Bayesian Nonparametric Poisson-Process Allocation for Time-Sequence Modeling

Hongyi Ding[*] **Mohammad Emtiyaz Khan**[†] **Issei Sato**[*†] **Masashi Sugiyama**[†*]

[*]The University of Tokyo, Japan [†]The RIKEN Center for AIP, Tokyo, Japan

Abstract

Analyzing the underlying structure of multiple time-sequences provides insights into the understanding of social networks and human activities. In this work, we present the *Bayesian nonparametric Poisson process allocation* (BaNPPA), a latent-function model for time-sequences, which automatically infers the number of latent functions. We model the intensity of each sequence as an infinite mixture of latent functions, each of which is obtained using a function drawn from a Gaussian process. We show that a technical challenge for the inference of such mixture models is the unidentifiability of the weights of the latent functions. We propose to cope with the issue by regulating the volume of each latent function within a variational inference algorithm. Our algorithm is computationally efficient and scales well to large data sets. We demonstrate the usefulness of our proposed model through experiments on both synthetic and real-world data sets.

1 Introduction

The Internet age has made it possible to collect a huge amount of temporal data available in the form of time-sequences. Each time-sequence consists of timestamps which record the arrival times of events, e.g., postings of tweets on Twitter or announcements of life events on Facebook. In real-world problems arising in areas such as social science (Gao et al., 2015), health care (Lian et al., 2015) and crime prevention (Liu and Brown, 2003), time-sequence modeling is extremely useful since it can help us in predicting future events and understanding the reasons behind them.

When modeling a collection of time-sequences, a key idea is to cluster the data into groups while allowing the groups to remain linked to share statistical strengths among them (Teh et al., 2005). Several models have been proposed on the basis of this simple idea, e.g., the convolution process (Gunter et al., 2014), nonnegative matrix factorization (NMF) (Miller et al., 2014), and latent Poisson process allocation (LPPA) (Lloyd et al., 2016). These models employ latent factors to share statistical strengths and combine these functions to model the correlations within and among time-sequences.

Among these models, LPPA is a powerful approach because it uses latent functions obtained from a Gaussian process (GP). Such continuous latent functions are able to flexibly model complex structures in the data, and do not require a careful discretization such as that used in NMF. However, a limitation of LPPA is that the number of latent functions needs to be set beforehand. If the chosen number is much larger than the actual number of latent functions required to explain the data, LPPA will still use all the latent functions. There is no mechanism in LPPA to prevent this "spread" of allocation, which creates a problem when our goal is to understand the reasons behind the events observed in the data. For example, this might make it difficult to explain the retweet patterns in Twitter where a sudden avalanche of retweets is quite common (Gao et al., 2015). For such cases, LPPA will simply use all its latent functions to explain these spiky patterns.

In theory, the above problem can be solved by using Bayesian nonparametric (BNP) methods (Hjort et al., 2010) which can automatically determine the number of relevant latent functions. However, as we show in this paper, a direct application of existing BNP methods to LPPA is challenging. An obvious issue is that such an application typically requires the use of Markov Chain Monte Carlo (MCMC) algorithms

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

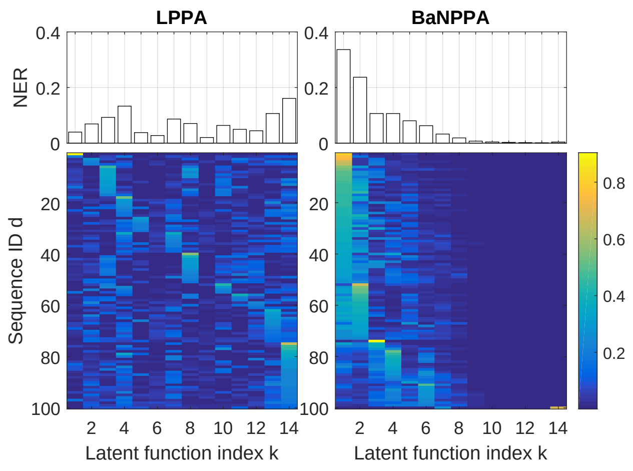

Figure 1: This figure illustrates that, even when a large number of latent functions are provided, BaNPPA automatically selects only a few to explain the data, while LPPA uses them all. The bottom plots show the weights of the latent functions for the Microblog dataset, where we see that BaNPPA assigns zero weights to many latent functions, while LPPA assigns every latent function to at least a few time-sequences. The top plots show a score which measures the average responsibility of the latent functions. See Section 5 for details.

which are slow to converge for large data sets. A more essential and technically intricate issue is that a naive application of BNP methods to LPPA suffers from an unidentifiability issue because the GP-modulated latent functions are not normalized. Unidentifiability is bad news when our focus is to understand the reasons behind the events.

In this paper, we propose a new model to solve these problems. Our model, which we call the *Bayesian nonparametric Poisson process allocation* (BaNPPA) model, enables automatic inference of the number of latent functions while retaining the accuracy, interpretability, and scalability of LPPA. Unlike hierarchical models (Teh et al., 2005) which promote sharing through a common base measure, latent functions in our model are shared across all time-sequences due to the size-biased ordering which promotes sharing by penalizing latent functions that belong to higher indices (Gopalan et al., 2014; Pitman et al., 2015). The size-biased ordering restricts the use of all latent functions. Figure 1 illustrates this on a real data set.

We propose a computationally efficient variational inference algorithm for BaNPPA and solve the unidentifiability issue by adding a constraint within the inference algorithm to regulate the volume of each latent function. Overall, we present a scalable and accurate Bayesian nonparameteric approach for time-sequence modeling. Figure 2 shows an example of the results obtained with BaNPPA on a real-world dataset.

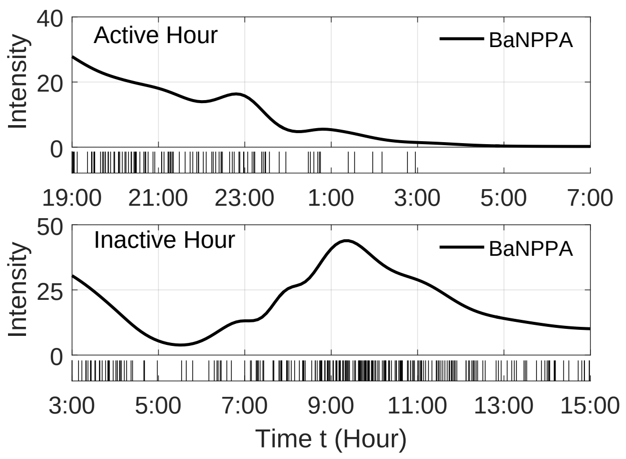

Figure 2: Illustrations of intensity functions obtained with BaNPPA on the Microblog dataset. Each plot shows a time-sequence (with small bars at the bottom) and the corresponding estimated intensity function (with solid lines). The top and bottom plots are for tweets posted during active and inactive hours of the day, respectively.

2 Time-Sequence Modeling and Its Challenges

Our goal is to develop a flexible model for time-sequences. We consider time-sequence that contain a set of time-stamps which record the occurrence of events. We denote a time-sequence by $\mathbf{y}_d = \{t_n^d \in \mathcal{T}\}_{n=1}^{N_d}$, where t_n^d is the n'th time-stamp in the d'th time-sequence, $\mathcal{T} \subset \mathbb{R}^+$ is a specified time window, and N_d is the number of events. The set of D time-sequences is denoted by Y.

A common approach to model such time-sequences is to use the temporal Cox process (Adams et al., 2009; Lloyd et al., 2015) which uses a stochastic intensity function $\lambda(t) : \mathbb{R}^+ \to \mathbb{R}^+$ to model the arrival times (Kingman, 1993). Given the intensity function $\lambda(t)$ and a time window $\mathcal{T} \subset \mathbb{R}^+$, the number of events $N(\mathcal{T})$ is Poisson distributed with rate parameter $\int_{\mathcal{T}} \lambda(s)ds$. Therefore, the likelihood of a sequence $\mathbf{y}_d$ drawn from the temporal Cox process is equal to:

$$\mathrm{P}(\mathbf{y}_d|\lambda_d) = \exp\Big(-\int_{\mathcal{T}} \lambda_d(s)ds\Big) \prod_{n=1}^{N_d} \lambda_d(t_n^d). \quad (1)$$

In LPPA, to model multiple time-sequences, the d'th time-sequence is assumed to be generated by a temporal Cox process with an intensity function $\lambda_d(t)$ which is modeled as follows:

$$\lambda_d(t) = \sum_{k=1}^{K} \theta_{dk} f_k^2(t), \quad \theta_{dk} \geq 0, \tag{2}$$

where $f_k(t)$ is a function drawn from a GP prior, θ_{dk} is its weight, and K is the number of latent functions. To ensure the non-negativity of λ_d, f_k are squared and weights θ_{dk} are required to be non-negative.

LPPA is a powerful approach which also enables scalable inference. Due to the GP prior, LPPA is capable of generating intensity functions with complex shapes. Scalable inference is made possible by using variational inference for sparse GPs (Titsias, 2009). The overall computational complexity is $O(KNM^2)$, where N is the total number of events in Y and M is the number of pseudo inputs in sparse GPs.

One issue with LPPA is that K needs to be set beforehand. This not only increases the computation cost, but also creates a serious interpretability issue which is undesirable when our goal is to understand the reasons behind the data. Specifically, when the number of latent functions is much larger than what it needs to be, LPPA uses all of them, making it difficult to interpret the results. We give empirical evidence in support of this claim and correct this behavior by using a BNP method.

Unfortunately, a direct application of the existing BNP methods increases the computation cost and limits the flexibility of the model. The problem lies in the strict requirement that the latent functions needs to be a *normalized density function*, i.e., a function with a volume[1] equal to 1. For example, previous studies, such as Kottas (2006); Ihler and Smyth (2007), model the intensity functions with the following Dirichlet process mixture model,

$$\lambda_d(t) = s_d \sum_{k=1}^{\infty} \theta_{dk} \tilde{f}(t; \psi_k), \tag{3}$$

where $\tilde{f}$ are normalized density functions with parameters ψ_k and the weights θ_{dk} are non-negative and sum to one $\sum_{k=1}^{\infty} \theta_{dk} = 1$ ($s_d > 0$ is the rate parameter that models the number of events $N(\mathcal{T})$). Since the function $\tilde{f}$ needs to be normalized, the choices are limited to well-known density function which may not be very flexible to model complex time-sequences, e.g., Kottas (2006) used the beta distribution and Ihler and Smyth (2007) used the truncated Gaussian distribution. In addition, such models require MCMC sampling algorithms which usually converge slowly on large data sets. To the best of our knowledge, it is still unclear how to build a nonparametric prior for such normalized density functions while enabling scalable inference, e.g., via variational methods.

[1]The volume of a function $f(t), t \in \mathcal{T}$ is defined as the integral $\int_{\mathcal{T}} f(t)dt$.

We propose a nonparameteric model, called the Bayesian nonparameteric Poisson process allocation (BaNPPA), which avoids the need to explicitly specify the number of latent functions while retaining the flexibility and scalability of the LPPA model. Our method combines the models shown in Equation (2) and (3). We show that this direct combination has an unidentifiability issue, and we fix the issue within a variational-inference algorithm. Our approach therefore combines the strengths of the LPPA and BNP models while keeping their best features.

3 Bayesian Nonparametric Poisson Process Allocation (BaNPPA)

As discussed earlier, we need to set the number of latent functions beforehand for LPPA. We fix this issue by proposing a new model called BaNPPA that combines the non-parametric model of Equation (3) with the LPPA model shown in Equation (2). Specifically, we let $\tilde{f}$ in Equation (3) to be equal to $f_k^2(t)$, as follows:

$$\lambda_d(t) = s_d \sum_{k=1}^{\infty} \theta_{dk} f_k^2(t), \text{ where } s_d, \theta_{dk} > 0, \ \sum_{k=1}^{\infty} \theta_{dk} = 1. \tag{4}$$

Similar to LPPA, we draw functions $f_k(t)$ from a Gaussian process. We draw the weights θ_{dk} using a stick-breaking process, and use a Gamma prior for the scalar rate parameter s_d. The final generative model of BaNPPA is shown below:

1. Draw $f_k \sim \text{GP}(m_k(t), \kappa_k(t, t'))$ for $k = 1, \ldots, \infty$.
2. For each sequence $d = 1, \ldots, D$,
   - Draw $\theta'_{dk} \sim \text{Beta}(1, \alpha)$ for $k = 1, \ldots, \infty$.
   - Calculate $\theta_{dk} = \theta'_{dk} \prod_{l=1}^{k-1}(1 - \theta'_{dl})$.
   - Draw $s_d \sim \text{Gamma}(a_0, b_0)$.
   - Draw the points $\mathbf{y}_d \sim \text{PP}(s_d \sum_{k=1}^{\infty} \theta_{dk} f_k^2(t))$.

In the model, we denote a Poisson process with rate parameter λ by $\text{PP}(\lambda)$, a beta distribution with shape parameters a and b by $\text{Beta}(a, b)$ and a gamma distribution with shape parameter a and rate parameter b by $\text{Gamma}(a, b)$.

The above model automatically determines the number of latent functions due to the size-biased ordering (Pitman et al., 2015) obtained by using the stick-breaking process. Both the latent functions $\{f_k^2(t)\}$

and the weights $\{\theta_{dk}\}$ use the same set of indices $k = 1, \ldots, \infty$. This implies that when generating the d'th time-sequence, the latent function at a lower index k is more likely to be assigned a larger weight θ_{dk}. This encourages the model to use some latent functions more than the others.

Unfortunately, the above model is unidentifiable. This is because, unlike the nonparametric model of Equation (3), the latent functions $\{f_k^2\}$ are unnormalized, and therefore many combinations of s_d, $\{\theta_{dk}\}$ and $\{f_k\}$ might give us the same model. For example, the following transformation gives the same intensity function for any $\epsilon_k > 0$:

$$s_d \bar{\epsilon}_d, \left\{ \frac{\theta_{dk}\epsilon_k}{\bar{\epsilon}_d} \right\}, \left\{ \frac{f_k}{\sqrt{\epsilon_k}} \right\}, \tag{5}$$

where $\bar{\epsilon}_d := \sum_{v=1}^{\infty} \theta_{dv}\epsilon_v$. We can check this by substituting the triplet in Equation (4). Since the volume of each f_k is not regulated, we can move the "mass" around between the components of the model.

This type of unidentifiability is problematic when our goal is to understand the reasons behind the patterns in the data. In our experiments, we observe that this leads to a shrinkage of the latent functions which affects interpretability as well as the quality of the estimated hyperparameters. In Section 4.2, we propose a way to fix this issue by adding a constraint on the volume of the latent function.

There is also another common identifiability problem in such mixture models. Lloyd et al. (2016) claimed that LPPA is unidentifiable and non-unique since there may be multiple decompositions that are well supported by the data. In BaNPPA, due to the ordering constraints imposed by size-biased ordering, this unidentifiability issue is reduced.

We also need to guarantee that the expected intensity function at any time $\mathbb{E}[\lambda_d(t)]$ is finite. This can be achieved by fixing the GP hyperparameters. For example, assuming a constant mean function $m_k(t) \equiv g$ with g being the constant, and an automatic relevance determination (ARD) covariance functions $\kappa_k(t, t') = \gamma_k \exp(-(t - t')^2/(2a_k^2))$, we can fix the hyperparameters g and γ_k, which ensures that the mean and variance of each latent function f_k are finite. In that case, the value of $\mathbb{E}[\lambda_d(t)]$ is bounded due to the following relation:

$$\begin{aligned}\mathbb{E}[\lambda_d(t)] &= \mathbb{E}\left[s_d \sum_{k=1}^{\infty} \theta_{dk} f_k^2(t) \right] \le \mathbb{E}[s_d] \max_k \mathbb{E}[f_k^2(t)] \\ &= \frac{a_0}{b_0} \max_k \Big(\mathbb{E}^2[f_k(t)] + \mathrm{Var}[f_k(t)] \Big).\end{aligned} \tag{6}$$

4 Inference

In this section, we first describe the general variational inference framework and provide a solution to the identifiability issue in Section 4.2. A derivation of the evidence lower bound (ELBO) and its derivatives are provided in the Appendix A.

4.1 Variational Inference

Denote $\boldsymbol{s} \triangleq \{s_d\}$, $\Theta \triangleq \{\theta_{dk}\}$ and $\boldsymbol{f} \triangleq \{f_k\}$. Let $\boldsymbol{H}$ be the set of hyperparameters of the GP covariance function. The joint distribution of BaNPPA can be expressed as

$$\begin{aligned}p(Y, \boldsymbol{\Theta}, \boldsymbol{s}, \boldsymbol{f}) = \prod_{d=1}^{D} p(\boldsymbol{y}_d | \boldsymbol{f}, \boldsymbol{\theta_d}, s_d) \prod_{d=1}^{D} \prod_{k=1}^{\infty} p(\theta_{dk}; \alpha) \\ \times \prod_{d=1}^{D} p(s_d; a_0, b_0) \prod_{k=1}^{\infty} p(f_k; g, \boldsymbol{H}).\end{aligned}$$

We approximate the posterior distribution over Θ and $\boldsymbol{f}$, while computing a point estimate of $\boldsymbol{s}$. We follow Blei et al. (2006) to truncate the number of latent functions to K which we select to be larger than the expected number of latent functions used by the data. For the GP part, we use the same set of pseudo inputs $\{t_m\}_{m=1}^{M}$, $M < N$ for each f_k to reduce the number of variational parameters (Lloyd et al., 2016). Denote $\boldsymbol{f}_{k,M}$ to be the vector $[f_k(t_1), \ldots, f_k(t_M)]^\top$, $\kappa_{k,MM}$ to be a covariance matrix whose i, j'th entry is equal to $\kappa_k(t_i, t_j)$, and $\boldsymbol{g}_M \in \mathbb{R}^M$ to be a vector all of whose elements are equal to g. We choose the following forms for the variational distributions of θ_{dk} and $\boldsymbol{f}_{k,M}$:

$$\begin{aligned}q(\theta_{dk}) &= \mathbb{I}(k < K)\mathrm{Gamma}(\tau_{dk,0}, \tau_{dk,1}) \\ &\quad + \mathbb{I}(k = K)\delta(1) + \mathbb{I}(k > K)p(\theta_{dk}), \\ q(\boldsymbol{f}_{k,M}) &= \mathbb{I}(k \le K)\mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k) \\ &\quad + \mathbb{I}(k > K)\mathcal{N}(\boldsymbol{g}_M, \kappa_{k,MM}),\end{aligned}$$

where $\mathbb{I}(\cdot)$ is the indicator function, $\delta(\cdot)$ is a dirac-delta function, and $\boldsymbol{\mu}_k$ and Σ_k are the mean and covariance of a Gaussian distribution. Following Lian et al. (2015), we use the re-parametrization $\Sigma_k = L_k L_k^T$ by Cholesky decomposition.

Using the approximation of Titsias (2009) and a mean-field assumption over Θ, we can use the following final variational distribution:

$$q(\boldsymbol{f}, \Theta) \triangleq \prod_{k=1}^{\infty} p(\boldsymbol{f}_{k,N} | \boldsymbol{f}_{k,M}) q(\boldsymbol{f}_{k,M}) \prod_{d=1}^{D} \prod_{k=1}^{\infty} q(\theta_{dk}).$$

Denoting $\boldsymbol{\tau} \triangleq \{(\tau_{dk,0}, \tau_{dk,1})\}$ and $\boldsymbol{L} \triangleq \{L_k\}$, we get the following set of variational parameters and hyperparameters to be optimized: $\Phi = \{\boldsymbol{\tau}, \boldsymbol{\mu}, \boldsymbol{L}, \boldsymbol{H}, a_0, b_0, \alpha, \boldsymbol{s}\}$.

4.2 An Alleviation Solution to the Identifiability Problem

So far, the framework seems very traditional. However, as we mentioned in Section 3, this model has an additional identifiability problem which might make interpretability difficult. In this section, we propose a solution to alleviate this issue.

A straightforward option is to directly impose a constraint on the volume of the latent functions $\int_{\mathcal{T}} f_k^2(t)dt$, where f_k is drawn from the posterior process $p(f_k|Y)$. However this is intractable. In order to obtain a tractable constraint, we could instead impose a constraint on the following expectation:

$$\int\int_{\mathcal{T}} p(f_k|Y)f_k^2(s)dsdf_k = A, \ k = 1, \ldots, K, \qquad (7)$$

where A is a positive constant. Within the variational inference framework, we use the variational distribution $q(f_k)$ to approximate the posterior $p(f_k|Y)$, and add the following constraint to each latent function:

$$\int\int_{\mathcal{T}} q(f_k)f_k^2(s)dsdf_k = A, \quad k = 1, \ldots, K. \qquad (8)$$

The above constraint can be easily computed unlike the volume constraint on the function f_k. In our experiments, we set $A = N/D$ where N is the total number of events in the data and D is the number of time-sequences in Y.

4.3 Optimization with Equality Constraints

Given the equality constraints in Equation (8), the optimization process can be formulated as follows, where we denote the ELBO as $\mathcal{L}_1(\Phi)$:

$$\max_{\Phi} \ \mathcal{L}_1(\Phi) \quad \text{s.t. } h_k(\Phi) = 0, \ k = 1, \ldots, K, \qquad (9)$$
$$h_k(\Phi) = \int_{\mathcal{T}} \mathbb{E}_q[f_k^2(s)]ds - A.$$

Problem (9) is an optimization problem with equality constraints and we use the augmented Lagrangian method (Bertsekas, 2014) to transform Problem (9) into a series of related optimization problems indexed by i:

$$\max_{\Phi} \ \mathcal{L}_1(\Phi) - \sum_{k=1}^{K} \Big(w_{ik}h_k(\Phi) + \frac{1}{2} v_{ik} h_k^2(\Phi) \Big), \qquad (10)$$

where $\{w_{ik}\}$ is a bounded sequence and $\{v_{ik}\}$ is a non-negative monotonically-increasing sequence with respect to i. We denote this objective $L_{\boldsymbol{v_i}}(\Phi, \boldsymbol{w_i})$. For each optimization problem in Equation (10), $L_{\boldsymbol{v_i}}(\Phi, \boldsymbol{w_i})$ is still upper bounded (a proof is given in Appendix A). Thus if we use coordinate ascent with respect to Φ, the algorithm is guaranteed to arrive at a local maximum.

Table 1: Data sets used for the experiments. Here, D is the number of time-sequences, N_{train} and N_{test} are the total number of events in the training and test set respectively, and $\mathcal{T}$ is the time window.

Data set	D	N_{train}	N_{test}	$\mathcal{T}$
Synthetic A	200	6,304	6,010	[0,60]
Synthetic B	250	8,074	8,110	[0,80]
Microblog	500	44,628	44,352	[3,15]
Citation	600	106,113	106,340	[0,20]

To set $\boldsymbol{v}_{ik}$ and $\boldsymbol{w}_{ik}$, we follow the suggestions from Bertsekas (2014), and set $\boldsymbol{v}_{i+1,k} = 4\boldsymbol{v}_{ik}$ and $\boldsymbol{w}_{i+1,k} = \boldsymbol{w}_{ik} + \boldsymbol{v}_{ik}\boldsymbol{h}_k(\Phi_i)$. We initialize $v_{1k} = 4, w_{1k} = 1, \forall k$.

4.4 Computational Complexity

Optimization problems shown in Equation (10) are not significantly more expensive than the original optimization problem. Although in Equation (10), we have to optimize additional parameters, the bottleneck is still the matrix-matrix multiplication in the evaluation of $q(\boldsymbol{f}_{k,N})$. For one iteration of the training procedure, the computational complexity is $\mathcal{O}(KNM^2)$, which is the same as LPPA.

One might expect that the total computational complexity of our algorithm is worse than LPPA because we have to solve a sequence of problems. We find that "warm starts" are very effective in improving the convergence of our algorithm (Bertsekas, 2014). Namely, we reuse the final value Φ_{i-1} of the previous optimization as the starting value for the i'th round and terminate the training process when the relative change in the likelihood is small. In our experiments, we observed that the convergence of BaNPPA is rather fast and comparable to LPPA.

5 Experiments

In this section, we evaluate our proposed BaNPPA model and compare it with LPPA. To measure the effect of adding the constraint shown in Equation (8), we also compare to a variant of BaNPPA which does not contain any constraints. We call it BaNPPA with No Constraints, i.e., BaNPPA-NC. This gives us three methods to compare: LPPA, BaNPPA-NC, and BaNPPA. The code to reproduce our experiments can be found at `github.com/Dinghy/BaNPPA`.

We test the three methods on two synthetic and two

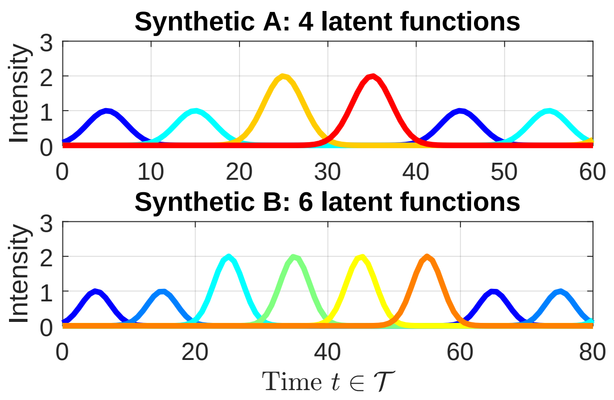

Figure 3: Latent functions used to create synthetic data set A and B are shown in the top and bottom plots, respectively. In both the data sets, there are two latent functions with two modes while the rest have only one mode.

real-world data sets. Table 1 summarizes the overall statistics and we give detailed information below.

1) Synthetic A. We sample 200 time-sequences from a mixture of 4 latent functions $\bar{f}(t)$ shown in the top plot of Figure 3. The intensity function is defined as follows: $\lambda_d(t) = s_d \sum_{k=1}^{4} \theta_{dk} \bar{f}(t)$, where $s_d \sim$ Gamma$(2, 3)$ and $\boldsymbol{\theta}_d \sim$ Dir$(1.2, 1, 0.8, 0.6)$. Here Dir$(\cdot)$ denotes the Dirichlet distribution. More details on the data generation process can be found in Appendix C.

2) Synthetic B. This data set is similar to Synthetic A but there are 6 latent functions shown in the bottom plot of Figure 3. In Synthetic B data set, $s_d \sim$ Gamma$(2, 3)$ and $\boldsymbol{\theta}_d \sim$ Dir$(1.2, 1, 0.8, 0.6, 0.5, 0.5)$.

3) Microblog data set. This data set contains 500 tweets and all retweets of each tweet from 7 tweet-posters on Sina micro-blog platform obtained through the official API[2]. Two examples are shown in Figure 2. Through time-sequence modeling, we can try to understand the retweet patterns. For example, one reason could be that the tweets posted at an inactive hour (late at night) will regain the attention from the followers several hours later next morning (Ding and Wu, 2015; Gao et al., 2015). BaNPPA could help us understand such reasons as illustrated in Figure 2.

4) Citation data set. This data set contains the Microsoft academic graph until February 5th, 2016 obtained from the KDDcup 2016 [3]. The original data set contains 126,909,021 papers and we use a subset of it. Time-sequence modeling can be used to understand the patterns of citations, e.g., some papers quickly get citations while some others get it slowly. Two examples of this data set are given in the Appendix C.1.

Evaluation Metrics. We evaluate the methods using the two metrics described below.

To measure the predictive performance, we follow Lloyd et al. (2015) and use the following approximation to the test likelihood which we denote by $\mathcal{L}_{\text{test}}(Y_{test}, \Theta, \boldsymbol{s}) =$

$$\sum_{d=1}^{D} \sum_{n=1}^{N_{\text{test}}^{d}} \ln \left(s_d \sum_{k=1}^{K} \theta_{dk} \exp \left(\mathbb{E}_q (\ln f_k^2(t_n^d)) \right) \right) - \sum_{d=1}^{D} s_d \sum_{k=1}^{K} \theta_{dk} \int_{\mathcal{T}} \mathbb{E}_q [f_k^2(s)] ds. \tag{11}$$

This is a lower bound to the test likelihood $\ln p(Y_{\text{test}}|Y_{\text{train}})$ and a higher value means a better approximation of the test likelihood. For LPPA, the allocation parameters θ_{dk} are the point-estimated weights and the rate parameter $s_d = 1$. For BaNPPA and BaNPPA-NC, we report averaged value over $q(\theta_d)$. We can compute a similar approximation $\mathcal{L}_{\text{train}}$ on the training data. Detailed derivations and explanations are given in Appendix B.

To measure the responsibility of each latent function in the model, we first define the normalized allocation matrix $\hat{\Theta} \in \mathbb{R}_{+}^{D \times K}$ whose (d, k)'th entry is equal to,

$$\hat{\theta}_{dk} = \frac{\mathbb{E}_q [\theta_{dk} \int_{\mathcal{T}} f_k^2(s) ds]}{\sum_{m=1}^{K} \mathbb{E}_q [\theta_{dm} \int_{\mathcal{T}} f_m^2(s) ds]}. \tag{12}$$

The normalization in the above matrix tries to remove the unidentifiability introduced due to the unconstrained volume of the latent functions in LPPA and BaNPPA-NC. Based on $\hat{\Theta}$, we can compute a normalized score that can measure the responsibility of each latent function. We define the normalized expected responsibility (NER) $\hat{\upsilon}_k = \sum_{d=1}^{D} \hat{\theta}_{dk}/D,\ k = 1, \dots, K$. A larger NER indicates that the corresponding latent function is more often occupied by the model. Another measure is the unnormalized expected responsibility (UNER) $\upsilon_k = \sum_{d=1}^{D} \mathbb{E}_q[\theta_{dk}]/D,\ k = 1, \dots, K$, which omits the contribution of the volume of f_k^2.

Experimental settings. Our goal is to measure the improvements obtained with the automatic inference of K using BaNPPA. To do so, we fix K to 14 for BaNPPA and BaNPPA-NC, and compare them to LPPA with a range of values for K. We expect BaNPPA to give a comparable performance to the best setting of K in LPPA.

Choice of K also affects hyperparameter estimation. To measure it, we conduct experiments for two different methods of setting the hyper-parameter α. In the first method, we learn α within a variational framework (initialize $\alpha = 1$, see details in Appendix A). In

[2] `http://open.weibo.com/wiki/Oauth/en`

[3] `https://kddcup2016.azurewebsites.net/`

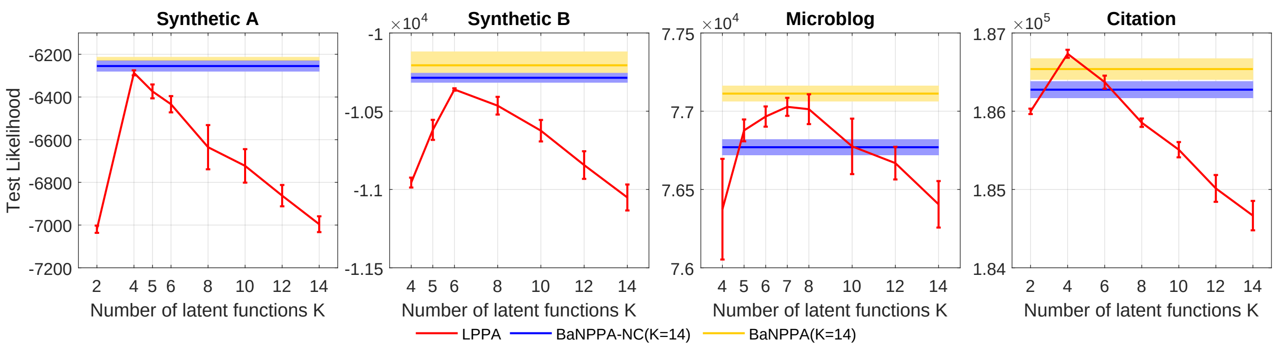

Figure 4: BaNPPA gives the best test-likelihoods (higher is better) and performs comparably to the best setting of K for LPPA. For BaNPPA/BaNPPA-NC, we use a fixed value of $K = 14$. Error bars and shaded areas show the 95% confidence intervals.

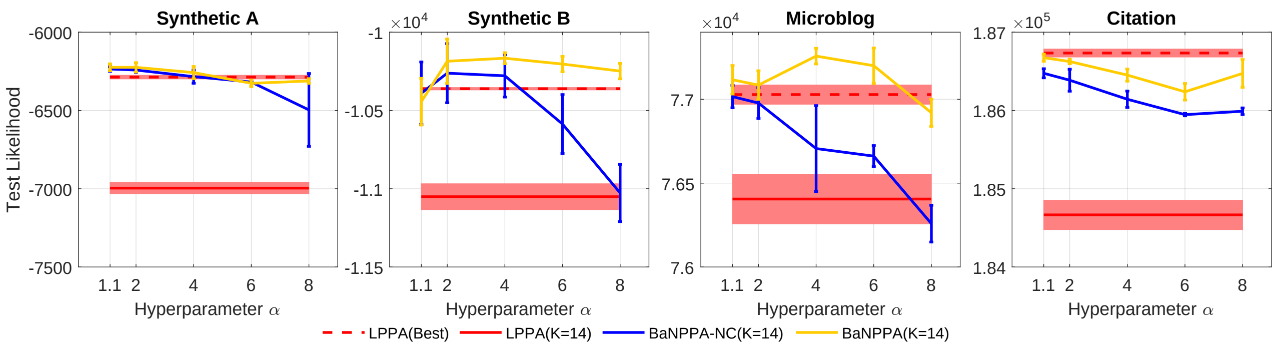

Figure 5: For a variety of hyperparameter values, BaNPPA gives the best performance which is also comparable to the best performance of LPPA and much better than LPPA with $K = 14$. Performance of BaNPPA-NC degrades with increasing α while performance of BaNPPA is relatively stable.

the second method, we do not learn α rather fix it to one of the value in the set $\{1.1, 2, 4, 6, 8\}$. In both methods, all experiments were repeated five times. We use a random initialization for the allocation matrix Θ and $\boldsymbol{\tau}$. For sparse GPs, we use 18, 24, 30 and 30 pseudo inputs for the four data sets, respectively. We follow the common practice and add a jitter term εI to the covariance matrix $\kappa_{k,MM}$ to avoid numerical instability (Bauer et al., 2016). For hyper-parameters a_0 and b_0 in the gamma distribution, we use the counts of events $\{N^d_{\text{train}}\}_{d=1}^D$ to initialize (a_0, b_0).

To maintain the positivity constraints on $\boldsymbol{L}$ and $\boldsymbol{\tau}$, we use the limited-memory projected quasi-Newton algorithm (Schmidt et al., 2009). For BaNPPA, we stop the training process when the relative change between $L_{\boldsymbol{v_i}}(\Phi_i, \boldsymbol{w_i})$ and $L_{\boldsymbol{v_{i+1}}}(\Phi_{i+1}, \boldsymbol{w_{i+1}})$ is less than 10^{-3}. For other methods, we terminate the training process when the relative change in ELBO is less than 10^{-3}.

Performance Evaluation. Figure 4 shows a comparison of the test-likelihoods when optimizing α, while Figure 5 shows the same when α is fixed. In Figure 4, the test likelihood of LPPA drops when increasing the number of latent functions K. As desired, BaNPPA achieves comparable results to the best setting of K in LPPA. BaNPPA-NC also performs well but slightly worse than BaNPPA. In Figure 5, when increasing α, the performance of BaNPPA stays relatively stable and comparable to the best setting of LPPA. The performance of BaNPPA-NC however degrades with increasing α for all data sets. This shows that the volume constraint in BaNPPA improves the performance. Other performance measures such as the training likelihood and computation time as well as the value of optimized α are given in Appendix C.

For the Synthetic A data set, we further plot the NER scores (averaged over the five trials for $K = 14$) in the top plot in Figure 6. We see that, under LPPA, all latent functions have nonzero NER, while for BaNPPA only a small number of latent functions have high NER score. In the bottom plot in Figure 6, we show the top four latent functions sorted according to the NER scores. For these plots, we used the best runs shown in Figure 4. We see that LPPA does not recover the true latent functions, while BaNPPA gives very similar results to the truth.

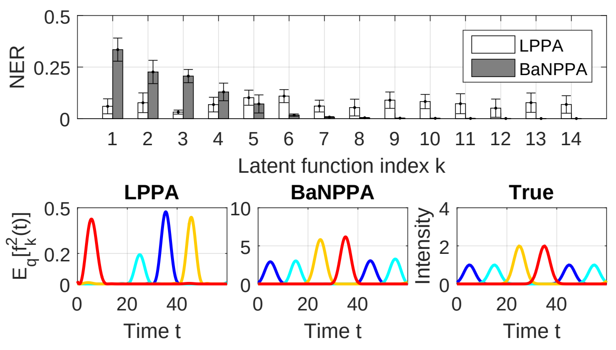

Figure 6: This figure shows that BaNPPA can reliably identify true latent functions for the Synthetic A data set. The top plot shows the NER scores for BaNPPA and LPPA for $K = 14$ where we see that, under LPPA, all latent functions have nonzero NER, while, under BaNPPA, only a handful of them have significant NER scores. The bottom plot shows the top four latent functions (sorted according to NER) obtained for both the methods along with the true latent functions. We see that BaNPPA recovers functions very similar to the true functions.

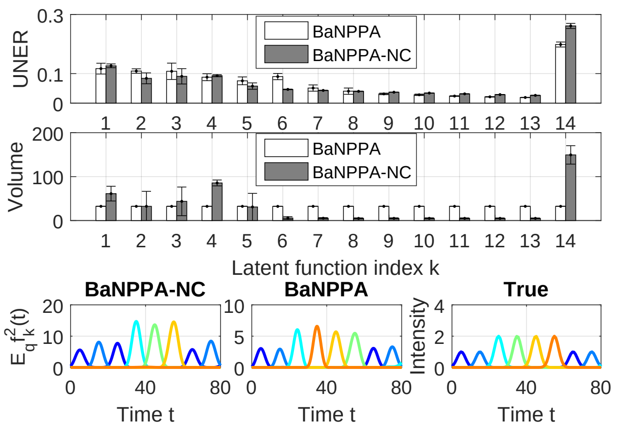

Figure 7: This figures shows that the volume constraint in BaNPPA is crucial to discover the true latent functions. Both BaNPPA and BaNPPA-NC obtain similar UNER score (top plot), yet the top latent functions obtained with the two methods are different (bottom plot). The imbalance in the volumes for BaNPPA-NC (middle plot) is the reason behind this difference. See the text for details.

To visualize the responsibilities further, we plot the NER score and the normalized allocation matrix $\hat{\Theta}$ for the Mircoblog dataset in Figure 1. We show results for LPPA and BaNPPA. We choose runs that obtained the best test-likelihood in Figure 4, and visualize 100 time-sequences sampled randomly. We see that as expected LPPA uses all latent functions to explain the data, while BaNPPA assigns almost zero weights to latent functions with higher indices. This further confirms that, even when a large number of latent functions are given, BaNPPA automatically selects only a few to explain the data, while LPPA might overfit.

Finally, we further explore the impact of the volume constraint of Equation (8) in BaNPPA. We compare BaNPPA and BaNPPA-NC on the Synthetic B data set in Figure 7. We use results for $K = 14$ and $\alpha = 8$. In the top plot, we see that BaNPPA and BaNPPA-NC both give similar UNER scores, yet as shown in the bottom plot, BaNPPA-NC does not recover the true latent functions. This result can be explained by looking at the expected volume $\mathbb{E}_q[\int_{\mathcal{T}} f_k^2(t)dt]$ shown in the middle plot. For BaNPPA, the volumes of all latent functions are equal, while, for BaNPPA-NC, the latent functions with higher UNER scores are assigned higher volume which eventually also get higher weights. This imbalance in the weights for some functions makes the results of BaNPPA-NC and BaNPPA different from each other. This result clearly shows that the volume constraint in BaNPPA plays an important role to recover the true latent functions which is important for interpretability.

Overall, BaNPPA-NC performs similarly to BaNPPA when the latent structure is simple but becomes less favorable when the structure gets complicated. We give three additional synthetic data experiments in Appendix C where the true K is large.

6 Conclusions and Future Work

We proposed a model for time-sequence data, called BaNPPA, to automatically infer the number of latent functions. We combined BNP methods with the existing LPPA method, and showed that this combination might result in undentifiability. We solve this problem by imposing a volume constraint within variational inference. In the future, we will consider further investigating the reasons behind the identifiability problem. We will also investigate ways to combine the volume constraint and the Gaussian process prior.

Acknowledgements

The authors would like to thank Wenzhao Lian for sharing the code and thank Chris Lloyd for helpful discussions. MS acknowledges support by KAKENHI 17H00757.

References

Adams, R. P., Murray, I., and MacKay, D. J. (2009). Tractable nonparametric bayesian inference in poisson processes with gaussian process intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 9–16. ACM.

Bauer, M., van der Wilk, M., and Rasmussen, C. E. (2016). Understanding probabilistic sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, pages 1533–1541.

Bertsekas, D. P. (2014). *Constrained optimization and Lagrange multiplier methods*. Academic press.

Blei, D. M., Jordan, M. I., et al. (2006). Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143.

Ding, H. and Wu, J. (2015). Predicting retweet scale using log-normal distribution. In *Multimedia Big Data (BigMM), 2015 IEEE International Conference on*, pages 56–63. IEEE.

Gao, S., Ma, J., and Chen, Z. (2015). Modeling and predicting retweeting dynamics on microblogging platforms. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 107–116. ACM.

Gopalan, P., Ruiz, F. J., Ranganath, R., and Blei, D. (2014). Bayesian nonparametric poisson factorization for recommendation systems. In *Artificial Intelligence and Statistics*, pages 275–283.

Gunter, T., Lloyd, C., Osborne, M. A., and Roberts, S. J. (2014). Efficient bayesian nonparametric modelling of structured point processes. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 310–319. AUAI Press.

Hjort, N. L., Holmes, C., Müller, P., and Walker, S. G. (2010). *Bayesian nonparametrics*, volume 28. Cambridge University Press.

Ihler, A. T. and Smyth, P. (2007). Learning time-intensity profiles of human activity using nonparametric bayesian models. In *Advances in Neural Information Processing Systems*, pages 625–632.

Kingman, J. F. C. (1993). *Poisson processes*. Wiley Online Library.

Kottas, A. (2006). Dirichlet process mixtures of beta distributions, with applications to density and intensity estimation. In *Workshop on Learning with Nonparametric Bayesian Methods, 23rd International Conference on Machine Learning (ICML)*.

Lian, W., Henao, R., Rao, V., Lucas, J., and Carin, L. (2015). A multitask point process predictive model. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2030–2038.

Liu, H. and Brown, D. E. (2003). Criminal incident prediction using a point-pattern-based density model. *International journal of forecasting*, 19(4):603–622.

Lloyd, C., Gunter, T., Osborne, M., and Roberts, S. (2015). Variational inference for gaussian process modulated poisson processes. In *International Conference on Machine Learning*, pages 1814–1822.

Lloyd, C., Gunter, T., Osborne, M., Roberts, S., and Nickson, T. (2016). Latent point process allocation. In *Artificial Intelligence and Statistics*, pages 389–397.

Miller, A., Bornn, L., Adams, R., and Goldsberry, K. (2014). Factorized point process intensities: A spatial analysis of professional basketball. In *International Conference on Machine Learning*, pages 235–243.

Pitman, J., Tran, N. M., et al. (2015). Size-biased permutation of a finite sequence with independent and identically distributed terms. *Bernoulli*, 21(4):2484–2512.

Schmidt, M., Berg, E., Friedlander, M., and Murphy, K. (2009). Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In *Artificial Intelligence and Statistics*, pages 456–463.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2005). Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in neural information processing systems*, pages 1385–1392.

Titsias, M. K. (2009). Variational learning of inducing variables in sparse gaussian processes. In *AISTATS*, volume 5, pages 567–574.