Journal of Statistical Software

July 2010, Volume 35, Issue 8. *http://www.jstatsoft.org/*

PtProcess: An R Package for Modelling Marked Point Processes Indexed by Time

David Harte
Statistics Research Associates

Abstract

This paper describes the package **PtProcess** which uses the R statistical language. The package provides a unified approach to fitting and simulating a wide variety of temporal point process or temporal marked point process models. The models are specified by an intensity function which is conditional on the history of the process. The user needs to provide routines for calculating the conditional intensity function. Then the package enables one to carry out maximum likelihood fitting, goodness of fit testing, simulation and comparison of models. The package includes the routines for the conditional intensity functions for a variety of standard point process models. The package is intended to simplify the fitting of point process models indexed by time in much the same way as generalized linear model programs have simplified the fitting of various linear models. The primary examples used in this paper are earthquake sequences but the package is intended to have a much wider applicability.

Keywords: **PtProcess** package, marked point process, ground intensity function, non-homogeneous Poisson process, R.

1. Introduction

This paper describes a unified approach to fitting and simulating a wide variety of temporal point process or temporal marked point process models using the R statistical language (R Development Core Team 2010). These models are defined in terms of a conditional intensity function, i.e., conditional on the *history* of the process over *time*.

Point processes are used for modelling a series of events occurring at points in time. A typical and important example is the series formed by the times that earthquakes occur in a given region. Point processes often show a lot of structure. For example, a large earthquake is often followed by a sequence of aftershocks. Some of the aftershocks may, themselves, have aftershocks. These characteristics can be observed in the upper panel of Figure 1. It shows the Phuket boxing day event on 2004-12-26, and its aftershock sequence, that caused the devastating boxing day tsunami in the Indian Ocean. The aftershock sequence of events is also very characteristic of those associated with large earthquakes. Initially, the aftershocks occur very frequently, and some have large magnitudes. Both the frequency and magnitude size diminish over time. One can also observe that the larger aftershocks also have their own associated aftershock sequence. An accessible review of these models is provided by Vere-Jones (2009), who also discusses these models in the context of forest-fires.

An important concept in temporal point process theory is the conditional intensity function (conditional on the *history* of the process over *time*). This describes the probability of a new event in the next instant of time given knowledge of the past up until the present. The conditional intensity function of a marked point process can be considered as having two parts: the ground intensity function together with the distribution of the marks, both conditional on the past, where the ground intensity function simply describes the non-homogeneous Poisson rate of the process over time. Examples of possible marks in the earthquake context are: the earthquake location, magnitude, and rupture direction. Rigorous definitions are given in Section 2 of this paper.

In the development of this software, the ground intensity function provides the unifying principle. Given the ability to calculate values of the ground intensity function, some integrals of the ground intensity function, and the conditional distribution of the marks, one can write *generic routines* to calculate likelihoods, fit parameters, carry out goodness of fit tests and do simulations. The R statistical language has the required structure for doing this and the package **PtProcess** (Harte 2010), described here, is an implementation of this approach within the R framework.

The concept of using a general framework for fitting a wide variety of models is well known, particularly for analysing both designed experiments (Nelder 1977) and generalized linear models (Nelder and Wedderburn 1972). This unified approach resulted in an enormous simplification in the analyses of linear models carried out by applied statisticians. It is the author's aim that the package **PtProcess** (Harte 2010) provides an appropriate unified approach for people analysing a range of temporal point process models.

The **PtProcess** package was originally written (Harte 1998) in S-PLUS to model processes indexed by time and conditional on their history. It has recently been revised to include marked point processes, and has been added to the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org/package=PtProcess`. The R open source software (R Development Core Team 2010) provides a sophisticated programming environment for the development of such models and is also available for people world-wide to freely use. While functions for some specific models have been provided in the **PtProcess** package, the purpose is mainly to provide a structure and environment for users to define and analyse their own marked point process models.

The development of the **PtProcess** package has followed a direction of including the models that we are currently applying in our earthquake modelling. However, we have endeavored to structure the package with sufficient generality to allow for the inclusion of a wider class of temporal point process model: both those of non marked point process type, and those with a non seismological application.

A package available in CRAN for modelling spatial point processes is **spatstat** (Baddeley and

Turner 2005, 2010). The models provided within **spatstat** have no natural definition of past history and are based on a Papangelou conditional intensity (i.e., conditional on the spatial locations of other points, with no time ordering), and hence are quite different to those models discussed in this paper. The **ptproc** package by Peng (2003) is based on an earlier version of **PtProcess**, extending it into multiple dimensions, but deals more directly with the conditional intensity function.

In Section 2 we provide a very brief theoretical background to marked point processes, with an emphasis on those parts that influence the structure of the package. In Section 3 we describe the main structural features of the package and how the underpinning theory has influenced that structure. In Section 4 we provide some example code for various analyses.

2. Mathematical background

In this section a very brief summary of marked point processes is provided. The two most important components of such a model are the ground intensity function and the mark distribution. The ground intensity function describes the rate at which events occur over time, and is not only influenced by the current time, but also the events that have occurred before the current time (i.e., process *history*). This is described in Section 2.1. The mark distribution describes other variables associated with the event, referred to as *marks*, and can also be dependent on the history. The mark distribution is described in Section 2.3. Daley and Vere-Jones (2003, Chapter 7.3) give an extensive theoretical background to this class of models and also many other point process models.

2.1. Ground intensity function

The conditional intensity function describes the instantaneous Poisson rate. For example, consider events that are ordered by time occurring in 2D space. The history up to but not including time t is denoted by $\mathcal{H}_t$, and in this case is

$$\mathcal{H}_t = \{(t_i, x_i, y_i) \quad \forall i : t_i < t\},$$

where t_i is the time of the ith event and (x_i, y_i) is its spatial location. The *conditional intensity function* is

$$\lambda(t, x, y \,|\, \mathcal{H}_t) = \lim_{\delta,\xi,\eta\to 0} \frac{1}{\delta\xi\eta} \Pr\{N_{\delta\xi\eta}(t, x, y) > 0 \,|\, \mathcal{H}_t\}, \tag{1}$$

where $N_{\delta\xi\eta}(t, x, y)$ is the number of events occurring in $[t, t + \delta) \times [x, x + \xi) \times [y, y + \eta)$.

The ground intensity function describes the instantaneous Poisson rate as a function of time only. Let $N_\delta(t)$ be the number of events in $[t, t + \delta)$. Then the *ground intensity function* is defined as

$$\lambda_g(t, \theta_1, \cdots, \theta_m \,|\, \mathcal{H}_t) = \lim_{\delta\to 0} \frac{1}{\delta} \Pr\{N_\delta(t) > 0 \,|\, \mathcal{H}_t\}, \tag{2}$$

where $(\theta_1, \cdots, \theta_m) \in \Theta_m$ are the parameters. Often these are excluded, and so $\lambda_g(t|\mathcal{H}_t)$ is interpreted as $\lambda_g(t, \theta_1, \cdots, \theta_m \,|\, \mathcal{H}_t)$.

An example of a ground intensity function is that of the simple (non-spatial) ETAS model, i.e., *epidemic type aftershock sequence* model used in modelling earthquake counts, see Ogata (1988, 1998, 1999). It assumes that certain earthquake aftershock sequences can be modelled like an epidemic, i.e., large earthquakes inducing more aftershocks (higher infection rate) in a given interval of time and also, the aftershock sequence extends for a longer time after the mainshock event. The ground intensity function is

$$\lambda_g(t|\mathcal{H}_t) = \mu + A \sum_{i:t_i<t} e^{\alpha(M_i - M_0)} \left(1 + \frac{t-t_i}{c}\right)^{-p}. \tag{3}$$

The parameters (μ, A, α, c, p) are all positive, t_i is the time of the ith event with magnitude M_i, and M_0 is a *threshold magnitude*, generally a value below which the dataset is incomplete. The history of the process up to time t includes all event times and magnitudes up to but not including t, i.e.,

$$\mathcal{H}_t = \{(t_i, M_i) \quad \forall i : t_i < t\}.$$

The ETAS model says that events are generated as a Poisson process with rate μ. The first term under the summation (i.e., $\exp\{\alpha(M_i - M_0)\}$) says that larger events raise the intensity more, and the last term determines the length (time) of the aftershock sequence. Obviously, the parameters require certain constraints, else the aftershock sequence (epidemic) could explode and never die out. See Ogata (1988, 1998, 1999) for more details of this model.

Figure 1 contains a plot of the ETAS ground intensity function that has been fitted to the aftershock sequence of the Phuket boxing day earthquake. The mainshock occurred on 2004-12-25 at 00:58:53.45 GMT. The aftershock sequence extends for a considerable time after the mainshock and also includes additional mainshock-aftershock sequences. The data can be found in Harte (2010) and are discussed further in Section 4.

2.2. Likelihood function of simple model

Here we derive the likelihood function for the simple model where the conditional intensity function is only a function of time, i.e., takes the form of the ground intensity function in Equation 2.

Let τ be the time of the last event before time t. Also let $\emptyset_{(\tau,t)}$ be the null outcome, i.e., no events in the interval (τ, t). Denote the conditional distribution of the time of the next event as

$$H(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}) = \Pr\{T \ge t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}\},$$

and $h(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)})$ as the corresponding conditional density function. Then

$$\lambda_g(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}) = \frac{h(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)})}{1 - H(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)})}.$$

Solving the differential equation gives

$$H(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}) = 1 - \exp\left\{-\int_\tau^t \lambda_g(u|\mathcal{H}_\tau \cap \emptyset_{(\tau,u)})du\right\},$$

where τ is the time of the last event occurring before t. Rearranging gives the density function as

$$h(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}) = \lambda_g(t|\mathcal{H}_\tau \cap \emptyset_{(\tau,t)}) \exp\left\{-\int_\tau^t \lambda_g(u|\mathcal{H}_\tau \cap \emptyset_{(\tau,u)})du\right\}.$$

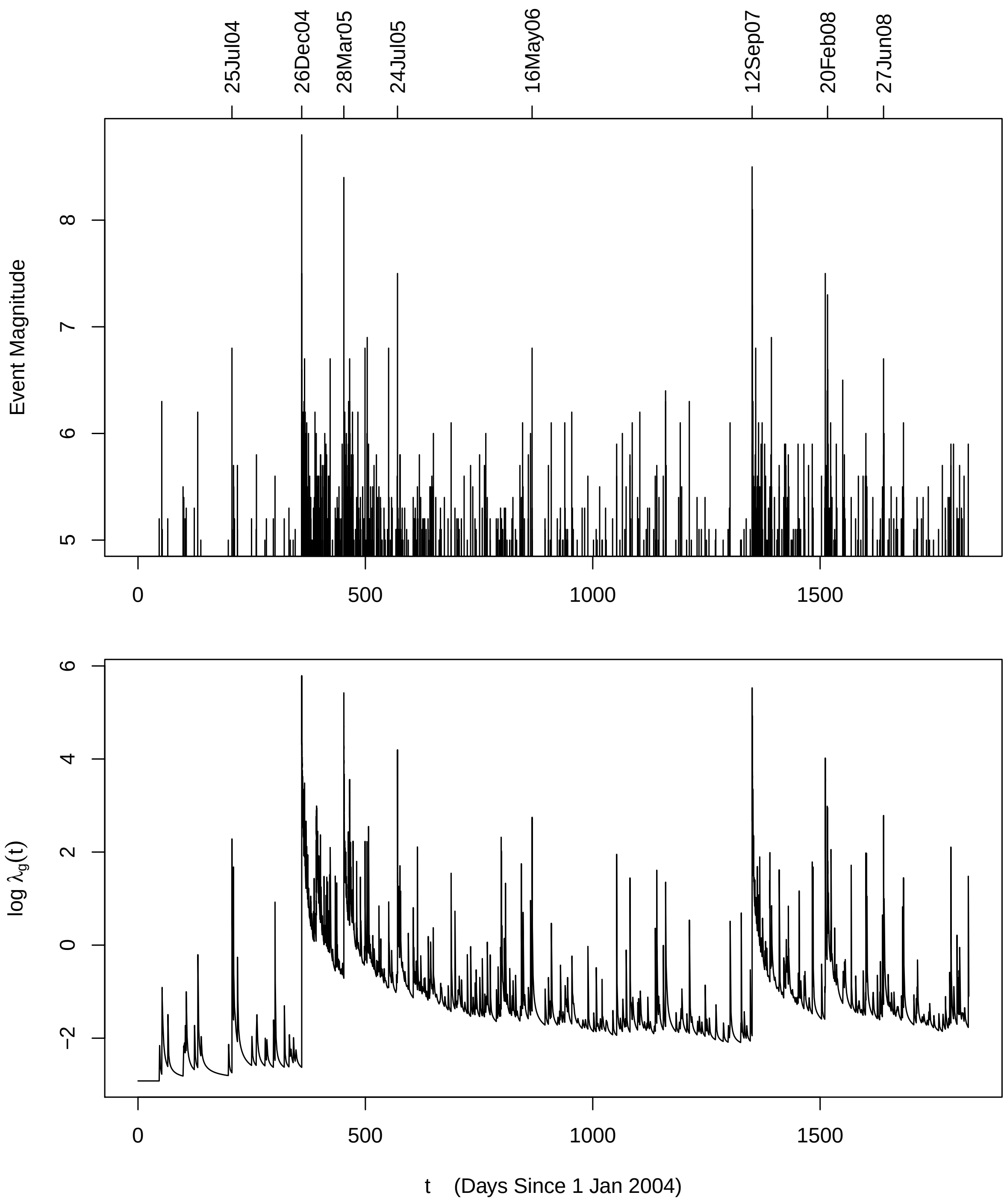

Figure 1: The upper plot shows the Phuket earthquake (2004-12-26 00:58:53.45 GMT), the Sumatra (Nias) earthquake (2005-03-28 16:09:36 GMT), and the 2007 Sumatra earthquakes; and their aftershock sequences. The lower plot shows the logarithm of the ground intensity function of the ETAS model (Equation 3) with parameter values $(\mu, A, \alpha, c, p) = (0.054, 3.15, 1.34, 0.021, 1.12)$. The spatial boundaries of the analysed region are 89°E, 105°E, 5°S, and 16°N. Events with magnitude ≥ 5 between midnight on 2004-01-01 and 2009-01-01 have been selected, giving 1248 events. An epicentral plot of the same events can be seen in Figure 6.

Say $\cdots < t_{-2} < t_{-1} < t_0 < T_1 < t_1 < t_2 < \cdots < t_n < T_2 < t_{n+1} < t_{n+2} < \cdots$, where t_i $i \in \mathbb{Z}$ are event times, and $[T_1, T_2]$ represents the interval within which events are explicitly included into the likelihood. Those events prior to T_1, if available, are included in the history of the process. In the model fitting context this distinction allows one to use the events prior to T_1 to enable the process to "reach equilibrium" or a steady state. Then

$$\begin{aligned}
\log L & \\
&= \log h(t_1 | \mathcal{H}_{T_1} \cap \emptyset_{(T_1,t_1)}) + \sum_{i=2}^{n} \log h(t_i | \mathcal{H}_{t_{i-1}} \cap \emptyset_{(t_{i-1},t_i)}) + \log\left(1 - H(T_2 | \mathcal{H}_{t_n} \cap \emptyset_{(t_n,T_2)})\right) \\
&= \sum_{i=1}^{n} \log \lambda_g(t_i | \mathcal{H}_{t_i}) - \int_{T_1}^{t_1} \lambda_g(u | \mathcal{H}_{T_1} \cap \emptyset_{(T_1,u)}) du - \sum_{i=2}^{n} \int_{t_{i-1}}^{t_i} \lambda_g(u | \mathcal{H}_{t_{i-1}} \cap \emptyset_{(t_{i-1},u)}) du \\
&\qquad - \int_{t_n}^{T_2} \lambda_g(u | \mathcal{H}_{t_n} \cap \emptyset_{(t_n,u)}) du \\
&= \sum_{i\,:\,T_1 \le t_i \le T_2} \log \lambda_g(t_i | \mathcal{H}_{t_i}) - \int_{T_1}^{T_2} \lambda_g(t | \mathcal{H}_t) dt \,. \qquad (4)
\end{aligned}$$

2.3. Marked point process

Consider again the situation where the conditional intensity function is given by Equation 1. By a similar argument as in Section 2.2, its log-likelihood is

$$\log L = \sum_{i\,:\,t_i \in \mathcal{T}} \log \lambda(t_i, x_i, y_i \,|\, \mathcal{H}_{t_i}) - \int_{\mathcal{T}} \int_{\mathcal{Y}} \int_{\mathcal{X}} \lambda(t, x, y \,|\, \mathcal{H}_t)\, dx\, dy\, dt \,,$$

where $\mathcal{T} \subseteq \mathbb{R}^+$ is a time interval, and $\mathcal{X}$ and $\mathcal{Y}$ are the domains of x and y, respectively. Obviously this can be extended to include many additional variables. For brevity, we rewrite it as

$$\log L = \sum_{i\,:\,t_i \in \mathcal{T}} \log \lambda(t_i, y_i \,|\, \mathcal{H}_{t_i}) - \int_{\mathcal{T}} \int_{\mathcal{Y}} \lambda(t, y \,|\, \mathcal{H}_t)\, dy\, dt \,, \qquad (5)$$

where y now represents a multivariate variable, referred to below as the *marks*.

A *marked point process* is a temporal point process $\{(t_i, y_i)\}$ on $\mathbb{R}^+ \times \mathcal{Y}$ with the additional property that the conditional intensity function can be written as the following product

$$\lambda(t, y \,|\, \mathcal{H}_t) = \lambda_g(t | \mathcal{H}_t) f(y | \mathcal{H}_t) \,, \qquad (6)$$

where $\lambda_g(t|\mathcal{H}_t)$ is a ground intensity function as in Equation 2, and $f(y|\mathcal{H}_t)$ is a multivariate density function of the marks (Daley and Vere-Jones 2003, Definition 6.4.1). The mark density could be denoted as $f(y, \psi_1, \cdots, \psi_p \,|\, \mathcal{H}_t)$, where $(\psi_1, \cdots, \psi_p) \in \Psi_p$ are the model parameters associated with the mark distribution. Generally the function parameters are omitted, and $f(y|\mathcal{H}_t)$ is interpreted as $f(y, \psi_1, \cdots, \psi_p \,|\, \mathcal{H}_t)$.

The general form of the log-likelihood of a marked point process then follows directly from Equation 5 as

$$\log L = \sum_{i\,:\,t_i \in \mathcal{T}} \log \lambda_g(t_i | \mathcal{H}_{t_i}) - \int_{\mathcal{T}} \lambda_g(t | \mathcal{H}_t)\, dt + \sum_{i\,:\,t_i \in \mathcal{T}} \log f(y_i | \mathcal{H}_{t_i}) \,. \qquad (7)$$

Note that the ground intensity and mark distribution could share some common parameters, hence Θ_m (Equation 2) and Ψ_p are not necessarily disjoint (i.e., $\Theta_m \cup \Psi_p \subseteq \mathbb{R}^q$ where $q \leq m+p$). While a trivial concept here, it is an important consideration in the software structure in Section 3.

Consider again the ETAS model applied to the Phuket earthquake sequence. Often the ETAS model is fitted under the assumption that the event magnitudes (i.e., $M_i - M_0$) have an exponential distribution that is independent of the history of the process. An alternative assumption could be that during the initial part of the aftershock sequence when $\lambda_g(t|\mathcal{H}_t)$ is relatively higher, we might expect the magnitude distribution to have fewer small events, hence a greater mean, and this mean would return to a lower level as the ground intensity returned to the relatively low background level. In this situation, the mark distribution (i.e., of the magnitude) is conditional on the history of the process, which will determine the current value of $\lambda_g(t|\mathcal{H}_t)$.

2.4. Residual process

Let t_i, for $i = 1, \cdots, n$, be the times of the observed events. Then consider the transformed times

$$\tau_i = \int_0^{t_i} \widehat{\lambda}_g(t|\mathcal{H}_t)dt\,, \tag{8}$$

where $\widehat{\lambda}_g(t|\mathcal{H}_t)$ is the fitted ground intensity function. If the data are sampled from a process with ground intensity $\widehat{\lambda}_g(t|\mathcal{H}_t)$, then the transformed time points will form a stationary Poisson process with rate parameter one. This sequence of transformed times is referred to as the *residual process*. Further details can be found in Ogata (1988) and Aalen and Hoem (1978).

A simple diagnostic graphical test of the model goodness of fit is to plot the event number i (i.e., $i = 1, \cdots, n$, horizontal axis) versus the transformed time τ_i (vertical axis). Note that the horizontal scale is *not* linear in time. The points should roughly follow the straight line $y = x$. Significant departures from this line indicate a weakness in the model. Further, if in a given sub-interval the line has a slope less than one then the transformed times τ_i are too small, which indicates that the fitted ground intensity function $\widehat{\lambda}_g(t|\mathcal{H}_t)$ is too small in this sub-interval. Conversely, if the slope is greater than one, the fitted ground intensity function $\widehat{\lambda}_g(t|\mathcal{H}_t)$ is too large in this sub-interval.

An alternative is to plot $\tau_i - i$ on the vertical scale. Such a plot is referred to as a *cusum* in the quality control literature and was devised by Page (1954), see also Lucas (1985). In the present context, the "cumulative sum" is that of the inter-event residual times, and the subtraction of i removes the mean. Hence, when the process is "under control", the cusum will have a zero slope. A positive or negative slope indicates that $\widehat{\lambda}_g(t|\mathcal{H}_t)$ is either too large or too small, respectively. A disadvantage of the first graphical representation is that the vertical scale of the plot increases linearly with the length of the series, and hence deviations from the line $y = x$ will become visually smaller and hence not as noticeable with increasing n. The cusum technique effectively reduces the vertical scale. In the same way as Brownian motion will deviate further from the origin with increasing time, so will the cusum, but it will be at a much slower rate than in the first plot, i.e., $O(\sqrt{n})$ compared to $O(n)$.

Zhuang (2006) and Schoenberg (2003) discuss the residual process in more general settings.

2.5. Model simulation

The **PtProcess** package uses the thinning method (Lewis and Shedler 1979; Ogata 1981) to simulate event times. Essentially one takes a small sub-interval at the beginning of the required simulation period. Then one finds the maximum of $\lambda_g(t|\mathcal{H}_t)$ on the sub-interval, and simulates an inter-event time according to this maximum rate $\lambda_{\max}$. Given this has been done using $\lambda_{\max}$, the inter-event time will, on average, be too small. Hence, events will be generated too frequently; thus we generate a potential event (too many on average), and then "thin". One then moves along the simulation period and considers the next sub-interval.

Thinning algorithm

1. Let τ be the start of a small simulation interval.

2. Take a small interval $(\tau, \tau + \delta)$.

3. Calculate the maximum of $\lambda_g(t|\mathcal{H}_\tau)$ in the interval as
$$\lambda_{\max} = \max_{t \in (\tau, \tau+\delta)} \lambda_g(t|\mathcal{H}_\tau)\,.$$

4. Simulate an exponential random number ξ with rate $\lambda_{\max}$.

5. If
$$\frac{\lambda_g(\tau + \xi)}{\lambda_{\max}} < 1$$
then go to step 6.

   Else no events occur in $(\tau, \tau + \delta)$, hence $\tau \leftarrow \tau + \delta$, and return to step 2.

6. Simulate a uniform random number U on the interval $(0, 1)$.

7. If
$$U \le \frac{\lambda_g(\tau + \xi)}{\lambda_{\max}}\,,$$
then a new "event" occurs at $t_i = \tau + \xi$. Simulate the associated marks for this new event.

8. Increment τ for the next event simulation:
$$\tau \leftarrow \tau + \xi\,.$$

9. Return to step 2.

If $\lambda_g(t|\mathcal{H}_t)$ is monotonically decreasing (except at event times), as in the ETAS model, then the selection of δ has no effect because $\lambda_{\max} = \lambda_g(\tau|\mathcal{H}_\tau)$. When $\lambda_g(t|\mathcal{H}_t)$ is monotonically increasing (except at event times), there are two extreme situations that could cause the simulation method to be inefficient. If δ is too small, $\lambda_{\max}$ will be relatively small, hence ξ quite large, possibly greater than $\tau + \delta$. Here many small intervals will be considered, but each with a very low likelihood of including an event. If δ is too large, $\lambda_{\max}$ will be relatively large, hence ξ will be quite small. This could be inefficient as many potential "events" will be "thinned". Hence, for best efficiency, one requires δ to be not too small, but also not too large. Our algorithm sets δ as the 70th percentile of an exponential distribution with rate equal to $\lambda_g(\tau|\mathcal{H}_\tau)$.

The thinning algorithm is not necessarily the most efficient or intuitive method of simulation. However, it does provide a method that is sufficiently general for a reasonably wide class of models. An alternative method for simulating the ETAS model, which also gives more intuition into the underlying model, is to initially simulate the ancestor (mainshock). Then one would simulate the children of this ancestor (1st generation), then their children (2nd generation), and continue until the given family line dies out. At this point the aftershock sequence (family tree) associated with the given mainshock event (ancestor) has expired and the process returns to the underlying rate of seismicity. In fact, the ETAS model assumes that all events are potential ancestors, though many are of sufficiently small magnitude that they have no offspring.

3. Package structure

3.1. Overview of structure

The essence of the package structure hinges around the concept of a model object. In the present paper, we are particularly interested in marked point process models. Hence, we want to define the structure for an R object which embodies all of the information that is required to define such a model. We want to do this in such a way that we can perform generic like operations (functions) on this object, for example, a model summary, calculate the residuals, calculate the log-likelihood, possibly an appropriate plot, simulate data, and so on. The R programming language has a number of generic functions (e.g., `plot`, `summary`, `residuals`, `logLik`, `simulate`, etc). We want to provide *methods* for some of the R generic functions for when such a generic function is applied to a *marked point process model object*.

The model object needs to contain the dataset and an appropriate specification of the mathematical structure of the model. We will need to calculate the log-likelihood, particularly if we want to maximize the likelihood with respect to the model parameters. Generally, optimisation functions require all estimable parameters to be represented as a single vector. We also want the users to be able to define their own conditional intensity functions and build an appropriate model object in such a way that the provided *methods* for marked point process objects will work on their user defined objects.

In the preceding section (Section 2), it was shown that the marked point process contains two important "building blocks", the ground intensity function and the mark density function. In fact, it can be seen that if the R representations of the ground intensity function and mark density function can both be evaluated at arbitrary values of time, and the integral of the ground intensity function can be calculated over arbitrary intervals, then we can calculate the log-likelihood, simulate a process, calculate the residual process, and so on. In Section 3.2 and Section 3.3 we define a general structure for the R representations of the ground intensity function and the mark density function, respectively.

As already noted, we require the model parameters to be represented to external functions as a single vector. This enables the log-likelihood as given by Equation 7 to be maximized over the complete model parameter space. The model parameter space is $\Theta_m \cup \Psi_p$ where Θ_m is the parameter space of $\lambda_g(t|\mathcal{H}_t)$, and Ψ_p is the parameter space of $f(y|\mathcal{H}_t)$. Further, $\Theta_m \cap \Psi_p$ is not necessarily $\emptyset$. Therefore, we need two mappings that map the complete parameter space into Θ_m, and another mapping it into Ψ_p. The structure for these mappings is described in Section 3.4.

In Section 3.5 we define the structure of the R *marked point process model object*, and in Section 3.6 we briefly describe generic functions in R that currently have *methods* defined for the marked point process model object. The functions that provide these *methods* are relatively simple because they can be written in a very general manner; everything that is specific to the model of interest is contained within the model object, namely the ground intensity function and the mark density function. This is why these two functions must conform to a general specification (Sections 3.2 and 3.3, respectively); they will be executed directly by the methods functions.

The last subsection (Section 3.7) describes how parameters can be estimated by maximising the likelihood function.

3.2. Ground intensity function

Ground intensity functions, defined in Section 2.1, are programmed as R functions with the following structure. In example code below, we generically refer to individual functions as `gif`. Those provided in the **PtProcess** package have a naming convention `name_gif`. Currently they include jump processes that are conditional on their history (e.g., `etas_gif`, `srm_gif`, `linksrm_gif`), and non-homogeneous Poisson processes that are not conditional on their history (e.g., `simple_gif`, `expfourier_gif`, `exppoly_gif`, `fourier_gif`, `poly_gif`).

The R `gif` functions not only evaluate values of $\lambda_g(t|\mathcal{H}_t)$, but also its integral, which is required to calculate the log-likelihood (Equation 7) and the residual process (Equation 8). In this sense, the R `gif` function does more than that specified by the mathematical definition given in Section 2.1.

Forms of usage

An R `gif` function has two usages, one to evaluate $\lambda_g(t|\mathcal{H}_t)$ at a specified vector of time points, and another to evaluate the integral of $\lambda_g(t|\mathcal{H}_t)$ on a specified interval. The usage is determined by the mix of supplied arguments. The two usages are, respectively,

```
gif(data, evalpts, params, tplus = FALSE)
```

and

```
gif(data, NULL, params, TT)
```

where the arguments are defined as follows.

- `data` is a data frame containing the history of the process ($\mathcal{H}_t$). It should contain all variables that are required to evaluate the `gif` function, though can contain others too. No history is represented as `NULL`.
- `evalpts` is a vector of times at which the `gif` function is to be evaluated.

`params` is vector containing values of the parameters required by the `gif` function.

`TT` is vector of length 2, being the time interval over which the integral of the ground intensity function is to be evaluated.

`tplus` is logical, $\lambda_g(t|\mathcal{H}_t)$ is evaluated as $\lambda_g(t^+|\mathcal{H}_t)$ if `TRUE`, else $\lambda_g(t^-|\mathcal{H}_t)$. It is important if a "jump" occurs at t.

All `gif` functions must have the same arguments even if they are redundant for the given point process under study (e.g., a homogeneous Poisson process is not dependent on its history).

Function attributes

Functions have been given an attribute `"rate"`, taking the values of `"bounded"`, `"decreasing"` or `"increasing"`. This is used within the `simulate` method for 'mpp' objects which uses the thinning method (see Section 2.5). This method requires a knowledge of the maximum of $\lambda_g(t|\mathcal{H}_t)$ in a given interval. The argument `tplus` is also used by the simulation routine, where it is necessary to determine the value of the intensity immediately after a simulated event.

Example function

Consider the following simple "conditional" intensity function

$$\lambda_g(t|\mathcal{H}_t) = \alpha + \beta t\,,$$

where α, β and t are all assumed to be positive. In fact the function is not dependent on the history of the process at all, but the essential structure of the R function below is still the same; the history argument (`data`) is simply not used within the function. It can be seen that the function has two parts, depending on whether the argument `TT` is missing. If an interval $\texttt{TT} = (T_1, T_2)$ has been specified, then the integral

$$\int_{T_1}^{T_2} \lambda_g(t|\mathcal{H}_t)\,dt = \alpha(T_2 - T_1) + \frac{\beta}{2}\left({T_2}^2 - {T_1}^2\right)$$

is returned, otherwise $\lambda_g(t|\mathcal{H}_t)$ is evaluated at each value of t specified within `evalpts`. Checking for positive values could also be included within the function, but this would be at the expense of execution time.

```
R> example_gif <- function(data, evalpts, params, TT = NA, tplus = FALSE) {
+    alpha <- params[1]
+    beta <- params[1]
+    if(any(is.na(TT))) {
+      if (is.vector(evalpts)) eval.times <- evalpts
+        else eval.times <- evalpts[, "time"]
+      ci <- alpha + beta * eval.times
+    } else {
+      ci <- alpha * (TT[2] - TT[1]) + beta/2 * (TT[2]^2 - TT[1]^2)
+    }
+    return(ci)
+  }
R> attr(example_gif, "rate") <- "bounded"
```

The other unused function argument in this example is `tplus`. In many processes that are dependent on the history, a jump occurs in $\lambda_g(t|\mathcal{H}_t)$ at the time of an event, more specifically, the process is assumed to be left continuous. If `tplus == TRUE`, then $\lambda_g(t^+|\mathcal{H}_t)$ is returned, i.e., the value of the function immediately after the jump. This value is required by the thinning simulation algorithm. Note also the attribute that has been attached at the bottom of the function code. This is also required by the thinning simulation algorithm to determine $\lambda_{\max}$ (see Section 2.5).

Code for other functions that are conditional on the process history can be scrutinized by printing out the function. The essential structure of the R function is the same as in the example above.

3.3. Mark distributions

In order to fit a marked point process, we need to define the appropriate R function to calculate the mark densities. If we want to simulate a marked point process, then we need to define the appropriate R function to simulate the mark variates. Consider a mark distribution, which could be multivariate, called "*xyz*". We will refer to the corresponding R mark density and mark simulation functions as `d`*xyz*`_mark` and `r`*xyz*`_mark`, respectively. The prefixes "`d`" and "`r`" are chosen to be consistent with the nomenclature used in the standard R probability functions. R functions representing mark density and random number generators have the follow generic argument structure:

```
dxyz_mark(x, data, params)
```

and

```
rxyz_mark(ti, data, params)
```

respectively, where:

`ti`	is the time of an event (scalar),
`x`	is a `data.frame` of mark values and times of specific events, generally a subset of the history,
`data`	is a `data.frame` containing the history of the process, denoted below as $\mathcal{H}_t$, and
`params`	is a numeric vector of parameters.

Mark density functions must return a vector with length being equal to the number of rows in `x`. Each element contains the *logarithm of the joint density* of the marks corresponding to each event time (row) in `x`.

The random number generator simulates each mark for a *single value* of `ti`. It must return a `list` of simulated marks corresponding to the specified time `ti`, and so each component in the list will be of length one. A list is used (rather than a numeric vector) because it allows marks to have a more complex structure.

Consider again the example of the Phuket earthquake sequence. Assume that after very large events, there is a deficit of smaller magnitude events and more larger magnitude events (i.e., relative to the exponential distribution)[1]; in particular a gamma distribution with shape parameter that is greater than one. When the shape parameter is equal to one, we have the exponential distribution. We want the distribution to return to roughly exponential after the ground intensity has returned to a low level that is characteristic of the background level of seismicity. Hence, define an R mark density function with seven parameters. Parameters $p_1, \cdots, p_5$ will be required to evaluate the R ground intensity function `etas_gif`. Assume that the magnitude distribution is gamma with scale parameter p_6, and with a shape parameter given by

$$\text{shape} = 1 + \sqrt{\lambda_g(t|\mathcal{H}_t)}\; p_7\,, \tag{9}$$

where p_7 ($p_7 > 0$) is a free estimable parameter. When $\lambda_g(t|\mathcal{H}_t)$ is small, the magnitude distribution roughly returns to the exponential distribution with an approximate rate of p_6. The square root transformation was chosen pragmatically, to have a similar effect as a log transform, but ensuring that the values all remain positive. The density function could be written as follows:

```
R> dmagnitude_mark <- function(x, data, params) {
+    lambda <- etas_gif(data, x[, "time"], params = params[1:5])
+    y <- dgamma(x[, "magnitude"], shape = 1 + sqrt(lambda) * params[7],
+      rate = params[6], log = TRUE)
+    return(y)
+  }
```

An R function to simulate such a mark distribution could be as below. Note that only one value of the mark that corresponds to time `ti` is simulated.

```
R> rmagnitude_mark <- function(ti, data, params) {
+    lambda <- etas_gif(data, ti, params = params[1:5])
+    y <- rgamma(1, shape = 1 + sqrt(lambda) * params[7], rate = params[6])
+    return(list(magnitude = y))
+  }
```

In the above example, we have modified the exponential distribution to restrict the number of very small events just above the threshold magnitude M_0. However, another problem with the exponential distribution is that its tail is too long, and hence if very large earthquake sequences are simulated, earthquake events with unrealistically large magnitudes[2] will occasionally be generated. The gamma distribution above actually compounds this problem. Ideally, one also wants to truncate the tail of the exponential distribution, see Harte (2010, topic `dpareto`).

3.4. Model parameter space

The parameter spaces of the ground intensity function and mark distribution (i.e., Θ_m and Ψ_p) are not necessarily disjoint. In order to provide sufficient flexibility, mappings need to

[1]This deficit of smaller events does occur, though the reason is not completely clear. One explanation is that the region is so active after a large event that signals from small events arriving at seismic stations will not be able to be distinguished from the signals of the larger events. Alternatively, the region could be in a state of super criticality, and any minor disturbance will coalesce into a larger event.

[2]It is not really possible for earthquakes with magnitude greater than about 9.5 to occur as the rupture size relative to the size of the Earth would be too great.

be defined from the full model parameter space to both Θ_m and Ψ_p. These are specified by means of R expressions.

The mapping expressions can contain any legitimate R arithmetic expression. The complete set of model parameters are assumed to be contained in a vector called `params`. Here is an example of a five parameter model, where the `gif` has four parameters (mapping `gmap`), and the mark distribution has three parameters (mapping `mmap`), with mappings specified as follows:

```
R> gmap <- expression(c(params[1:3], exp(params[5])))
R> mmap <- expression(c((params[1] + params[2])/2, params[4:5]))
```

Note that `params[4]` is exclusive to `mmap`, `params[3]` is exclusive to `gmap`, and all other parameters are shared. Note the inclusion of the R combine (`c`) function, because the `expression` must create a vector of parameters. These expressions are embedded directly into the code of various functions.

3.5. Marked point process model object

A marked point process is defined within a `list` object of class '`mpp`'. By giving the object a class, we can use the object orientated nature of the R programming language and provide appropriate *methods* for some generic functions that are already defined within R.

The '`mpp`' object must contain all of the information required to define the marked point process model. It can be constructed using the `mpp` function which has the following usage

```
mpp(data, gif, marks, params, gmap, mmap, TT)
```

where the arguments are as follows.

`data` is a `data.frame` containing the history of the process, denoted below as $\mathcal{H}_t$. It should contain all variables that are required to evaluate the `gif` function and the mark distribution, though can contain others too. No history is represented as `NULL`.

`gif` is the ground intensity function as described in Section 3.2.

`marks` is a `list` containing the mark distribution as described in Section 3.3. The first component of the list contains the mark density function and the second the random number generating function, e.g., `list(d`xyz`_mark, r`xyz`_mark)`. Undefined components are represented as `NULL`, e.g., `list(dxyz_mark, NULL)`.

`params` is a numeric vector of *all* model parameters.

`gmap` is an `expression` that maps the model parameters (`params`) into the parameter sub-space Θ_m of the ground intensity function; see Section 3.4.

`mmap` is an `expression` that maps the model parameters (`params`) into the parameter sub-space Ψ_p of the mark distribution; see Section 3.4.

`TT` is a vector of length 2, being a time interval. Its use depends on whether the log-likelihood is being calculated or events are being simulated; see Section 3.6 for further details.

3.6. Generic functions

The R generic functions `logLik`, `plot`, `residuals`, `simulate` and `summary` currently have methods for the ‘`mpp`’ model object. The algorithms to calculate the log-likelihood, residual process, and simulate data were given in Section 2.3, Section 2.4 and Section 2.5, respectively. The R code can be scrutinized by printing the `logLik`, `residuals`, or `simulate` methods, respectively. Similarly, the methods for `plot` and `summary` can be scrutinized by printing `plot` or `summary` methods, respectively.

Let `x` be an object with class ‘`mpp`’. We now briefly describe these functions.

`logLik(x)` returns the log-likelihood. If `x$marks[[1]]` is `NULL`, then $\log L$ as in Equation 4 is returned, else $\log L$ as in Equation 7 is returned. Events included in the interval `x$TT` explicitly enter the likelihood, however, all events (even before `x$TT[1]`) are included in the history. This allows one to initially wait until `x$TT[1]` for the process to reach a steady state.

`plot(x)` plots the ground intensity function as in the lower panel of Figure 1.

`residuals(x)` returns a vector (`"ts"` object) of event times of the residual process, see Equation 8.

`simulate(x)` returns an object of class ‘`mpp`’. By default, the function simulates events between `x$TT[1]` and `x$TT[2]`. This can be overridden by a more sophisticated stopping condition. The new object will contain all events in the object `x` prior to `x$TT[1]` and then the newly simulated events.

`summary(x)` returns a list object containing a summary of the model.

3.7. Model fitting

Maximising the likelihood function is a non-trivial computational exercise. There are a number of possible problems: some optimisation routines are very sensitive to poor initial starting values of the parameters, hard boundaries on the parameter space, and different parameters may have very different scales or ranges of appropriate values. We will discuss these problems and how they are related at the end of this subsection.

Initially we want to set up a fairly general framework that gives the user flexibility to use different optimizers, utilize provided parameter scaling within the optimizer or implement their own, and also implementation of parameter constraints. This is achieved with a generic-like function called `neglogLik`. It returns the negative log-likelihood of the model. The function code is essentially as follows:

```
R> neglogLik <- function(params, object, pmap) {
+    object <- pmap(object, params)
+    return(-logLik(object))
+  }
```

where:

`params` is a vector of parameter values that could belong to a subspace of the full parameter space, further details below, `object` is a model object, and

`pmap` is a user provided function mapping the revised parameter values `params` into the appropriate locations in `object`. The default is that the `params` vector is mapped one to one (untransformed) onto the model vector.

Initially new parameter values contained in `params` are mapped into the appropriate locations in the model `object`. The log-likelihood function is then calculated, the *method* being determined by the class of the `object`. It is essentially a "wrapper" function for the generic `logLik` function. We now show that it serves a number of useful purposes: as well as outputting the negative value of the log-likelihood, it also allows one to restrict the parameter space to a subspace of the full model, and also allows one to transform the parameters to eliminate boundary problems.

Parameter transformation: Case 1

Consider a model defined in an object `x`. Assume that the full model has five parameters, and we want to estimate the parameters by maximising the likelihood. We can achieve this by using the R function `nlm` which is in fact a minimizer (can also use `optim`), hence the calculation of the negative log-likelihood.

We initially provide starting values for `nlm`. Say the following values are reasonable:

```
R> initial <- c(10, 20, 30, 40, 50)
```

We want `nlm` to start with these `initial` values, and iteratively calculate new versions until convergence is achieved. The `logLik` function call within the function `neglogLik` will use those parameter values that are contained within the current model `object`, so we need a function to update these values with the new estimates at each iteration:

```
R> allmap <- function(y, p) {
+    y$params <- p
+    return(y)
+ }
```

Now the `neglogLik` function can be minimized using `nlm`, where the behaviour of the function `neglogLik` is determined by the class of the object `x`:

```
R> z <- nlm(neglogLik, initial, object = x, pmap = allmap)
R> x <- allmap(x, z$estimate)
R> print(logLik(x))
```

Parameter transformation: Case 2

The iterative process can get into trouble if the domain of some parameters have boundaries and new estimates fall outside of this domain. Consider the above five parameter example again, but where we only want to maximize the likelihood over parameters 1, 3, 4, and 5 while fixing parameter 2 at 20. Further, assume that parameter 1 can only take values between 9 and 11, parameter 3 must be positive, and parameters 4 and 5 can take any real values.

This would be achieved with the following code. The modified logit transformation allows `nlm` to select values for the first parameter from the entire real line (eliminating boundary problems), which then get mapped into the required interval. Similarly, the logarithmic transformation ensures that parameter 3 will always be positive.

```
R> x$params[2] <- 20
R> logitab <- function(y, a, b) log((y - a) / (b - y))
R> invlogitab <- function(eta, a, b) a + (b - a) * exp(eta)/(1 + exp(eta))
R> initial <- c(logitab(10, 9, 11), log(30), 40, 50)
R> mapcase2 <- function(y, p) {
+    y$params[1] <- invlogitab(p[1], 9, 11)
+    y$params[3] <- exp(p[2])
+    y$params[4:5] <- p[3:4]
+    return(y)
+ }
R> z <- nlm(neglogLik, initial, object = x, pmap = mapcase2)
R> x <- mapcase2(x, z$estimate)
R> print(logLik(x))
```

Hessian, standard errors, AIC and parameter scaling

Routines that find the maximum by essentially taking the "steepest ascent" will generally be quite efficient. However, to take this route, numerical derivatives (possibly 2nd too if a quadratic approximation is used) will need to be calculated (we have not included formulae for derivatives in the model object). If the provided initial values are too poor, then this strategy also performs poorly or may not even work. Ideally one would like a robust initial method, particularly if one is uncertain of the parameter values. The R function `optim` provides a number of possibilities. Its default method uses the Nelder-Mead simplex algorithm (Nelder and Mead 1965) which appears to be more robust than methods based on derivatives.

There are a few possible ways to calculate the standard errors of parameters. One way is to calculate the Hessian matrix. Again, depending on the method that the optimizer uses, the Hessian may well have been calculated, and an argument simply needs to be set to request the optimisation function to output this matrix. The covariance matrix of the parameter estimates can then be calculated by inverting this matrix, and the standard errors are the square roots of the diagonal elements of this inverted matrix. One needs to take some care here though too. For example, the estimated Hessian matrix will be dependent on how many iterations the optimizer has performed prior to the final iteration. At each iteration, the optimizer will generally revise the parameter scaling factors (see below), which in turn will affect the calculated numerical derivatives. The estimated matrix will be rather poor if few iterations have been done because the scaling factors will not have had sufficient time to "settle down". If the model variables have been transformed, then the Hessian provided by the optimisation routine will be of the transformed variables. One may want to revert to untransformed variables close to the solution if standard errors really are required. Further, use of such standard errors appeals to asymptotic theory of maximum likelihood parameter estimates (normality) which may provide a poor approximation if the sample dataset is too small.

An alternative to Hessian-based standard errors is to simulate data of the same size as the observed dataset using the estimated parameter values for the observed dataset. One would then estimate the parameters for the simulated dataset. By repeating this simulation and model fitting many times, one can derive empirical distributions for each of the parameters. Given that some of these models can take a considerable amount of computing time to fit, this would only be a viable option in the case of relatively small datasets, where the asymptotic theory is less likely to hold. At least in the case of the larger datasets, the asymptotic theory is more likely to hold.

In some respects, the problems associated with calculating the standard errors are not really such a problem. Often one wants to compare a sequence of different fitted models. However, the difference between this sequence of fitted models to a sequence of fitted regression models in the linear model context is that the point process models are generally not nested. Hence, it is not simply a matter of testing whether a particular parameter is equal to zero or not. In the point process application areas, greater use seems to be made of measures like the AIC to compare competing models.

Scaling of the parameters may also be required if the different parameters are on very different scales. Both of the R functions `nlm` and `optim` have an argument where the parameter scales can be specified. Alternatively, scaling can be jointly achieved in many situations by using the log-based transformations already discussed, in much the same way where one uses a log transformation in classical regression models.

4. Example

In this section we fit two models to the Phuket aftershock sequence (Figure 1). The analyses here are very preliminary, and to provide a satisfactory model description of the process is a very complex task. For example, one complication is the temporal-spatial-magnitude boundaries imposed on the dataset. The data sequence is part of a geophysical process that has and will continue ad infinitum, and the spatial region selected is part of a larger tectonic environment. Further, all earthquake catalogues are incomplete for small magnitude events, and the catalogue completeness magnitude is not necessarily constant in time and space[3]. While the models fitted below describe certain features of the given dataset, they probably have a very poor forecasting ability of future events.

The dataset is contained in the **PtProcess** package (Harte 2010). The data were originally extracted from the PDE (preliminary determination of epicentres) catalogue provided by the US Geological Survey (`ftp://hazards.cr.usgs.gov/pde/`). The catalogue is incomplete for events with magnitude below about 4.5. The magnitudes are rounded to one decimal place. To further restrict the size of the dataset, we have only included events with magnitude ≥ 5, so that the effective magnitude cut-off is 4.95. The selected spatial region has boundaries 89°E, 105°E, 5°S and 16°N. It is arbitrary what unit of time is used in the ground intensity function, though some of the parameter values will be peculiar to the chosen unit. For studies over hundreds of years, one would probably use years. In this example the time variable is the number of days since the beginning of the dataset, i.e., midnight on 2004-01-01 is time

[3]One cause of temporal inhomogeneity is the implementation of temporary seismographs in a region after a major event. Geophysicists particularly want to determine information about the stress-fracturing process, but the resulting inhomogeneity in the catalogued events causes modelling problems. This can be minimized by taking a reasonably high threshold magnitude, here selected as $M_0 = 4.95$.

zero. All 1248 recorded events up until midnight on 2009-01-01 have been included, i.e., 1827 days later.

In Section 4.1 an ETAS ground intensity function with an exponential magnitude distribution (independent of the history) is fitted to the data. In Section 4.2 a model with a gamma density for the event magnitudes (Equation 9) is fitted to the data. This mark distribution is dependent on the history of the process. The residual process for each model is calculated and plotted in Section 4.3, and a forecast based on simulation is performed in Section 4.4.

4.1. Null model

In this subsection we fit a model to the Phuket earthquake data (Figure 1) with an ETAS ground intensity function (Equation 3) and an exponential distribution that is independent of the process history for the event magnitudes.

We initially set up the model object in a way that it can be used more generally in the following subsections. This is done by using the gamma density for the event magnitudes (mark distribution) as in Equation 9. In this subsection, we then impose restrictions on the parameters to ensure that the exponential distribution is actually fitted.

```
R> library("PtProcess")
R> data("Phuket")
R> Phuket$magnitude <- Phuket$magnitude - 4.95
R> dmagn_mark <- function(x, data, params) {
+    if (params[7] > 0) {
+      lambda <- etas_gif(data, x[, "time"], params = params[1:5])
+      y <- dgamma(x[, "magnitude"], shape = 1 + sqrt(lambda) * params[7],
+        rate = params[6], log = TRUE)
+    } else y <- dexp(x[, "magnitude"], rate = params[6], log = TRUE)
+    return(y)
+  }
R> rmagn_mark <- function(ti, data, params) {
+    if (params[7]>0) {
+      lambda <- etas_gif(data, ti, params = params[1:5])
+      y <- rgamma(1, shape = 1 + sqrt(lambda) * params[7], rate = params[6])
+      } else y <- rexp(1, rate = params[6])
+      return(list(magnitude = y))
+  }
R> TT <- c(0, 1827)
R> params <- c(0.05, 3.1, 1.3, 0.02, 1.1, 1/mean(Phuket$magnitude), 0)
R> x <- mpp(data = Phuket, gif = etas_gif,
+    mark = list(dmagn_mark, rmagn_mark), params = params, TT = TT,
+    gmap = expression(params[1:5]), mmap = expression(params))
```

The gamma distribution reduces to the exponential distribution when the shape parameter is one (i.e., `params[7]=0`, the default value within the model object). In this situation the magnitudes are independent of the history and hence the rate is easily estimated as the inverse of the mean. This has also been set as the default value within the model object. Since the rate parameter of the exponential distribution has already been estimated and is contained within the model object, we only need to estimate the five parameters associated with the ground intensity function. The function `expmap` below restricts the estimation to only these five parameters. Further, the model parameters are all positive. By allowing the minimizers to work with the logarithms of the estimated parameters, values on the entire real line can be selected (hence no boundary problems), and these get mapped back onto the positive real line within the function `expmap`.

Initially the function `optim` is used to estimate the parameters. Estimates from this are then used as starting values for `nlm`. The function `optim` uses, by default, the Nelder-Mead simplex algorithm (Nelder and Mead 1965) which is generally more robust to poor starting values. The function `nlm` uses a steepest ascent method, based on numerical derivatives. It is much more sensitive to poor initial values, but convergence is relatively fast with good starting values.

```
R> expmap <- function(y, p) {
+    y$params[1:5] <- exp(p)
+    return(y)
+  }
R> initial <- log(params[1:5])
R> z <- optim(initial, neglogLik, object = x, pmap = expmap,
+    control = list(trace = 1, maxit = 100))
R> initial <- z$par
R> z <- nlm(neglogLik, initial, object = x, pmap = expmap,
+    print.level = 2, iterlim = 500, typsize = initial)
R> x0 <- expmap(x, z$estimate)
```

4.2. Full model

In this subsection we fit a model to the Phuket earthquake data (Figure 1) with an ETAS ground intensity function (Equation 3) and a gamma density for the event magnitudes (mark distribution) as in Equation 9. We use the model object `x` that has already been defined in Section 4.1.

```
R> allmap <- function(y, p) {
+    y$params <- exp(p)
+    return(y)
+  }
R> initial <- log(c(0.05, 3.1, 13, 0.02, 1.1, 1/mean(Phuket$magnitude), 0.1))
R> z <- optim(initial, neglogLik, object = x, pmap = allmap,
+    control=list(trace = 1, maxit = 200))
R> initial <- z$par
R> z <- nlm(neglogLik, initial, object = x, pmap = allmap,
+    print.level = 2, iterlim = 500, typsize = initial)
R> x1 <- allmap(x, z$estimate)
```

4.3. Goodness of fit

The log-likelihood of both fitted models be calculated as follows.

	Null model	Full model
$p_1 = \mu$	0.0540	0.0542
$p_2 = A$	3.1512	3.0949
$p_3 = \alpha$	1.3429	1.3547
$p_4 = c$	0.0211	0.0211
$p_5 = p$	1.1205	1.1192
p_6	2.6931	2.9490
p_7	0	0.0290
$\log L$	309.6458	320.4775

Table 1: Parameter estimates of the ETAS model (Equation 3) fitted to the Phuket earthquake of 2004-12-26 (00:58:53.45 GMT) and aftershock sequence. The "full model" refers to that model using a gamma distribution which is dependent on the history for the event magnitudes, and the "null model" refers to that model where the magnitudes have an exponential distribution which is independent of the history.

```
R> print(logLik(x0))
R> print(logLik(x1))
```

The log-likelihood and parameter estimates of the (null) model with exponential magnitudes and the (full) model with gamma magnitudes are listed in Table 1. Using only one extra parameter, the difference between the log-likelihoods is quite significant. The difference between the parameters associated with the ETAS ground intensity function are quite small. The ground intensity function can be plotted as `plot(x1, log = TRUE)` (lower panel of Figure 1) and a summary of the model object as `summary(x1)`.

A graphical representation of the model goodness of fit is to plot the residual process (Equation 8) as described in Section 2.4. The code is written below and the resultant graph is shown in Figure 2.

```
R> plot(residuals(x1), xlab = "Event Number", ylab = "Transformed Time",
+    pty = "s")
R> points(residuals(x0), lty = 2, type = "l", col = "red")
R> big <- list(id = c(35, 511, 733, 875, 972, 1124, 1194),
+    date = c("26Dec04", "28Mar05", "24Jul05", "16May06", "12Sep07",
+      "20Feb08", "27Jun08"))
R> axis(3, at = big$id, labels = big$date, las = 2)
R> abline(v = big$id, lty = 3, col = "blue")
R> abline(a = 0, b = 1, lty = 3, col = "blue")
```

An alternative (see discussion in Section 2.4) is to plot the cusum process. The code is written below and the resultant graph is shown in Figure 3.

```
R> n <- nrow(Phuket)
R> plot(residuals(x1) - 1:n, xlab = "Event Number", ylab = "Cusum")
```

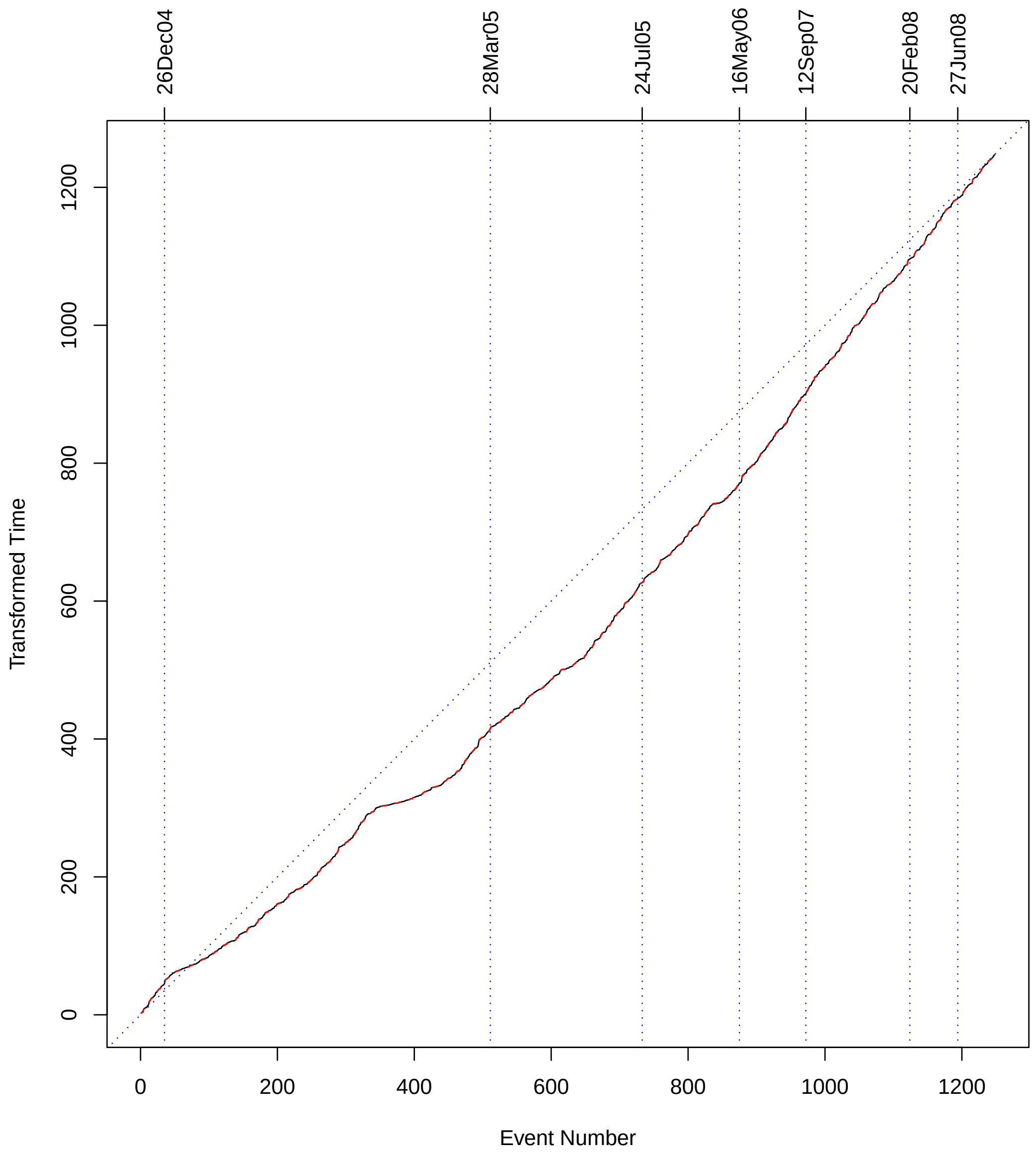

Figure 2: Residual process times for the ETAS model (Equation 3) fitted to the Phuket earthquake of 2004-12-26 (00:58:53.45 GMT) and aftershock sequence. The solid black line represents the model using a gamma distribution that is dependent on the history for the event magnitudes, and the dashed red line represents the model where the magnitudes have an exponential distribution that is independent of the history. The dashed line cannot be distinguished from the solid line in this plot. The dates of the larger events are marked on the top axis.

```
R> points(residuals(x0) - 1:n, lty = 2, type = "l", col = "red")
R> axis(3, at = big$id, labels = big$date, las = 2)
R> abline(v = big$id, lty = 3, col = "blue")
R> abline(a = 0, b = 0, lty = 3, col = "blue")
```

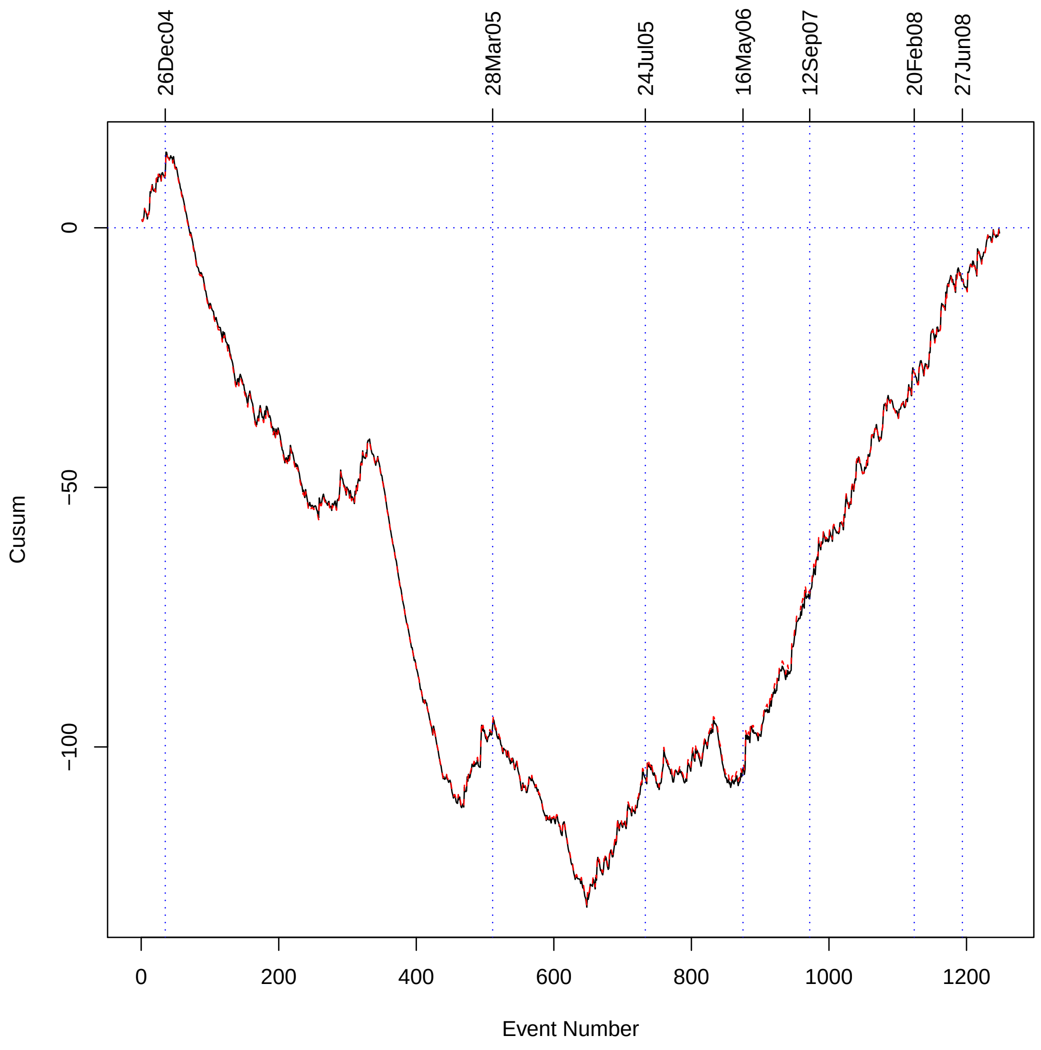

Figure 3: Cusum of residual process times for the ETAS model (Equation 3) fitted to the Phuket earthquake of 2004-12-26 (00:58:53.45 GMT) and aftershock sequence. The solid black line represents the model using a gamma distribution that is dependent on the history for the event magnitudes, and the dashed red line represents the model where the magnitudes have an exponential distribution that is independent of the history. As in Figure 2, the difference between the cusum residual times for both models is also very small.

Plots of the residual process are essentially checking the goodness of fit of the ground intensity function. Figures 2 and 3 indicate that there is very little difference between these functions. This is consistent with the very similar parameter estimates for the two models as listed in Table 1.

A useful diagnostic to evaluate the goodness of fit of the mark distribution is to plot their cusums over time. Under the null model, the magnitudes have an exponential distribution with mean $1/p_6$. Similarly, in the full model the expected value of the magnitude of the event occurring at t_i is

$$\frac{1 + \sqrt{\lambda_g(t_i|\mathcal{H}_{t_i})}\, p_7}{p_6}\,.$$

The code written below compares the magnitude cusums over time for the two models, the graph being shown in Figure 4. This shows an obvious difference and is the main reason for the difference in the log-likelihood between the two models.

```
R> lambda <- etas_gif(data = x1$data, evalpts = x1$data$time,
+    params = x1$params[1:5])
R> mean1 <- (1 + sqrt(lambda) * x1$params[7]) / x1$params[6]
R> plot(ts(cumsum(x1$data$magnitude - mean1)),
+    ylab = "Cusum", xlab = "Event Number", ylim = c(-13, 20))
R> mean0 <- 1/x0$params[6]
R> points(ts(cumsum(x0$data$magnitude - mean0)), type = "l", lty = 2,
+    col = "red")
R> abline(h = 0, lty = 3, col = "blue")
R> axis(3, at = big$id[-1], labels = big$date[-1], cex.axis = 0.7, las = 2)
R> abline(v = big$id[-1], lty = 3, col = "blue")
```

4.4. Simulation

The Phuket example dataset has an observation period that ceases at midnight on 2009-01-01. Consider the situation where the current time is midnight on 2009-01-01 (time 1827), and we want to determine the probability distribution of the time to the next event with magnitude ≥ 6.5. This distribution can be determined empirically by simulation. In this example, we use the model with a gamma distribution for event magnitudes.

We need to define a stopping condition as a function (see code below). The object `data` will contain all historical events and simulated events up until the most recently simulated event. The variable names in the object `data` will be those from the original history. This function will be evaluated immediately after the marks have been simulated for the most recent event (see the `simulate` method for '`mpp`' objects in package **PtProcess**). If this event has a magnitude ≥ 6.5 (recall that $M_0 = 4.95$), then the function returns `TRUE`, which will cause the `simulate` method to stop simulating further events. We also change `x$TT=c(1827, Inf)`, and since `x$TT[2]` is infinity, the simulation can only stop by meeting the stopping condition.

Then 2000 simulations are performed, and from each the time to the first magnitude 6.5 event is recorded in the vector `y`. Finally a histogram of these times (in days from 2009-01-01) is plotted and shown in Figure 5.

```
R> stop.cond <- function(data) {
+    n <- nrow(data)
+    return(data$magnitude[n] >= 1.55)
+ }
R> x2 <- x1
R> x2$TT <- c(1827, Inf)
R> y <- rep(NA, 2000)
```

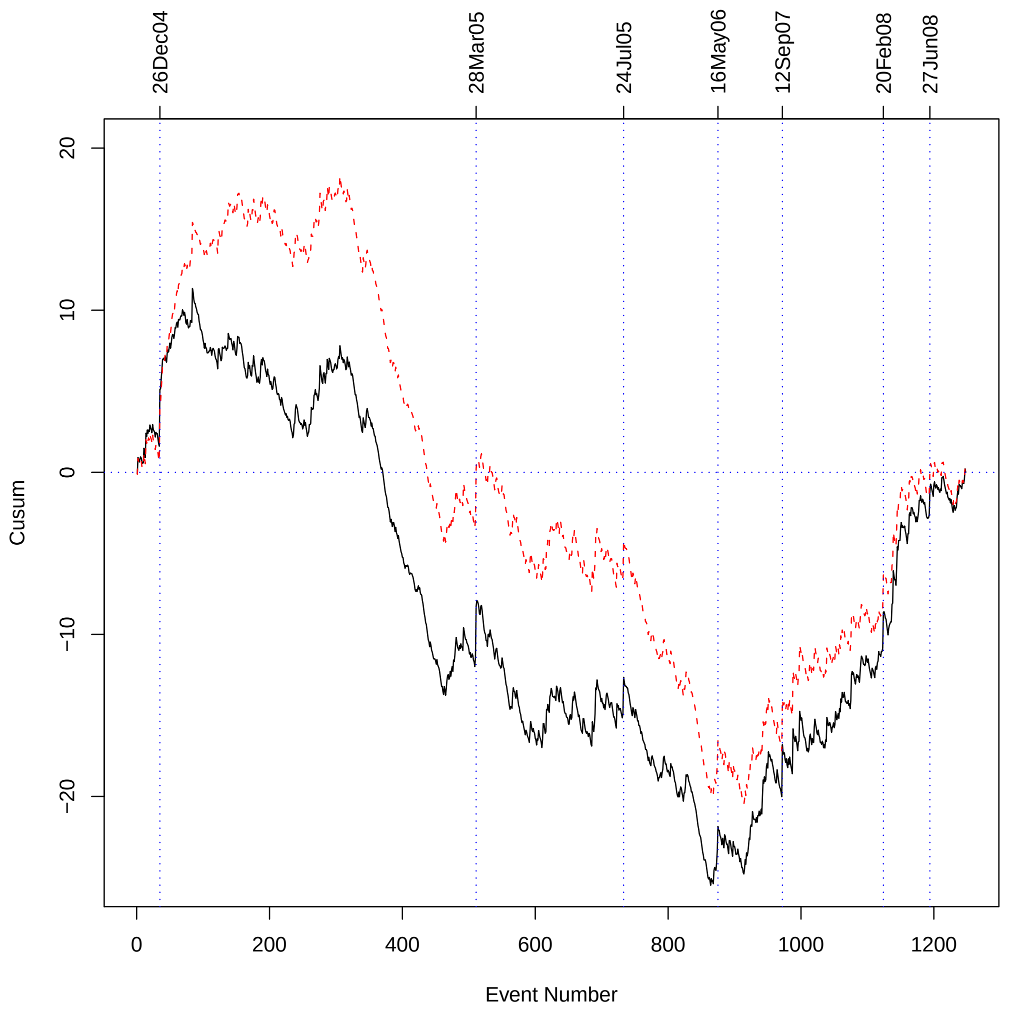

Figure 4: Cusum of event magnitudes for the ETAS model (Equation 3) fitted to the Phuket earthquake of 2004-12-26 (00:58:53.45 GMT) and aftershock sequence. The solid black line represents the model using a gamma distribution that is dependent on the history for the event magnitudes, and the dashed red line represents the model where the magnitudes have an exponential distribution that is independent of the history.

```
R> for (i in 1:2000) {
+    print(i)
+    x3 <- simulate(x2, seed = i, stop.cond = stop.cond)
+    y[i] <- x3$data$time[nrow(x3$data)]
+  }
R> hist(y - 1827, breaks = seq(0, max(y - 1827), length.out = 20),
+    xlab = "Days Since 1 January 2009", main = "")
R> z <- quantile(y - 1827, probs = c(0.99, 0.95, 0.90, 0.80, 0.50))
R> abline(v = z, lty = 2, col = "blue")
R> axis(3, at = z, labels = c(0.99, 0.95, 0.90, 0.80, 0.50))
R> box()
```

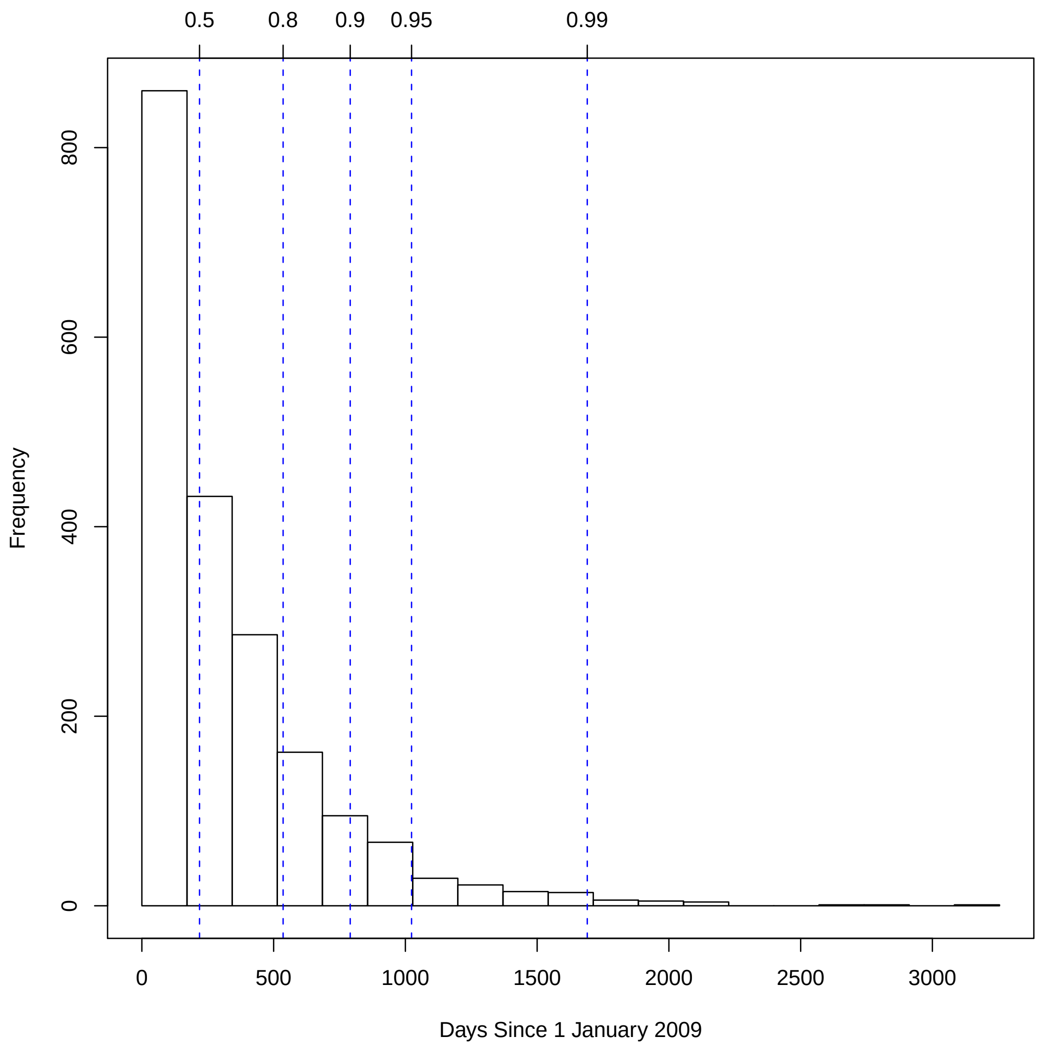

Figure 5: The ETAS model (Equation 3) with gamma magnitudes is initially fitted to the Phuket earthquake of 2004-12-26 (00:58:53.45 GMT) and aftershock sequence. The example dataset runs until midnight on 2009-01-01. 2000 simulations were commenced from this date until the first magnitude 6.5 event in each occurred. The histogram represents the empirical distribution of the times to this event. The dash lines represent the 0.5, 0.8, 0.9, 0.95 and 0.99 quantiles.

5. Current development

There are two problems where the current development of the package is being focussed. The first term in the log-likelihood function generally has a double sum. When upwards of 10,000 events are in the dataset, evaluation of the likelihood function is very slow. The second problem involves the product structure of the conditional intensity function given by Equation 6. In some models this condition can be too restrictive. In this section we briefly discuss these two problems, and the solutions being implemented.

5.1. Computational complexity

The main computational work in calculating the likelihood function, and hence fitting a model, is the calculation of the term

$$\sum_{i\,:\,t_i \in \mathcal{T}} \log \lambda_g(t_i|\mathcal{H}_{t_i})$$

in Equation 7. This is because it is generally conditional on the history of the process. Hence, given a value of t, we must effectively look through the historical events to see how they affect the ground intensity at t, which will occur as a summation like operation. In some cases, some sufficient statistics can be calculated (e.g., see function `linksrm1_gif`) and stored for subsequent function calls, and in others all calculations are repeated every time the function is called (e.g., see function `etas_gif`).

One way to speed up the calculation of this term is to use parallel processing. We have recently implemented this using the R package called **snow**. One question with parallel processing is at what level in the program structure should it be implemented, for example, at a relatively fine level like each time we multiply matrices, or at a much coarser level? If one is using a cluster of nodes (machines) each with multiple processors, the efficiency is very dependent on the speed of communication between processors. In clusters this communication is very slow compared to a super computer. Hence, it is best to allocate large amounts of work to the nodes, and minimize communication. We have implemented the parallel processing at a relatively coarse level, being done within the `logLik` method. This ensures that some benefits can be had when using any ground intensity function. Further, the most computationally intensive aspects of these models is the model fitting. Implementation within the likelihood function also ensures benefits during estimation too.

If there are n events, then the double sum in the first term of Equation 7 will contain approximately $n(n+1)/2$ terms. There may be more if there are further historical events that occur before those that occur within the interval $\mathcal{T}$. These terms are then split equally between the available processors. When we use two processors on the same node (machine) with a dataset of about 8,300 events, we get an increase in speed of about 1.8 over a single processor. When we use 6 processors (all roughly of comparable speed), 2 on each of 3 nodes, we get an increase in speed of about about 4 over a single processor. This reflects the slow connection between nodes compared to the much faster connection between processors on the same node. The speed factors reduce for datasets of fewer events. The test model was that discussed in the following section (Section 5.2). No parallel processing was applied to other terms in the likelihood function.

5.2. Spatial distribution as a mark

In Section 2, it was shown that the main aspects of modelling a *marked point process* involve the ability to calculate values of the ground intensity function $\lambda_g(t|\mathcal{H}_t)$, calculate the integral of the ground intensity function, and calculate values of the mark density $f(y|\mathcal{H}_t)$. It also requires the condition that the conditional intensity function can be expressed as a product as in Equation 6.

In the case of earthquake modelling, we could, therefore, include the spatial location of the event, the rupture length, rupture direction, and other measured information about a given event. The catch is that the conditional intensity function must satisfy Equation 6. This can put a considerable constraint on the form of the model. To see the problem, consider again

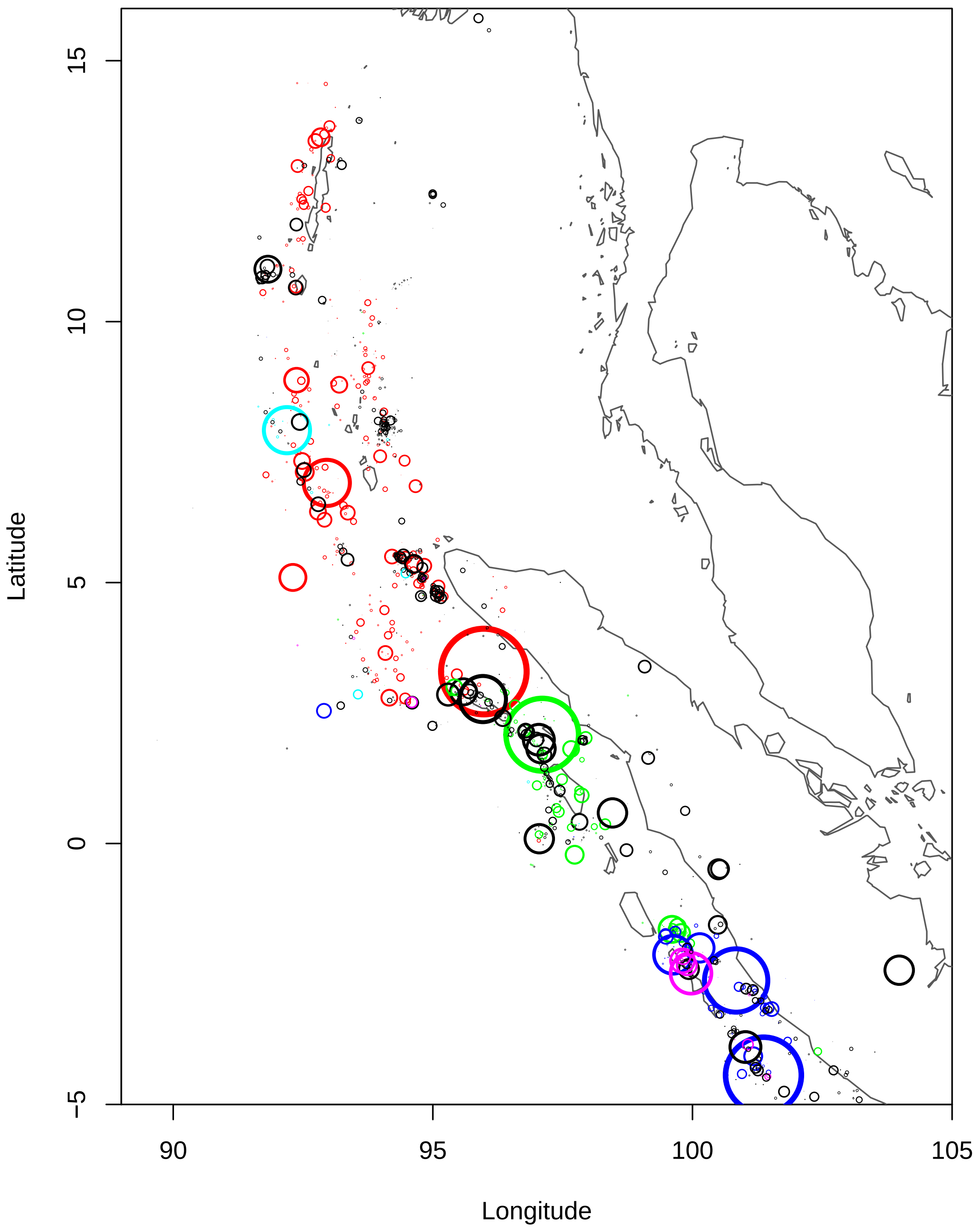

Figure 6: Epicentral plot of events in the Phuket example dataset. The size of the circle is proportional to the event magnitude. The location of the event on 2004-12-26 and all events during the 30 days afterwards are colored red. Similarly, the event locations on 2005-03-28, 2005-07-24, 2007-09-12 and 2008-02-20, and events occuring in the 30 days after each are colored green, cyan, blue and magenta, respectively. Other events are represented in black. The same events are plotted in Figure 1. The map is drawn using the R packages by Becker *et al.* (2010b,a).

the ETAS model. Let

$$\xi_i(t) = e^{\alpha(M_i - M_0)} \left(1 + \frac{t - t_i}{c}\right)^{-p}$$

then the ground intensity function of the non-spatial ETAS model given by Equation 3 can be written as

$$\lambda_g(t|\mathcal{H}_t) = \mu + A \sum_{i:t_i<t} \xi_i(t). \tag{10}$$

If a spatial version of this model was to be a marked point process, then we would require a density function to describe the spatial distribution of the events, say $f(x, y)$, such that the product with $\lambda_g(t|\mathcal{H}_t)$, as in Equation 6, gives the conditional intensity function $\lambda(t, x, y \mid \mathcal{H}_t)$. Unfortunately this model has the wrong structural form.

Consider the following argument. Given Equation 10, it is natural to think of the non-spatial ETAS model as a composite model, i.e., a composition of many processes. It contains a Poisson part with rate parameter μ, and the term $A\xi_i(t)$ can be thought of as the ground intensity function of the aftershock sequence associated with the ith event. We require $\xi_i(t)$ to be integrable so that the ith family line dies out, i.e., the aftershocks associated with the ith event eventually stop. Now assume that the ith event has spatial location (x_i, y_i) (in 2D, disregarding depth) and magnitude M_i. Then we would expect the aftershock sequence to be located relatively close to (x_i, y_i), and the spatial spread would be expected to be larger if the magnitude M_i was greater. This suggests that a possible very simplistic mark density to describe the spatial distribution of the aftershock sequence associated with the ith event could be something like

$$f_i(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{(x - x_i)^2 + (y - y_i)^2}{-2\sigma^2}\right),$$

where $\sigma^2 = de^{\beta(M_i - M_0)}$, and d and β are estimable parameters (see Ogata and Zhuang (2006) for other possibilities). Then a possible form of the conditional intensity function of a spatial ETAS model is

$$\lambda(t, x, y|\mathcal{H}_t) = \zeta(x, y) + A \sum_{i:t_i<t} \xi_i(t) f_i(x, y), \tag{11}$$

where, given that $\mathcal{X} \times \mathcal{Y}$ is the observation region,

$$\int_{\mathcal{Y}} \int_{\mathcal{X}} \zeta(x, y)\, dx\, dy = \mu,$$

and

$$\int_{\mathcal{Y}} \int_{\mathcal{X}} f_i(x, y)\, dx\, dy = B(x_i, y_i) < 1.$$

The function $B(x_i, y_i)$ is effectively a boundary correction accounting for the *finite* spatial observation region (i.e., $\mathcal{X} \times \mathcal{Y}$). It can be seen that the summation in Equation 11 is being taken over many marked point process models. This composite marked point process model still has one important structural characteristic of the more simple marked point process model. If we integrate over the mark space, the marginal intensity is similar to the overall ground intensity function (Equation 10), except for the inclusion of the boundary correction, i.e.,

$$\int_{\mathcal{Y}} \int_{\mathcal{X}} \lambda(t, x, y|\mathcal{H}_t)\, dx\, dy = \mu + A \sum_{i:t_i<t} B(x_i, y_i)\, \xi_i(t),$$

denoted by $\widetilde{\lambda}_g(t|\mathcal{H}_t)$, say. Because of this, a number of the *generic methods* written for the simple marked point process work also in this situation with only minor modifications. For example, we could simulate a spatial ETAS point process by generating the time to the next event based on the current value of $\widetilde{\lambda}_g(t|\mathcal{H}_t)$ as in the simple case[4]. Once a new event time has been simulated (say t_j), the event would be allocated to one of the existing clusters or the background process, depending on the relative values of μ and $B(x_i, y_i)\xi_i(t_j)$ for all $t_i < t_j$. Other marks, including magnitude, would be allocated to the new event at this stage as well. Estimation of the process is reasonably straightforward too. The **PtProcess** package currently has a prototype function for this situation.

The isotropic mark density function defined above is obviously unsatisfactory to model the Phuket example dataset. In Figure 6 we see that the spatial distributions of the aftershocks associated with the largest mainshocks are not isotropic, and further, they are not even centred about the main shocks. The aftershocks of the 2004 Boxing Day event tend to occur to the north. This often occurs, i.e., the mainshock can be on the periphery of what will develop as the aftershock region. There is some current argument (McGuire, Zhao, and Jordan 2002) that various second moment properties of the mainshock provide information of whether the aftershock region will be offset in this manner. If this is true, these measured properties should be built into the spatial density function. Further, note the alignment of the events within a long thin approximately elliptical region whose major axis is roughly parallel to the island of Sumatra. One may also want to consider functions with a powerlaw decay rather than the exponential decay of the bivariate normal density function, for example, see Ogata and Zhuang (2006).

Acknowledgements

I would like to thank David Vere-Jones and Robert Davies for their interest and help. This work was done as part of a Foundation for Research, Science and Technology (FRST) subcontract with the Institute of Geological and Nuclear Science, Lower Hutt, New Zealand.

References

Aalen OO, Hoem JM (1978). "Random Time Changes for Multivariate Counting Processes." *Scandinavian Actuarial Journal*, **1978**(2), 81–101.

Baddeley A, Turner R (2005). "**spatstat**: An R Package for Analyzing Spatial Point Patterns." *Journal of Statistical Software*, **12**(6), 1–42. URL `http://www.jstatsoft.org/v12/i06/`.

Baddeley A, Turner R (2010). ***spatstat**: Spatial Point Pattern Analysis, Model Fitting, Simulation, Tests.* R package version 1.19-3, URL `http://CRAN.R-project.org/package=spatstat`.

Becker RA, Wilks AR, Brownrigg R (2010a). ***mapdata**: Extra Map Databases.* R package version 2.1-3, URL `http://CRAN.R-project.org/package=mapdata`.

[4]For this particular model, it is more efficient to simulate the data for a given family line until that line becomes extinct or the final time boundary has been exceeded; then consider the next family line.

Becker RA, Wilks AR, Brownrigg R, Minka TP (2010b). ***maps**: Draw Geographical Maps.* R package version 2.1-4, URL `http://CRAN.R-project.org/package=maps`.

Daley DJ, Vere-Jones D (2003). *An Introduction to the Theory of Point Processes. Volume I: Elementary Theory and Methods.* 2nd edition. Springer-Verlag, New York. ISBN 0-387-95541-0.

Harte DS (1998). "Documentation for the Statistical Seismology Library." *School of Mathematical and Computing Sciences Report 98-10*, Victoria University of Wellington.

Harte DS (2010). ***PtProcess**: Time Dependent Point Process Modelling.* R package version 3.2-5, URL `http://CRAN.R-project.org/package=PtProcess`.

Lewis PAW, Shedler GS (1979). "Simulation of Nonhomogeneous Poisson Processes by Thinning." *Naval Research Logistics Quarterly*, **26**(3), 403–413. `doi:10.1002/nav.3800260304`.

Lucas JM (1985). "Counted Data Cusums." *Technometrics*, **27**(2), 129–144. `doi:10.2307/1268761`.

McGuire JJ, Zhao L, Jordan TH (2002). "Predominance of Unilateral Rupture for a Global Catalog of Large Earthquakes." *Bulletin of the Seismological Society of America*, **92**(8), 3309–3317. `doi:10.1785/0120010293`.

Nelder JA (1977). "A Reformulation of Linear Models." *Journal of the Royal Statistical Society A*, **140**(1), 48–76. `doi:10.2307/2344517`.

Nelder JA, Mead R (1965). "A Simplex Method for Function Minimization." *Computer Journal*, **7**(4), 308–313. `doi:10.1093/comjnl/7.4.308`.

Nelder JA, Wedderburn RWM (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society A*, **135**(3), 370–384. `doi:10.2307/2344614`.

Ogata Y (1981). "On Lewis' Simulation Method for Point Processes." *IEEE Transactions on Information Theory*, **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. `doi:10.2307/2288914`.

Ogata Y (1998). "Space-Time Point-Process Models for Earthquake Occurrences." *Annals of the Institute of Statistical Mathematics (Tokyo)*, **50**(2), 379–402. `doi:10.1023/A:1003403601725`.

Ogata Y (1999). "Seismicity Analysis Through Point-Process Modeling: A Review." *Pure and Applied Geophysics*, **155**(2-4), 471–507. `doi:10.1007/s000240050275`.

Ogata Y, Zhuang JC (2006). "Space-Time ETAS Models and an Improved Extension." *Tectonophysics*, **413**(1-2), 13–23. `doi:10.1016/j.tecto.2005.10.016`.

Page ES (1954). "Continuous Inspection Schemes." *Biometrika*, **41**(1-2), 100–115. `doi:10.1093/biomet/41.1-2.100`.

Peng R (2003). "Multi-Dimensional Point Process Models in R." *Journal of Statistical Software*, **8**(16), 1–27. URL http://www.jstatsoft.org/v08/i16/.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. ISBN 3-900051-07-0. URL http://www.R-project.org/.

Schoenberg FP (2003). "Multidimensional Residual Analysis of Point Process Models for Earthquake Occurrences." *Journal of the American Statistical Association*, **98**(464), 789–795. doi:10.1198/016214503000000710.

Vere-Jones D (2009). "Some Models and Procedures for Space-Time Point Processes." *Environmental and Ecological Statistics*, **16**(2), 173–195. doi:10.1007/s10651-007-0086-0.

Zhuang JC (2006). "Second-Order Residual Analysis of Spatiotemporal Point Processes and Applications in Model Evaluation." *Journal of the Royal Statistical Society B*, **68**(4), 635–653. doi:10.1111/j.1467-9868.2006.00559.x.

Affiliation:

David Harte
Statistics Research Associates
PO Box 12 649
Wellington 6144, New Zealand
E-mail: david@statsresearch.co.nz
URL: http://www.statsresearch.co.nz/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/

Volume 35, Issue 8 *Submitted:* 2009-06-11
July 2010 *Accepted:* 2010-05-03