: generate the immigrant arrivals, then generate the descendants for each immigrant. Algorithm 3 describes the procedure in full, with Fig. 6.2 showing an example realisation.

Immigrants form a homogeneous Poisson process of rate λ, so over an interval $[0, T]$ the number of immigrants is $\text{Pois}(\lambda T)$ distributed. Conditioned on knowing that there are k immigrants, then their arrival times $C_1, C_2, \ldots, C_k$ are distributed as the order statistics of i.i.d. $\text{Unif}(0, T)$ random variables.

Each immigrant's descendants form an inhomogeneous Poisson process. The ith immigrant's descendants arrive with intensity $\mu(t - C_i)$ for $t > C_i$. Denote D_i to be the number of descendants of immigrant i, then $\mathbb{E}[D_i] = \int_0^\infty \mu(s)\, \mathrm{d}s = n$, and hence $D_i \overset{\text{i.i.d.}}{\sim} \text{Poi}(n)$. Say that the descendants of the ith immigrant arrive at times $(C_i + E_1, C_i + E_2, \ldots, C_i + E_{D_i})$. Conditioned on knowing D_i, E_j are i.i.d. random variables distributed with p.d.f. $\mu(\cdot)/n$. For exponentially decaying intensities, this simplifies to $E_j \overset{\text{i.i.d.}}{\sim} \text{Exp}(\beta)$.

Algorithm 3 Generate an Hawkes process by clusters

procedure HAWKESBYCLUSTERS(T, λ, α, β)
  $P \leftarrow \{\}$.
  Immigrants: $k \leftarrow \text{Poi}(\lambda T)$, $C_1, C_2, \ldots, C_k \overset{\text{i.i.d.}}{\leftarrow} \text{Unif}(0, T)$.
  Descendants: $D_1, D_2, \ldots, D_k \overset{\text{i.i.d.}}{\leftarrow} \text{Poi}(\alpha/\beta)$.
  for $i \leftarrow 1$ **to** k **do**
    if $D_i > 0$ **then**
      $E_1, E_2, \ldots, E_{D_i} \overset{\text{i.i.d.}}{\leftarrow} \text{Exp}(\beta)$.
      $P \leftarrow P \cup \{C_i + E_1, C_i + E_2, \ldots, C_i + E_{D_i}\}$.
    end if
  end for
  Remove descendants outside $[0, T]$: $P \leftarrow \{P_i : P_i \in P, P_i \leq T\}$.
  Add in immigrants and sort: $P \leftarrow \text{Sort}(P \cup \{C_1, C_2, \ldots, C_k\})$.
  return P
end procedure

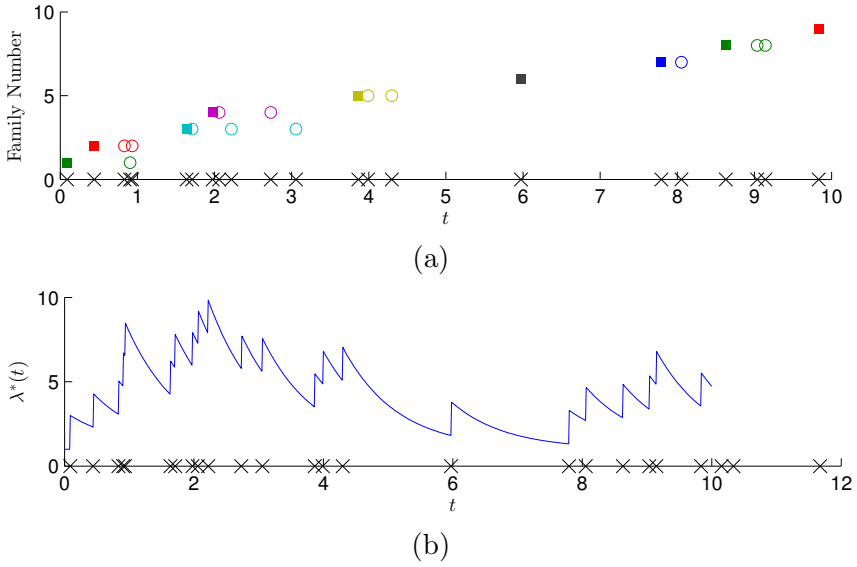


Figure 6.2: A Hawkes Poisson process generated by clusters. Plot (a) shows the points generated by the immigrant–birth representation; it can be seen as a sequence of vertically stacked "family trees". The immigrant points are plotted as squares, following circles of the same height and color are its offspring. The intensity function, with $(\lambda, \alpha, \beta) = (1, 2, 1.2)$, is plotted in (b). The resulting Hawkes process arrivals are drawn as crosses on the axis.

6.4 Other methods

This chapter's contents are by no means a complete compilation of simulation techniques available for Hawkes processes. Dassios and Zhao (2013) and Møller and Rasmussen (2005) are alternatives to the methods listed above (see Appendix C.2.4 for a MATLAB implementation of the former). Also not discussed is the problem of simulating mutually-exciting Hawkes processes, however there are many free software packages that will perform this functionality. Fig. 6.3 shows an example realisation generated using the R 'hawkes' package (see also Roger Peng's 'ptproc' package).

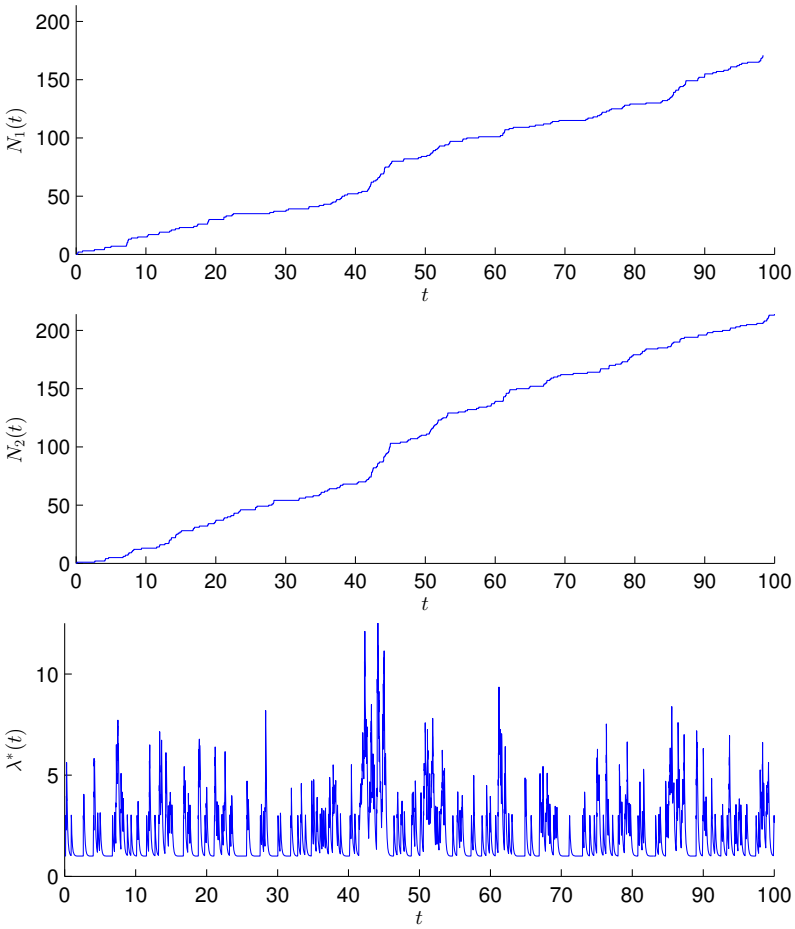


Figure 6.3: A pair of mutually exciting Hawkes processes $N_1(t)$ and $N_2(t)$ with parameters: $\lambda_1 = \lambda_2 = 1$, $\alpha_{1,1} = \alpha_{1,2} = \alpha_{2,1} = \alpha_{2,2} = 2$, $\beta_{1,1} = \beta_{1,2} = \beta_{2,1} = \beta_{2,2} = 8$. Note that the symmetry in parameters means that $\lambda_1^*(t) = \lambda_2^*(t)$ so only one is plotted.

Chapter 7
Conclusion

Hawkes processes are fundamentally fascinating models of reality. Standard probability theory models are Markovian and hence display a disregard for the history of the process rarely seen in nature. The Hawkes process is structured around the premise that the history matters, which partly explains why they have been studied in such a broad range of applications.

If the exponentially decaying intensity can be utilised, then the joint process $(N(\cdot), \lambda^*(\cdot))$ satisfies the Markov condition, and both processes exhibit amazing analytical tractability. Explosion is avoided by $\alpha < \beta$. The covariance density is a simple symmetric scaled exponential curve, and the power spectral density is a shifted scaled Cauchy p.d.f. The likelihood function and the compensator are elegant, and efficient to calculate using recursive structures. Exact simulation algorithms can generate this type of Hawkes process with optimal efficiency. Many aspects of the Hawkes process remain obtainable with any selection of excitation function; for example, the random time change theorem completely solves the problem of testing the goodness of a model's fit.

The use of Hawkes processes in finance appears itself to have been a self-exciting process. Filimonov and Sornette (2012), Aït-Sahalia et al. (2010), and Da Fonseca and Zaatour (2014) formed the primary sources for the financial section of Chapter 3; these papers are surprisingly recent (given the fact that model was introduced in 1971) and are representative of a current surge in Hawkes process research.

"So we beat on, boats against the current,
borne back ceaselessly into the past."[1]

[1] F. Scott Fitzgerald, "The Great Gatsby".

Bibliography

Aït-Sahalia, Y., Cacho-Diaz, J., and Laeven, R. J. (2010). Modeling financial contagion using mutually exciting jump processes. Technical Report 15850, National Bureau of Economic Research, USA.

Asmussen, S. (2003). *Applied Probability and Queues*. Applications of Mathematics: Stochastic Modelling and Applied Probability. Springer, second edition.

Azizpour, S., Giesecke, K., and Schwenkler, G. (2010). Exploring the sources of default clustering. Working paper.

Bartlett, M. S. (1963a). The spectral analysis of point processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 25(2):264–296.

Bartlett, M. S. (1963b). Statistical estimation of density functions. *Sankhyā: The Indian Journal of Statistics, Series A*, 25(3):245–254.

Bartlett, M. S. (1964). The spectral analysis of two-dimensional point processes. *Biometrika*, 51(3/4):299–311.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *The Journal of Political Economy*, 81(3):637–654.

Bowsher, C. G. (2007). Modelling security market events in continuous time: intensity based, multivariate point process models. *Journal of Econometrics*, 141(2):876–912.

Brémaud, P. and Massoulié, L. (1996). Stability of nonlinear Hawkes processes. *The Annals of Probability*, 24(3):1563–1588.

Brown, E., Barbieri, R., Ventura, V., Kass, R., and Frank, L. (2002). The time-rescaling theorem and its application to neural spike train data analysis. *Neural computation*, 14(2):325–346.

Carstensen, L. (2010). *Hawkes processes and combinatorial transcriptional regulation*. PhD thesis, University of Copenhagen.

Chatalbashev, V., Liang, Y., Officer, A., and Trichakis, N. (2007). Exciting times for trade arrivals. Stanford University MS&E 444 group project submission.

Cox, D. R. (1955). Some statistical methods connected with series of events. *Journal of the Royal Statistical Society. Series B (Methodological)*, 17(2):129–164.

Cox, D. R. and Lewis, P. A. (1966). *The Statistical Analysis of Series of Events*. Monographs on Applied Probability and Statistics, London: Chapman and Hall.

Crowley, S. (2013). Point process models for multivariate high-frequency irregularly spaced data. Available at `http://vixra.org/pdf/1211.0094v6.pdf`.

Da Fonseca, J. and Zaatour, R. (2014). Hawkes process: Fast calibration, application to trade clustering, and diffusive limit. *Journal of Futures Markets*, 34(6):548–579.

Daley, D. and Vere-Jones, D. (2003). *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*. Springer.

Dassios, A. and Zhao, H. (2013). Exact simulation of Hawkes process with exponentially decaying intensity. *Electronic Communications in Probability*, 18(62).

Durbin, J. (1961). Some methods of constructing exact tests. *Biometrika*, 53(3/4):41–55.

Dyson, F. (2004). A meeting with Enrico Fermi. *Nature*, 427(6972):297–297.

Embrechts, P., Liniger, T., and Lin, L. (2011). Multivariate Hawkes processes: an application to financial data. *Journal of Applied Probability*, 48A:367–378. Special volume: a Festschrift for Søren Asmussen.

Filimonov, V. and Sornette, D. (2012). Quantifying reflexivity in financial markets: toward a prediction of flash crashes. *Physical Review E*, 85(5):056108.

Filimonov, V. and Sornette, D. (2013). Apparent criticality and calibration issues in the Hawkes self-excited point process model: application to high-frequency financial data. *arXiv preprint arXiv:1308.6756*.

Giesecke, K. and Tomecek, P. (2005). Dependent events and changes of time. Working paper.

Grimmett, G. and Stirzaker, D. (2001). *Probability and Random Processes*. Oxford University Press.

Haug, E. G. and Taleb, N. N. (2014). Why we have never used the Black–Scholes–Merton option pricing formula. *Wilmott Magazine*, 71.

Hautsch, N. (2011). *Econometrics of Financial High-Frequency Data*. Springer.

Hawkes, A. G. (1971a). Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90.

Hawkes, A. G. (1971b). Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 33(3):438–443.

Hawkes, A. G. and Oakes, D. (1974). A cluster process representation of a self-exciting process. *Journal of Applied Probability*, 11(3):493–503.

Kallenberg, O. (1976). *Random measures*. Akademie-Verlag Berlin.

Kim, S.-H. and Whitt, W. (2013). The power of alternative Kolmogorov–Smirnov tests based on transformations of the data. Submitted to ACM Transactions on Modeling and Computer Simulation.

Knuth, D. E. (2014). *Art of Computer Programming, Volume 2: Seminumerical Algorithms, The*. Addison-Wesley Professional.

Kroese, D., Taimre, T., and Botev, Z. I. (2011). *Handbook of Monte Carlo methods*. Wiley.

Kroese, D. P. and Botev, Z. I. (2014). Spatial process generation. To appear in: Lectures on Stochastic Geometry, Spatial Statistics and Random Fields, Volume II: Analysis, Modeling and Simulation of Complex Structures.

Lewis, P. A. (1964). A branching Poisson process model for the analysis of computer failure patterns. *Journal of the Royal Statistical Society. Series B (Methodological)*, 26(3):398–456.

Lewis, P. A. (1965). Some results on tests for Poisson processes. *Biometrika*, 52(1/2):67–77.

Lewis, P. A. (1970). Remarks on the theory, computation and application of the spectral analysis of series of events. *Journal of Sound and Vibration*, 12(3):353–375.

Lewis, P. A. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413.

Liniger, T. J. (2009). *Multivariate Hawkes Processes*. PhD thesis, Eidgenössische Technische Hochschule ETH Zürich.

Lorenzen, F. (2012). *Analysis of Order Clustering Using High Frequency Data: A Point Process Approach*. PhD thesis, Swiss Federal Institute Of Technology Zurich.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1):125–144.

Meyer, P.-A. (1971). Demonstration simplifiee d'un theoreme de knight. In *Séminaire de Probabilités V Université de Strasbourg*, pages 191–195. Springer.

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

Møller, J. and Rasmussen, J. G. (2005). Perfect simulation of Hawkes processes. *Advances in Applied Probability*, pages 629–646.

Ogata, Y. (1978). The asymptotic behaviour of maximum likelihood estimators for stationary point processes. *Annals of the Institute of Statistical Mathematics*, 30(1):243–261.

Ogata, Y. (1981). On Lewis' simulation method for point processes. *Information Theory, IEEE Transactions on*, 27(1):23–31.

Ogata, Y. (1988). Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical Association*, 83(401):9–27.

Ogata, Y. (1999). Seismicity analysis through point-process modeling: A review. *Pure and Applied Geophysics*, 155(2/4):471–507.

Ozaki, T. (1979). Maximum likelihood estimation of Hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics*, 31(1):145–155.

Papangelou, F. (1972). Integrability of expected increments of point processes and a related random change of scale. *Transactions of the American Mathematical Society*, 165:483–506.

Rasmussen, J. G. (2009). Temporal point processes: the conditional intensity function. Course notes for 'rumlige punktprocesser' (spatial point processes).

Rasmussen, J. G. (2013). Bayesian inference for Hawkes processes. *Methodology and Computing in Applied Probability*, 15(3):623–642.

Rubin, I. (1972). Regular point processes and their detection. *Information Theory, IEEE Transactions on*, 18(5):547–557.

Schoenberg, F. (2002). On rescaled poisson processes and the brownian bridge. *Annals of the Institute of Statistical Mathematics*, 54(2):445–457.

Tankov, P. (2003). *Financial Modelling With Jump Processes*. Chapman & Hall/CRC Financial Mathematics Series. Taylor & Francis.

Watanabe, S. (1964). On discontinuous additive functionals and Lévy measures of a Markov process. *Japan. J. Math*, 34(53-70):82.

Zhu, L. (2013). Central limit theorem for nonlinear Hawkes processes. *Journal of Applied Probability*, 50(3):760–771.

Appendix A
Extra Proof Details

A.1 Supplementary to Theorem 2 (part one)

$$\begin{aligned}
R(\tau) &= \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\left(\lambda + \int_{-\infty}^{t+\tau} \mu(t+\tau-s)\,\mathrm{d}N(s)\right)\right] - \overline{\lambda^*}^2 \\
&= \lambda\mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\right] + \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\left(\int_{-\infty}^{t+\tau} \mu(t+\tau-s)\,\mathrm{d}N(s)\right)\right] - \overline{\lambda^*}^2 \\
&= \lambda\overline{\lambda^*} + \mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\int_{-\infty}^{t+\tau} \mu(t+\tau-s)\,\mathrm{d}N(s)\right] - \overline{\lambda^*}^2 .
\end{aligned}$$

Introduce a change of variable $v = s - t$ and multiply by $\frac{\mathrm{d}v}{\mathrm{d}v}$:

$$\begin{aligned}
R(\tau) &= \lambda\overline{\lambda^*} + \mathbb{E}\left[\int_{-\infty}^{\tau} \mu(\tau-v)\frac{\mathrm{d}N(t)}{\mathrm{d}t}\frac{\mathrm{d}N(t+v)}{\mathrm{d}v}\,\mathrm{d}v\right] - \overline{\lambda^*}^2 \\
&= \lambda\overline{\lambda^*} + \int_{-\infty}^{\tau} \mu(\tau-v)\mathbb{E}\left[\frac{\mathrm{d}N(t)}{\mathrm{d}t}\frac{\mathrm{d}N(t+v)}{\mathrm{d}v}\right]\mathrm{d}v - \overline{\lambda^*}^2 .
\end{aligned}$$

The expectation is (a shifted) $R^{(c)}(v)$. Substitute that and Eq. (3.6) in:

$$\begin{aligned}
R(\tau) &= \lambda\overline{\lambda^*} + \int_{-\infty}^{\tau} \mu(\tau-v)\big(R^{(c)}(v) + \overline{\lambda^*}^2\big)\,\mathrm{d}v - \overline{\lambda^*}^2 \\
&= \lambda\overline{\lambda^*} + \int_{-\infty}^{\tau} \mu(\tau-v)\left(\overline{\lambda^*}\delta(\tau) + R(\tau)\right)\mathrm{d}v + \overline{\lambda^*}^2\int_{-\infty}^{\tau} \mu(\tau-v)\,\mathrm{d}v - \overline{\lambda^*}^2 \\
&= \lambda\overline{\lambda^*} + \overline{\lambda^*}\mu(\tau) + \int_{-\infty}^{\tau} \mu(\tau-v)R(\tau)\,\mathrm{d}v + n\overline{\lambda^*}^2 - \overline{\lambda^*}^2 \\
&= \overline{\lambda^*}\mu(\tau) + \int_{-\infty}^{\tau} \mu(\tau-v)R(\tau)\,\mathrm{d}v + \overline{\lambda^*}(\lambda - (1-n)\overline{\lambda^*}) .
\end{aligned}$$

Using Eq. (3.5) yields

$$\lambda - (1-n)\overline{\lambda^*} = \lambda - (1-n)\frac{\lambda}{1-n} = 0 .$$

$$\therefore R(\tau) = \overline{\lambda^*}\mu(\tau) + \int_{-\infty}^{\tau} \mu(\tau-v)R(\tau)\,\mathrm{d}v .$$

A.2 Supplementary to Theorem 2 (part two)

Split the right-hand side of the equation into three functions g_1, g_2 and g_3:

$$R(\tau) = \underbrace{\overline{\lambda^*}\mu(\tau)}_{g_1(\tau)} + \underbrace{\int_0^\infty \mu(\tau+v)R(v)\,\mathrm{d}v}_{g_2(\tau)} + \underbrace{\int_0^\tau \mu(\tau-v)R(v)\,\mathrm{d}v}_{g_3(\tau)}. \tag{A.1}$$

Taking the Laplace transform of each term gives

$$\begin{aligned}
\mathscr{L}\left\{g_1(\tau)\right\}(s) &= \int_0^s \mathrm{e}^{-s\tau}\overline{\lambda^*}\alpha\mathrm{e}^{-\beta\tau}\,\mathrm{d}\tau = \frac{\alpha}{s+\beta}\overline{\lambda^*}, \\
\mathscr{L}\left\{g_2(\tau)\right\}(s) &= \int_0^\infty \mathrm{e}^{-s\tau}\int_0^\infty \alpha\mathrm{e}^{-\beta(\tau+v)}R(v)\,\mathrm{d}v\,\mathrm{d}\tau \\
&= \alpha\int_0^\infty \mathrm{e}^{-\beta v}R(v)\int_0^\infty \mathrm{e}^{-\tau(s+\beta)}\,\mathrm{d}\tau\,\mathrm{d}v \\
&= \frac{\alpha}{s+\beta}\int_0^\infty \mathrm{e}^{-\beta v}R(v)\,\mathrm{d}v \\
&= \frac{\alpha}{s+\beta}\mathscr{L}\left\{R\right\}(\beta), \text{ and} \\
\mathscr{L}\left\{g_3(\tau)\right\}(s) &= \mathscr{L}\left\{\mu(\tau)\right\}(s)\mathscr{L}\left\{R(\tau)\right\}(s) = \frac{\alpha}{s+\beta}\mathscr{L}\left\{R(\tau)\right\}(s).
\end{aligned}$$

Therefore the Laplace transform of Eq. (A.1)

$$\mathscr{L}\left\{R(\tau)\right\}(s) = \frac{\alpha}{s+\beta}\left(\overline{\lambda^*} + \mathscr{L}\left\{R(\tau)\right\}(\beta) + \mathscr{L}\left\{R(\tau)\right\}(s)\right). \tag{A.2}$$

Substituting $s = \beta$ and rearranging gives that

$$\mathscr{L}\left\{R(\tau)\right\}(\beta) = \frac{\alpha\overline{\lambda^*}}{2(\beta-\alpha)}. \tag{A.3}$$

So substituting the value of $\mathscr{L}\left\{R(\tau)\right\}(\beta)$ into Eq. (A.2) means

$$\begin{aligned}
\mathscr{L}\left\{R(\tau)\right\}(s) &= \frac{\alpha}{s+\beta}\left(\overline{\lambda^*} + \frac{\alpha\overline{\lambda^*}}{2(\beta-\alpha)} + \mathscr{L}\left\{R(\tau)\right\}(s)\right) \\
\Rightarrow \mathscr{L}\left\{R(\tau)\right\}(s) &= \frac{\frac{\alpha}{s+\beta}\left(\overline{\lambda^*} + \frac{\alpha\overline{\lambda^*}}{2(\beta-\alpha)}\right)}{1-\frac{\alpha}{s+\beta}} \\
\therefore \mathscr{L}\left\{R(\tau)\right\}(s) &= \frac{\alpha\overline{\lambda^*}(2\beta-\alpha)}{2(\beta-\alpha)(s+\beta-\alpha)}.
\end{aligned}$$

Appendix B

Open Source Contributions

B.1 R package: 'Hawkes'

The R 'Hawkes' package uses C++ to compute likelihoods. This particular operation has been optimised to be 20–30 times faster. Below is the optimised function (revision 63, `https://r-forge.r-project.org/projects/hawkes/`):

```
1  // [[Rcpp::export]]
2  double  likelihoodHawkes(SEXP lambda0,SEXP alpha,SEXP beta,SEXP ↙
       ↳ history)
3  {
4    int dimension = getDimension(lambda0);
5
6
7    double res = 0;
8
9    if (dimension == 1)
10   {
11     double m_lambda0 = as<double>(lambda0);
12     double m_alpha = as<double>(alpha);
13     double m_beta = as<double>(beta);
14     Rcpp::NumericVector m_history(history);
15     double m_T = m_history[m_history.size()-1];
16
17     if(m_beta<m_alpha){
18       stop("Unstable. You must have alpha < beta");
19     }
20     double *A = new double[m_history.size()];
21     A[0] = 0;
22     for (int i = 1; i < m_history.size(); i++)
23     {
24         A[i] = (1.0+A[i-1])*exp(-m_beta * (m_history[i] - ↙
               ↳ m_history[i-1])) ;
25     }
26
27     double sum = 0.0;
28     for (int i = 0; i < m_history.size(); i++)
```

```
29      {
30          sum = sum+    (1 - exp(-m_beta * (m_T - m_history[i]))) ;
31      }
32      sum = (m_alpha / m_beta) * sum;
33      res = - m_lambda0 * m_T - sum;
34
35
36      for (int i = 0; i < m_history.size(); i++)
37      {
38          res = res + log(m_lambda0+m_alpha*A[i]);
39      }
40      delete [] A;
41
42        return (-res);
43    }else{
44      Rcpp::NumericVector lambda0_internal(lambda0);
45      Rcpp::NumericMatrix alpha_internal(alpha);
46      Rcpp::NumericVector beta_internal(beta);
47      arma::vec m_lambda0(lambda0_internal.begin(), dimension, ↙
            ↳ false);
48      arma::mat m_alpha(alpha_internal.begin(), dimension, ↙
            ↳ dimension, false);
49      arma::vec m_beta(beta_internal.begin(), dimension, false);
50      Rcpp::List m_history(history);
51
52      double m_T = 0;
53      int *sizes = new int[dimension];
54      Rcpp::NumericVector *vectors = new ↙
            ↳ Rcpp::NumericVector[dimension];
55      for (int n = 0; n < dimension; n++)
56      {
57        vectors[n] = as<Rcpp::NumericVector>(m_history[n]);
58        sizes[n] = vectors[n].size();
59        m_T = std::max(vectors[n][sizes[n]-1],m_T);
60      }
61
62      for (int m = 0; m < dimension; m++)
63      {
64          double sum = 0.0;
65
66        double *Rdiag = new double[sizes[m]];
67          double *RNonDiag = new double[sizes[m]];
68          int * index=new int[dimension];
69          for (int n = 0; n < dimension; n++)
70          {
71              index[n] = 0;
```

```
72          }
73          Rdiag[0] = 0;
74          RNonDiag[0] = 0;
75          for (int i = 1; i < sizes[m]; i++)
76          {
77              Rdiag[i] = (1.0+Rdiag[i-1])*exp(-m_beta(m) * ↙
                ↳ (vectors[m][i] - vectors[m][i-1])) ;
78          }
79          for (int i = 1; i < sizes[m]; i++)
80          {
81              RNonDiag[i] = (RNonDiag[i-1])*exp(-m_beta(m) * ↙
                ↳ (vectors[m][i] - vectors[m][i-1])) ;
82              for (int n = 0; n < dimension; n++)
83              {
84                  if (m==n)
85                  {
86                      continue;
87                  }
88                  for (int k = index[n]; k < sizes[n]; k++)
89                  {
90                      if (vectors[n][k] >= vectors[m][i-1])
91                      {
92                          if (vectors[n][k] < vectors[m][i])
93                          {
94                              RNonDiag[i] += exp(-m_beta(m) * ↙
                                ↳ (vectors[m][i] - vectors[n][k]));
95                          }
96                          else
97                          {
98                              index[n] = k;
99                              break;
100                         }
101                     }
102                 }
103             }
104         }
105         for (int n = 0; n < dimension; n++)
106         {
107             for (int k = 0; k < sizes[n]; k++)
108             {
109                 sum = sum + (m_alpha(m,n) / m_beta(m)) *
110                     (1-exp(-m_beta(m) * (m_T - ↙
                        ↳ vectors[n][k])));//Beta diagonal
111
112             }
113         }
```

```
114
115         res = res - m_lambda0(m) * m_T - sum;
116
117
118           for (int i = 0; i < sizes[m]; i++)
119           {
120               sum = m_lambda0(m);
121               for (int n = 0; n < dimension; n++)
122               {
123
124                   if(m==n)
125                   {
126                       sum = sum+m_alpha(m,n)*Rdiag[i];
127                   }
128                   else
129                   {
130                       sum = sum +m_alpha(m,n)*RNonDiag[i];
131                   }
132               }
133               res = res+log(sum);
134           }
135           delete[] Rdiag;
136           delete[] RNonDiag;
137           delete[] index;
138       }
139       delete[] sizes;
140       delete[] vectors;
141
142         return (-res);
143     }
144 }
```

B.2 Hawkes explanatory article

A very gentle introduction to Hawkes processes using R and Python can be found online at `http://jheusser.github.io/2013/09/08/hawkes.html`. However the corresponding example code had become stale as of April 2014. The following figure shows the git diff of the commit which fixed that example (commit hash: caebd645c1a3e310276379f4477ccdbf585b4656, modified file: hawkes.py):

```
45   ## plot settings
46   import matplotlib as mpl
47  +import matplotlib.pyplot as plt
48   mpl.rc('font', **{'sans-serif':'Verdana','family':'sans-serif','size':8})
49   mpl.rcParams['xtick.direction'] = 'out'
50   mpl.rcParams['ytick.direction'] = 'out'
     @@ -84,7 +85,7 @@ def rate_series(t, x):
85   ax.set_title('Fitted vs Empirical Intensities')
86
87   plt.draw()
     -
88  +plt.show()
89
90   ## QQ plot of residuals
91   import scipy.stats as stats
     @@ -94,4 +95,4 @@ def rate_series(t, x):
95   ax.set_title('Residual Interevent Times')
96
97   plt.draw()
     -
98  +plt.show()
```

Appendix C

MATLAB Implementations

C.1 Goodness of fit

C.1.1 Multiple tests

To generate Fig. 5.1, Fig. 5.2, and Fig. 5.3, and to run the described hypothesis tests:

```
%% Setup work environment
clear all; rng(7);

% Simulate and plot the counting process.
figure(1); clf; hold on;

% Hawkes parameters.
lambda = 0.5; % Background intensity.
alpha = 2; % Jump in intensity after arrival.
beta = 2.1; % Decay rate.
k = 1000; % Number of jump times to look at.

% Simulate the process and plot it.
T = SimulateHawkes(lambda, alpha, beta, k);
stairs([0 T], 0:numel(T));

% Discretise time.
t = linspace(0, max(T), 1e3);

% Plot the expectation of N(t).
kappa = beta - alpha;
EN = (lambda*beta/kappa)*t +  ...
     (lambda/kappa)*(1-1*beta/kappa)*(1-exp(-kappa*t));
plot(t, EN, 'r--');

axis([0 max(T) 0 numel(T)]);
legend({'$N(t)$', '$\mathbb{E}[N(t)]$'}, 'Location', ↙
    ↳ 'NorthWest', 'interpreter', 'latex');
drawnow;
set(gcf,'OuterPosition',[500,500,300,300]);
```

```
30 matlab2tikz('..\images\goodness1.tikz');
31
32 %% Plot the conditional intensity.
33 figure(2); clf; hold on;
34
35 % Plot lambda(t).
36 both = sort([t, (T+0.1)]);
37 lboth = cif(both, T, lambda, alpha, beta);
38 l = cif(t, T, lambda, alpha, beta);
39 plot(both, lboth);
40
41 % Plot its expectation.
42 El = (lambda*beta/kappa) + lambda*(1-beta/kappa)*exp(-kappa*t);
43 plot(t, El, 'r--');
44
45 axis([0 max(T) 0 max(l)]);
46 legend({'$\lambda^*(t)$', '$\mathbb{E}[\lambda^*(t)]$'}, ↙
       ↳ 'Location', 'NorthWest', 'interpreter', 'latex');
47 set(gcf,'OuterPosition',[500,900,300,300]); drawnow;
48 matlab2tikz('..\images\goodness2.tikz');
49
50 %% Plot the compensator.
51 figure(3); clf; hold on;
52
53 comp = zeros(1, numel(t));
54 for i=1:numel(t)
55     comp(i) = compensator(t(i),[lambda, alpha, beta], T);
56 end
57 plot(t, comp, 'm');
58
59 axis([0 max(T) 0 max(comp)]);
60 legend({'$\Lambda(t)$'}, 'Location', 'NorthWest', 'interpreter', ↙
       ↳ 'latex');
61 set(gcf,'OuterPosition',[500,500,300,300]); drawnow;
62 matlab2tikz('..\images\goodness3.tikz');
63
64 %% Transform the arrival times by the compensator to get unit ↙
       ↳ rate PP.
65 figure(4); clf; hold on;
66
67 % Calculate the compensator exactly for the original arrival times.
68 unitT = zeros(1, numel(T));
69 for i=1:numel(T)
70     unitT(i) = compensator(T(i),[lambda, alpha, beta], T);
71 end
72
```

```
% Plot transformed PP.
stairs([0 unitT], 0:numel(unitT));

% Plot the expected unit rate PP.
plot(0:ceil(max(unitT)), 0:ceil(max(unitT)), 'r--');

axis([0 max(unitT) 0 numel(unitT)]);
legend({'$N^*(t)$', '$\mathbb{E}[N^*(t)]$'}, 'Location', 'NorthWest', 'interpreter', 'latex');
set(gcf,'OuterPosition',[500,500,300,300]); drawnow;
matlab2tikz('..\images\goodness4.tikz');

% MLE Estimate for rate assuming tranformed into a PP
lHat = numel(unitT) / max(unitT);
fprintf('Assuming this is a PP, estimate that lambda = %g\n', lHat);

%% Hypothesis test: null H is lambda = 1, alt H is lambda != 1.
% test statistic T = n_0 = numel(unitT), assuming t_0 = max(unitT) is
% fixed. Then T ~ Pois(t_0).
n_0 = numel(unitT);
t_0 = unitT(numel(unitT)-1);
p = 2 * min([poisscdf(n_0, t_0) (1-poisscdf(n_0-1, t_0))]);
fprintf('Null hyp: This is a unit rate PP, p value = %g\n', p);
if p < 0.1 || isnan(p)
    fprintf('Failure!\n');
else
    fprintf('Success!\n');
end

%% Q-Q plot to check interarrival times are exponentially distributed.
figure(5); clf; hold on;

% Specify the quantiles to display.
quants = 0.01:0.01:0.99;

% Calculate the expected theoretical result from Exp(1).
x = -log(1-quants);

% Find empirical quantiles from the data.
y = quantile(diff(unitT), quants);

% Plot them against each other. Should get points around y=x.
scatter(x, y);
```

```
115 plot([0 max([x y])], [0 max([x y])], 'r');
116 axis([0 max([x y]) 0 max([x y])]);
117 xlabel('Expected', 'interpreter', 'latex');
118 ylabel('Observed', 'interpreter', 'latex');
119
120 set(gcf,'OuterPosition',[500,500,300,300]); drawnow;
121 matlab2tikz('..\images\goodness5.tikz');
122
123 %% Test for autocorrelation in transformed interarrival times.
124 % Reference: Lorenzen (2012) S4.2 pg 26.
125 figure(6); clf;
126
127 % Interarrival times
128 iit = diff(unitT);
129
130 % Transform via c.d.f. of Exp(1) r.v.
131 unif = 1 - exp(-iit);
132
133 % Plot U_k vs U_{k+1}. Should get 2D points uniformly on [0,1]^2.
134 scatter(unif(1:(numel(unif)-1)), unif(2:numel(unif)), '*');
135 xlabel('$U_k = F_{\mathrm{Exp}(1)}(t^*_k-t^*_{k-1})$', ↙
    ↳ 'interpreter', 'latex');
136 ylabel('$U_{k+1} = F_{\mathrm{Exp}(1)}(t^*_{k+1}-t^*_k)$', ↙
    ↳ 'interpreter', 'latex');
137 drawnow;
138 set(gcf,'OuterPosition',[500,500,300,300]); drawnow;
139 matlab2tikz('..\images\goodness6.tikz');
140
141 %% Goodness of fit test: Kolmogorov-Smirnov test
142 % Reference: Daley, Vere-Jones Algorithm 7.4V pg 262
143 figure(7); clf; hold on;
144 alpha = 0.05;
145
146 stairs([0, unitT ./ max(unitT)], (0:numel(unitT)) ./ numel(unitT));
147 ciWidth = norminv(1-alpha/2)/sqrt(max(unitT));
148 plot([0, 1], [ciWidth, 1+ciWidth], 'r--');
149 plot([0, 1], [-ciWidth, 1-ciWidth], 'r--');
150
151 xlabel('Normalised time', 'interpreter', 'latex');
152 ylabel('Normalised number of arrivals', 'interpreter', 'latex');
153 axis([0,1,0,1]);
154 drawnow;
155 set(gcf,'OuterPosition',[500,500,400,400]);
156 export_fig('..\images\broken_test.pdf', '-pdf', '-transparent');
157
158 %% Good of fit test: Lewis Test
```

```
159 % References:
160 %   - The Power of Alternative Kolmogorov-Smirnov Tests Based on
161 %     Transformations of the Data, Song-Hee Kim, Ward Whitt,
162 %   - Some results on tests for Poisson processes By PETER A. W. ↙
      ↳ LEWIS
163 %
164 % Conditional uniform: null hyp is X_k, k=1,.., n are ↙
      ↳ distribution F
165 % U_k = F(X_k) for k=1,.., n ~ U[0,1]
166 % Y_k = -log(1-U_k) for k=1,.., n ~ Exp(1), view as IA times of ↙
      ↳ PP(1)
167 % T_k = sum_{i<=k} Y_i, for k=1,..,n, view as arrival times of ↙
      ↳ PP(1)
168 % Use T_k/T_n k=1,.., n-1 ~ Unif[0,1].
169 %
170 % Lewis test: null hyp is X_k, k=1,.., n are distribution F
171 % Use T_k/T_n k=1,.., n-1 ~ Unif[0,1].
172 % U_k = F(X_k) for k=1,.., n ~ U[0,1]
173 % Y_k = -log(1-U_k) for k=1,.., n ~ Exp(1), view as IA times of ↙
      ↳ PP(1)
174 % T_k = sum_{i<=k} Y_i, for k=1,..,n, view as arrival times of ↙
      ↳ PP(1)
175 % U_k = T_k/T_n k=1,.., n-1 ~ Unif[0,1].
176 %
177 % Now apply Durbin's modification (for n-1 points instead of n)
178 % U_(k) k=1,...,n-1 are ordered, set U_(0) = 0, U(n) = 1.
179 % X'_k = (n+1-k) (U_(k)-U_(k-1)) k=1,...,n ~ Exp(1)
180 % T'_k = sum_{i<=k} X'_i, for k=1,..,n, view as arrival times of ↙
      ↳ PP(1)
181 % Use T_k/T_n k=1,.., n-1 ~ Unif[0,1].
182 % Then calculate C(X) = Coefficient of Variation = std(X)/mean(X)
183 % If C(diff(unitT)) > 1 use upper KS test D+_n
184 % If C(diff(unitT)) < 1 use lower KS test D-_n
185
186 % Use Monte Carlo simulation to find an approx. dist. for KS test.
187 % Reference: Statistical Modeling & Computation, Kroese & Chan
188 K = 10000; % Number of trials
189 DN = zeros(1, K);
190 n = numel(unitT);
191 for k=1:K
192    i = 1:n;
193    y = sort(rand(1, n));
194    DN(k) = max( max(abs(y-i/n)), max(abs(y-(i-1)/n)) );
195 end
196
197 R = quantile(DN, 0.20);
```

```
198
199 % MY CASE: null hypothesis is that X_1, X_2, ... X_N are from a ↙
      ↳ Hawkes
200 % process. Daley-Vere Jones says that w.p 1 then tau_i = ↙
      ↳ Lambda*(t_i) is
201 % is unit rate PP iff t_i is a realization from the point ↙
      ↳ process defined
202 % by Lambda*(t_i). In the code tau_i is stored in "unitT".
203
204 % New null hypothesis: T_i, i = 1,..., n are from unit rate PP.
205 % Use something like conditional uniformity to transform this ↙
      ↳ problem into
206 % U_k = T_k/T_n k=1,.., n-1 ~ Unif[0,1].
207 % Next apply Durbin's modification
208
209 % Apply conditional uniformity to transform this problem into
210 % U_k = T_k/T_n k=1,.., n-1 ~ Unif[0,1].
211 NT = numel(unitT);
212 U = unitT(1:(NT-1)) ./ unitT(NT);
213
214 % Reset so that U_k is defined for k=1,..,n. It is already sorted.
215 n = numel(U);
216
217 % C = U_(j) - U_(j-1) 2 <= j <= n+1, C_1 = U_(1), C_(n+1) = 1-U_(n)
218 C = diff([0 U 1]);
219
220 % X'_k = (n+2-k) (C_(k)-C_(k-1)) k=1,...,n+1 ~ Exp(1)
221 k=1:(n+1);
222 XPrime = (n+2-k) .* diff(sort([0 C]));
223
224 % T'_k = sum_{i<=k} X'_i, for k=1,..,n, view as arrival times of ↙
      ↳ PP(1)
225 TPrime = cumsum(XPrime);
226
227 % Use U' = T'_k/T'_n k=1,.., n-1 ~ Unif[0,1].
228 UPrime = TPrime(1:(n-1)) / TPrime(n);
229
230 figure(5);
231 clf(5);
232
233 subplot(2, 1, 1);
234 hold on;
235 ecdf(U);
236 title('Conditional Uniformity Test for Poissonness');
237 plot([0 1], [R 1+R], 'r');
238 plot([0 1], [-R 1-R], 'r');
```

```
axis([0 1 0 1]);
legend('Original ECDF (Uniform)', 'Approx. 80% CI', ...
    'Location', 'NorthWest');

subplot(2, 1, 2);
hold on;
ecdf(UPrime);
title('Extended Test for Poissonness (includes Durbins ↙
    ↳ Modification)');
plot([0 1], [R 1+R], 'r');
plot([0 1], [-R 1-R], 'r');
axis([0 1 0 1]);
legend('Transformed ECDF (Uniform)', 'Approx. 80% CI', ...
    'Location', 'NorthWest');

% Now apply Kolmogorov-Smirnov test to UPrime.
n = n-1; % now disregarding the last point
dn_plus = max(((1:n)./n) - UPrime);
dn_minus = max(UPrime - ((0:(n-1))./n));
dn = max([dn_plus dn_minus]);

% Estimate the p value for this statistic
p = sum(DN >= dn) / K;
fprintf('Performed test and found p=%g\n', p);
if p >= 0.20
    fprintf('Accept null hypothesis, this is a Hawkes Process!\n');
else
    fprintf('Reject null hypothesis, this is not a Hawkes ↙
        ↳ Process!\n');
end
```

C.1.2 Brownian motion approximation

To generate Fig. 5.4a and Fig. 5.5:

```
%% Testing Daley, Vere-Jones Algorithm 7.4.V (p. 262).
rng(1);
alphas = 0.01:0.01:0.99;

% Observed time horizon.
T = 1e3;

% Number of tests to perform for each alpha value.
numTests = 1e3;
```

```
10
11 % Create a number of Poisson processes; create 2*T interarrival ↙
       ↳ times.
12 e = exprnd(1, 2*T, numTests);
13
14 % Take the cumulative sum to get arrival times.
15 tMany = cumsum(e);
16
17 % The test relies on a scaled stair function, so preprocess this ↙
       ↳ stair
18 % function for each Poisson process before the tests.
19 stairVals = cell(numTests, 3);
20
21 % Also collect the maximiser of the compensated scaled Poisson ↙
       ↳ process.
22 Ms = zeros(numTests, 1);
23
24 for i=1:numTests
25     % Extract the arrival times, and crop those after the obs. ↙
           ↳ window.
26     points = tMany(:,i); points = points(points < T);
27     k = numel(points);
28
29     % Create the step function.
30     x = points ./ T;
31     y = (1:k) ./ k;
32     [xx, yy] = stairs(x, y);
33
34     % Store these values.
35     stairVals{i,1} = xx;
36     stairVals{i,2} = yy;
37
38     % Create the compensated scaled Poisson process.
39     stairVals{i,3} = (yy.*k - (xx .*T))./sqrt(T);
40
41     % Find the maximiser of the compensated scaled Poisson process.
42     Ms(i) = xx(stairVals{i,3} == max(stairVals{i,3}));
43 end
44
45 observedPasses = zeros(numel(alphas), 1);
46
47 for i = 1:numel(alphas)
48     % Significance level: calculate z value.
49     alpha = alphas(i); z = norminv(1-alpha/2);
50
```

```
51      % Binary score for whether the corresponding test passed or ↙
            ↳ failed.
52      results = zeros(numTests, 1);
53
54      % For each Poisson process see whether it lies within the ↙
            ↳ confidence
55      % bands.
56      for test=1:numTests
57          ciWidth = norminv(1-alpha/2)/sqrt(T);
58          yUp = stairVals{test,1} + ciWidth;
59          yDown = stairVals{test,1} - ciWidth;
60
61          isAbove = any(stairVals{test,2} > yUp);
62          isBelow = any(stairVals{test,2} < yDown);
63          if ~(isAbove || isBelow)
64              results(test) = 1;
65          end
66      end
67
68      observedPasses(i) = sum(results) / numTests;
69  end
70
71  % Plot the results.
72  figure(1); clf; hold on;
73  plot([0,1], [0,1], 'r--');
74  plot(1-alphas, observedPasses, 'b*-');
75
76  xlabel('$1-\alpha$', 'interpreter', 'latex');
77  ylabel('Observed acceptance fraction', 'interpreter', 'latex');
78  set(gcf,'OuterPosition',[500,500,400,400]);
79  export_fig('..\images\bm_approx_test.pdf', '-pdf', '-transparent');
80
81  %% Fixed algorithm.
82  % Plot one of the 'Brownian motions'.
83  figure(2); clf; plot(stairVals{1,1}, stairVals{1,3});
84  set(gcf,'OuterPosition',[500,500,400,400]);
85  export_fig('..\images\compensated_pp.pdf', '-pdf', '-transparent');
86
87  fixedObservedPasses = zeros(numel(alphas), 1);
88
89  for i = 1:numel(alphas)
90      % Significance level: calculate z value.
91      alpha = alphas(i);
92
93      % Binary score for whether the corresponding test passed or ↙
            ↳ failed.
```

```
94      results = zeros(numTests, 1);
95
96      % 100(1-alpha)% confidence interval for the location of the ↙
            ↳ maximum.
97      lower = betainv(alpha/2, 0.5, 0.5);
98      upper = betainv(1-alpha/2, 0.5, 0.5);
99
100     % For each 'Brownian motion' see whether it's maximiser lies ↙
            ↳ within
101     % the confidence inveral.
102     for test=1:numTests
103         % Check whether inside the confidence intervals.
104         m = Ms(test); %max(stairVals{test, 2});
105
106         if m >= lower && m <= upper
107             results(test) = 1;
108         end
109     end
110
111     fixedObservedPasses(i) = sum(results) / numTests;
112 end
113
114 % Plot the results.
115 figure(3); clf; hold on;
116 plot([0,1], [0,1], 'r--');
117 plot(1-alphas, fixedObservedPasses, 'b-');
118
119 xlabel('$1-\alpha$', 'interpreter', 'latex');
120 ylabel('Observed acceptance fraction', 'interpreter', 'latex');
121 set(gcf,'OuterPosition',[500,500,400,400]);
122 export_fig('..\images\fixed_bm_approx_test.pdf', '-pdf', ↙
        ↳ '-transparent');
123
124 %% Plot compensated Poisson processes.
125 rng(4);
126 Ts = [10, 100, 1e4];
127 for i=1:numel(Ts);
128     T = Ts(i);
129
130     % Create a number of Poisson processes; create 2*T ↙
            ↳ interarrival times.
131     e = exprnd(1, 2*T, 1);
132
133     % Take the cumulative sum to get arrival times, and crop ↙
            ↳ those after
134     % the obs. window.
```

```
135     points = cumsum(e); points = points(points < T);
136     k = numel(points);
137
138     % Create the step function.
139     x = [0; points ./ T];
140     y = [0; (1:k)' ./ k];
141     [xx, yy] = stairs(x, y);
142
143     % Create the compensated scaled Poisson process.
144     compPP = (yy.*k - (xx .*T))./sqrt(T);
145
146     % Plot the process.
147     figure(3+i); clf;
148     plot(xx, compPP); %title(sprintf('T = %d', T));
149     xlabel('$t$', 'interpreter', 'latex');
150     ylabel('$M(t)$', 'interpreter', 'latex');
151
152     % Save it.
153     set(gcf,'OuterPosition',[500,500,300,300]);
154     name = sprintf('..\\images\\comp_pp_%d.pdf', T);
155     export_fig(name, '-pdf', '-transparent');
156 end
157
158 % Plot actual Brownian motion for comparison.
159 figure(6); clf; dt = 1e-4;
160 bm = cumsum(normrnd(0, sqrt(dt), 1/dt, 1));
161 plot(linspace(0, 1, numel(bm)), bm);
162 xlabel('$t$', 'interpreter', 'latex');
163 ylabel('$B(t)$', 'interpreter', 'latex');
164 set(gcf,'OuterPosition',[500,500,300,300]);
165 export_fig('..\images\comp_pp_infinity.pdf', '-pdf', ↙
        ↳ '-transparent');
```

C.2 Simulation methods

C.2.1 Inhomogeneous Poisson process by thinning

To generate Fig. 6.1a:

```
rng(2);
lambda = @(t) 2 + sin(t);
M = 4;
T = 4*pi;

p = []; py = [];

r = []; ry = [];

t = 0;

while t < T
    t = t + exprnd(1/M);
    if t < T
        u = rand();
        if u <= lambda(t)/M
            p = [p, t];
            py = [py, M*u];
        else
            r = [r, t];
            ry = [ry, M*u];
        end
    end
end

figure(1); clf; hold on;
t = 0:0.01:T;
xlabel('$t$', 'interpreter', 'latex');
ylabel('$U$', 'interpreter', 'latex');
plot(t, lambda(t));
line([0, T], [M, M], 'LineWidth', 4);
scatter(p, py, 'o');
scatter(r, ry, '+');
scatter(p, zeros(size(p)), 80, [0 .5 0], 's', 'filled');
for i=1:numel(p)
    line([p(i), p(i)], [0, py(i)], 'LineStyle', '--', 'Color', ↙
        ↳ [0 .5 0]);
end
```

```
axis([0, T, 0, M+.01]);
legend({'$\lambda(t)$','$M$','Accepted Points','Rejected Points'}, 'interpreter', 'latex');

set(gcf,'OuterPosition',[700,500,700,500])
export_fig('..\images\pp.pdf', '-pdf', '-transparent');
```

C.2.2 Hawkes process by thinning

To generate Fig. 6.1b:

```
% HP_CLUSTER Generate a Hawkes process using the thinning algorithm.
clear all; rng(8);

% The time to simulate until.
T = 4;

% Hawkes process parameters.
lambda = 1; alpha = 1; beta = 1.2;

% Current maximum of the process.
M = lambda;

p = []; py = [];
r = []; ry = [];

t = 0;
figure(1); clf; hold on;

MXs = [];
MYs = [];

while t < T
    lastM = M; lastT = t;
    M = cif(t+1e-10, p, lambda, alpha, beta);

    t = t + exprnd(1/M);
    MXs = [MXs, [lastT; t]];
    MYs = [MYs, [M; M]];

    if t < T
        u = rand();
        if u <= cif(t, p, lambda, alpha, beta)/M
```

```
            p = [p, t];
            py = [py, M*u];
        else
            r = [r, t];
            ry = [ry, M*u];
        end
    end
end


t = 0:0.01:T; lambdas = cif(t, p, lambda, alpha, beta);
xlabel('$t$', 'interpreter', 'latex');
ylabel('$U$', 'interpreter', 'latex');


h = zeros(4, 1);
h(1) = plot(t, lambdas);
h(3) = scatter(p, py, [], [0 .5 0], 'o');
h(4) = scatter(r, ry, 'r+');

many = line(MXs, MYs, 'Color', 'b', 'LineWidth', 3);
h(2) = many(1);

scatter(p, zeros(size(p)), 80, [0 .5 0], 's', 'filled');
for i=1:numel(p)
    line([p(i), p(i)], [0, py(i)], 'LineStyle', '--', 'Color', ↙
        ↳ [0 .5 0]);
end
axis([0, T, 0, max(lambdas)*1.05]);

legend(h, {'$\lambda^*(t)$','$M$', 'Accepted Points','Rejected ↙
    ↳ Points'}, 'interpreter', 'latex');

set(gcf,'OuterPosition',[0,0,700,500])
export_fig('..\images\hp_thinning.pdf', '-pdf', '-transparent');
```

C.2.3 Hawkes process by clustering

To generate Fig. 6.2:

```
% HP_CLUSTER Generate a Hawkes process using the clustering ↙
    ↳ approach.
% I.e. generate cluster centres (i.e. immigrants), then create ↙
    ↳ the rest
```

```
% of the clusters (i.e. the offspring).
clear all; rng(4);

% The time to simulate until.
T = 10;

% Hawkes process parameters.
lambda = 1; alpha = 2; beta = 1.2;

% Generate the number and location of cluster centres, and the number of
% descendents in each cluster.
k = poissrnd(lambda*T);
C = sort(T * rand(k, 1));
D = poissrnd(alpha/beta, k, 1);

% For each cluster centre generate an inhomogenous PP.
allOff = [];
figure(1); clf; hold on;
colorOrder = get(gca, 'ColorOrder');

% Generate each cluster.
for c=1:k
    color = colorOrder(mod(c, size(colorOrder, 1))+1,:);
    numOffspring = poissrnd(alpha/beta);

    off = C(c) + exprnd(1/beta, numOffspring, 1);
    scatter(C(c), c, [], color, 'filled', 's');
    s = scatter(off, c.*ones(size(off)), [], color);
    allOff = [allOff; off];
end

points = sort([C; allOff]);

scatter(points, zeros(size(points)), 100, [0,0,0], 'x');
xlabel('$t$', 'interpreter', 'latex');
ylabel('Family Number', 'interpreter', 'latex');
a = axis(); axis([0, T, a(3), a(4)]);

set(gcf,'OuterPosition',[0,0,700,250])
export_fig('..\images\hp_cluster_a.pdf', '-pdf', '-transparent');

%% Plot the conditional intensity function for this realisation.
figure(2); clf; hold on;
t = 0:0.01:T; lambdas = cif(t, points, lambda, alpha, beta);
plot(t, lambdas);
```

```
scatter(points, zeros(size(points)), 100, [0,0,0], 'x');
xlabel('$t$', 'interpreter', 'latex');
ylabel('$\lambda^*(t)$', 'interpreter', 'latex');

set(gcf,'OuterPosition',[0,0,700,250])
export_fig('..\images\hp_cluster_b.pdf', '-pdf', '-transparent');
```

C.2.4 Exact simulation of Hawkes process

```
function T = SimulateHawkes(lambda_0, mean_jump, delta, K_hat)
    % Parameters:
    %   * lambda_0 - Starting intensity
    %   * mean_jump - Amount intensity increases after an arrival
    %   * delta - Decay of intensity over time
    %   * K_hat - Number of arrival times to simulate
    % Reference: Exact simulation of Hawkes process
    %            with exponentially decaying intensity 2013
    %            http://ecp.ejpecp.org/article/view/2717
    W = zeros(1, K_hat);

    a = lambda_0;
    lambda_Tplus = lambda_0;

    for k = 1:K_hat
        u = rand;
        D = 1 + delta*log(u)/(lambda_Tplus - a);

        if D > 0
          S = min([-1/delta * log(D), -(1/a) * log(rand)]);
        else
           S = -(1/a) * log(rand);
        end

        W(k) = S;

        lambda_Tminus = (lambda_Tplus-a) * exp(-delta * S) + a;
        lambda_Tplus = lambda_Tminus + mean_jump;
    end

    T = cumsum(W);
end
```