Dirichlet Process Mixtures with Sparse Multivariate Poisson with Applications to Mining I/O Traces for Cache Preloading

LAVANYA SITA TEKUMALLA, Department of CSA, Indian Institute of Science
CHIRANJIB BHATTACHARYYA, Department of CSA, Indian Institute of Science

Cache Preloading involves proactively loading data into cache to improve application latency. However, state-of-the-art caching strategies, are limited to exploiting short range correlations in memory accesses spanning milli-seconds. We investigate for the first time, techniques to analyze *long range correlations* spanning minutes or hours in I/O traces for preloading, opening new avenues for data mining algorithms that are less stringently bound by timing constraints. Our strategy involves spatio-temporal aggregation of data to get a coarser view, transforming the trace into a temporal sequence highly correlated count vectors.

Our first contribution is a novel Bayesian non-parametric modeling technique for predictive modeling of such correlated count vectors for cache preloading based on Dirichlet Process(DP) Mixture models for multivariate count data. While DP mixture models are well explored, there is no work that explores such techniques for multivariate count data. We addresses this gap by proposing DP based mixture model of Multivariate-Poisson(DP-MVP) and its temporal extension(TDP-MVP) that captures the full covariance structure of multivariate count data. However, modeling full covariance structure for count vectors with the Multivariate-Poisson leads to overfitting and added inference complexity, particularly for high-dimensional data. Hence, as our second contribution, we propose the *Sparse-Multivariate-Poisson(SMVP)* distribution to model the full covariance of only a subset of active components exploiting sparsity in multivariate count data. We further explore priors to model sparsity based on the inherent correlation structure of the dataset, and propose the Correlated-Sparse-Multivariate Poisson(CSMVP) and the Independant-Sparse-Multivariate Poisson(ISMVP). We introduce DP-mixture extensions(DP-CSMVP, DP-ISMVP) of our models and temporal non-parametric extensions(TDP-CSMVP, TDP-ISMVP) to our models for cache preloading. While our models have wide applicability outside the caching domain, CSMVP is particularly conducive for trace modeling. As our final contribution, we outline a cache preloading strategy using our models and take the first step towards mining historical data, to capture long range patterns in traces for cache preloading. Experimentally, we show a dramatic improvement in hitrates on benchmark traces and lay the groundwork for further research in storage domain to reduce latencies using data mining techniques for trace modeling.

Categories and Subject Descriptors: [**Mathematics of computing**]: Probability and statistics

General Terms: Sparse Multivariate Poisson, Bayesian Non-parametric modeling, Cache Preloading

Additional Key Words and Phrases: Temporal Models, Dirichlet Process Mixture Models

1. INTRODUCTION

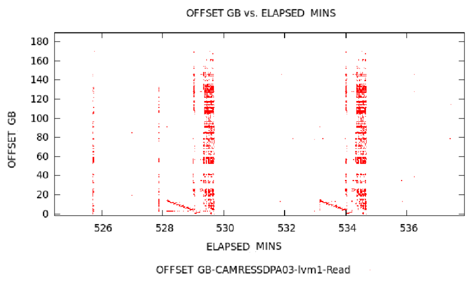
Caching is the most important determinant of a storage system's performance. *Cache Preloading* is the process of proactively loading data into cache to reduce storage system latency. We explore Bayesian non-parametric models for sparse multivariate count data to mine block I/O traces for *Cache Preloading*.

State of the art caching strategies in the storage domain are suited for a limited set of workloads where nearby memory accesses are correlated in extremely short inter-

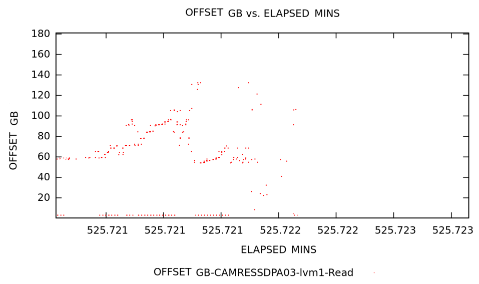
Author's addresses: Lavanya Sita Tekumalla and Chiranjib Bhattacharyya, Dept of CSA, Indian Institute of Science, Bangalore, India;

DOI: http://dx.doi.org/10.1145/0000000.0000000

vals of time, typically in milli-secs. Often in real workloads, we find correlated memory accesses spanning long intervals of time, in the order of minutes and hours. For example, figure 1(a) illustrates the visualization of a trace, where X-axis is time and Y-axis represents the memory block accessed. We note correlations arising from repetition of long-range access patterns in the granularity of minutes. However, when we zoom into the image, and observe a few milli-seconds of data to observe block level accesses in figure 1(b), it looks seemingly random without any discernible correlations. Hence, existing caching strategies that operate over milli-sec granularity fail for such traces.



(a) LBA accesses vs time plot for MSR trace(CAMRESWEBA03-lvm2): A Zoomed out(coarser view) shows repeating access patterns over a time span of minutes.



(b) A zoom-in access level view of this trace, into one of these patterns, over a time span of $\sim$ 100 milli-secs, does not show any discernible correlation. Also note the sparsity of data; during any interval of time, only a small subset of memory locations are accessed

There has been no prior work for cache preloading to learn such long range access patterns for predicting future accesses by analyzing trace data . In this work, we attempt to leverage such long range dependencies that span minutes or even hours for prediction. Such an approach of leveraging the longer range dependencies is also not bound by time constraints as stringent as the traditional caching approaches (that work in the milli-second range). This immediately opens up avenues to use a wide variety of data mining techniques that are not as strictly constrained by the milli-second range timing requirements. Motivated by this, we explore caching alternatives for analyzing the trace to capture such long-range spatio-temporal correlations using novel predictive models based on Bayesian Non-parametric modeling techniques for multivariate count data.

We approach this problem of capturing the longer range dependencies in trace data through *data aggregation*, by spatially and temporally aggregating the trace data. Our aggregation process converts the trace into a temporal sequence of correlated count vectors. This aggregation process is described in more detail in section 8.3. The components of count vector instances after aggregation are often correlated within each instance. This leads to a rich covariance structure. Further, due to the long range temporal dependencies in access patterns (fig 1(a)), the sequence of aggregated count vectors are also temporally correlated. Count vectors thus obtained by aggregating over time and space are also *sparse*, since only a small portion of memory is accessed in any time interval (fig 2). Hence a small subset of count vector dimensions have significant non zero values. An important challenge lies in building appropriate predictive models for such sparse correlated count vector sequences to assist cache preloading.

A common technique for modeling temporal correlations are Hidden Markov Models(HMMs) which are mixture model extensions for temporal data. Owing to high variability in access patterns inherent in storage traces across workloads, finite mixture models do not suffice for the caching application since the number of mixture components varies often based on the type of workload being modeled and the kind of access patterns. Bayesian Non-parametric modeling(BNP) techniques address this issue by automatically adjusting the number of mixture components based on complexity of data. Non-parametric clustering with Dirichlet Process(DP) mixtures and their temporal variants have been extensively studied over the past decade [Teh 2010], [Teh et al. 2004]. However, to the best of our knowledge we are not aware of such models in the context of multivariate count data, particularly in a temporal setting, furthermore for sparse data.

1.1. Challenges in Modeling Sparse multivariate count data:

Poisson distribution is a natural prior for counts and the Multivariate Poisson(MVP) for correlated count vectors. However, owing to the structure of multivariate Poisson and its computational intractability [Tsiamyrtzis and Karlis 2004] non parametric mixture modeling for multivariate count vectors has received less attention. Hence, we first bridge this gap by paralleling the development of *DP based non-parametric mixture models with efficient inference algorithms for multivariate count data* along the lines of those for Gaussians and multinomials.

However, modeling the full covariance structure using the MVP is still computationally expensive and often leads to a suboptimal fit of the data due to the large number of latent variables in its definition. While significant reduction in the number of parameters, that also translates into significant improvement in efficiency can be realized by modeling the sparsity, there has been no prior work that models sparsity in multivariate count data.

Finally, there has been no work that explores temporal mixtures models for multivariate count data that can be used for predictive modeling in applications such as cache preloading. Further, from our application perspective, there has been no prior work on statistical models for cache preloading based on analyzing long range-correlations in data based on data aggregation. Preloading algorithms based on analyzing such longer-range dependencies are not as stringently bound by the timing constraints as are traditional caching techniques based on sub-milli-second granularity. This enables us to explore a rich set of statistical models suitable for the cache preloading task. We also hope our preloading strategy based on analyzing longer-range dependencies spawns further interdisciplinary research of machine learning techniques for the caching domain.

1.2. Contributions

— We propose Dirichlet process mixture models for Multivariate count data using the Multivariate Poisson(MVP) and propose the DP based non-parametric mixture of MVP (DP-MVP). Inference with MVP distribution is hard due to complex dependencies in latent variables in its definition. We propose an efficient inference algorithm for the MVP based on collapsed Gibbs sampling.

— We exploit the sparsity in data, and propose the sparse multivariate Poisson distribution (SMVP) that captures the covariance structure for a subset of active dimensions. We are not aware of any prior work that addresses modeling sparsity for multivariate count data. We extend the DP-MVP for non parametric clustering of sparse high dimensional count vectors with the sparse DP-mixture of Multivariate Poisson (DP-SMVP). This methodology not only leads to a better fit for sparse multidimensional count data but is also computationally more tractable than modeling full covariance when there exists sparsity in data.

— Multivariate count data obtained from memory access traces exhibit a inherent spatial correlation, where adjacent dimensions are likely to be active together. Hence we explore sparsity inducing priors exploiting the spatial correlation in traces for selecting the active set of dimensions in such sparse models. We propose two variants, the Independent-Sparse-MVP(ISMVP) and the Correlated-Sparse-MVP (CSMVP) that uses a beta-Bernoulli with a log-normal prior for its parameters to select the active feature set. We find the CSMVP particularly conducive for trace modeling in incorporating the rich correlation structure arising from locality of space in traces in inducing sparsity. We further explore DP-mixture extensions of our Sparse Multivariate Poisson models.

— We explore temporal extensions of MVP based non-parametric mixtures for predictive modeling of multivariate count data, a previously unexplored topic. (We propose the TDP-MVP, TDP-ISMVP and TDP-CSMVP models). Our temporal models can be used for cache preloading.

— As our final contribution, in this inter-disciplinary work, we take the first steps in outlining a framework for cache preloading to capture long range spatio-temporal dependencies in memory accesses. We perform spatio-temporal aggregation of data to get a higher level view and outline a prediction strategy based on our MVP based models for Cache Preloading.

We evaluate the relative performance of our MVP based models and their DP-mixture extensions on synthetic data. We evaluate our temporal extensions that lead to predictive models for cache preloading by performing experiments on real-world benchmark traces showing dramatic hitrate improvements. In particular, for the trace in Fig 1(a), our preloading yielded a 0.498 hitrate over 0.001 of baseline (without preloading), a 498X improvement (trace MT2: Tab VII).

2. RELATED WORK

In this section, we review related work on Bayesian Non-parametric models for multivariate count data. We note that related work for the cache preloading problem is reviewed later in , where we describe our caching strategy.

Poisson distribution is a natural prior for count data. But the multivariate Poisson(MVP) [Kawamura 1979] has seen limited use due to its computational intractability due to the complicated form of the joint probability function[Tsiamyrtzis and Karlis 2004]. There has been relatively little work on MVP mixtures [Karlis and Meligkotsidou 2007; Meligkotsidou 2007; Schmidt and Rodriguez 2010]. On the important problem of designing MVP mixtures with an unknown number of components, [Meligkotsidou 2007] is the only reference we are aware of. The authors explore MVP mixture with an unknown number of components using a truncated Poisson prior for the number of mixture components, and perform uncollapsed RJMCMC inference. DP based models are a natural truncation free alternative that are well studied and amenable to hierarchical extensions [Teh et al. 2004; Fox et al. 2011] for temporal modeling which are of immediate interest to the caching problem. To the best of our knowledge, there has been no work that examines a truncation free non-parametric approach with DP-based mixture modeling for MVP.

Another modeling aspect we address is the sparsity of data. Often, data encountered in real world applications is sparse. For instance, in a document modeling setting, where a document is represented as a count vector of word counts, a small subset of related words are likely to be used in the document leading to sparse correlated count vectors. Similarly, for the caching application, only a small portion of memory is likely to be accessed in each time interval, leading to a sparse multivariate count data on aggregation. There has been some work on sparse mixture models for Gaussian and multinomial densities [Law et al. 2002] [Wang]. However they are specialized to the individual distributions and do not apply for sparse multivariate counts. Feature selection has further been explored in the infinite feature setting in models based on the Indian Buffet Process [Griffiths and Ghahramani 2005; Doshi-Velez and Ghahramani. 2009], unlike our setting with a finite feature set. Our approach, for modeling sparsity, in the context of multivariate Poisson emissions, involves leveraging sparsity by modeling the full correlation of the multivariate Poisson for a subset of dimensions that are 'active'. We note that a full multivariate Poisson emission density over all dimensions for each cluster results in parameter explosion for even moderately high dimensional data, since the MVP introduces latent variables quadratic in the number of dimensions, for each data point. Hence, handling sparsity, not only leads to a better fit of data, but also significantly reduces computational overhead when modeling with the MVP.

We are not aware of any prior work on modeling sparse multivariate count data or mixture model extensions in this setting. Furthermore, while modeling sparsity, we explore priors to effectively select active dimensions to exploit the fact that certain dimensions tend to be active together based on the inherent correlation in the data, for instance, spatial correlation in traces. We are not aware of any existing work that explores such priors to induce sparsity in multivariate count data. Finally, we investigate temporal models for sparse correlated multivariate count data. We are not aware of any prior work that investigates mixture models for temporal multivariate count data.

3. DIRICHLET PROCESS EXTENSIONS OF MULTIVARIATE POISSON MIXTURES

The Dirichlet Process is a prior on the mixture distribution with countably infinite mixture components, thus allowing the model to automatically adapt the number of mixture components based on the data. Dirichlet Process mixture models while extensively explored for real and multinomial data [Teh 2010; Teh et al. 2004; Fox et al. 2011], have not been explored for multivariate count data. We fill this gap by developing Bayesian non-parametric models for multivariate count data with the Multivariate Poisson(MVP) [Kawamura 1979]. This immediately leads to several open issues. The first and foremost is a suitable prior over the number of mixture components in a Mixture of MVPs which can allow efficient MCMC algorithms. We parallel the development of DP based MVP mixtures and non-parametric HMM based MVP mixture models along the lines of those for multinomial mixtures [Teh et al. 2004].

To this end, we first propose the Dirichlet Process Mixture of Multivariate Poisson (DP-MVP) in section 3.3. We next introduce the Sparse multivariate Poisson and their DP-mixture extensions in 4. Finally we introduce Bayesian non-parametric Hidden Markov Models(HMMs) with MVP based emissions to model temporal multivariate count data in section 7. Inference for these models is hard and we propose efficient inference techniques based on Gibbs Sampling.

We first discuss background related to Bayesian non parametric modeling and modeling count data with the Multivariate Poisson before we start describing our models in detail from section 3.3.

3.1. Background: Dirichlet Process Mixture Models

In this section, we review some preliminary material related to Bayesian non-parametric modeling with Dirichlet Process mixture models.

We now describe the **Dirichlet Process** prior by recalling some definitions:

$$\begin{gathered}\beta \sim GEM(\alpha) \qquad if \\ \beta_1 \sim Beta(1,\alpha); \beta_k = \eta_k \Pi_{p=1}^{k-1}(1-\eta_p), \forall k \geq 2 \\ \eta_k \sim Beta(1,\alpha), \forall k \end{gathered} \tag{1}$$

A probability distribution G is a draw from a Dirichlet Process(DP) $G \sim DP(\alpha, H)$

$$G = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k}, \beta \sim GEM(\alpha),\ \theta_k \sim H, k = 1 \ldots$$

where H is a diffused measure.

The *Hierarchical Dirichlet process*(HDP)[Teh et al. 2004] is an extension of the DP mixture model when multiple distributions $G_1, \ldots, G_J$ drawn from a DP need to share atoms. Probability measures $G_1, \ldots, G_J$ follow the HDP if

$$G_j \sim DP(\alpha, G_0), j = 1 \ldots J \text{ where } G_0 \sim DP(\gamma, H)$$

The HDP can be used as the prior for the state transition matrix of a Hidden Markov Model(HMM) leading to a powerful extension of HMM, the **HDP-HMM**, that can have a countably infinite number of states. This alleviates the need for fixing the number of states. Such an extension is useful since the choice of number of hidden states in a HMM is crucial in real world applications though there are no clear guidelines on how to choose the number of states which can work for most applications. The HDP-HMM is defined as follows, where Z_t is the cluster assignment for each data point $t \in [T]$.

$$\begin{gathered} \beta \sim GEM(\gamma) \\ \pi_{\mathbf{k}}|\beta, \alpha_k \sim DP(\alpha_k, \beta), \theta_k|H \sim H, k = 1, 2, \ldots \\ Z_t|Z_{t-1}, \pi \sim \pi_{Z_{t-1}}, X_t|Z_t \sim f_{Z_t}(\theta_k), t \in [T] \end{gathered} \quad (2)$$

We note that Z_t is sampled based on the cluster assignment Z_{t-1} due to the Markov property. For each cluster k, the transition probability vectors to the remaining clusters(states) π_k is drawn from a HDP. This permits π_k to be a distribution over a countably infinite number of clusters at the same time ensuring that multiple π_k share atoms and have common clusters to transition to. Commonly used base distributions for H include the multivariate Gaussian and the Dirichlet distributions where mixture components are Gaussians and multinomials respectively. There has been little work in exploring emissions based on models for multivariate count data.

3.2. Background : The Multivariate Poisson for modeling multivariate counts

In this section we review some background on count modeling with Multivariate Poisson. Modeling multivariate count data is an important problem that has wide applicability outside the area of caching. Count data occurs in a varied set of real world applications, like document modeling[Canny 2004], ecology[Schmidt and Rodriguez 2010], crime analysis [Dimitris Karlis 2007]. The Poisson distribution defines the probability mass on the number of events that occur within a fixed interval of time and is most commonly used to model count data. A common prior for the univariate Poisson distribution is the Gamma distribution due to its conjugacy.

Consider T samples of M dimensional count vectors of the form $X_{t,j}$, where $t \in [T]$ and $j \in [M]$. To model multivariate data the simplest conceivable extension of the univariate Poisson is to model each dimension independently, leading to the **Multivariate independent Poisson(MIP)** model with parameter $\lambda \in [0, \infty)^M$ for count vectors. Let $\bar{a}, \bar{b} > 0$.

$$X_{tj} \sim Poisson(\lambda_j), \forall t, j; \lambda_j \sim Gamma(\bar{a}, \bar{b}) \forall j \quad (3)$$

This simplistic construction in equation 3 however fails to account for correlations across dimensions of count vectors.

Multivariate Poisson : the Multivariate Poisson(MVP) was proposed by Kawamura [Kawamura 1979] to model correlated count vectors by specifying the covariance structure of the random variables, mimicking the multivariate normal distribution.

A random vector, $X_t \in \mathbf{Z}^M$ is Multivariate Poisson(MVP) distributed, denoted by $X_t \sim MVP(\Lambda)$, if

$$X_t = Y_t 1_M \text{ where } \forall j \leq l \in [M], \lambda_{l,j} = \lambda_{j,l} \sim Gamma(\bar{a}, \bar{b})$$
$$Y_{t,j,l} = Y_{t,l,j} \sim Poisson(\lambda_{j,l}) \quad (4)$$

where 1_M is a M dimensional vector of all 1s. In this construction, X is constructed as a linear combination by introducing $\binom{M}{2}$ independently drawn Poisson latent variables as $X_{t,j} = \Sigma_l Y_{t,j,l}$. It is useful to note that the mean of multivariate Poisson $E(X) = \Lambda 1_M$, where Λ is an $M \times M$ symmetric matrix with entries $\lambda_{j,l}$ as elements and $Cov(X_j, X_l) = \lambda_{j,l}$.

We note that Setting $\lambda_{j,l} = 0, j \neq l$ yields $X_i = Y_{i,i}$, leading to the Multivariate Independent Poisson model in equation 3. Hence the MIP model is a special case of the Multivariate Poisson in equation 3.2.

Multivariate Poisson Mixtures A finite mixture of multivariate Poisson [Dimitris Karlis 2007] is a natural extension of the multivariate Poisson distribution, that allows for over dispersion and both positive and negative correlation between dimensions [Schmidt and Rodriguez 2010]. Let α be a hyper parameter, and $X = \{X_{t,j}\}, t \in [T], j \in [M]$ be the observed data with T datapoints of M dimensions. Let $\{Z_t\}, t \in T$, denote the cluster index of each data point. A finite mixture of MVP [Dimitris Karlis 2007] can be defined through the following generative model with K mixture components and a mixture distribution β.

$$\lambda_{k,j,l} \sim Gamma(\bar{a}, \bar{b}) \forall j \leq l, k \in [K] \quad (5)$$
$$\beta \sim Dir(\alpha/K); Z_t|\beta \sim \beta \forall t; X_t|\Lambda, Z_t \sim MVP(\Lambda_{Z_t}) \forall t;$$

Often, in a real world modeling setting, selecting the number of mixture components K is cumbersome motivating a non-parametric treatment of this problem. However, there has been very little work on exploring non-parametric mixture modeling extensions with multivariate count data. The only attempt to address this problem was made by [Meligkotsidou 2007] with MVP mixtures using RJMCMC techniques. However, it is often hard to design RJMCMC algorithms that mix well. On the other hand, Dirichlet Process(DP) based models are a natural alternative that are well studied and amenable to hierarchical and temporal extensions like [Teh et al. 2004; Fox et al. 2011] which are of immediate interest to the problem of predictive models for Caching.

In the next few sections, we propose various Bayesian Non-parametric models with Multivariate Poisson. A list of our models is given in table I along with their acronyms for easy reference, that are discussed in detail through the rest of the paper.

Table I: Table of important Acronyms : Models with a * are new models proposed in this paper

Acronym	Expansion
DP	Dirichlet Process
MIP	Multivariate Independent Poisson
MVP	Multivariate Poisson
SMVP*	Sparse Multivariate Poisson
ISMVP*	Independently-Sparse Multivariate Poisson
CSMVP*	Correlated-Sparse Multivariate Poisson
DP-MIP*	DP Mixture of Multivariate Independent Poisson
DP-MVP*	DP Mixture of Multivariate Poisson
DP-ISMVP*	DP Mixture of Independently-Sparse Multivariate Poisson
DP-CSMVP*	DP Mixture of Correlated-Sparse Multivariate Poisson
TDP-MIP*	Temporal DP Mixture of Multivariate Independent Poisson
TDP-MVP*	Temporal DP Mixture of Multivariate Poisson
TDP-ISMVP*	Temporal DP Mixture of Independently-Sparse Multivariate Poisson
TDP-CSMVP*	Temporal DP Mixture of Correlated-Sparse Multivariate Poisson

3.3. DP Mixture of Multivariate Poissons (DP-MVP)

In this section we extend the finite MVP mixture described in equation 5 to a DP based mixture of multivariate Poisson **(DP-MVP)**. Following this, in section 4 we explore models for sparse multivariate data by introducing the Sparse-Multivariate Poisson, following which we explore temporal models for prediction in section 7. We note that these models are of independent interest, outside the caching domain, as they can apply to a wide variety of clustering settings such as text mining, where often counts are used.

We propose to draw the mixture distribution G from a DP prior, $G \sim DP(\alpha, H)$. H is a suitably chosen Gamma conjugate prior for the parameters of MVP, $\Lambda = \{\Lambda_k : k = 1, \ldots\}$, keeping conjugacy in mind.

We thus extend the finite mixture of MVP to DP-MVP as follows.

$$\begin{aligned} \lambda_{kjl} &\sim Gamma(\bar{a}, \bar{b}), \forall j \leq l \in [M], k = 1, \ldots \\ G &= \sum_{k=1}^{\infty} \beta_k \delta_{\Lambda_k}, \beta \sim GEM(\alpha) \\ Z_t|\beta &\sim Mult(\beta), X_t|Z_t \sim MVP(\Lambda_{Z_t}) \text{for } t = 1, \ldots T \end{aligned} \tag{6}$$

where T is the number of observations and $(\Lambda_k)_{jl} = (\Lambda_k)_{lj} = \lambda_{kjl}$. The conjugacy leads to the ability to collapse Λ during inference leading to faster convergence. This is discussed in section 7.3. We also note that the MVP introduces $\binom{M}{2}$ latent variables $\{Y_{tjl}\} \forall j < l$, for each data point. This leads to computationally expensive inference as we will see in the Inference section 7.3.

The DP Mixture of Independent Poissons **(DP-MIP)** can be obtained as a special case of the DP-MVP by restricting $\lambda_{k,j,l} = 0, \forall j \neq l, k = 1, \ldots$.

4. SPARSE MULTIVARIATE POISSON: NON-PARAMETRIC MIXTURES FOR SPARSE MULTIVARIATE COUNT DATA

In this section we explore models for sparse multivariate count data and their mixture model extensions. The multivariate Poisson introduces $\binom{M}{2}$ latent variables for each observation to capture correlations within components of count vectors. However, often real world count data is sparse where only a few components of the count vector are significant. For such sparse count vectors, capturing the full covariance structure leads to overfitting and increased inference complexity. To address this problem we introduce the Sparse Multivariate Poisson that handles sparse multivariate count data by modeling the covariance structure of a relevant subset of data dimensions while allowing for a small amount of independent noise in other dimensions. Consider an indicator vector $b \in \{0, 1\}^M$ to choose the set of active dimensions.

Definition 4.1. The Sparse MultiVariate Poisson **(SMVP)** distribution: Let $\hat{\lambda}_j \geq 0$ and $b \in \{0, 1\}^M$. Consider observation $X_t \in Z^M$. We define $X_t \sim SMVP(\Lambda, \hat{\lambda}, b)$ if

$$\begin{aligned} \forall 1 \leq j \leq l \leq M, Y_{t,j,l} \sim Poisson(\lambda_{j,l}) b_j b_l + Poisson(\hat{\lambda}_j)(1 - b_j)\delta(j, l)) \\ \text{where } Y_{t,j,l} = Y_{t,l,j} \forall j, l,\ X_{t,j} = \Sigma_l Y_{t,j,l} \forall j \end{aligned} \tag{7}$$

where Λ is a symmetric matrix with $(\Lambda)_{jl} = \lambda_{jl}$.

We note that if $b_j = 1, b_l = 1$ then $Y_{t,j,l}$ is distributed as $Poisson(\lambda_{j,l})$. On the other hand if $b_j = 0$ the variables $Y_{t,j,j}$ are distributed as $Poisson(\hat{\lambda}_j)$. The selection variables b_j decide if the jth dimension is active otherwise we consider it noisy, modulated by $Poisson(\hat{\lambda}_j)$ and is independent of other dimensions. The parameter $\hat{\lambda}_j$ will be close to zero to ensure the non-active dimensions have a small mean to account for noise.

Further, one could place priors for parameters Λ and $\hat{\lambda}$ once again, based on conjugacy: $\forall j \in [M]\ \lambda_{l,j} = \lambda_{j,l} \sim Gamma(\bar{a}, \bar{b}),\ \hat{\lambda}_j \sim Gamma(\hat{a}, \hat{b}), \forall j \in [M]$

Definition 4.2. Let $m = \Sigma_{j=1}^{M} b_j$ be the number of active dimensions and M the total number of dimensions. We define an SMVP distribution to be s-sparse when $m/M = s$.

In this work, our goal is to automatically learn the sparsity while fitting the model based on the inherent sparsity in the dataset. To facilitate this, we place appropriate priors for the b_j variables. This is discussed in more detail in section 4.1.

THEOREM 4.3. *Let* $X_t \sim SMVP(\Lambda, \hat{\lambda}, \mathbf{b})$, *then,* $E(X_{t,j}) = \Sigma_l(\lambda_{j,l} b_j b_l + (1 - b_j)\delta_{j,l}\hat{\lambda}_j)$

$$Var(X_{t,j}, X_{t,l}) = \begin{cases} 0 & \textit{if } j \neq l \\ \lambda_{j,l}, & \textit{otherwise} \end{cases}$$

PROOF. The proof for the first part follows from linearity of expectation. For the second part, $Var(X_{t,j}, X_{t,l}) = E(X_{t,j}X_{t,l}) - E(X_{t,j})E(X_{t,l})$. Substituting $X_{t,j}$ and $X_{t,l}$ from definition 4.1 and expanding, we get the result for variance. □

4.1. Modeling Sparsity

An important question that arises while modeling sparsity is the choice of active dimensions for which to model the full correlation. We now explore ways to automatically learn which dimensions are active and which are sparse, by imposing an appropriate prior on the b_j variables $\forall j$. We propose two priors for selecting the active dimensions as follows:

4.1.1. Independently-Sparse Multivariate Poisson(ISMVP). The simplest way of selecting the active dimensions is to chose $b_i, \forall M$ independently from a Bernaulli distribution. A natural prior for the parameter of the Bernaulli distribution is the beta distribution due to its conjugacy that enables simple inference algorithms. This gives rise to a beta-bernaulli prior with parameters a'_j and b'_j as follows: $\eta_j \sim Beta(a'_j, b'_j)\forall j$; $b_j \sim Bernaulli(\eta_j)\forall j$. We call the resultant model the Independently-Sparse Multivariate Poisson (ISMVP).

4.1.2. Correlated-Sparse Multivariate Poisson(CSMVP). Independently drawing $b_j, \forall j$, from the simplest conceivable Beta-Bernaulli prior, fails to incorporate prior information about the inherent correlation structure in the multivariate data when modeling sparsity. For instance, in the case of trace data, due to locality of space, a reasonable prior would assume that adjacent dimensions have a higher correlation, and hence are likely to be active together. One way to capture such a dependency during feature selection is to enforce this by selecting a suitable prior for a'_j and b'_j. The log normal is a common prior to generate a set of correlated positive continuous variables. Hence, we propose The Correlated-Sparse Multivariate Poisson model with Σ, the covariance matrix of log-normal as a hyperparameter. $X_t \sim CSMVP(\Lambda, \hat{\lambda}, \Sigma)$ if :

$$\begin{gathered} W \sim N(0, \Sigma);\ \hat{W}_j = e_j^W \\ a'_j = \hat{W}_j;\ \ b'_j = 1 \\ \eta_j \sim Beta(a'_j, b'_j)\forall j \\ b_j \sim Bernoulli(\eta_j), \forall j \\ X_t \sim SMVP(\Lambda, \hat{\lambda}, b) \end{gathered} \tag{8}$$

The covariance matrix Σ induces covariance between a'_i and a'_j as follows:

$$cov(a'_i, a'_j) = e^{1/2(\Sigma_{ii}+\Sigma_{jj})}e^{\Sigma_{ij}-1} \tag{9}$$

Both these models for feature selection are further explored in the non-parametric mixture modeling context in the next section and are evaluated for synthetic data in section 6 and real-world data from our application's perspective for Cache preloading in the experiments section in 9.

4.2. Dirichlet Process Mixtures of Sparse Multivariate Poisson (DP-ISMVP, DP-CSMVP)

In this section, we extend DP-MVP for modeling sparse multivariate data with the Sparse DP Mixture of multivariate Poissons. We build a non-parametric Mixture of Sparse MVPs as follows. For every mixture component k we introduce an indicator vector $b_k \in \{0,1\}^M$. Hence, $b_{k,j}$ denotes whether a dimension j is active for mixture component k. The DP-CSMVP, with $\hat{\lambda}$ a common Gamma prior for inactive dimensions over all clusters is given by:

$$\begin{gathered}\hat{\lambda}_j \sim Gamma(\hat{a},\hat{b}), \forall j \in [M]; G = \sum_{k=1}^{\infty} \beta_k \delta_{\Lambda_k}, \beta \sim GEM(\alpha) \\ \eta_{k,j} \sim Beta(a'_j, b'_j) \forall k,j; b_{kj} \sim Bernoulli(\eta_{kj}), \forall j,k \\ \lambda_{kjl} \sim Gamma(\bar{a},\bar{b})\ \forall j \le l \in [M], k = 1,\dots \\ Z_t|\beta \sim Mult(\beta), X_t|Z_t,\Lambda,\hat{\lambda},b_{Z_t} \sim CSMVP(\Lambda_{Z_t},\hat{\lambda},b_{Z_t}), t \in [T]\end{gathered} \tag{10}$$

Generating the $X_t, \forall t$ from an ISMVP instead of a CSMVP leads to DP-ISMVP.

In previous section 3.3 we have introduced the DP-MVP model, while we have introduced the DP-ISMVP and the DP-CSMVP models in this section. In the following section, we explore inference for these models.

5. INFERENCE FOR MVP BASED NON-PARAMETRIC MIXTURE MODELS

While inference for DP based mixture models is well explored [Teh et al. 2004][Teh 2010], MVP and Sparse-MVP emissions introduce additional challenges due to the latent variables involved in the definition of the MVP and the introduction of sparsity in a DP mixture setting for the MVP. We describe approximate inference using Gibbs sampling for DP-MVP in section 5.1 and DP-ISMVP and DP-CSMVP in section 5.2.

5.1. Inference for DP-MVP

The existence of $Y_{t,j,l}$ latent variables in the MVP definition differentiates the inference procedure of an MVP mixture from standard inference for DP mixtures. (The large number of $Y_{t,j,l}$ variables also leads to computationally expensive inference for higher dimensions motivating sparse modeling).

We collapse Λ variables exploiting the Poisson-Gamma conjugacy for faster mixing. The latent variables $Z_t, t \in [T]$, and $Y_{t,j,l}, j \le l \in [M], t \in [T]$ require to be sampled. Throughout this section, we use the following notation: $Y = \{Y_t : t \in [T]\}$, $Z = \{Z_t : t \in [T]$ and $X = \{X_t : t \in [T]$. A set with a subscript starting with a hyphen($-$) indicates the set of all elements except the index following the hyphen.

Update for Z_t: The update for cluster assignments Z_t are based on the conditional obtained on integrating out G, based on the CRP[Aldous 1985] process leading to the following product.

$$p(Z_t = k|Z_{-t}, X, \beta, Y; \alpha, \bar{a}, \bar{b}) \propto p(Z_t = k|Z_{-t}; \alpha) f_k(Y_t)$$

$$\propto \begin{cases} n_k^{-t} f_k(Y_t) & \text{k} \in [K] \\ \alpha f_k(Y_t) & \text{k=K+1} \end{cases} \tag{11}$$

Where $n_{k,}^{-t} = \sum_{\bar{t} \neq t} \delta(Z_{\bar{t}}, k)$. The second term $f_k(Y_t) = p(Y_t, Y_{-t} | Z_t = k, Z_{-t}; \bar{a}, \bar{b})$ can be simplified by integrating out the Λ variables based on their conjugacy.
Let $S_{k,j,l} = \sum_{\bar{t}} Y_{\bar{t},j,l} \delta(Z_{\bar{t}}, k)$, for $j \leq l \in [M]$ and

$$F_{k,j,l} = \frac{\Gamma(\bar{a} + S_{k,j,l})}{(\bar{b} + n_k)^{(\bar{a} + S_{k,j,l})} \prod_{\bar{t}: Z_{\bar{t}} = k} Y_{\bar{t},j,l}!} \tag{12}$$

$$\text{By collapsing } \Lambda,\ f_k(Y_t) \propto \prod_{1<=j<=l<=M} F_{k,j,l} \tag{13}$$

Update for $Y_{t,j,l}$: This is the most expensive step since we have to update $\binom{M}{2}$ variables for each observation t. The Λ variables are collapsed, owing to the Poisson-Gamma conjugacy due to the choice of a gamma prior for the MVP.

In each row j of Y_t, $X_{t,j} = \sum_{l=1}^{M} Y_{t,j,l}$. To preserve this constraint, suppose for row j, we sample $Y_{t,j,l}, j \neq l$, $Y_{t,j,j}$ becomes a derived quantity as $Y_{t,j,j} = X_{t,j} - \sum_{p=1,p\neq j}^{M} Y_{t,p,j}$.

The update for $Y_{t,j,l}, j \neq l$ can be obtained by integrating out Λ to get an expression similar to that in equation 12. We however note that, updating the value of $Y_{t,j,l}$ impacts the value of only two other random variables i.e $Y_{t,j,j}$ and $Y_{t,l,l}$. Hence we get the following update for $Y_{t,j,l}$

$$p(Y_{t,j,l} | Y_{-t,j,l}, Z, \bar{a}, \bar{b}) \propto F_{k,j,l} F_{k,j,j} F_{k,l,l} \tag{14}$$

The support of $Y_{t,j,l}$, a positive, integer valued random variable, can be restricted as follows for efficient computation. We have $Y_{t,j,j} = X_{t,j} - \sum_{p=1,p\neq j}^{M} Y_{p,j} \geq 0$ Similarly, $Y_{t,l,l} = X_{t,l} - \sum_{p=1,p\neq l}^{M} Y_{p,l} \geq 0$. Hence, we can reduce the support of $Y_{t,j,l}$ to the following:

$$0 \leq Y_{t,j,l} \leq min \left((X_{t,j} - \sum_{p=1,p\neq l}^{M} Y_{p,j}), (X_{t,l} - \sum_{p=1,p\neq j}^{M} Y_{p,l}) \right) \tag{15}$$

5.2. Inference for DP-ISMVP and DP-CSMVP

Inference is computationally less expensive with the Sparse Multivariate Poisson due to the selective modeling of covariance structure. However, inference for DP-ISMVP and DP-CSMVP requires sampling of $b_{k,j}, j \in [M], k = 1, \ldots$ in addition to latent variables in section 5.1 introducing challenges in the non-parametric setting that we discuss in this section. Note: Variables, η, Λ and $\hat{\lambda}$ are collapsed for faster mixing. Inference for DP-CSMVP involves sampling latent variable W additionally.

Update for $b_{k,j}$: The update can be written as a product:

$$p(b_{k,j} | b_{-k,j}, Y, Z) \sim p(b_{k,j} | b_{-k,j}) p(Y | b_{k,j}, b_{-k,j}, Z; \bar{a}\bar{b}) \tag{16}$$

By integrating out η, we simplify the first term as follows where $c_j^{-k} = \sum_{\bar{k}\neq k, \bar{k}=1}^{K} b_{k,j}$ is the number of clusters (excluding k) with dimension j active.

$$p(b_{k,j} | b_{-k,j}) \propto \frac{c_j^{-k} + b_{k,j} + a' - 1}{K + a' + b' - 1}$$

The second term can be simplified as follows in terms of $F_{k,j,l}$ as defined in equation 12 by collapsing the Λ variables and $\hat{F}_j$ obtained from integrating out the $\hat{\lambda}$ variables.

$$p(Y|b_{k,j}, b_{-k,j}, Z; \bar{a}\bar{b}) \propto \prod_{j \leq l \in [M]} F_{k,j,l}^{b_{k,j} b_{k,l}} \prod_{j \in [M]} \hat{F}_j \tag{17}$$

$$\text{Where } \hat{F}_j = \frac{\Gamma(\hat{a} + \hat{S}_j)}{(\hat{b} + \hat{n}_j)^{(\hat{a}+\hat{S}_j)} \prod_{t,j: b_{Z_t,j}=0} Y_{t,j,j}!} \tag{18}$$

. And $\hat{S}_j = \sum_t Y_{t,j,j}(1 - b_{Z_t,j})$ and $\hat{n}_j = \sum_t (1 - b_{Z_t,j})$

Update for Z_t : Let $\mathbf{b^o} = \{\mathbf{b_k} : k \in [K]\}$ be the variables selecting active dimensions for the existing clusters. The update for cluster assignments Z_t, $t \in [T]$ while similar to the direct assignment sampling algorithm of HDP[Teh et al. 2004], has to handle the case of evaluating the probability of creating a new cluster with an unknown b_{k+1} .

The conditional for Z_t can be written as a product of two terms as that in equation 11

$$\begin{aligned} &p(Z_t = k|Z_{-t}, \mathbf{b^{old}}, X, \beta, Y; \alpha, \bar{a}, \bar{b}) \\ &\propto p(Z_t = k, Z_{-t}|\beta; \alpha, \bar{a}, \bar{b}) \\ &p(Y_t|Z_t = k, Z_{-t}, Y_{-t}, \mathbf{b^{old}}; \mathbf{\bar{a}}, \mathbf{\bar{b}}) \end{aligned} \tag{19}$$

The first term can be simplified in a way similar to [Fox et al. 2011]. To evaluate the second term, two cases need to be considered.

Existing Cluster ($k \in [K]$) :
In this case, the second term $p(Y_t|\mathbf{b^o} Z_t = k, Z_{-t}, Y_{-t}; \bar{a}, \bar{b})$ can be simplified by integrating out the Λ variables as in equation (17).

New Cluster ($k = K + 1$) :
In this case, we wish to compute $p(Y_t|\mathbf{b^o}, Z_t = K + 1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b})$. Since this expression is not conditioned on b_{K+1}, evaluation of this term requires summing out b_{K+1} as follows.

$$p(Y_t|\mathbf{b^o}, Z_t = K + 1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b}) = \sum_{b_{K+1}} p(b_{K+1}|\mathbf{b^o}, \eta) p(Y_t|\mathbf{b^o}, b_{K+1}, Z_t = K + 1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b})$$

Evaluating this summation involves exponential complexity. Hence we resort to a simple numerical approximation as follows. Let us denote $p(Y_t|\mathbf{b^o}, b_{K+1}, Z_t = K + 1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b})$ as $h(b_{K+1})$ The above expression can be viewed as an expectation $E_{b_{K+1}}[h(b_{K+1})|\mathbf{b^o}]$. and can be approximated numerically by drawing samples of b_{K+1} with probability $p(b_{K+1}|\mathbf{b^o})$. We use Metropolis Hastings algorithm to get a fixed number S of samples using the proposal distribution that flips each element of b independently with a small probability $\hat{p}$. The intuition here is that we expect the feature selection vector for new cluster, b_{K+1} to be reasonably close to $b_{Z_t^{old}}$, the selection vector corresponding to the previous cluster assignment for this data point. In our experiments we set S=20 and $\hat{p}$=0.2 to give reasonable results.

We note that in [Wang], the authors address a similar problem of feature selection, however in a multinomial DP-mixture setting, by collapsing the b selection variable. However, their technique is specific to sparse Multinomial DP-mixtures.

Sampling W: The inference for DP-CSMVP entails sampling random variable W. We use metropolis hastings to sample W due to non-conjugacy of prior. The proposal distribution is chosen as the normal distribution while the acceptance ratio is computed based on the target distribution (with η_j integrated out for all j):

$$p(W|\mathbf{b}) \propto \mathbf{p(W)\Pi_j p(b_j|a'_j, b'_j)}$$
$$\propto \mathcal{N}(0, W)\Pi_j \frac{\gamma(a'_j + b'_j + c^k_j)\gamma(a'_j)\gamma(b'_j)}{\gamma(a'_j + b'_j)\gamma(a'_j + c^k_j)\gamma(n'_j + K - c^k_j)} \tag{20}$$

We note that this random variable is absent and need not be sampled in the DP-ISMVP model.

Update for $Y_{t,j,l}$: The update for $Y_{t,j,l}$ is similar to that in section 5.1 with the following difference. We sample only $\{Y_{t,j,l} : b_j = 1, b_l = 1\}$ and the rest of the elements of Y are set to 0 with the exception of diagonal elements for the inactive dimensions. We note that for the inactive dimensions $\{j : j \in [M], b_{Z_t,j} = 0\}$, the value of $X_{t,j} = Y_{t,j,j}$ and hence can be set directly from the observed data without sampling.

For the active dimensions, $\{Y_{t,j,l} : b_j = 1, b_l = 1, j \leq l \in [M]\}$ we sample using a procedure similar to that in section 5.1 by sampling $Y_{t,j,l}, j \neq l$ to preserve the constraint $X_{t,j} = \sum_{l=1}^{M} Y_{t,j,l}$, restricting the support of the random variable in a procedure similar to section 5.1.

$$p(Y_{t,j,l}|Y_{-t,j,l}, Z, \bar{a}, \bar{b}, \hat{a}, \hat{b}) \propto F_{k,j,l}F_{k,j,j}F_{k,l,l} \prod_{1<=j<=M} \hat{F}_j \tag{21}$$

ALGORITHM 1: Inference: DP-CSMVP Inference steps(The steps for DP-MVP are similar and are shown as alternate updates in brackets)

```
repeat
    for $t = 1, \ldots, T$ do
        Sample $Z_t$ from Eqn 19 (For DP-MVP: Eqn 11)
        for $j \leq l \in [M]$ do
            if $b_{Z_t,j} = b_{Z_t,k} = 1$ then
                Sample $Y_{t,j,l}$ from Eqn 21 (For DP-MVP: Eqn 14)
                Set $Y_{t,j,j} = X_{t,j} - \sum_{\bar{j}=1}^{M} Y_{t,j,\bar{j}}$
                Set $Y_{t,l,l} = X_{t,l} - \sum_{\bar{l}=1}^{M} Y_{t,l,\bar{l}}$
            end
        end
    end
    for $j = 1, \ldots, M, k = 1, \ldots, K$ do
        Sample $b_{k,j}$ from Eqn 16 (For DP-MVP: Set $b_{k,j} = 1_M$)
    end
    Sample $W$ from Eqn 20 (This step not applicable for DP-ISMVP, DP-MVP)
until convergence;
```

The inference procedure for the SMVP based models and the full MVP based models is similar, with different updates shown in Algorithm 3.

5.3. Relative Comparison of Complexity: DP-ISMVP/DP-CSMVP with DP-MVP:

In this section, we compare the sparse models based on ISMVP, CSMVP with models based on the full MVP in terms of their run-time efficiency. For the DP-MVP all dimensions are active for all clusters. Hence, it is required to sample $\binom{M}{2}$ random variables for the symmetric matrix Y_t in each Gibbs sampling iteration for each $t \in [T]$. On the other hand, for DP-ISMVP and DP-CSMVP with $\bar{m}_k$ active components in cluster k, we sample only $\binom{\bar{m}_k}{2}$ random variables.For an s-sparse model(recall $s = m/M$) based on SMVP is O($1/s^2$) times faster, which is a significant improvement when $s << 1$.

6. EXPERIMENTAL EVALUATION: SYNTHETIC DATA

In this section, we perform experiments on synthetically generated data to evaluate the relative merits of our models MIP, MVP, ISMVP and CSMVP. We compare (1)All the models in terms of Likelihood on 4 types of synthetically generated datasets (2) To demonstrate the effect of modeling sparsity, we compare the MVP and ISMVP in terms of likelihood and runtime with varying dataset size and varying level of sparsity (3) We compare ISMVP and CSMVP in terms of likelihood with varying dataset size to demonstrate the effect of selecting active dimension based on inherent correlation in the data.

Data Generation and Experiment Setup: For all the experiments, we use 6 dimensional data with 500 points (except for experiments with varying dataset size, where the dataset size is specified on X axis). The details of the data generation process and the parameters used during the generation process are shown in appendix A. Furthermore, all results reported in this section are obtained as a mean over 25 simulations.

6.1. Likelihood Comparison of Various Models

In this section we first compare MIP, MVP,ISMVP and CSMVP in terms of their likelihood. We note in figure II that the simplest dataset based on independent Poisson yields a similar likelihood with all three methods since the MIP already adequately captures the diagonal covariance. However, for the full covariance dataset, the MIP with independence assumption yields a significantly lower likelihood while the MVP and SMVP based models ISMVP and CSMVP yield a higher likelihood owing to the capturing of the correlation structure across dimensions. For a sparse dataset with half the dimensions inactive, we note that the sparse multivariate Poisson based models ISMVP and CSMVP yield a higher likelihood compared to the with full covariance model MVP. This could be attributed to the large number of latent variables in the MVP model that often leads to a poorer fit without a substantial amount of training data. We explore this in the next experiment in more detail by comparing the MVP and ISMVP with varying dataset size in figure 1.

Table II: Likelihood Comparison for Synthetic data

Data	MIP	MVP	ISMVP	CSMVP
	$\times 10^3$	$\times 10^3$	$\times 10^3$	$\times 10^3$
Diagonal Covariance Dataset	-4.806	-4.811	-4.806	-4.806
Full Covariance Dataset	-4.974	-3.469	-3.411	-3.411
Sparse Covariance Dataset	-6.060	-5.128	-2.046	-2.046
Correlated Sparse Dimensions Dataset	-6.128	-5.742	-5.142	-4.803

6.2. Comparison of MVP and ISMVP

In this subsection we analyze MVP vs ISMVP in terms of how well the models fit the data with changing dataset size and effect of changing sparsity on computational efficiency.

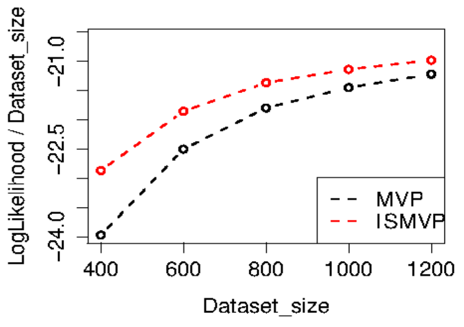


Fig. 1: Likelihood: MVP vs ISMVP with data size. We note that the likelihood of ISMVP is higher than that of MVP due to large number of latent variables to be estimated in MVP, that often leads to suboptimal fit

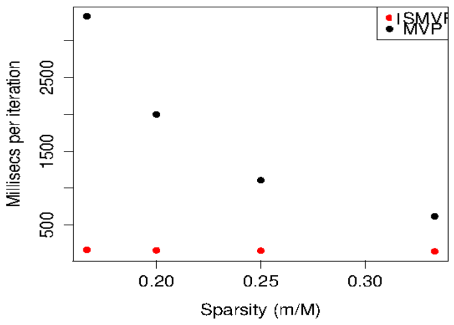


Fig. 2: Runtime per iteration with dimension: MVP vs ISMVP. We note that the full MVP does not scale well for higher dimensions.

Fig. 3: Synthetic Data: Comparison of MVP and ISMVP

Likelihood with Dataset size: We generate synthetic data with a sparse covariance structure Λ (details in appendix A) and vary the dataset size from 400 data points to 1200 datapoints. We plot the likelihood of MVP vs ISMVP with each dataset. We note that ISMVP performs better than MVP consistently in terms of likelihood. This can be attributed to the large number of latent variables per data point that need to be inferred in the MVP model resulting in a poor fit.

Runtime with Degree of sparsity: We note the inference complexity of MVP vs SMVP in figure 2. To illustrate the difference in complexity, we take synthetic data with 500 points with a single active dimension m=1 varying the total number of dimensions M from 3 to 6 and observe with the data getting more and more sparse, the MVP inference complexity increases quadratically with a huge gap between the per iteration run time compared to ISMVP.

6.3. Comparison of ISMVP and CSMVP

In this section, we analyze ISMVP with CSMVP in terms of likelihood with changing dataset size. The CSMVP enables exploiting knowledge of inherent correlation structure in the dataset as a hyper parameter through the log-normal prior for the beta-bernaulli for selecting the active features. To illustrate this, we chose the covariance matrix shown in heatmap in figure 5 as the hyper parameter of the log-normal for synthetic data generated and show the likelihood with datasize in figure 6. The CSMVP clearly outperforms the ISMVP with independent feature selection in this case. We also examine a case to the contrary where the sparsity of the data is not governed by such spatial correlation and note that the CSMVP performs worse than the ISMVP in figure 7. This shows that the CSMVP should be the model of choice where one can incorporate known prior knowledge, as in the case of aggregated memory traces where adjacent dimensions are likely to be active together.

6.4. DP-Mixture Setting

We now evaluate our models in a DP-mixture setting. We once again use 4 synthetic datasets of 500 points : based on Independent covariance, MVP with full covariance, ISMVP where active components are independently selected with a beta bernaulli and finally CSMVP where active components are selected based on the inherent correlation

1.0	0.8	0.6	0.4	0.2	0.0
0.8	1.0	0.8	0.6	0.4	0.2
0.6	0.8	1.0	0.8	0.6	0.4
0.4	0.6	0.8	1.0	0.8	0.6
0.2	0.4	0.6	0.8	1.0	0.8
0.0	0.2	0.4	0.6	0.8	1.0

Fig. 4: Covariance used for the log-normal for selecting active dimensions

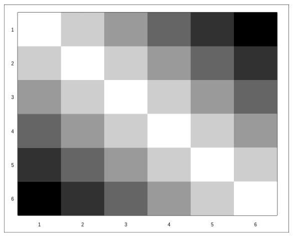


Fig. 5: Heatmap representation of covariance used for the log-normal for selecting active dimensions, where we observe nearby dimensions are more correlated

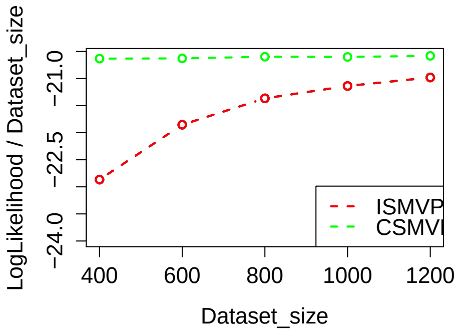


Fig. 6: SMVP vs CSMVP with data generated from covariance shown in heatmap in fig 5 that exhibits spatial correlation like that inherent in traces

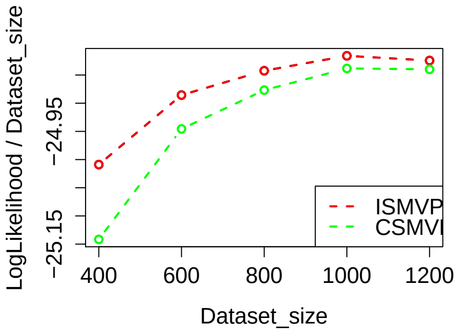


Fig. 7: SMVP vs CSMVP where adjacent features are not necessarily correlated in terms of selecting active dimensions

in the data with log-normal prior. The details of data generation are in appendix A. The result of likelihood comparison with our models DP-MIP, DP-MVP, DP-ISMVP and DP-CSMVP is shown in table III. We once again note a relative performance similar to that of the experiment in section 6.1.

Table III: Comparison of likelihood of DP-MIP, DP-MVP, DP-ISMVP and DP-CSMVP

Dataset	DP-MIP $\times 10^4$	DP-MVP $\times 10^4$	DP-ISMVP $\times 10^4$	DP-CSMVP $\times 10^4$
Diagonal Covariance	-1.4135	-1.4135	-1.4134	-1.4134
Full Covariance	-1.4927	-1.3990	-1.2824	-1.2824
Sparse Covariance	-1.3620	-1.2519	-1.2015	-1.2017
Correlated Sparse	-1.3330	-1.3263	-1.2922	-1.2887

6.5. A discussion on relative merits of models

The Multivariate Poisson (MVP) distribution captures the correlation across dimensions of multivariate count data. However, it introduces a large number of latent variables, learning which during inference often leads to a poorer fit of the data. Our proposed Sparse Multivariate Poisson distribution provides a viable alternative to the MVP by automatically reducing the number of latent variables leveraging sparsity in data when applicable and leads not only to a better fit, but also more efficient inference as seen in section (6.2).

We have proposed two variants of the Sparse Multivariate Poisson, the ISMVP and the CSMVP. We note that the performance of both these models is comparable and the CSMVP that utilizes prior correlation in the data for selecting active dimensions, shows a better fit to the data when there exists such inherent correlation that influences which dimensions are active. We observe similar relative performance of models in terms of quality of fit in a DP-mixture setting in section 6.4.

In the next section, we discuss temporal extensions of our models in a DP-mixture setting that we apply to real world I/O trace data. The temporal extensions are discussed in section 7 while we introduce our cache preloading strategy for capturing long range spatio-temporal correlations for the first time in I/O traces in section 8. We show results on real world benchmark datasets in section 9.

7. TEMPORAL EXTENSIONS OF NON-PARAMETRIC MULTIVARIATE POISSON MIXTURES

We now look at temporal extensions of MVP based mixture models. While DP-MVP, DP-ISMVP and DP-CSMVP cluster multivariate count data effectively capturing the cluster specific covariance structure through the covariance matrix Λ, they do not capture temporal correlations across data instances. The caching application warrants predictive models that capture temporal dependencies for preloading motivating us to explore non-parametric temporal extensions of DP-MVP, DP-ISMVP and DP-CSMVP that extend the HDP-HMM [Teh et al. 2004] to handle multivariate count emissions.

7.1. Temporal NonParametric Mixture of Multivariate Poisson (TDP-MVP)

We propose TDP-MVP as follows that extends DP-MVP by capturing the temporal correlation between multivariate count instances/ The TDP-MVP adapts to an unknown number of hidden states owing to a Dirichlet-Process prior on the state(cluster) specific transition probabilities (π_k) that allow transitions to a countably infinite number of states. We extend the HDP-HMM in equation 3.1 with the Multivariate Poisson emissions as follows:

Let $X_t \in \mathbf{Z}^M, t \in [T]$ be vectors of temporally correlated data.

$$\begin{gathered}
\beta \sim GEM(\gamma), \pi_{\mathbf{k}}|\beta, \alpha_k \sim DP(\alpha_k, \beta) \forall k = 1, \dots \\
\lambda_{kjl} \sim Gamma(\bar{a}, \bar{b}) \; \forall j \le l \in [M], k = 1, \dots \\
Z_t | Z_t - 1, \pi \;\sim\; \pi_{Z_{t-1}}, t \in [T] \\
Y_{t,j,l} | Z_t, \lambda \sim Poisson(\lambda_{Z_t,j,l}), \forall j \le l \in [M] \\
X_t = Y_t 1_M
\end{gathered} \tag{22}$$

Once again this model can again be restricted to the special case of diagonal covariance matrix giving rise to the TDP-MIP by extending the DP-MIP model. The TDP-MIP model captures the temporal dependence across count vectors, but not the correlation across dimensions within each count vector instance. We use both the the TDP-MIP and TDP-MVP as baselines in our experimentation for the caching problem.

7.2. Temporal Sparse Multivariate Poisson Mixture (TDP-ISMVP,TDP-CSMVP)

To handle temporal dependencies in sparse, multivariate count data, we propose the TDP-ISMVP and the TDP-CSMVP that extend the models DP-ISMVP and DP-CSMVP respectively. The generative model is specified in algorithm 2.

We note that the TDP-MVP model described in the previous section 7.1 is a restricted form of TDP-ISMVP where $b_{k,j}$ is fixed to 1. The TDP-CSMVP captures the correlation between dimensions of multivariate count data, models the temporal dependency, at the same time exploiting the sparseness inherent in trace data.

7.3. Inference for TDP-MVP and TDP-ISMVP and TDP-CSMVP

We now explore approximate inference through Gibbs Sampling for our temporal models TDP-MVP, TDP-ISMVP and TDP-CSMVP. The inference for our temporal exten-

ALGORITHM 2: TDP-CSMVP and TDP-ISMVP Generative Model

```
$\beta \sim GEM(\gamma)$
$W \sim \mathcal{N}(0, \Sigma)$ (For TDP-CSMVP)
for $j = 1, \ldots, M$ do
    $a'_j = e^{W_j}, b'_j = 1$ (For TDP-ISMVP $a'_j = 1$)
    $\eta_j \sim Beta(a'_j, b'_j)$
end
for $j = 1, \ldots, M$ do
    $\hat{\lambda}_j \sim Gamma(\hat{a}, \hat{b})$
end
for $k = 1, \ldots$ do
    $\pi_{\mathbf{k}} | \beta, \alpha_k \sim DP(\alpha_k, \beta)$
    for $j = 1 \ldots M$ do
        $b_{k,j} \sim Bernoulli(\eta_j)$
    end
    for $1 \leq j \leq l \leq M$ do
        $\lambda_{kjl} \sim Gamma(\bar{a}, \bar{b})$
    end
end
for $t = 1, \ldots, T$ do
    $X_t \sim SMVP(\Lambda, \hat{\lambda}, b)$
end
```

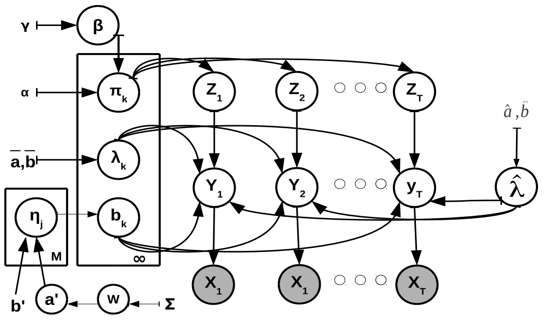


Fig. 8: *Plate Diagram : TDP-CSMVP*

sions is similar to that in section 6 except for the latent variables Z_t due to the HDP introduced as a prior for the HMM transition matrix. Further, for HDP inference we introduce the stick breaking weights $\beta = \{\beta_1, \ldots, \beta_K, \beta_{K+1} = \sum_{r=K+1}^{\infty} \beta_r\}$. and an auxiliary variable m_k is introduced as a latent variable to aid the sampling of β based on the direct sampling procedure from HDP[Teh et al. 2004]. Updates for Z, m and β are similar to [Fox et al. 2011], as detailed in algorithm 3. The latent variables Λ and π are collapsed. The inference procedure for both models is similar, with different updates shown in Algorithm 3.

We once again note the difference in train time complexity between TDP-MVP and TDP-ISMVP/TDP-CSMVP, for these temporal models in the context of caching. For the TDP-MVP all dimensions are active for all clusters leading to sampling $\binom{M}{2}$ random variables during each iteration $t \in [T]$. On the other hand, for TDP-SMVP with $\bar{m}_k$

ALGORITHM 3: Inference: TDP-SMVP
Inference steps(The steps for TDP-MVP are similar and are shown as alternate updates in brackets)

```
Repeat until convergence
for $t = 1, \ldots, T$ do
    // Sample $Z_t$ from
        $$p(Z_t = k | Z_{-t,-(t+1)}, z_{t+1} = l, \mathbf{b^{old}}, X, \beta, Y; \alpha, \bar{a}, \bar{b})$$
        $$\propto p(Z_t = k, Z_{-t,-(t+1)}, {}_{t+1} = l | \beta; \alpha, \bar{a}, \bar{b})$$
        $$p(Y | Z_t = k, Z_{-t}, Y_{-t}, \mathbf{b^{old}}; \mathbf{\bar{a}}, \mathbf{\bar{b}})$$
    //Case 1: For TDP-MVP $b_{k,j} = 1\ \forall j, \forall k$) //and TDP-SMVP For existing k
        $$p(Y | Z_t = k, Z_{-t}, Y_{-t}, \mathbf{b^{old}}; \bar{a}, \bar{b}) \propto \prod_{j \le l \in [M]} F_{k,j,l}^{b_{k,j} b_{k,l}} \prod_{j \in [M]} \hat{F}_j$$
        $$\text{Where } F_{k,j,l} = \frac{\Gamma(\bar{a} + S_{k,j,l})}{(\bar{b} + n_k)^{(\bar{a} + S_{k,j,l})} \prod_{\bar{t}: Z_{\bar{t}} = k} Y_{\bar{t},j,l}!}$$
        $$\text{and } \hat{F}_j = \frac{\Gamma(\hat{a} + \hat{S}_j)}{(\hat{b} + \hat{n}_j)^{(\hat{a} + \hat{S}_j)} \prod_{t,j: b_{Z_t,j} = 0} Y_{t,j,j}!}$$
    , With $\hat{S}_j = \sum_t Y_{t,j,j}(1 - b_{Z_t,j})$ , $\hat{n}_j = \sum_t (1 - b_{Z_t,j})$ and $S_{k,j,l} = \sum_{\bar{t}} Y_{\bar{t},j,l} \delta(Z_{\bar{t}}, k)$, for $j \le l \in [M]$
    //Case 2: For TDP-SMVP for new k, //compute following numerically,where $\mathbf{b^{old}} = \{b_1, \ldots, b_K\}$
        $$p(Y | \mathbf{b^{old}}, Z_t = K+1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b}) = \sum_{b_{K+1}} p(b_{K+1} | \mathbf{b^{old}}, \eta)$$
        $$p(Y | \mathbf{b^{old}}, b_{K+1}, Z_t = K+1, Z_{-t}, Y_{-t}; \bar{a}, \bar{b})$$
    for $j \le l \in [M]$ do
        if $b_{Z_t,j} = b_{Z_t,k} = 1$ then
            // Sample $Y_{t,j,l}$ from
                $$p(Y_{t,j,l} | Y_{-t,j,l}, Z, \bar{a}, \bar{b}) \propto F_{k,j,l} F_{k,j,j} F_{k,l,l}$$
            Set $Y_{t,j,j} = X_{t,j} - \sum_{\bar{j}=1}^{M} Y_{t,j,\bar{j}}$
            Set $Y_{t,l,l} = X_{t,l} - \sum_{\bar{l}=1}^{M} Y_{t,l,\bar{l}}$
        end
    end
end
for $j = 1, \ldots, M, k = 1, \ldots, K$ do
    // Sample $b_{k,j}$ (for TDP-SMVP) from
        $$p(b_{k,j} | b_{-k,j}, Y, Z) \sim p(b_{k,j} | b_{-k,j}) p(Y | b_{k,j}, b_{-k,j}, Z; \bar{a}\bar{b})$$
        $$p(b_{k,j} | b_{-k,j}) \propto \frac{c_j^{-k} + b_{k,j} + a' - 1}{K + a' + b' - 1}$$
    where $c_j^{-k} = \sum_{\bar{k} \neq k, \bar{k}=1}^{K} b_{k,j}$ is the number of clusters (excluding k) with dimension $j$ active
end
for $k = K, \ldots, M, k = 1, \ldots$ do
    $m_k = 0$
    for $i = 1, \ldots, n_k$ do
        $u \sim Ber(\frac{\alpha\beta_k}{i + \alpha\beta_k})$, if $(u == 1) m_k + +$
    end
end
$[\beta_1 \beta_2 \ldots \beta_K \beta_{K+1}] | m, \gamma \sim Dir(m_1, \ldots, m_k, \gamma)$
```

active components in cluster k, we sample only $\binom{\bar{m}_k}{2}$ random variables which leads to significant saving in train time when $\bar{m}_k << M$, which is a common feature in data aggregated from memory traces.

8. A FRAMEWORK FOR CACHE PRELOADING BASED ON MINING BLOCK I/O TRACES

In this section we briefly describe the caching problem and describe our framework for cache preloading using our models developed in the previous sections. While prior techniques for cache preloading are based on analyzing block level patterns in storage traces and operate at a granularity of milli-seconds or shorter, ours is the first work to capture longer range dependencies from data. By capturing long range correlations, and making predictions for minutes or hours of data, we open avenues for alternate caching strategies that are not bound by micro-second level constraints traditionally applicable for caching. Driven by this, our approach is also the first attempt to learn by applying sophisticated modeling techniques by observing workloads and making predictions. Hence our strategy involves (1) A learning phase that happens offline(say once a day), where a prior execution of the trace is observed to learn a model (like the TDP-CSMVP) that captures long-range correlations by learning from aggregated trace data and (2) An operating phase where live predictions are made for an ongoing trace, once every few minutes based on the model learnt and the recent trace history. This is elaborated through the rest of the section.

8.1. The Caching Preloading Problem:

Before we describe our preload strategy for capturing long range correlations, we give a brief introduction to the caching problem and cache preloading.

Application data is usually stored on a slower persistent storage medium like hard disk. A subset of this data is usually stored on **cache**, a faster storage medium. When an application makes an I/O request for a specific block, if the requested block is in cache, it is serviced from cache. This constitutes a **cache hit** with a low application latency (in microseconds). Else, it constitutes a **Cache miss**, when the requested block is unavailable in the cache. In the event of a miss, the requested block is first retrieved from hard disk into cache and then serviced from cache leading to much higher application latency (in milliseconds).

Thus, the application's performance improvement is measured by hitrate. A higher cache hit rate usually leads to a low latency.

$$hitrate = \frac{\#cachehits}{\#cachehits + \#cachemisses}$$

.

Most existing caching strategies (like the LRU: Least Recently Used) are based on eviction techniques to remove the least relevant data from the cache on a miss. *Cache preloading* is the process of proactively loading data into cache to reduce storage system latency. Existing attempts at Cache preloading [Gokul Soundararajan and Amza 2008; Li and Zhou 2004; Gill and Modha 2005] operate based on short range correlations at the milli-second granularity. We explore a new framework, that of preloading based on long-range correlations at a granularity of minutes or hours that can run alongside algorithms based on short-range correlations.

8.2. The Caching Preloading Problem: Relevant Work

We now review related literature on the topic of Cache Preloading. Preloading has been studied before [Zhang et al. 2013] in the context of improving cache performance on enterprise storage servers for the problem of cache warm-up, of a cold cache by preloading the most recently accessed data, by analyzing block I/O traces. Our goal is however different, and more general, in that we are seeking to improve the cumulative hit rate by exploiting long ranging temporal dependencies even in the case of an already warmed up cache. [Zhang et al. 2013] also serves as an excellent reference for state of the art caching related studies and presents a detailed study of the properties of MSR traces[Dushyanth Narayanan 2008], which we have used as our benchmark. Improving cache performance by predicting future accesses based on modeling file-system events was studied in [Kroeger and Long 1996]. They operate over NFS traces containing details of file system level events. This technique is not amenable for our setting, where the only data source is block I/O traces, with no file system level data.

Fine-grained prefetching techniques to exploit short range correlations [Gokul Soundararajan and Amza 2008; Li and Zhou 2004], some specialized for sequential workload types [Gill and Modha 2005] (SARC) have been investigated in the past. Our focus, however is to work with general non-sequential workloads, to capture long range access patterns, exploring prediction at larger timescales. We are not aware of prior work that captures long-range correlations by mining block I/O traces, except our own prior work [Tekumalla and Bhattacharyya 2015] that takes the first steps to address this problem.

In the following section 8.3, we describe our strategy for data aggregation to captures long range correlations in I/O traces

8.3. Spatio-Temporal Data Aggregation

We approach this problem of capturing long-range correlations through data aggregation (also commonly known as temporal abstraction [Bao et al. 2003; Sacchi et al. 2007; I. Batal and Hauskrecht 2009; Bongki et al. 2003; Tao et al. 2004; Lopez et al. 2010]) by taking a more aggregated view of the data to understand longer range dependencies. A block I/O traces is a raw sequence of timestamped block addresses(memory addresses) accessed over a period of time. To aggregate data, we partition the duration of the trace into discrete time intervals and the addressable memory into discrete chunks (bins) to get a coarser view. We then construct histograms over memory bins for each time slice (spanning several seconds) to get a coarser view of the data, counting the number of accesses that occurred with a specific time interval over a specific bin. Hence, we aggregate the data into a sequence of count vectors, $X_1, \ldots, X_T \in \mathcal{Z}^M$, one for each time slice, where each component of the count vector $X_{t,j}$ records the count of memory access in a specific bin (j) (a large region of memory blocks) in that time interval (indexed by t). This transforms a trace into a into a sequence of count vectors.

We describe the aggregation process more formally as follows: Formally, a **trace**, $\mathcal{D}$, of length Q is a sequence of access triplets $< timestamp, blockaddress, \#blocks >$: $r_q = \{s_q, l_q, n_q\}$ where s_q is the timestamp of qth memory access, l_q is the starting LBA (logical block address) requested and n_q denote the number of consecutive blocks starting at l_q that were requested. We first aggregate the raw data temporally along the time axis into T units of ν seconds. The result of the aggregation process is sets $A_1, \ldots, A_T$ containing the actual requests in each of the ν length time intervals. Hence, $A_t = \{r_q : (t-1)\nu \leq s_q < t\nu, q \in [Q]\}$, for each $t \in [T]$. The trace is now further aggregated by dividing the LBA axis into M units of size τ each transforming the trace into a sequence of count vectors $X_1, \ldots, X_T \in \mathcal{Z}^M$ where $X_{tj} = Cardinality\{r_q \in A_t : (j-1)\tau \leq l_q \leq j\tau\}$.

The outcome of the aggregation process is a temporal sequence of count vectors $X_1, \ldots, X_T \in \mathcal{Z}^M$ that represents trace, and becomes the input to our temporal model for multivariate count data, TDP-CSMVP.

8.4. The Cache Preloading Strategy:

We now describe our strategy for trace analysis with statistical predictive models and applying such models for preload. Our strategy involves observing a part of the trace

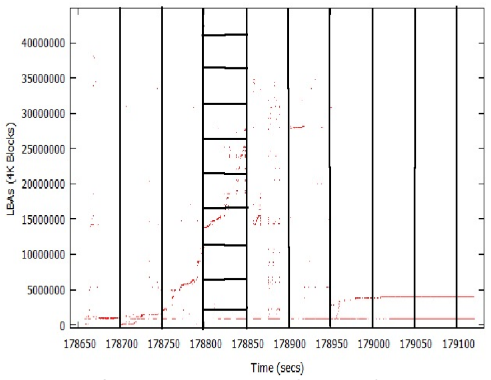


Fig. 9: *Converting trace data to histograms*

$\mathcal{D}^{lr}$, spatio-temporally aggregated, for some period of time for training and deriving a model, which we term as **Learning Phase**. We then use this model on a *live trace* $\mathcal{D}^{op}$ to keep predicting appropriate data to preload in cache every ν minutes (length of timeslice) to improve hitrates in **Operating Phase** based on a aggregated history of the live trace observed thus far.

In terms of the *execution time of our algorithms,* the computation in every preload cycle in the operational phase is designed to run in time much less than the slice length of ν secs. The learning phase, on the other hand, can take a few hours to learn the model. In this paper we restrict ourselves to learning from a fixed initial portion of the trace, by learning the model once, and preloading periodically based on the model. In practice, the learning phase can be repeated periodically (say once a day), to get a more up-to date model as we see more data.

As the goal is to improve hitrates by preloading data exploiting *long range* dependencies, we capture this by *aggregating* trace data into count vector sequences in both the learning phase and the operating phase as described in section 8.3.

In the learning phase, we aggregate the trace $\mathcal{D}^{lr}$ into T_{lr} fixed length time interval slices of length ν seconds each. Let $A_1, \ldots, A_{T_{lr}}$ be the set of actual access requests in each interval of ν seconds. We now aggregate the trace into a sequence of T_{lr} count vectors $X_1, \ldots X_{T_{lr}} \in \mathbf{Z}^M$, each of M dimensions. Recall each count vector X_t is a histogram of accesses in A_t over M memory bins in the tth time slice of the trace spanning ν seconds. We learn models for such count vector sequences in the training phase. Similarly in the operating phase, we work with the live trace $\mathcal{D}^{op}$ that is aggregated into $\{X'_1, \ldots X'_{t'}, \ldots\}$ as data is observed and predictions are made using the live trace history thus far, using the model built from the learning phase. The learning phase and the live phase are described in more detail in the following sections.

8.5. Learning Phase (Learning the statistical predictive model):

The input to the learning phase is a set of sparse count vectors $X_1, \ldots, X_{T_{lr}} \in \mathbf{Z}^M$, correlated within and across instances obtained through data aggregation as described earlier. These count vectors can be intrinsically grouped into cohesive clusters which arise as a result of long range access patterns (see Figure 1(a)) that repeat over time albeit with some randomness. Hence we would like to explore unsupervised learning techniques based on clustering for these count vectors, that capture temporal dependencies between count vector instances and the correlation within instances.

Hidden Markov Models(HMM), are a natural choice of predictive models for such temporal data. In a HMM, latent variables $Z_t \in \{1, \ldots, K\}$ are introduced that follow a Markov chain. Each X_t is generated based on the choice of Z_t, inducing a clustering of count vectors. Hence, as an outcome of the learning phase, we learn the HMM parameters, denoted by θ.

Owing to the variability of access patterns in trace data, a fixed value of K is not suitable for use in realistic scenarios motivating the use of non-parametric techniques of clustering. In section 3 we propose the TDP-MVP, a temporal model for non-parametric clustering of correlated count vector sequences capturing their full covariance structure, followed by the TDP-ISMVP and TDP-CSMVP in section 7.2, that exploit the sparsity in count vectors to better model the data, also leading to more efficient inference. These models proposed by us also have wide applicability outside the caching domain. We describe our models in detail in 7.2. The focus of this section however is to describe how our models fit in the context of our cache preloading framework.

As an outcome of the learning phase, we have a HMM based model(say the TDP-CSMVP) with appropriate parameters. Such a model has the predictive ability to infer the next hidden state on observing a sequence of count vectors from the recent history of the live trace. However, since the final prediction required is that of memory accesses (and not just the state index), we maintain a map from every value of hidden state k to the set of all raw access requests from various time slices during training that were assigned latent state k. $H(k) = \cup_{\{t|Z_t=k\}} A_t$, for $\forall k \in [K]$.

8.6. The Operating Phase (Applying the predictive model for Preloading):

Having observed $\{X_1, \ldots X_{T_{lr}}\}$ aggregated from $\mathcal{D}^{lr}$, the learning phase learns a latent variable model. In the Operating Phase, as we keep observing $\mathcal{D}^{op}$, after the time interval t', the data is incrementally aggregated into a sequence $\{X'_1, \ldots X'_{t'}\}$. At this point, we would like the model to predict the best possible choice of blocks to load into cache for interval $t'+1$ with knowledge of aggregated data $\{X'_1, \ldots X'_{t'}\}$.

This prediction happens in two steps. In the first step, our HMM based model TDP-ISMVP infers hidden state $Z'_{t'+1}$ from observations $\{X'_1, \ldots, X'_{t'}\}$, using a Viterbi style algorithm as follows.

$$(X'_{t'+1}, \{Z'_r\}_{r=1}^{t'+1}) = \underset{(X'_{t'+1}, \{Z'_r\}_{r=1}^{t'+1})}{\text{argmax}} \; p(\{X'_r\}_{r=1}^{t'+1}, \{Z'_r\}_{r=1}^{t'+1}|\theta) \tag{23}$$

Note the slight deviation from usual Viterbi method as $X'_{t'+1}$ is not yet observed. We also note that alternate strategies based on MCMC might be possible based on Bayesian techniques to infer the hidden state $Z'_{t'+1}$. However, *in the operating phase, the execution time becomes important* and is required to be much smaller than ν, the slice length. Hence we explore such a Viterbi based technique, that is quite efficient and runs in a very small fraction of ν in practice for each prediction. The algorithm is detailed in the supplementary material.

In the second step, having predicted the hidden state $Z'_{t'+1}$, we would now like to load the appropriate accesses. Our prediction scheme consists of loading all accesses defined by $H(Z'_{t'+1})$ into the cache (with H as defined previously).

9. EXPERIMENTAL EVALUATION: REAL WORLD DATA

We perform experiments on real-world benchmark traces, to evaluate our models in terms of likelihood and evaluate our caching framework in terms of measuring hitrates using our predictive models.

9.1. Dataset Details:

We perform experiments on publicly available real world block I/O traces from enterprise servers at Microsoft Research Cambridge [Dushyanth Narayanan 2008]. They represent diverse enterprise workloads. These are about 36 traces comprising about a week worth of data, thus allowing us to study long ranging temporal dependencies. We eliminated 26 traces that are write heavy (write percentage $> 25\%$) as we are focused on read cache. See Table IV for the datasets and their read percentages. We present our results on the remaining 10 traces. We also validated our results on an internal industrial workload, NT1 comprising data collected over 24 hours.

We divide the available trace into two parts $\mathcal{D}^{lr}$ that is aggregated into T_{lr} count vectors and $\mathcal{D}^{op}$ that is aggregated into T_{op} count vectors. In our initial experimentation, for aggregation, we fix the number of memory bins M=10 (leading to 10 dim count vectors), length of time slice ν=30 seconds. Further, we use a test train split of 50% for both experiments such that $T_{lr} = T_{op}$. (Later, we also perform some experiments to study the impact of some of these parameters with $M = 100$ dimensions on some of the traces.)

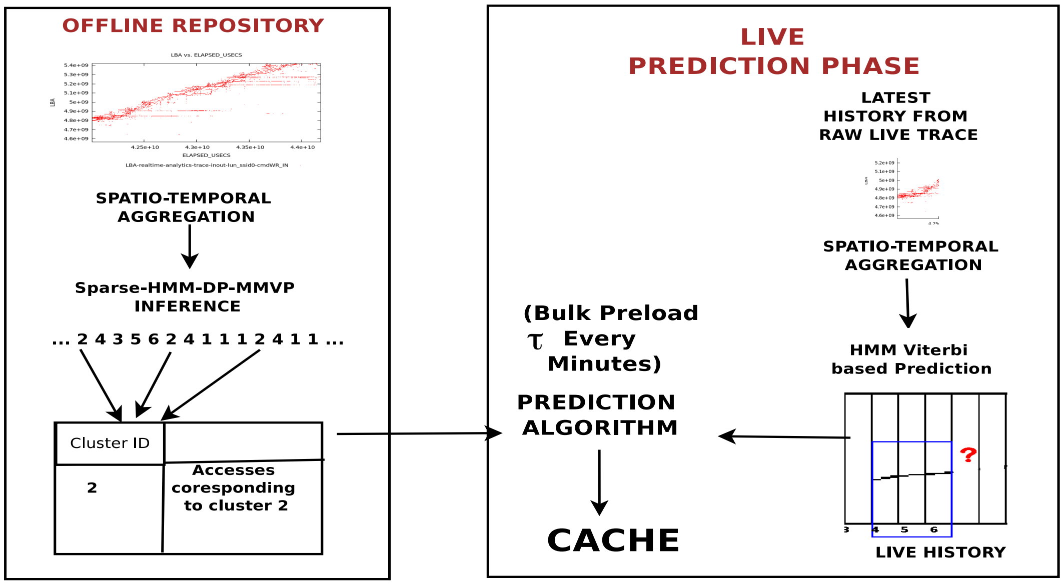

OFFLINE REPOSITORY
SPATIO-TEMPORAL AGGREGATION
Sparse-HMM-DP-MMVP INFERENCE
... 2 4 3 5 6 2 4 1 1 1 2 4 1 1 ...
Cluster ID
2
Accesses coresponding to cluster 2
LIVE PREDICTION PHASE
LATEST HISTORY FROM RAW LIVE TRACE
SPATIO-TEMPORAL AGGREGATION
HMM Viterbi based Prediction
(Bulk Preload τ Every Minutes)
PREDICTION ALGORITHM
CACHE
?
LIVE HISTORY

Fig. 10: The Caching Strategy

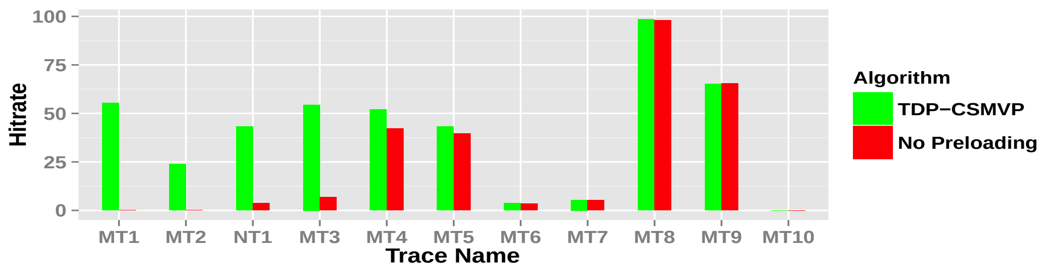


Fig. 11: *Comparison of Hitrates with Preloading based on TDP-CSMVP against LRU baseline without long-range preloading : On MT1 we show the highest hitrate increase of 0.52 vs baseline 0.0002. On MT10 we show the least improvement where both baseline and our model yield 0 hitrate.*

Table IV: Dataset Description

Acronym	Trace Name	Description	Rd %
MT1	CAMRESWEBA03-lvm2	Web/SQL Srv	99.3
MT2	CAMRESSDPA03-lvm1	Source control	97.9
MT3	CAMRESWMSA03-lvm1	Media Srv	92.9
MT4	CAM-02-SRV-lvm1	User files	89.4
MT5	CAM-USP-01-lvm1	Print Srv	75.3
MT6	CAM-01-SRV-lvm2	User files	81.1
MT7	CAM-02-SRV-lvm4	User files	98.5
MT8	CAMRESSHMA-01-lvm1	HW Monitoring	95.3
MT9	CAM-02-SRV-lvm3	project files	94.8
MT10	CAM-02-SRV-lvm2	User files	87.6
NT1	InHouse Trace	Industrial	95.0

9.2. Experiments:

We perform two types of experiments, (1) to understand how well our model fits data in terms of likelihood, (2) the next to show how our model and our framework can be used to improve cache hitrates.

Experiment Set 1: Likelihood Comparison: We show a likelihood comparison between TDP-MVP, TDP-ISMVP, TDP-CSMVP and baseline model TDP-MIP. We train the three models using the inference procedure detailed in section 7.3 on $\mathbf{T}_{lr}$ and compute Log-likelihood on the held out test trace $\mathbf{T}_{op}$. Results are tabulated in Tab V.

Results: We observe that the TDP-CSMVP model performs the best in terms of likelihood, followed by TDP-ISMVP, while the TDP-MVP outperforms TDP-MIP by a large margin. Poor performance of TDP-MIP clearly shows that spatial correlation present across the M dimensions is an important aspect and validates the necessity for the use of a Multivariate Poisson model over an independence assumption between the dimensions. Superior performance of TDP-ISMVP over TDP-MVP is again testimony to the fact that there exists inherent sparsity in the data and this is better modeled by the sparse model. Finally, TDP-CSMVP performs better than TDP-ISMVP in most cases due to the spatial correlation incorporated into the prior when selecting active dimensions. Figures 4,5 show a visualization of the covariance matrix for the log-normal supplied to the CSMVP model.

Table V: Log-likelihood on held out data: TDP-CSMVP model fits data with Best Likelihood

Trace Name	TDP-MIP ($\times 10^6$)	TDP-MVP ($\times 10^6$)	TDP-ISMVP ($\times 10^6$)	TDP-CSMVP ($\times 10^6$)
NT1	-24.43	-20.18	-19.35	-18.1
MT1	-14.02	-8.12	-8.10	-8.06
MT2	-0.46	-0.44	-0.27	-0.25
MT3	-0.09	-0.08	-0.06	-0.05
MT4	-1.15	-1.01	-0.93	-0.64
MT5	-20.23	-12.70	-12.61	-16.3
MT6	-69.12	-50.86	-50.53	-32.3
MT7	-87.25	-84.97	-80.24	-86.2
MT8	-2.44	-2.33	-2.03	-1.99
MT9	-12.45	-12.85	-11.53	-4.64
MT10	-16.49	-13.52	-12.91	-12.45

Experiment Set 2: Hitrates:

We now describe our experiment comparing hitrates obtained through our cache preloading framework, using various proposed models. Before this, we describe the design of our cache simulator that computes hitrates.

Baseline: We note that there is no prior work on long-range preloading that could serve as an additional baseline for our work. Hence we focus on exploring the impact of long-range preloading on hitrates and the relative merits of various models for such preloading. We design two cache simulators as described below, since off the shelf simulators are not available for our purpose (1) a baseline simulator without long-range preloading and (2) a simulator with long-range preloading using our strategy.

–*Baseline: Cache Simulator without long-range Preloading*: We build a cache simulator that services access requests from the trace maintaining a cache. When a request for a new block comes in, the simulator checks the cache first. If the block is already in the cache it records a hit, else it records a miss and adds this block to the cache. The cache has a limited size (fixed to 5% the total trace size). When the cache is full, and a new block is to be added to the cache, the LRU replacement policy is used to select an existing block to remove. We use the hitrates obtained by running the traces on this simulator as our baseline without long-range preloading.

–*Cache Simulator with long-range Preloading*: In this augmented simulator, at the end of every $\nu = 30s$, predictions are made using the framework described in Section 8 and loaded into the cache (evicting existing blocks based on the LRU policy as necessary). While running the trace, hits and misses are kept track of, similar to the previous setup. The cache size used is the same as that in the previous setting.

We compute hitrate, on each of the 11 traces, with a baseline simulator without long-range preloading and an augmented simulator with the ability to preload predicted blocks every $\nu = 30s$. Both the simulators use LRU for eviction.

<u>*Results*</u>: We see in the barchart in figure 11 prediction improves hitrates over baseline without long-range preloading. We see that our augmented simulator gives order of magnitude better hitrates on certain traces (0.52 with preloading against 0.0002 with using plain LRU). The experiment also shows the relative performance of models TDP-CSMVP, TDP-ISMVP and TDP-MVP. This is discussed in more detail in sec 9.3.

Trace	Preloading With TDP-CSMVP	Preloading With TDP-ISMVP	Preloading With TDP-MVP	Without Long-Range Preloading
MT1	**55.48%**	52.45 %	52.45 %	**0.02** %
MT2	23.86%	**23.97** %	22.67 %	**0.1** %
NT1	**43.32** %	34.03 %	16.62 %	**03.66** %
MT3	**54.48** %	34.40 %	34.10 %	**6.90** %
MT4	**52.15%**	52.96 %	51.94 %	**42.12** %
MT5	**43.34%**	40.15 %	40.15 %	**39.75** %
MT6	**3.8** %	3.80 %	3.80 %	**3.60** %
MT7	**5.44** %	5.44 %	5.41 %	**5.22** %
MT8	**98.48** %	98.15 %	98.15 %	**98.04** %
MT9	65.12 %	**65.54** %	65.54 %	**65.10** %
MT10	**0.0** %	0.0 %	0.0 %	**0.0** %

Table VI: Comparing the hitrate with preloading with TDP-MVP, TDP-SMVP and TDP-CSMVP with hitrate of LRU without long-range preloading. We note that TDP-CSMVP beat the baseline showing dramatic improvement for the first four traces, and do well in comparison with the baseline for all the remaining traces. Further, the TDP-CSMVP performs better than TDP-ISMVP in most of the traces with over 50% improvement in MT3

Effect of Training Data Size: We expect to capture long range dependencies in access patterns when we observe a sufficient portion of the trace for training where such dependencies manifest. We show this by running our algorithm for different splits of train and test data (corresponding to the learning phase and the operational phase) for NT1 trace.

We observe that when we see at least 50% of the trace, there is a marked improvement in hitrate for the NT1 trace. Hence we use 50% as our data for training for our experimentation.

In a real world setting, we expect the amount of data required for training to vary across workloads. To adapt our methodology in such a setting periodic retraining to update the model with more and more data for learning as it is available is required. Exploring an online version of our models might also prove useful in such settings.

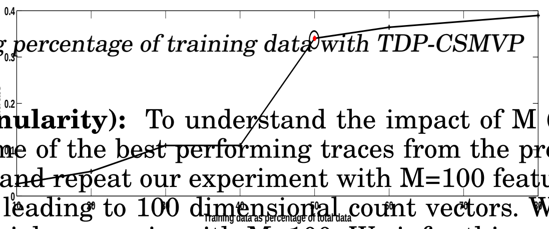


Fig. 12: *Hitrate with increasing percentage of training data with TDP-CSMVP*

Effect of M (aggregation granularity): To understand the impact of M (count vector dimensionality), we pick some of the best performing traces from the previous experiment (barchart in figure 11) and repeat our experiment with M=100 features, a finer 100 bin memory aggregation leading to 100 dimensional count vectors. We find that we beat baseline by an even higher margin with M=100. We infer this could be attributed to the higher sensitivity of our algorithm to detail in traces leading to superior clustering. This experiment also brings to focus the *training time* of our algorithm. We observed that the TDP-CSMVP and TDP-ISMVP outperform TDP-MVP not only in terms of likelihood and hitrates but also in terms of training time. We fixed the training time to at most 4 hours to run our algorithms and report hitrates in table VII. We find that TDP-CSMVP and TDP-ISMVP ran to convergence while TDP-MVP did not finish even a single iteration for most traces in this experiment. This corroborates our understanding that handling sparsity reduces the number of latent variables in TDP-MVP, improving inference efficiency translating to faster training time, particularly for higher dimensional count vectors.

Table VII: Hitrate after training 4 hours: M=10,M=100(x indicates that inference did not complete one iteration)

Trace	TDP-CSMVP M=100	TDP-ISMVP M=100	TDP-MVP M=100	TDP-CSMVP M=10	TDP-ISMVP M=10	No long-range Preloading
NT1	58%(43 clusters)	56%	34%	43.32%	34.0 %	3.66%
MT1	74%(190 clusters)	71%	x	55.48%	52.45%	0.02 %
MT2	51%(23 clusters)	49%	x	23.86%	23.97%	0.10 %
MT3	54.52%(53 clusters)	46%	x	54.48%	34.4 %	3.66%
Avg	59.3 %	56.5%	-	44.28%	36.2 %	1.86%

9.3. Discussion of Results:

We observe both from table V and the barchart (fig 11) that TDP-MVP outperforms TDP-MIP, and TDP-CSMVP performs the best, outperforming TDP-ISMVP and TDP-MVP in terms of likelihood and hitrates, showing that traces indeed exhibit spatial correlation that is effectively modeled by the full covariance MVP and that handling sparsity with the log-normal prior leads to a better fit of the data.

The best results are tabulated in Table VII where we observe that when using 100 bins TDP-CSMVP model achieves an average hitrate of $h = 58\%$, 30 times improvement over LRU without long-range preloading, $h = 0.0186$. On all the other traces, LRU without long-range preloading is outperformed by the TDP-CSMVP, the improvement being dramatic for 4 of the traces. On computing average hitrate for the 11 traces in figure 11, we see 58% hitrate improvement.

Figure VII also shows the varying number of clusters for different traces that comes from varying amount of detail in traces, necessitating the need for non-parametric modeling.

Extensions and Limitations: While we focus on capturing long range correlations, our framework can be augmented with other algorithms geared towards specific workload types to capture short range correlations, like sequential read ahead and Sarc [Gill and Modha 2005] for even higher hitrates, that we hope to investigate in future.

We have shown that our models lead to dramatic improvement for a subset of traces and work well for the rest of our diverse set of traces. We note that there may not be discernible long range correlations present in all traces. However, we have shown, that when we can predict, the scale of its impact is huge. There are occasions, when the prediction set is larger than cache where we would have to understand the temporal order of predicted reads, to help efficiently schedule preloads. Cache size, prediction size and preload frequency, all play an important role to evaluate the full impact of our method. Incorporating our models within a full-fledged storage system involves further challenges, such as real time trace capture, smart disk scheduling algorithms for preloading, etc. These issues are beyond the scope of the paper, but form the basis for future work.

10. CONCLUSIONS

We have explored modeling multivariate count data with models based on the Multivariate Poisson and have proposed Dirichlet Process mixture model extensions. We have proposed the DP-MVP for non-parametric clustering of multivariate count data with the Multivariate Poisson. The MVP introduces a large number of latent variables in attempting to capture the full covariance structure in multivariate counts. We have proposed the Sparse Multivariate Poisson(SMVP) to exploit sparsity in data and proposed Dirichlet Process mixture models DP-ISMVP and DP-CSMVP that capture the correlation within a subset of dimensions for each cluster, leading to a better fit and more efficient inference algorithms. We are not aware of prior work that explores modeling sparsity in multivariate count data. Further we propose temporal extensions of these algorithms (TDP-MVP, TDP-ISMVP and TDP-CSMVP). Inference for models based on the MVP and SMVP are hard due to the intricate dependencies introduced in defining the MVP. We propose efficient Gibbs Sampling based inference techniques for our models.

Finally, in this interdisciplinary work, we have taken the first steps in outlining a preloading framework for leveraging long range dependencies in block I/O Traces to improve cache hitrates. Our algorithms achieve a 30X hitrate improvement on 4 real world traces, and outperform baselines on all traces.

Acknowledgements

The authors are grateful to colleagues from Netapp for useful advise and suggestions. This research was partially supported by a grant from Netapp and DST.

Appendix A: Synthetic Data Generation

In this section of appendix, we discuss the data generation process for experiments in section 6.1, 6.2,6.3 and 6.4.

Data Generation for Section 6.1: The four datasets in this experiment contain 500 points each with 6 dimensions, created using the generative process of MIP, MVP, ISMVP and CSMVP with parameters DP-MIP-lambda1, DP-MVP-lambda1, DP-ISMVP-lambda1, DP-CSMVP-lambda1

Fig. 13: DP-MIP-lambda1

```
[500   0    0    0    0    0 ;
 0   500    0    0    0    0 ;
 0     0  500    0    0    0 ;
 0     0    0  400    0    0 ;
 0     0    0    0  350    0 ;
 0     0    0    0    0  400];
```

Fig. 14: DP-MIP-lambda2

```
[50    0    0    0    0    0 ;
 0    50    0    0    0    0 ;
 0     0   50    0    0    0 ;
 0     0    0   40    0    0 ;
 0     0    0    0   35    0 ;
 0     0    0    0    0   40];
```

Fig. 15: DP-MIP-lambda3

```
[50    0     0    0    0    0 ;
 45   50     0    0    0    0 ;
 30   45   500    0    0    0 ;
 0     0     0  500    0    0 ;
 0     0     0    0  500    0 ;
 0     0     0    0    0   50];
```

Fig. 16: DP-MVP-lambda1

```
[500  450  300  300  450  500;
 450  500  450  450  500  450;
 300  450  500  500  450  300;
 300  350  400  400  350  300;
 350  400  350  400  350  300;
 400  350  300  300  350  400];
```

Fig. 17: DP-MVP-lambda2

```
[50   45   30   30   45   50;
 45   50   45   45   50   45;
 30   45   50   50   45   30;
 30   35   40   40   35   30;
 35   40   35   40   35   30;
 40   35   30   30   35   40];
```

Fig. 18: DP-MVP-lambda3

```
[50   45   30    30   45   50;
 45   50   45    45   50   45;
 30   45   500  450  300   30;
 30   45   450  500  450   30;
 45   50   300  450  500   0;
 50   45   30    30    0   50];
```

Fig. 19: DP-ISMVP-Labda1

```
[400 0   0   450 0   480;
0    30  0   0    0   0;
0    0   30  0    0   0;
450  0   0   400  0   0;
0    0   0   0    30  0;
480  0   0   0    0   400];
```

Fig. 20: DP-ISMVP-Labda2s

```
[400   0  450   0   480  0;
 0    30   0    0    0   0;
 450   0  400   0    0   0;
 0     0   0   30    0   0;
 480   0   0    0  400   0;
 0     0   0    0    0  30];
```

Fig. 21: DP-ISMVP-Labda3

```
[30    0   0    0    0   0;
 0   400   0    0  450   0;
 0     0  30    0    0   0;
 0     0   0   30    0   0;
 0   450   0    0  400   0;
 0     0   0    0    0  30];
```

Fig. 22: DP-CSMVP-lambda1

```
[500  450  300  0    0    0;
 450  500  450  0    0    0;
 300  450  500  0    0    0;
  0    0    0  100   0    0;
  0    0    0    0  100   0;
  0    0    0    0    0  100];
```

Fig. 23: DP-CSMVP-lambda2

```
[100   0    0     0    0    0;
  0   100   0     0    0    0;
  0    0   500  450  300    0;
  0    0   450  500  450    0;
  0    0   300  450  500    0;
  0    0    0     0    0  100];
```

Fig. 24: DP-CSMVP-Labda3

```
[ 100   0    0    0    0    0;
  0   100    0    0    0    0;
  0     0  100    0    0    0;
  0     0    0  450  500  450;
  0     0    0  300  450  500;
  0     0    0  300  450  500];
```

Fig. 25: Parameters for Synthetic Data Generation

respectively shown in figure 25.

Data Generation for Section 6.2: MVP is compared with ISMVP in terms of likelihood and runtime in section 6.2. For the the likelihood comparison in figure 1, data is generated from the generative process of 6 dimensional ISMVP with parameters DP-ISMVP-lambda1 shown in figure 25 with increasing dataset size as shown in the X-axis of the graph in 1. For the runtime comparison infigure 2, 500 data points are generated in each dataset where the degree of sparsity is changed through changing m=1,2,3,4 by selecting full covariance structure from DP-MVP-lambda1 for the first m dimensions, with just diagonals elements for the inactive dimensions.

Data Generation for Section 6.3: ISMVP is compared with CSMVP using data generated from a CSMVP with parameter DP-CSMVP-lambda1 in figure 6 and with DP-ISMVP-Lambda1 in figure 7. Once again, varying datasizes are generated with these parameters. The covariance of the log-normal prior used while fitting data with the two models CSMVP and ISMVP is shown in figure 4.

Data Generation for Section 6.4: The four datasets in this experiment contain 500 points each with 6 dimensions, created using the generative process of DP-MIP (with parameters DP-MIP-lambda1,DP-MIP-lambda2,DP-MIP-lambda3), DP-MVP (with parameters DP-MVP-lambda1,DP-MVP-lambda2,DP-MVP-lambda3), DP-ISMVP (with parameters DP-ISMVP-lambda1,DP-ISMVP-lambda2,DP-ISMVP-lambda3), and DP-CSMVP (with parameters DP-CSMVP-lambda1,DP-CSMVP-lambda2,DP-CSMVP-lambda3) respectively shown in figure 25.

ACKNOWLEDGMENTS

This work was carried out incollaboration with NetApp. We would like to thank members of NetApp Advanced Technology Group (ATG) for their inputs and feedback on the caching front.

REFERENCES

David J. Aldous. 1985. In *École d'été de probabilités de Saint-Flour, XIII—1983*. Springer, Berlin, 1–198.

Ho Tu Bao, Trong Dung Nguyen, Saori Kawasaki, Si Quang Le, Dung Duc Nguyen, Hideto Yokoi, and Katsuhiko Takabayashi. 2003. Mining hepatitis data with temporal abstraction. *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)* (2003).

Moon Bongki, Ines Fernando Vega Lopez, and Vijaykumar Immanuel. 2003. Efficient algorithms for large-scale temporal aggregation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* Vol 15, no. 3 (2003): 744-759 (2003).

John Canny. 2004. GaP: A Factor Model for Discrete Data. In *International ACM conference on research and development in Information Retrieval (SIGIR)*.

Loukia Meligkotsidou Dimitris Karlis. 2007. Finite mixtures of multivariate Poisson distributions with applications. *Journal of Statistical Planning and Inference* (2007).

Finale Doshi-Velez and Zoubin Ghahramani. 2009. Correlated non-parametric latent feature models. In *Uncertanity in Artificial Intelligence (UAI)*.

Antony Rowstron Dushyanth Narayanan, A Donnelly. 2008. Write Off-Loading: Practical Power Management for Enterprise Storage. *USENIX, FAST* (2008).

E.B. Fox, E.B. Sudderth, M.I. Jordan, and A.S. Willsky. 2011. A Sticky HDP-HMM with Application to Speaker Diarization. *Annals of Applied Statistics* (2011).

Binny S. Gill and Dharmendra S. Modha. 2005. SARC: Sequential Prefetching in Adaptive Replacement Cache. In *USENIX Annual Technical Conference (UATC)* (2007-02-06).

Madalin Mihailescu Gokul Soundararajan and Cristiana Amza. 2008. Context-Aware Prefetching at the Storage Server. In *USENIX Annual Technical Conference (UATC)*. USENIX.

Tom Griffiths and Zoubin Ghahramani. 2005. Infinite latent feature models and the Indian buffet process.. In *Advances in Neural Information Processing Systems (NIPS)*. http://books.nips.cc/papers/files/nips18/AdvancesinNeuralInformationProcessingSystems2005_0130.pdf

R. Bellazzi I. Batal, L. Sacchi and M. Hauskrecht. 2009. Multivariate Time Series Classification with Temporal Abstractions. *Florida Artificial Intelligence Research Society Conference (FLAIRS)* (2009).

Dimitris Karlis and Loukia Meligkotsidou. 2007. Finite mixtures of multivariate Poisson distributions with application. *Journal of Statistical Planning and Inference* 137, 6 (June 2007), 1942–1960.

Kazutomo Kawamura. 1979. The structure of multivariate Poisson distribution. *Kodai Mathematical Journal* (1979).

Thomas M. Kroeger and Darrell D. E. Long. 1996. Predicting File System Actions from Prior Events. In *USENIX Annual Technical Conference (UATC)*.

Martin H. C. Law, Anil K. Jain, and Mrio A. T. Figueiredo. 2002. Feature Selection in Mixture-Based Clustering.. In *Advances in Neural Information Processing Systems (NIPS)* (2004-12-01). 625–632.

Chen Z. Srinivasan S. M. Li, Z. and Y. Zhou. 2004. Mining Block Correlations in Storage Systems. In *FAST*. USENIX.

Lopez, Ines Fernando Vega, Richard T. Snodgrass, and Bongki Moon. 2010. Spatiotemporal aggregate computation: A survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* IEEE Transactions on 17, no. 2 (2005): 271-286 (2010).

Loukia Meligkotsidou. 2007. Bayesian multivariate Poisson mixtures with an unknown number of components. *Statistics and Computing* 17, Iss 2, pp 93-107 (2007).

Lucia Sacchi, Cristiana Larizza, Carlo Combi, and Riccardo Bellazzi. 2007. Data mining with temporal abstractions: learning rules from time series. *Data Mining and Knowledge Discovery, Springer, 15.2 (2007): 217-247* (2007).

Alexandra M. Schmidt and Marco A. Rodriguez. 2010. Modelling multivariate counts varying continuously in space. In *Bayesian Statistics Vol 9*.

Yufei Tao, George Kollios, Jeffrey Considine, Feifei Li, and Dimitris Papadias. 2004. Spatio-temporal aggregation using sketches. *International Conference on Data Engineering (ICDE)* (2004).

Y. W. Teh. 2010. Dirichlet Processes. In *Encyclopedia of Machine Learning*. Springer.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2004. Hierarchical Dirichlet processes. *Journal of American Statistical Association (JASA)* (2004).

Lavanya Sita Tekumalla and Chiranjib Bhattacharyya. 2015. Mining Block I/O Traces for Cache Preloading with Sparse Temporal Non-parametric Mixture of Multivariate Poisson. In *SIAM International Conference of Data Mining (SDM), 2015*.

Panagiotis Tsiamyrtzis and Dimitris Karlis. 2004. Strategies for Efficient Computation of Multivariate Poisson Probabilities. *Communications in Statistics (Simulation and Computation)* Vol. 33, No. 2, pp. 271292 (2004).

Blei David M. Wang, Chong. Decoupling Sparsity and Smoothness in the Discrete Hierarchical Dirichlet Process. In *Advances in Neural Information Processing Systems (NIPS)*.

Yiying Zhang, Gokul Soundararajan, Mark W Storer, Lakshmi N Bairavasundaram, Sethuraman Subbiah, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. 2013. Warming up storage-level caches with bonfire. In *USENIX Conference on File and Storage Technologies (FAST)*.