Event Series Prediction via Non-homogeneous Poisson Process Modelling

James Goulding
N-LAB
University of Nottingham
james.goulding@nottingham.ac.uk

Simon Preston
School of Mathematics
University of Nottingham
simon.preston@nottingham.ac.uk

Gavin Smith
N-LAB
University of Nottingham
gavin.smith@nottingham.ac.uk

***Abstract*—Data streams whose events occur at random arrival times rather than at the regular, tick-tock intervals of traditional time series are increasingly prevalent. *Event series* are continuous, irregular and often highly sparse, differing greatly in nature to the regularly sampled time series traditionally the concern of hard sciences. As mass sets of such data have become more common, so interest in predicting future events in them has grown. Yet repurposing of traditional forecasting approaches has proven ineffective, in part due to issues such as sparsity, but often due to inapplicable underpinning assumptions such as stationarity and ergodicity.**

In this paper we derive a principled new approach to forecasting event series that avoids such assumptions, based upon: 1. the processing of event series datasets in order to produce a parameterized mixture model of non-homogeneous Poisson processes; and 2. application of a technique called *parallel forecasting* that uses these processes' rate functions to directly generate accurate temporal predictions for new query realizations. This approach uses forerunners of a stochastic process to shed light on the distribution of future events, not for themselves, but for realizations that subsequently follow in their footsteps.

I. Introduction

Data streams whose events occur at random arrival times rather than at the regular, tick-tock intervals of traditional time series are becoming increasingly prevalent. This is due in no small part to the upsurge in human behavioural data now being recorded in the form of transactional logs. Such *event series* are continuous, irregular and often highly sparse, and differ greatly in nature to the metronomic, regularly sampled time series that have traditionally been the concern of econometrics, engineering and the hard sciences. As mass sets of event series have become increasingly commonplace, so interest in predicting future events from them has also grown. Application areas are diverse, and include: earthquake forecasting [17], neural spike analysis [3], forecasting of health episodes [15]; financial event prediction [4]; and finding the limits of movement predictability [23], [22].

Unfortunately, while a huge amount of prior research has been undertaken into forecasting time series, it has been shown that event series do not generally avail themselves to the field's techniques [15], [6], [18]. Any attempt to use those techniques by converting event series into a time-series format (traditionally by aggregating event counts into temporal bins) introduces either information loss or extreme sparsity. Both situations severely impact on the predictive performance of traditional predictive algorithms [26]. Worse still, when traditional techniques developed for time series (whether ARIMA models or Markov chains) are applied to an event stream one of their key underpinning assumptions is brought into doubt - that the series contains enough structure *within its own history* to facilitate robust prediction of its future. In order to hold, such an assumption requires:

- *Stationarity*: the statistical properties of a process must not change over time (or be transformable into that condition), otherwise patterns that have occurred in the past will not be robust predictors of the future.
- *Ergodicity*: the event series under consideration must contain sufficient information to deduce all or the properties of its generating process. If this does not hold, and the realization only casts light on some small part of the process' underlying nature, statistical inference cannot be confidently used to predict that process' future events.

For most real-world event series, and especially those generated via human behaviour, satisfying these requirements is unlikely: individual realizations are often too sparse and irregular to assume ergodicity; and even if the process were ergodic and we *could* observe a realization for sufficiently long time, little real world behaviour is stationary.

This paper presents a principled new approach to forecasting event series that avoids these assumptions. The technique is based upon two key steps: 1. the processing of event series datasets in order to produce a parameterized mixture model of non-homogeneous stochastic *point processes*; and 2. application of a technique called *parallel forecasting* that uses the discovered processes' rate functions to generate accurate temporal predictions for new query realizations. This technique is tested via both synthetic and real world data, comparing to a range of contemporary techniques.

II. Parallel Forecasting of Point Processes

The proposed model views a set of n event series realizations as having been generated from a mixture model of Z non-stationary point processes (where $Z < n$), with each individual process describing the probability that an event will occur at timepoint, t. If we can correctly ascertain the nature of those underlying processes, and subsequently classify a *query* event series to its correct parent process, knowledge of that process' rate function can be leveraged in order to

Published in the Proceedings of the IEEE International Conference on Data Mining (ICDM 2016).

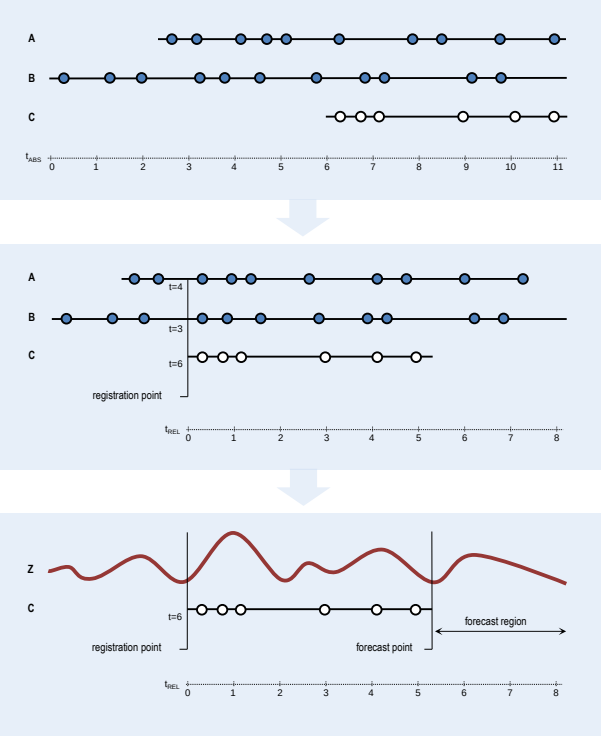

Fig. 1. Illustration of Parallel Forecasting. Step 1 - here it's recognised that event series A and B (along with query series C) are generated by the same underlying process, Z, but are asynchronous having begun that process at different absolute times. Step 2 shows the commonalities more clearly via a registration process (the absolute time points of A, B and Q are actually aligned at $t = 4$, $t = 3$ and $t = 6$ respectively). Step 3 shows how the stochastic model created from A and B extends past the endpoint of Q, producing a labelled forecast region up to $t = 8$ in relative time.

estimate a distribution of the series' future event times. From analyzing this distribution we can then establish the most likely timings of future events, and therefore start making robust predictions - even when the query realization's history is itself sparse and/or simply contains insufficient information to make accurate event forecasts on its own.

A. A non-stationary, non-ergodic model for sparse event series

As illustrated in Figure 1, this approach is only possible by using forerunner realizations from a stochastic process to shed light on the distribution of future events, not for themselves, but for realizations that will subsequently follow in their footsteps. Such a model no longer requires an individual time series to be stationary - however, it does require realizations generated from a similar generative process to have occurred in the past. The approach also no longer requires ergodicity - instead it relies on sufficient event series realizations generated from the same process to have occurred in the past. Further, treating event-series as samples of an overarching parent process helps to ameliorate issues of sparsity in any *individual* series. In an era of 'Big Data' we believe that 1. trading ergodicity assumptions for a greater number of samples; and 2. dropping stationarity requirements by assuming instead that individuals follow common patterns of time-lagged behaviour; is a preferable situation in which to position a predictive model.

B. Non-homogeneous Poisson processes

In our implementation we model the data as asynchronous realizations of *Non-homogeneous Poisson processes* (NHPPs). While other options exist, NHPP have a strong history in successfully modelling the irregular and continuous characteristics of event series. NHPP are stochastic in nature and underpinned by a time-varying *rate function*, $\lambda(t)$, that describes the probability at time t that an event will occur. An NHPP is also an example of a counting process which can be defined by via the probability that the *no. events* occurring between two timepoints (t_1 and t_2) will be some count, c:

$$P(N(t_1, t_2) = c) = e^{-\Lambda(t_1,t_2)} \frac{\Lambda(t_1, t_2)^c}{c!} \quad (1)$$

$$\text{where:} \quad \Lambda(t_1, t_2) = \int_{t_1}^{t_2} \lambda(t)dt$$

Moreover, the assumption of event independence made by NHPP allows us to easily treat multiple realizations in parallel. Due to this, it will be shown in §V that it is possible to obtain an optimal prediction, π, of the time until the next event, x, for any event series we believe generated by the process.

NHPP have never previously been used in this fashion, likely due to the fact that when NHPP are estimated from a realization the resulting process is time bounded to that realization's endpoint - and thus cannot make predictions past that time. Parallel forecasting frees us from this constraint to a limited extent, allowing us to make predictions up to the final timepoint of the *longest item in a family of realizations*. Therefore, if we can accurately classify a new query realization as also belonging to that family, and it is currently shorter than that family's longest realization, an informed prediction can be made for its future event times. The more examples that exist, the greater confidence we may have in those predictions.

III. Related Work

While NHPP have not been used for direct forecasting of event series, they have been used for other forms of predictive tasks, such as predicting the shape of rate functions. Variations of Poisson processes have long been used in domains such as queueing theory, with call centre practitioners estimating a rate function to provide a look-up of the expected call arrival rates at different points in the day [9], [25]. Shen and Huang [19] extended this notion, decomposing a set of daily rate functions via SVD, treating the resulting vectors as multivariate time series and applying traditional ARIMA models.

NHPP are of course not the only way to model non-stationary event series. Alternative approaches generally fall into two distinct classes. The first consists of Continuous Time Markov Chain (CTMC) models and their extensions (e.g. [14], [5]), which are an adaption of standard Markov Chains that allow for transitions between states in continuous time. This is achieved by replacing a Markov model's state transition probability matrix with an intensity matrix that jointly encodes distributions of how long the system will stay in the current state and the expected transition time to each potential new one [16]. While CTMC more closely reflects the continuous,

irregular nature of event series than discrete time models, the paradigm still focuses on state space transitions rather than directly modelling the generative event processes themselves.

The second class of continuous time techniques differ in that they do explicitly model sequences of event occurrences in continuous time. These models include Poisson Networks and Cascades [21], CT-NOR models [20], Gaussian-process-modulated renewal process models [24], [10], Piecewise-Constant Conditional Intensity Models (PCIMs) [7] and Multiplicative-Forest Point Processes (MFPPs) [26]. Several of these approaches achieve very effective results, but often at the cost of being highly parameterized and requiring *a priori* understanding of the process being modelled. For example, [20], [21] propose a model in which each event triggers a new mixture of Poisson process, trading parsimony for the ability to reason about process interaction and at the cost of non-trivial parametrisation of *delay* and *transition distribution* choice.

In a similar vein, both [24] and [10] propose the use of Gaussian-Process-Modulated Renewal Processes. These assume that the collection of rate functions (for an underlying Poisson [24] or Gamma [10] process) across time are realised as the interaction of a Gaussian process and a hazard function. This allows for certain non-stationary behaviours to be modelled, but assumes that means and co-variances across temporal instants follow a fixed functional form. Learning then involves deducing the best coefficients for the functional form one chooses. The difficulty of picking the correct function at design time, and specifying which restrictions should be placed on it (e.g. stationarity, or isotropicity) stands in contrast to the broadly unparameterized nature of NHPP.

In terms of a parallel forecasting approach both [7] and [26] represent the closest work to that presented here. Both of these propose models designed for multi-event point processes, based on a piecewise constant approximation of the process' rate function. Each piecewise segment is represented by a conditional intensity model in which the probability of a given rate is conditioned on the existence or absence of events within a set of lagged historical windows. Prediction occurs by building a decision tree reflecting the conditional relationships between each of these windows. PCIMs [7] use manually specified and problem specific windows, whereas MFPPs [26] extend this approach by learning intensities conditional on a more complex set of interactions between those windows. Importantly both PCIM and MFPP are non-parametric, were designed specifically for forecasting and, having achieved best reported results in the literature, will be used as the prime basis for comparison to our proposed model.

IV. Model

The model we present is a new next-step prediction technique made possible through derivation of a mixture model of aligned point processes. Given a set of historical event series $\mathcal{T} = \{\overrightarrow{\mathbf{t}}_1, \overrightarrow{\mathbf{t}}_2, \ldots, \overrightarrow{\mathbf{t}}_m\}$, the model assumes that each realization, $\overrightarrow{\mathbf{t}}_i = \langle t_{i1}, t_{i2}, \ldots, t_{ij}\rangle$, was generated from one of Z parent NHPP processes. As detailed in §II-B, each of these processes is characterized by a time-varying rate function $\lambda_z(t)$. From this rate function, for each process we can derive a cumulative density function, $F_z^s(x)$, that describes the times we expect to wait from some given point s for a new event to occur (see Appendix A for a proof):

$$F_z^s(x) = 1 - e^{-\Lambda_z(s,s+x)} \tag{2}$$
$$\text{where:} \quad \Lambda_z(s, s+x) = \int_s^{s+x} \lambda_z(t)dt$$

This cdf will be useful for prediction, but it also allows us to derive a joint probability density function, $g_z(\overrightarrow{\mathbf{t}})$, describing the probability of any specific series of event times being produced by the process (see Appendix C for a full proof):

$$g_z(\overrightarrow{\mathbf{t}}|\tau) = e^{-\Lambda_z(0,\tau)} \prod_{j=1}^{n} \lambda_z(t_j) \tag{3}$$

For each process the mixture model takes this pdf, and assigns it a 'mixing' probability, p_z, reflecting the chance that any event series in the historical dataset $\mathcal{T}$ was drawn from it. Thus the likelihood function for the model as a whole is:

$$L(\theta|\mathcal{T}) = \prod_{i=1}^{m} \sum_{z=1}^{Z} p_z \, g_z(\overrightarrow{\mathbf{t_i}}) \tag{4}$$

Here $\theta = (\overrightarrow{\lambda}, \overrightarrow{\mathbf{p}})$ represents the model's parameters and correspond to the processes' rate functions $\langle \lambda_1, \lambda_2, \ldots, \lambda_Z \rangle$ and mixing probabilities $\langle p_1, p_2, \ldots, p_Z \rangle$ respectively. A variety of ad-hoc algorithms might be used to cluster realizations into groups and then estimate characteristics of assumed parent processes. Instead, we present a principled method for estimating the set of parent rate functions via expectation maximisation, followed by a technique to directly predict future events in query event series from the model.

A. Constructing the Rate Functions

The flexibility of this NHPP-based model lies in the fact that each $\lambda_z(t)$ can be a function *of any form*. In practice we must build these from some combination of basis functions. Here numerous options exist, such as linear combinations of polynomials, Fourier series or exponentials. In our case, we model $\lambda(t)$ as the sum of K weighted Gaussian bases, $\{B_1, B_2, \ldots B_K\}$, all with unit area and identical variances, but with means spread uniformly between 0 and endpoint τ:

$$\lambda_z(t) = \sum_{k=1}^{K} a_{zk} \, B_k(t) \tag{5}$$
$$\text{where:} \quad B_k(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu_k)^2/2\sigma^2}$$
$$\text{and:} \quad \mu_k = \frac{(k-1)\,\tau}{K-1}, \; \sigma = \tau/k$$

Note that it is only the weightings, a_{zk}, that vary between processes. It would also be possible to allow means and variances to vary as hyperparameters of the model, and allow even more flexibility in rate function construction - but this is currently left for future work due to the additional computational load it would require during optimization. K was

uniformly set to 120 to ensure over-saturation of the temporal domain in our experiments (resulting in numerous $a_{zk} = 0$) and thus rendering the model parameterless outside of Z.

B. Expectation Maximization

With these building blocks, we now show how the model's parent processes can be found via EM. This requires estimation of the parameters in θ, which in our case distills to finding the weights attributed to each of the Z rate function's set of k Gaussian bases, a_{zk}. Recall from equation 4 that:

$$\begin{aligned} L(\theta|\mathcal{T}) &= \prod_{i=1}^{m} \sum_{z=1}^{Z} p_z \, g(\overrightarrow{\mathbf{t_i}}|\lambda_z(t)) \\ &= \prod_{i=1}^{m} \sum_{z=1}^{Z} p_z \, e^{-\Lambda_z(0,\tau)} \prod_{j=1}^{n} \lambda_z(t_{ij}) \end{aligned} \tag{6}$$

Given this Likelihood function, maximum likelihood estimates for each of the Z rate functions can be found by iteratively solving the EM algorithm's *expectation* and *maximisation* steps (note we indicate the current iteration by the superscript, r). Beginning with an initial estimate for θ:

Expectation Step: Given the current value for the parameters, θ^r, this step computes the probability that each event series, $\overrightarrow{\mathbf{t_i}}$, came from each of the parent processes. For the first iteration, these individual probabilities, p^r_{zi}, are calculated as:

$$p^r_{zi} = \frac{e^{-\Lambda_z(0,\tau)} \prod_{j=1}^{n} \lambda_z(t_{ij})}{\sum_{q=1}^{Z} e^{-\Lambda_q(0,\tau)} \prod_{j=1}^{n} \lambda_q(t_{ij})} \tag{7}$$

Maximization Step: With these probabilities in hand, new estimates are obtained for the parameters, θ^{r+1}, by solving:

$$\theta^{r+1} = \underset{\theta}{\operatorname{argmax}} \left(\sum_{i=1}^{m} \sum_{z=1}^{Z} p_{iz} \ln \left(p_z \, e^{-\Lambda_z(0,\tau)} \prod_{j=1}^{n} \lambda_z(t_{ij}) \right) \right)$$

In practice, this involves solving a concave maximization problem with respect to the set of basis Gaussian functions through substituting in equation 5. We therefore maximize[1]:

$$\sum_{i=1}^{m} \sum_{z=1}^{Z} p^r_{iz} \left(\ln p_z - \Lambda_z(0,\tau) + \sum_{j=1}^{n} \ln \sum_{k=1}^{K} a_{zk} \, B_k(t) \right) \tag{8}$$

Finally updates for the aggregate mixing probabilities, p_z, are calculated by simply averaging all of the individual membership probabilities, as follows:

$$p^{r+1}_z = \frac{\sum_{i=1}^{m} p^r_{iz}}{n} \tag{9}$$

This whole process then iterates, halting after a specified number of iterations or until convergence. When complete, we are left with final estimates for each weighting, a_{zk}, and hence the final form of each parent NHPP's rate function. Since the process we use to model rate functions currently uses fixed parameters and we oversaturate the number of Gaussian components required, the model has a single hyperparameter - the number of parent processes, Z.

[1]In our implementation we negate equation 8 and use the python *scipy.optimize.minimize()* function to perform this optimization step.

V. Making Predictions

In order to predict the next event in an event series, $\overrightarrow{\mathbf{t}}$, we must first assign it to one of our derived model's parent processes. To do this we simply determine which rate function has the highest (log) likelihood of producing the series:

$$\underset{z \in \{1,2,\ldots,Z\}}{\operatorname{argmax}} \quad \ln g_z(\overrightarrow{\mathbf{t}}) \tag{10}$$

Hence, for NHPP, we substitute in equation 3 to obtain:

$$\begin{aligned} \ln g_z(\overrightarrow{\mathbf{t}}) &= \ln \left(e^{-\Lambda_z(0,\tau)} \prod_{j=1}^{n} \lambda_z(t_j) \right) \\ &= -\int_0^{\tau} \lambda_z(t) dt + \sum_{j=1}^{n} \ln \lambda_z(t_{ij}) \end{aligned} \tag{11}$$

By substituting in the Gaussian bases from equation 5, the final function calculated for each of the Z candidate process in order to determine an optimal classification is:

$$\int_0^{\tau} \sum_{k=1}^{K} a_{zk} \, B_k(t) dt + \sum_{j=1}^{n} \ln \left(\sum_{k=1}^{K} a_{zk} \, B_k(t_{ij}) \right) \tag{12}$$

Note that it is at this point that classification of an event series could incorporate temporal alignment. Currently, however, we restrict ourselves to situations where alignment is easily performed in the data pre-processing stage (such as when constructing daily event sequences), and solve equation 12 directly. Incorporation of temporal alignment during model construction/classification is left as future work.

A. Predicting Time until the Next Event

Once a new event series has been classified, if it is *shorter* than the parent process to which it has been assigned, it is then possible to use that process to predict the series' next event. For this prediction, π, we use the median of the pdf of wait times, $f^s_z(x)$, to the process' endpoint (which is preferred to the mean of the pdf due to its mathematical properties). This occurs when its cdf, $F^s_z(\pi) = \frac{1}{2}$. From equation 2:

$$\begin{aligned} F^s_z(\pi) = 1 - e^{-\Lambda_z(s,s+\pi)} &= \frac{1}{2} \\ \Lambda_z(s, s+\pi_1) - \ln(2) &= 0 \end{aligned} \tag{13}$$

Because we are modelling the rate function via weighted Gaussian bases, we again substitute in equation 5 to yield:

$$\sum_{k=1}^{K} \int_s^{s+\pi} a_{zk} B_k(\pi) dt - \ln(2) = 0$$

By performing the integration, we arrive at the final equation we need to solve to produce our prediction:

$$\sum_{k=1}^{K} \frac{a_{zk}}{2} \left[\operatorname{erf} \left(\frac{s + \pi - \mu_k}{\sqrt{2}\sigma_k} \right) - \operatorname{erf} \left(\frac{s - \mu_k}{\sqrt{2}\sigma_k} \right) \right] - ln(2) = 0 \tag{14}$$

Importantly, given that the means, variances and the weighting parameters, a_{zk} have already been estimated in the model training stage, π is left as our only free variable so a solution can be found in rapid time again using a numerical solver.

VI. Implementation & Experiments

Experiments were performed on both synthetic and real world data, with comparison made to the predictors below[2].

Baseline predictors:

Global Mean Predictor (GMP): A baseline scalar predictor that forecasts time to next event as the mean gap between events across the whole training dataset.

Individual Mean Predictor (IMP): A predictor that, given timepoint s, forecasts time to next event as the mean gap between events in the event-series' history (therefore treating the query as an independent constant Poisson process).

Local Mean Predictor (LMP): This predictor works by calculating the mean time until next event from time s from all realizations in the training set extending past that point.

State-of-the-art predictors:

K-Nearest Trajectories (KNTP): Predictor that returns mean time until next event following time-point s across the k nearest realizations (via sub-sequence Dynamic Time Warping). In common use in the literature via [12], [8].

Piecewise-Constant Intensity Model (PCIM): This predictor treats the dataset as stemming from a hierarchical set of component piecewise constant rate functions, whose conditional relationships are modelled via a regression tree [7]. Prediction is made by simulating a series of realizations after s, and taking the median of first events occurring.

Multiplicative Forest Point Process (MFPP): An extension of PCIM prediction that applies theory in multiplicative forest continuous-time Bayesian networks. MFPP simulates using regression forests rather than single trees, for which evidence of improved results has been shown [26].

NHPP-based predictors:

Parallel Global Mean Predictor (P-GMP): Given the Z parent NHPPs whose rate functions have been established via EM (as detailed in §IV-B), a query realization is allocated to its most likely parent class but prediction then made simply using the mean inter-event gap for members of that class.

Parallel Local Mean Predictor (P-LMP): As above, but once the query is then classified, prediction is made using the mean next-event time after s for members of that class.

Parallel NHPP Predictor (P-NPP): Our proposed approach. Functions as above, but with prediction being made through direct integration of the assigned class' rate function following s (as detailed in §V). This returns the parent process' median next-event time as a prediction.

Global and individual baselines were selected in order to contrast behaviour of parallel predictors against extremes. We did not convert event series into time series via binning in order to apply traditional forecasting algorithms given the reported lack of effectiveness of this approach [15], [6], [18]. State-of-the art predictors were selected for comparison as discussed in the related work section.

[2]Python implementation code for all predictors and experiments is available at the paper's accompanying website, which can be found at *¡removed¿*

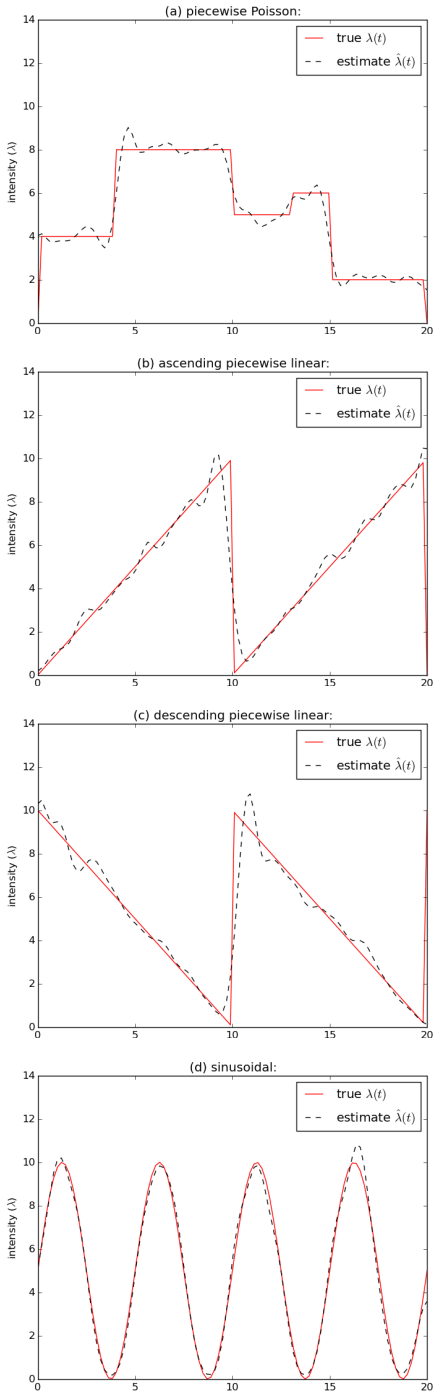

Fig. 2. Synthetic Non-stationary rate functions. Fitted models (dotted) illustrate 1. the approach's ability to model both sharp and gradual rate changes; and 2. how numerous sparse event series can combine to produce a fuller picture of underlying motivations.

A. Synthetic Data

Synthetic datasets are used to first evaluate the mechanisms separating event series into Z classes (clustering) and to map new event processes to a given set of underlying processes (classification) are independently evaluated. Synthetic event series are generated from 4 families of non-stationary rate functions: *piecewise Poisson*; *ascending piecewise linear*; *descending piecewise linear*; and *sinusoidal*, as illustrated in

Figure 2. Each function was constrained so that for a fixed observation window $\tau_w = 20$ all had an identical integral $\Lambda(t) = 100$ and therefore produce a similar expected number of events. This precluded the danger that each event series could be trivially distinguished via length. Each rate function was used to generate m observations via *thinning* [11] running the process up to a cut-off time-point, τ.

Classification performance: This was tested via a process of random re-sampling. In each iteration a training set of $m = [10, 200]$ event series was simulated for each class using its whole rate function. These were used to produce an estimate of the class' generating rate function (examples of these estimates for $m = 100$ are illustrated in Figure 2). A further 100 event series were then produced for each class limited to a fixed timepoint τ) and combined to create a test set, $\mathcal{T}$. Each element in $\mathcal{T}$ was then classified via equation 12, and the average classification accuracy recorded. This process was then repeated with a different value for $\tau = [1..20]$ (and hence a different average no. of events in each series, $\bar{m}$), iterating the whole process 50 times for each value. Extremely high results were observed, as illustrated in Figure 3. First, it can be seen that once the training set had reached 25 event series performance became highly consistent (and broadly independent of increasing the model size). As one would expect as test series became longer they also became easier to classify. However accuracy improved extremely rapidly and by the time $\tau = 5$ (with an average of 25 events in each series) 90% accuracy was already being achieved across the board, increasing to 97% by $\tau = 10$.

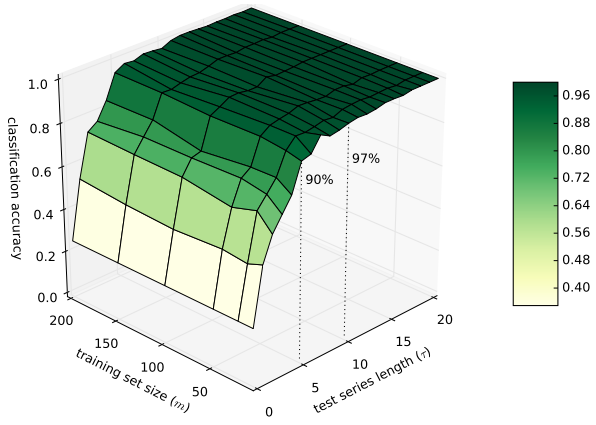

Fig. 3. Classification performance using the 4 synthetic datasets detailed in Figure 2, varying no. realizations generated from each rate function (m) and sparsity/series length as defined by the end time-point of the process (τ).

Clustering performance: Again the full rate function was used to generate a fixed number of event series, $m = [10, 200]$, for each class, combining these to form dataset, $\mathcal{T}$. This dataset was then clustered into 4 classes using the unsupervised method described in §IV-B. The purity [1] of those classes was then recorded, with the whole process being iterated 50 times in order to come up with a mean purity score for that value of m. The experiment was then repeated for different dataset sizes. The results of this analysis are illustrated in Figure 4. Results were highly encouraging, with the clusters

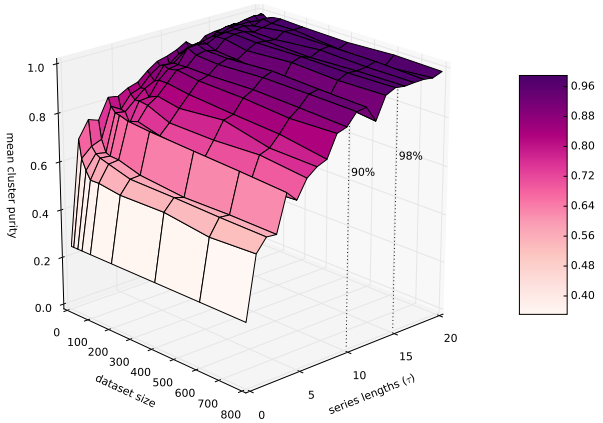

Fig. 4. Clustering performance as measured by the purity of the 4 clusters retrieved, varying dataset size and end-points (τ) of the realizations simulated in equal numbers from the four parent processes in Figure 2. 90% accuracy is achieved by the time $\tau = 10$ and 98% accuracy is achieved by $\tau = 15$.

retrieved having a 90% correspondence to the ground truth even with relatively sparse event series (at $\tau = 10$). By the time $\tau = 15$ there was 98% purity across the board, with this score continuing to increase the more of the rate function used. Any effect due to the size of the dataset being clustered is only observable at lower series length (i.e. $\tau < 10$), where increased numbers of realizations lead to more errors occurring.

Next-event prediction: Finally, we tested the predictive strength of our approach versus the other candidate predictors using a regime of random re-sampling tests[3]. In each iteration 100 event series were simulated from each of the four parent processes (using their full rate functions). Labels denoting the generating process were removed from the realizations before combining them into a training set of 400 event series. To prepare for parallel prediction a model was then created using the unsupervised clustering approach detailed in §IV-B assuming $k = 4$, with a rate function being estimated for each cluster as described in §IV-A.

For each iteration, a test set of 200 series was simulated (again containing an equal number of realizations from each parent process, but with each having a randomized final time-point, $\tau = [5, 15]$, in order to reflect the radically different lengths of event series expected in real data). Prediction accuracies were assessed by removing each test realizations final event time, t_m, and using each candidate predictor to forecast that 'next event' time given the remaining event history, $\langle t_1, t_2, \ldots, t_{m-1} \rangle$. Each predictor's mean error was then recorded, and the whole process then repeated 100 times using a different random seed.

Final results are reported Table I with box-plots illustrated in Figure 5. As can be seen, the P-NPP predictor performed extremely well, improving on all other candidates to a statistically significant level (Bonferroni adjusted t-test, $p < 0.05$). Performances of PCIM and MFPP were still surprisingly good, if inseparable from each other and the best KNTP predictor,

[3]This test procedure is designed to account for training set variance and maximise replicability, as per the prescription in [13], corroborated in [2].

IMP	GMP	P-GMP	LMP	KNTP	P-LMP	PCIM	MFPP	P-NPP
0.14107	0.13209	0.12934	0.15590	0.12491	0.13503	0.12263	0.12264	0.11669

TABLE I

MEAN ERROR RESULTS FOR EACH OF THE 9 NEXT-EVENT PREDICTORS ON SYNTHETIC DATASETS.

IMP	GMP	P-GMP	LMP	KNTP	P-LMP	PCIM	MFPP	P-NPP4
0.18082	0.15508	0.15627	0.13269	0.12492	0.13023	0.12492	0.12498	0.11663

TABLE II

MEAN ERROR RESULTS FOR EACH OF THE 9 NEXT-EVENT PREDICTORS ON A&E EMERGENCY DATASET. HERE P-NPP SCORES REFLECT A 4-PROCESS ASSUMPTION.

for which numerous neighbourhood sizes were attempted ($k = \{1, 5, 10, 15, 25, 50, 100\}$ - we report scores for the most effective, which was 25). Given the nature of the synthetic data the success of the parallel NHPP approach is somewhat to be expected, but does highlight: 1. the effectiveness of our technique when the underlying data can be assumed to have been produced by a non-stationary independent process; and 2. that parallel forecasting is preferable to both global and individual prediction when data are likely generated from multiple underlying processes.

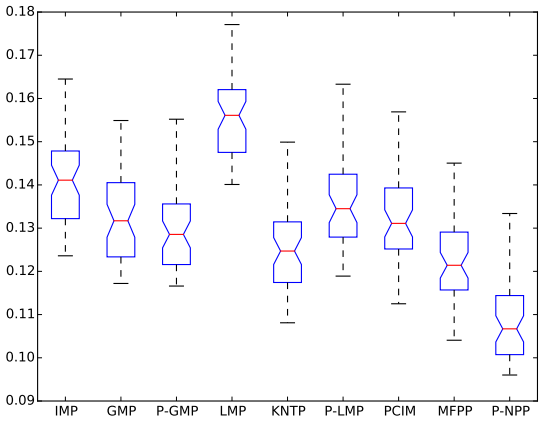

Fig. 5. Next-event prediction results, illustrating performance on synthetic data sets.

B. Accident and Emergency Data

Effectiveness of the predictors was also tested on real world datasets, for which we use two examples. The first dataset analysed is extracted from event logs of patient arrivals at Accident and Emergency (A&E) departments across the UK. Among numerous labels, each patient arrival is attributed a date, time and a diagnosis code. These allowed us to extract 1460 daily event series (all representing a 24 hour period starting at 12.00am) from the period 1st Jan 2013 to 31st Dec 2014. Each of these daily series corresponds to the arrival of patients at any of the UK's five busiest A&E departments[4] for one of 4 distinct emergency conditions - respiratory illness, lacerations, gastrointestinal conditions and sprain/ligament injuries[5].

Forecasting performance for each predictor was tested by a series of random resampling tests. In each of the 50 iterations performed 400 series were sampled with 80% of that data used for training the predictors and 20% for testing their performance. Each event series in the test set was curtailed at a random point in the day, and a forecast made using each predictor as to the time of the next event following that point. As with the synthetic datasets, KNTP was tested with neighbourhood ranges of $k = \{1, 5, 10, 15, 25, 50, 100\}$. Equally, with there being no 'ground truth' as to the correct natural number of underlying processes at work, a range of $z = [1, 15]$ was considered during cross validation for P-NPP.

Results are illustrated in the boxplot in Figure 6, and to some extent echo the performance of the predictors on the synthetic datasets. Baseline methods of IMP and GMP both perform relatively poorly, as expected. LMP however delivers much better performance than on the synthetic data, and in its KNTP form (with $k = 10$) performs almost as well as PCIM and MFPP.

P-NPP, however, again achieved strongest performance scores. Table II reports the performance when using 4 processes, which reflects an informed guess given the 4 conditions featured in the dataset. This resulted in a prediction error of 0.1166 for P-NPP4 in comparison to 0.1250 for MFPP. This represents a statistically significant improvement using a Bonferroni-corrected paired t-test ($p < 0.05$), but more importantly an improvement in the prediction of the next patients arrival by 12.1 minutes.

The rate functions recognized by the four process P-NPP predictor are visualized in Figure 7 and it was interesting to note that three of the rate functions often corresponded to specific conditions. *Rate A* predominantly reflected sprains which have a major influx at 9am and tail off throughout the day. *Rate C* connects with empirical observation of gastro-intestinal conditions which occur more frequently than other illnesses at night, but remain relatively flat throughout the day. *Rate D* also showed a strong correspondence with laceration attendances, which have a slightly lower daily incidence. However, the remaining condition, respiratory illnesses had little to no correspondence with the trend found in *Rate B*.

The suspicion therefore was that 4 was not the optimal number of processes, and variation in other factors (e.g. day of the week, month, temperature, etc.) influenced the true underlying processes. During training, cross validation of the hyper-parameter Z identified 7 as the most common no. of processes settled upon, overall reducing mean error to 0.115 (reflecting a further 2.3 minute mean improvement in prediction accuracy). Figure 8 illustrates the resulting rate

[4]QMC (Nottingham), New University College (London), Frimley Park (Surrey), Queen's (London) and Queen Elizabeth (Birmingham).

[5]This dataset is available on request from http://www.hscic.gov.uk/dars.

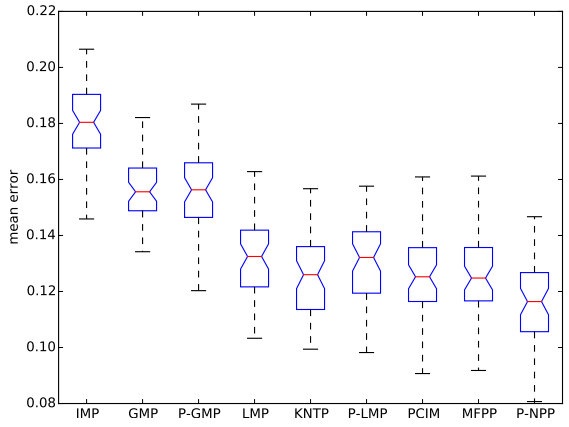

Fig. 6. Performance of each of the 9 predictors on the A&E events dataset, measured in days (n.b. P-NPP results are for 4 processes being assumed).

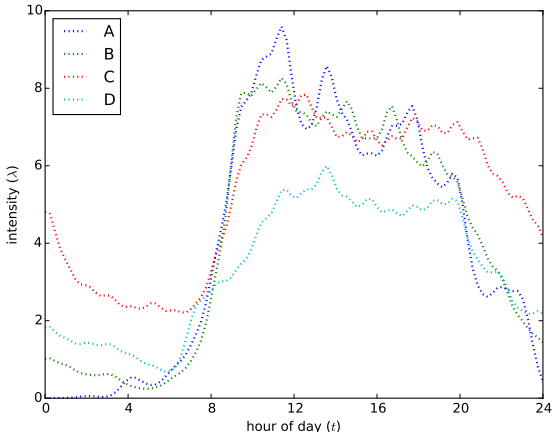

Fig. 7. Rate functions as determined and then used by the P-NPP4 predictor, reflecting the different daily temporal event patterns of different illnesses.

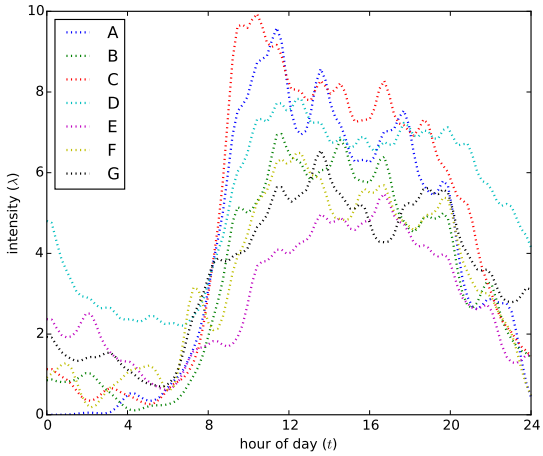

Fig. 8. Rate functions as determined by the P-NPP predictor using 7 processes.

functions when we directly set $Z = 7$ - the miscellaneous function from P-NPP4 results has now disappeared, replaced by rates that reveal spikes for injuries in the late hours of the weekends (which one might attribute to social behaviours over those periods).

C. Retail Customer Data

Our second dataset contains household level transactions over two years from a group of 2,500 households who are frequent shoppers at an anonymous retailer, collected over a 2 year period [6]. For our experiments we extracted event series covering a 3 month period for 500 anonymous households (beginning at day 100 and filtering out series with less than 5 events, to ensure all households were active over the period). Testing occurred in exactly the same manner as per §VI-B. For the retail dataset the optimal neighbourhood size for KNTP was 4, and the number of underlying processes used by the P-NPP predictor predominantly occurred when $Z = 5$ during the testing procedure.

Results were again encouraging (see Figure 9 and Table III), with P-NPP achieving improved performance over PCIM based-approaches. However, in this case its predictive accuracy was inseparable from using a KNTP predictor (both achieving a mean improvement in predicting a customer's next visit of ≈9.15 hours over using the global average GMP). An advantage of P-NPP, however, is that the underlying patterns of behaviour it reveals can also be used for descriptive purposes - these are illustrated in Figure 10 and highlight that while one group of customers are characterized by their higher intensity (*Rate C*, featuring customers who visited a mean of ≈29 times), there are distinct group behaviours for both medium frequency (*Rates D/E*, with a mean of ≈11 visits) and low frequency households (*Rates A/B*, with a mean of ≈6.5 visits).

VII. Discussion & Conclusions

Experimental results for the parallel NHPP technique derived in this paper show high performance, demonstrating the potential of the approach and suggesting that event series forecasting may well be best viewed as a distinct predictive task, rather than being conflated within time-series research. In particular, the sparsity inherent to most event series means that the sharing of cross-realization structural information may be a particularly valuable area of ongoing investigation (indicated by the strong results for both KNTP and P-NPP predictors). The real value of 'Big Data' may well lie not in the statistical significance it provides, but in the fact that we no longer need to rely on assumptions of stationarity or ergodicity to make predictions. Instead we can begin to piece together the statistical properties of underlying behavioural processes through principled recombination of mass sets of individual realizations - or 'small data' which can happily be sparse individually.

A parallel NHPP approach will work most effectively on datasets which have stemmed from distinct, separable classes of behaviour rather than a continuum (in which a case a local neighbourhood approach would be more advantageous). P-NPP proved most effective on the health data, given the distinct causes for A&E attendance and differing behaviours at weekends compared to weekdays. The higher variation of

[6]The raw data is publicly available as "The Complete Journey" dataset from https://www.dunnhumby.com/sourcefiles

IMP	GMP	P-GMP	LMP	KNTP	P-LMP	PCIM	MFPP	P-NPP4
1.91014	1.90108	1.86326	1.73295	1.52008	1.60209	1.75941	1.75712	1.52494

TABLE III
MEAN ERROR RESULTS FOR EACH OF THE 9 NEXT-EVENT PREDICTORS ON THE RETAIL DATASET.

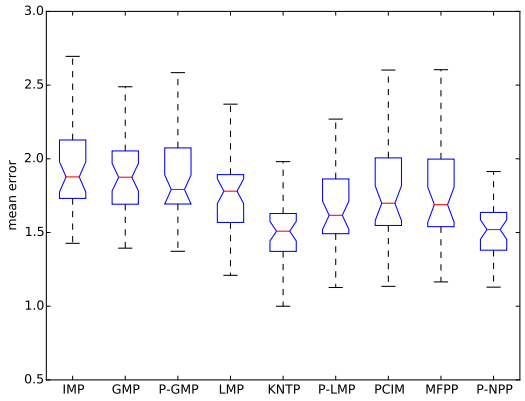

Fig. 9. Next-event prediction results, illustrating performance of each of the 9 predictors on the Retail dataset.

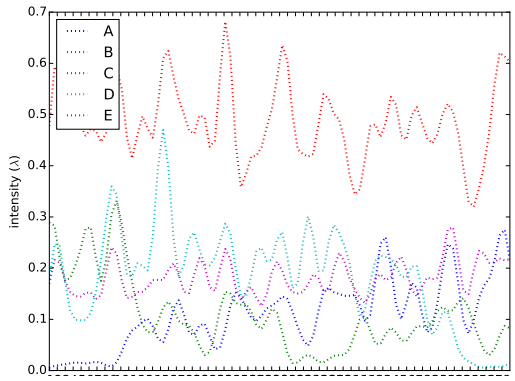

Fig. 10. Processes reflecting different household shopping habits over two months (each x-axis tick representing one day), exhibiting strong periodical behaviour.

behaviours inherent to household shopping mean that a local approach was also effective - however, P-NPP is also disadvantaged by the high-frequency cyclical behaviours exhibited. While it can cope with stark changes in intensity rates (as demonstrated by the synthetic datasets), the fact that we model rate functions using a fixed number of Gaussian bases meant the approach is more amenable to modelling smooth rate functions such as those in the health data. As mentioned in §IV-A, however, alternative bases for the rate function and/or alternate process models are possible under the framework, providing the potential for future work to provide formulations that better suit datasets with such properties.

One attractive feature of the proposed technique (that became apparent during experimental runs) is its lack of reliance on simulation - direct use of the rate function's integral provided a computational advantage over PCIM and MFPP. A further extension lies in deriving formulae not only for the next step distribution, $\mathcal{W}_1(s)$, but for further steps ahead, $\mathcal{W}_x(s)$, in the same manner as described in Appendix A. This would allow direct prediction of events further into the future, without having to turn to iterated predictions (which are thought only to be stable for short term forecasts [12]).

Limitations certainly exist - use of Gaussian bases means predictions cannot be made past the endpoint of the longest $\overrightarrow{\mathbf{t}}$ (alternative bases such as Fourier series may offer a solution here). Further, the independence assumption made by NHPP may lead the technique to be less effective in modelling bursty data streams (e.g. email generation). Again, however, the exploration of other process models within the parallel prediction framework better suited to cope with inter-event dependencies may remedy this situation. Finally the flexibility NHPPs offer may come at the cost of greater requirements on size of training dataset. While this requires more investigation, we note pressure here would be placed on the number of event-series in T and not the density of any individual event series.

In this paper we have presented a new technique for event series prediction, deriving a new estimation process for parallel NHPP mixture model based on expectation maximization, before layering upon it a novel parallel forecasting technique. The applicability and effectiveness of this technique was demonstrated on both real world and synthetic datasets. There is much scope for further development - in particular the process of aligning asynchronous series within that mixture modelling task deserves further investigation.

VIII. ACKNOWLEDGMENTS

This work was supported by grant *¡removed¿*.

APPENDIX

Given a set of event series $\mathcal{X} = \{\overrightarrow{\mathbf{x}}_1, \overrightarrow{\mathbf{x}}_2, \ldots, \overrightarrow{\mathbf{x}}_m\}$, where each realization is made up of a vector of inter-event 'gaps', $\overrightarrow{\mathbf{x}}_i = \langle x_{i1}, x_{i2}, \ldots, x_{ij}\rangle$, the first tools our model requires are density functions for event occurrences. Point processes have two key attributes that aid in this task: the number of events, $N(t_1, t_2)$, the process generates within the time interval $(t_1, t_2]$; and the waiting time, $\mathcal{W}_n(t)$, until the next n events occur, following time t. There is a direct relationship between these two random variables: if $\mathcal{W}_n(t) > x$ then at time $(t+x)$ we must still be waiting for the nth event to occur. Hence:

$$P(\mathcal{W}_n(t) > x) = P(N(t, t+x) < n)$$

[A] **Next-Event CDF:** Given an event has occurred at time s, this allows us to define the cumulative density function (cdf), $F^s(x)$, for the wait time until the next event occurs, $\mathcal{W}_1(s)$:

$$F^s(x) = 1 - P(N(s, s+x) = 0)$$

For an NHPP, at time-point s, the probability that after period, x, no more events will have occurred is given by:

$$P(N(s, s+x) = 0) = e^{-\Lambda(s,s+x)}\frac{\Lambda(s, s+x)^0}{0!} = e^{-\Lambda(s,s+x)}$$

Hence the cdf for an NHPP is:

$$F^s(x) = 1 - e^{-\Lambda(s,s+x)} \tag{15}$$

[B] **Next-Event Gap PDF:** The probability density function for time of next event given we are time-point s, can then be found by differentiating the cdf given in equation 15:

$$f^s(x) = \frac{d}{dt}\Big(F^s(x)\Big) = \frac{d}{dt}\Big(1 - P\big(N(s, s+x) = 0\big)\Big)$$

For NHPP we find this pdf by differentiating equation 2:

$$f^s(x) = \frac{d}{dx}\left(1 - e^{-\Lambda(s,s+x)}\right) = \lambda(s+x)e^{-\Lambda(s,s+x)} \tag{16}$$

[C] **Event-Times Joint PDF:** In order to find the MLE for an NHPP it is easier to use the pdf of event times, $g(t_i)$, rather than of inter-event gaps, $f^s(t)$. A natural accompanying representation to $\mathcal{X}$, is therefore to cast $\overrightarrow{\mathbf{x_i}}$, as an ordered series of time points (where each t is the sum of inter-event gaps that preceded it):

$$\overrightarrow{\mathbf{t_i}} = \langle t_{i1}, \ldots, t_{in} \rangle \quad \textit{where} \quad t_{ij} = \sum_{k \le j} x_{ik}.$$

This translates the items in $\mathcal{X}$, into their event-time equivalents, $\mathcal{T}$, with no information loss. For NHPP every inter-event gap is independent, so if we see an event at time s, then the distribution for the next event time, $g^s(t)$, is identical to the distribution of inter-event times, $f^s(x)$, where $t = s + x$:

$$g^s(t) = f^s(t - s)$$

Because NHPPs possess the Markov property, this relationship provides a way of describing the distribution of the j^{th} time point in the process. If we let $t_0 = 0$:

$$g(t_j) = f_{t_{j-1}}(t_j - t_{j-1}) = \lambda(t_j)\, e^{-\Lambda(t_{j-1}, t_j)}$$

A joint probability distribution for a set of time-points is then:

$$g(\overrightarrow{\mathbf{t}}) = g(t_1, t_2, \ldots, t_n) = \prod_{j=1}^{n} g(t_j) = e^{-\Lambda(0,t_n)} \prod_{j=1}^{n} \lambda(t_j)$$

One final step remains - if our point processes were generated by cutting off at some specified time, τ, then we also need to account for the lack of new events between t_n and τ. Now:

$$P(N(t_n, \tau) = 0) = e^{-\Lambda(t_n,\tau)}$$

Therefore, the joint pdf given this cut-off time is:

$$\begin{aligned} g(\overrightarrow{\mathbf{t}}|\tau) &= g(\overrightarrow{\mathbf{t}})\; P(N(t_n, \tau) = 0) \\ &= e^{-\Lambda(0,\tau)} \prod_{j=1}^{n} \lambda(t_j) \end{aligned} \tag{17}$$

REFERENCES

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics. *Information retrieval*, 12(4), 2009.

[2] R. R. Bouckaert and E. Frank. Evaluating the replicability of significance tests for comparing learning algorithms. In *Advances in knowledge discovery and data mining*, pages 3–12. Springer, 2004.

[3] E. Brown, R. Kass, and P. Mitra. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature neuroscience*, 7(5):456–461, 2004.

[4] R. F. Engle. The econometrics of ultra-high-frequency data. *Econometrica*, 68(1):1–22, 2000.

[5] K. Gopalratnam, H. Kautz, and D. S. Weld. Extending continuous time bayesian networks. In *Proceedings of the national conference on artificial intelligence*, volume 20, page 981, 2005.

[6] J. Grabocka, A. Nanopoulos, and L. Schmidt-Thieme. Classification of sparse time series via supervised matrix factorization. In *AAAI*, 2012.

[7] A. Gunawardana, C. Meek, and P. Xu. A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems 24*. 2011.

[8] G. R. Hjaltason and H. Samet. Index-driven similarity search in metric spaces (survey article). *ACM Transactions on Database Systems (TODS)*, 28(4):517–580, 2003.

[9] G. Jongbloed and G. Koole. Managing uncertainty in call centres using poisson mixtures. *Applied Stochastic Models in Business and Industry*, 17(4):307–318, 2001.

[10] T. A. Lasko. Efficient inference of gaussian process modulated renewal processes with application to medical event data. In *13th conference on Uncertainty in Artificial Intelligence*, pages 469–476, 2014.

[11] P. A. Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes by thinning. Technical report, DTIC Document, 1978.

[12] J. McNames. A nearest trajectory strategy for time series prediction. In *Proceedings of the International Workshop on Advanced Black-Box Techniques for Nonlinear Modeling*, pages 112–128, 1998.

[13] C. Nadeau and Y. Bengio. Inference for generalization error. *Machine Learning*, 52(3):239–281, 2003.

[14] U. Nodelman, C. Shelton, and D. Koller. Continuous time bayesian networks. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 378–387, San Fran., 2002.

[15] U. Nodelman, C. R. Shelton, and D. Koller. Learning continuous time bayesian networks. In *19th conference on Uncertainty in AI*, pages 451–458, 2002.

[16] J. R. Norris. *Markov chains*. Cambridge university press, 1998.

[17] Y. Ogata. Statistical models for earthquake occurrences. *Journal of the American Statistical Association*, 83(401):9–27, 1988.

[18] C. R. Shelton and G. Ciardo. Tutorial on structured continuous-time markov processes. *Journal of AI Research*, 51:725–778, 2014.

[19] H. Shen and J. Z. Huang. Forecasting time series of inhomogeneous poisson processes with application to call center workforce management. *The Annals of Applied Statistics*, pages 601–623, 2008.

[20] A. Simma, M. Goldszmidt, J. MacCormick, P. Barham, R. Black, R. Isaacs, and R. Mortier. Ct-nor: Representing and reasoning about events in continuous time. In *Proceedings of UAI-08*, pages 484–493, Corvallis, Oregon, 2008. AUAI Press.

[21] A. Simma and M. Jordan. Modeling events with cascades of poisson processes. In *Proceedings of UAI-10*, pages 546–555. AUAI Press, 2010.

[22] G. Smith, R. Wieser, J. Goulding, and D. Barrack. A refined limit on the predictability of human mobility. In *IEEE Pervasive Computing and Communications (PerCom)*, pages 88–94. IEEE, 2014.

[23] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[24] Y. W. Teh and V. Rao. Gaussian process modulated renewal processes. In *Advances in Neural Information Processing Systems 24*. 2011.

[25] J. Weinberg, L. D. Brown, and J. R. Stroud. Bayesian forecasting of an inhomogeneous poisson process with applications to call center data. *Journal of the American Statistical Association*, 102(480), 2007.

[26] J. Weiss and D. Page. Forest-based point process for event prediction from electronic health records. In *Machine Learning and Knowledge Discovery in Databases*, volume 8190, pages 547–562. 2013.