Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aleksandar Dimitriev

Modeliranje multivariatnih diskretnih podatkov z latentnimi Gaussovimi procesi

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Erik Štrumbelj

Ljubljana, 2017

University of Ljubljana
Faculty of Computer and Information Science

Aleksandar Dimitriev

Modelling multivariate discrete data with latent Gaussian processes

masters thesis

the 2nd cycle masters study programme
computer and information science

Supervisor: doc. dr. Erik Štrumbelj

Ljubljana, 2017

Declaration of Masters Thesis authorship

I, the undersigned Aleksandar Dimitriev am the author of the Master Thesis entitled:

Modelling multivariate discrete data with latent Gaussian processes

With my signature, I declare that:

- the submitted Thesis is my own unaided work under the supervision of doc. dr. Erik Štrumbelj
- all electronic forms of the Masters Thesis, title (Slovenian, English), abstract (Slovenian, English) and keywords (Slovenian, English) are identical to the printed form of the Masters Thesis,
- I agree with the publication of the electronic form of the Masters Thesis in the collection "Dela FRI".

In Ljubljana, 25. January 2017 Author's signature:

Acknowledgments

I would like to thank my advisor, Prof. Erik Štrumbelj, for his guidance through this year-long endeavour. I also wish to thank my mother and Iva, for their unending support and assistance.

Aleksandar Dimitriev, 2017

Contents

List of Abbreviations

ADVI	Automatic Differentiation Variational Inference
CLGP	Categorical Latent Gaussian Process
ELBO	Evidence Lower Bound
EM	Expectation Maximization
FA	Factor Analysis
GLM	Generalized Linear Model
GP	Gaussian Process
GPFA	Gaussian Process Factor Analysis
GPLVM	Gaussian Process Latent Variable Model
ICA	Independent Component Analysis
i.i.d.	independent and identically distributed
KL	Kullback-Leibler divergence
MCMC	Markov Chain Monte Carlo
MF-ADVI	Mean Field Automatic Differentiation Variational Inference
ML	Machine Learning
MVP	Multivariate Poisson
PCA	Principal Component Analysis
PFA	Poisson Factor Analysis
PFALGP	Poisson Factor Analysis using Latent Gaussian Processes
PPCA	Probabilistic Principal Component Analysis
RMSE	Root Mean Square Error

Povzetek

Multivariatni števni podatki so pogosti na področjih kot so šport, nevroznanost in besedilno rudarjenje. Modeli, ki lahko natančno opravljajo faktorsko analizo, so potrebni zlasti za strukturirane podatke kot na primer števne matrike s časovnimi vrstami. Predstavljamo Poissonovo faktorsko analizo z latentnimi Gaussovimi procesi, ki je nova metoda za analizo multivariatnih števnih podatkov. Naš pristop omogoča analizo odvisnih podatkov, ki so povezani v latentnem prostoru s pomočjo Gaussovega procesa. Zaradi eksponentne nelinearnosti v modelu ne obstaja analitična rešitev. Zato smo razvili postopek maksimizacije pričakovane vrednosti z Laplacovim približkom za lažjo uporabo. Predstavljamo tudi rezultate na različnih podatkovnih naborih, tako sintetičnih kot realnih, v primerjavi z drugimi metodami faktorske analize. Naša metoda je kvalitativno in kvantitativno boljša za podatke iz ne-i.i.d. Poisson porazdelitev, saj so predpostavke, ki jih naredi, primerne za podatke.

Ključne besede

faktorska analiza, Gaussovi procesi, latentni prostor, Poisson, števni podatki

Abstract

Multivariate count data are common in some fields, such as sports, neuroscience, and text mining. Models that can accurately perform factor analysis are required, especially for structured data, such as time-series count matrices. We present Poisson Factor Analysis using Latent Gaussian Processes, a novel method for analyzing multivariate count data. Our approach allows for non-i.i.d observations, which are linked in the latent space using a Gaussian Process. Due to an exponential non-linearity in the model, there is no closed form solution. Thus, we resort to an expectation maximization approach with a Laplace approximation for tractable inference. We present results on several data sets, both synthetic and real, of a comparison with other factor analysis methods. Our method is both qualitatively and quantitatively superior for non-i.i.d Poisson data, because the assumptions it makes are well suited for the data.

Keywords

factor analysis, Gaussian process, latent space, Poisson, count data

Razširjen povzetek

Z nedavnim povečanjem strojnega učenja skupaj z vedno več podatki, so možnosti in potrebe po analizo različnih podatkov večje. Ti podatki so lahko realni, binarni, števni itd. V tem magistrskem delu se bomo osredotočali na analizo števnih podatkov, natančneje, na nenadzorovano učenje iz večvariatnih števnih podatkov - matrike z nenegativnimi celimi števili v celicah. Naš cilj je identificirati zmanjšano predstavitev teh podatkov, kjer zmanjšamo število stolpcev brez izgube informacij. Z drugimi besedami, delamo faktorsko analizo nad števnimi matrikami ob predpostavki, da je verjetnost števil porazdeljena po Poissonovi porazdelitvi.

Poissonova faktorska analiza ni nov pojem. Obstaja veliko pristopov, ki hočejo najti skrite faktorje iz števnih matrik, vendar večina metod je narejenih za neodvisne podatke. Naša pozornost bo usmerjena na podatke s strukturo kot na primer, analiza evolucije več koreliranih števcev čez čas. Podoben primer lahko pogosto najdemo v športu - recimo, skozi košarkarsko sezono lahko štejemo število metov (2 in 3 točk), preobratov, in prekrškov. Opazimo lahko, da imamo več takih časovnih vrst čez eno leto. Poleg tega vemo, da so pari teh števcev zelo korelirani kot na primer število zadetih metov in število poskusnih metov. Torej obstaja neka latentna struktura nižje dimenzije kot na primer en faktor za proste mete, en faktor za tritočkovni met, itd. Bilo bi koristno, če bi te faktorje poznali. Prav tako bi bilo dobro odstraniti šum ene tekme in videti kako se spreminja ta faktor čez čas. Na primer, neka ekipa je mogoče začela sezono z nizko natančnostjo, ki pa se proti koncu sezone počasi izboljša. Število metov v eni tekmi je zelo šumno

in odvisno od nasprotnika, ampak trend iz več kot 40 tekem mora biti očiten. Hkrati hočemo gladko funkcijo čez čas.

Odločili smo se za latentne Gaussove procese. Gaussovi procesi so relativno mlad neparametrični pristop, katerega popularnost narašča. Neparametričnost pomeni, da ne predpostavljamo, kako se bodo faktorji spreminjali čez čas (npr. pogosta predpostavka so avtoregressivni modeli, kjer je naslednja točka verjetnostno porazdeljena glede na eno ali več prejšnjih točk), ampak pustimo podatkom, da določijo strukturo. Odvisno od jedrske funkcije, ki jo moramo izbrati, se latenti prostor lahko počasi ali hitro spreminja in je lahko odvedljiv (ali ne), itd.

S tem smo določili naš model, ki ima naslednji formalni opis. Naj bo N, M, in D, število časovnih točk na vrsto, število časovnih vrst in število latentnih faktorjev v tem zaporedju. Predpostavljamo, da lahko določimo nek $N \times D$ latentni prostor (matrika X) nekoreliranih faktorjev iz danih $N \times M$ številk (matrika Y). To lahko dosežemo v dveh korakih. Najprej predpostavimo, da je Y_{ij} pogojno neodvisen od ostalih številk v matriki in porazdeljen po Poissonovi porazdelitvi, če poznamo njegov parameter λ_{ij}, t.j, $Y_{ij} \sim \text{Pois}(\lambda_{ij})$. Struktura (časovna) in model korelacije se skrivata v tej matriki parametrov Λ. Predpostavljamo, da je $\Lambda = \exp(X \cdot L + 1_N \cdot b^T)$, kjer je X latentni prostor, L je t.i. matrika obremenitvenih faktorjev in b je vektor povprečja. Korelacije med opazovanimi vrstami dobimo s pomočjo L, med tem ko časovno korelacijo določimo z apriorno porazdelitvijo nad stolpci matrike X. Podrobneje, predpostavljamo, da imamo D Gaussovskih procesov, vsak s svojo kovariančno matriko K_d, iz katerih izvlečemo N-dimenzionalne korelirane vektorje: $x_d \sim \mathcal{GP}(0, K_d)$. Kot jedrno funkcijo uporabljamo kvadratno eksponentno (Gaussovo) funkcijo: $(K_d)_{ij} = \sigma \exp((||t_i - t_j||_2/\rho_d)^2) + \delta(i,j) \cdot \epsilon$, kjer $\sigma = 1$ in $\epsilon = 0.001$ zaradi identifikacije modela. Torej, parametri modela so $\Theta = \{X, L, b, \rho\}$, katerih vrednosti popolnoma določajo model. Za apriorne porazdelitve ostalih parametrov uporabljamo šibko informativne normalne porazdelitve.

Kljub preprosti definiciji je učenje parametrov tega modela težje zaradi

nelinearnosti, ki se skriva v λ, in metode vzorčenja lahko trajajo veliko ur, odvisno od števila podatkov. Zato smo razvili aproksimacijo z metodo maksimizacije pričakovanja (EM) za učinkovito sklepanje skupaj z Laplaceovo aproksimacijo. Motivacija za EM je naslednja. če fiksiramo latentni prostor, je iskanje najboljših parametrov, kot je L, konveksen problem, ki je enostavno rešljiv. Hkrati, če fiksiramo vse razen X, je problem spet konveksen in lahko najdemo X, ki maksimizira logaritem posteriorne porazdelitve. Ker EM zahteva pogojno verjetnostno porazdelitev X, ki nima zaprte rešitve, jo aproksimirano z Gaussovo porazdelitivjo centrirano na maksimum (ker je konveksna v X, je maximum edini in globalni). Gradiente in Hessovo matriko vseh spremenljivk modela določimo zato, da lahko izračunamo vsako iteracijo EM algoritma s katerokoli gradientno metodo kot na primer gradientni spust, L-BFGS ali Newtonova metoda.

Učinkovitost aproksimacije primerjamo z dejanskimi vzorci iz metode MCMC, ki vračajo natančno porazdelitev (če je model identficiran), in aproksimacijo srednjega polja varijacijskega računa. Natančnost metode primerjamo tudi z dvema ostalima metodama faktorske analize, kjer ena uporablja Gaussove procese v latentnem prostoru. Za identifikacijo našega modela uporabljamo inicializacijo z rezultati faktorske analize, in s fiksiranjem povprečja in variance latentnega prostora. Nad umetno generiranimi podatki se naša metoda (PFALGP) in aproksimacije izkažejo kot najboljše glede na prečno preverjanje, tudi kvalitativno glede na določene faktorje. PFALGP najde enake faktorje, kot so resnični latentni faktorji, ki so generirali te podatke.

Modele primerjamo na treh realnih podatkovnih naborih iz treh različnih domen: besedila (twitter), šport (NBA), in računalniški vid (MNIST). Iz vseh treh lahko sklepamo, da naš model da boljše napovede in najde bolj informativne latente faktorje od ostalih metod. Iz twitter podatkov vidimo, da naš model najde tedensko nihanje kot en faktor in konico na koncu tedna kot drugi faktor. Na NBA podatkih vidimo, da naš model povezuje podobne faktorje tudi če imamo samo dve latentne dimenzije. Vidimo tudi, da s

povečanjem števila dimenzij lahko dosežemo manjšo napako. Interpretacija faktorjev je zelo očitna pri MNIST podatkih, ko vizualiziramo latentni prostor kot 2D sliko, in vidimo da naš model najde faktorje, ki ustrezajo 5 uporabljenim številkam.

V prihodnosti bi želeli najti boljši način določanja števila faktorjev ali celo marginalizacijo preko vseh števil ter pospešitev celotne metode, npr. z redkimi Gaussovimi procesi ali aproksimacije kovariančne matrike z nizkim rangom.

Chapter 1

Introduction

With the recent emergence of machine learning (ML) and the ever larger amount of data available, there has been a surge of applying ML to different kinds of data (real-valued, binary, count etc.). The two types of ML commonly used on this data are supervised and unsupervised learning. The former is more easily defined: it seeks to learn an association between the data and labels. On the other hand, unsupervised learning seeks to find structure in unlabeled data. This structure can be useful in and of itself, although unsupervised learning is mostly used as a pre-processing step in supervised learning. The data is usually put through some dimensionality reduction technique such as Principal Components Analysis (PCA) and then a supervised method learns the association between the resulting data and the labels.

Our focus, however, is on the latent structure. The earliest approaches to finding hidden structure in the data date back to the formulation of PCA and factor analysis (FA) for continuous, more specifically Gaussian, data. The former finds a unique rotation matrix such that the columns explain as much variance as possible in descending order. The solution is easily obtained by singular value decomposition. The latter is a broader method that seeks an explanation of the data by a linear projection from a lower-dimensional space that differs from the original data by more than just rotation. Its generality,

however, leaves too many degrees of freedom and there is no unique solution.

Unfortunately, the kind of data we are interested in is multivariate count data, which is not amenable to analysis using these basic methods. What we are trying to model is non-negative integer matrices, which arise in many different areas including, but not limited to: number of auto insurance claims, highway accidents, crimes, voting, user-item recommendation or, in general, text analysis, images (computer vision), stock volume trading, and sports matches statistics.

To illustrate with a specific example, in sports it is common to gather information for each match, and this information is largely different counts. In basketball, for example, each match records the number of shots (two-point and/or three-point) attempted and made, number of fouls, rebounds etc. Thus, our matches are the observations, and the features are the different recorded counts, which clearly fits our criteria.

Our problem can more formally be stated as follows: let N be the number of observations and M the dimensionality, so that we have at our disposal a $Y \in \mathbb{N}_0^{N \times M}$ non-negative integer matrix, where rows correspond to observations and columns to features. We can also think of having N data points lying in an M-dimensional non-negative discrete space. Our chief assumption is that the observations in M-dimensional space lie on a lower D-dimensional continuous manifold $X \in \mathbb{R}^{N \times D}$ that is unknown to us, a manifold which we seek to discern. This problem is generally called dimensionality reduction, but we are considered in factor analysis , which has received considerable attention in the literature. Most research, however, has tackled the continuous, most commonly Gaussian, case for Y. Moreover, the observations Y are usually assumed to be independent and identically distributed (i.i.d.) or at least exchangeable. Our approach considers alternate formulations for both assumptions. We wish to deal with data sets that are discrete and not continuous like the Gaussian distribution. In addition, the data we are interested in, such as basketball matches, has (a time-series) structure, so we forego the i.i.d. assumption as well. Depending on our assumptions, we obtain different

Table 1.1: Comparison of the assumptions of different general model structures.

model	unsupervised	distribution	latent	mapping	i.i.d
Poisson regression [3]	✗	Poisson	✗	non-linear (exp)	✓
MVP [4]	✗	Poisson	✗	linear	✓
PPCA [5]	✓	Gaussian	✗	linear	✓
ICA [6]	✓	non-Gaussian	✓	linear	✓
FA [7]	✓	Gaussian	✓	linear	✓
PFA [8]	✓	Poisson	non-negative	linear	✓
GPLVM [2]	✓	Gaussian	✓	non-linear (GP)	✓
CLGP [1]	✓	Categorical	✓	non-linear (GP)	✓
GPFA [9]	✓	Gaussian	GP	linear	time-series
PFALGP (ours)	✓	Poisson	GP	non-linear (exp)	time-series

models (see 1.1 for an overview).

Our motivation stems from a recent work on using a Gaussian Process for non-linear factor analysis of (multivariate) categorical variables by Gal et al. [1]. Their use of latent variables mirrors the well-known Gaussian Process Latent Variable Model (GPLVM) [2], except that it is the softmax inputs that are projected from a latent space. Our goal is thus to transfer this approach to count data and develop an unsupervised approach to modeling multivariate count data using latent Gaussian Processes.

Chapter 2

Related work

On the side of supervised multivariate count analysis several methods have been developed [3, 4, 10]. Chib and Winklemann [3] develop a multivariate generalized linear model (GLM) for Poisson and perform inference using Markov Chain Monte Carlo (MCMC), used by Park and Lord [10] for modelling intersection accidents. Although most methods for count data employ Poisson likelihoods, a negative binomial regression in combination with a copula to model the correlations has been used [11, 12, 13]. For copulas with Poisson marginals see Bauerle and Grubel [14].

A general formulation of a M-variate Poisson distribution and regression is given by Karlis et al. [4]. Unfortunately as the number of features M grows, estimating the probability of a draw from such a multivariate Poisson distribution requires summing over a large number of terms. This number grows exponentially with M, and inference quickly becomes intractable with this method.

Poisson Factor Analysis (PFA) [8, 15, 16, 17] is an overarching term for latent models for (usually i.i.d.) count data. For a unified view of Poisson and other discrete component analysis see Buntine and Jakulin [17]. Depending on the prior distributions imposed on the latent factors we obtain different aproaches from the literature. For example, placing a prior with support on the positive reals results in non-negative matrix factorization. Different

link functions for the observed data define different likelihoods, e.g. Poisson for counts results in general PFA. For an overview of link functions and distributions for the parameters in PFA see Wedel et al. [15].

Topic models like the well-known Latent Dirichlet Allocation (LDA) [18] have been shown to be a type of non-negative matrix factorization. LDA can be seen as PFA with a Dirichlet prior for the latent topic matrix [8]. Zhou et al. [8] generalize this while developing a beta-binomial model for PFA with possibly infinite latent variables. A more recent work by Acharya et al. [19] tackles the case of time-dependent count data with the aforementioned Gamma-Poisson PFA. This is achieved by adjusting the Gamma parameters of a time-series point using the Gamma parameters of the previous point. Most recently, Schein et al. [20] extend this by modeling temporal dependence using the gamma shape parameter conditioned on the previous point's shape parameters.

Deep Exponential Families [21] is a recent generalization of factor analysis in the prior-hyperprior direction, where the parameters of the latent variables are again latent variables with an exponential family distribution, and so on continuing "deeper". Similarly, Gan et al. [22] use a deep Boltzmann machine as a prior for a binary latent variable that induces sparsity, such that each observation is associated with only some of the latent factors. Improving on this, Henao et al. [23] construct a "deep" PFA by replacing the Boltzmann machine with a whole latent PFA. Analogously, Zhou et al. [24] model Poisson counts by constructing a Poisson-Gamma belief network. They draw the shape parameter of a Gamma-distributed latent factor layer from a Gamma prior, which in turn becomes another latent layer.

PFA can be augmented with a possibly infinite amount of factors by using Bayesian nonparametrics [8, 16, 19, 25, 26] and marginalizing over the number of factors. Titsias [16] develops a Gamma-Poisson feature model with an implicitly infinite number of latent factors. Gopalan et al. [26] derive an infinite Poisson-Gamma model by setting a Gamma Process prior (with finite base measure to ensure a non-infinite dot product) on the latent space.

Lopes et al. [27] use reversible-jump MCMC to include the number of factors as parameter to be estimated.

Whereas most approaches model matrix data, Schein et al. [28] approach count matrices as a special case of a second-order tensor, developing the general case and applying it to dyadic events (matrices). For a coupled matrix factorization such as words-documents and documents-users Gopalan et al. [29] derive a shared latent space approach and apply it to topic modeling.

In econometrics, dynamic factor models are commonly used [30, 31, 32], but usually for continuous data. Jung et al. [32], however, use a dynamic factor model with a Poisson likelihood, but their temporal structure is an autoregressive Gaussian.

Several approaches have been developed in the neuroscience literature for Poisson spike counts of neurons [33, 9, 34, 35, 36, 37]. Cunningham et al, [33] use an inhomogeneous Gamma interval process, with a truncated Gaussian Process (GP) prior on the rate (because of the non-negativity). Petreska et al. [34] link several markov chains to ensure temporal dependence, but model the count data as Gaussian. Semedo et al. [36] model the neural time dependence by adding temporal constraints on canonical correlation analysis, whereas Archer et al. [35] use a Poisson linear dynamical system. More recently, Park et al. [37] use the same dynamical system combined with a GP prior across different trials (instead of across time).

Our derivation has several similarities with the method of Yu et al. [9], which they call Gaussian Process Factor Analysis (GPFA). Like our approach, they use it to model multivariate Poisson data (neuronal activity through time specifically), and they also use a GP prior in the time domain to ensure temporal smoothness, but their derivation otherwise follows the standard factor analysis approach, because there is no non-linearity employed between the observations and the latent space, whereas we use the log link. Their simplifying assumptions result in a tractable posterior, as the the posterior is jointly Gaussian, whereas we have to resort to approximations. However, they do not employ a Poisson likelihood, but instead work

with the square roots of the counts to do away with the heteroscedasticity of the Poisson rate. This is because the variance of a Gaussian, which is independent of its mean, cannot model the mean-variance relationship of the Poisson properly.

So far we've considered a discretized view of time-series. By considering an N-by-M matrix we have implicitly binned the Poisson realization events of M Poisson point processes into N time intervals (bins). There is another group of related approaches from the stochastic process perspective, with considerable work being done on the most common approach: Cox processes [38, 39, 40], the doubly stochastic Poisson point process. It is doubly stochastic because the underlying rate λ is no longer constant, but can vary through time. A suitably transformed Gaussian Process can be used to model the non-negative point process intensity, e.g. by using the the log link function: $ln(\lambda(t)) \sim \mathcal{GP}(0, K)$ or the logit function: $logit(\frac{\lambda(t)}{\lambda^*}) \sim \mathcal{GP}(0, K)$, where λ^* is some upper bound on the intensity [39]. This is used by Adams et al. [39] for the one-dimensional case with MCMC sampling, whereas Lloyd et al. [41] developed a variational approach.

For several linked point processes Miller et al. [42] use non-negative matrix factorization on the intensity of each process through time. Gunter et al. [40] develop the first Bayesian treatment of dependent Cox processes using convolutions to model the structure and MCMC for inference. Lloyd et al. [43] develop Latent Poisson Process Allocation (LPPA) as a continuous version of non-negative matrix factorization, being, in essence, a topic model for Poisson point processes. In our areas of interest, however, the data are already time-discretized (e.g. sports matches) and thus the continuous nature of the point process can avoided, so that there is no need to invoke the complex machinery needed to estimate the latent space and factors.

Chapter 3

Poisson Factor Analysis using Latent Gaussian Processes

We now derive a factor analysis method for Poisson observations with a latent space governed by Gaussian Processes. For convenience, we state our notation once more: the observed data are $\mathbf{Y} \in \mathbb{N}_0^{N \times M}$, where N and M are the number of observations and Poisson-distributed features, respectively. In addition, uppercase bold letters indicate matrices, whereas lowercase bold letters indicate column vectors. The latent space is $\mathbf{X} \in \mathbb{R}^{N \times D}$ where D is its dimensionality. We assume that the observed count data are Poisson distributed:

$$P(Y|\mathbf{\Lambda}) = \prod_{n=1}^{N} \prod_{m=1}^{M} \text{Pois}(Y_{nm}|\lambda_{nm}) = \prod_{n=1}^{N} \prod_{m=1}^{M} \frac{\lambda_{nm}^{Y_{nm}} e^{-\lambda_{nm}}}{Y_{nm}!} \tag{3.1}$$

where λ_{nm} is the rate parameter for the Y_{nm}^{th} observation. Furthermore, we assume that the counts are conditionally independent given the λ's. The underlying rates, however, are connected through an exponential nonlinearity using the factor loading matrix $\mathbf{L} \in \mathbb{R}^{D \times M}$ and the latent space $\mathbf{X}$ as follows:

$$\mathbf{\Lambda} = [\lambda_{nm}] = \exp(\mathbf{X} \cdot \mathbf{L} + \mathbf{1}_M \cdot \mathbf{b}^T) \tag{3.2}$$

where exp denotes elementwise exponentiation, $\mathbf{b} \in {\mathbb{R}_+}^M$ is a vector of baseline rates for each feature and $\mathbf{1}_M$ is an M-dimensional vector of ones. The

$\mathbf{X} \cdot \mathbf{L}$ product models the dependence between the M features. More specifically, we have:

$$\lambda_{nm} = \exp(\mathbf{x}_n^T \cdot \boldsymbol{l}_m + b_m) \tag{3.3}$$

where $\mathbf{x}_n^T$ denotes the n^{th} row of $\mathbf{X}$ and $\boldsymbol{l}_m$ the m^{th} column of $\mathbf{L}$. To tackle the time-series nature of our data, we put a Gaussian process prior on each of the D N-dimensional vectors (columns of $\mathbf{X}$):

$$\mathbf{x}_d \sim \mathcal{GP}(0, \mathbf{K}_d) \tag{3.4}$$

For the covariance matrix of the GP, we use the squared exponential (SE) kernel, also known as a Gaussian kernel, whose ij^{th} element equals:

$$(\mathbf{K}_d)_{ij} = \sigma^2 \exp(-\frac{1}{2}\frac{(t_i - t_j)^2}{\rho_d^2}) \tag{3.5}$$

where for numerical stability, we add a small value ϵ^2 to the diagonal to ensure positive definiteness. Rewriting this in matrix notation we have:

$$\mathbf{K}_d = \sigma^2 \exp(-\frac{1}{2\rho_d^2}\mathbf{T}) + diag(\epsilon^2) \tag{3.6}$$

where $T_{ij} = ||t_i - t_j||_2^2$, i.e. T is a matrix of dot products of the differences (in case t_i is multidimensional). Here $||x||_2 = \sqrt{\sum_i x_i^2}$ denotes the L_2 norm.

The parameters of this kernel are: the kernel variance σ^2, the noise variance ϵ^2 and the length scale ρ_d^2. The kernel and noise variance are fixed to ensure identifiability, whereas each dimension of the latent space has its own length scale ρ_d^2 that is free to vary.

For the loading matrix and base rates, we employ weakly informative normal priors elementwisely:

$$L_{dm} \sim \mathcal{N}(0, \sigma_0^2), \quad m = \{1 \,..\, M\}, \;\; d = \{1 \,..\, D\} \tag{3.7}$$

$$b_m \sim \mathcal{N}(0, \sigma_0^2), \quad m = \{1 \,..\, M\} \tag{3.8}$$

where we set the variance to a large value, e.g. $\sigma_0^2 = 100$.

3.1 Identification

Almost all factor analysis methods, both Bayesian and frequentist, are plagued by unidentifiability [30, 44, 45]. A model is said to unidentifiable if the same probability distribution is produced by two or more distinct sets of parameters, or in other words, these parameters that can not be uniquely distinguished on the basis of the data. For example, in a K-mixture model, permuting the indexes of the latent variables (denoting which mixture each point belongs to) does not change the probability, which means there are $K!$ equally likely configurations.

In our case, the factor loading matrix L and the latent space X are doubly unidentifiable (ignoring the base rates at the moment), although their product is identifiable. Too see this, notice that first, multiplying L by a scalar and X by its inverse results in the same product. Second, even if we fixed the scale of L or X, we can still rotate L by some amount, and apply the inverse rotation to X to obtain an equivalent solution (actually any invertible matrix will suffice, but a rotation is intuitively understandable) [45].

In Bayesian methodology, an unidentifiable modeled can sometimes be fixed with a prior distribution on the parameters. For example, the issue of scale for L and X is easily taken care of by assigning them normal priors. In the case of X, since it is drawn from a GP, we give it a zero mean and a variance of 1 in the SE kernelin Eq. 3.5, setting σ^2 to 1 and ϵ^2 to 0.001. The rotational ambiguity is resolved by imposing a lower triangular structure on L, with non-negative elements on the diagonal. Since L is M-by-D and not square, the constraint applies to the $D(D-1)$ elements above the diagonal. Other approaches include post-ex identification, by transforming the samples to align them to the same mode [44]. We use weakly informative priors and initialization using FA.

3.2 Inference

Depending on the size of the model (number of samples, features etc.) sampling-based methods can run from hours to days. In addition to MCMC sampling in `Stan` [46], we employ auto-diff variational inference present in `Stan` [46] as well as an Expectation Maximization approach with Laplace approximation for more tractable inference.

3.2.1 Markov Chain Monte Carlo

We implemented the model in `Stan` [46], a general probabilistic programming language written in `C++`, with an interface for `R` among other languages. `Stan` uses Hamiltonian Monte Carlo [47], specifically the No-U-Turn Sampler (NUTS) [48] to perform Markov chain Monte Carlo (MCMC) inference of the posterior distribution. The bias of MCMC samples is zero, and the probability distribution of the samples converges to the true distribution as $N \to \infty$. However, MCMC is slow for larger models, so we have also employed approximations to the posterior for faster inference at the cost of accuracy.

3.2.2 Variational inference

An alternative way of approximate inference are variational methods. This entails choosing an approximate posterior distribution $q(X)$, that is easier to calculate than the true posterior distribution $p(X|Y;\Theta)$. We briefly outline variational inference below, though a longer overview can be found in Blei et al. [49]. More formally, we are trying to find $q(X)$ from our family of tractable distributions that is closest to the intractable posterior $p(X|Y;Theta)$:

$$\arg\min_{q(X)} \quad \mathrm{KL}(Q(X) \;||\; P(X|Y;\Theta) \tag{3.9}$$

To measure the "closeness" of the two distributions we use Kullback-Leibler divergence (KL), a non-negative asymmetric measure:

$$\begin{aligned}\mathrm{KL}(Q \mid\mid P) &= \int q(x) \log \frac{q(x)}{p(x)} dx = \int q(x) \log q(x) dx - \int q(x) \log p(x) dx \\ &= E_{q(x)}[\log q(x)] - E_{q(x)}[\log p(x)]. \end{aligned} \tag{3.10}$$

The asymmetry of KL divergence is a useful property, because KL divergence penalizes areas of high $p(x)$ and low $q(x)$ and does not penalize areas of low $p(x)$ if $q(x)$ badly approximates there. Intuitively, we want this because when the probability of something occurring is high, we want the approximation to be good, and we don't care if we've badly approximated events with low probability that aren't likely to happen[1]. Thus, the specific KL divergence in our case is:

$$\mathrm{KL}(Q(X) \mid\mid P(X|Y;\Theta) = E_{q(x)}[\log q(X)] - E_{q(x)}[\log p(X|Y;\Theta)] \tag{3.11}$$

$$= E_{q(x)}[\log q(X)] - E_{q(x)}[\log p(X,Y;\Theta)] \tag{3.12}$$

$$+ \log p(Y;\Theta) \tag{3.13}$$

Defining the Evidence Lower BOund as $ELBO = E_{q(x)}[\log p(X,Y;\Theta)] - E_{q(x)}[\log q(x)]$, we see that the marginal likelihood can be decomposed into the ELBO and KL divergence, and maximizing the ELBO is equivalent to minimizing the KL divergence:

$$\log p(Y;\Theta) = ELBO + \mathrm{KL}(Q(X) \mid\mid P(X|Y;\Theta) \tag{3.14}$$

and since KL divergence is non-negative, we see that the ELBO is a lower bound on the marginal likelihood (also called the evidence), hence the name.

The main drawback of variational inference is that the variational approximation needs to be hand-derived for each model. An easier and less error-prone way is to employ Automatic Differentiation Variational Inference (ADVI) [50]. ADVI is a recent approach to so-called Black Box Variational

[1]The term variational inference is synonymous with $KL(Q \mid\mid P)$, but there are methods that use the opposite $KL(P \mid\mid Q)$, such as Expectation Propagation.

Inference [51]. It leverages the fact that `Stan` already computes gradients automatically. Thus, ADVI uses the gradients of the model, i.e. the posterior distribution, instead of resorting to gradients of the variational approximation as [51] requires, which leads to lower variance of the gradient. Since `Stan` has the ADVI functionality, obtaining samples from the model posterior is as effortless as changing one function in the code, to perform approximate inference instead of MCMC sampling.

3.2.3 Laplace approximation

When full inference is too costly, we may use tractable methods to approximate the posterior distribution. Since the logarithm is a monotonically increasing function, the parameters that maximize the posterior also maximize the log posterior. Let $\mathbf{\Theta} = \{\mathbf{L}, \mathbf{b}, \boldsymbol{\rho}\}$ denote all the unknown parameters except the latent space $\mathbf{X}$. The Laplace approximation entails using a Gaussian centered at a maximum of the log posterior with a covariance that is the inverse of the Hessian at that maximum.

The following identities will be useful in deriving the Laplace approximation:

$$
\begin{aligned}
vec(\mathbf{ABC}) &= (\mathbf{C}^T \otimes \mathbf{A}) \cdot vec(\mathbf{B}) & \qquad \frac{\partial}{\partial \mathbf{x}} \mathbf{x}^T \mathbf{S} \mathbf{x} &= 2\mathbf{S}\mathbf{x} \\
tr(\mathbf{A}^T \mathbf{B}) &= vec(\mathbf{A})^T \cdot vec(\mathbf{B}) & \qquad \frac{\partial}{\partial \mathbf{x}\mathbf{x}^T} \mathbf{x}^T \mathbf{S} \mathbf{x} &= \mathbf{S} \\
\frac{\partial}{\partial x} \mathbf{K}^{-1} &= -\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial x} \mathbf{K}^{-1} & \qquad \frac{\partial}{\partial x} \ln \det \mathbf{K} &= tr(\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial x})
\end{aligned}
$$

where S denotes a symmetric matrix, $\otimes$ is the Kronecker product, $vec()$ is the vec operator that stacks the columns of a matrix into one long vector, $tr()$ is the trace operator, $\frac{\partial}{\partial \mathbf{x}}$ denotes the gradient, and $\frac{\partial}{\partial \mathbf{x}\mathbf{x}^T}$ is the Hessian matrix. An overview of matrix calculus and the proofs of these identities can be found in Petersen et al. [52].

The log posterior is proportional to the joint:

$$\begin{aligned}\mathcal{L} &\propto \ln P(\mathbf{Y}, \mathbf{X}, \mathbf{\Theta}) = \ln P(\mathbf{Y}|\mathbf{X}, \mathbf{\Theta}) + \ln P(\mathbf{X}) + \ln P(\mathbf{\Theta}) \\ &= \sum_{n=1}^{N}\sum_{m=1}^{M} \ln \text{Pois}(Y_{nm} \mid \lambda_{nm}) + \sum_{d=1}^{D} \ln \mathcal{N}(\mathbf{x}_d \mid \mathbf{0}, \mathbf{K}_d) \\ &\qquad + \sum_{d=1}^{D}\sum_{m=1}^{M} \ln \mathcal{N}(L_{dm} \mid 0, \sigma_0^2) + \sum_{m=1}^{M} \ln \mathcal{N}(b_m \mid 0, \sigma_0^2) \\ &= \sum_{n=1}^{N}\sum_{m=1}^{M} Y_{nm} \cdot \ln \lambda_{nm} - \lambda_{nm} - \sum_{d=1}^{D} \frac{1}{2}\mathbf{x}_d^T \mathbf{K}_d^{-1} \mathbf{x}_d \\ &\qquad - \sum_{d=1}^{D}\sum_{m=1}^{M} \frac{L_{dm}}{2\sigma_0^2} - \sum_{m=1}^{M} \frac{b_m}{2\sigma_0^2} + const\end{aligned} \tag{3.15}$$

where $\lambda_{nm} = \exp(\mathbf{x}_n^T \cdot \boldsymbol{l}_m + b_m)$ as before. The above can be written more succinctly using matrix notation and by omitting constant terms not containing $\mathbf{X}$ or $\mathbf{\Theta}$. Let $\mathbf{\Sigma}_0 = diag(\sigma_0^2)$ denote the diagonal prior covariance matrix of $\mathbf{L}$ and $\mathbf{b}$ (though we are abusing notation since $\mathbf{L}$ and $\mathbf{b}$ are of different sizes). We thus have:

$$\begin{aligned}\mathcal{L} &\propto tr(\mathbf{Y}^T \ln \mathbf{\Lambda}) - tr(\mathbf{\Lambda} \cdot \mathbf{1}_{M\times N}) - \frac{1}{2}\mathbf{x}^T \mathbf{K}^{-1} \mathbf{x} - \frac{1}{2}\boldsymbol{l}^T \mathbf{\Sigma}_0^{-1} \boldsymbol{l} - \frac{1}{2}\mathbf{b}^T \mathbf{\Sigma}_0^{-1} \mathbf{b} \\ &= tr(\mathbf{Y}^T \cdot (\mathbf{XL} + \mathbf{B})) - tr(\exp\Big(\mathbf{XL} + \mathbf{B}\Big) \cdot \mathbf{1}_{M\times N}) \\ &\qquad - \frac{1}{2}(\mathbf{x}^T \mathbf{K}^{-1} \mathbf{x} + \boldsymbol{l}^T \mathbf{\Sigma}_0^{-1} \boldsymbol{l} + \mathbf{b}^T \mathbf{\Sigma}_0^{-1} \mathbf{b})\end{aligned} \tag{3.16}$$

where exp() denotes elementwise exponentiation, $\mathbf{B} = \mathbf{1}_N \cdot \mathbf{b}^T$ is a N-by-M matrix whose rows are the base rate vector $\mathbf{b}^T$, whereas $\mathbf{K}$ is the block-diagonal matrix containing the covariance matrices $\mathbf{K}_d \;\; d = \{1 \,..\, D\}$ on its diagonal.

We now derive the gradient and hessian of $\mathcal{L}$ with respect to each variable. Let $\mathbf{y} = vec(\mathbf{Y})$, $\mathbf{x} = vec(\mathbf{X})$, $\boldsymbol{l} = vec(\mathbf{L})$. We rewrite $\mathcal{L}$ in terms of $\mathbf{x}$ when deriving its gradient:

$$\begin{aligned}\mathcal{L} &\propto \mathbf{y}^T \cdot ((\mathbf{L}^T \otimes \mathbf{I}_N) \cdot \mathbf{x} + vec(\mathbf{B})) - \frac{1}{2}\mathbf{x}^T \mathbf{K}^{-1} \mathbf{x} \\ &\qquad - vec(\mathbf{1}_{M\times N})^T \cdot \exp\Big((\mathbf{L}^T \otimes \mathbf{I}_N) \cdot \mathbf{x} + vec(\mathbf{B})\Big)\end{aligned} \tag{3.17}$$

where we omit terms not containing X as their derivative is zero. Taking the gradient with respect to $\mathbf{x}$ we have:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{x}} &= (\mathbf{L} \otimes \mathbf{I}_N) \cdot (\mathbf{y} - \exp\Big((\mathbf{L}^T \otimes \mathbf{I}_N) \cdot \mathbf{x} + vec(\mathbf{B}))\Big) - \mathbf{K}^{-1}\mathbf{x} \\ &= (\mathbf{L} \otimes \mathbf{I}_N) \cdot (\mathbf{y} - vec(\mathbf{\Lambda})) - \mathbf{K}^{-1}\mathbf{x}\end{aligned} \tag{3.18}$$

and its Hessian is then:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}\mathbf{x}^T} = -(\mathbf{L} \otimes \mathbf{I}_N) \cdot diag(vec(\mathbf{\Lambda})) \cdot (\mathbf{L}^T \otimes \mathbf{I}_N) - \mathbf{K}^{-1}. \tag{3.19}$$

To find the gradient with respect to $\boldsymbol{l} = vec(\mathbf{L})$ we again rewrite the log posterior:

$$\begin{aligned}\mathcal{L} \propto\ & vec(\mathbf{X}^T\mathbf{Y})^T \cdot vec(\mathbf{L}) - \frac{1}{2} vec(\mathbf{L})^T \mathbf{\Sigma}_0^{-1} vec(\mathbf{L}) \\ & - vec(\mathbf{1}_{M\times N})^T \cdot \exp\Big((\mathbf{I}_M \otimes \mathbf{X}) \cdot vec(\mathbf{L}) + vec(\mathbf{B})\Big) \\ =\ & vec(\mathbf{X}^T\mathbf{Y})^T \cdot \boldsymbol{l} - vec(\mathbf{1}_{M\times N})^T \cdot \exp\Big((\mathbf{I}_M \otimes \mathbf{X}) \cdot \boldsymbol{l} + vec(\mathbf{B})\Big) - \frac{1}{2}\boldsymbol{l}^T \mathbf{\Sigma}_0^{-1} \boldsymbol{l}\end{aligned} \tag{3.20}$$

so that the gradient is easily obtained as:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \boldsymbol{l}} &= vec(\mathbf{X}^T\mathbf{Y}) - (\mathbf{I}_M \otimes \mathbf{X}^T) \cdot \exp\Big((\mathbf{I}_M \otimes \mathbf{X}) \cdot \boldsymbol{l} + vec(\mathbf{B})\Big) - \mathbf{\Sigma}_0^{-1}\boldsymbol{l} \\ &= vec(\mathbf{X}^T\mathbf{Y}) - (\mathbf{I}_M \otimes \mathbf{X}^T) \cdot vec(\mathbf{\Lambda}) - \mathbf{\Sigma}_0^{-1}\boldsymbol{l}\end{aligned} \tag{3.21}$$

and the Hessian is:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \boldsymbol{l}\, \boldsymbol{l}^T} &= (\mathbf{I}_M \otimes \mathbf{X}^T) \cdot \exp\Big((\mathbf{I}_M \otimes \mathbf{X}) \cdot \boldsymbol{l} + vec(\mathbf{B})\Big) \cdot (\mathbf{I}_M \otimes \mathbf{X}) - \mathbf{\Sigma}_0^{-1} \\ &= (\mathbf{I}_M \otimes \mathbf{X}^T) \cdot vec(\mathbf{\Lambda}) \cdot (\mathbf{I}_M \otimes \mathbf{X}) - \mathbf{\Sigma}_0^{-1}.\end{aligned} \tag{3.22}$$

For $\mathbf{b}$ we again rewrite $\mathcal{L}$ as:

$$\mathcal{L} \propto tr(\mathbf{Y}^T\mathbf{X}\mathbf{L}) + tr(\mathbf{b}^T\mathbf{Y}^T\mathbf{1}_N) - tr(\mathbf{1}_{M\times N} \cdot \exp\Big(\mathbf{X}\mathbf{L} + \mathbf{1}_N \cdot \mathbf{b}^T\Big)) - \frac{1}{2}\mathbf{b}^T\mathbf{\Sigma}_0^{-1}\mathbf{b} \tag{3.23}$$

where we used the cyclical property of the trace: $tr(ABC) = tr(CAB) = tr(BCA)$ and the fact that $tr(A) = tr(A^T)$. The gradient now becomes:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{b}} &= \mathbf{Y}^T \cdot \mathbf{1}_N - \left(\exp\Big(\mathbf{X}\mathbf{L} + \mathbf{1}_N \cdot \mathbf{b}^T\Big)\right)^T \cdot \mathbf{1}_N - \mathbf{\Sigma}_0^{-1}\mathbf{b} \\ &= (\mathbf{Y}^T - \mathbf{\Lambda}^T) \cdot \mathbf{1}_N - \mathbf{\Sigma}_0^{-1}\mathbf{b}\end{aligned} \tag{3.24}$$

and the Hessian is:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}\mathbf{b}^T} = diag(\mathbf{\Lambda}^T \cdot \mathbf{1}_N) - \mathbf{\Sigma}_0^{-1} \tag{3.25}$$

Lastly, we derive the gradient update for the length scale ρ_d. The terms of $\mathcal{L}$ not constant in ρ_d are:

$$\mathcal{L} \propto -\frac{1}{2} \ln \det \mathbf{K}_d - \frac{1}{2} \mathbf{x}_d^T \mathbf{K}_d^{-1} \mathbf{x}_d \tag{3.26}$$

where we omit the absolute value of the determinant inside the logarithm, because the covariance matrix is positive semi-definite by definition (resulting in non-negative real eigenvalues and a non-negative determinant). Using the chain rule, the gradient of $\mathcal{L}$ with respect to ρ_d is:

$$\frac{\partial \mathcal{L}}{\partial \rho_d} = \frac{\partial \mathcal{L}}{\partial \mathbf{K}_d} \frac{\partial \mathbf{K}_d}{\partial \rho_d} = -\frac{1}{2} \Big(tr(\mathbf{K}_d^{-1} \frac{\partial \mathbf{K}_d}{\partial \rho_d}) + \mathbf{x}_d^T (-\mathbf{K}_d^{-1} \frac{\partial \mathbf{K}_d}{\partial \rho_d} \mathbf{K}_d^{-1}) \mathbf{x}_d \Big) \tag{3.27}$$

The gradient of $\mathbf{K}_d$ with respect to ρ_d is:

$$\frac{\partial \mathbf{K}_d}{\partial \rho_d} = \frac{\sigma^2}{\rho_d^3} \exp(-\frac{1}{2\rho_d^2} \mathbf{T}) = \rho_d^{-3} \cdot \mathbf{T} \circ \mathbf{K}_d \tag{3.28}$$

where $\circ$ is the Hadamard product, i.e. elementwise multiplication of the matrices. The final form of the gradient is thus:

$$-\frac{1}{2} \Big(tr(\rho_d^{-3} \cdot \mathbf{K}_d^{-1} (\mathbf{T} \circ \mathbf{K}_d)) - \rho_d^{-3} \cdot \mathbf{x}_d^T \mathbf{K}_d^{-1} (\mathbf{T} \circ \mathbf{K}_d) \mathbf{K}_d^{-1} \mathbf{x}_d \Big) \tag{3.29}$$

3.2.4 Expectation Maximization

We can employ an Expectation Maximization (EM) scheme to find maximum posterior (MAP) estimates of the parameters and an approximate posterior distribution of $\mathbf{X}$. To achieve this, we alternate between estimating $\mathbf{X}$ and $\mathbf{\Theta}$. If we hold the parameters $\mathbf{\Theta}$ fixed, we can maximize the latent $\mathbf{X}$, using the derived gradients. Neal and Hinton [53] provide an alternative view of EM. They show that EM alternates between maximizing $\mathcal{L}$ with respect to the model parameters, and maximizing with respect to the distribution of the latent variables:

$$\begin{aligned} &\text{E Step:} && \text{Set } q(\mathbf{X}) \text{ to the distribution that maximizes } \mathcal{L}. \\ &\text{M Step:} && \text{Set } \mathbf{\Theta} \text{ to the values that maximize } \mathcal{L}. \end{aligned} \tag{3.30}$$

In our case, since there is no closed form for the conditional distribution of the latent $\mathbf{X}$ we use a multivariate Gaussian for $q(\mathbf{X})$ (called Laplace approximation). Additionally, the terms in $\mathcal{L}$ that depend on $\boldsymbol{\rho}$ are disjoint from those that depend on the $\mathbf{L}$ and $\mathbf{b}$, so we can optimize them separately.

The E step makes use of the gradients derived in Eq. 3.18 and Hessian in Eq. 3.19. Note that the Hessian is negative definite, because for any $\mathbf{x}$ and a pre-and-post-multiplied positive diagonal matrix D we have: $\mathbf{x}^T(L^T DL)\mathbf{x} = (\mathbf{x}^T L^T)D(L\mathbf{x}) = (L\mathbf{x})^T D(L\mathbf{x}) = \mathbf{y}^T D\mathbf{y} > 0$. Thus, if the Hessian is negative definite everywhere, then $\mathcal{L}$ is strictly concave in $\mathbf{X}$ for fixed $\boldsymbol{\Theta}$. We can use any first or second-order convex optimization method to find the posterior mode, such as Newton's method:

$$\mu_{\mathbf{x}} = \mu_{\mathbf{x}} - \frac{\partial \mathcal{L}}{\partial \mathbf{x}}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{x}\mathbf{x}^T}\right)^{-1} \tag{3.31}$$

Interestingly, the same situation arises for $\mathbf{L}$ and $\mathbf{b}$ in the M step. This can be intuitively understood if we note that by fixing one of the two matrices, we are essentially fitting multivariate GLMs. The Hessians of $\mathbf{L}$ and $\mathbf{b}$ can be proven to be negative definite using the same argument and we can optimize them using Newton's method again:

$$\begin{aligned} \mu_{\mathbf{L}} &= \mu_{\mathbf{L}} - \frac{\partial \mathcal{L}}{\partial \mathbf{L}}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{L}\mathbf{L}^T}\right)^{-1} \\ \mu_{\mathbf{b}} &= \mu_{\mathbf{b}} - \frac{\partial \mathcal{L}}{\partial \mathbf{b}}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{b}\mathbf{b}^T}\right)^{-1} \end{aligned} \tag{3.32}$$

Note that because each iteration of optimization for ρ_d requires inverting the full covariance matrix, we only optimize the length scale once every several iterations. At convergence, we center a multivariate Gaussian around the modes with a covariance that is the inverse of the Hessian to obtain samples of the approximate posterior. Thus we can find a local maximum of the posterior by alternating between iteratively applying 3.30.

Chapter 4

Results

In the following datasets, we compare our Laplace approximate inference (EM) approach to: a Hamiltonian MCMC in Stan (PFALGP), mean-field ADVI in Stan (MF-ADVI), GPFA [9], and regular FA. All computation was carried out on a consumer-grade laptop. Initial values for the factor loadings were obtained using *R*'s factor analysis method `factanal`, and for the latent space were obtained by taking the first D eigenvectors of PCA.

4.1 Synthetic data

First, we generated 100 data points with 10 features that come from 3 underlying latent factors, such that the first factor is a linear combination of the first 4 features, the second factor of the next 3 features, and the last factor from the last 3 features. The three factors can be seen in the first column of Fig. 4.1. The first factor has the lowest length scale, whereas the last factor has the highest, as is evident from the figure.

We performed 10-fold cross-validation to obtain the root mean squared error (RMSE) of each method, shown in Table 4.1. We also report the computation time, as it varied considerably. We see that the approximate expectation maximization works as well as the fully Bayesian version implemented in Stan. MF-ADVI performs considerably worse (one possibility is the mean-

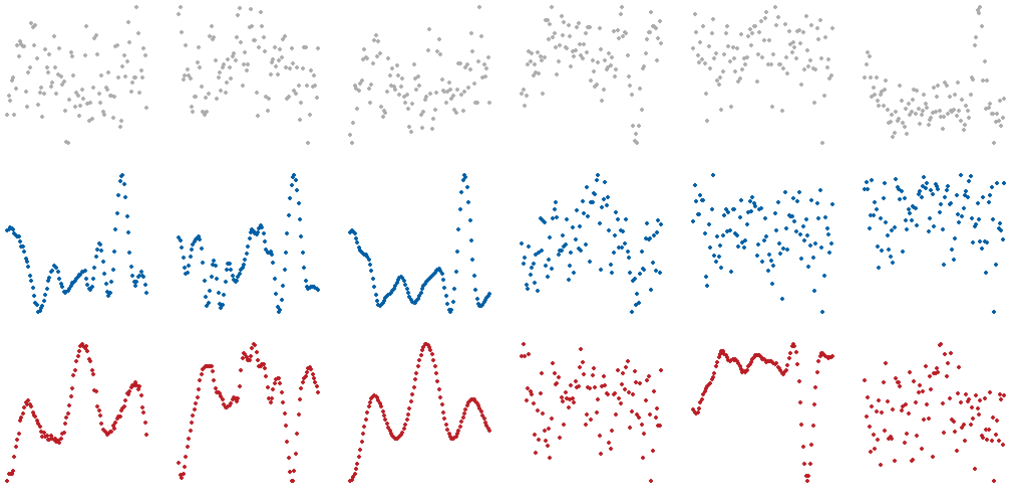

Figure 4.1: Synthetic data latent space, where each row corresponds to one dimension. From left to right: ground truth, EM, PFALGP, MF-ADVI, GPFA, FA.

model	PFALGP	EM	MF-ADVI	GPFA	FA
RMSE	1.0132	1.085	1.5304	1.7246	1.8298
time[s]	326	6	40	341	1

Table 4.1: RMSE and time complexity of each method on synthetic data.

field assumption), but still better than GPFA and FA. As expected, FA finds no time-structure, whereas GPFA struggles with the Poisson observations. The same conclusion can be reached by looking at the latent dimensions in Fig. 4.1. In the figure, we can see our approach correctly finds the latent space, whereas other methods struggle.

Lastly, Fig. 4.2 shows the log posterior probability as a function of time, or number of iterations until convergence. We can see that it quickly reaches a good value (note the logarithmic x axis) and then more slowly converges to the maximum.

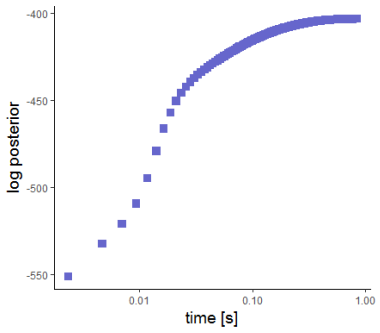

Figure 4.2: Value of the log posterior probability as a function of time during one optimization of Laplace EM.

4.2 Real world data

In the following section, to illustrate our method on real-world data, we present an analysis of three real-world data sets: *twitter* time-series data, *NBA* season, and *MNIST* digits.

4.2.1 Twitter

Using the `twitteR` package in `R`, which employs the twitter API, we collected all tweets from January 12, 2017 to January 20, 2017 that contain one of these six phrases: *gambia*, *senegal*, *yahya jammeh*, *adama barrow*, *basketball*, *nba*. We used `twitter`'s geographical information to only consider tweets in a 100 kilometer radius of the general New York Area, so that the language and timezone are consistent across the data.

Hourly frequency of each of the six phrases in that week can be seen in Fig. 4.3. We can see that there is a very distinct daily repeating pattern for common words like *basketball*, and *nba*. On the other hand, a recent transfer of power in Gambia (a small African country nestled inside of Senegal) between the then-president Yahya Jammeh and the president-elect Adama Barrow resulted in spikes of the other four phrases around January 19. Setting $D = 2$, we try to find the underlying latent space corresponding to

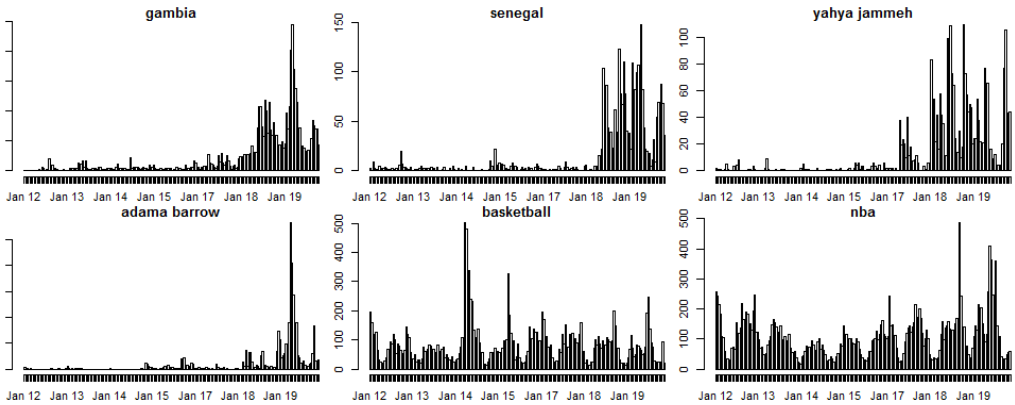

Figure 4.3: Histogram of the number of tweets per hour for six phrases in the week of January 12-18, 2017.

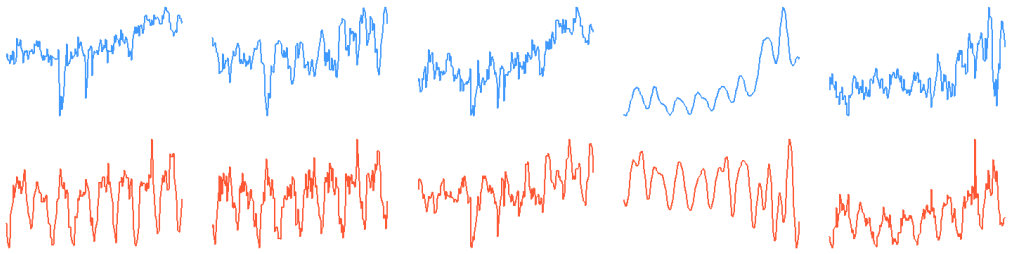

Figure 4.4: Twitter data latent space, where each row corresponds to one dimension. From left to right: ground truth, EM, PFALGP, MF-ADVI, GPFA, FA. The x axis denotes time.

the Gambia event, and the baseline daily trend. The results can be seen in Fig. 4.4. What we should be ideally seeing: one dimension with constant intensity through time, but with daily fluctuations, and one dimension with no fluctuations but a spike near the end (corresponding to the event). It seems that some methods are close to the desired outcome, except the "event" dimension is more noisy than expected.

Additionally, we perform the same quantitative analysis from the previous section, except that we use a rolling window approach instead of cross-validation, since this is time-series data. This means that we test on folds

model	PFALGP	EM	MF-ADVI	GPFA	FA
RMSE	17.70	18.84	21.9	22.28	31.18
time[s]	341	8	64	452	7

Table 4.2: RMSE and time complexity of each method on *twitter* data.

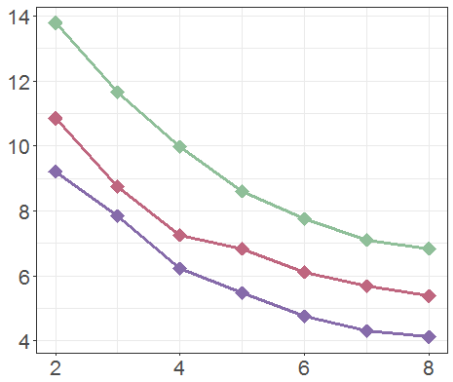

Figure 4.5: RMSE as a function of the number of factors D from 2 to 8. Green, red, and purple denote FA, GPFA, and PFALGP, respectively.

2-10, and we train only on previous folds. The results, shown in Fig. 4.2, are once again as expected, with Poisson likelihood methods achieving lower error than real-valued approaches.

4.2.2 Basketball

The 2014 *NBA* basketball season consists of $N = 1311$ games (including the playoffs). For each game we have $M = 10$ count data features for both teams, which include the number of 2-points attempted (2PA), 2-points made (2PM), 3-points attempted (3PA), 3-points made (3PM), free throws attempted (FTA), free throws made (FTM), turnovers (TOV), defensive rebounds (DRB), offensive rebounds (ORB), and fouls (FOU).

Before we present our analysis, we must specify the exact structure of the model for this data. In contrast to the previous data sets, in basketball we have match between teams, and thus we have two sets of counts of the

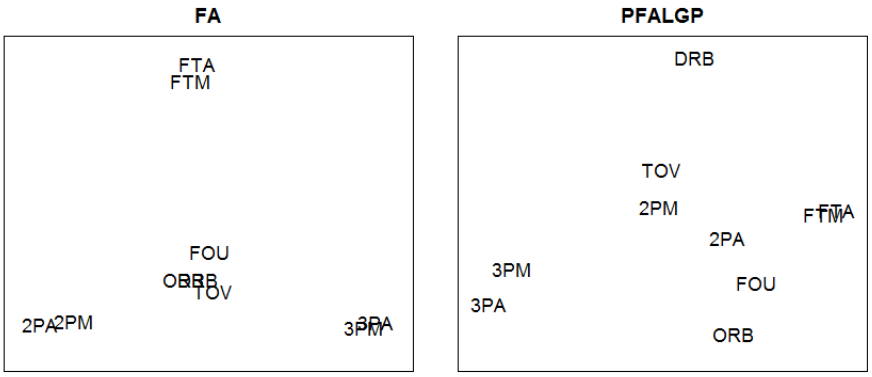

Figure 4.6: Visualization of the factor loadings for $D = 2$.

aforementioned variables for each observation. Our approach was to split the data by team, so that we have N_{team} multivariate count matrices, where N_{team} is the number of teams in the season. We can think of it as training N_{team} separate PFALGP models, except we tie the matches together by a common factor loading matrix. This is because we want the latent dimensions of each team to map in the same way back to the count variables. We present the reconstruction error of each method as a function of the number of factors in Fig. 4.5. As expected, with more factors we can better approximate the counts and separate the factors. This can also be understood from a matrix algebra perspective, as D is an upper bound of the rank of the matrix Y, and increasing D allows for the approximation to span a larger subspace of the M-dimensional matrix Y.

We can compare the closeness of each factor to all the others in Fig. 4.6. Regular FA separates some pairs of factors well, such as two-point, three-point, and free throws made/attempted. PFALGP, on the other hand, brings closer some factors which are not obviously connected. At first glance, the closeness of two points attempted to the number of fouls seems strange. However, this can be confirmed using domain knowledge. Free throws given to a team are a direct consequence of a foul on the other team. Thus,

it is in fact expected that the number of fouls would be correlated to the number of free throws attempted (given). In addition, we have the number of offensive rebounds close to the number of fouls. This too, can be explained, by noting that offensive rebounds commonly happen after the team misses their (usually) second free throw, and is given a chance to take possession of the ball again. Thus, with PFALGP, we glean more insight into the data than with regular FA.

4.2.3 MNIST

Lastly, we present a slightly different qualitative comparison and analysis of our method. Although we have so far specified Gaussian Processes as a smoothness operation in time, it is not necessarily their only function. As time-series impose one-dimensional constraints on the latent space, we can similarly impose two-dimensional constraints on the latent space. An application of 2-D constraints can be seen in images, where we want neighboring pixels to be correlated. This can easily be incorporated into the GP covariance kernel by swapping the temporal difference for a difference in 2-D coordinate space.

To illustrate this approach, we presented a small-scale analysis of MNIST [54], the Mixed National Institute of Standards and Technology data set of handwritten digits. It contains several thousand 28-by-28 pixel images of digits and is one of the most used benchmarks in computer vision. A small sample of these images can be seen in Fig. 4.7.

To showcase our method on this data set, we transform the image into one long vector so that our data set Y consists of M images that take the role of features and each pixel is one observation, so $N = 28 \cdot 28 = 784$. Furthermore, we binarize the pixel values to 0-1 as the image histograms are distinctly bi-modal at 0 and 255, which should not be modeled as Poisson.

We take 10 images for each of the digits $0, 1, 2, 3, 4$. Because we have to fit 50 784-dimensional Gaussian Processes, we employ MF-ADVI for each method instead of complete MCMC inference, which was intractable on this

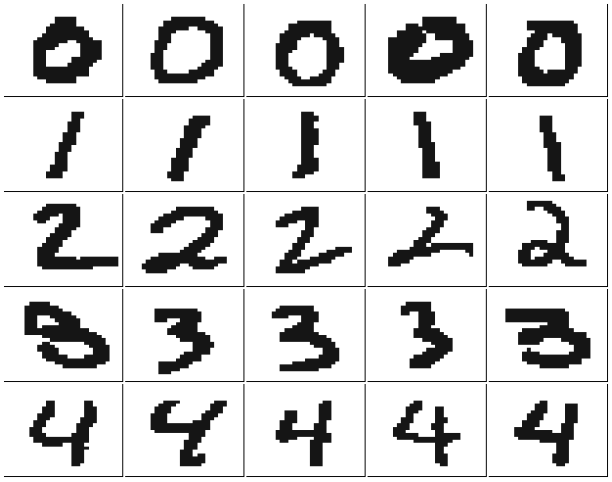

Figure 4.7: A few examples of 0-4 digits from MNIST used in our evaluation.

data set. We also employ Laplace EM for PFALGP, as it is another tractable alternative. By setting $D = 5$ our goal is to find 5 factors in this data set that correspond to each of the digits. For visual "ground truth" we take an average of all the images of that digit.

The results of each method is shown in Fig. 4.8. We can see that the Laplace EM approximation is by far the best in terms of corresponding to the each digit. The other methods, which employ the mean field assumption perform considerably worse, and as a consequence their subspace is not nearly as well-separated.

Lastly, we can also visualize the factor loadings of each method. Since it is a 5-dimensional space, we use multidimensional scaling to project it into 2-D, which we show in Fig. 4.9. Surprisingly, all methods are quite successful in separating most digits, judging by a quick visual inspection of the 2-D plots.

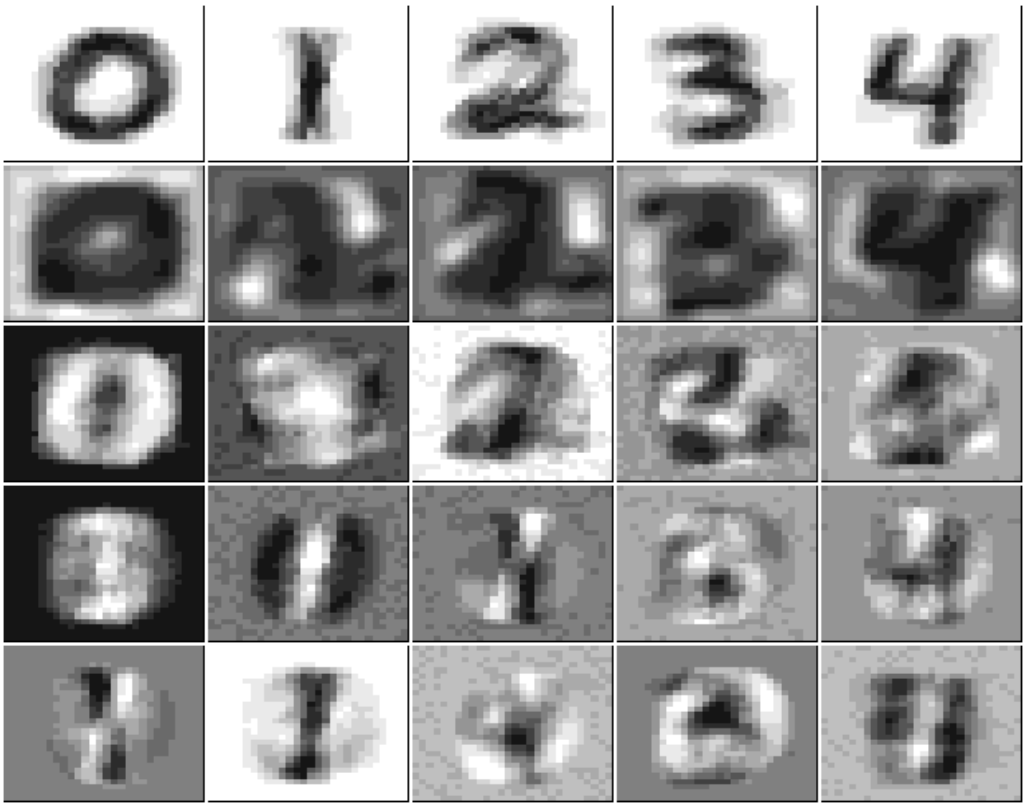

Figure 4.8: Visualization of the latent space, where each column corresponds to one latent dimension. From top to bottom: ground truth, EM, MF-ADVI PFALGP, MF-ADVI GPFA, FA.

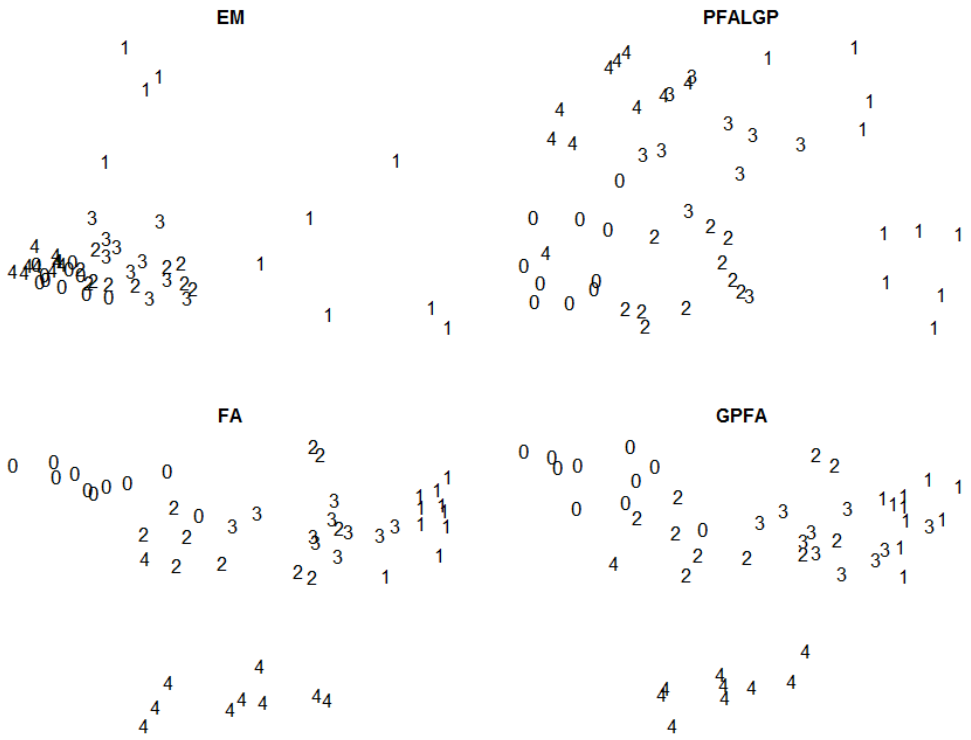

Figure 4.9: Visualization of the factor loadings in 2-D space using multidimensional scaling. Clockwise from top-left: EM, MF-ADVI PFALGP, MF-ADVI GPFA, FA.

Chapter 5

Discussion

We see that PFALGP and the Laplace EM approximation perform the best across all datasets. The reason MF-ADVI performs worse is probably due to the too stringent independence assumptions of mean field, since the latent space is correlated across time and non-i.i.d. Therefore, we see that it performs much better on data sets where the time-series are not as correlated or the length scale is smaller, so that values are not completely determined by its neighbors. Furthermore, it is expected that the models that explicitly model a Poisson likelihood will have better RMSE than models that can predict real values which never occur in our observations (such as GPFA and FA).

We have thus presented a general approach to Poisson factor analysis, although our model is used for time-series multivariate count data. Furthermore, since a diagonal GP kernel yields an i.i.d. latent space, our approach generalizes i.i.d. multivariate counts. It improves on GPFA [9] by mapping real-valued outputs through an exponential function that becomes the rate of a Poisson variable. Although more mathematically complex, it results in non-negative integer predictions that correspond to the actual observations, as opposed to real-valued outputs from GPFA.

Since the added complexity arises in a lack of closed-form integral, we developed a fast inference method using Expectation Maximization and the

Laplace approximation, which uses the gradient and Hessian of the log of the posterior. Consequently, we were able to fit dozens of Gaussian Processes simultaneously in minutes, each with close to a thousand points. We presented our approach on several data sets, both synthetic and real, and show it outperformed its real-valued counterpart in accuracy and speed. Thus, our method can become a standard for factor analysis of dynamic count matrices, or any kind of multivariate count data with non-i.i.d. structure in both its observations and latent space.

Several things can be improved in future work. The number of latent dimensions D is a parameter that has to be manually set. Except in cases where we know the underlying factors there is no clear answer as to which value is optimal - too few dimensions and the predictive power is too low, too many and there is needless computation added. Promising work in Bayesian nonparametrics may allow for marginalization over this number. In addition, we have only experimented with the squared exponential kernel. It is possible that other, possibly non-stationary, kernels could prove to be superior, or at least attain equivalent results for a lower computational cost. Similarly, we could use sparse Gaussian Processes to speed-up inference since we are already approximating the posterior. Alternatively, we can possibly speed up inference by using a low-rank approximation of the covariance matrix.

Bibliography

[1] Yarin Gal, Yutian Chen, and Zoubin Ghahramani. Latent gaussian processes for distribution estimation of multivariate categorical data. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 645–654, 2015.

[2] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In S. Thrun, L. K. Saul, and P. B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 329–336. MIT Press, 2004.

[3] Siddhartha Chib and Rainer Winkelmann. Markov chain monte carlo analysis of correlated count data. *Journal of Business & Economic Statistics*, 19(4):428–435, 2001.

[4] Dimitris Karlis and Loukia Meligkotsidou. Multivariate poisson regression with covariance structure. *Statistics and Computing*, 15(4):255–265, 2005.

[5] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[6] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

[7] Derrick Norman Lawley and Albert Ernest Maxwell. *Factor analysis as a statistical method*, volume 18. JSTOR, 1971.

[8] Mingyuan Zhou, Lauren Hannah, David B Dunson, and Lawrence Carin. Beta-negative binomial process and poisson factor analysis. In *AISTATS*, volume 22, pages 1462–1471, 2012.

[9] Byron M Yu, John P Cunningham, Gopal Santhanam, Stephen I. Ryu, Krishna V Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1881–1888. Curran Associates, Inc., 2009.

[10] Eun Park and Dominique Lord. Multivariate poisson-lognormal models for jointly modeling crash frequency by severity. *Transportation Research Record: Journal of the Transportation Research Board*, (2019):1–6, 2007.

[11] Peng Shi and Emiliano A Valdez. Multivariate negative binomial models for insurance claim counts. *Insurance: Mathematics and Economics*, 55:18–29, 2014.

[12] Aristidis K Nikoloulopoulos and Dimitris Karlis. Modeling multivariate count data using copulas. *Communications in Statistics-Simulation and Computation*, 39(1):172–187, 2009.

[13] Esther Hee Lee. Copula analysis of correlated counts. In *Bayesian Model Comparison*, pages 325–348. Emerald Group Publishing Limited, 2014.

[14] Nicole Bäuerle and Rudolf Grübel. Multivariate counting processes: copulas and beyond. *Astin Bulletin*, 35(02):379–408, 2005.

[15] Michel Wedel, Ulf Böckenholt, and Wagner A Kamakura. Factor models for multivariate count data. *Journal of Multivariate Analysis*, 87(2):356–369, 2003.

[16] Michalis K Titsias. The infinite gamma-poisson feature model. In *Advances in Neural Information Processing Systems*, pages 1513–1520, 2008.

[17] Wray Buntine and Aleks Jakulin. Discrete component analysis. In *Subspace, Latent Structure and Feature Selection*, pages 1–33. Springer, 2006.

[18] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[19] Ayan Acharya, Joydeep Ghosh, and Mingyuan Zhou. Nonparametric bayesian factor analysis for dynamic count matrices. In *AISTATS*, 2015.

[20] Aaron Schein, Hanna Wallach, and Mingyuan Zhou. Poisson-gamma dynamical systems. In *Advances in Neural Information Processing Systems*, pages 5006–5014, 2016.

[21] Rajesh Ranganath, Linpeng Tang, Laurent Charlin, and David M Blei. Deep exponential families. In *AISTATS*, 2015.

[22] Zhe Gan, Changyou Chen, Ricardo Henao, David E Carlson, and Lawrence Carin. Scalable deep poisson factor analysis for topic modeling. In *ICML*, pages 1823–1832, 2015.

[23] Ricardo Henao, Zhe Gan, James Lu, and Lawrence Carin. Deep poisson factor modeling. In *Advances in Neural Information Processing Systems*, pages 2800–2808, 2015.

[24] Mingyuan Zhou, Yulai Cong, and Bo Chen. The poisson gamma belief network. In *Advances in Neural Information Processing Systems*, pages 3043–3051, 2015.

[25] Sunil Kumar Gupta, Dinh Phung, and Svetha Venkatesh. A nonparametric bayesian poisson gamma model for count data. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1815–1818. IEEE, 2012.

[26] Prem Gopalan, Francisco J Ruiz, Rajesh Ranganath, and David M Blei. Bayesian nonparametric poisson factorization for recommendation systems. In *AISTATS*, pages 275–283, 2014.

[27] Hedibert Freitas Lopes and Mike West. Bayesian model assessment in factor analysis. *Statistica Sinica*, pages 41–67, 2004.

[28] Aaron Schein, John Paisley, David M Blei, and Hanna Wallach. Bayesian poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1045–1054. ACM, 2015.

[29] Prem K Gopalan, Laurent Charlin, and David Blei. Content-based recommendations with poisson factorization. In *Advances in Neural Information Processing Systems*, pages 3176–3184, 2014.

[30] John Geweke and Guofu Zhou. Measuring the pricing error of the arbitrage pricing theory. *Review of Financial Studies*, 9(2):557–587, 1996.

[31] Omar Aguilar and Mike West. Bayesian dynamic factor models and portfolio allocation. *Journal of Business & Economic Statistics*, 18(3):338–357, 2000.

[32] Robert C Jung, Roman Liesenfeld, and Jean-François Richard. Dynamic factor models for multivariate count data: An application to stock-market trading activity. *Journal of Business & Economic Statistics*, 29(1):73–85, 2011.

[33] John P Cunningham, M Yu Byron, Krishna V Shenoy, and Maneesh Sahani. Inferring neural firing rates from spike trains using gaussian processes. In *NIPS*, pages 329–336, 2007.

[34] Biljana Petreska, M Yu Byron, John P Cunningham, Gopal Santhanam, Stephen I Ryu, Krishna V Shenoy, and Maneesh Sahani. Dynamical

segmentation of single trials from population neural data. In *Advances in neural information processing systems*, pages 756–764, 2011.

[35] Evan W Archer, Urs Koster, Jonathan W Pillow, and Jakob H Macke. Low-dimensional models of neural population activity in sensory cortical circuits. In *Advances in Neural Information Processing Systems*, pages 343–351, 2014.

[36] Joao Semedo, Amin Zandvakili, Adam Kohn, Christian K Machens, and Byron M Yu. Extracting latent structure from multiple interacting neural populations. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2942–2950. Curran Associates, Inc., 2014.

[37] Mijung Park, Gergo Bohner, and Jakob H Macke. Unlocking neural population non-stationarities using hierarchical dynamics models. In *Advances in Neural Information Processing Systems*, pages 145–153, 2015.

[38] David R Cox. Some statistical methods connected with series of events. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 129–164, 1955.

[39] Ryan Prescott Adams, Iain Murray, and David JC MacKay. 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, 2009.

[40] Tom Gunter, Chris Lloyd, Michael A. Osborne, and Stephen J. Roberts. Efficient bayesian nonparametric modelling of structured point processes. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, pages 310–319, Arlington, Virginia, United States, 2914. AUAI Press.

[41] Chris M Lloyd, Tom Gunter, Michael A Osborne, and Stephen J Roberts. Variational inference for gaussian process modulated poisson processes. In *ICML*, pages 1814–1822, 2015.

[42] Andrew Miller, Luke Bornn, Ryan P Adams, and Kirk Goldsberry. Factorized point process intensities: A spatial analysis of professional basketball. In *ICML*, pages 235–243, 2014.

[43] Chris Lloyd, Tom Gunter, Tom Nickson, Michael A Osborne, and Stephen J Roberts. Latent poisson process allocation.

[44] Christian Aßmann, Jens Boysen-Hogrefe, and Markus Pape. The directional identification problem in bayesian factor analysis: An expost approach. Technical report, Economics Working Paper, Christian-Albrechts-Universität Kiel, Department of Economics, 2012.

[45] Gabriella Conti, Sylvia Frühwirth-Schnatter, James J Heckman, and Rémi Piatek. Bayesian exploratory factor analysis. *Journal of econometrics*, 183(1):31–57, 2014.

[46] Bob Carpenter, Andrew Gelman, Matt Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Michael A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 20, 2016.

[47] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[48] Matthew D Hoffman and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

[49] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*, 2016.

[50] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *arXiv preprint arXiv:1603.00788*, 2016.

[51] Rajesh Ranganath, Sean Gerrish, and David M Blei. Black box variational inference. In *AISTATS*, pages 814–822, 2014.

[52] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[53] Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[54] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998.

Appendix A

Stan Model Code

A.1 Poisson Factor Analysis Using Latent Gaussian Processes

```
data {
  int N;  // number of observations
  int M;  // number of features
  int D;  // number of latent factors
  int S;  // dimensions (S=1 time series, S=2 images)
  int Y[N, M]; // observations
  vector[S] t[N]; // structure
}

parameters {
  vector<lower=0>[D] lenscale;  // GP parameter
  vector[M] base;    // mean vector
  matrix[N, D] X;    // latent space
  matrix[D, M] L;    // factor loadings
}

model {
  matrix[N, M] rates;
  matrix[N, N] Cov[D];
  real sigma;
```

```
  sigma = 1;
  lenscale ~ normal(0, 1);

  for (d in 1:D) {
    Cov[d] = cov_exp_quad(t, sigma, lenscale[d]) +
             diag_matrix(rep_vector(0.001, N));
    X[, d] ~ multi_normal(rep_vector(0, N), Cov[d]);
    L[d, ] ~ normal(0, 1);
  }

  rates = exp(X * L + rep_vector(1, N) * base');
  for (m in 1:M) {
    Y[, m] ~ poisson(rates[, m]);
  }
}
```

For the more complicated basketball case where we have dozens of teams and many teams played a different number of games, the following model was used:

```
data {
  int Nmax;       // maximum number of games played
  int Nteams;     // number of teams
  int Nhome[Nteams];   // number of home games of each team
  int Naway[Nteams];   // number of away games of each team
  int M;     // number of features
  int D;     // number of latent factors
  int Y[Nteams, Nmax, M];   // observations
  real time[Nteams, Nmax]; // dates of each played game
}

transformed data {
  int Ngames[Nteams];
  for (nt in 1:Nteams) Ngames[nt] = Nhome[nt] + Naway[nt];
}
```

```
parameters {
  vector<lower=0>[M] baserate;  // mean vector
  vector[M] home_adv;  // advantage of home team
  positive_ordered[D] lenscale; // for identifiability
  matrix[Nmax, D] X[Nteams];
  matrix[D, M] L; // factor loadings
}

model {
  matrix[Nmax, Nmax] Cov[D, Nteams];
  real sigma;
  int N;
  matrix[Nmax, M] lmbd;

  sigma = 1;

  baserate ~ normal(0, 5);
  home_adv ~ normal(0, 5);
  lenscale ~ normal(0, 5);

  Lnnz ~ cauchy(0, 5);
  Ld ~ cauchy(0, 5);

  for (nt in 1:Nteams) {
    N = Ngames[nt];
    for (d in 1:D) {
      Cov[d,nt,1:N,1:N] =
          cov_exp_quad(time[nt, 1:N], sigma,
                       lenscale[d]) +
                       diag_matrix(rep_vector(0.001, N));
      X[nt, 1:N, d] ~ multi_normal(rep_vector(0, N),
                                   Cov[d, nt, 1:N, 1:N]);
```

```
      X[nt, (Ngames[nt]+1):Nmax, d] ~ normal(0, 0.001);
    }

    lmbd[1:N, ] = exp(X[nt, 1:N, ] * L +
                      rep_vector(1, N) * baserate' +
                      append_row(rep_vector(1, Nhome[nt]),
                      rep_vector(0, Naway[nt]))*home_adv');

    for (m in 1:M) Y[nt, 1:N, m] ~ poisson(lmbd[1:N, m]);
  }
}
```

A.2 Comparison Models

We compared our method with GPFA and FA, also coded in Stan, for reproducibility purposes and ease of comparison.

A.2.1 Gaussian Process Factor Analysis

```
data {
  int N;  // number of observations
  int Q;  // number of features
  int P;  // number of latent dimensions
  int S;  // dimensions (S=1 time series, S=2 images)
  real Y[N, Q]; // (square rooted) observations
  vector[S] t[N]; // structure
}

parameters {
  matrix[N, P] X;    // GP latent space
  matrix[P, Q] C;    // factor loadings
  vector[Q] d;       // mean vector
  vector<lower=0>[P] tau;     // length scale
  vector<lower=0>[Q] R_diag; // diagonal variance
```

```
}

model {
  matrix[N, N] Cov[P];
  matrix[N, Q] rates;
  real sigma;

  sigma = 1;
  d ~ normal(0, 5);
  tau ~ normal(0, 5);
  R_diag ~ normal(0, 5);

  for (p in 1:P) {
    Cov[p] = cov_exp_quad(t, sigma, tau[p]) +
             diag_matrix(rep_vector(0.001, N));
    X[, p] ~ multi_normal(rep_vector(0, N), Cov[p]);
    C[p, ] ~ normal(0, 5);
  }

  rates = X * C + rep_vector(1, N) * d';
  for (n in 1:N)
    Y[n, ] ~ normal(rates[n, ], R_diag);
}
```

A.2.2 Factor Analysis

```
data {
  int N;  // number of observations
  int M;  // number of features
  int D;  // number of latent factors
  real Y[N, M]; // observations
}

parameters {
```

```
  matrix[N, D] X;          // latent space
  matrix[D, M] Lambda; // factor loadings
  row_vector[M] mu;        // mean vector
  row_vector<lower=0>[M] phi; // diagonal variances
}

model {
  for (n in 1:N) {
    X[n, ] ~ normal(0, 1);
    Y[n, ] ~ normal(mu + X[n, ] * Lambda, phi);
  }
}
```