

High-Dimensional Inference in Dynamic Environments

by

Eric C. Hall

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

Rebecca Willett, Co-Supervisor

Robert Calderbank, Co-Supervisor

Sayan Mukherjee

Garvesh Raskutti

Galen Reeves

Dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in the Department of Electrical and Computer Engineering
in the Graduate School of Duke University

2015

ABSTRACT

High-Dimensional Inference in
Dynamic Environments

by

Eric C. Hall

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

Rebecca Willett, Co-Supervisor

Robert Calderbank, Co-Supervisor

Sayan Mukherjee

Garvesh Raskutti

Galen Reeves

An abstract of a dissertation submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in the Department of Electrical and Computer
Engineering
in the Graduate School of Duke University
2015

Copyright © 2015 by Eric C. Hall
All rights reserved except the rights granted by the
Creative Commons Attribution-Noncommercial Licence

Abstract

This thesis addresses several challenges unanswered in classical statistics. The first is the problem of sequentially arriving data in changing environments. It is often the case that the order in which data is received can be critically important to accurate inference, and this observation is especially important when the underlying environment generating the data is likely to be changing and one wishes to study the mechanisms driving this change. For instance, in a wide range of applications, including video analytics, social networks, microscopy, and astronomy, analysts must cope with large quantities of high-dimensional data generated by an environment which is known to be changing. Classical methods often fail on such data because they ignore the underlying dynamics, neglect physical models of data generation and statistics, or fail to scale up to high-dimensions both computationally and statistically. While incorporating time-varying, dynamical models can significantly increase statistical accuracy, the difficulty and complexity of learning are increased because there are now two sources of potential errors: observational noise and environmental change. Classical methods which do not account for time variation would attribute all errors to observational noise, while some techniques, such as those used in autoregressive moving average models, would attribute most of the variation to system dynamics. This thesis presents work which learns a systems' dynamics and state simultaneously, while avoiding such strong assumptions.

The second challenge crucial to the learning process is the ability to handle the

recent onslaught of information known as “big data.” Two types of challenges big data present are large volumes of data and large dimensionality of data, and algorithmic design must address these challenges. In order to process high volume data algorithms must process data very efficiently in order to complete all its calculations in a reasonable amount of time, while high-dimensional data requires algorithms to perform efficient inference in a potentially vastly underdetermined system. Within both of these regimes analysts face the question of “how much data must be collected before reliable inference can be guaranteed?” Such questions are typically addressed via sample complexity and regret bounds, but the existing literature on these bounds does not account for the sequential nature and inherent dependencies among the observed data. Methods such as LASSO have become popular in the last decade, as they prove that the amount of data necessary should scale more closely to the low-dimensional subspace on which the data resides, as opposed to the larger, ambient dimension. These types of observations are critical, and the need to find low-dimensional structure in high dimensional signals is of paramount importance and should be addressed in proposed algorithms. Nevertheless, most LASSO analysis does not address streaming data or temporal dependencies common when tracking dynamic environments.

This thesis presents novel algorithms and analyses which address the issues posed by big data in dynamic environments. The first contribution is to propose methods to analyze sequential data in a dynamic environment using online optimization. Online optimization schemes are inherently designed for high volume data, as they receive one datum at a time, make a quick calculation, and then discard it. These methods are additionally adept at processing high-dimensional data, as they incorporate regularization functions that attempt to find low-dimensional structure in high-dimensional data. The proposed online optimization routines account for time varying environments with associated regret bounds, while making minimal assump-

tions on the underlying system. Unlike previous work in sequential filtering, these methods do not need the dynamical model to be specified *a priori*, but instead have built-in mechanisms to find the best model within a family of candidate dynamics. The next contribution is to use these methods to analyze data generated by a Hawkes process, a model which is used to characterize complex interactions among a collection of point processes, for instance neurons firing within a network. Using this analysis, network inference on streaming data can be performed with more robustness to poorly-specified model parameters than previously existing methods. Finally, previously unknown sample complexity bounds for a discretized version of the Hawkes process, the log-linear Poisson autoregressive process, are shown and proven. These results, which utilize recent advances in statistical learning theory, show how the assumption of low dimensional structure can greatly aid in the estimation procedure of high dimensional data.

“A noble spirit embiggens the smallest man”

- Jebediah Springfield

Contents

Abstract	iv
List of Figures	xii
List of Abbreviations and Symbols	xiv
Acknowledgements	xvi
1 Introduction	1
1.1 Overview of Sequential Data	1
1.2 Online Learning	6
1.3 Autoregressive Point Processes	9
1.4 Summary of Thesis Contributions	11
2 Online Convex Optimization in Dynamic Environments	13
2.1 Introduction	13
2.1.1 Organization of Chapter	15
2.2 Problem Formulation	15
2.3 Dynamic Mirror Descent	17
2.3.1 Tracking Regret of DMD	20
2.3.2 Data-dependent Dynamics	22
2.4 Prediction with a Finite Family of Dynamical Models	23
2.5 Parametric Dynamical Models	26
2.5.1 Covering the Set of Dynamical Models	26

2.5.2	Additive Dynamics in Exponential Families	28
2.6	Conclusions	32
2.7	Proofs	33
2.7.1	Proof of Theorem 2.2	33
2.7.2	Proof of Theorem 2.3	36
2.7.3	Proof of Theorem 2.4	37
2.7.4	Proof of Lemma 2.6	38
2.7.5	Proof of Theorem 2.7	39
3	DMD and DFS Experiments and Results	40
3.1	Introduction	40
3.2	DMD Experiment: Dynamic Textures with Missing Data	41
3.3	DMD Experiment: Solar Flare Detection with Missing Data	43
3.4	DFS Experiment: Compressive Video Reconstruction	44
3.5	DFS Experiment: Downsampled Traffic Surveillance Reconstruction	48
3.6	DMD with Parametric Additive Dynamics	51
3.7	DMD with Parametric Additive Dynamics Experiment - Enron Email Corpus	54
3.8	Foreground-Background Separation in Poisson Video	57
3.8.1	Problem Formulation	61
3.8.2	Method	62
3.8.3	Experimental Results	65
3.9	Conclusions	68
4	Tracking Dynamic Point Processes on Networks	69
4.1	Introduction	69
4.2	Problem Formulation	73
4.3	Online Learning	75

4.4	Loss Function and Dynamic Model	77
4.4.1	Time Discretized Loss Function	78
4.4.2	Dynamical Models	81
4.5	Proposed Algorithms	86
4.5.1	DMD Algorithm - W Known	86
4.5.2	Proposed Algorithm - W Unknown	88
4.6	Computational Complexity	92
4.7	Experimental Results	92
4.7.1	Model mismatch, W known	94
4.7.2	Learning W	95
4.7.3	Memetracker	100
4.8	Conclusion	102
4.9	Appendix of Chapter 4	103
4.9.1	Lemma 4.7	103
4.9.2	Proof of Lemma 4.2	104
4.9.3	Proof of Lemma 4.3	108
4.9.4	Proof of Theorem 4.4	109
4.9.5	Proof of Lemma 4.5	114
4.9.6	Proof of Theorem 4.6	114
4.9.7	Online Gradient Descent	116
5	Log-Linear Poisson Autoregressive Model	118
5.1	Introduction	118
5.2	Problem Formulation	121
5.3	Stationary Distribution and Mixing Time	123
5.4	Bounds for Estimation of Adjacency Matrix	125

5.5	Experimental Results	134
5.6	Conclusion	137
5.7	Supplementary Lemmas	137
5.7.1	Proof of Lemma 2	137
5.7.2	Proof of Lemma 3	138
5.8	Appendix for Chapter 5	148
5.8.1	Poisson Concentration Bounds	148
5.8.2	Exponential Martingale	149
6	Conclusion and Future Directions	150
	Bibliography	153
	Biography	164

List of Figures

3.1	Loss curves for the dynamic textures experiment for Dynamic Mirror Descent and Mirror Descent	43
3.2	Loss curves for the solar flare experiment for DMD and MD	45
3.3	Loss curves for the compressed sensing microscopy experiment for DMD and MD	46
3.4	Instantaneous predictions for DMD and COMID in compressed sensing experiment	47
3.5	Moving average loss for DMD, DFS and COMID for downsampled traffic surveillance experiment	50
3.6	Reconstruction of DFS and COMID methods on downsampled traffic surveillance experiment	51
3.7	Loss curve and estimate of DMD-exp algorithm in self-exciting point process network experiment	53
3.8	Relative gain of DMD vs MD for Enron dataset	57
3.9	Filtered loss for anomaly detection of DFS vs MD for Enron dataset	58
3.10	Demonstration of challenge of background subtraction for photon-limited video	60
3.11	Absolute difference between moving average and true background	63
3.12	Foreground and background reconstruction for a Poisson video at $t = 250$	66
3.13	Foreground and background reconstruction for a Poisson video at $t = 925$	67
4.1	Performance of Algorithm 4.1 with incorrect exponential decay influence function	95

4.2	Performance of Algorithm 4.1 with incorrect rectangular influence function	95
4.3	True generating network	96
4.4	Performance of Algorithm 4.1 with different values of W	96
4.5	Performance of Algorithm 4.1 with different values of W and mismatched system parameters	97
4.6	Estimates of underlying network using Algorithm 4.2 and OGD	98
4.7	ROC Curves to find significant relationships in the network	99
4.8	Batch loss of network estimates	100
4.9	Amount of relationships above threshold	102
5.1	MSE curves of RMLE for log-linear Poisson autoregressive process	135
5.2	Example RMLEs for log-linear Poisson autoregressive process	136

List of Abbreviations and Symbols

Symbols

Notation is consistent within each section, but we have tried to adhere to the same basic conventions and notation for the entire document. Below are the most common symbols and abbreviations used.

T	The time horizon of the experiment
t	A single instance in time ranging from 1 to T
θ	A value in the decision space of the Forecaster/Learner
$\hat{\theta}$	Values with a $\hat{}$ indicate estimates
f_t	Overall convex loss function at time t
ℓ_t	Convex loss function at time t , data-fit
r	Convex regularization function
x_t	Data at time t
Φ	Dynamical model
η_t	Primary step-size parameter
ρ_t	Secondary step-size parameter
W	Adjacency matrix of a network
$D(\theta_1\ \theta_2)$	Bregman divergence
$Z(\theta)$	Log-partition function
λ	Legendre-Fenchel dual of θ

Abbreviations

DMD	Dynamic Mirror Descent.
DFS	Dynamic Fixed Share
MD	Mirror Descent
COMID	Composite Objective Mirror Descent
SGD	Stochastic Gradient Descent
OGD	Online Gradient Descent
RMLE	Regularized Maximum Likelihood Estimator

Acknowledgements

I have been lucky to have a tremendous amount of support on my path to a PhD. I would first like to thank my parents, who have aggressively supported me in all my endeavors. Without them I would have never even been able to take a single step on this journey. I would also like to thank Allison Fratto for not only encouraging me, but also for being by my side the whole way.

I'd like to thank all of my fellow graduate students who have been in the trenches along side me for all the technical and emotional help they have provided. Specifically, I'd like to thank Zachary Harmany and Kalyani Krishnamurthy for their leadership and for showing me that there is a finish line in sight, my lab group mates Xin Jiang, Peng Guan and Albert Oh, for their never ending energy and enthusiasm, the Nowak group, Aniruddha Bhargava, Daniel Pimentel-Alarcón, Gautam Dasarathy and Nikhil Rao for welcoming me to Madison and their group, and John Marcus, David Carlson and Lalit Jain for always being willing to have an impromptu conversation about math.

I am grateful to my committee, Robert Calderbank, Galen Reeves, Sayan Mukherjee and Garvesh Raskutti for their support and valuable feedback. Most importantly, I'd like to thank my advisor Rebecca Willett, for her patience, encouragement, collaboration and guidance. Pursuing a PhD is a difficult task no matter the circumstances, but without a great advisor to work with it would be nearly impossible.

I am thankful to have been able to do research at the University of Wisconsin -

Madison during my graduate career. I'd like to thank the organizations which have provided the funding which have allowed this work to happen: AFOSR, NSF, and ARO.

Introduction

1.1 Overview of Sequential Data

Modern sensors are collecting very high-dimensional data at unprecedented rates, often from platforms with limited processing power. These large datasets allow scientists and analysts to consider richer physical models with larger numbers of variables, and thereby have the potential to provide new insights into underlying complex phenomena. For example, the Large Hadron Collider (LHC) at CERN “generates so much data that scientists must discard the overwhelming majority of it – hoping hard they’ve not thrown away anything useful.” [1] Typical NASA missions collect hundreds of terabytes of data every hour [2]: the Solar Data Observatory generates 1.5 terabytes of data daily [3], and the upcoming Square Kilometer Array (SKA, [4]) is projected to generate an exabyte of data daily, “more than twice the information sent around the internet on a daily basis and 100 times more information than the LHC produces” [5]. In these and a variety of other science and engineering settings, there is a pressing need to recover *relevant or anomalous* information *accurately and efficiently* from a high-dimensional, high-velocity data stream.

Rigorous analysis of such data poses major issues, however. First, we are faced with the notorious “curse of dimensionality”, which states that the number of observations required for accurate inference in a stationary environment grows exponentially with the dimensionality of each observation. This requirement is often unsatisfied even in so-called “big data” settings, as the underlying environment varies over time in many applications. Furthermore, any viable method for processing massive data must be able to scale well to high data dimensions with limited memory and computational resources. Finally, in a variety of large-scale streaming data problems, ranging from motion imagery formation to network analysis, the underlying environment is dynamic yet predictable, but many general-purpose and computationally efficient methods for processing streaming data lack a principled mechanism for incorporating dynamical models. Thus a fundamental mathematical and statistical challenge is accurate and efficient tracking of dynamic environments with high-dimensional streaming data.

A generic statistical learning problem could be described in the following way:

- Environment generates data sequence $\{x_t\}_{t=1}^T, x_t \in \mathbb{R}^n$ drawn from the distributions $\{p(\cdot; \theta_t)\}_{t=1}^T$ each parameterized by $\theta_t \in \mathbb{R}^d$
- Learner observes data sequence $\{x_t\}_{t=1}^T$
- Learner creates estimate sequence $\{\hat{\theta}_t\}_{t=1}^T$.

In the classic independent, identically distributed (iid) setting, it would be assumed that $\theta_1 = \theta_2 = \dots = \theta_T = \theta$ for all t , and therefore a single point estimate $\hat{\theta}$ would be produced. Of major interest to the research community is the analysis of time-series or sequential data. As opposed to the iid setting the order and timing of the data can be crucial, as the underlying environment may be changing, which manifests by allowing the parameter θ_t to be changing in time. If the environment is

changing rapidly, classical inference methods will break because they are inherently built on the idea that all data share a common ground truth and any differences between observations stem primarily from noise. However, in many applications the environment is obviously changing and these changes need to be accounted for. For instance, the field of contextual anomaly detection[6] tries to identify data outliers while acknowledging that anomalous behavior at one moment may not be anomalous at another. An example of this would be a temperature reading of 30° F. In winter this would not necessarily be anomalous, but in the heat of summer this should raise red flags. Due to the inherently changing seasonal weather pattern simple inference techniques need to be modified.

While we would like to add the flexibility of allowing θ_t to vary in time, we also require the value at any given time to be related in some way to values at recent times, or else the learning process degenerates into a sequence of completely disjoint learning problems. Therefore we can make the assumption that θ_t is changing according to some time evolving process. While assuming some unknown relationship of the data from one time to the next can add to the predictive power of the learner, it also adds complexity to the inference process. We can propose time evolving models which we believe reflect the way the underlying environment is changing, but by imposing these models we allow for the possibility of accruing estimation errors from using a poor model. Additionally, even after fitting a time-evolving model it is not always obvious when poor estimation is due to a poor model or just a high level of noise in the system. Thus, we need to create algorithms which can incorporate time evolving models, but are robust to modeling errors.

A second major challenge that is side-stepped by the ubiquitous iid assumption is dependence between observations. Even if all $\theta_1 = \theta_2 = \dots = \theta_T$, we still must consider settings where the observations might be dependent on one another. For instance, neurons firing in the brain have been shown to trigger or inhibit other

connected neurons. Therefore, we require algorithms and theory which account for dependencies between observations in order to accurately predict which neurons will next fire. It is crucial in statistical learning to be able to answer the question, "how much data does one need to collect in order to to achieve a desired level of accuracy?" In the iid setting, this question is answered using well known concentration inequalities and central limit theorem type properties. However, these bounds inherently assume all the data received are equally informative. In the setting where data are dependent, we must characterize how informative data are based on their dependence structure, and incorporate this structure into novel concentration and sample complexity bounds.

Additionally, it is common in the current age of "big data" for datasets to be extremely large, and algorithms which account for this must be considered. Big data come in two main varieties: high volume and high dimension. High volume data means we are receiving so many data points that it could be difficult to even store all of the raw data (T large). The solution to a high rate of data is to create algorithms capable of processing the sequence very quickly, while still performing accurate statistical inference. High dimensional data is the setting where the parameters of interest are so large that we might not ever receive enough data to perform accurate inference under the classical regime, a phenomena known as the "curse of dimensionality" (d large). One solution to the high dimensionality problem is the exploitation of assumed structure in the parameters of interest. Many techniques utilize the idea of sparsity [7; 8; 9; 10] in order to learn parameters of interest, and show that the amount of data necessary is on the order of the sparsity level and not the overall dimension, which can be a huge difference. Both of the big data regimes require careful analysis of the data through algorithms which are both fast and exploit important low-dimensional structure. Therefore, the main challenges with sequential data is to be able to understand the mechanics of the underlying

dynamic process while still being robust to modeling errors and noise while using efficient algorithms.

This thesis first approaches the subject of sequential data by proposing several online learning algorithms which incorporate dynamical models in Chapter 2. The primary goal of online learning is to create very fast algorithms by keeping track of a running estimate $\hat{\theta}_t$, then processing a single new data point to update the estimate and repeating. In this way every data point is used, but the algorithm is very efficient because of the relatively small amount of processing per datum. This fast processing allows for analysis of very fast streaming data. Previous online methods such as least mean squares, recursive least squares and mirror descent fail to incorporate rapidly changing dynamics and other methods such as Kalman filtering are very inflexible in the allowable dynamical models, loss functions and noise models. Chapter 3 is devoted to showing experiments using these proposed methods in a wide variety of problem settings including both synthetic and real data.

Chapters 4 and 5 look at specific examples of time-series problems, the Hawkes process [11; 12; 13] and the closely related log-linear Poisson autoregressive model [14; 15; 16]. These are multivariate point process models where each process has an associated, time-varying rate which we would like to learn. However, each process' rate is influenced by the actions taken by the other processes, and the magnitude of this effect is unknown. Therefore, we can pose these problems in the form of sequential data with a changing underlying environment. At any given moment we are receiving data about which processes have acted as a noisy observation of each process' rate. However, these rates are changing rapidly in response to the other process' actions. Chapter 4 demonstrates the methods outlined in Chapter 2 applied to the Hawkes process in the online setting. The main advantages of our method is that we can process information very quickly due to the streaming nature of our algorithm, and we are robust to modeling errors if the data were not

Algorithm 1.1 General Framework for Online Framework (Prediction)

Learner chooses $\hat{\theta}_1 \in \Theta$.
for $t = 1, \dots, T$ **do**
 Environment produces x_t
 Learner incurs loss $\ell_t(\hat{\theta}_t) = \ell(\hat{\theta}_t; x_t)$
 Learner uses $\hat{\theta}_t, \{x_i\}_{i=1}^t, \ell_t$ and available side information to create estimate $\hat{\theta}_{t+1}$
end for

generated precisely according to a Hawkes process, for instance. Chapter 5 shows previously unproven error rates for the log-linear Poisson autoregressive model, where the underlying adjacency matrix is additionally assumed to have a sparse structure. This chapter therefore emphasizes the importance of incorporating structure into learning processes in order to get better estimates with less data.

1.2 Online Learning

A general online learning routine is outlined in Algorithm 1.1. The basic idea is that at every time point the learner has a current estimate on the state of the system, and then compares that estimate to the data given by the environment. After seeing the data, the learner makes some adjustment to the estimate which will hopefully match the upcoming data points closely. There are two main regimes for this framework, in the first the learner is allowed to use observations x_1, \dots, x_t in creating estimate $\hat{\theta}_t$, which is called the filtering problem. The other regime is where the learner can only use observations up to time $t - 1$ when creating estimate $\hat{\theta}_t$, which is called the prediction problem. In this thesis we focus primarily on the prediction problem. As we can see in this general framework, the main differences in most algorithms comes in how the learner actually creates its estimate for the next time point, which will depend heavily on the underlying assumptions about the data sequence.

One approach to this problem is through the field of online convex programming also known as online optimization[17; 18; 19; 20]. The basic idea in this field is that

the underlying parameter that we wish to estimate, θ is either stationary, changing slowly or changing only at a few distinct time instances. Because the parameter is changing slowly, then we know that the data x_t should be very informative to the parameter at the next time θ_{t+1} , and therefore only slight modifications of the estimate need to be made at every given time point. A common method in this field is called Composite Objective Mirror Descent (COMID) [21], where the estimate at each time is created as

$$\hat{\theta}_{t+1} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla \ell_t(\hat{\theta}_t), \theta \rangle + \eta r(\theta) + D(\theta \| \hat{\theta}_t)$$

where $\nabla \ell_t(\hat{\theta}_t)$ is an arbitrary subgradient of the convex function ℓ_t at $\hat{\theta}_t$, η_t is a step-size parameter, r is a convex regularization function and $D(\theta_1 \| \theta_2)$ is a Bregman divergence which measures distance from points θ_1 and θ_2 . The idea of this minimization routine is that it seeks a value of θ which both decreases the loss function by looking at the gradient of the current estimate, attempts to find a parsimonious solution which is immune to overfitting by incorporating the regularization function, all while simultaneously not straying too far from the previous estimate, by including the Bregman divergence.

Algorithms like COMID have several useful properties. The first is that the framework is extremely general and flexible. Depending on the problem setting at hand, we can use any choice of appropriate loss function and Bregman divergence. Secondly, the theoretical guarantees for these methods make very mild assumptions about the loss functions, data, and decision space. Finally, by including the function r we can induce low-dimensional features in our estimate such as sparsity, which can greatly decrease the amount of data needed. However, a large drawback is that this and similar methods inherently assume that the underlying environment is changing very slowly. This is why a Bregman divergence is included in the optimization routine, which enforces that the estimate at one time point should be similar to

previous time points, essentially because it is believed that the true parameters are also not changing very rapidly. It is this penalty that needs to be altered if we are in a dynamic environment.

There are some pre-existing techniques which aim to incorporate dynamics into the estimation process, one of the most famous being the Kalman filter [22]. The basic set-up for the Kalman filter is that true state of the system is evolving according to

$$\theta_t = A_t \theta_{t-1} + u_t$$

where $u_t \sim \mathcal{N}(0, \Sigma_1)$, and A_t and Σ_1 are known matrices. Observations are of the form

$$x_t = C_t \theta_t + v_t$$

where $v_t \sim \mathcal{N}(0, \Sigma_2)$ and, C_t and Σ_2 are known. The goal is to then incorporate the Gaussian observation model, with the underlying linear dynamical model. This problem is solved optimally by tracking an entire distribution of θ and at every time making the following innovations

$$\tilde{\theta}_{t+1} = (I - K_t A_t) \hat{\theta}_t + K_t x_t$$

$$\hat{\theta}_{t+1} = A_{t+1} \tilde{\theta}_{t+1}$$

for some optimally chosen matrix K_t called the Kalman gain. We can see what this method is essentially doing is innovating the prediction estimate with the new observation forming a filtering estimate in $\tilde{\theta}_{t+1}$ and then simply applying the mean dynamics to generate the prediction estimate $\hat{\theta}_{t+1}$. Unlike current online optimization routines, this method successfully incorporates the linear dynamics and has the optimal mean squared error. However the Kalman filter makes many assumptions: the noise is Gaussian, the loss we are interested in is mean squared error, and the dynamics are linear and known. While there are extensions to the Kalman filter

which aim to get around a few of these issues, they generally still have somewhat constraining assumptions.

This thesis proposes a novel optimization framework in Chapter 2 which aims to combine the benefits of both the online optimization and sequential filtering fields. In our method known as Dynamic Mirror Descent, we propose a two step online optimization routine, the first step of which is an innovation step inspired by methods like COMID, while the second step applies a known dynamical model, much in the same way as the second step of the Kalman filter. The rest of Chapter 2 is devoted to proving theoretical results about DMD, and extensions to learn the dynamical model simultaneously to the estimation procedure.

1.3 Autoregressive Point Processes

One specific type of problem that uses the sequential nature of data is autoregressive point processes. Such models have been used in a wide variety of applications including seismology, neurology, epidemiology and finance. In these problems we assume we observe a series of labeled events $\{(k_n, \tau_n)\}_n$ where k_n denotes the process involved in the n^{th} action, which occurs at time τ_n . Each process k is associated with a time varying rate function $\mu_k(t)$, and the likelihood of the process having an event between times τ_1 and τ_2 is controlled by the value $\int_{\tau_1}^{\tau_2} \mu_k(t) dt$. The autoregressive nature of these problems comes when the functions $\mu_k(t)$ are assumed to be dependent on the events that occurred before time t . One example of such an autoregressive point process is called the Hawkes process wherein the rate functions take the following form:

$$\mu_k(t) = \bar{\mu} + \sum_{n=1}^{N_t} h_{k,k_n}(t - \tau_n)$$

where $\bar{\mu}$ is a baseline likelihood, N_t is the number of events that have occurred up to time t , and $h_{k,k_n}(\tau)$ is a function which measures how much an event by process k_n will influence process k after a delay of τ units of time. Specifically, we will assume these h functions take the form $h_{k_1,k_2}(\tau) = W_{k_1,k_2}h(\tau)$, where all the functions have the same time profile and are only modulated by a different magnitude based on the two processes involved and the relative strength of the influence. The goal of studying these processes is to be able to estimate the rate functions in order to make predictions about possible upcoming actions, while only observing the labelled events. It is clear that in order to make accurate predictions, accurate estimates of the rates are needed, which in turn require accurate estimates of the network W . Therefore this is a sequential data problem with strong dynamics which are modeled by a parametric family of functions, with the parameter being the value of W . In Chapter 4 we solve this problem by applying the methods of the previous chapters to a discretized version of the Hawkes process. By applying our online methods we create an algorithm which can both estimate the rates and the network, while processing the information efficiently and quickly in a streaming manner and also being robust to model mismatch.

A related, discrete time model called the log-linear Poisson autoregressive model is defined as

$$x_t \sim \text{Poisson}(\exp(\bar{\mu} + Wx_{t-1}))$$

which we study in Chapter 5, under the assumption that the matrix W is sparse, with at most s non-zero elements. By making the assumption that the matrix is sparse we can prove that the estimation process requires less data than in the general dense W scenario. We show novel error rates which scale with the sparsity level s and only logarithmically in the overall dimension of the matrix.

1.4 Summary of Thesis Contributions

In this thesis I present several novel contributions:

- I propose an algorithm, Dynamic Mirror Descent (DMD), which successfully marries the fields of online convex optimization and sequential filtering. By considering the benefits of these two fields, I have created an algorithm which makes far fewer assumptions on the underlying environment than sequential filters and successfully incorporates dynamics into the model, unlike conventional online convex optimization.
- I propose several extensions of DMD in different settings to learn not only the time-evolving state of the system, but also the parameters of the dynamics driving the system. Previous methods seek to learn the parameters of the dynamic environment and then deterministically run the model forward from the current state to do prediction. By estimating both the dynamical model and the current state, I can make fewer assumptions and be more robust to errors in the assumed generative model.
- For the proposed methods, I present associated regret bounds, which scale sub-linearly in time and characterize how the methods should scale in the scenarios where the dynamical model is known exactly, is known to be within a collection of dynamics, and is a member of a parametric family of functions. These bounds not only show correct scaling in time, but also characterize performance of the algorithms when the dynamical model is slightly misspecified.
- I show how the flexibility of the DMD framework can be applied to many settings such as Compressed Sensing, Poisson video reconstruction, foreground background estimation in video, and network estimation.

- I present novel algorithms and analyses to learn network structure from streaming data using the principles of DMD and the Hawkes' process. By applying my methods to the Hawkes process, I show how network structure can be efficiently learned from streaming data which is a crucial problem in settings such as neurology, seismology, social networks and finance. By applying the DMD method, these estimates are more robust to modeling errors than previous methods.
- I prove novel risk bounds for the regularized maximum likelihood estimator for the log-linear Poisson autoregressive process, which utilizes recent advances in statistical learning theory. This model has been used successfully in real world applications to estimate network structure, and this is the first risk bound which characterizes the amount of data necessary to reach a desired level of accuracy.

Online Convex Optimization in Dynamic Environments

2.1 Introduction

Classical stochastic gradient descent methods, including the least mean squares (LMS) or recursive least squares (RLS) algorithms do not have a natural mechanism for incorporating dynamics. Classical stochastic filtering methods such as Kalman or particle filters or Bayesian updates [22] readily exploit dynamical models for effective prediction and tracking performance. However, these methods are also limited in their applicability because (a) they typically assume an accurate, fully known dynamical model and (b) they rely on strong assumptions regarding a generative model of the observations. Some techniques have been proposed to learn the dynamics [23; 24], but the underlying model still places heavy restrictions on the nature of the data. Performance analysis of these methods usually does not address the impact of “model mismatch”, where the generative models are incorrectly specified.

A contrasting class of prediction methods, receiving widespread recent attention within the machine learning community, is based on an “individual sequence” or

“universal prediction” [25] perspective; these strive to perform provably well on any individual observation sequence without assuming a generative model of the data. *Online convex programming* provides a variety of tools for sequential universal prediction [17; 18; 19; 20]. Here, a Learner measures its predictive performance according to a convex loss function, and with each new observation it computes the negative gradient of the loss and shifts its prediction in that direction. Stochastic gradient descent methods stem from similar principles and have been studied for decades, but recent technical breakthroughs allow these approaches to be understood without strong stochastic assumptions on the data, even in adversarial settings, leading to more efficient and rapidly converging algorithms in many settings.

This chapter describes a novel framework for prediction in the individual sequence setting which incorporates dynamical models – effectively a novel combination of state updating from stochastic filter theory and online convex optimization from universal prediction. We establish tracking regret bounds for our proposed algorithm, *Dynamic Mirror Descent*, which characterize how well we perform relative to some alternative approach (*e.g.*, a computationally intractable batch algorithm) operating on the same data to generate its own predictions, called a “comparator sequence.” Our novel regret bounds scale with the deviation of this comparator sequence from a dynamical model. These bounds simplify to previously shown bounds when there are no dynamics. In addition, we describe methods based on DMD for adapting to the best dynamical model from either a finite or parametric class of candidate models. In these settings, we establish tracking regret bounds which scale with the deviation of a comparator sequence from the *best sequence* of dynamical models.

While our methods and theory apply in a broad range of settings, we are particularly interested in the setting where the dimensionality of the parameter to be estimated is very high. In this regime, the incorporation of both dynamical models and sparsity regularization plays a key role. With this in mind, we focus on a class of

methods which incorporate regularization as well as dynamical modeling. The role of regularization, particularly sparsity regularization, is increasingly well understood in batch settings and has resulted in significant gains in ill-posed and data-starved settings [7; 8; 9; 10]. More recent work has examined the role of sparsity in online methods such as recursive least squares (RLS) algorithms, but do not account for dynamic environments [26].

2.1.1 Organization of Chapter

The remainder of this chapter is structured as follows. In Section 2.2, we formulate the problem and introduce notation used throughout the chapter, and Section 2.3 introduces the *Dynamic Mirror Descent* method, and gives brief comparison to existing methods along with novel tracking regret bounds. This section also describes the application of data-dependent dynamical models and their connection to recent work on online learning with predictable sequences. DMD uses only a single series of dynamical models, but we can use it to choose among a family of candidate dynamical models. This is described for finite families in Section 2.4 using a fixed share algorithm, and for parametric families in Section 2.5. Section 2.6 makes concluding remarks while proofs are relegated to Section 2.7. Experimental results of our methods in a variety of contexts ranging from imaging to self-exciting point processes are compiled in the Chapter 3

2.2 Problem Formulation

The problem of sequential prediction is posed as an iterative game between a Learner and the Environment. At every time point, t , the Learner generates a prediction $\hat{\theta}_t$ from a bounded, closed, convex set $\Theta \subset \mathbb{R}^d$. After the Learner makes a prediction, the Environment reveals the loss function $\ell_t(\cdot)$ where ℓ_t is a convex function which maps the space Θ to the real number line. We will assume that the loss function

is the composition of a convex function $f_t : \Theta \rightarrow \mathbb{R}$ from the Environment and a convex regularization function $r : \Theta \rightarrow \mathbb{R}$ which does not change over time. Frequently the loss function, f_t will measure the accuracy of a prediction compared to some new data point $x_t \in \mathsf{X}$ where X is the domain of possible observations. The regularization function promotes low-dimensional structure (such as sparsity) within the predictions. We additionally assume that we can compute a subgradient of ℓ_t or f_t at any point $\theta \in \Theta$, which we denote $\nabla \ell_t$ and ∇f_t . Thus the Forecaster incurs the loss $\ell_t(\hat{\theta}_t) = f_t(\hat{\theta}_t) + r(\hat{\theta}_t)$.

The goal of the Learner is to create a sequence of predictions $\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_T$ that has a low cumulative loss $\sum_t^T \ell_t(\hat{\theta}_t)$. Because the loss functions are being revealed sequentially, the prediction at each time can only be a function of all previously revealed losses to ensure causality. Thus, the task facing the Learner is to create a new prediction, $\hat{\theta}_{t+1}$, based on the previous prediction and the new loss function $\ell_t(\cdot)$, with the goal of minimizing loss at the next time step. We characterize the efficacy of $\hat{\boldsymbol{\theta}}_T \triangleq (\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_T) \in \Theta^T$ relative to a comparator sequence $\boldsymbol{\theta}_T \triangleq (\theta_1, \theta_2, \dots, \theta_T) \in \Theta^T$ using a concept called *regret*, which measures the difference of the total accumulated loss of the Learner with the total accumulated loss of the comparator:

Definition 2.1 (Regret). *The regret of $\hat{\boldsymbol{\theta}}_T$ with respect to a comparator $\boldsymbol{\theta}_T \in \Theta^T$ is*

$$R_T(\boldsymbol{\theta}_T) \triangleq \sum_{t=1}^T \ell_t(\hat{\theta}_t) - \sum_{t=1}^T \ell_t(\theta_t).$$

Notice that this definition of regret is very general and simply measures the performance of an algorithm versus an arbitrary sequence $\boldsymbol{\theta}_T$. We are particularly interested in comparators which correspond to the output of a *batch* algorithm (with access to all the data simultaneously) that is too computationally complex or memory-intensive for practical big data analysis problems. In this sense, regret

encapsulates how much one loses due to working in an online setting as opposed to a batch setting with full knowledge of all observations.

Much of the online learning literature is focused on algorithms with guaranteed sublinear regret (*e.g.*, $R_T(\boldsymbol{\theta}_T) = O(\sqrt{T})$) in the special case where the comparator $\boldsymbol{\theta}_T$ is constrained so that $\theta_1 = \theta_2 = \dots = \theta_T$. Unfortunately, this is a highly unrealistic constraint in most practical streaming big data settings. The parameters θ_t could correspond to frames in a video or the weights of edges in a dynamic network and by nature are highly variable.

This chapter focuses more generally on arbitrary comparator sequences $\boldsymbol{\theta}_T$ and shows how the regret scales as a function of the temporal variability in that comparator. This idea is typically referred to as “tracking” or “shifting” regret [27; 28], which is closely-related to “adaptive” regret [29; 30]. Existing methods guarantee sublinear regret for all $\boldsymbol{\theta}_T$ which do not vary at all over time, or which change only at a few discrete points in time, or which only vary extremely slowly over time; generally, these regret bounds depend linearly on a *variation* term of the form

$$V(\boldsymbol{\theta}_T) \triangleq \sum_{t=1}^{T-1} \|\theta_{t+1} - \theta_t\|;$$

sublinear regret is only possible for comparator sequences for which this variation is small. *In contrast, the proposed methods in this paper guarantee sublinear regret for different classes of $\boldsymbol{\theta}_T$ that allow quite large temporal variability.*

2.3 Dynamic Mirror Descent

To begin, we propose a simple modification to the “mirror descent” online learning paradigm. Specifically, we incorporate a dynamical model Φ_t at each time t ; this approach is called *Dynamic Mirror Descent*. For now, we assume Φ_t is fixed and known. For example,

- Streaming observations correspond to (potentially distorted) frames in a video sequence, θ_t corresponds to a set of wavelet coefficients for that frame, and Φ_t corresponds to a video motion model at time t .
- Streaming observations correspond to interactions within a social network, θ_t corresponds to the likelihood of each pair of people interacting at time t , and Φ_t captures diurnal patterns and social network evolution models.
- Streaming observations correspond to the price of stocks, θ_t parameterizes a probability distribution governing stock prices, and Φ_t captures underlying autoregressive behavior and side information derived from other financial instruments.

In all of these settings, it is possible to posit a dynamical model Φ_t based on prior knowledge of the application, akin to developing a state space model for stochastic filters.

The DMD algorithm is presented in Algorithm 2.1. In this algorithm $\nabla f_t(\theta)$ denotes an arbitrary subgradient of f_t at θ , $D(\theta\|\hat{\theta}_t)$ is the *Bregman divergence* [31; 32] which measures the distance between θ and $\hat{\theta}$, and $\eta_t > 0$ is a step size parameter.

We may compare the DMD approach with classical online learning methods [17; 29; 19; 18] and more recent regularized formulations [21; 33; 34]. For instance, mirror

Algorithm 2.1 Dynamic mirror descent (DMD) with known dynamics

Given non-increasing sequence of step sizes $\eta_t > 0$

Initialize $\hat{\theta}_1 \in \Theta$.

for $t = 1, \dots, T$ **do**

Observe x_t and incur loss $\ell_t(\hat{\theta}_t)$

Environment produces dynamical model Φ_t

Set

$$\tilde{\theta}_{t+1} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla f_t(\hat{\theta}_t), \theta \rangle + \eta_t r(\theta) + D(\theta\|\hat{\theta}_t) \quad (2.1a)$$

$$\hat{\theta}_{t+1} = \Phi_t(\tilde{\theta}_{t+1}) \quad (2.1b)$$

end for

descent (MD, [17; 18]) sets

$$\hat{\theta}_{t+1} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla \ell_t(\hat{\theta}_t), \theta \rangle + D(\theta \| \hat{\theta}_t), \quad (2.2)$$

and Composite Objective Mirror Descent (COMID¹, [21]) sets

$$\hat{\theta}_{t+1} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla f_t(\hat{\theta}_t), \theta \rangle + \eta_t r(\theta) + D(\theta \| \hat{\theta}_t). \quad (2.3)$$

Specifically, note that if Φ_t is the identity operator for all t , so that $\Phi_t(\theta) \equiv \theta$ for all θ and t , then DMD corresponds exactly to COMID.

Methods like MD and COMID have sublinear regret bounds only for comparators with small $V(\boldsymbol{\theta}_T)$, such as when $\theta_t = \theta$ for some $\theta \in \Theta$ and all t . In contrast, our method is an algorithm which incorporates a dynamical model, denoted $\Phi_t : \Theta \mapsto \Theta$, and admits a tracking regret bound of the form $O(\sqrt{T}[1 + \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t)\|])$ (shown in the next section).

By including Φ_t in the process, we effectively search for a predictor which (a) attempts to minimize the loss and (b) which adheres to the dynamical model Φ_t . The DMD approach effectively includes dynamics into the COMID framework.²

The important feature of Alg. 2.1 is Equation 2.1b, where the predetermined function Φ_t is incorporated in the learning process. The main intuition is that instead of believing the next loss function is well approximated by all the previous loss functions as in MD and COMID, we are instead assuming the loss functions will be well predicted by the trajectory encoded by the functions Φ_t . This results in changing the class of comparators which lead to sublinear regret bounds. MD and

¹ The COMID formulation is helpful when the regularization function $r(\theta)$ promotes sparsity in θ , and helps ensure that $\hat{\theta}_t$ is indeed sparse.

² Rather than considering COMID, we might have used other online optimization algorithms, such as the Regularized Dual Averaging (RDA) method [33], which has been shown to achieve similar performance with more regularized solutions. However, to the best of our knowledge, no tracking or shifting regret bounds have been derived for dual averaging methods (regularized or otherwise). Recent results on the equivalence of COMID and RDA [35] suggest that the bounds derived here might also hold for a variant of RDA, but proving this remains an open problem.

COMID have sublinear regret when the comparator θ_T is static or changes very slowly. However, as we will show, DMD yields sublinear regret when the comparator evolves according to the functions Φ_t or only deviates from these functions by a small amount, meaning $\sum_{t=1}^T \|\theta_{t+1} - \Phi_t(\theta_t)\|$ is small. Thus, while the class of θ_T that has low regret may be the same size as with MD or COMID, the classes contain very different comparator sequences. In later sections we will address how to learn the sequence Φ_t from the data. Additionally, we make no assumption about whether the *data* actually follow these dynamics, but instead we derive a regret bound which scales with how well the *comparator* evolves according to these functions.

2.3.1 Tracking Regret of DMD

Our main result uses the following assumptions:

- Let $\psi : \Theta \rightarrow \mathbb{R}$ denote a continuously differentiable function that is σ -strongly convex for some parameter $\sigma > 0$ and some norm $\|\cdot\|$

$$\psi(\theta_1) \geq \psi(\theta_2) + \langle \nabla \psi(\theta_2), \theta_1 - \theta_2 \rangle + \frac{\sigma}{2} \|\theta_1 - \theta_2\|^2. \quad (2.4)$$

- The Bregman Divergence used in our algorithm is defined as $D(\theta_1 \|\theta_2) \triangleq \psi(\theta_1) - \psi(\theta_2) - \langle \nabla \psi(\theta_2), \theta_1 - \theta_2 \rangle$. Because ψ is σ -strongly convex we have

$$D(\theta_1 \|\theta_2) \geq \frac{\sigma}{2} \|\theta_1 - \theta_2\|^2 \quad (2.5)$$

Additionally, the definition of the Bregman Divergence implies the following relationship: for all $\theta_1, \theta_2, \theta_3 \in \Theta$

$$D(\theta_1 \|\theta_2) = D(\theta_3 \|\theta_2) + D(\theta_1 \|\theta_3) + \langle \nabla \psi(\theta_2) - \nabla \psi(\theta_3), \theta_3 - \theta_1 \rangle. \quad (2.6)$$

- For all $t = 1, \dots, T$ the functions ℓ_t and ψ are Lipschitz with constants G and M respectively, such that $\|\nabla \ell_t(\theta)\|_* \leq G$ and $\|\nabla \psi(\theta)\|_* \leq M$ for all $\theta \in \Theta$. The function $\|\cdot\|_*$ used in these assumptions is the dual to the norm that used in Equation 2.4.

- There exists a constant D_{\max} such that $D(\theta_1\|\theta_2) \leq D_{\max}$ for all $\theta_1, \theta_2 \in \Theta$.
- For all $t = 1, \dots, T$, the transformation Φ_t has a maximum distortion factor Δ_Φ such that $D(\Phi_t(\theta_1)\|\Phi_t(\theta_2)) - D(\theta_1\|\theta_2) \leq \Delta_\Phi$ for all $\theta_1, \theta_2 \in \Theta$. When $\Delta_\Phi \leq 0$ for all t , we say that Φ_t satisfies the contractive property.

Theorem 2.2 (Tracking regret of dynamic mirror descent). *Let Φ_t be a dynamical model such that $\Delta_\Phi \leq 0$ for $t = 1, 2, \dots, T$ with respect to the Bregman divergence used in Equation 2.1. Let the sequence $\hat{\boldsymbol{\theta}}_T$ be generated using Alg. 2.1 using a non-increasing series $\eta_{t+1} \leq \eta_t$, with a convex, Lipschitz function ℓ_t on a closed, convex, bounded set Θ , and let $\boldsymbol{\theta}_T$ be an arbitrary sequence in Θ^T . Then*

$$R_T(\boldsymbol{\theta}_T) \leq \frac{D_{\max}}{\eta_{T+1}} + \frac{2M}{\eta_T} V_\Phi(\boldsymbol{\theta}_T) + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t$$

$$\text{with } V_\Phi(\boldsymbol{\theta}_T) \triangleq \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t)\|$$

where $V_\Phi(\boldsymbol{\theta}_T)$ measures variations or deviations of the comparator sequence $\boldsymbol{\theta}_T$ from the sequence of dynamical models $\Phi_1, \Phi_2, \dots, \Phi_T$. If $\eta_t \propto \frac{1}{\sqrt{t}}$ or $\eta_t \propto \frac{1}{\sqrt{T}}$, then for some $C > 0$ independent of T ,

$$R_T(\boldsymbol{\theta}_T) \leq C\sqrt{T} (1 + V_\Phi(\boldsymbol{\theta}_T))$$

This bound scales with the comparator sequence's deviation from the sequence of dynamical models $\{\Phi_t\}_{t>0}$ – a stark contrast to previous tracking regret bounds which are only sublinear for comparators which change slowly with time or at a small number of distinct time instances. Note that when Φ_t corresponds to an identity operator, the bound in Theorem 2.2 corresponds to existing tracking or shifting regret bounds [20; 28].

It is intuitively satisfying that this measure of variation, $V_\Phi(\boldsymbol{\theta}_T)$, appears in the tracking regret bound. First, if the comparator sequence evolves approximately like

$\theta_{t+1} = \Phi_t(\theta_t)$, this variation term will be very small, leading to low regret. This fact holds whether Φ_t is part of the generative model for the observations or not. Secondly, we can get a dynamic analog of static regret, where we enforce $V_\Phi(\boldsymbol{\theta}_T) = 0$. This is equivalent to saying that the batch comparator is fitting the best single trajectory using Φ_t instead of the best single point. Using this, we would recover a bound analogous to a static regret bound in a stationary setting.

The condition that $\Delta_\Phi \leq 0$ is similar to requiring that Φ_t be a contractive mapping. This restriction is important; without it, any poor prediction made at one time step could be exacerbated by repeated application of the dynamics. For instance, linear dynamic models with all eigenvalues less than or equal to unity satisfy this condition with respect to the squared ℓ_2 Bregman Divergence, similar in spirit to restrictions made in more classical adaptive filtering work such as [36]. Notice also that if $\Phi_t(\theta) = \theta$ in for all t , then Theorem 2.2 gives a novel, previously unknown tracking regret bound for COMID.

2.3.2 Data-dependent Dynamics

An interesting example of dynamical models is the class of data-dependent dynamical models. In this regime the state of the system at a given time is not only a function of the previous state, but also the actual observations. One key example of this scenario arises in self-exciting point processes, where the state of the system is directly related to the previous observations. Our algorithm can account for such models since the function $\Phi_t(\theta)$ is time varying, and therefore can implicitly depend on all data up to time t , i.e. $\Phi_t(\theta) = \Phi_t(\theta; x_1, x_2, \dots, x_t)$. Our regret bounds therefore scale with how well the comparator series matches these data dependent dynamics:

$$R_T(\boldsymbol{\theta}_T) \leq C \left(\sqrt{T} \left[1 + \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t; x_1, \dots, x_t)\| \right] \right).$$

Notice now that the data plays a part in the regret bounds, whereas before we only measured the variation of the comparator. Data-dependent regret bounds are not new. Concurrent related work considers online algorithms where the data sequence is described by a “predictable process” [37]. The basic idea of that work is that if one has a sequence of functions M_t which predict x_t based on x_1, x_2, \dots, x_{t-1} , then the output of a standard online optimization routine should be combined with the predictor generated by M_t to yield tighter regret bounds that scale with $(\sum_t \|x_t - M_t(x_1, \dots, x_{t-1})\|^2)^{1/2}$. However, [37] only works with static regret (*i.e.*, regret with respect to a static comparator) and their regret has a variation term that expresses the deviation of the *input data* from the underlying process. In contrast, our tracking regret bounds scale with the deviation of a *comparator sequence* from a prediction model.

2.4 Prediction with a Finite Family of Dynamical Models

The DMD algorithm in the previous section uses a single sequence of dynamical models. In practice, however, we may not know the best dynamical model to use, or the best model may change over time in nonstationary environments. To address this challenge, we assume a finite set of candidate dynamical models $\{\Phi_t^{(1)}, \Phi_t^{(2)}, \dots, \Phi_t^{(N)}\}$ at every time t , and describe a procedure which uses this collection to adapt to nonstationarities in the environment. In particular we establish tracking regret bounds which scale not with the deviation of a comparator from a single dynamical model, but with how it deviates from a *series of different dynamical models on different time intervals* with at most m switches. These switches define $m + 1$ different time segments $[t_i, t_{i+1} - 1]$ with time points $1 = t_1 < \dots < t_{m+2} = T$. We can bound the regret associated with the best dynamical model on each time segment and then bound the overall regret using a Prediction with Experts Advice algorithm.

Our *dynamic fixed share* (DFS) estimate is presented in Algorithm 2.2. Let $\tilde{\theta}_t^{(i)}$

Algorithm 2.2 Dynamic fixed share (DFS)

Given decreasing sequence of step sizes $\eta_t > 0$ and $\rho_t > 0$

Initialize $\hat{\theta}_1 \in \Theta$, $\hat{\theta}_1^{(i)} \in \Theta$ and $w_1^{(i)} = \frac{1}{N}$ for $i = 1, \dots, N$, $\gamma \in (0, 1)$, and $\eta_t, \rho_t > 0$.

for $t = 1, \dots, T$ **do**

Observe x_t and incur loss $\ell_t(\hat{\theta}_t)$

Receive dynamical model $\Phi_t^{(i)}$ for $i = 1, \dots, N$

for $i = 1, \dots, N$ **do**

Set

$$\tilde{w}_{t+1}^{(i)} = \frac{w_t^{(i)} \exp\left(-\rho_t \ell_t\left(\hat{\theta}_t^{(i)}\right)\right)}{\sum_{j=1}^N w_t^{(j)} \exp\left(-\rho_t \ell_t\left(\hat{\theta}_t^{(j)}\right)\right)}$$

$$w_{t+1}^{(i)} = \frac{\gamma}{N} + (1 - \gamma) \tilde{w}_{t+1}^{(i)}$$

$$\tilde{\theta}_{t+1}^{(i)} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla f_t(\hat{\theta}_t^{(i)}), \theta \rangle + \eta_t r(\theta) + D(\theta \| \hat{\theta}_t^{(i)})$$

$$\hat{\theta}_{t+1}^{(i)} = \Phi_t^{(i)}(\tilde{\theta}_{t+1}^{(i)})$$

end for

Set

$$\hat{\theta}_{t+1} = \sum_{i=1}^N w_{t+1}^{(i)} \hat{\theta}_{t+1}^{(i)}$$

end for

denote the output of Equation 2.1b at time t using the sequence of dynamical models $\Phi_1^{(i)}, \Phi_2^{(i)}, \dots, \Phi_t^{(i)}$; we choose $\hat{\theta}_t$ by using the Fixed Share forecaster on these outputs.³

In Fixed Share, each expert (here, each sequence of candidate dynamical models) is assigned a weight that is inversely proportional to its cumulative loss at that point yet with some weight shared amongst all the experts. Additionally, there is a parameter

³ There are many algorithms from the Prediction with Expert Advice literature which can be used to form a single prediction from the predictions created by the set of dynamical models. We use the Fixed Share algorithm [38] as a means to combine estimates with different dynamics; however, other methods could be used with various tradeoffs. One of the primary drawbacks of the Fixed Share algorithm is that an upper bound on the number of switches m must be known a priori. However, this method has a simple implementation and tracking regret bounds. One common alternative to Fixed Share allows the switching parameter (γ in Alg. 2.2) to decrease to zero as the algorithm runs [39; 40]. This has the benefit of not requiring knowledge about the number of switches, but comes at the price of higher regret. Alternative expert advice algorithms exist which decrease the regret but increase the computational complexity. For a thorough treatment of existing methods see [41].

γ which is the amount of weight divided evenly amongst experts so that an expert with previously high loss quickly regain enough weight to become the leader [38; 28]. In this update, $\gamma \in (0, 1)$ is how much of the weight is shared amongst the experts. Notice each expert “donates” a γ fraction of its weight which is then replaced by γ/N . For experts with large weights this will cause some weight to be lost, but ensures that a minimum weight of γ/N is attained for each expert. Sharing weight allows fast switching between experts.

Theorem 2.3 (Tracking regret of DFS algorithm). *Assume all the candidate dynamic sequences are contractive such that $\Delta_\Phi \leq 0$ for $\Phi_t^{(i)}$ for all $t = 1, \dots, T$ and $i = 1, \dots, N$ with respect to the Bregman divergence in Alg 2.1. Then for some $C > 0$, the dynamic fixed share algorithm in Algorithm 2.2 with parameter γ set equal to $\frac{m}{T-1}$, $\rho_t = \sqrt{\frac{8((m+1)\log(N)+m\log(T)+1)}{T}}$ and $\eta_t \propto 1/\sqrt{t}$ or $\eta_t \propto 1/\sqrt{T}$ with a convex, Lipschitz function ℓ_t on a closed, bounded, convex set Θ , has tracking regret*

$$R_T(\boldsymbol{\theta}_T) = \sum_{t=1}^T \ell_t(\hat{\theta}_t) - \sum_{t=1}^T \ell_t(\theta_t) \leq C \left(\sqrt{T} \left(\sqrt{(m+1)\log N + m\log T} + V^{(m+1)}(\boldsymbol{\theta}_T) \right) \right),$$

where

$$V^{(m+1)}(\boldsymbol{\theta}_T) \triangleq \min_{t_2, \dots, t_{m+1}} \sum_{k=1}^{m+1} \min_{i_k \in \{1, \dots, N\}} \sum_{t=t_k}^{t_{k+1}-1} \|\theta_{t+1} - \Phi_t^{(i_k)}(\theta_t)\|$$

measures the deviation of the sequence $\boldsymbol{\theta}_T$ from the best sequence of dynamical models with at most m switches (where m does not depend on T).

The choice of m is important, as low values of m will have low regret but for a smaller class of comparators, comparators with a small number of switches. Oppositely, larger values of m will have low regret for a larger class of comparators, but there is an overhead to be paid in the constant terms. Note that the family of comparator sequences $\boldsymbol{\theta}_T$ for which $R_T(\boldsymbol{\theta}_T)$ scales sublinearly in T is *significantly*

larger than the set of comparators yielding sublinear regret for MD. This is because $V^{(m+1)}(\boldsymbol{\theta}_T) \leq V_{\Phi_{i,t}}(\boldsymbol{\theta}_T)$ for any fixed $i \in \{1, \dots, N\}$, thus this approach yields a lower variation term than using a fixed dynamical model. However, we incur some loss by not knowing the optimal number of switches m or when the optimal switching times are.

2.5 Parametric Dynamical Models

Rather than having a finite family of dynamical models, as we did in Section 2.4, we may consider a parametric family of dynamical models, where the parameter $\alpha \in \mathbb{R}^n$ of Φ_t is allowed to vary across a closed, bounded, convex domain, denoted \mathcal{A} . In other words, we consider $\Phi_t : \Theta \times \mathcal{A} \mapsto \Theta$. In this context we would like to *jointly* predict both α and θ .

We consider two approaches. First, in Section 2.5.1 we consider tracking only a finite subset of the possible model parameters, in a manner similar to when we had a finite collection of possible dynamical models, which provide a “covering” of the parameter space. In this case, the overall regret and computational complexity both depend on the resolution of the covering set. Second, in Section 2.5.2, we consider a special family of additive dynamical models; in this setting, we can efficiently learn the optimal dynamics.

2.5.1 Covering the Set of Dynamical Models

In this section we show that by tracking a subset which appropriately covers the entire space of candidate models, we can bound the overall regret, as well as bound the number of parameter values we have to track, and the inherent tradeoff between the two. We propose to choose a finite collection of parameters from a closed, convex set \mathcal{A} and perform DFS (Alg. 2.2) on this collection. For the sake of exposition, we specifically consider the case where the true dynamical model $\alpha^* \in \mathcal{A}$ is unchanging

in time and use DFS with $m = 0$. (Fixed share with $m = 0$ amounts to the Exponentially Weighted Averaging Forecaster [42; 29; 20].) In the below, for any $\alpha \in \mathcal{A}$, let

$$V_{\Phi}(\boldsymbol{\theta}_T, \alpha) \triangleq \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t, \alpha)\|.$$

Theorem 2.4 (Covering sets of dynamics parameter space). *Let $\varepsilon_N > 0$ and \mathcal{A}_N denote a covering set for \mathcal{A} with cardinality N , such that for every $\alpha \in \mathcal{A}$, there is some $\alpha' \in \mathcal{A}_N$ such that $\|\alpha - \alpha'\| \leq \varepsilon_N$. Define candidate dynamical models as $\Phi_t(\cdot, \alpha)$ for $\alpha \in \mathcal{A}_N$ and assume they are all contractive with respect to the Bregman Divergence used in Alg. 2.1. If $\|\Phi_t(\theta, \alpha) - \Phi_t(\theta, \beta)\| \leq L\|\alpha - \beta\|$ for some $L > 0$ for all $\alpha, \beta \in \mathcal{A}$, then for some constant $C > 0$, the Alg 2.2 with $\eta_t = \frac{1}{\sqrt{t}}, \rho_t = \sqrt{\frac{2\log(N)}{T}}, \gamma = 0$ yields a tracking regret bounded by*

$$C \left(\sqrt{T} \left[\sqrt{\log(N)} + \min_{\alpha \in \mathcal{A}} V_{\Phi}(\boldsymbol{\theta}_T, \alpha) + T\varepsilon_N \right] \right).$$

Intuitively, we know that if we set ε_N to be very small we will have good performance because any possible parameter value $\alpha \in \mathcal{A}$ would have to be close to a candidate dynamic; however, we would need to choose many candidates. Conversely, if we run DFS on only a few candidate models, it will be computationally much more efficient but our total regret will grow due to parameter mismatch.

Corollary 2.5. *Assume $\mathcal{A} \subseteq [A_{\min}, A_{\max}]^n$, and let $\rho > 0$ be given. Let $k = \lceil (A_{\max} - A_{\min})nT^{\rho}/2 \rceil$ and $\partial = (A_{\max} - A_{\min})/(2k)$; let $\mathcal{A}_N = \{A_{\min} + \partial, A_{\min} + 3\partial, \dots, A_{\min}(2k - 1)\partial\}^n$ correspond to an n -dimensional grid with k^n grid points over \mathcal{A} . Then*

$$\max_{\alpha \in \mathcal{A}} \min_{\alpha' \in \mathcal{A}_N} \|\alpha - \alpha'\|_1 \leq T^{-\rho}.$$

Additionally, the total number of grid points is upper bounded by

$$N \leq \left(\frac{(A_{\max} - A_{\min})nT^{\rho}}{2} + 1 \right)^n = O(T^{\rho n})$$

Under the assumptions of Theorem 2.4, with this set \mathcal{A}_N and using the fact that norms are equivalent on finite-dimensional vectors (i.e., there's a finite $Z > 0$ such that $\|\alpha - \beta\|_1 \leq Z\|\alpha - \beta\|$ for any $\alpha, \beta \in \mathcal{A}$ for any norm), we get the following bound on regret for some constant $C > 0$.

$$R_T(\boldsymbol{\theta}_T) \leq C \left(\sqrt{T} \left[\sqrt{\rho n \log(T)} + \min_{\alpha \in \mathcal{A}} V_{\Phi}(\boldsymbol{\theta}_T, \alpha) \right] + T^{1-\rho n} \right)$$

Here we have an explicit tradeoff between regret and computational accuracy controlled by ρ , since the computational complexity is linear in $N = O(T^{\rho n})$.

We can further control the tradeoff between computation complexity and performance by allowing ε_N to vary in time. This could be done by using the doubling trick, setting temporary time horizons, and then refining the grid once the temporary time horizon is reached using a slightly different experts algorithm which could account for the changing number of experts as in [43].

2.5.2 Additive Dynamics in Exponential Families

The approach described above for generating a covering set of dynamical models may be effective when the dimension of parameters is small; however, in higher dimensions, this approach can require significant computational resources. In this section, we consider an alternative approach that only requires the computation of predictions for a single dynamical model. We will see that in some settings, the prediction produced by DMD and a certain set of parameters for the dynamic model can quickly be converted to the prediction for a different set of parameters. While the method described in this section is efficient and admits strong regrets bounds, it is applicable only for loss functions derived from exponential families.

The basics of exponential families are described in [44; 45], and Mirror Descent in this setting is explored in [46; 47]. We assume some $\phi : \mathbf{X} \rightarrow \mathbb{R}^d$ which is a

measurable function of the data, and let ϕ_k , $k = 1, 2, \dots, d$, denote its components:

$$\phi(x) = (\phi_1(x), \dots, \phi_d(x))^T.$$

We use the specific loss function

$$\ell_t(\theta) = -\log p_\theta(x_t) \tag{2.7a}$$

where

$$p_\theta(x) \triangleq \exp\{\langle \theta, \phi(x) \rangle - Z(\theta)\} \tag{2.7b}$$

for a sufficient statistic ϕ and $Z(\theta) \triangleq \log \int \exp\{\langle \theta, \phi(x) \rangle\} dx$, known as the *log-partition function*, ensures that $p_\theta(x)$ integrates to a constant independent of θ . Furthermore, as in [46; 47], we use the Bregman divergence corresponding to the Kullback-Leibler divergence between two members of the exponential family, defined by the log-partition function:

$$D(\theta_1 \parallel \theta_2) = Z(\theta_1) - Z(\theta_2) - \langle \nabla Z(\theta_2), \theta_1 - \theta_2 \rangle.$$

In our analysis we will be using the *Legendre–Fenchel dual* of Z [48; 49]:

$$Z^*(\lambda) \triangleq \sup_{\theta \in \Theta} \{\langle \lambda, \theta \rangle - Z(\theta)\}.$$

Let Θ^* denote the image of $\text{Int } \Theta$ under the gradient mapping ∇Z i.e. $\Theta^* = \nabla Z(\text{Int } \Theta)$. An important fact is that the gradient mappings ∇Z and ∇Z^* are inverses of one another [18; 20; 50]:

$$\left. \begin{array}{l} \nabla Z^*(\nabla Z(\theta)) = \theta \\ \nabla Z(\nabla Z^*(\lambda)) = \lambda \end{array} \right\} \quad \forall \theta \in \text{Int } \Theta, \lambda \in \text{Int } \Theta^*$$

Following [20], we may refer to the points in $\text{Int } \Theta$ as the *primal points* and to their images under ∇Z as the *dual points*. For simplicity of notation, in the sequel we will write $\lambda = \nabla Z(\theta)$, $\theta = \nabla Z^*(\lambda)$, $\hat{\lambda}_t = \nabla Z(\hat{\theta}_t)$, etc.

Additionally, we will use a dynamical model that takes on a specific form:

$$\Phi_t(\theta, \alpha) = \nabla Z^*(A_t \nabla Z(\theta) + B_t \alpha + c_t) \quad (2.8)$$

for $\theta \in \text{Int } \Theta$, $c_t \in \mathbb{R}^d$, $\alpha \in \mathbb{R}^n$, $A_t \in \mathbb{R}^{d \times d}$, and $B_t \in \mathbb{R}^{d \times n}$ where A_t, B_t and c_t are considered known. This dynamical model encompasses some important scenarios. For instance if the log-partition function is the regular ℓ_2 function, this model includes all dynamics in the form of $\theta_{t+1} = A_t \theta_t + B_t \alpha + c_t$, which is akin to an autoregressive moving average model as in Section 3.5. These types of models could also be used to push estimates towards a known temporal structure, e.g. a diurnal pattern with unknown amplitude as in Section 3.7. Additionally, B_t could encode additional side information. Using these dynamics, we let $\hat{\theta}_t^{(\alpha)}$ denote the output of DMD (Alg. 2.1) at time t using parameters α and $\hat{\lambda}_t^{(\alpha)}$ be its dual. Under all these conditions, we have the following Lemma.

Lemma 2.6. *For any $\alpha, \beta \in \mathcal{A}$, let $\hat{\lambda}_1^{(\alpha)} = \hat{\lambda}_1^{(\beta)}$ be the duals of the initial prediction for DMD and $K_1 = \mathbf{0} \in \mathbb{R}^{d \times n}$. Additionally assume that the minimizer of equation 2.1a is a point in $\text{Int } \Theta$ for any parameter $\alpha \in \mathcal{A}$. Then the DMD prediction under a dynamical model parameterized by α can be calculated directly from the DMD prediction under a dynamical model parameterized by β for $t > 0$ as*

$$\hat{\lambda}_t^{(\alpha)} = \hat{\lambda}_t^{(\beta)} + K_t(\alpha - \beta)$$

$$\text{where } K_t = (1 - \eta_{t-1})A_{t-1}K_{t-1} + B_{t-1}.$$

From Lemma 2.6, we see that the prediction for dynamical model α can be computed simply from the prediction using parameters β and the value K_t . This is a significant computational gain compared to DFS, where we had to keep track of predictions for each candidate dynamical model individually and therefore needed to bound the number of experts for tractability.

Algorithm 2.3 leverages Lemma 2.6 to simultaneously track both $\hat{\theta}_t$ and the best

Algorithm 2.3 Dynamic mirror descent (DMD) with parametric additive dynamics

Given decreasing sequence of step sizes $\rho_t, \eta_t > 0$
 Initialize $\hat{\alpha}_1 = \mathbf{0}$, $K_1 = \mathbf{0}$, $\hat{\theta}_1 \in \Theta$, $\hat{\lambda}_1 = \nabla Z(\hat{\theta}_1)$
for $t = 1, \dots, T$ **do**
 Observe x_t
 Incur loss $\ell_t(\hat{\theta}_t) = -\langle \hat{\theta}_t, \phi(x_t) \rangle + Z(\hat{\theta}_t)$
 Set $g_t(\alpha) = \tilde{\ell}_t(\hat{\lambda}_t^{(\alpha)}) \equiv \ell_t(\hat{\theta}_t^{(\alpha)})$
 Set $\hat{\alpha}_{t+1} = \text{proj}_{\mathcal{A}}(\hat{\alpha}_t - \rho_t \nabla g_t(\hat{\alpha}_t))$
 Set $\lambda'_{t+1} = \hat{\lambda}_t + K_t(\hat{\alpha}_{t+1} - \hat{\alpha}_t)$
 Set $\tilde{\lambda}_{t+1} = (1 - \eta_t)\lambda'_{t+1} + \eta_t \phi(x_t)$
 Set $\hat{\theta}_{t+1} = \nabla Z^*(\tilde{\lambda}_{t+1})$
 Set $\hat{\theta}_{t+1} = \Phi_t(\hat{\theta}_{t+1}, \hat{\alpha}_{t+1})$
 Set $K_{t+1} = (1 - \eta_t)A_t K_t + B_t$
end for

dynamical model parameter α . In this algorithm, $\tilde{\ell}_t$ is the function defined as

$$\tilde{\ell}_t(\lambda) \triangleq \ell_t(\nabla Z^*(\lambda)) \equiv \ell_t(\theta).$$

The basic idea is the following: we use Online Gradient Descent to compute an estimate of the best dynamical model parameter, compute the DMD prediction associated with that parameter, and then use DMD to update that prediction for the next round.

Theorem 2.7. *Assume that the observation space X is bounded. Let $\Theta \subset \mathbb{R}^d$ be a bounded, convex set satisfying the following properties for a given constant $H > 0$:*

- For all $\theta \in \Theta$,

$$Z(\theta) \triangleq \int_{\mathsf{X}} \exp\{\langle \theta, \phi(x) \rangle\} d\nu(x) < +\infty.$$

- For all $\theta \in \Theta$, $\nabla^2 Z(\theta) \geq 2HI_{d \times d}$.
- Let f_t denote the objective function in (2.1a). For every $x \in \mathsf{X}$ and $t \in \{1, 2, 3, \dots\}$, the solution to $\arg \min_{\theta \in \Theta} f_t(\theta)$ occurs where $\nabla f_t = \mathbf{0}$.

If the assumptions of Lemma 2.6 hold, and $\Phi_t(\theta, \alpha)$ is contractive for all $\alpha \in \mathcal{A}$ with respect to the Bregman Divergence induced by $Z(\theta)$, the loss function is of the form (2.7) and $\tilde{\ell}_t(\lambda) \triangleq \ell_t(\nabla Z^*(\lambda))$ is convex in λ , and $\eta_t, \rho_t \propto 1/\sqrt{t}$ or $1/\sqrt{T}$, then the regret for any comparator sequence $\boldsymbol{\theta}_T \in \Theta^T$ associated with Algorithm 2.3 for dynamical models of the form (2.8) is

$$R_T(\boldsymbol{\theta}_T) \leq C\sqrt{T} \left(1 + \min_{\alpha \in \mathcal{A}} V_{\Phi}(\boldsymbol{\theta}_T, \alpha) \right)$$

for some constant $C > 0$.

The condition that $\tilde{\ell}_t(\lambda)$ is convex is a sufficient condition to ensure that $g_t(\alpha)$ is convex, which we need in order to search for the optimal value of α . While this may not hold true for all exponential family distributions, it holds for many common choices such as Gaussian, Poisson, Binomial, Exponential and many others. Theorem 2.7 shows that Algorithm 2.3 allows us to simultaneously track predictions and dynamics, and we perform nearly as well as if we knew the best dynamical models for the entire sequence in hindsight. While this approach is only applicable for specific forms of the loss functions and dynamical models, those forms arise in a wide variety of practical problems. Additionally, this methods allows for a much larger class of comparators which would yield sublinear regret. In fact, we now have an algorithm with sublinear regret for any class of comparators which has a low variation term $\sum_{t=1}^T \|\theta_{t+1} - \Phi_t(\theta_t, \alpha)\|$ for any value $\alpha \in \mathcal{A}$.

2.6 Conclusions

Processing high-velocity streams of high-dimensional data is a central challenge to big data analysis. Scientists and engineers continue to develop sensors capable of generating large quantities of data, but often only a small fraction of that data is carefully examined or analyzed. Fast algorithms for sifting through such data can

help analysts track dynamic environments and identify important subsets of the data which are inconsistent with past observations.

In this Chapter we have proposed a novel online optimization method, called Dynamic Mirror Descent (DMD), which incorporates dynamical models into the prediction process and yields low regret bounds for broad classes of comparator sequences. The proposed methods are applicable for a wide variety of observation models, noise distributions, and dynamical models. There is no assumption within our analysis that there is a “true” known underlying dynamical model, or that the best dynamical model is unchanging with time. The proposed Dynamic Fixed Share (DFS) algorithm adaptively selects the most promising dynamical model from a family of candidates at each time step. Additionally we show methods which learn in parametric families of dynamical models. The experiments in Chapter 3 demonstrate DMD showing strong tracking behavior even when underlying dynamical models are switching, in such applications as dynamic texture analysis, compressive video, and self-exciting point process analysis.

2.7 Proofs

2.7.1 Proof of Theorem 2.2

The proof of Theorem 2.2 shares some ideas with the tracking regret bounds of [19], but uses properties of the Bregman Divergence to eliminate some terms, while additionally incorporating dynamics. We employ the following lemma.

Lemma 2.8. *Let the sequence $\hat{\theta}_T$ be as in Alg. 2.1, and let θ_T be an arbitrary sequence in Θ^T ; then*

$$\ell_t(\hat{\theta}_t) - \ell_t(\theta_t) \leq \frac{1}{\eta_t} \left[D(\theta_t \| \hat{\theta}_t) - D(\theta_{t+1} \| \hat{\theta}_{t+1}) \right] + \frac{\Delta_\Phi}{\eta_t} + \frac{2M}{\eta_t} \|\theta_{t+1} - \Phi_t(\theta_t)\| + \frac{\eta_t}{2\sigma} G^2.$$

Proof of Lemma 2.8: The optimality condition of (2.1a) implies

$$\langle \nabla f_t(\hat{\theta}_t) + \nabla r(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle \leq \frac{1}{\eta_t} \langle \nabla \psi(\hat{\theta}_t) - \nabla \psi(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle. \quad (2.9)$$

The proof has a similar structure to that in [21]

$$f_t(\hat{\theta}_t) - f_t(\theta_t) + r(\hat{\theta}_t) - r(\theta_t) \quad (2.10a)$$

$$= f_t(\hat{\theta}_t) - f_t(\theta_t) + r(\hat{\theta}_t) - r(\tilde{\theta}_{t+1}) + r(\tilde{\theta}_{t+1}) - r(\theta_t) \quad (2.10b)$$

$$\leq \langle \nabla f_t(\hat{\theta}_t), \hat{\theta}_t - \theta_t \rangle + \langle \nabla r(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle + \langle \nabla r(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle \quad (2.10c)$$

$$\leq \langle \nabla f_t(\hat{\theta}_t) + \nabla r(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle + \langle \nabla f(\hat{\theta}_t) + \nabla r(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \quad (2.10d)$$

$$\leq \frac{1}{\eta_t} \langle \nabla \psi(\hat{\theta}_t) - \nabla \psi(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle + \langle \nabla f_t(\hat{\theta}_t) + \nabla r(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \quad (2.10e)$$

$$= \frac{1}{\eta_t} \left(D(\theta_t \| \hat{\theta}_t) - D(\theta_t \| \tilde{\theta}_{t+1}) - D(\tilde{\theta}_{t+1} \| \hat{\theta}_t) \right) + \langle \nabla f_t(\hat{\theta}_t) + \nabla r(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \quad (2.10f)$$

$$\begin{aligned} &= \frac{1}{\eta_t} D(\theta_t \| \hat{\theta}_t) - D(\theta_{t+1} \| \hat{\theta}_{t+1}) + \frac{1}{\eta_t} \underbrace{D(\theta_{t+1} \| \hat{\theta}_{t+1}) - D(\Phi_t(\theta_t) \| \hat{\theta}_{t+1})}_{T_1} \\ &\quad + \frac{1}{\eta_t} \underbrace{D(\Phi_t(\theta_t) \| \hat{\theta}_{t+1}) - D(\theta_t \| \tilde{\theta}_{t+1})}_{T_2} \\ &\quad - \frac{1}{\eta_t} \underbrace{D(\tilde{\theta}_{t+1} \| \hat{\theta}_t) + \langle \nabla f_t(\hat{\theta}_t) + \nabla r(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle}_{T_3} \end{aligned}$$

where (2.10c) follows from the convexity of f_t and r , (2.10e) follows from the optimality condition in (2.1a), and (2.10f) follows from (2.6). Each of these terms can be bounded individually, and then recombined to complete the proof.

$$T_1 = \psi(\theta_{t+1}) - \psi(\Phi_t(\theta_t)) - \langle \nabla \psi(\hat{\theta}_{t+1}), \theta_{t+1} - \Phi_t(\theta_t) \rangle \quad (2.11a)$$

$$\leq \langle \nabla \psi(\theta_{t+1}) - \nabla \psi(\hat{\theta}_{t+1}), \theta_{t+1} - \Phi_t(\theta_t) \rangle \quad (2.11b)$$

$$\leq \|\nabla \psi(\theta_{t+1}) - \nabla \psi(\hat{\theta}_{t+1})\|_* \|\theta_{t+1} - \Phi_t(\theta_t)\| \quad (2.11c)$$

$$\leq 2M \|\theta_{t+1} - \Phi_t(\theta_t)\| \quad (2.11d)$$

$$T_2 = D(\Phi_t(\theta_t) \|\Phi_t(\tilde{\theta}_{t+1})) - D(\theta_t \|\tilde{\theta}_{t+1}) \leq \Delta_\Phi \quad (2.11e)$$

$$T_3 \leq -\frac{\sigma}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|^2 + \|\nabla f_t(\hat{\theta}_t) + \nabla r(\hat{\theta}_t)\|_* \|\hat{\theta}_t - \tilde{\theta}_{t+1}\| \quad (2.11f)$$

$$\leq -\frac{\sigma}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|^2 + \frac{\sigma}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|^2 + \frac{\eta_t}{2\sigma} G^2, \quad (2.11g)$$

where (2.11b) is due to the convexity of ψ and (2.11c) is from the Cauchy-Schwarz inequality. Additionally, (2.11f) is due to (2.5) and (2.11g) uses Young's Inequality [51, Prob 9.1] which states $ab \leq \frac{a^2}{2\epsilon} + \frac{b^2\epsilon}{2}$. Combining these inequalities with (2.11a) gives the lemma as it is stated. \square

The proof of the theorem concludes by summing the bounds of Lemma 2.8 over time. Denote $D_t \triangleq D(\theta_t \|\hat{\theta}_t)$ and $V_t \triangleq \|\theta_{t+1} - \Phi_t(\theta_t)\|$. Remember, we have assumed that $\Delta_\Phi \leq 0$.

$$\begin{aligned} R_T(\boldsymbol{\theta}_T) &\leq \sum_{t=1}^T \left(\frac{D_t}{\eta_t} - \frac{D_{t+1}}{\eta_{t+1}} \right) + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t + D_{\max} \sum_{t=1}^T \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t} \right) + \sum_{t=1}^T \frac{2M}{\eta_t} V_t \\ &\leq \frac{D_1}{\eta_1} - \frac{D_{T+1}}{\eta_{T+1}} + D_{\max} \left(\frac{1}{\eta_{T+1}} - \frac{1}{\eta_1} \right) + \frac{2M}{\eta_T} V_\Phi(\boldsymbol{\theta}_T) + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t \\ &\leq \frac{D_{\max}}{\eta_{T+1}} + \frac{2M}{\eta_T} V_\Phi(\boldsymbol{\theta}_T) + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t. \quad \square \end{aligned}$$

2.7.2 Proof of Theorem 2.3

The tracking regret can be decomposed as:

$$\begin{aligned}
R_T(\boldsymbol{\theta}_T) &= \underbrace{\sum_{t=1}^T \ell_t(\hat{\theta}_t) - \min_{\substack{i_1, \dots, i_T \\ \sum_{t=1}^{T-1} \mathbf{1}[i_t \neq i_{t+1}] \leq m}} \sum_{t=1}^T \ell_t(\hat{\theta}_t^{(i_t)})}_{T_4} \\
&+ \underbrace{\min_{\substack{i_1, \dots, i_T \\ \sum_{t=1}^{T-1} \mathbf{1}[i_t \neq i_{t+1}] \leq m}} \sum_{t=1}^T \ell_t(\hat{\theta}_t^{(i_t)}) - \min_{\theta \in \Theta} \sum_{t=1}^T \ell_t(\theta)}_{T_5} \tag{2.12}
\end{aligned}$$

where the minimization in the second term of T_4 and first term of T_5 is with respect to sequences of dynamical models with at most m switches, such that $\sum_{t=1}^T \mathbf{1}[i_t \neq i_{t+1}] \leq m$. In (2.12), T_4 corresponds to the tracking regret of our algorithm relative to the best sequence of dynamical models within the DMD framework, and T_5 is the regret of that sequence relative to the best comparator in the class Θ_m . We use Corollary 3 of [28] to bound T_4 , using $\rho_t = \sqrt{\frac{8((m+1)\log(N)+m\log(T)+1)}{T}}$ and $\gamma = m/(T-1)$. A slight modification to their proof needs to be considered, because their losses are bounded between $[0, 1]$. In our case, we assume our loss function ℓ_t is convex, and Lipschitz on a bounded, closed, convex set Θ . Therefore, we can say there exists a value Λ such that $\Lambda \triangleq \max_{t \in [1, T], \theta \in \Theta} \ell_t(\theta) - \min_{t \in [1, T], \theta \in \Theta} \ell_t(\theta)$. This value can be easily incorporated into the proofs and bounds of [28] to give

$$\begin{aligned}
T_4 &\leq \Lambda^2 \sqrt{\frac{T}{2} \left((m+1) \log N + (T-1) h\left(\frac{m}{T-1}\right) \right)} \\
&\leq \Lambda^2 \sqrt{\frac{T}{2} ((m+1) \log N + m \log T + 1)}
\end{aligned}$$

where $h(x) = -x \log(x) - (1-x) \log(1-x)$ with respect to the natural logarithm. T_5 can be bounded using Lemma 2.8 on each time interval $[t_k, t_{k+1} - 1]$ and summing

over the $m + 1$ intervals, yielding

$$T_5 \leq \frac{(m+1)D_{\max}}{\eta_{T+1}} + \frac{2M}{\eta_T} V^{(m+1)}(\boldsymbol{\theta}_T) + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t. \quad \square$$

2.7.3 Proof of Theorem 2.4

Let α_* be the dynamical parameter in our candidate covering set, \mathcal{A}_N which minimizes the cumulative loss, α^* as the dynamical parameter in the entire space, \mathcal{A} which minimizes total loss, and $\tilde{\alpha}$ to the parameter in \mathcal{A}_N closest to α^* . Formally, we use the following definitions:

$$\alpha_* \triangleq \min_{\alpha \in \mathcal{A}_N} \sum_{t=1}^T \ell_t(\hat{\theta}_{\alpha,t}), \quad \alpha^* \triangleq \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \ell_t(\hat{\theta}_{\alpha,t}),$$

$$\tilde{\alpha} \triangleq \min_{\alpha \in \mathcal{A}_N} \|\alpha - \alpha^*\|.$$

We decompose the regret in the following way:

$$R_T(\boldsymbol{\theta}_T) = \underbrace{\sum_{t=1}^T \ell_t(\hat{\theta}_t) - \sum_{t=1}^T \ell_t(\hat{\theta}_t^{(\alpha_*)})}_{T_6}$$

$$+ \underbrace{\sum_{t=1}^T \ell_t(\hat{\theta}_t^{(\alpha_*)}) - \sum_{t=1}^T \ell_t(\hat{\theta}_t^{(\tilde{\alpha})})}_{T_7} + \underbrace{\sum_{t=1}^T \ell_t(\hat{\theta}_t^{(\tilde{\alpha})}) - \sum_{t=1}^T \ell_t(\boldsymbol{\theta}_t)}_{T_8}$$

To bound T_6 we use the bounds of [20, Corr 2.2] for the Exponentially Weighted Average Forecaster, which requires $\rho_t = \sqrt{\frac{2 \log N}{T}}$. Similarly to the proof of Theorem 2.3, the bound needs to be adjusted to account for the fact that our loss function, instead of being bounded by $[0, 1]$ instead has arbitrary bounds $[\ell_{\min}, \ell_{\max}]$. Because, ℓ_t is a convex function with a Lipschitz gradient defined on a bounded, closed, convex set, these bounds are finite, and incorporating them into the proof of [20] is not

difficult, yielding

$$T_6 \leq (\ell_{\max} - \ell_{\min})\sqrt{2T \log(N)}.$$

The term T_7 is upper bounded by 0 by the definitions of α_* and $\tilde{\alpha}$. Finally, the bound on T_8 is just the DMD regret bound (Theorem 2.2) with respect to $\tilde{\alpha}$:

$$T_8 \leq \frac{D_{\max}}{\eta_{T+1}} + \frac{2M}{\eta_T} \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t, \tilde{\alpha})\| + \frac{G^2}{2\sigma} \sum_{t=1}^T \eta_t.$$

We now use the Lipschitz assumption on Φ_t to show the bound with respect to α^* . Notice the variation term can be separated.

$$\begin{aligned} V_{\Phi}(\boldsymbol{\theta}_T) &= \sum_{t=1}^{T-1} \|\Phi_t(\theta_t, \tilde{\alpha}) - \Phi_t(\theta_t, \alpha^*) + \Phi_t(\theta_t, \alpha^*) - \theta_{t+1}\| \\ &\leq \sum_{t=1}^{T-1} \|\Phi_t(\theta_t, \alpha^*) - \theta_{t+1}\| + TL\varepsilon_N \end{aligned}$$

This shows we get a regret bound which scales like the variation from the best possible parameter α^* , in the set \mathcal{A} . Setting $\eta_t \propto \frac{1}{\sqrt{t}}$ gives the result.

2.7.4 Proof of Lemma 2.6

The proof is by induction, and we assume without loss of generality that $\beta = \mathbf{0}$. Assume that $\hat{\lambda}_t^{(\alpha)} = \hat{\lambda}_t^{(0)} + K_t\alpha$; this is trivially true for $t = 1$ since $K_1 = \mathbf{0}$ and $\hat{\lambda}_1^{(\alpha)} = \hat{\lambda}_1^{(\beta)}$ by construction. If we use DMD with $\Phi_t(\theta, \alpha)$ for $t \geq 1$, applying Algorithm (2.1) in this setting yields

$$\begin{aligned} \hat{\lambda}_{t+1}^{(0)} &= A_t[(1 - \eta_t)\hat{\lambda}_t^{(0)} + \eta_t\phi(x_t)] + c_t \\ \hat{\lambda}_{t+1}^{(\alpha)} &= A_t[(1 - \eta_t)\hat{\lambda}_t^{(\alpha)} + \eta_t\phi(x_t)] + B_t\alpha + c_t \\ &= A_t[(1 - \eta_t)(\hat{\lambda}_t^{(0)} + K_t\alpha) + \eta_t\phi(x_t)] + B_t\alpha + c_t \\ &= A_t[(1 - \eta_t)\hat{\lambda}_t^{(0)} + \eta_t\phi(x_t)] + c_t + (1 - \eta_t)A_tK_t\alpha + B_t\alpha \\ &= \hat{\lambda}_{t+1}^{(0)} + (1 - \eta_t)A_tK_t\alpha + B_t\alpha \end{aligned}$$

$$=\widehat{\lambda}_{t+1}^{(0)} + K_{t+1}\alpha. \quad \square$$

Notice that we must assume that $\widehat{\theta}_t$ must lie on the interior of the set Θ such that we can set the gradient to 0 to find the minimizer of equation 2.1a without projecting back onto the set.

2.7.5 Proof of Theorem 2.7

Since $\widehat{\lambda}_t^{(\alpha)}$ is an affine function of α given $\widehat{\lambda}_t^{(\beta)}$ for any β , and $\widetilde{\ell}_t(\lambda)$ is a convex function of λ , we have that $g_t(\alpha)$ is a convex function of α . For any $\alpha \in \mathbb{R}^n$, we have

$$\sum_{t=1}^T \ell_t(\widehat{\theta}_t) - \sum_{t=1}^T \ell_t(\theta_t) = \underbrace{\sum_{t=1}^T \ell_t(\widehat{\theta}_t) - \sum_{t=1}^T \ell_t(\widehat{\theta}_t^{(\alpha)})}_{T_9} + \underbrace{\sum_{t=1}^T \ell_t(\widehat{\theta}_t^{(\alpha)}) - \sum_{t=1}^T \ell_t(\theta_t)}_{T_{10}}.$$

To bound T_9 , note that $\widehat{\theta}_t = \widehat{\theta}_t^{(\widehat{\alpha}_t)}$ and

$$\sum_{t=1}^T \ell_t(\widehat{\theta}_t) - \sum_{t=1}^T \ell_t(\widehat{\theta}_t^{(\alpha)}) = \sum_{t=1}^T \widetilde{\ell}_t(\widehat{\lambda}_t) - \sum_{t=1}^T \widetilde{\ell}_t(\widehat{\lambda}_t^{(\alpha)}).$$

Furthermore,

$$g_t(\alpha) = \widetilde{\ell}_t(\widehat{\lambda}_t^{(\alpha)}) = \widetilde{\ell}_t(\widehat{\lambda}_t^{(\widehat{\alpha}_t)}) + K_t(\alpha - \widehat{\alpha}_t)$$

is convex in α , where we have used Lemma 2.6. Thus

$$T_9 \leq \sum_{t=1}^T [\widetilde{\ell}_t(\widehat{\lambda}_t) - \widetilde{\ell}_t(\widehat{\lambda}_t^{(\alpha)})] = \sum_{t=1}^T [g_t(\widehat{\alpha}_t) - g_t(\alpha)] \leq C_1 \sqrt{T}$$

via [19]. The term T_{10} can be bounded directly using Theorem 2.2, yielding for any α

$$T_{10} \leq C_2 \sqrt{T} \left(1 + \sum_{t=1}^T \|\theta_{t+1} - \Phi_t(\theta_t, \alpha)\| \right). \quad \square$$

DMD and DFS Experiments and Results

3.1 Introduction

In the previous chapter we described algorithms and associated theoretical guarantees for incorporating dynamics into the online learning paradigm. Many online learning problems can benefit from the incorporation of dynamical models, especially when the underlying state of the system is known to be changing in some systematic way. In this chapter, we describe how the ideas described and analyzed in the previous chapter can be incorporated in many different settings. We start by showing how examples of how a known dynamical model can be incorporated with a dynamic textures (Section 3.2) and image registration (Section 3.3) example using DMD. We then show how DFS can be used in situations where a finite set of candidate dynamics is known in a pair streaming video applications: compressed sensing microscopy (Section 3.4) and downsampled traffic surveillance (Section 3.5). Then we show how DMD with parametric additive dynamics can be used for Poisson point-processes (Section 3.6) as well as analyzing the Enron email corpus (Section 3.7). Finally, we show an extended example of Foreground-Background separation in Poisson video

which will implement multiple layers of the DFS algorithm to analyze video in low photon settings (Section 3.8).

3.2 DMD Experiment: Dynamic Textures with Missing Data

Sensors such as the Solar Data Observatory are generating data at unprecedented rates. Heliophysicists have physical models of solar dynamics, and often wish to identify portions of the incoming data which are inconsistent with their models. This “data thinning” process is an essential element of many big data analysis problems. We simulate an analogous situation in this section.

In particular, we consider a datastream corresponding to a dynamic texture [52][53], where spatio-temporal dynamics within motion imagery are modeled using an autoregressive process. In this experiment, we consider a setting where “normal” autoregressive parameters are known, and we use these within DMD to track a scene from noisy image sequences with missing elements. (Missing elements arise in our motivating solar astronomy application, for instance, when cosmic rays interfere with the imaging detector array.) As suggested by our theory, the tracking will fail and generate very large losses when the posited dynamical model is inaccurate.

More specifically, the idea of dynamic textures is that a low dimensional, autoregressive model can be used to simulate a video which replicates a moving texture such as flowing water, swirling fog, solar plasma flows, or rising smoke. This process is modeled in the following way:

$$\begin{aligned}\theta_t &= A\theta_{t-1} + Bu_t \\ x_t &= C_0 + C\theta_t + Dv_t \\ v_t, u_t &\sim \mathcal{N}(0, I).\end{aligned}$$

In the above, θ_t denotes the true underlying parameters of the system, and x_t the observations. The matrix A is the autoregressive parameters of the system, which

will be unique for the type of texture desired, C_0 the average background intensity, C is the sensing matrix which is usually a tall matrix, and B and D encode the strength of the driving and observation noises respectively. Using the toolbox developed in [54] and samples of a 220 by 320 pixel ocean scene [53], we learned two sets of parameters $A, A' \in \mathbb{R}^{50 \times 50}$, one representing water flowing when the data is played forward, and the other when played backwards, as well as corresponding parameters $C_0 \in \mathbb{R}^{70400}$, $C, C' \in \mathbb{R}^{70400 \times 50}$, $B, B' \in \mathbb{R}^{50}$ and $D, D' \in \mathbb{R}^{70400}$. Parameters $\theta_t \in [-500, 500]^{50}$ and data $x_t \in [-500, 500]^{70400}$ were then generated using these parameters, with the parameters A', B', C', D' and C_0 on $t = 100, \dots, 120$ and $t = 300, \dots, 320$ and the parameters A, B, C, D, C_0 on the rest of $t = 1, \dots, 550$ according to the above equations. Finally, every observation is corrupted by 50% missing values, chosen uniformly at random at every time point.

The parameters A, C_0 , and C were then used to define our (imperfect) dynamical model for DMD, $\Phi_t(\theta) = A\theta$, and a loss function $\ell_t(\theta) = \|\Omega_t(C\theta - C_0 - x_t)\|_2^2$, where Ω_t is a linear operator accounting for the missing data. Note that B and D are not reflected in these choices despite playing a role in generating the data; our theoretical results hold regardless. We use $\psi(\cdot) = \frac{1}{2}\|\cdot\|_2^2$ so the Bregman Divergence $D(x\|y) = \frac{1}{2}\|x - y\|_2^2$ is the usual squared Euclidean distance, and we perform no regularization ($r(\theta) = 0$). We set $\eta_t = \frac{1}{2\sqrt{t}}$, and ran 100 different trials comparing the DMD method to regular Mirror Descent (MD) to see the advantage of accounting for underlying dynamics. The results are shown in Figure 3.1.

There are a few important observations about this procedure. The first is that by incorporating the dynamic model, we produce an estimate which visually looks like the dynamic texture of interest, instead of the Mirror Descent prediction, which looks like a single snapshot of the water. Second, we can recover a good representation of the scene with a large amount of missing data, due to the autoregressive parameters

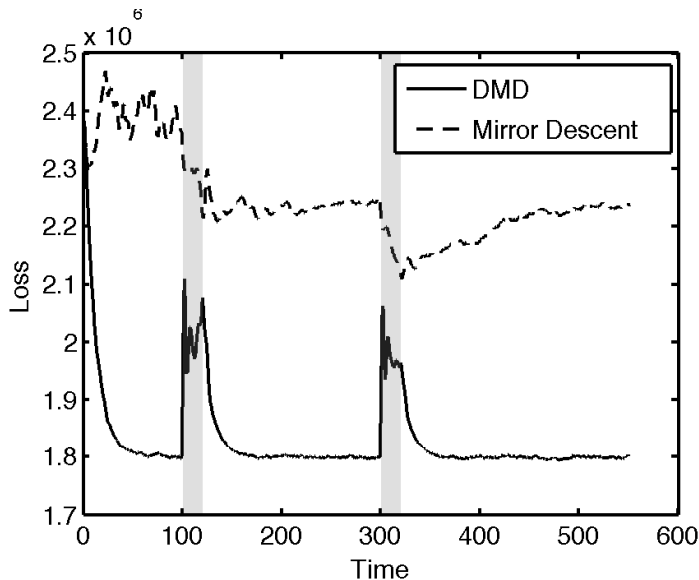


FIGURE 3.1: Loss curves for the proposed DMD method and classical MD vs. time, averaged over 100 trials for the experiment in Section 3.2. The gray areas indicate the intervals where the flow being imaged changed direction and hence the posited dynamical model was not reflected by the underlying data; note the sharp increases in the losses associated with DMD over those intervals, particularly in contrast with the losses associated with MD. Standard online learning methods like MD do not facilitate the detection of time periods with anomalous dynamics.

being of a much lower dimension than the data itself. Finally, because we are using the dynamics of forward moving water, when the true data starts moving backward, *a change that is imperceptible visually*, the loss spikes, alerting us of the abnormal behavior.

3.3 DMD Experiment: Solar Flare Detection with Missing Data

We use DMD in order to detect solar flares in image sequences from the Solar Data Observatory, in the presence of missing data and camera jitter. Solar flares represent temporally short, and spatially localized bursts of activity of the solar scene, but can be hard to detect when there is larger motion, and only partial observations. By explicitly accounting for camera jitter, we can do a better job of predicting what the scene should look like, and therefore quickly observe when and where in the scene solar flares occur.

In order to run DMD we choose an ℓ_2 loss function on the values of observed pixels

$$\ell_t(\hat{\theta}_t) = \frac{1}{2} \|\Omega_t(\hat{\theta}_t) - \Omega_t(x_t)\|_2^2$$

where Ω_t is a function that preserves the value of the true scene at the observed locations and sets the value to 0 otherwise. We use Euclidean distance as our Bregman Divergence. The dynamical model, Φ_t accounts for the camera jitter at time t . For our experiment we chose Φ_t to be a random pixel shift one pixel up, down, left or right. This scheme is run where each pixel is missing uniformly at random with probability 0.5. These results shown in Figure 3.2.

We can see in the loss plots that accounting for camera jitter with DMD explicitly we are less likely to get erroneous spikes in the loss function. This is important, because after the initial learning time, we can threshold the loss to detect anomalies as in [47], which would correspond to solar flares in this setting. However, if we do not explicitly account for the camera jitter, the Mirror Descent loss plot has spikes that are of the same magnitude as the spikes corresponding to a solar flare, and therefore thresholding would lead to either many false positives or missed detections.

3.4 DFS Experiment: Compressive Video Reconstruction

There is increasing interest in using “big data” analysis techniques in applications like high-throughput microscopy, where scientists wish to image large collections of specimens. This work is facilitated by the development of novel microscopes, such as the recent fluorescence microscope based on structured illumination and compressed sensing principles [55]. However, measurements in such systems are acquired sequentially, posing significant challenges when imaging live specimens.

Knowledge of underlying motion in compressed sensing image sequences can allow for faster, more accurate reconstruction [56; 57; 58]. By accounting for the underlying

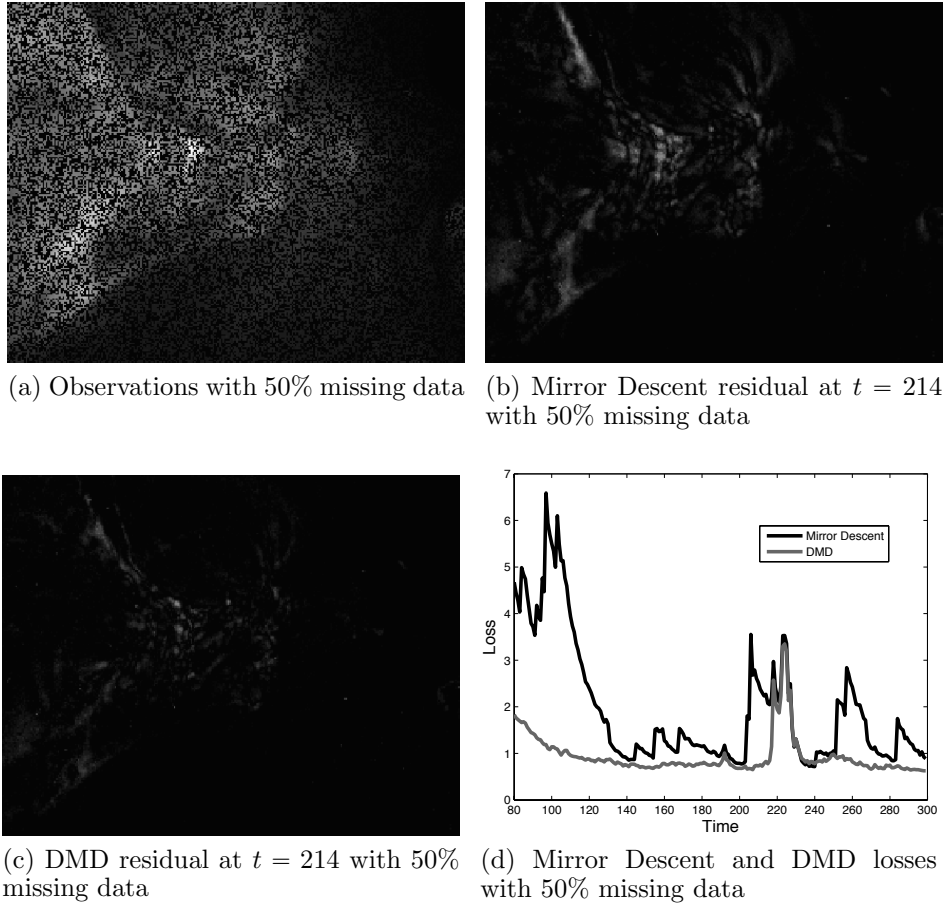


FIGURE 3.2: Results of DMD experiment in Section 3.3. Images display data and residuals with respect to ground truth with missing data. The residuals for the proposed DMD method are largest in the locations where a true solar flare occurs, while the MD residuals are large in locations exhibiting normal solar activity. Furthermore, DMD produces consistently lower, more stable losses. Consequently, real anomalous behavior is more easily visible at $t = 220$ via DMD, while MD produces large spikes in the losses in several inactive periods.

motion in the image sequence, we can have an accurate prediction of the scene before receiving compressed measurements, and when the measurements are noisy and the number of observations is far less than the number of pixels of the scene, these predictions allow both fast and accurate reconstructions. If the dynamics are not accounted for, and previous observations are used as prior knowledge, the reconstruction could end up creating artifacts such as motion blur or overfitting to

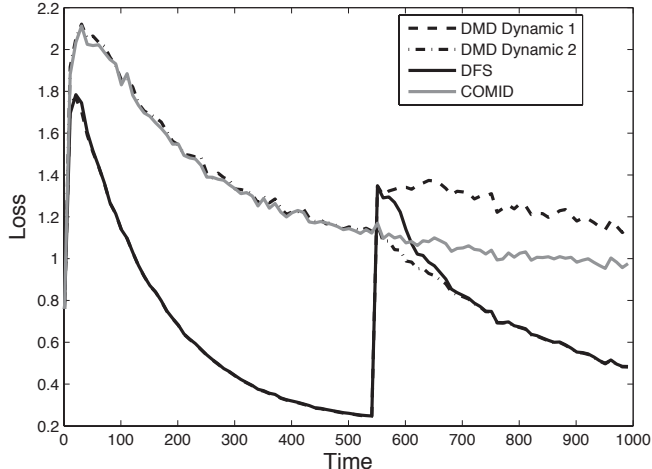


FIGURE 3.3: Losses averaged over 100 trials using DFS and comparing individual models for directional motion for the experiment in Section 3.4. $N = 9$ candidate dynamical models were considered within the DFS algorithm; plots for the two temporarily accurate models are shown for clarity. Before $t = 550$ the upward motion dynamic model incurs small loss, where as after $t = 550$ the motion to the right does well, and DFS successfully tracks this change.

noise. There has been significant recent interest in using models of temporal structure to improve time series estimation from compressed sensing observations [59; 60]; the associated algorithms, however, are typically batch methods poorly suited to large quantities of streaming data. In this section we demonstrate that DMD helps bridge this gap.

In this section, we simulate fluorescence microscopy data generated by the system in [55] while imaging a paramecium moving in a 2-dimensional plane; the t^{th} frame is denoted θ_t (a 120×120 image stored as a length-14400 vector) which takes values between 0 and 1. The corresponding observation is $x_t = A_t \theta_t + n_t$, where A_t is a 50×14400 matrix with each element drawn iid from $\mathcal{N}(0, 1)$ and n_t corresponds to measurement noise with $n_t \sim \mathcal{N}(0, \sigma^2)$ with $\sigma^2 = 0.1$. This model coincides with several compressed sensing architectures [61; 55].

Our loss function uses $f_t(\theta) = \frac{1}{2\sigma^2 d} \|x_t - A_t \theta\|_2^2$ and $r(\theta) = \tau \|\theta\|_1$, where $\tau > 0$ is a tuning parameter. We construct a family of $N = 9$ dynamical models, where $\Phi_t^{(i)}(\theta)$ shifts the (unvectorized) frame, θ , one pixel in a direction corresponding

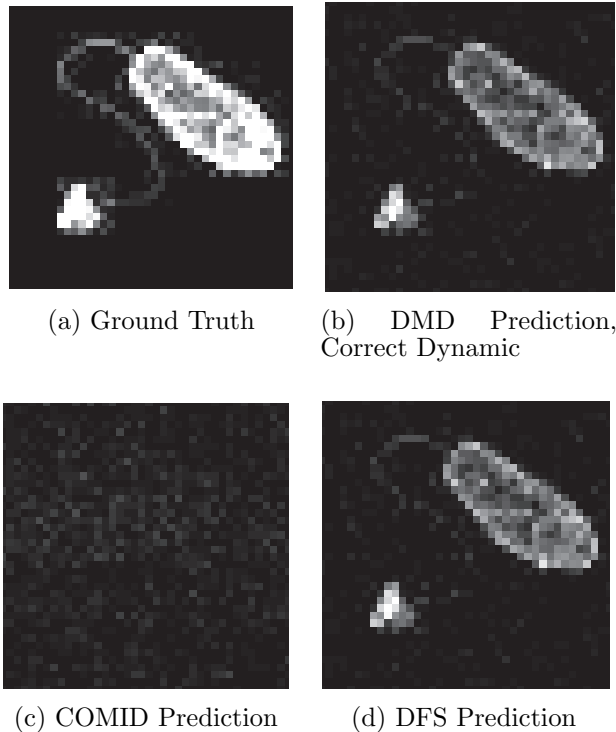


FIGURE 3.4: Zoomed in instantaneous predictions at $t = 1000$ for the experiment in Section 3.4. Top Left: True θ_t . Top Right: proposed DMD estimate $\hat{\theta}_t^{(\text{Right})}$. Bottom Left: $\hat{\theta}_t^{(\text{COMID})}$. Bottom Right: proposed DFS estimate (without prior knowledge of true dynamics) $\hat{\theta}_t$. The prediction made with the prevailing motion is an accurate representation of the ground truth, while the prediction with the wrong dynamic is an unclear picture. The DFS algorithm correctly picks out the most accurate dynamical model.

to an angle of $2\pi i/(N - 1)$ as well as a “dynamic” corresponding to no motion. (With the zero motion model, DMD reduces to COMID.) The true video sequence uses different dynamical models over $t = \{1, \dots, 550\}$ (upward motion) and $t = \{551, \dots, 1000\}$ (motion to the right). Finally, we use $\psi(\cdot) = \frac{1}{2} \|\cdot\|_2^2$ so the Bregman Divergence $D(x\|y) = \frac{1}{2} \|x - y\|_2^2$ is the usual squared Euclidean distance. The DMD sub-algorithms use $\eta_t = \frac{1}{\sqrt{t}}$, $\tau = .002$ and the DFS forecaster uses $\gamma = \frac{m}{T-1} = \frac{1}{999}$ and ρ_t is set as in Theorem 2.3. The experiment was then run 100 times.

Figures 3.3 and 3.4 show the impact of using DFS. We see that DFS switches

between dynamical models rapidly and outperforms all of the individual predictions, including COMID, used as a baseline, to show the advantages of incorporating knowledge of the dynamics.

3.5 DFS Experiment: Downsampled Traffic Surveillance Reconstruction

The DFS framework can also be used to reconstruct and predict traffic surveillance data by incorporating the approximately known traffic patterns into the estimation procedure. For a traffic scene with moving objects, one frame will only be informative for the next frame if the motion of the objects are incorporated into the prediction scheme and thus DFS can be used to improve the prediction process for this type of data. We used the Highway I video from the ATON shadow detection database (<http://cvrr.ucsd.edu/aton/shadow/index.html>) which contains 440 frames of 240×320 pixel images. The foreground of this video was then extracted using the inexact Augmented Lagrangian Multiplier method of [62]. This models the ability of a surveillance camera to do on-board pre-processing of data to remove long term background information and only transmit relevant, transient information. From here the image is blurred by a 7×7 Gaussian kernel with width defined by $\sigma = 1.75$, downsampled by a factor of 4 in both directions (for an overall downsampling factor of 16), and finally Gaussian white noise is added. The overall process can be described as follows:

$$x_t = DH\theta_t + n_t$$

where H encodes the blurring operation and D the downsampling, of the vectorized frame θ_t . The convolution is normalized such that $\|DH\|_2 = 1$ to ensure signal strength is maintained. The white noise has standard deviation of 20. The Gaussian noise leads intuitively to using an ℓ_2 data fit function and because we are estimating foreground objects with background removed we use ℓ_1 regularization to induce

sparsity. Combining these leads to the following loss function:

$$\ell_t(\hat{\theta}_t) = \frac{1}{2d} \|DH\hat{\theta}_t - x_t\|_2^2 + \frac{\tau}{d} \|\hat{\theta}_t\|_1$$

where d is the dimension of $\hat{\theta}_t$ and τ is the parameter which controls the relative amount of sparsity in the solution.

In order to implement DFS, we need a collection of feasible dynamics which can describe the data. The video shows cars moving generally from the top of the screen to the bottom at a relatively constant speed, so 5 different dynamical models are postulated which are simple row shifts of 10, 14, 18, 22 and 26 pixels respectively. One slightly more complicated dynamical model is also used which is based on a Block Matching Algorithm. For this dynamical model, a training set of the first 20 frames of the full data was used. The idea behind the Block Matching Algorithm is that for every 8×8 block in a given frame, a search is done over the entire previous frame for the block which best matches. For our purposes, we define the values of a block in the image after applications of the dynamics, $\Phi_t(\theta_t)$, to be the values of the block in θ_t which most consistently predicted the block of interest in the original training set. This process is defined below, where θ represents the 240×320 images, and the subscripts $[i : i + 7, j : j + 7]$ denote the 8×8 block starting at location i, j .

$$\Phi_t(\theta)_{[i:i+7,j:j+7]} = \theta_{[l(i,j):l(i,j)+7,m(i,j):m(i,j)+7]}$$

$$l(i, j), m(i, j) = \arg \min_{l, m} \sum_{\tau=1}^{19} \|\theta_{\tau+1,[i:i+7,j:j+7]} - \theta_{\tau,[l:l+7,m:m+7]}\|^2$$

This map was then smoothed slightly to ensure that objects in one frame remained intact after the application of the dynamics and not distorted. This procedure produced a dynamical model which mostly showed downward motion, similar to the pixel shifts, but allowed for some horizontal motion as well as allowing slightly different velocities in different regions of the image. Overall this brings the number of candidate dynamical models to $N = 6$.

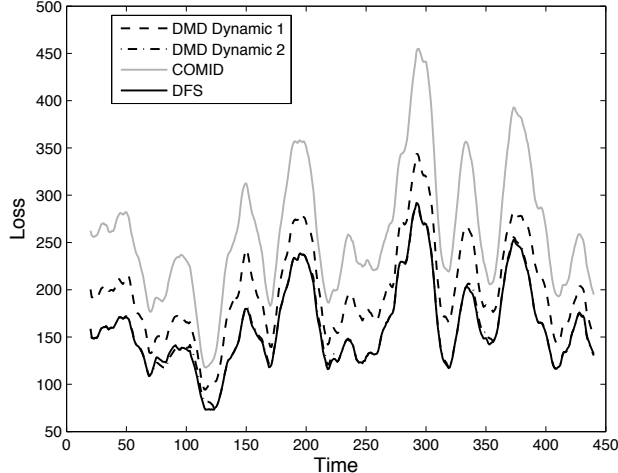


FIGURE 3.5: Moving average loss with time window=20 of DFS algorithm in the downsampled, noisy traffic surveillance experiment described in Section 3.5. Only shown are DMD losses for the best pixel shift dynamic and the block matching dynamic for clarity. Notice that the pixel shift consistently out performs COMID, and the block matching dynamic does even better. DFS quickly finds the best dynamic and follows it closely.

Using this setup, DFS is implemented for the $T = 440$ frames with $\tau = 10$, $d = 240 \times 320 = 76800$, $\eta_t = \frac{10d}{\sqrt{T}}$, $\rho_t = \sqrt{\frac{8(3 \log(N) + 2 \log(T) + 1)}{T}}$, and $\gamma = \frac{2}{T-1}$. The results are shown in Figure 3.5. We immediately see that COMID which does not account for dynamics performs very poorly and is easily outperformed even by the very simple pixel shift models (DMD Dynamic 1 in the plots corresponds to pixel shift of 18). Additionally we see that the block matching dynamic (DMD Dynamic 2) performs even better, and that the DFS algorithm quickly hones in on this dynamic model and follows it very closely.

In addition to the loss plots, we can observe actual reconstructions to see how close DFS comes to approximating the true image. An example of one segment of one prediction is shown in Figure 3.6. We see how the DFS procedure effectively upsamples, deblurs and denoises the data. By incorporating the dynamics, we can much more effectively find the details in the image that is not possible by using just COMID. Additionally, we have revealed details unobservable in the raw data. For

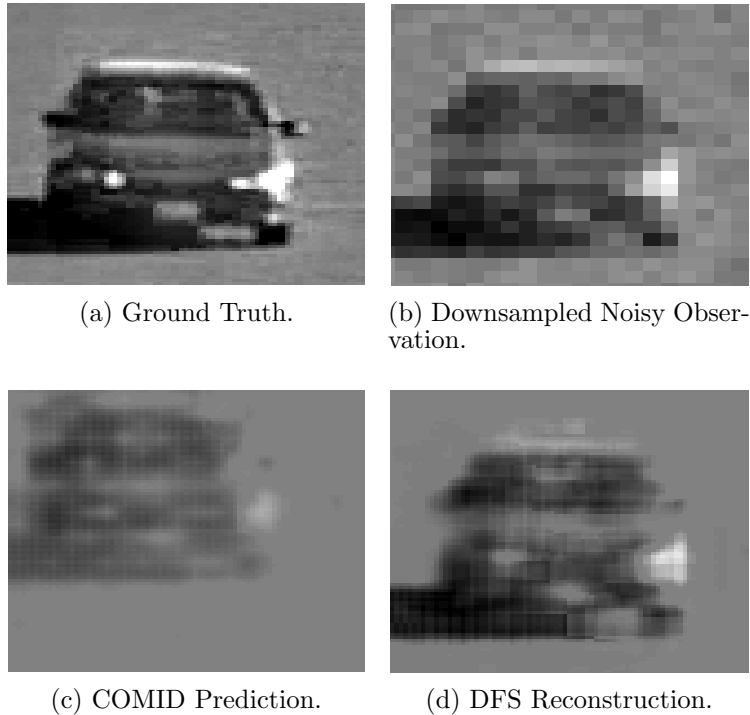


FIGURE 3.6: Zoomed in reconstruction of DFS algorithm in the downsampled, noisy traffic surveillance experiment described in Section 3.5. Here we see the ground truth image, observations, COMID prediction and DFS prediction.

instance, notice the lighter shade object inside the car in The lighter shade is visible in the DFS reconstruction, while it is impossible to see in the data. Additionally, details such as shape and edges on the headlights and license plate are much clearer in the DFS reconstruction compared to the data.

3.6 DMD with Parametric Additive Dynamics

We look at self-exciting point processes on connected networks [63; 64]. Here we assume there is an underlying rate for nodes in a network which dictate how likely each node is to participate in an action. Then, based on which nodes act, it will increase other nodes likelihood to act in a dynamic fashion. For example, in a social network a node could correspond to a person and an action could correspond to crime [65]. In a biological neural network, a node could correspond to a neuron and

an action could correspond to a neural spike [66].

We simulate observations of a such a self-exciting point process in the following way:

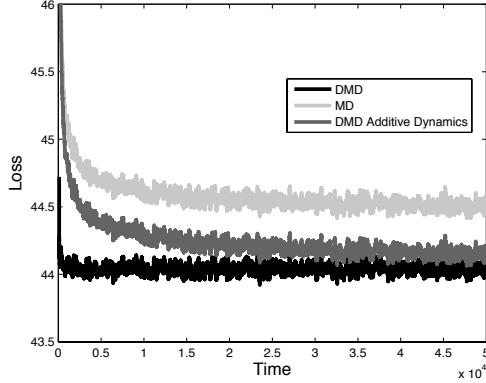
$$\begin{aligned}\lambda_{t+1} &= \Phi_t(\lambda_t, W) = \alpha\lambda_t + Wx_t + (1 - \alpha)\bar{\mu} \\ x_t &\sim \text{Poisson}(\lambda_t)\end{aligned}$$

For our experiments $\lambda_t \in (0, 5]^{100}$ represents the average number of actions each of 100 nodes will make during time interval t , and $W \in [0, 5]^{100 \times 100}$ reflects the unknown underlying network structure which encodes how much an event by a one node will increase the likelihood of an event by another node in future time intervals. Here we assume α is a known parameter between zero and one, and $\bar{\mu} \in \mathbb{R}^{100}$ is a underlying base event rate.

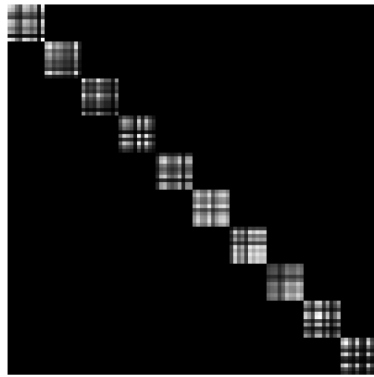
Our goal is to track the event rates λ_t and the network model W simultaneously; Algorithm 2.3 is applied with

$$\begin{aligned}\ell_t(\theta) &= \langle \mathbf{1}, \exp(\theta) \rangle - \langle x_t, \theta \rangle, & \tilde{\ell}_t(\lambda) &= \langle \mathbf{1}, \lambda \rangle - \langle x_t, \log \lambda \rangle, \\ Z(\theta) &= \langle \mathbf{1}, \exp(\theta) \rangle, & \lambda &= \nabla Z(\theta) = \exp(\theta).\end{aligned}$$

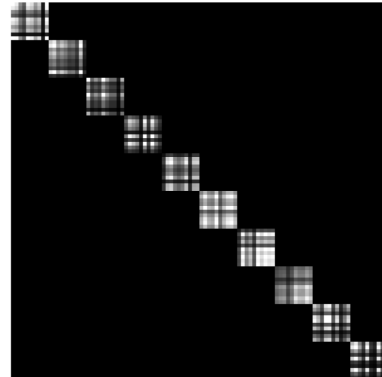
We generated data according to this model for $t = 1, \dots, 50000$ for 1000 different trials, using $\alpha = 0.5$, $\bar{\mu} = 0.1$ and W generated such that it is all zeros except on each distinct 10×10 block along the diagonal, where elements are chosen to be uu^T for a vector $u \in [0.1, 1.1]^{10}$ with elements chosen uniformly at random. The matrix W is then normalized so that its spectral norm is 0.25 for stability. Using this generated data we ran DMD with known W (Alg. 2.1), MD, and DMD with additive dynamics (Alg. 2.3) to learn the dynamic rates. The step size parameters were set as $\eta_t = .9/\sqrt{t}$ and $\rho_t = .005/\sqrt{t}$. The results are shown for DMD with the matrix W known in advance, MD and Alg. 2.3 in Figure 3.7. This basic model is examined in more detail in Chapter 4 in the context of Hawkes Processes.



(a) Moving average loss over previous 100 time points for DMD with a known W matrix, MD, and DMD exp averaged over 1000 trials.



(b) The true value of W



(c) T final estimate of W computed using Alg 2.3.

FIGURE 3.7: Experimental results tracking a self-exciting point process on a network, described in Section 3.6. Notice how the loss curve for Alg. 2.3 approaches the DMD curve (associated with clairvoyant knowledge of the underlying network matrix W) as the estimate of W improves, and significantly outperforms conventional mirror descent.

We again see several important characteristics in these plots. The first is that by incorporating knowledge of the dynamics, we incur significantly less loss than standard Mirror Descent. Secondly, we see that even without knowing the values of the matrix W , we can learn it simultaneously with the rate vectors λ_t from streaming data, and the resulting accurate estimate leads to low loss in the estimates of the rates.

3.7 DMD with Parametric Additive Dynamics Experiment - Enron Email Corpus

In this section, we analyze the Enron email corpus[67]. The goal of analyzing this dataset is to be able to read in the emails in a streaming fashion, and be able to detect anomalous moments, and determine if they align with important known events in the company’s history. In order to do this, the approximately 500,000 emails were combined into 1 hour time bins for each of the 150 given employees, from December 16th 2000 until January 1st 2003. For each employee for each hour, we receive an indicator saying whether that employee sent or received an email at that specific hour.

We model the indicator values as a Bernoulli random variable for each employee for each hour, and want to learn the probability that each employee will send or receive an email at a given time. Given an estimate $p_t \in [0, 1]^{150}$ and the actual observations $x_t \in \{0, 1\}^{150}$ at time t , we have the following loss function

$$\tilde{\ell}_t(\hat{p}_t) = -\langle x_t, \log \hat{p}_t \rangle - \langle \mathbf{1} - x_t, \log \mathbf{1} - \hat{p}_t \rangle$$

which can be rewritten as

$$\ell_t(\hat{\theta}_t) = -\langle x_t, \hat{\theta}_t \rangle + \left\langle \mathbf{1}, \log \left(\mathbf{1} + \exp \hat{\theta}_t \right) \right\rangle$$

under the transformation $\theta = \log \left(\frac{p}{1-p} \right)$. This formulation gives us the primal (θ) and dual (p) variables and loss functions necessary for DMD with parametric additive dynamics algorithm described in Alg. 2.3.

In order to use Alg. 2.3 on this data set we need to postulate a sequence of dynamics, and this dataset lends itself to several possibilities. The first possibility is to search for possible relationships in the network that might effect a person’s likelihood to send or receive and email. For this we consider the following dynamic

model:

$$\Phi_t(p) = p + \eta_t(W - I)x_t \text{ s.t. } W_{i,j} \geq 0, \|W\|_\infty \leq 1.$$

This dynamic model might look a little bit strange at first, but when used in Alg. 2.3 we are left with the overall update equation of the form

$$\hat{p}_{t+1} = (1 - \eta_t)\hat{p}_t + \eta_t\widehat{W}_t x_t$$

where we can search for \widehat{W}_t over the space $\mathcal{W} = \{W \text{ s.t. } W_{i,j} \geq 0, \|W\|_\infty \leq 1\}$. This update equation shows that we not only update our estimate of an individual's likelihood of sending or receiving an email based on their previous history, but also the history of people they might be associated with in the network. For instance, if someone is working closely with another employee, they will be more likely to respond quickly to that person's email as opposed to someone who works in another department.

The next two dynamical models attempt to incorporate the weekly cycles in email sending behavior. The first dynamical model incorporates a weighted average over each employee's behavior at corresponding hours in previous weeks to predict their behavior at the current moment, and the second model incorporates a weighted average of the entire company's behavior in previous weeks. The first model uses the following function:

$$\Phi_t(p) = p + \eta((\alpha_0 - 1)x_t + \alpha_1 x_{t+1-K} + \alpha_2 x_{t+1-2K} + \dots + \alpha_M x_{t+1-MK})$$

where K corresponds to the amount of time backwards we wish to look, in this case 24×7 hours, and M is how many of these periods back we wish to look. This leads to the total update equation

$$\hat{p}_{t+1} = (1 - \eta_t)p_t + \eta_t(\alpha_0 x_t + \alpha_1 x_{t+1-K} + \dots + \alpha_M x_{t+1-MK})$$

where we can use Alg 2.3 to search for optimal values of α over the $(M + 1)$ dimensional simplex. A similar dynamical model leads to the final update equation

$$\hat{p}_{t+1} = (1 - \eta_t)p_t + \eta_t \left(\alpha_0 x_t + \alpha_1 \frac{\mathbf{1}\mathbf{1}^T x_{t+1-K}}{d} + \dots + \alpha_M \frac{\mathbf{1}\mathbf{1}^T x_{t+1-MK}}{d} \right)$$

which uses the same mechanism for incorporating weekly patterns, but considers the company-wide information from the previous weeks, instead of the employee specific information. Once again our algorithm allows us to use this update equation while searching for optimal values of α over the $(M + 1)$ dimensional simplex. For our experiments, we considered $M=20$.

Using this loss function and dynamical models, Alg. 2.3 was used to predict each employee’s probability of sending or receiving an email for every hour in the given time range. The DMD step size was set as $\eta_t = \frac{10}{\sqrt{T}}$, where T is the total number of hours observed. Additionally, the step sizes ρ_t were set at $\rho_t = \frac{10^{-9}}{\sqrt{T}}$ for the network dynamics, and $\rho_t = \frac{10^{-1}}{\sqrt{T}}$ for the temporal behavior dynamics. In addition to these three models, we also used Alg. 2.2 to assign weights to each of these methods, and regular Mirror Descent, to create one combined prediction. We compared the performance of each of these methods to Mirror Descent, and calculated the relative gain through time τ as $\frac{\sum_{t=1}^{\tau} \ell_t(\hat{\theta}_t^{(\text{MD})}) - \ell_t(\hat{\theta}_t)}{\sum_{t=1}^{\tau} \ell_t(\hat{\theta}_t^{(\text{MD})})}$, where $\hat{\theta}_t^{(\text{MD})}$ corresponds to the prediction made by Mirror Descent. These results are shown in Figure 3.8.

Immediately we see that all of the proposed dynamic models improve the predictive power compared to Mirror Descent. Including the company’s weekly email behavior has the biggest single impact, followed by the individual’s weekly behavior, and finally the network dynamics. When all three are combined using fixed share, we attain a performance boost of about 8% in the end, and 12% at its peak. Additionally, our method can more accurately detect anomalies. One heuristic for determining anomalies would be to compare a method’s instantaneous loss to the av-

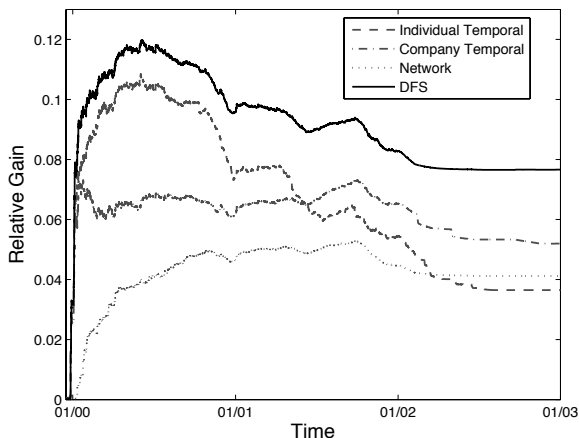


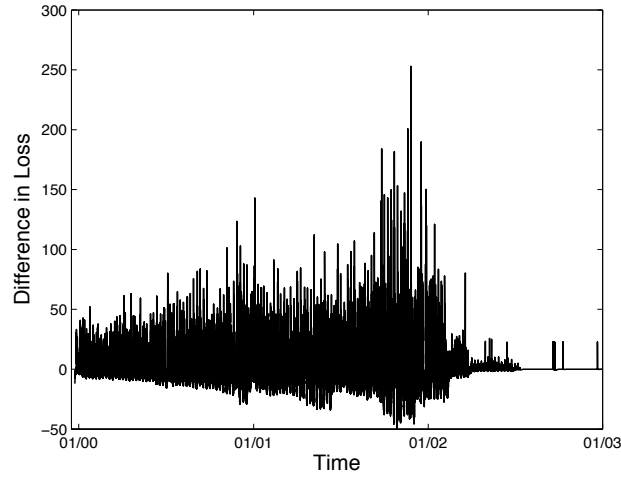
FIGURE 3.8: Relative gain of our method’s prediction using dynamical models compared to Mirror Descent. Notice that all our proposed dynamics improve performance compared to Mirror Descent in the 4-8% range at the end of the data set, and at a maximum of around 12%.

erage loss for the preceding week. High spikes would then correspond to anomalous moments. This value is shown in Figure 3.9 for Mirror Descent and our DFS method. We see that our method does a better job of finding the spikes in December 2000 and December 2001, corresponding to true anomalies, as the spikes stick out more from the noise around them. In December 2000 energy commodity trading became deregulated in California [68] and in December 2001 Enron filed for Chapter 11 [69].

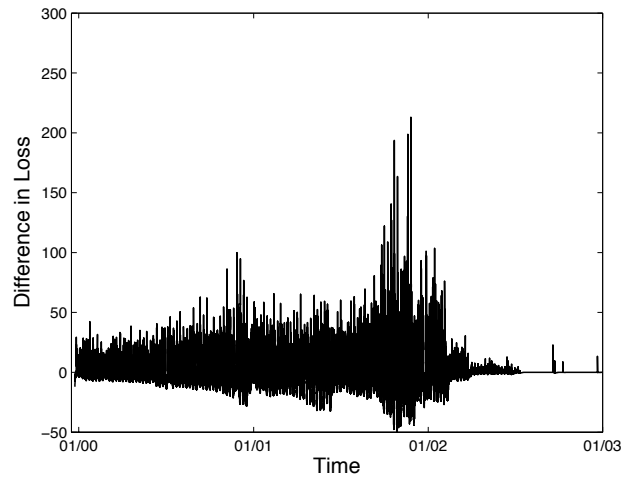
3.8 Foreground-Background Separation in Poisson Video

Many imaging applications such as night vision, infrared imaging, and certain astronomical imaging systems are characterized by limited amounts of available light. In these and other settings, the goal is to reconstruct spatially distributed and dynamic phenomena from data collected by counting discrete independent events, such as photons hitting a detector. More specifically, we can model our observations at time t as

$$x_t \sim \text{Poisson}(\theta_t), \tag{3.1}$$



(a) Comparison of instantaneous loss versus average of preceding week for Mirror Descent.



(b) Comparison of instantaneous loss versus average of preceding week for our DFS method.

FIGURE 3.9: Filtered loss vs. time plots for the Enron experiment using (a) losses stemming from MD and (b) losses stemming from the proposed DFS method. These two plots show that our method has larger relative spikes around December 2000 and December 2001 corresponding to significant moments in the company’s history.

where $x_t \in \mathbb{Z}_+^n$ is the vector of photon counts across n detectors and $\theta_t \in \mathbb{R}_+^n$ is the intensity of interest (i.e., the n -pixel scene) [70].

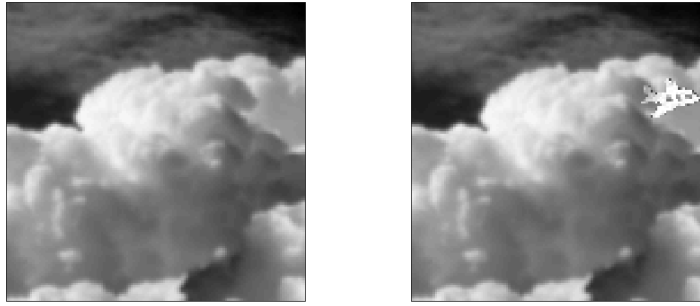
We are interested in the case where θ_t has two components: a dynamic foreground ϕ_t which occupies a relatively small portion of the scene, and a static or slowly-varying background β_t , so that

$$\theta_t = \phi_t + \beta_t.$$

The goal is to recover accurate estimates of ϕ_t and β_t from x_t , especially when the photon counts are very low and when the underlying dynamics are unknown.

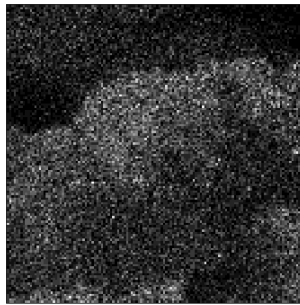
There exists a rich literature on image estimation and background subtraction methods, and a wide variety of effective tools in high SNR regimes. For instance, a common method for object tracking is to form an estimate of the background scene, and subtract this from the observation to get an estimate for the foreground [71]. Many of these methods make the assumption that the observed pixel values are the true scene corrupted with white Gaussian noise distributed around the true, slowly varying background mean [72], which is untrue both by the Poisson observation model and in settings with dynamic backgrounds. Alternatively, another technique is to learn and track a low-dimensional subspace representation of the background [73]. While such a method can be modified for the Poisson setting, simply subtracting this background estimate from the observations will still not yield an accurate foreground estimate in the low-light setting. In fact, even if the background were known exactly, subtraction will not give a very accurate estimate of the foreground, as shown in Figure 3.10.

The photon-limited image estimation problem is particularly challenging because it introduces intensity-dependent Poisson statistics which require specialized algorithms and analysis for optimal performance. Simply transforming Poisson data to produce data with approximately Gaussian noise (via, for instance, the variance

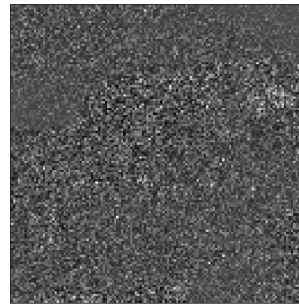


(a) Known Background

(b) True Scene



(c) Poisson Observation



(d) Observation minus true background

FIGURE 3.10: Challenges of background subtractions for photon-limited video. The background (a) and a foreground object in the top right corner form the true scene (b). Poisson observations are then collected from the true scene (c). Even if the background was known exactly and subtracted from the observations, the resulting image (d) is still very noisy, making accurate inference about foreground objects challenging.

stabilizing Anscombe transform [74; 75] or Fisz transform [76; 77]) can be effective when the number of counts is sufficiently high [78; 79]. However, applying these methods to foreground estimation is a difficult problem due to the non-linearities induced by the transforms. Specifically, these tools may make it possible to estimate θ_t effectively, but the inverse problem of estimating ϕ_t and β_t is significantly more challenging because of the nonlinear relationship between the unknowns and variance stabilized observations.

In addition, the dynamic setting presents significant opportunity for improved photon-limited surveillance. Consider the case in which the temporal dynamics are known exactly. For the Gaussian noise setting, the Kalman filter has proved enormously effective. The known dynamics can effectively act as a prior probability model for the scene at time t , and once x_t has been observed, this prior knowledge can dramatically improve reconstruction accuracy even when the number of available photons is small.

Generalizations of this approach to Poisson noise are possible with particle filters [22], but particle degeneracy is a major practical challenge. Furthermore, classical stochastic filtering methods typically assume an accurate, fully known dynamical model; if a dynamical model is learned from data, it is typically assumed not to change over time.

We show how we can use the DMD framework to estimate the underlying, time-varying dynamical model, and use this estimate to generate online predictions of the foreground and background video sequences.

3.8.1 Problem Formulation

We model the data as Poisson observations of a scene which is mostly background with some dynamic foreground. In order to distinguish foreground from background, we assume that the two have discernibly different underlying dynamics, and that the foreground obscures only a small fraction of the background. We denote the observation at time t as x_t , the background as β_t and the foreground as ϕ_t . Therefore the probability density function of the observation is given as:

$$p(x_t|\phi_t, \beta_t) = \prod_{i=1}^d \frac{(\phi_{t,i} + \beta_{t,i})^{x_{t,i}}}{x_{t,i}!} \exp [-(\phi_{t,i} + \beta_{t,i})]. \quad (3.2)$$

Here, t indicates time index, and i indicates pixel location. Notice that this model assumes that the observed scene is the superposition of background and foreground

at every pixel. In actuality every pixel would either be completely foreground or completely background, but it is difficult to model this explicitly because the locations of the foreground pixels would need to be known exactly *a priori*. Using this model we wish to reconstruct β_t and ϕ_t as accurately as possible in a time-efficient manner.

3.8.2 Method

In order to solve the problems of background and foreground estimation, we will use an adapted version of the DFS algorithm. In this section, we will describe the multi-layered DFS method applied to background subtraction problems.

It is important to note that we use the DFS algorithm for the background instead of a moving average:

$$\hat{\beta}_t = \frac{\sum_{s=1}^t \alpha^{t-s} y_s}{\sum_{s=1}^t \alpha^{t-s}}$$

for some $\alpha \in [0, 1]$. This is important because if the background has some dynamic motion, the moving average would perform poorly. If α were set too low, then the background estimate would be heavily corrupted by Poisson noise artifacts. On the other hand, if α were very close to 1, the motion of the background would cause blur in the estimate. Even if α is chosen in between these two extremes, the estimate would not reflect the true background very well as shown in figure 3.11.

Our first step is to estimate the background, so we must first find a loss function for estimating β_t . We use the negative Poisson log-likelihood function of the observation omitting the $x_i!$ term since it is an offset not dependent on β :

$$f_{\beta,t}(\beta) = \sum_{i=1}^d (\beta_i - x_i \log(\beta_i + \delta)). \quad (3.3)$$

A small constant, δ is added to ensure numerical stability. Notice that this is the same loss function that would be used if the video sequence was assumed to only

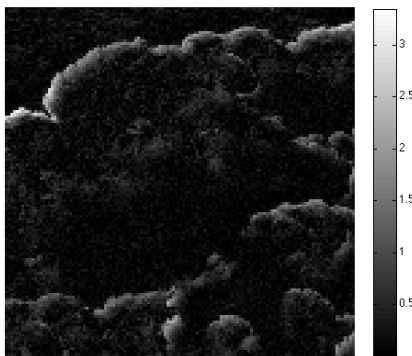


FIGURE 3.11: Absolute difference between moving average and true background with $\alpha = 0.99$. The true background has max value of 5, meaning the errors are relatively large. Notice that this image contains both errors at the leading edge due to motion, and noise errors from the observation model. Both of these errors would adversely affect the foreground estimation performance

have background content.

We then wish to estimate ϕ_t . We again start by using the negative Poisson log-likelihood as a basis for the loss function for ϕ_t , but now assume access to an estimate of the background, $\hat{\beta}_t$.

$$-\log(p(x_t|\phi, \hat{\beta}_t)) = \sum_{i=1}^d \left(\phi_i + \hat{\beta}_{t,i} - \log \left(\frac{(\phi_i + \hat{\beta}_{t,i})^{x_{t,i}}}{x_{t,i}!} \right) \right) \quad (3.4)$$

Assuming that the background estimate has already been found, this leads to the following data fit function for the foreground:

$$f_{\phi,t}(\phi; \hat{\beta}_t) = \sum_{i=1}^d \left(\phi_i - x_{t,i} \log \left(\frac{\phi_i}{\hat{\beta}_{t,i} + \delta} + 1 \right) \right). \quad (3.5)$$

This loss function comes from the negative log-likelihood function by subtracting $\sum_{i=1}^d \hat{\beta}_{t,i} + \log(x_{t,i}!) - x_{t,i} \log(\hat{\beta}_{t,i})$ which is independent of ϕ_t . Again, a small positive constant δ is used to ensure numerical stability.

Finally, we include regularization penalties, r_β and r_ϕ . For this application, we use a total variation penalty [80; 81], which insures that the estimates are somewhat

Algorithm 3.1 Background and Foreground Estimation

```

for  $t = 1, \dots, T$  do
  Observe  $y_t$ 
  for  $i = 1, \dots, N_1$  do
     $\tilde{w}_{\beta,t+1}^{(i)} = w_{\beta,t}^{(i)} \exp(-\rho_t \ell_{\beta,t}(\hat{\beta}_{i,t}))$ 
     $w_{\beta,t+1}^{(i)} = \frac{\gamma}{N_1} \sum_{j=1}^{N_1} \tilde{w}_{\beta,t+1}^{(j)} + (1 - \gamma) \tilde{w}_{\beta,t+1}^{(i)}$ 
     $\tilde{\beta}_{t+1}^{(i)} = \arg \min_{\beta} \eta_t \langle \nabla f_{\beta,t}(\hat{\beta}_t^{(i)}), \beta \rangle + \tau_{\beta} \|\beta\|_{\text{TV}} + \|\beta - \hat{\beta}_t^{(i)}\|_2^2$ 
     $\hat{\beta}_{t+1}^{(i)} = \Phi^{(\beta,i)}(\tilde{\beta}_{t+1}^{(i)})$ 
  end for
   $\tilde{\beta}_{t+1} = \sum_{i=1}^{N_1} w_{\beta,t+1}^{(i)} \tilde{\beta}_{t+1}^{(i)} / \sum_{i=1}^{N_1} w_{\beta,t+1}^{(i)}$ 
   $\hat{\beta}_{t+1} = \sum_{i=1}^{N_1} w_{\beta,t+1}^{(i)} \hat{\beta}_{t+1}^{(i)} / \sum_{i=1}^{N_1} w_{\beta,t+1}^{(i)}$ 
  for  $k = 1, \dots, N_2$  do
     $\tilde{w}_{\phi,t+1}^{(k)} = w_{\phi,t}^{(k)} \exp(-\rho_t \ell_{\phi,t}(\hat{\phi}_t^{(k)}; \tilde{\beta}_{t+1}))$ 
     $w_{\phi,t+1}^{(k)} = \frac{\gamma}{N_2} \sum_{j=1}^{N_2} \tilde{w}_{\phi,t+1}^{(j)} + (1 - \gamma) \tilde{w}_{\phi,t+1}^{(k)}$ 
     $\tilde{\phi}_{t+1}^{(k)} = \arg \min_{\phi} \eta_t \langle \nabla f_{\phi,t}(\hat{\phi}_t^{(k)}; \tilde{\beta}_{t+1}), \phi \rangle + \tau_{\phi} \|\phi\|_{\text{TV}} + \|\phi - \hat{\phi}_t^{(k)}\|_2^2$ 
     $\hat{\phi}_{t+1}^{(k)} = \text{SoftThresh}(\Phi^{(\phi,k)}(\tilde{\phi}_{t+1}^{(k)}))$ 
  end for
   $\tilde{\phi}_{t+1} = \sum_{k=1}^{N_2} w_{\phi,t+1}^{(k)} \tilde{\phi}_{t+1}^{(k)} / \sum_{k=1}^{N_2} w_{\phi,t+1}^{(k)}$ 
   $\hat{\phi}_{t+1} = \sum_{k=1}^{N_2} w_{\phi,t+1}^{(k)} \hat{\phi}_{t+1}^{(k)} / \sum_{k=1}^{N_2} w_{\phi,t+1}^{(k)}$ 
end for

```

smooth, as would be expected in natural images. This makes the overall loss functions the following:

$$\ell_{\beta,t}(\beta) = f_{\beta,t}(\beta) + \tau_{\beta} \|\beta\|_{\text{TV}} \quad (3.6)$$

$$\ell_{\phi,t}(\phi; \beta) = f_{\phi,t}(\phi; \beta) + \tau_{\phi} \|\phi\|_{\text{TV}}. \quad (3.7)$$

where τ_{β} and τ_{ϕ} are tradeoff parameters between data fidelity and regularization for the background and foreground respectively. Notice how this process essentially tries to find a coarse estimate for the underlying scene in the background, and then tries to find a foreground which fine tunes this estimate. These loss functions, as constructed, are convex which means we can use online convex optimization techniques. We will also use the fact that the background and foreground should have different dynamics to help with separation and reconstruction.

The overall procedure is described in Algorithm 3.1. For both of the inner loops,

the minimization can be done using the FISTA algorithm of Beck and Teboulle [81]. Additionally, a small amount of soft thresholding is applied to the foreground estimate at each time step, to ensure that ambiguous areas that could be considered either background or foreground are removed from the foreground estimate. Without this thresholding, these ambiguous areas would appear in the foreground estimate as an underlying haze. It is important to notice that for the background and foreground we have two slightly different estimates. The values denoted $\tilde{\beta}_t$ and $\tilde{\phi}_t$ are the filtering estimates, meaning that they are reconstructions for time t using all the observations up to time t . The values $\hat{\beta}_{t+1}$ and $\hat{\phi}_{t+1}$ are the prediction values, meaning they use all the data up to time t to predict the observation at time $t + 1$.

3.8.3 Experimental Results

To test this method, we created a data set that featured an object moving across a slowly varying background in the following way:

$$\begin{aligned}\phi_t^* &= \Phi_{t-1}^{(\phi)}(\phi_{t-1}^*) \\ \beta_t^* &= \Phi_{t-1}^{(\beta)}(\beta_{t-1}^*) \\ \theta_t &= \sum_{i=1}^d e_i^T [\mathbf{1}_{\phi,t}(i)\phi_t^* + (1 - \mathbf{1}_{\phi,t}(i))\beta_t^*] e_i \\ x_t &\sim \text{Poisson}(\theta_t),\end{aligned}$$

where $\mathbf{1}_{\phi(t),t}(i)$ is the indicator of pixel i being foreground or not and e_i is the i^{th} standard basis vector. This process shows a foreground object being translated through the function $\Phi_t^{(\phi)}$ on top of a background image moving with dynamics $\Phi_t^{(\beta)}$. The images were compiled by letting certain pixels be designated as a foreground object, and everything else being background. Notice, that the algorithm assumes every pixel is the addition of foreground and background, but the data used is more realistic in that each pixel is either one or the other.

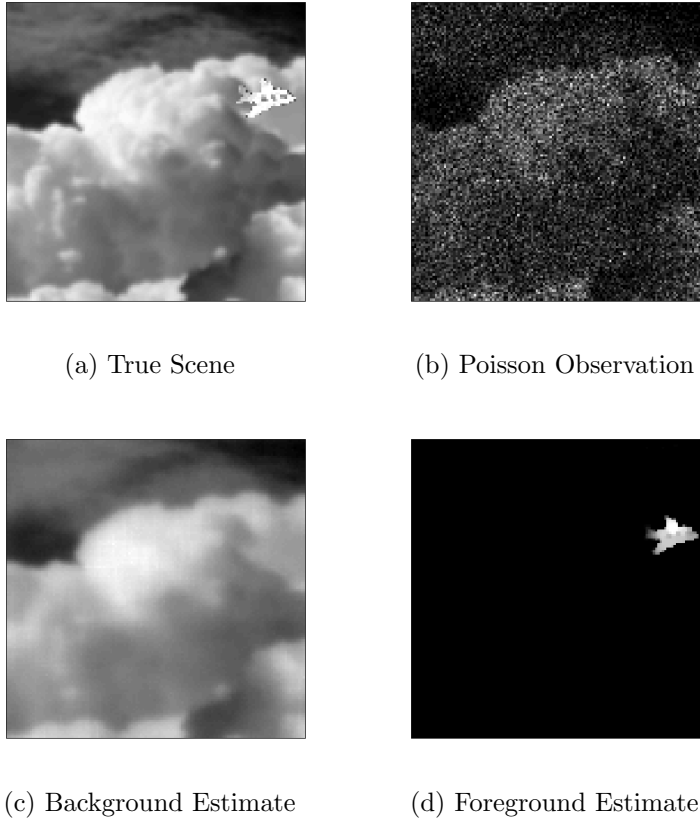


FIGURE 3.12: Foreground and background reconstruction at $t = 250$. The true image (a) has foreground and background content, and the observations (b) are extremely noisy. We form a background estimate (c) and use it to obtain a foreground estimate (d). Notice the details visible in the foreground estimate such as the windows and tail structure of the plane.

Each image is 150×150 , and no pixel has mean value greater than 5, so the video is extremely photon limited. For the background, the true underlying dynamics was a subpixel shift of $1/50$ th of a pixel to the top left at every time step. For the foreground the true dynamics is a full pixel shift to the top right for the first 500 frames and bottom right for the second 500 frames. The candidate dynamic models used for the background were subpixel shifts of $1/50$ th of a pixel shift in directions of $k\pi/4$ for $k = 1, 2, \dots, 8$ and stationary ($\Phi^{(\beta)} = I$). The foreground candidate dynamics were full pixel shifts in the same directions as well as a stationary dynamic.

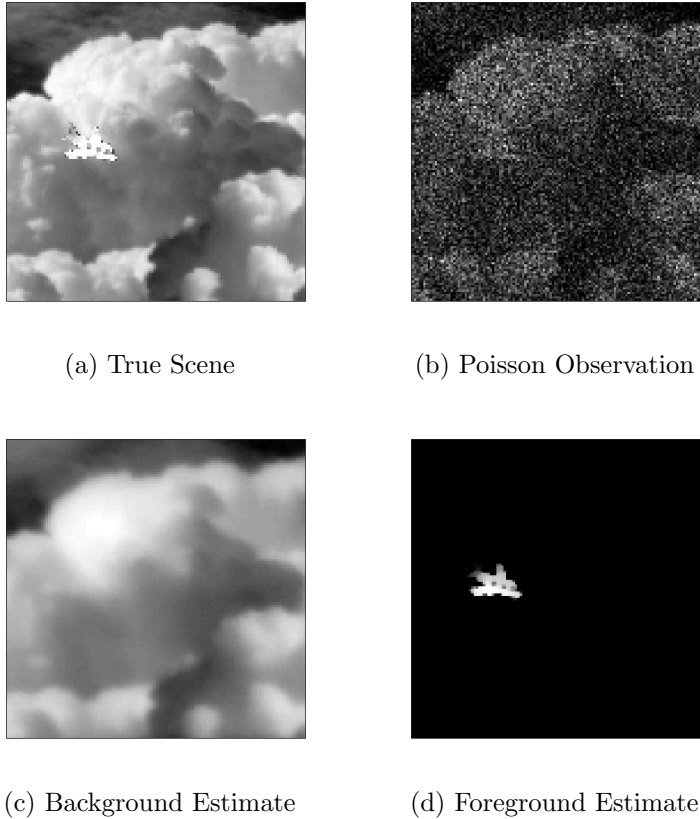


FIGURE 3.13: Foreground and background reconstruction at $t = 925$. Again notice how the foreground object in (a) is basically imperceptible in the observations (b). By estimating the background (c) an accurate foreground estimate can be constructed (d)

Figures 3.12 and 3.13 show examples of the DFS algorithm taking the series of Poisson observations, and making accurate representations of the foreground and background. It is especially important to notice that including the foreground and background dynamics allows for the foreground image to become clear. Without incorporating dynamics and regularization, the additional foreground image would simply be the transient errors of the background estimation. By including the dynamics in the optimization process, the systematic difference of the background estimate can be found to be the foreground object.

3.9 Conclusions

In this chapter we have shown how incorporating even modest dynamical models into the online learning routines can increase the predictive ability of the Learner. Specifically, we have shown that when the underlying system dynamics are known the DMD algorithm can be used to incorporate the dynamics and obtain accurate predictions of the system as in the dynamic textures example. We have also shown how by having a finite set of possible dynamical models, the DFS algorithm can quickly hone in on the best model to improve predictive power as in the Compressed Sensing and traffic surveillance examples. We then extended the finite family scenario, to the parametric family model and showed how that can be used in the Poisson Autoregressive and Enron examples. Finally, we shown a hierarchical type DFS algorithm for Foreground-Background separation to successfully handle the Poisson streaming video case, where the statistics of low light observations must be explicitly accounted for. Such object tracking in low light situations has important applications such as surveillance and astronomy.

Tracking Dynamic Point Processes on Networks

4.1 Introduction

In a variety of settings, we observe a series of events associated with a group of actors whose interactions trigger future events. The interactions between these actors can be modeled using a network. For example:

- **Social networks:** we observe events such as people meeting, corresponding, voting, or sharing information [47; 82; 65; 63; 83];
- **Biological neural networks:** spiking action potentials can trigger or inhibit spikes in neighboring neurons according to time-varying functional networks [66; 84; 85; 86; 87; 88; 89].
- **Financial networks:** stock market shocks can trigger jumps across the global network of financial instruments and indices[90; 91; 92];
- **Epidemiological networks:** as a contagion spreads through a community, observations of symptoms in one person are strong predictors of future symptoms among that person's neighbors [93]; and

- **Seismological networks:** substantial seismic activity is often predicated by foreshocks and followed by aftershocks, with the epicenter of these shock events determined by the local geography and plate tectonics [94; 95; 96].

In all the above settings, the interactions between actors are critical to a fundamental understanding of the underlying network structure and accurate predictions of likely future events.

We can model these interactions between nodes using a network or graph, where directed edge weights correspond to the degree to which one node’s activity stimulates activity in another node. For instance, the network structure may indicate who is influencing whom within a social network, or the connectivity of neurons. In these and other contexts the underlying network structure may be changing over time, for instance as people’s relationships evolve or as a function of the activity in which the brain is engaged. In many cases, we are interested in both the rates at which different nodes or actors participate in events and the underlying network structure.

Our goal is to filter and track such processes. We present methods and associated theoretical performance bounds in two settings: (a) where the underlying network structure is known and (b) where the underlying network structure is unknown. Our approach incorporates concepts and tools from multivariate Hawkes model of point processes [11; 12; 13] and online convex optimization methods for dynamic environments described in Chapter 2. In particular, the multivariate Hawkes process is akin to an autoregressive model for point processes, where events up to time t dictate the rate at which events are anticipated after time t .

Estimating the parameters associated with these processes is the subject of much current research, but existing methods typically assume that the underlying network parameters are static rather than changing with time, and require computationally-prohibitive batch processing algorithms. Specifically, there has been substantial work

in estimating parameters of the system through methods which seek to estimate both the “parent event” of each event, and then use this information to learn the parameters of the influence function and/or the network [97] using either EM-type algorithms or Bayesian techniques [98; 99; 96; 100; 96; 64]. The difficulty with using this approach in the online setting is that in order to accurately estimate parent events, we need a potentially large buffer of stored event times, and do processing that scales poorly with the number of previously observed events. The work closest to ours is [101], which uses a Bayesian framework to learn the parameters of a discretized version of the Hawkes process, which is computationally more efficient with regards to the number of events observed. However, they still require at least mini-batches and having access to data more than once, which is not a truly streaming setting. Additionally, all of these methods require the data to exactly follow the Hawkes model, and have no guarantees for performance in the case of misspecified influence functions or generative model, whereas our results both theoretically and empirically offer protection against such model mismatch.

In the settings described above, we face several key challenges:

- (a) the underlying networks are dynamic,
- (b) we receive either a large volume of data or data that is streaming at a high velocity, necessitating sequential processing, and
- (c) we seek performance guarantees that are robust to model mismatch (i.e. perform well even when the data was not truly generated by the Hawkes model).

Our proposed method will simultaneously track the time-varying rates at which events are expected and the underlying time-varying network structure. In contrast, most methods assume that the rates are a known, closed-form function of the observed data and the network structure, and focus solely on estimating the network;

we will see that this approach is more fragile with respect to modeling errors. Additionally, due to the streaming nature of our algorithms, we are in a regime where we can easily do forecasting, which is valuable in financial, epidemiological, seismological and other networks. Our algorithms create an estimate of the process rates at every time before seeing the actual events at the time. Therefore, the online framework allows us to do one step ahead forecasting. This would be much harder to do using previous methods which learn networks in Hawkes processes, because to do prediction all the previous data would have to be processed, which is computationally intensive, and then projected forward to new time points. These methods would either have to be run at every time point, which is computationally infeasible, or would only be run a few times and not be able to track short term changes in the network.

The remainder of the chapter is organized as follows: Section 4.2 introduces the basics of the Hawkes process and a mathematical description of our learning objective. Section 4.3 reviews some of the basics for online learning with dynamics in and why it should be applied to this setting. Section 4.4 then describes the time discretized loss function and dynamical model which corresponds to the Hawkes process which are needed for our online learning framework. Section 4.5 introduces our two proposed online algorithms for tracking these processes, one which assumes the network is known and is based on the DMD algorithm (Algorithm 2.1) and one which attempts to learn both the time varying-rates as well as the network simultaneously, based on the DMD in exponential families algorithm (Algorithm 2.3). Section 4.6 has a brief discussion on the computational complexity of the methods. Finally Section 4.7 shows how our methods perform in practice on synthetic data both when the generative model is known and when it is misspecified, as well as experiments performed on the Memetracker data set. Proofs are placed in the appendix.

4.2 Problem Formulation

We monitor p actors in a network, and record the identities of the actor and time of each event. An actor and event may represent a person “liking” a photo or article shared by another person in a social network, a neuron firing in the brain, or the incidence of disease. That is, we observe a time series of the form $\{(k_n, \tau_n)\}_n$, where $k_n \in \{1, 2, \dots, p\}$ is the actor involved in the n^{th} event and $\tau_n \in \mathbb{R}_+$ is the time at which it occurs. With each new event, we wish to accurately predict which events are most likely in the immediate future and the underlying network of influence. We define $\tau_0 \triangleq 0$.

We wish to track the time-varying rate for each of the p actors. To do so, we adopt a multivariate Hawkes process model [11; 12; 13] and track the parameters of this model over time. In particular, for each actor k we have a point process with time-varying rate function $\mu_k(\tau)$. Let $N_{k,\tau}$ denote the number of recorded events for actor k up to and including time τ , and let $N_\tau \triangleq \sum_{k=1}^p N_{k,\tau}$ denote the total number of events (across all actors) up to and including time τ . The likelihood of actor k participating in an event between times t_1 and t_2 is controlled by the integral $\int_{t_1}^{t_2} \mu_k(\tau) d\tau$. More formally, the collection of all observed events up to time T can be denoted

$$\mathcal{H}^T \triangleq \{N_{k,\tau}\}_{\substack{k \in \{1, \dots, p\} \\ \tau \in (0, T]}}$$

The log likelihood of observing \mathcal{H}^T given the p rate functions $\mu_k(t)$ for $k \in \{1, \dots, p\}$ is then [102]

$$\log p(\mathcal{H}^T | \mu) = \sum_{n=1}^{N_T} \log \mu_{k_n}(\tau_n) - \sum_{k=1}^p \int_0^T \mu_k(\tau) d\tau. \quad (4.1)$$

Thus far, everything explained is common to a wide class of point processes. The multivariate Hawkes processes considered in this paper are essentially an autoregres-

sive point process, where each rate function $\mu_k(\tau)$ depends on the history of past events, H^τ . In particular, a multivariate Hawkes process assumes the rate functions can each be expressed as

$$\mu_k(\tau) = \bar{\mu}_k + \sum_{n=1}^{N_\tau} h_{k,k_n}(\tau - \tau_n). \quad (4.2)$$

Here $\bar{\mu}_k$ is a baseline rate representing the nonzero likelihood of actor k acting even without having been influenced by any previous actions, with $\bar{\mu} \triangleq [\bar{\mu}_1, \dots, \bar{\mu}_p]^\top$. Furthermore, we have p^2 functions of the form $h_{k_1,k_2}(\tau)$ which describe how events associated with actor k_2 will impact the likelihood of events associated with actor k_1 . In order to assure causality we assume $h_{k_1,k_2}(\tau) = 0$ if $\tau \leq 0$ for all $(k_1, k_2) \in \{1, \dots, p\}^2$. These functions depend on the underlying network connectivity; if actors k_1 and k_2 are unconnected, the corresponding function h_{k_1,k_2} should be identically zero for all τ .

One of the main challenges in statistical estimation for multivariate Hawkes processes is the estimation of these p^2 functions. In general, this problem is highly underdetermined and challenging. Recent work has attempted to mitigate these challenges using low-rank and sparse models [102; 103; 104]. In this paper, we make the common (*cf.* [83; 105]) simplifying assumption that these interactions all have the same functional form but different (and often zero-valued) amplitudes, so that

$$h_{k_1,k_2}(\tau) = W_{k_1,k_2} h(\tau) \quad (4.3)$$

where $h(\tau)$ is known but the amplitude matrix $W \in \mathbb{R}_+^{p \times p}$ may be unknown. We will refer to $h(\tau)$ as the *influence function*, as it depicts how an action's influence on an actor will vary in time. The matrix W indicates the strength of influence between actors; from a graph theory perspective, W acts like the weighted adjacency matrix of a graph representation of the network.

Our goal is to obtain an estimate for $\mu(t)$ as it evolves and to infer W *online* from streaming network data. Furthermore, we seek methods with performance guarantees that hold even when the observed data is not generated strictly in accordance with the above Hawkes model. That is, while we use the Hawkes model to measure how well estimates fit the data, we recognize that the model will never be perfectly accurate (*e.g.*, we may have errors in our estimate of the influence function $h(\tau)$ or the linear model in (4.2) may not reflect nonlinearities present in real data) and wish performance guarantees even in the face of these uncertainties.

4.3 Online Learning

As described above, we wish to estimate the rate functions $\mu_k(\tau)$ for $k = 1, \dots, p$ and the corresponding likelihood of future events, based solely on previous events and the (possibly learned) network structure. In this section, we describe several key ideas from the field of online learning which we will leverage in our problem.

As described in chapter 2, online learning techniques are generally based on the following paradigm: at every time point t we make a prediction, receive some data, and then do a few computationally inexpensive calculations to improve our previous prediction. In the setting of autoregressive event tracking, this means we would have an estimate about each actor's likelihood of acting and then see who actually does act. Using the previous prediction, the current actions, and information about the network itself, we update our belief of who is most likely to act next. Unlike traditional online learning techniques, there are strong dynamics involved in the evolution of the system that must be incorporated.

Once again, we characterize the efficacy of $\hat{\boldsymbol{\theta}}_T \triangleq (\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_T) \in \Theta^\top$ relative to a comparator sequence $\boldsymbol{\theta}_T \triangleq (\theta_1, \theta_2, \dots, \theta_T) \in \Theta^\top$ as follows:

Definition 4.1 (Regret). *The regret of $\hat{\boldsymbol{\theta}}_T$ with respect to a comparator $\boldsymbol{\theta}_T \in \Theta^\top$ is*

$$R_T(\boldsymbol{\theta}_T) \triangleq \sum_{t=1}^T \ell_t(\hat{\boldsymbol{\theta}}_t) - \sum_{t=1}^T \ell_t(\boldsymbol{\theta}_t).$$

The comparator series can be thought of as the predictions from either an oracle with knowledge of future data or a batch algorithm with access to all the data. Therefore, the regret characterizes the amount of excess loss suffered from the online algorithm. Previous work proposed algorithms which yielded regret of $O(\sqrt{T})$ for *static* comparators $\boldsymbol{\theta}_t$, where $\boldsymbol{\theta}_t = \boldsymbol{\theta}$ for all t (cf. [19; 18; 21]). Basically, these methods can only perform well if the comparator is a single point or changes either very slowly or very infrequently. It is this characteristic that causes most existing methods to be poorly-suited to the autoregressive nature of interactions within a network. However, the ability to incorporate a dynamical model like Dynamic Mirror Descent (DMD) leads to significant improvements in performance in these dynamic environments.

In the context of multivariate Hawkes processes, a known dynamical model amounts to knowing the exact weighted adjacency structure of the network. In many problems the network structure may be unavailable in practical contexts, and will need to be estimated simultaneously with the rates. This will be discussed further in Section 4.5.2

Before defining an optimization routine specifically for multivariate Hawkes data, we briefly review the main advantage of the DMD method [106]. Let $\Phi_t : \Theta \times \mathcal{W} \mapsto \Theta$ denote a sequence of known dynamical models that takes as input a value in our decision space and some side information. By including Φ_t in the process, we effectively search for a predictor which (a) attempts to minimize the loss and (b) adheres closely to the trajectory defined by Φ_t with side information W . In our setting W will correspond to the known or estimated values of the network. This is similar to a stochastic filter which alternates between using a dynamical model to

update the “state”, and then uses this state to perform the filtering action. However, we make no assumptions about Φ_t ’s relationship to the true underlying parameters. It has been shown, under mild conditions on the sequence $\{\Phi_t\}_{t>0}$, that the regret of this algorithm obeys the following:

$$R_T(\boldsymbol{\theta}_T) \leq C\sqrt{T} \left(1 + \sum_{t=1}^{T-1} \|\theta_{t+1} - \Phi_t(\theta_t, W)\| \right)$$

for some $C > 0$ independent of T . This bound scales with the comparator sequence’s deviation from the sequence of dynamical models $\{\Phi_t\}_{t>0}$ – a stark contrast to previous tracking regret bounds which are only sublinear in T for comparators which change slowly with time or at a small number of distinct time instances. This is where the large advantage of using these types of methods comes into play in autoregressive settings, which will generally not change slowly, but instead will change according to some dynamic model. In order to use this framework to learn the rates and network of a Hawkes process we need to derive two key ingredients, the loss function and the dynamical model. These ingredients will take us from the general setup presented in this section, to the specific application being studied. These functions will be derived in the next section. Once these have been derived, we can use and expand upon the existing theory for online learning.

4.4 Loss Function and Dynamic Model

In order to analyze and make estimates of our point process network data, we must use the Hawkes model described in Section 4.2 to define a loss function, Bregman divergence, dynamical models, and other ingredients of the online learning framework described in Section 4.3.

4.4.1 Time Discretized Loss Function

We discretize time into intervals of length $\delta > 0$, where δ is small enough that it is very infrequent that the same actor acts multiple times in the same time window. (For simplicity, we assume the total sensing time, T , is selected such that T/δ is an integer.) We let $t = 1, 2, \dots, T/\delta$ index these intervals, and note $N_{k,t\delta}$ is the number of events observed in the k^{th} process (i.e. by the k^{th} actor) up to the end of the t^{th} interval, $((t-1)\delta, t\delta]$, with $N_{k,0} \triangleq 0$.

The value $x_{t,k} = N_{k,t\delta} - N_{k,t(\delta-1)}$ denotes how many times actor k acted during the t^{th} interval, which will mostly be either zero or one for an appropriately chosen δ . The vector $x_t \triangleq [x_{t,1}, \dots, x_{t,p}]^{\top}$ will be our data vector at each time point. Using the negative log likelihood of the Hawkes process up to time δt , we can formulate appropriate loss functions to apply to an online setting. We introduce an approximation of the Hawkes process, $\lambda_t = [\lambda_{t,1}, \dots, \lambda_{t,p}]^{\top} \in \mathbb{R}_+^p$. For this approximation and in the rest of the paper, we denote the summation over a set of events $\{n : \bar{\tau}_n < \delta t\}$ by simply saying we sum over $\bar{\tau}_n < \delta t$.

$$\lambda_{t,k} = \bar{\mu}_k + \sum_{\bar{\tau}_n < \delta t} W_{k,k_n} h(\delta t - \tau_n) \quad (4.4)$$

Here we define a new set of times $\{\bar{\tau}_n\}_n$ which are the ends of the discrete time intervals that the events occur. These times are defined by $\bar{\tau}_n = \lceil \frac{\tau_n}{\delta} \rceil \delta$. Equation 4.4 acts the same way as the original Hawkes process, but we do not immediately update the rates with the events as they occur but instead push them to integer multiples of δ . Notice that although we wait until $\bar{\tau}_n$ to include the event in the rate, we update it with the full knowledge of when the event actually occurred i.e. we use $h(\tau - \tau_n)$ instead of $h(\tau - \bar{\tau}_n)$. Equation 4.4 suggests a loss function with analogous changes to the original, continuous time log likelihood. We define $L_T(\mu)$ to be the negative log likelihood of the Hawkes process at time T , and $L_T^{(\delta)}(\lambda)$ to be the discrete time

equivalent.

$$\begin{aligned}
L_T(\mu) &\triangleq \sum_{k=1}^p \int_0^T \mu_k(\tau) d\tau - \sum_{n=1}^{N_T} \log \mu_{k_n}(\tau_n) \\
&\approx \sum_{k=1}^p \left(\sum_{t=1}^{T/\delta} \delta \lambda_{t,k} - \sum_{t=1}^{T/\delta} x_{t,k} \log \lambda_{t,k} \right) \triangleq L_T^{(\delta)}(\lambda)
\end{aligned} \tag{4.5}$$

This new loss function is based on replacing the integral term with a summation and replacing $\mu_{k_n}(\tau_n)$ with $\lambda_{\bar{\tau}_n/\delta, k_n}$. Both of these substitutions become closer to the truth as $\delta \rightarrow 0$. In Lemma 4.2 we characterize the difference between the two functions.

Lemma 4.2. *Given the influence function $h(\tau) = \alpha^\tau \mathbf{1}_{[\tau>0]}$ and data with a maximum activity rate of x_{\max} events per actor per unit time, the negative log likelihood of the true Hawkes Process, given in Equation 4.1, and the approximate negative log likelihood for the discrete time rate, given in Equation 4.5, both generated by the same matrix W and vector $\bar{\mu}$ with all elements $0 \leq W_{i,j} \leq W_{\max}$, there exists a constant $C > 0$ depending on x_{\max}, p, W_{\max} and α such that*

$$|L_T(\mu) - L_T^{(\delta)}(\lambda)| \leq CN_T\delta.$$

Remark: *For a general influence function $h(\tau)$ which is Lipschitz on $(0, T]$, a similar proof gives a slightly higher bound of $C(TN_T\delta + N_T \log(1 + N_T\delta))$. As we focus mostly on $h(\tau) = \alpha^\tau \mathbf{1}_{[\tau>0]}$, we show the bound for this specific function.*

This Lemma says that if δ is set small enough, the discrete approximation can be used in a learning algorithm, without many errors coming from the discretization approximation. However, the smaller δ is the more frequently the rates will have to be updated, leading to a higher computational burden.

The proposed method uses the loss function:

$$\tilde{\ell}_t(\lambda_t) \triangleq -\langle \log(\delta \lambda_t), x_t \rangle + \langle \delta \lambda_t, \mathbf{1} \rangle. \tag{4.6}$$

Here and for the remainder of the chapter, \log and \exp of a vector are assumed to be taken element-wise. Notice that $L_T^\delta(\lambda) = \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t) + \langle x_t, \log \delta \rangle$, and thus the total cumulative loss is the summation of instantaneous losses and a term which is independent of the rate estimate. We now introduce a change of variables, defining $\theta_t = [\theta_{t,1}, \dots, \theta_{t,p}]^\top \triangleq \log(\delta \lambda_t)$. Using this change of variable gives a loss function in terms of θ .

$$\ell_t(\theta_t) = \tilde{\ell}_t \left(\frac{1}{\delta} \exp(\theta_t) \right) = -\langle \theta_t, x_t \rangle + \langle \exp(\theta_t), \mathbf{1} \rangle. \quad (4.7)$$

It is important to note that this is a one-to-one relationship between λ_t and θ_t , and thus we may operate in either the λ or θ space. Notice that Equation 4.7 corresponds to the negative log-likelihood of an exponential family distribution of the form

$$p_t(\theta) = \exp \{ \langle \theta, x_t \rangle - Z(\theta) \}$$

where we omit factors depending only on x_t and not θ , and where

$$Z(\theta) \triangleq \log \int \exp \{ \langle \theta, x \rangle \} dx$$

is the so-called *log-partition function*. Important to our analysis is the dual of the log-partition function, $Z^*(\delta\lambda) \triangleq \sup_{\theta \in \Theta} \{ \langle \delta\lambda, \theta \rangle - Z(\theta) \}$. In our multivariate Hawkes setting, we have $Z(\theta) = \langle \exp(\theta), \mathbf{1} \rangle$. Performing online optimization in the θ parameter space allows us to exploit several properties of exponential families, as described in general settings in [47] and in our specific context in Section 4.5.2. In particular, we will use the following facts:

$$\begin{aligned} \nabla Z(\theta) &= \exp(\theta) = \delta\lambda \\ \nabla Z^*(\delta\lambda) &= \log(\delta\lambda) = \theta \end{aligned}$$

We list some important properties of the loss function $\ell_t(\theta)$ that will be used in the proof of the regret bounds, in section 4.5.

- We assume a convex set of possible rate functions $\lambda \in [\lambda_{\min}, \lambda_{\max}]^p$ with $\lambda_{\min} > 0$, and therefore the feasible set for θ is equivalently $\Theta = [\log(\delta\lambda_{\min}), \log(\delta\lambda_{\max})]^p$.
- We will assume that there is a maximum number of times each actor can act per time unit, x_{\max} , and the observations space \mathbf{X} is simply $[0, \delta x_{\max}]^p$. Furthermore, on the set Θ :

$$\begin{aligned} \|\nabla \ell_t(\theta)\|_2 &\leq \|\exp(\theta)\|_2 + \|x_t\|_2 \\ &\leq \sqrt{p}\delta(\lambda_{\max} + x_{\max}) \end{aligned} \quad (4.8)$$

- The function $Z(\theta) = \langle \mathbf{1}, \exp(\theta) \rangle$ is strongly convex on Θ with strong convexity parameter $\delta\lambda_{\min}$ with respect to the ℓ_2 norm. Therefore the Bregman divergence induced by Z obeys the following:

$$D(\theta_1 \parallel \theta_2) = Z(\theta_1) - Z(\theta_2) - \langle \nabla Z(\theta_2), \theta_1 - \theta_2 \rangle \geq \frac{\delta\lambda_{\min}}{2} \|\theta_1 - \theta_2\|_2^2 \quad (4.9)$$

- The Bregman divergence induced by $Z(\theta) = \langle \mathbf{1}, \exp(\theta) \rangle$ is always non-negative and can be upper bounded for any $\theta_1, \theta_2 \in \Theta$:

$$\begin{aligned} D(\theta_1 \parallel \theta_2) &= Z(\theta_1) - Z(\theta_2) - \langle \nabla Z(\theta_2), \theta_1 - \theta_2 \rangle \\ &\leq \langle \nabla Z(\theta_1) - \nabla Z(\theta_2), \theta_1 - \theta_2 \rangle \\ &\leq \|\nabla Z(\theta_1) - \nabla Z(\theta_2)\|_2 \|\theta_1 - \theta_2\|_2 \\ &= \delta \|\lambda_1 - \lambda_2\|_2 \left\| \log \left(\frac{\lambda_1}{\lambda_2} \right) \right\|_2 \leq \frac{\delta \lambda_{\max}^2 p}{\lambda_{\min}} \end{aligned} \quad (4.10)$$

4.4.2 Dynamical Models

In order to use the DMD framework, we model the autoregressive nature of the Hawkes process using a series of data-dependent dynamical models, $\tilde{\Phi}_t$, which update a rate parameter λ given a weighted adjacency matrix W and the previously observed

data \mathcal{H}^t . We can model this dependence using the dynamical model

$$\tilde{\Phi}_t(\lambda, W) = A_t \lambda + W y_t + c_t,$$

for $\lambda, y_t, c_t \in \mathbb{R}^p$, $A_t \in \mathbb{R}^{p \times p}$, and $W \in \mathbb{R}_+^{p \times p}$. If we let $A_t = \beta I$ for some $\beta \in [0, 1]$, where I is the identity matrix, it suggests that our dynamical model causes the rates in λ to decay at a rate depending on β in the absence of other effects. The term $W y_t$ allows us to model autoregressive effects. In particular, the matrix W could correspond to a weighted adjacency matrix associated with the network of interest, and y_t could contain information about previous events as specified below. More generally, we might replace the term $W y_t$ with $\sum_{r=0}^{m-1} W y_{t-r}$ for an m^{th} -order process if we thought there should be some latency in the response times for pairs of actors.

Recall that the influence functions $h(\tau)$ describe how the causal influence between actors varies over time. Dynamical models for various forms of $h(\tau)$ can be developed for the time discretized multivariate Hawkes process described in Equation 4.4. First, a general function $h(\tau)$ is considered, with some mild assumptions, and then is used to derive models for some specific choices of $h(\tau)$.

- **General influence functions:** We assume $h(\tau)$ is a continuous, non-negative function on $\tau > 0$. Additionally, we assume if $h(B) = 0$ for some $B > 0$, then $h(B + t) = 0$ for any $t > 0$. Finally, let $e_{k_n} \in \mathbb{R}^p$ be a vector of all zeros, with a single 1 in the k_n^{th} entry indicating the actor involved in the n^{th} action. Then

we derive the following dynamical model:

$$\begin{aligned}
\lambda_{t+1} &= \bar{\mu} + \sum_{\bar{\tau}_n < \delta(t+1)} We_{k_n} h(\delta(t+1) - \tau_n) \\
&= \bar{\mu} + \sum_{\bar{\tau}_n < \delta t} a_{t,n} We_{k_n} h(\delta t - \tau_n) \\
&\quad + \sum_{\bar{\tau}_n = \delta t} We_{k_n} h(\delta(t+1) - \tau_n) \\
&= \bar{\mu} + A_t \sum_{\bar{\tau}_n < \delta t} We_{k_n} h(\delta t - \tau_n) + W y_t \\
&= A_t \lambda_t + W y_t + (I - A_t) \bar{\mu}
\end{aligned}$$

In the above we have used the following values $a_{t,n}$, A_t and y_t :

$$\begin{aligned}
a_{t,n} &\triangleq \begin{cases} 1, & h(\delta t - \tau_n) = 0 \\ \frac{h(\delta(t+1) - \tau_n)}{h(\delta t - \tau_n)}, & \text{else} \end{cases} \\
A_{t,k} &\triangleq \begin{cases} 1, & \sum_{\bar{\tau}_n < \delta t} W_{k,k_n} h(\delta t - \tau_n) = 0 \\ \frac{\sum_{\bar{\tau}_n < \delta t} a_{t,n} W_{k,k_n} h(\delta t - \tau_n)}{\sum_{\bar{\tau}_n < \delta t} W_{k,k_n} h(\delta t - \tau_n)}, & \text{else} \end{cases} \\
A_t &= \text{Diag}(A_{t,1}, A_{t,2}, \dots, A_{t,p}) \\
y_t &\triangleq \sum_{\bar{\tau}_n = \delta t} e_{k_n} h(\delta(t+1) - \tau_n)
\end{aligned}$$

Thus, we have the dynamics in the desired form.

$$\lambda_{t+1} = \tilde{\Phi}_t(\lambda_t, W) = A_t \lambda_t + W y_t + (I - A_t) \bar{\mu}$$

Notice that in general A_t may be a function of W .

- **Rectangular influence functions:** Using the above framework, dynamical models can be worked out for the specific instance when $h(\tau) = \mathbf{1}_{[0 < \tau < B]}$ for

some positive $B > \delta$. We first show the values $a_{t,n}$.

$$\begin{aligned} a_{t,n} &= \begin{cases} 1, & \mathbf{1}_{[0 < \delta t - \tau_n < B]} = 0 \\ \mathbf{1}_{[0 < \delta(t+1) - \tau_n < B]} / \mathbf{1}_{[0 < \delta t - \tau_n < B]}, & \text{else} \end{cases} \\ &= \mathbf{1}_{[\tau_n \leq \delta t - B]} + \mathbf{1}_{[\tau_n > \delta(t+1) - B]} \end{aligned}$$

This leads to the following form of $A_{t,k}$.

$$A_{t,k} = \begin{cases} 1, & \sum_{\substack{\bar{\tau}_n < \delta t \\ \delta t - \tau_n < B}} W_{k,k_n} = 0 \\ \sum_{\substack{\bar{\tau}_n < \delta t \\ \delta(t+1) - B < \tau_n}} W_{k,k_n} / \sum_{\substack{\bar{\tau}_n < \delta t \\ \delta t - B < \tau_n}} W_{k,k_n}, & \text{else} \end{cases}$$

Notice that the elements of A_t , are the weighted ratio of how many events influence the current rate compared to how many events influence the rate at the previous time point. Importantly, notice that $A_{t,k} \leq 1$.

- **Exponential influence functions:** Here we consider influence functions of the form

$$h(\tau) = \alpha^\tau \mathbf{1}_{[\tau > 0]}$$

for $\alpha \in (0, 1)$. We then have

$$\begin{aligned} \lambda_{t+1} &= \bar{\mu} + \sum_{\bar{\tau}_n < \delta(t+1)} W e_{k_n} h(\delta(t+1) - \tau_n) \\ &= \bar{\mu} + \sum_{\bar{\tau}_n < \delta(t+1)} W e_{k_n} \alpha^{\delta(t+1) - \tau_n} \\ &= \bar{\mu} + \alpha^\delta \sum_{\bar{\tau}_n < \delta t} W e_{k_n} \alpha^{\delta t - \tau_n} + \sum_{\bar{\tau}_n = \delta t} W e_{k_n} \alpha^{\delta(t+1) - \tau_n} \\ &= (1 - \alpha^\delta) \bar{\mu} + \alpha^\delta \lambda_t + W y_t \end{aligned}$$

yielding the dynamical model

$$\tilde{\Phi}_t(\lambda, W) = \alpha^\delta \lambda + W y_t + (1 - \alpha^\delta) \bar{\mu}.$$

- **Delayed exponential influence functions:** The exponential decay might be a reasonable influence function, however reactions might not always be able to take place immediately. To model this we use $h(\tau) = \alpha^{\tau-D} \mathbf{1}_{[\tau>D]}$ for some positive delay $D \geq \delta$. In this scenario, a similar dynamical model can be derived as with the exponential decay, but with slight change in the additive Wy_t term:

$$\begin{aligned}
\lambda_{t+1} &= \bar{\mu} + \sum_{\bar{\tau}_n < \delta(t+1)} We_{k_n} \alpha^{\delta(t+1)-\tau_n-D} \mathbf{1}_{[(\delta(t+1)-\tau_n > D)]} \\
&= \bar{\mu} + \sum_{\tau_n < \delta(t+1)-D} We_{k_n} \alpha^{\delta(t+1)-\tau_n-D} \\
&= \bar{\mu} + \alpha^\delta \sum_{\tau_n + D < \delta t} We_{k_n} \alpha^{\delta t - \tau_n - D} \\
&\quad + \sum_{\delta t \leq \tau_n + D < \delta(t+1)} We_{k_n} \alpha^{\delta(t+1)-\tau_n-D} \\
&= \alpha^\delta \lambda_t + Wy'_t + (1 - \alpha^\delta) \bar{\mu} \\
y'_t &\triangleq \sum_{\delta t - D \leq \tau_n < \delta(t+1) - D} e_{k_n} \alpha^{\delta(t+1)-\tau_n-D}
\end{aligned}$$

This takes the same basic form as the non-delayed exponential, but with a slightly different term y'_t instead of y_t . Also, notice that when D is equal to δ these equations become equivalent to the non-delayed version, suggesting that the time discretization is creating estimation error on the order of a slight delay in the influence function.

In the general setting, the dynamical model is written in the form

$$\tilde{\Phi}_t(\lambda, W) = A_t \lambda + Wy_t + c_t \quad (4.11a)$$

for some linear operator A_t , a vector y_t which is a known function of h and previously observed data, and a known constant c_t . In the following, we assume a generic dynamical model of the form (4.11a). Furthermore, note that an equivalent dynamical

model can be defined in the θ parameter space as

$$\Phi_t(\theta, W) = \nabla Z^*(\delta \tilde{\Phi}_t(\nabla Z(\theta)/\delta, W)). \quad (4.11b)$$

The first method we present will depend on the ease of computation of the matrix A_t , and if many observations need to be held in memory to compute A_t , then the methods could be quite slow. However, if the influence function is the exponential decay or the delayed exponential decay, then A_t is constant in time thus we only need to compute it once. Another important feature of both the exponential decay and delayed exponential decay functions is that the linear operator A_t does not depend on the values of the matrix W . It is this separation that will allow simultaneously estimation of the rates, λ and the values of W .

4.5 Proposed Algorithms

Our main contribution is to propose two algorithms, depending on whether or not the weighted adjacency matrix W is known, and we show relevant regret bounds for both.

4.5.1 DMD Algorithm - W Known

We first show a method for tracking the rate vector λ_t from streaming observations x_t for $t = 1, 2, \dots, T/\delta$ using Algorithm 4.1. The basic idea is the following: at time t we start with the current rate estimate $\hat{\lambda}_t$ and the dual, $\hat{\theta}_t$. We then observe x_t and incur the loss $\ell_t(\hat{\theta}_t)$. Based on this incurred loss, we update our prediction of the quantity $\tilde{\theta}_{t+1}$, which can be thought of as an *a posteriori* estimate of the dual at time t , given all the data up to and including t . From here, we make our prediction of the rate at the next time by applying our dynamic model, Φ_t , and converting back to λ parameter space.

Algorithm 4.1 MV Hawkes Tracking - W Known

- 1: Initialize $\hat{\lambda}_1 = \bar{\mu}$, $\hat{\theta}_1 = \log \delta \hat{\lambda}_1$
 - 2: **for** $t = 1, \dots, T/\delta$ **do**
 - 3: Observe x_t and incur loss $\tilde{\ell}_t(\hat{\lambda}_t) = \langle \mathbf{1}, \delta \hat{\lambda}_t \rangle - \langle x_t, \log(\delta \hat{\lambda}_t) \rangle$
 - 4: Set $\tilde{\theta}_{t+1} = \arg \min_{\theta \in \Theta} \eta_t \langle \nabla \ell_t(\hat{\theta}_t), \theta \rangle + D(\theta \| \hat{\theta}_t)$
 - 5: Set $\hat{\theta}_{t+1} = \Phi_t(\tilde{\theta}_{t+1}, W)$
 - 6: Set $\hat{\lambda}_{t+1} = \exp(\hat{\theta}_{t+1})/\delta$
 - 7: **end for**
-

Before proving regret bounds for this algorithm, we note that the dynamical functions Φ_t must be contractive for the algorithm to have provably good performance. By definition, we say that a dynamical model, Φ_t is contractive with respect to the Bregman divergence D on the set Θ if for any $\theta_1, \theta_2 \in \Theta$ we have:

$$D(\Phi_t(\theta_1, W) \| \Phi_t(\theta_2, W)) - D(\theta_1 \| \theta_2) \leq 0.$$

This is a condition which works to ensure some amount of stability in the output sequence by preventing small estimation errors at any one time step from getting worse and worse as the algorithm continues.

Lemma 4.3. *If the dynamical model $\Phi_t(\theta, W) = \log(A_t \exp(\theta) + b_t)$, where A_t is a diagonal matrix with all elements in the range $[0, 1]$ for all t and $b_t \geq 0$, then Φ_t is contractive with respect to the Bregman divergence induced by $Z(\theta) = \langle \mathbf{1}, \exp(\theta) \rangle$ on $\Theta = [\log(\delta \lambda_{\min}), \log(\delta \lambda_{\max})]^p$.*

All the dynamical models we have proposed satisfy the conditions that A_t is diagonal. Additionally, $b_t \geq 0$ as long as all elements of W and $\bar{\mu}$ are non-negative and $h(t) \geq 0$. The most restrictive assumption that this lemma makes is that the elements of A_t are upper bounded by one, which is true if $h(t)$ is non-increasing after the initial impulse.

Using Algorithm 4.1 combined with the fact that the dynamics are contractive, leads to the following result, which bounds the amount of excess error of the output

sequence generated by the algorithm compared to any comparator sequence.

Theorem 4.4 (Tracking regret of Algorithm 4.1). *Using the Bregman divergence induced by $Z(\theta) = \langle \mathbf{1}, \exp(\theta) \rangle$, and a contractive dynamical model $\Phi_t(\theta, W)$ for all t , if we choose η_t proportional to either $1/\sqrt{t}$ or $1/\sqrt{T/\delta}$, then there exists a constant $C > 0$ depending on $\delta, p, x_{max}, \lambda_{max}$ and λ_{min} such that the regret of $\hat{\lambda}_1, \hat{\lambda}_2, \dots, \hat{\lambda}_{T/\delta}$ generated by Algorithm 4.1 for any data sequence $x_1, x_2, \dots, x_{T/\delta} \in [0, x_{max}]^p$ with respect to a comparator sequence $\lambda_1, \dots, \lambda_{T/\delta} \in [\lambda_{min}, \lambda_{max}]^p$ is bounded by:*

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \tilde{\ell}_t(\lambda_t) \leq C \left(1 + \sum_{t=1}^{T/\delta} \|\lambda_{t+1} - \tilde{\Phi}_t(\lambda_t)\|_2 \right) \sqrt{T}.$$

There are a few important things revealed by this regret bound. The first is that if the complexity measure of the comparator sequence relative to the dynamics $\tilde{\Phi}_t$ is low, then the algorithm has \sqrt{T} regret, which is sublinear as desired. Secondly, no assumptions have been made about how the comparator sequence λ was actually generated. Instead we simply measure how well it is approximated by a Hawkes process with dynamics dictated by $\tilde{\Phi}_t$. Therefore, if the true process acts like a Hawkes process, there will be low regret, but if the sequence is not generated this way or is generated as a Hawkes process with different parameters such as a different W matrix, or a different influence function, we have an understanding about how much this will influence the performance of the algorithm.

4.5.2 Proposed Algorithm - W Unknown

When the influence function $h(t)$ was a decaying exponential function, the dynamical model used had the form $\tilde{\Phi}_t(\lambda, W) = A_t \lambda + W y_t + (I - A_t) \bar{\mu}$, where $A_t = \alpha^\delta I$ was independent of the value of W . This fact paired with the additional assumption that the solution to line 4 of Algorithm 4.1 is a point on the interior of Θ , allows for a method of tracking both the rates $\lambda_1, \dots, \lambda_T$ as well as the matrix W . We denote

$\lambda_t^{(W)}$ as the estimate at time t of Algorithm 4.1 using matrix W in line 5. When the solution $\tilde{\theta}_{t+1}$ is on the interior of the set Θ , the value of $\hat{\lambda}_{t+1}^{(W)}$ takes the form:

$$\hat{\lambda}_{t+1}^{(W)} = (1 - \eta_t)\alpha^\delta \hat{\lambda}_t^{(W)} + \eta_t \alpha^\delta \frac{x_t}{\delta} + W y_t + (1 - \alpha^\delta)\bar{\mu} \quad (4.12)$$

It is this closed form solution that leads to Lemma 4.5. It would seem that the assumption that $\hat{\theta}_{t+1} \in \text{Int}(\theta)$ would be very restrictive. However, since we are already assuming that there is a maximum amount of times any actor can act per unit time of x_{\max} , setting $\lambda_{\max} \geq x_{\max}$ insures the condition on $\tilde{\theta}_{t+1}$. Therefore, our space Θ would be a bounded region, but the solution of line 4 would always be on the interior of this feasible set under the same assumptions as Theorem 4.4.

Lemma 4.5. *If Algorithm 4.1 is run separately for W_1 and W_2 producing estimates $\hat{\lambda}_t^{(W_1)}$ and $\hat{\lambda}_t^{(W_2)}$ respectively at time t , with the dynamical model $\tilde{\Phi}_t(\lambda, W) = \alpha^\delta \lambda + W y_t + (1 - \alpha^\delta)\bar{\mu}$, and assuming that the value $\tilde{\theta}_{t+1}$ is always in the interior of Θ , then at any given point in time the predictions of the algorithms corresponding to W_1 and W_2 will be related in the following way:*

$$\hat{\lambda}_t^{(W_1)} = \hat{\lambda}_t^{(W_2)} + (W_1 - W_2)K_t$$

with

$$K_{t+1} = (1 - \eta_t)\alpha^\delta K_t + y_t, \quad K_1 = \mathbf{0}.$$

Using this Lemma, the losses that would have been incurred with a different weighted adjacency matrix W can be calculated and used to update \widehat{W}_t using gradient descent, yielding \widehat{W}_{t+1} , as described in Algorithm 4.2. To do this, a convex feasible set of influence matrices, denoted \mathcal{W} , must be defined. For instance, we might consider families of sparse W

$$\mathcal{W} = \{W \in \mathbb{R}_+^{p \times p} : \|W\|_1 \leq c\},$$

or even W with partially known support (*i.e.*, prior knowledge of a subset of the elements of W that are zero-valued). First, the prediction $\hat{\lambda}_{t+1}$ is updated using the

Algorithm 4.2 MV Hawkes Tracking - W Unknown

- 1: Initialize $\widehat{W}_1 = W_0$, $K_1 = \mathbf{0}$, $\widehat{\lambda}_1 = \bar{\mu}$
 - 2: **for** $t = 1, \dots, T/\delta$ **do**
 - 3: Observe x_t and incur loss $\tilde{\ell}_t(\widehat{\lambda}_t) = \langle \mathbf{1}, \delta \widehat{\lambda}_t \rangle - \langle x_t, \log \delta \widehat{\lambda}_t \rangle$
 - 4: Set $\tilde{\lambda}_{t+1} = (1 - \eta_t) \widehat{\lambda}_t + \eta_t x_t / \delta$
 - 5: Define $y_t \triangleq \sum_{\tau_n = \delta t} e_{k_n} h(\delta(t+1) - \tau_n)$
 - 6: Set $g_t(W) = \tilde{\ell}_t(\widehat{\lambda}_t^{(W)}) = \langle \mathbf{1}, \delta \widehat{\lambda}_t^{(W)} \rangle - \langle x_t, \log \delta \widehat{\lambda}_t^{(W)} \rangle$
 - 7: Set $\nabla g_t(W) = \delta \mathbf{1} K_t^\top - \text{Diag}(\widehat{\lambda}_t^{\widehat{W}_t})^{-1} x_t K_t^\top$
 - 8: Set $\widehat{W}_{t+1} = \text{proj}_{\mathcal{W}} \left(\widehat{W}_t - \rho_t \nabla g_t(\widehat{W}_t) \right)$
 - 9: Set $K_{t+1} = (1 - \eta_t) \alpha^\delta K_t + y_t$
 - 10: Set $\widehat{\lambda}_{t+1} = \tilde{\Phi}_t(\tilde{\lambda}_{t+1}, \widehat{W}_t) + (\widehat{W}_{t+1} - \widehat{W}_t) K_{t+1}$
 - 11: **end for**
-

previous estimate of the network, \widehat{W}_t . Then the estimate of W is updated, and the transformation described in Lemma 4.5 is applied.

The next result establishes tracking regret bounds for Algorithm 4.2:

Theorem 4.6 (Tracking regret of Algorithm 4.2). *Let $\tilde{\Phi}_t(\lambda, W) = \alpha^\delta \lambda + W y_t + (1 - \alpha^\delta) \bar{\mu}$ with $0 < \alpha < 1$ for all W and $t = 1, 2, \dots, T/\delta$. Additionally, let the sequence $\widehat{\lambda}_1, \widehat{\lambda}_2, \dots, \widehat{\lambda}_T$ be the output of Algorithm 4.2, and let $\lambda_1, \lambda_2, \dots, \lambda_T$ be an arbitrary sequence in $[\lambda_{\min}, \lambda_{\max}]^p$. If we set η_t and ρ_t both proportional to either $1/\sqrt{t}$ or $1/\sqrt{T/\delta}$, then for any data sequence $x_1, \dots, x_{T/\delta}$ in $[0, \delta x_{\max}]^p$ with $x_{\max} < \lambda_{\max}$, there exists a constant $C > 0$ which depends on $\delta, p, x_{\max}, \lambda_{\max}$ and λ_{\min} such that*

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\widehat{\lambda}_t) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t) \leq C \left(1 + \min_{W \in \mathcal{W}} \sum_{t=1}^{T/\delta} \|\lambda_{t+1} - \tilde{\Phi}_t(\lambda_t, W)\|_2 \right) \sqrt{T}$$

This theorem is proved in Appendix 4.9.6. This bound says that running Algorithm 4.2 achieves an average per-round loss which is nearly as low as what would have been achieved with access to all data to choose the optimal time-varying rate vectors with a batch method. The gap between the losses of the proposed method and the losses accrued with a batch method scale with how closely the batch output

(*i.e.*, comparator sequence) follows the dynamical model associated with the *best* estimate of the social network structure as encapsulated by W . For instance, imagine that there existed a true, fixed W representing a social network, and an oracle used this W to estimate a sequence of rate vectors which followed the model in (4.11a) exactly and, subject to that constraint, minimized the sum of losses. For that oracle, the variation $\sum_{t=1}^{T-1} \|\lambda_{t+1} - \tilde{\Phi}_t(\lambda_t, W)\|_2 = 0$. Clearly such an estimator is not practical because we do not know W and are operating in an online setting. Despite these disadvantages, the difference of the average per-round losses of Algorithm 4.2 and the oracle estimator scales like $1/\sqrt{T}$, so that as $T \rightarrow \infty$, the performance gap between the two methods vanishes.

These bounds do not rely on any assumptions about the data actually being generated by a multivariate Hawkes process or even being stochastic (which would be a fallacy in any real-world application). Rather, the ideas underlying the multivariate Hawkes model are used to generate a loss function and dynamical model. These values are used to characterize how well our methods perform tracking in an online setting relative to how well any other method might perform *on the same set of observations*. Further, the comparator sequence against which performance is measured might be computed in batch (rather than online) or using significantly more computational and memory resources than are required by Algorithm 4.2.

The intuition behind why this method is robust to model mismatch can be seen in the algorithm itself and in the regret bound. Notice in line 4 of the algorithm, we are directly adjusting our estimate of the rate, prior to adjusting the network weights. Because we are adjusting not only our estimate of the network weights and directly calculating the resulting rate, our sequence of estimates is allowed to deviate from a pure Hawkes process. This amount of deviation can allow us to be more flexible to combat the errors due to model mismatch. Additionally, the form of the regret bound tells us that the method will perform competitively against any set of

comparators that nearly follows the dynamical model. Therefore if the generative model is similar, although not exactly a Hawkes process, then this variation term will still be low, and result in low overall loss.

4.6 Computational Complexity

One important feature of the proposed method is the low computational cost per iteration. Algorithm 4.2 performs the tasks of estimating both the current intensity $\hat{\lambda}_t \in \mathbb{R}^p$ and the network relationships $\widehat{W}_t \in \mathbb{R}_+^{p \times p}$. A brief examination of lines 3 - 6 of the algorithm shows mostly vector operations on length p vectors, requiring $O(p)$ operations. The main computational burden of the algorithm comes in line 7, with the matrix multiplications requiring $O(p^2)$ operations and in line 8 projecting on to the space \mathcal{W} . Without the projection step, this leaves the algorithm at an overall complexity of $O(p^2)$ to estimate $p^2 + p$ values. Depending on the space \mathcal{W} , the algorithm may be slower. For instance, a reasonable space would be the space of matrices with a bounded nuclear norm, which requires computing a singular value decomposition at each step requiring $O(p^3)$ operations. Projecting onto other spaces, such as an ℓ_1 ball with some radius, would only require $O(p^2)$ operations, maintaining our baseline complexity.

4.7 Experimental Results

In this section we present several experiments to demonstrate salient features of the proposed algorithms. We first focus on the scenario where the network influence matrix, W , is known. In this scenario the important observation is that our method is more robust to model mismatch than just calculating the rate directly from the observations, assumed influence function and matrix W . The next set of experiments demonstrates Algorithm 4.2 for unknown W on synthetic data and demonstrates how it can be used to learn both rates and the network of interest. Finally, we use

Algorithm 4.2 to analyze six months of Memetracker data corresponding to posts by a selection of well known news websites to try to determine what relationships exist amongst these organizations.

Throughout this section, we compare our method to the classical online learning algorithm Online Gradient Descent (OGD) used to learn the network W . This method is described formally in Appendix 4.9.7. One alternative algorithm to learn the network would be to simply count all the times one process has an event immediately after another process had an event, with larger counts corresponding to larger influence. OGD uses a current estimate of the network, and then uses the loss function and the assumed influence function, to update the network estimate in the direction of the most recent data point. Therefore the estimate is a weighted average of previously seen data, with more weight put on more recent data. In this way OGD is basically the same as the counting process but with more information put into the system. We compare against this method and show in several ways that our method performs comparatively when both methods know the correct influence function, but our method performs better when model information is misspecified.

One challenge of online methods is the tuning of the step-size parameters, in our case η and ρ . In all of our experiments, we measure the effectiveness of the step-size based solely on accumulated loss on a small subset of the data. For instance, given a new dataset, we could run the method several times on the first 5% of the data with a range of step-size parameters, observe the total accumulated loss, and choose the parameters which minimize the loss over that time empirically. The basic setup of online learning protects us from over-fitting because setting step-sizes too large would push our estimates very close to the immediately preceding observation, which would cause large loss on the next observation due to over-fitting. Conversely, very small step-sizes would not adapt or learn the parameters at all, also causing high accumulated loss.

Throughout this section we plot several curves of interest to demonstrate the efficacy of our methods. The first is the average per round loss of estimates $\lambda_1, \lambda_2, \dots, \lambda_t$ at time t defined as $\frac{1}{t} \sum_{\tau=1}^t \tilde{\ell}_\tau(\lambda_\tau)$. The next metric is excess cumulative loss with respect to a comparator λ_t^* as $\sum_{\tau=1}^t \tilde{\ell}_\tau(\lambda_\tau) - \sum_{\tau=1}^t \tilde{\ell}_\tau(\lambda_\tau^*)$, where the comparator will usually be the true underlying rate if known, or the rate calculated from Algorithm 4.1 with the true W matrix. Finally, we plot a moving average curve of excess instantaneous loss, defined at time t as $\frac{\delta}{D} \sum_{i=0}^{D/\delta-1} \left(\tilde{\ell}_{t-i}(\lambda_{t-i}) - \tilde{\ell}_{t-i}(\lambda_{t-i}^*) \right)$, for a time window of width D .

4.7.1 Model mismatch, W known

For our first experiment, 1000 data points were generated in a two-actor network ($p = 2$), with W an identity matrix scaled by $3/4$, the influence function $h(t) = e^{-t} \mathbf{1}_{[t>0]}$ and $\bar{\mu}$ randomly generated uniformly on $[0, .01]^2$, using the method of [105]. The data was then processed in several ways. The first was to calculate the discrete time rates using an incorrect influence function, $\tilde{h}(t) = (2e)^{-t} \mathbf{1}_{[t>0]}$, without doing any learning. In other words, a rate is estimated by plugging observation event times into Equation 4.4 and using W , $\tilde{h}(\cdot)$, and $\bar{\mu}$. We will call this method direct calculation. However, we expect suboptimal performance due to the fact that $\tilde{h}(t) \neq h(t)$. This method is compared to the output of Algorithm 4.1 with the same, incorrect $\tilde{h}(t)$ function, with $\delta = 0.1$ and $\eta_t = 5/\sqrt{T/\delta}$, to show that robustness to model mismatch has been added. These results are shown in Figure 4.1. The first plot is compared to the losses of the continuous and discrete rates calculated with the true generating parameters, as described in Equations 4.2 and 4.4 respectively.

Another experiment was run on the same data, generated using $h(t) = e^{-t} \mathbf{1}_{[t>0]}$, and the same true W matrix, but this time the influence function used to estimate the rates was $\tilde{h}(t) = \mathbf{1}_{[0<t<5]}$ and all other parameters are kept the same. These results are shown in Figure 4.2. Again, learning the rates instead of performing

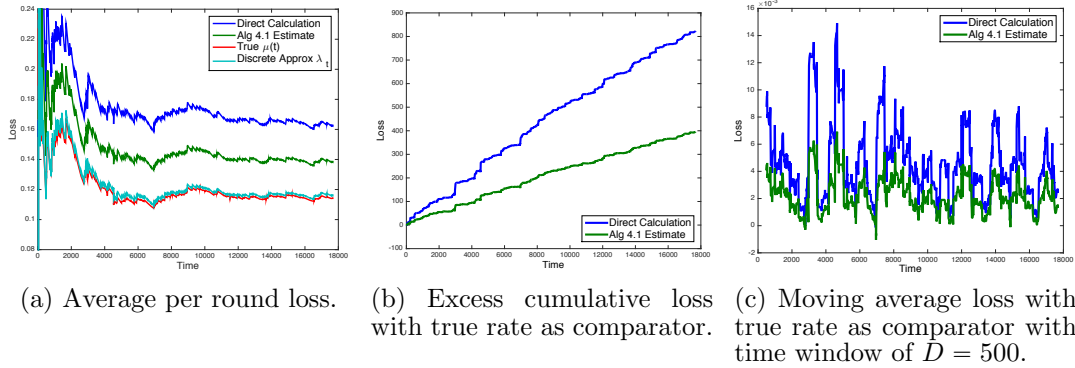


FIGURE 4.1: Performance of Algorithm 4.1 using an incorrect exponential decay influence function. Tracking the rate instead of just calculating it from known network values and the assumed influence function leads to better overall performance.

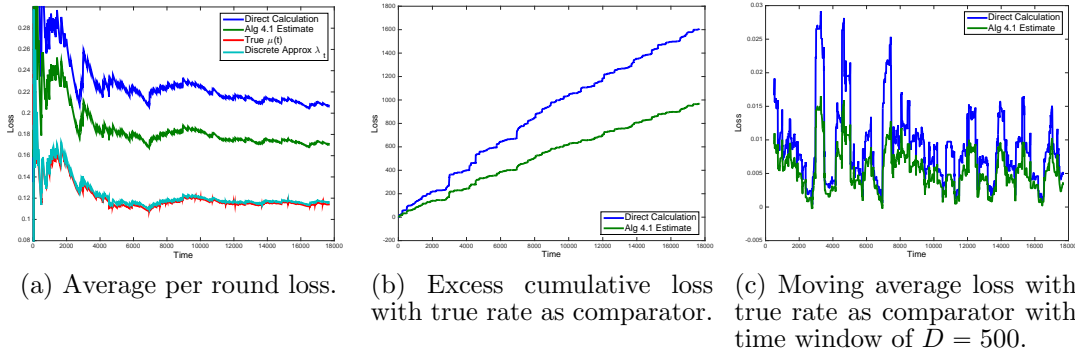


FIGURE 4.2: Performance of Algorithm 4.1 using an incorrect rect influence function. Again, tracking the rate has added robustness to misspecified system parameters.

direct calculations using an incorrect influence function, has added more robustness to model mismatch.

4.7.2 Learning W

In the next set of experiments, the ability of Algorithm 4.2 to learn the network structure and rates from just event timing data is tested. A network of 100 actors ($p = 100$) is used with a true underlying network generated with a small number of actors each with extensive influence. This network, shown in Figure 4.3, was generated by randomly selecting a few actors to be influential, then giving them

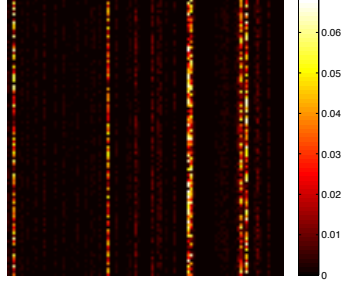


FIGURE 4.3: True network used to generate event times. Each pixel represents the influence the actor represented by the column has on the actor represented by the row, with lighter colors meaning more influence, and black meaning no influence.

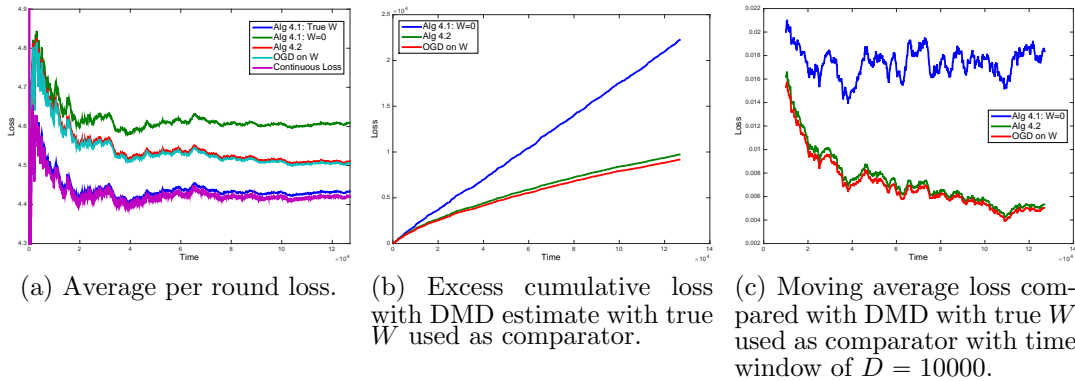


FIGURE 4.4: Performance of Algorithm 4.1 with different values of W , compared to Algorithm 4.2. Both our method and Online Gradient Descent (OGD) learn the structure of the true network and has performance that approaches Algorithm 4.1 with the true value of W .

a random, high amount of influence, and giving everyone else a random, but low amount of influence. The smallest edge weights of the network were set to 0, such that about 50% of the the edge weights were non-zero. Finally this matrix was normalized such that it had a maximum singular value of 0.8 for stability.

Similarly to the previous section, the value of $\bar{\mu}$ was uniformly generated on $[0.001, .01]^p$, and the influence function used was $h(t) = e^{-t}\mathbf{1}_{[t>0]}$ and $\delta = .01$. Using these parameters, 100,000 event times were generated. Algorithm 4.2 was then run with $\eta_t = 7/\sqrt{T/\delta}$ and $\rho = .0055/\sqrt{T/\delta}$. Figure 4.4 shows the results for Algorithm 4.1 where the value of W used was the generating value, and where W was all 0s.

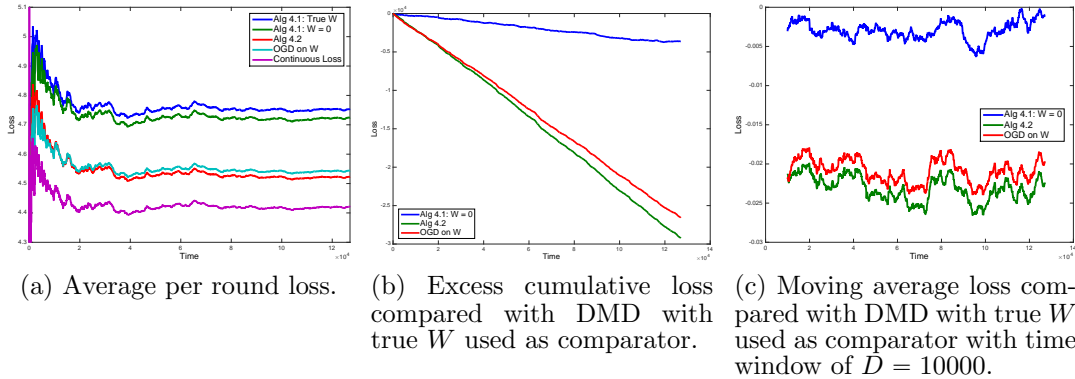


FIGURE 4.5: Performance of Algorithm 4.1 with different values of W , compared to Algorithm 4.2. When the system parameters are misspecified, our method outperforms OGD on W in predicting likelihood of actors participating in events.

We compare these two to the result of Algorithm 4.2 and estimating W using OGD with step size ρ_t .

The important feature of Figure 4.4 is that the results of Algorithm 4.2 start poorly when the estimate of W is bad, but as more and more data are revealed, the loss approaches the loss of the algorithm with full knowledge of the true matrix W as predicted by Theorem 4.6. Additionally, the performance of Algorithm 4.2 very closely mirrors the performance of using OGD to estimate W directly and using that to get an estimate of the instantaneous rate using Equation 4.4. This shows again that in the case where the influence function is known precisely, little is lost by tracking both rates and the network.

Using the same set of event times, another set of trials was run, this time using a mismatched influence function $\tilde{h}(t) = .9^t \mathbf{1}_{[t>0]}$, and otherwise all the same parameters. The results of these trials are shown in Figure 4.5. In these results, the performance is not as accurate as when the ground truth influence function was known, but Algorithm 4.2 steadily outperforms directly estimating W using OGD, again demonstrating that our method has added robustness to poorly specified parameters.

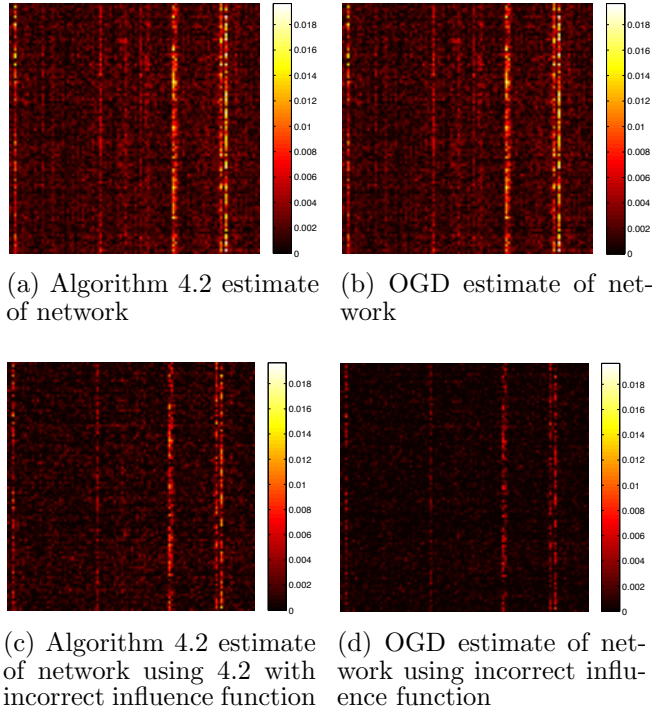
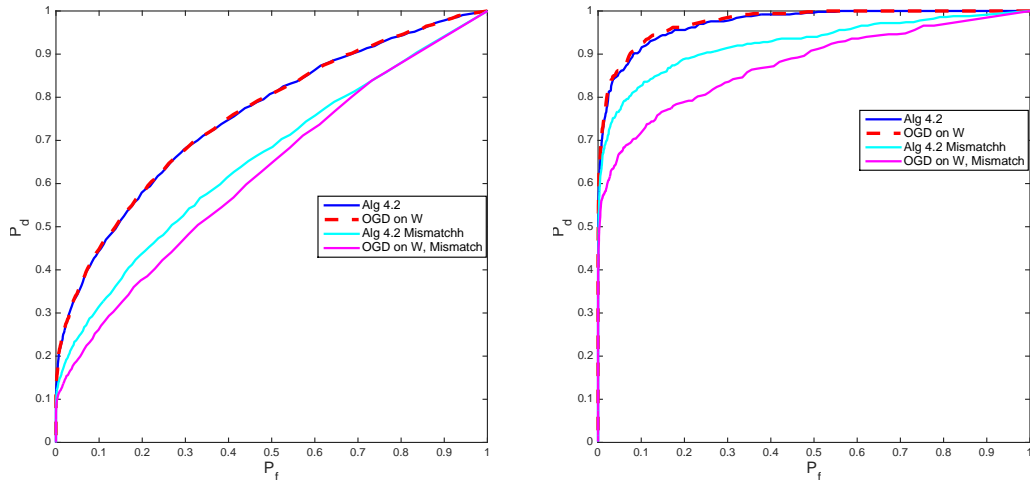


FIGURE 4.6: Estimates of the underlying network using Algorithm 4.2 and OGD with both correct (a,b) and incorrect (c,d) influence functions. Our method captures more of the network structure when influence function is misspecified.

Additionally, how well the networks have been estimated can be examined, both with the correct and incorrect influence function. The final estimates of the networks are shown in Figure 4.6. The estimates recover the vertical striping characteristic of the true network. When the influence function is known correctly these structures are more pronounced, and the networks produced by Algorithm 4.2 and OGD on W are very similar. However, when the influence function is misspecified, our method still recovers the vertical bands whereas directly performing OGD on W has failed to recover many of these bands.

We also observe how well the significant elements of the network are recovered by setting various thresholds and declaring all elements of the estimate above this threshold as significant relationships in the network. These relationships are then compared with the true, non-zero elements of the network to generate ROC curves



(a) ROC Curve for Full Support

(b) ROC Curve for Largest 500 Elements

FIGURE 4.7: ROC Curves to demonstrate the method’s abilities to find the significant relationships in the network. Again, Algorithm 4.2 and OGD on W perform very similarly when the h function is known, but Algorithm 4.2 does better when it is misspecified

for each method. This curve is computed both for the full W matrix and just the largest 500 elements of W as baselines. We choose to also focus on these largest edge weights as they represent the majority of the total weight in the network. These ROCs are shown in Figure 4.7, which show our method’s increased ability to find the important relations in the network compared to OGD.

As a final test of how well we are learning the matrix W , instantaneous estimates, \widehat{W}_t , are used to compute the total loss of the entire data using this matrix as in Equation 4.5. As more data are revealed, each estimate is produced with an increasing amount of training data and then tested on the full data set. Each estimate approaches the cumulative loss of directly calculating the rates with the true matrix W . These batch losses were all calculated using the true influence function, and are shown in Figure 4.8. Our online method is decreasing the overall loss and approaches the same performance of knowing the true network. Additionally, the estimation with the generative influence function is very similar for Algorithm 4.2

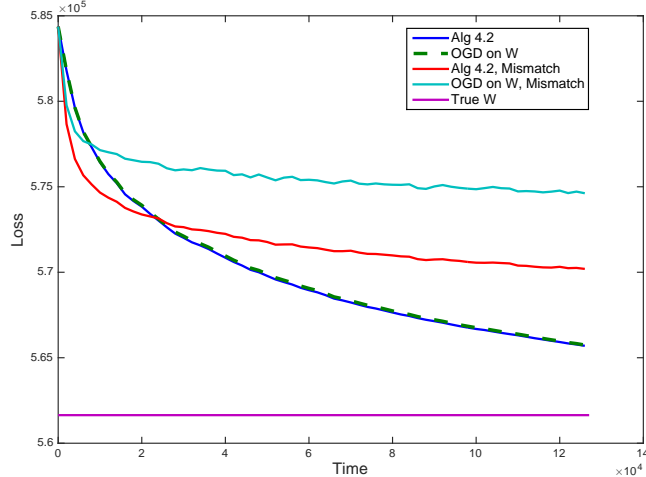


FIGURE 4.8: Batch loss of network estimates. As more data are revealed each method’s performance improves. When the influence function is known precisely our method and OGD on W both perform well. When the influence function is misspecified our method outperforms OGD.

and OGD on W , but our method performs better than OGD when the influence function is misspecified.

4.7.3 Memetracker

For the final experiment, we used the raw phrases Memetracker [107] data set (<http://www.memetracker.org/data.html>) and extracted every post from websites analyzed by the authors as reporting a high percentage of important news (<http://www.memetracker.org/lag.html>). These 217 distinct websites made up our network of interest. We extracted the posts from these websites for a six month span from August 2008 through the end of January 2009, totaling over 3.5 million events. The only information considered was which made a post and at what times, without using information about content or links. The event times were on the resolution of 1 second intervals, and thus we chose $\delta = 1$ sec.

Algorithm 4.2 was run on this data to learn influences within the network and validated with the delays discovered in the lag time of stories being reported. We used the data from the first half of the first month as a test set to tune parameters and

found that $\alpha = .995$, $\eta = 6.1 \times 10^{-4}$, $\rho = 6.1 \times 10^{-10}$ and $\bar{\mu} = 2 \times 10^{-5}$ accumulated relatively low loss. Using $\alpha = .995$, which corresponds to a reaction's "half-life" being about 2 minutes and 20 seconds, is long enough time window for meaningful reactions to take place, but not long enough for many significantly different topics to be published. It is worth noting this performance focuses on immediate dependencies, and thus we work on the scale of minutes, whereas the average lag times are reported on the scale of hours, and thus we are more likely to discover which organizations publish content faster, rather than finding websites which are likely using others as references.

The websites were ordered based on their average lag time described on the Memetracker results, from smallest to largest. Therefore, we expected many of the significant relationships to be beneath the diagonal. Recall $W_{i,j}$ reflects influence of actor j on actor i . Because the actors are ordered from smallest to largest lag time, we expect larger elements $W_{i,j}$ to have $i > j$, which corresponds to more significant relationships being beneath the diagonal. Relationships are declared "significant" when $W_{i,j}$ is above a threshold. Figure 4.9 shows the number of significant relationships across a range of thresholds. For most choices of threshold, more significant relationships are below the diagonal.

We found that among the 20 most influential websites were dailyherald.com, washingtonpost.com, post-gazette.com, denverpost.com, news.bbc.co.uk, and cnn.com. All of these are either local news organizations in major metropolitan areas or important national news organizations. Among the top dependencies were apnews.myway.com reacting to dailyherald.com, elections.foxnews.com reacting to washingtonpost.com and mclatchydc.com reacting to washingtonpost.com. The first two pairs are examples of large national organizations being slower to respond than local sources. Alternatively, the third pair is an example of two local news sources, where the organization with more journalists is able to publish faster than

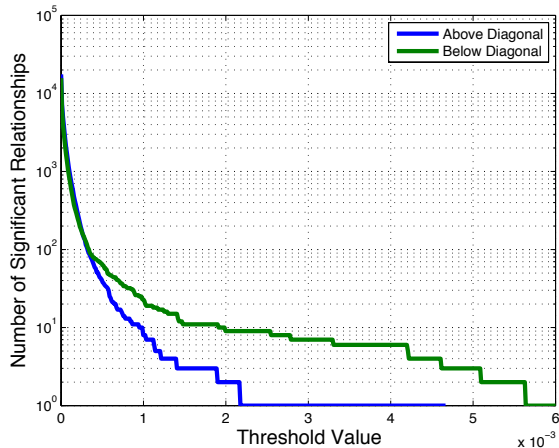


FIGURE 4.9: Amount of relationships above a threshold for the network learned using the Memetracker dataset. For most choices of the threshold, there are more relationships that are greater than the threshold above the diagonal than below.

a competitor with less journalists.

4.8 Conclusion

In many real world applications, such as social, neural and financial networks, actions by actors at one point in time will influence the future actions of others in the network. In these applications it is beneficial to estimate the likelihood of each actor acting at any given point in time, given only timing information about previous occurrences. This task is particularly challenging when data are streaming at a rate that precludes traditional batch processing methods. We have proposed two online methods for tracking the time varying rates at which actors participate in events; one method for when the underlying network is known and the other for when it is unknown. Relevant regret bounds for both methods scale with the deviation of a comparator series of point process rate or intensity functions, with no assumptions on the actual generative model of the data. These methods were tested using both synthetic and real data to show that they successfully track the intensities of interest, can recover the underlying network structure, and are robust to model mismatch.

4.9 Appendix of Chapter 4

4.9.1 Lemma 4.7

The proof of Lemma 4.2, which compares the loss of the continuous time and discrete time autoregressive rates, relies on the following relationship.

Lemma 4.7. *For $0 < \alpha < 1$ and $\delta > 0$ the following inequalities hold:*

$$\frac{-1}{\log(\alpha)} - \frac{\delta}{2} \leq \frac{\delta\alpha^\delta}{1 - \alpha^\delta} \leq \frac{-1}{\log(\alpha)}. \quad (4.13)$$

Proof: The proof starts with the following observations:

$$\begin{aligned} \lim_{\delta \rightarrow 0} \frac{\delta\alpha^\delta}{1 - \alpha^\delta} &= \frac{-1}{\log(\alpha)} \\ \frac{\partial}{\partial \delta} \frac{\delta\alpha^\delta}{1 - \alpha^\delta} &= \frac{\alpha^\delta - \alpha^{2\delta} + \delta\alpha^\delta \log(\alpha)}{(1 - \alpha^\delta)^2}. \end{aligned}$$

Because all three terms in Equation 4.13 are equal when $\delta = 0$, showing that the derivative of $\frac{\delta\alpha^\delta}{1 - \alpha^\delta}$ with respect to δ is between $-\frac{1}{2}$ and 0 suffices to prove the Lemma.

The upper bound stems from the following inequality:

$$\begin{aligned} 1 + \delta \log(\alpha) &\leq \alpha^\delta \\ \implies 1 - \alpha^\delta + \delta \log(\alpha) &\leq 0 \\ \implies \frac{\partial}{\partial \delta} \frac{\delta\alpha^\delta}{1 - \alpha^\delta} &= \frac{\alpha^\delta - \alpha^{2\delta} + \delta\alpha^\delta \log(\alpha)}{(1 - \alpha^\delta)^2} \leq 0. \end{aligned}$$

The proof of the lower bound requires the analysis of another function, $\alpha^\delta - \alpha^{-\delta}$.

$$\begin{aligned} \frac{\partial}{\partial \delta} (\alpha^\delta - \alpha^{-\delta}) &= \log(\alpha)(\alpha^\delta + \alpha^{-\delta}) \\ \frac{\partial^2}{\partial \delta^2} (\alpha^\delta - \alpha^{-\delta}) &= \log^2(\alpha)(\alpha^\delta - \alpha^{-\delta}) \leq 0 \text{ for } \delta > 0 \end{aligned}$$

Because this function is concave for $\delta > 0$, the following inequality holds:

$$\alpha^\delta - \alpha^{-\delta} \leq 2\delta \log(\alpha)$$

which simply says that for $\delta > 0$ the function lies below its tangent line at $\delta = 0$. This inequality can be used to derive the desired lower bound.

$$\begin{aligned}
& \alpha^\delta - \alpha^{-\delta} \leq 2\delta \log(\alpha) \\
\implies & \frac{\alpha^{2\delta} - 1}{2} \leq \delta \alpha^\delta \log(\alpha) \\
\implies & \frac{2\alpha^\delta - \alpha^{2\delta} - 1}{2} \leq \alpha^\delta - \alpha^{2\delta} + \delta \alpha^\delta \log(\alpha) \\
\implies & -\frac{1}{2} \leq \frac{\alpha^\delta - \alpha^{2\delta} + \delta \alpha^\delta \log(\alpha)}{(1 - \alpha^\delta)^2} = \frac{\partial}{\partial \delta} \frac{\delta \alpha^\delta}{1 - \alpha^\delta}
\end{aligned}$$

Therefore, the derivative of $\frac{\delta \alpha^\delta}{1 - \alpha^\delta}$ is between $-\frac{1}{2}$ and 0 for $\delta > 0$ and $0 < \alpha < 1$, which combined with the limit statement, proves the Lemma.

4.9.2 Proof of Lemma 4.2

This proof shows that the negative log likelihood of the true underlying Hawkes process (Equation 4.1) given an excitation matrix W , differs by a factor proportional to δ from the discretized version (Equation 4.5).

$$\begin{aligned}
& \left| \sum_{k=1}^p \int_0^T \mu_k(\tau) d\tau - \sum_{n=1}^{N_T} \log \mu_{k_n}(\tau_n) - \sum_{k=1}^p \left(\sum_{t=1}^{T/\delta} \delta \lambda_{t,k} - x_{t,k} \log \lambda_{t,k} \right) \right| \\
& \leq \left| \sum_{k=1}^p \left(\int_0^T \mu_k(\tau) d\tau - \sum_{t=1}^{T/\delta} \delta \lambda_{t,k} \right) \right| + \left| \sum_{k=1}^p \sum_{t=1}^{T/\delta} x_{t,k} \log \lambda_{t,k} - \sum_{n=1}^{N_t} \log \mu_{k_n}(\tau_n) \right|
\end{aligned}$$

The two absolute value terms will be bounded separately. We start with the first term, involving the integral of the true rate, and the approximation to it.

$$\begin{aligned}
& \left| \sum_{k=1}^p \left(\int_0^T \mu_k(\tau) d\tau - \sum_{t=1}^{T/\delta} \delta \lambda_{t,k} \right) \right| \\
& \leq \sum_{k=1}^p \left| \int_0^T \sum_{\tau_n < \tau} W_{k,k_n} \alpha^{\tau - \tau_n} d\tau - \delta \sum_{t=1}^{T/\delta} \sum_{\bar{\tau}_n < \delta t} W_{k,k_n} \alpha^{\delta t - \tau_n} \right| \\
& = \sum_{k=1}^p \left| \sum_{n=1}^{N_T} W_{k,k_n} \int_{\tau_n}^T \alpha^{\tau - \tau_n} d\tau - W_{k,k_n} \delta \sum_{t=\bar{\tau}_n/\delta + 1}^{T/\delta} \alpha^{\delta t - \tau_n} \right| \\
& = \sum_{k=1}^p \left| \sum_{n=1}^{N_T} W_{k,k_n} \left(\frac{\alpha^{T - \tau_n} - 1}{\log(\alpha)} - \delta \alpha^\delta \frac{\alpha^{\bar{\tau}_n - \tau_n} - \alpha^{T - \tau_n}}{1 - \alpha^\delta} \right) \right| \\
& = \sum_{k=1}^p \left| \sum_{n=1}^{N_T} W_{k,k_n} \left(\alpha^{T - \tau_n} \left(\frac{\delta \alpha^\delta}{1 - \alpha^\delta} - \frac{-1}{\log(\alpha)} \right) + \frac{-1}{\log(\alpha)} - \alpha^{\bar{\tau}_n - \tau_n} \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \right) \right|
\end{aligned}$$

These lines evaluate the value of the integral and the approximation to them using a geometric sum to get a closed form for their difference. From this point, the use of Lemma 4.7 allows us to find an upper bound

$$\begin{aligned}
& \alpha^{T - \tau_n} \left(\frac{\delta \alpha^\delta}{1 - \alpha^\delta} - \frac{-1}{\log(\alpha)} \right) + \frac{-1}{\log(\alpha)} - \alpha^{\bar{\tau}_n - \tau_n} \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \\
& \leq \frac{-1}{\log(\alpha)} - \alpha^\delta \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \leq \frac{\delta}{2} + (1 - \alpha^\delta) \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \leq \frac{3\delta}{2}.
\end{aligned}$$

We can similarly use Lemma 4.7 to find the lower bound.

$$\begin{aligned}
& \alpha^{T - \tau_n} \left(\frac{\delta \alpha^\delta}{1 - \alpha^\delta} - \frac{-1}{\log(\alpha)} \right) + \frac{-1}{\log(\alpha)} - \alpha^{\bar{\tau}_n - \tau_n} \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \\
& \geq \frac{-\delta}{2} \alpha^{T - \tau_n} + \frac{-1}{\log(\alpha)} - \frac{\delta \alpha^\delta}{1 - \alpha^\delta} \geq -\frac{\delta}{2}
\end{aligned}$$

Combining these upper and lower bounds gives an overall bound on the integral approximation:

$$\left| \sum_{k=1}^p \left(\int_0^T \mu_k(\tau) d\tau - \sum_{t=1}^{T/\delta} \delta \lambda_{t,k} \right) \right| \leq p N_T W_{\max} \frac{3\delta}{2} \quad (4.14)$$

This shows that the integral terms deviate by no more than a linear factor of δ and can be controlled by setting δ small.

Next the difference involving the log terms must be bounded. We make the following observation:

$$\sum_{k=1}^p \sum_{t=1}^{T/\delta} x_{t,k} \log \lambda_{t,k} = \sum_{n=1}^{N_T} \log \lambda_{\bar{\tau}_n/\delta, k_n}$$

This says that instead of going from time step to time step and incurring loss at every point, we just step through every event and incur loss at the following discrete time point $\bar{\tau}_n$. Now the log terms can be compared as

$$\sum_{k=1}^p \sum_{t=1}^{T/\delta} x_{t,k} \log \lambda_{t,k} - \sum_{n=1}^{N_T} \log \mu_{k_n}(\tau_n) = \sum_{n=1}^{N_T} \log \left(\frac{\lambda_{\bar{\tau}_n/\delta, k_n}}{\mu_{k_n}(\tau_n)} \right).$$

In order to bound this, we make the distinction between two classes of events. The first set of events, $\mathcal{A}_n = \{i | \tau_i < \tau_n, \bar{\tau}_i < \bar{\tau}_n\}$, are events that happen before the n^{th} event and are not in the same time window. The second set, $\mathcal{B}_n = \{i | \tau_i < \tau_n, \bar{\tau}_i = \bar{\tau}_n\}$, are events that happen before the n^{th} event but in the same time window.

$$\sum_{n=1}^{N_T} \log \left(\frac{\lambda_{\bar{\tau}_n/\delta, k_n}}{\mu_{k_n}(\tau_n)} \right) = \sum_{n=1}^{N_T} \log \left(\frac{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\bar{\tau}_n - \tau_i}}{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i} + \sum_{i \in \mathcal{B}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i}} \right)$$

This term needs to be upper and lower bounded to get the final result. Because every element of $W \geq 0$ and $\alpha > 0$, all the terms are positive so the terms in set \mathcal{B}_n can be dropped, to get the following upper bound:

$$\sum_{n=1}^{N_T} \log \left(\frac{\lambda_{\bar{\tau}_n/\delta, k_n}}{\mu_{k_n}(\tau_n)} \right) \leq \sum_{n=1}^{N_T} \log \left(\frac{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\bar{\tau}_n - \tau_i}}{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i}} \right) \quad (4.15a)$$

$$\leq \sum_{n=1}^{N_k} \log \left(\frac{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i}}{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i}} \right) = 0. \quad (4.15b)$$

In Equation 4.15a positive terms have been removed from the denominator and in 4.15b the fact that $\bar{\tau}_n \geq \tau_n$ and therefore $\alpha^{\bar{\tau}_n} \leq \alpha^{\tau_n}$. Next, we lower bound T_1 .

$$\sum_{n=1}^{N_T} \log \left(\frac{\lambda_{\bar{\tau}_n/\delta, k_n}}{\mu_{k_n}(\tau_n)} \right) \geq \sum_{n=1}^{N_T} \log \left(\frac{\bar{\mu}_{k_n} + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\bar{\tau}_n - \tau_i}}{\bar{\mu}_k + \sum_{i \in \mathcal{A}_n} W_{k_n, k_i} \alpha^{\bar{\tau}_n - \tau_i - \delta} + \sum_{i \in \mathcal{B}_n} W_{k_n, k_i} \alpha^{\tau_n - \tau_i}} \right) \quad (4.16a)$$

$$\geq - \sum_{n=1}^{N_T} \log \left(\alpha^{-\delta} + \frac{|\mathcal{B}_n| W_{\max}}{\mu_{\min}} \right) \quad (4.16b)$$

$$\geq - \sum_{n=1}^{N_T} \log \left(\alpha^{-\delta} + \frac{p x_{\max} \delta W_{\max}}{\mu_{\min}} \right) \quad (4.16c)$$

$$\geq - N_T \left(\frac{W_{\max} x_{\max} p}{\mu_{\min}} - \log(\alpha) \right) \delta \quad (4.16d)$$

Equation 4.16a comes from $\alpha^{\bar{\tau}_n - \delta} \geq \alpha^{\tau_n}$, Equation 4.16b uses $\alpha^{-\delta} > 1$ and cancels out like terms from the numerator and denominator, Equation 4.16c bounds the number of events in \mathcal{B}_n by $p x_{\max} \delta$ which is the maximum number of events each actor can participate in a δ length time window, times the number of actors, and finally Equation 4.16d uses $\log(x + a) \leq \frac{x}{a} + \log(a)$ for $a > 0$ which is a consequence of the concavity of the log function. The upper and lower bound gives

$$\left| \sum_{n=1}^{N_T} \log \left(\frac{\lambda_{\bar{\tau}_n/\delta, k_n}}{\mu_{k_n}(\tau_n)} \right) \right| \leq N_T \left(\frac{W_{\max} x_{\max} p}{\mu_{\min}} - \log(\alpha) \right) \delta$$

which when combined with Equation 4.14 gives the result:

$$|L_T(\mu) - L_T^{(\delta)}(\lambda)| \leq \left(\frac{3pW_{\max}}{2} + \frac{W_{\max} x_{\max} p}{\mu_{\min}} - \log(\alpha) \right) N_T \delta.$$

4.9.3 Proof of Lemma 4.3

We start the proof with the following important observation about the Bregman divergence in question:

$$\begin{aligned}
 D(\theta_1 \parallel \theta_2) &= \langle \exp(\theta_1) - \exp(\theta_2), \mathbf{1} \rangle - \langle \exp(\theta_2), \theta_1 - \theta_2 \rangle \\
 &= \sum_{k=1}^p \exp(\theta_{1,k}) - \exp(\theta_{2,k}) - \exp(\theta_{2,k})(\theta_{1,k} - \theta_{2,k}) \\
 &= \sum_{k=1}^p d(\theta_{1,k} \parallel \theta_{2,k}).
 \end{aligned}$$

Above, $\theta_{1,k}$ and $\theta_{2,k}$ denote the k^{th} element of vectors θ_1 and θ_2 respectively, and d is the scalar Bregman divergence induced by $\exp(\theta)$. This shows that the p -dimensional Bregman divergence can be broken into the sum of p terms. We will prove bounds for the one dimensional version and then combine them to show that overall the dynamics are contractive. We start by showing the result for when the k^{th} diagonal element of the matrix A_t , denoted by $A_{t,k}$ is 0. We denote the k^{th} element of the vector after the application of the dynamics as $[\Phi_t(\theta)]_k$ and the k^{th} element of b_t as $b_{t,k}$.

$$\begin{aligned}
 d([\Phi_t(\theta_1)]_k \parallel [\Phi_t(\theta_2)]_k) &= A_{t,k}(\exp(\theta_{1,k}) - \exp(\theta_{2,k})) \\
 &\quad - (A_{t,k} \exp(\theta_{2,k}) + b_{t,k}) \log \left(\frac{A_{t,k} \exp(\theta_{1,k}) + b_{t,k}}{A_{t,k} \exp(\theta_{2,k}) + b_{t,k}} \right) \\
 &= -b_{t,k} \log \left(\frac{b_{t,k}}{b_{t,k}} \right) = 0 = A_{t,k} d(\theta_{1,k} \parallel \theta_{2,k})
 \end{aligned}$$

When $A_{t,k} = b_{t,k} = 0$ we define $\log\left(\frac{0}{0}\right) \triangleq 0$. Next, we show the result for the case when $A_{t,k} > 0$ and $b_{t,k} = 0$:

$$\begin{aligned}
d([\Phi_t(\theta_1)]_k \| [\Phi_t(\theta_2)]_k) &= A_{t,k} (\exp(\theta_{1,k}) - \exp(\theta_{2,k})) \\
&\quad - A_{t,k} \exp(\theta_{2,k}) \log \left(\frac{A_{t,k} \exp(\theta_{1,k})}{A_{t,k} \exp(\theta_{2,k})} \right) \\
&= A_{t,k} (\exp(\theta_{1,k}) - \exp(\theta_{2,k}) - \exp(\theta_{2,k})(\theta_{1,k} - \theta_{2,k})) \\
&= A_{t,k} d(\theta_{1,k} \| \theta_{2,k}).
\end{aligned}$$

Finally, we show that the one dimensional Bregman is non-increasing in $b_{t,k}$ for $A_{t,k} > 0$.

$$\begin{aligned}
\frac{d([\Phi_t(\theta_1)]_k \| [\Phi_t(\theta_2)]_k)}{db_{t,k}} &= -\log \left(\frac{A_{t,k} \exp(\theta_{1,k}) + b_{t,k}}{A_{t,k} \exp(\theta_{2,k}) + b_{t,k}} \right) \\
&\quad - (A_{t,k} \exp(\theta_{2,k}) + b_{t,k}) \left(\frac{1}{A_{t,k} \exp(\theta_{1,k}) + b_{t,k}} - \frac{1}{A_{t,k} \exp(\theta_{2,k}) + b_{t,k}} \right) \\
&= \log \left(\frac{A_{t,k} \exp(\theta_{2,k}) + b_{t,k}}{A_{t,k} \exp(\theta_{1,k}) + b_{t,k}} \right) + \left(1 - \frac{A_{t,k} \exp(\theta_{2,k}) + b_{t,k}}{A_{t,k} \exp(\theta_{1,k}) + b_{t,k}} \right) \leq 0
\end{aligned}$$

The final inequality comes from the fact that $1 - x \leq -\log x$. The result of this is that we have shown that when $b_{t,k} > 0$ the one dimensional Bregman divergence is less than if $b_{t,k} = 0$. Combining all the results with the assumption that all the elements of the diagonal matrix A_t are upper bounded by one, gives the conclusion that these dynamics are contractive.

$$\begin{aligned}
D(\Phi_t(\theta_1) \| \Phi_t(\theta_2)) &= \sum_{k=1}^p d([\Phi_t(\theta_1)]_k \| [\Phi_t(\theta_2)]_k) \\
&\leq \sum_{k=1}^p A_{t,k} d(\theta_{1,k} \| \theta_{2,k}) \leq \sum_{k=1}^p d(\theta_{1,k} \| \theta_{2,k}) = D(\theta_1 \| \theta_2)
\end{aligned}$$

4.9.4 Proof of Theorem 4.4

The proof of Theorem 4.4 is based on the proof of Theorem 2.2, specialized to the Hawkes process. The strategy is to bound the excess loss at any given moment, and

then add all of these bounds from $t = 1$ to T/δ . Importantly we use the fact that $\tilde{\ell}_t(\hat{\lambda}_t) = \ell_t(\hat{\theta}_t)$ and $\tilde{\ell}_t(\lambda_t) = \ell_t(\theta_t)$. We start with some important properties. The first is the first order optimality condition of line 4 of Algorithm 4.1, which states, for any $\theta \in \Theta$ we have:

$$\langle \eta_t \nabla \ell_t(\hat{\theta}_t) + \nabla Z(\tilde{\theta}_{t+1}) - \nabla Z(\hat{\theta}_t), \tilde{\theta}_{t+1} - \theta \rangle \leq 0.$$

By rearranging terms we get the form that is used.

$$\langle \ell_t(\hat{\theta}_t), \tilde{\theta}_{t+1} - \theta_t \rangle \leq \frac{1}{\eta_t} \langle Z(\hat{\theta}_t) - \nabla Z(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle \quad (4.17)$$

The second important fact is that a Bregman divergence induced by a function Z takes on the form $D(a\|b) = Z(a) - Z(b) - \langle \nabla Z(b), a - b \rangle$, and therefore we have the following:

$$D(a\|b) - D(a\|c) - D(c\|b) = \langle \nabla Z(b) - \nabla Z(c), c - a \rangle. \quad (4.18)$$

Using these key facts, we start by bounding the excess loss at a single time point.

$$\ell_t(\hat{\theta}_t) - \ell_t(\theta_t) \leq \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \theta_t \rangle \quad (4.19a)$$

$$\begin{aligned} &= \langle \nabla \ell_t(\hat{\theta}_t), \tilde{\theta}_{t+1} - \theta_t \rangle + \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \\ &\leq \frac{1}{\eta_t} \langle \nabla Z(\hat{\theta}_t) - \nabla Z(\tilde{\theta}_{t+1}), \tilde{\theta}_{t+1} - \theta_t \rangle + \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \end{aligned} \quad (4.19b)$$

$$\begin{aligned} &= \frac{1}{\eta_t} \left(D(\theta_t\|\hat{\theta}_t) - D(\theta_t\|\tilde{\theta}_{t+1}) - D(\tilde{\theta}_{t+1}\|\hat{\theta}_t) \right) + \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle \\ &\quad (4.19c) \end{aligned}$$

Equation 4.19a is due the convexity of the function ℓ_t , Equation 4.19b uses Equation 4.17, and Equation 4.19c is the application of equation 4.18. We add and subtract

necessary Bregman divergences, and bound differences separately.

$$\begin{aligned}
\ell_t(\hat{\theta}_t) - \ell_t(\theta_t) &\leq \frac{1}{\eta_t} \left(D(\theta_t \|\hat{\theta}_t) - D(\theta_{t+1} \|\hat{\theta}_{t+1}) \right) \\
&\quad + \frac{1}{\eta_t} \left(D(\theta_{t+1} \|\hat{\theta}_{t+1}) - D(\Phi_t(\theta_t, W) \|\hat{\theta}_{t+1}) \right) \\
&\quad + \frac{1}{\eta_t} \left(D(\Phi_t(\theta_t, W) \|\hat{\theta}_{t+1}) - D(\theta_t \|\tilde{\theta}_{t+1}) \right) \\
&\quad - \frac{1}{\eta_t} D(\tilde{\theta}_{t+1} \|\hat{\theta}_t) + \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle
\end{aligned}$$

We will bound each of these lines separately.

$$\begin{aligned}
&D(\theta_{t+1} \|\hat{\theta}_{t+1}) - D(\Phi_t(\theta_t, W) \|\hat{\theta}_{t+1}) \\
&= Z(\theta_{t+1}) - Z(\Phi_t(\theta_t, W)) + \langle \nabla Z(\hat{\theta}_{t+1}), \Phi_t(\theta_t, W) - \theta_{t+1} \rangle \quad (4.20a)
\end{aligned}$$

$$\leq \langle \nabla Z(\hat{\theta}_{t+1}) - \nabla Z(\theta_{t+1}), \Phi_t(\theta_t, W) - \theta_{t+1} \rangle \quad (4.20b)$$

$$\leq \|\nabla Z(\hat{\theta}_{t+1}) - \nabla Z(\theta_{t+1})\|_2 \|\Phi_t(\theta_t, W) - \theta_{t+1}\|_2 \quad (4.20c)$$

$$\begin{aligned}
&= \|\delta(\hat{\lambda}_{t+1} - \lambda_{t+1})\|_2 \|\log(\delta\tilde{\Phi}_t(\lambda_t, W)) - \log(\delta\lambda_{t+1})\|_2 \\
&\leq \delta\sqrt{p}\lambda_{\max} \frac{1}{\lambda_{\min}} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_{t+1}\|_2 \quad (4.20d)
\end{aligned}$$

Equation 4.20a uses the definition of the Bregman divergence, Equation 4.20b the convexity of Z , Equation 4.20c the Cauchy-Schwarz inequality, and Equation 4.20d uses the bounded domain of $\lambda \in [\lambda_{\min}, \lambda_{\max}]^p$ with $\lambda_{\min} > 0$ and the Lipschitz property of the natural logarithm on $[\lambda_{\min}, \lambda_{\max}]$. The next term we bound by using the contractivity assumption on Φ_t .

$$D(\Phi_t(\theta_t, W) \|\hat{\theta}_{t+1}) - D(\theta_t \|\tilde{\theta}_{t+1}) = D(\Phi_t(\theta_t, W) \|\Phi_t(\tilde{\theta}_{t+1}, W)) - D(\theta_t \|\tilde{\theta}_{t+1}) \leq 0$$

To bound the final term, we use the strong convexity property of $Z(\theta)$ which implies

that $D(\theta_1\|\theta_2) \geq \frac{\delta\lambda_{\min}}{2}\|\theta_1 - \theta_2\|_2^2$ (Equation 4.9).

$$\begin{aligned} & \langle \nabla \ell_t(\hat{\theta}_t), \hat{\theta}_t - \tilde{\theta}_{t+1} \rangle - \frac{1}{\eta_t} D(\tilde{\theta}_{t+1}\|\hat{\theta}_t) \\ & \leq \|\nabla \ell_t(\hat{\theta}_t)\|_2 \|\hat{\theta}_t - \tilde{\theta}_{t+1}\|_2 - \frac{\delta\lambda_{\min}}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|_2^2 \end{aligned} \quad (4.21a)$$

$$\leq \frac{\eta_t}{2\delta\lambda_{\min}} \|\nabla \ell_t(\hat{\theta}_t)\|_2^2 + \frac{\delta\lambda_{\min}}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|_2^2 - \frac{\delta\lambda_{\min}}{2\eta_t} \|\tilde{\theta}_{t+1} - \hat{\theta}_t\|_2^2 \quad (4.21b)$$

$$\leq \frac{\eta_t p \delta (\lambda_{\max} + x_{\max})^2}{2\lambda_{\min}} \quad (4.21c)$$

In the above, Equation 4.21a uses the Cauchy-Schwarz inequality and Equation 4.9, Equation 4.21b uses Young's inequality, and finally Equation 4.21c applies Equation 4.8. Combining all the terms, we get an upper bound on the excess loss at any given time point of the following form:

$$\begin{aligned} \tilde{\ell}_t(\hat{\lambda}_t) - \tilde{\ell}_t(\lambda_t) & \leq \frac{1}{\eta_t} \left(D(\theta_t\|\hat{\theta}_t) - D(\theta_{t+1}\|\hat{\theta}_{t+1}) \right) \\ & \quad + \frac{\delta\sqrt{p}\lambda_{\max}}{\eta_t\lambda_{\min}} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_t\|_2 + \frac{\eta_t p \delta (\lambda_{\max} + x_{\max})^2}{2\lambda_{\min}}. \end{aligned}$$

To get the final bound, we must add these terms over the entire length of the optimization process from $t = 1, \dots, T/\delta$. To do this, we first show how the telescoping of the Bregman divergence terms happens. In the following lines, we use the assumption that η_t is positive and non-increasing in t as well as the upper bound on the

Bregman divergence from Equation 4.10.

$$\begin{aligned}
& \sum_{t=1}^{T/\delta} \frac{1}{\eta_t} \left(D(\theta_t \| \hat{\theta}_t) - D(\theta_{t+1} \| \hat{\theta}_{t+1}) \right) \\
&= \frac{1}{\eta_1} D(\theta_1 \| \hat{\theta}_1) - \frac{1}{\eta_{T/\delta}} D(\theta_{T/\delta+1} \| \hat{\theta}_{T/\delta+1}) + \sum_{t=2}^{T/\delta} D(\theta_t \| \hat{\theta}_{t+1}) \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) \\
&\leq \frac{1}{\eta_1} D(\theta_1 \| \hat{\theta}_1) - \frac{1}{\eta_{T/\delta}} D(\theta_{T/\delta+1} \| \hat{\theta}_{T/\delta+1}) + \frac{\delta \lambda_{\max}^2 p}{\lambda_{\min}} \left(\frac{1}{\eta_{T/\delta}} - \frac{1}{\eta_1} \right) \\
&\leq \frac{\delta \lambda_{\max}^2 p}{\eta_{T/\delta} \lambda_{\min}}
\end{aligned}$$

Using this, we combine all the terms to get the final bound.

$$\begin{aligned}
& \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \tilde{\ell}_t(\lambda_t) \\
&\leq \frac{\delta \lambda_{\max}^2 p}{\eta_{T/\delta} \lambda_{\min}} + \frac{\delta \sqrt{p} \lambda_{\max}}{\lambda_{\min}} \sum_{t=1}^{T/\delta} \frac{1}{\eta_t} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_{t+1}\|_2 + \frac{p \delta (\lambda_{\max} + x_{\max})^2}{2 \lambda_{\min}} \sum_{t=1}^{T/\delta} \eta_t
\end{aligned}$$

If the time horizon, T , is known, we choose $\eta_1 = \eta_2 = \dots = \eta_{T/\delta}$ to be a constant proportional to $\frac{1}{\sqrt{T/\delta}}$, or if T is unknown, we choose η_t to be proportional to $\frac{1}{\sqrt{t}}$. For

the former choice the regret bound becomes:

$$\sqrt{\delta} \left(\frac{\lambda_{\max}^2 p}{\lambda_{\min}} + \frac{\sqrt{p} \lambda_{\max}}{\lambda_{\min}} \sum_{t=1}^{T/\delta} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_{t+1}\|_2 + \frac{p(\lambda_{\max} + x_{\max})^2}{2 \lambda_{\min}} \right) \sqrt{T}.$$

And for the later choice, we use the fact that $\sum_{t=1}^{T/\delta} \frac{1}{\sqrt{t}} \leq 1 + \int_1^{T/\delta} \frac{1}{\sqrt{t}} dt = 2\sqrt{T/\delta} - 1 < 2\sqrt{T/\delta}$. This brings the overall bound to

$$\sqrt{\delta} \left(\frac{\lambda_{\max}^2 p}{\lambda_{\min}} + \frac{\sqrt{p} \lambda_{\max}}{\lambda_{\min}} \sum_{t=1}^{T/\delta} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_t\|_2 + \frac{p(\lambda_{\max} + x_{\max})^2}{\lambda_{\min}} \right) \sqrt{T}.$$

Both of these are order \sqrt{T} proving the result.

4.9.5 Proof of Lemma 4.5

The proof is a simple inductive argument. We start with the base scenario, at $t = 1$. Since Algorithm 4.1 begins with $\hat{\lambda}_1 = \bar{\mu}$, then we trivially have

$$\hat{\lambda}_1^{(W_1)} = \hat{\lambda}_1^{(W_2)} + (W_1 - W_2)\mathbf{0} = \bar{\mu}.$$

Therefore the results hold for $t = 1$, with $K_1 = \mathbf{0}$. Now we show the inductive step. If we use the update form from Equation 4.12, we can explicitly compute the difference for different values of W .

$$\begin{aligned} \hat{\lambda}_{t+1}^{(W_1)} - \hat{\lambda}_{t+1}^{(W_2)} &= (1 - \eta_t)\alpha^\delta(\hat{\lambda}_t^{(W_1)} - \hat{\lambda}_t^{(W_2)}) + (W_1 - W_2)y_t \\ &= (W_1 - W_2)((1 - \eta_t)\alpha^\delta K_t + y_t) \\ &= (W_1 - W_2)K_{t+1} \end{aligned}$$

Here we assumed that $\hat{\lambda}_t^{(W_1)} = \hat{\lambda}_t^{(W_2)} + (W_1 - W_2)K_t$ and then proved that the next step holds true for $K_{t+1} = (1 - \eta_t)\alpha^\delta K_t + y_t$, as the Lemma states.

4.9.6 Proof of Theorem 4.6

In order to bound the regret of this algorithm, we split the regret into two separate difference terms and bound them individually.

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t) = \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t^{(W)}) + \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t^{(W)}) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t)$$

Here, $\hat{\lambda}_t$ represents the output of Algorithm 4.2 at time t , and $\hat{\lambda}_t^{(W)}$ is the output of Algorithm 4.1 had we used W for any given W . We will show a bound which holds for all $W \in \mathcal{W}$. The bound on the second difference follows directly from Theorem 4.4.

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t^{(W)}) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t) = C_1 \left(1 + \sum_{t=1}^{T/\delta} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_{t+1}\|_2 \right) \sqrt{T} \quad (4.22)$$

In order to bound the first difference, we use the results of Lemma 4.5 to express the loss function in terms of W .

$$\begin{aligned}\tilde{\ell}_t(\hat{\lambda}_t^{(W)}) &= \langle \mathbf{1}, \delta \hat{\lambda}_t^{(W)} \rangle - \langle x_t, \log(\delta \hat{\lambda}_t^{(W)}) \rangle \\ &= \langle \mathbf{1}, \delta(\hat{\lambda}^0 + WK_t) \rangle - \langle x_t, \log(\delta(\hat{\lambda}_t^0 + WK_t)) \rangle = g_t(W)\end{aligned}$$

Given this loss function that is convex in W allows for searching amongst the outputs of the Algorithm 4.1 for different values of W at every time step, to find which W would have produced the best estimate for the current datum. The important observation now is that every output, $\hat{\lambda}_t$ of Algorithm 4.2 is the output of Algorithm 4.1 for the specific value \widehat{W}_t . To see this, notice that if $\hat{\lambda}_t = \hat{\lambda}_t^{(\widehat{W}_t)}$, then $\tilde{\lambda}_{t+1}$ is equivalent to the output of line 4 from Algorithm 4.1 as long as $\tilde{\theta}_{t+1} \in \text{Int } \Theta$. Then in line 9 of Algorithm 4.2 we apply the dynamics and Lemma 4.5 thus producing $\hat{\lambda}_{t+1} = \hat{\lambda}_{t+1}^{(\widehat{W}_{t+1})}$. Since $\hat{\lambda}_1 = \bar{\mu} = \hat{\lambda}_1^{(W)}$ for any W , this shows that at any time $\hat{\lambda}_t = \hat{\lambda}_t^{(\widehat{W}_t)}$.

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t^{(W)}) = \sum_{t=1}^{T/\delta} g_t(\widehat{W}_t) - \sum_{t=1}^{T/\delta} g_t(W)$$

Since Algorithm 4.2, line 8, performs a gradient descent method [18] to produce estimates \widehat{W}_t , we know that the difference is bounded as

$$\sum_{t=1}^{T/\delta} g_t(\widehat{W}_t) - \min_{W \in \mathcal{W}} \sum_{t=1}^{T/\delta} g_t(W) \leq C_2 \sqrt{T/\delta}. \quad (4.23)$$

Combining Equation 4.22 taken with the $W = \arg \min_{W \in \mathcal{W}} \sum_{t=1}^T \|\tilde{\Phi}(\lambda_t, W) - \lambda_{t+1}\|_2$ and

Equation 4.23 gives the result:

$$\sum_{t=1}^{T/\delta} \tilde{\ell}_t(\hat{\lambda}_t) - \sum_{t=1}^{T/\delta} \tilde{\ell}_t(\lambda_t) \leq C \left(1 + \min_{W \in \mathcal{W}} \sum_{t=1}^{T/\delta} \|\tilde{\Phi}_t(\lambda_t, W) - \lambda_{t+1}\|_2 \right) \sqrt{T}.$$

4.9.7 Online Gradient Descent

As a comparison to our methods, we describe an implementation of Online Gradient Descent (OGD) that can be used to learn the network weights, W . In order to do this, we will take the rate, $\hat{\lambda}_t$ to be a direct function of the network estimate \widehat{W}_t using the exponential influence function of Section 4.4 and use the loss function described in Equation 4.6.

$$\begin{aligned}\lambda_t(W) &= \bar{\mu} + \sum_{\tau=1}^{t-1} \alpha^{\delta(t-1-\tau)} W y_\tau = \bar{\mu} + W K_t \\ K_t &\triangleq \sum_{\tau=1}^{t-1} \alpha^{\delta(t-1-\tau)} y_\tau = \alpha^\delta K_{t-1} + y_{t-1} \\ g_t(W) &= \langle \delta \lambda_t(W), \mathbf{1} \rangle - \langle \log(\delta \lambda_t(W)), x_t \rangle \\ \nabla g_t(W) &= \delta \mathbf{1} K_t^T - \text{Diag}(\lambda_t(W))^{-1} x_t K_t^T\end{aligned}$$

Using these values as a framework, we can derive an Online Gradient Descent algorithm for the learning the network in a Hawkes process.

Algorithm 4.3 Learning Network W with Online Gradient Descent

- 1: Initialize $\widehat{W}_1 = W_0, K_1 = \mathbf{0}$
 - 2: **for** $t = 1, \dots, T/\delta$ **do**
 - 3: Observe x_t and incur loss $g_t(W) = \langle \mathbf{1}, \delta \widehat{W}_t K_t \rangle - \langle x_t, \log \delta(\bar{\mu} + \widehat{W}_t K_t) \rangle$
 - 4: Set $\nabla g_t(W) = \delta \mathbf{1} K_t^T - \text{Diag}(\widehat{\lambda}_t^{(\widehat{W}_t)})^{-1} x_t K_t^T$
 - 5: Set $\widehat{W}_{t+1} = \text{proj}_{\mathcal{W}} \left(\widehat{W}_t - \rho_t \nabla g_t(\widehat{W}_t) \right)$
 - 6: Define $y_t \triangleq \sum_{\bar{\tau}_n=\delta t} e_{k_n} \alpha^{(\delta(t+1)-\tau_n)}$
 - 7: Set $K_{t+1} = \alpha^\delta K_t + y_t$
 - 8: **end for**
-

Comparing Algorithms 4.2 and 4.3, we can see how our proposed algorithm, is actually a generalization of OGD, in which instead of learning just the network weights and plugging them into the equation for the current rate, we are also allowed to slightly alter the value of the rate to deviate from the direct computation.

Additionally, comparing the two shows how OGD is simply our algorithm with the parameter $\eta_t = 0$ for all time steps t .

Log-Linear Poisson Autoregressive Model

5.1 Introduction

Imagine recording the times at which each neuron in a biological neural network fires or “spikes” and wishing to infer the structure of the underlying network. Action potentials or neuron spikes can trigger or inhibit spikes in neighboring neurons [66; 84; 85; 86; 87; 88; 89], so understanding excitation and inhibition among neurons provides key insight into the structure and operation of the neural network. A central question in the design of this experiment is “*for how long must I collect data before I can be confident that my inference of the neural network is accurate?*” Clearly the answer to this question will depend not only on the number of neurons being recording, but also on what we may assume *a priori* about the network. Unfortunately, existing statistical and machine learning theory do not address this problem, which is the focus of this chapter.

The above example of a network of firing neurons can be modeled as an autoregressive point process. That is, at each time t , we observe a high-dimensional vector of counts, and the distribution of those counts depends on past observed counts. In

general, auto-regressive models are a widely-used mechanism for studying time series in which each observation depends on the past sequence of observations. Inferring these dependencies is a key challenge in many settings, including finance, biology, geoscience, and machine learning for several reasons. Specifically, a precise understanding of these dependencies facilitates more accurate predictions and interpretable models of the forces that determine the distribution of each new observation.

Much of the auto-regressive modeling literature focuses on Gaussian noise and perturbation models, but in many settings Gaussian noise fails to capture the data at hand. This challenge arises, for instance, when observations correspond to count data (*i.e.*, when we collect data by counting individual events such as neurons spiking). Time series count data arise in a variety of applications, including vehicular traffic analysis [108; 109], finance [110; 90; 111; 91], social network analysis [47; 82; 65; 63; 83], biological neural networks [66; 84; 85; 86; 87; 88; 89], epidemiology [93], and seismology [94; 95]. Because of their prevalence across application domains, time series count data have been studied for decades (*cf.* [112; 113; 114; 115; 116]). In these and other settings, the interactions among entities are critical to a fundamental understanding of the underlying dependencies and/or network structure and accurate predictions of likely future events.

Although a substantial fraction of this literature is focused on univariate time series, this work focuses on multivariate settings, particularly where the vector observed at each time is high-dimensional relative to the length of the time series. In the above examples, the dimension of each observation vector would be the number of neurons in a neural network, the number of people in a social network, or the number of interacting financial instruments.

The above examples of time series count data are auto-regressive in that the likelihood of future events depends on the past events. In this chapter, we conduct a detailed investigation of a particular time series count data model: the *vector*

log-linear Poisson autoregressive (PAR) model. The PAR model has been explicitly studied in [14; 15; 16] and is closely related to the continuous-time Hawkes point process model [11; 12; 13] and the discrete-time INGARCH model [117; 118; 119].

This work focuses on estimating the parameters of a vector PAR model from a time series of count data. We adopt a regularized likelihood estimation approach that generalizes past work on Poisson inverse problems (*cf.* [120; 121; 122; 123]). While similar algorithms have been proposed in the above-mentioned PAR literature, little is known about their *sample complexity* or *how inference accuracy scales with the key parameters such as the size of the network or number of entities observed, the time spent collecting observations, and the density of edges within the network or dependencies among entities*. The temporal dependence among events can make such analyses particularly challenging and beyond the scope of much current research in high-dimensional statistical inference (see [124] for an overview).

That said, there has been a large body of work providing theoretical results for certain high-dimensional models under low-dimensional structural constraints (see *e.g.*, [125; 123; 126; 127; 128; 129; 130; 131]). The majority of prior work has focussed on the setting where samples are independent and/or follow a Gaussian distribution. In the Poisson auto-regressive model, we have dependent count data samples and signal-dependent Poisson noise. There has been some work [125; 123; 132] which provides results for non-Gaussian noise but still rely on independent noise.

Perhaps the most closely related prior work to our setting in the high-dimensional setting is [133]. In [133], several performance guarantees are provided for different linear Gaussian problems with dependent samples including the Gaussian autoregressive model. Since [133] deal exclusively with linear Gaussian models, they exploit many properties of linear systems and Gaussian random variables that cannot be applied to non-Gaussian and non-linear autoregressive models. Another similar work is [134] which studies a general framework for counting processes and provides

estimation bounds for a LASSO type estimator. However, unlike their work, our work considers the very high-dimensional setting with an assumed underlying sparsity of relations, in the non-asymptotic setting.

In this chapter, we develop performance guarantees for the vector PAR model that provide sample complexity guarantees on the high-dimensional setting under the low-dimensional structural assumption of sparsity of the underlying auto-regressive parameter matrix. In particular, our main contributions are the following:

- Derivation of the stationary distribution for the multivariate log-linear Poisson autoregressive model, under the assumption of a symmetric adjacency matrix.
- Formulation of a maximum penalized likelihood estimator for vector PAR models in high-dimensional settings with sparse structure.
- Mean-squared-error bounds on the proposed estimator as a function of the problem dimension, sparsity, and the number of observations in time.
- Novel analysis techniques leveraging Poisson concentration inequalities which account for the unboundedness of the observed data.

The rest of the chapter is structured as follows: Section 5.2 introduces the log-linear Poisson autoregressive model, Section 5.3 derives the stationary distribution for the model as well as characterizes convergence properties of the model to its stationary distribution. Section 5.4 presents the novel risk bounds associated with the RMLE of the process. Experiments of the inference process are shown in Section 5.5, while proofs and supplementary lemmas are put in Section 5.7.

5.2 Problem Formulation

In this chapter we consider the log-linear vector Poisson autoregressive model:

$$x_{t+1}|x_t \sim \text{Poisson}(e^{\bar{\mu}-Wx_t}), \tag{5.1}$$

where $\{x_t\}_{t=0}^\infty$ are M -variate vectors and $W \in [0, W_{\max}]^{M \times M}$ is some unknown parameter matrix and $\bar{\mu} \in [\mu_{\min}, \mu_{\max}]^M$ is a known, constant rate parameter. A similar model appears in [16], but that work focuses on maximum likelihood and weighted least squares estimators in univariate settings that are known to perform poorly in high-dimensional settings (as is our focus).

We can express state the conditional distribution explicitly as:

$$\mathbb{P}(x_{t+1} = y | x_t = x) = \prod_{m=1}^M \frac{\exp(-e^{\bar{\mu}_m - W_m^T x}) \cdot e^{(\bar{\mu}_m - W_m^T x) y_m}}{y_m!},$$

where x and y are M -variate vectors and W_m is a column vector representing the m^{th} row of the matrix W (i.e. W_m^T is the m^{th} row of W).

In general, we observe T samples $\{x_t\}_{t=0}^T$ and our goal is to infer the matrix W . In the setting where M is large, we need to impose additional assumptions on W in order to have strong performance guarantees. In particular we may assume sparsity or low-rank structure. We will assume that the matrix W is s -sparse meaning that W belongs to the following class:

$$\mathcal{M}_S = \{W \in [0, W_{\max}]^{M \times M} \mid \sum_{\ell=1}^M \sum_{m=1}^M \mathbf{1}(|W_{\ell,m}| \neq 0) \leq s\}.$$

We can assess this parameter class using the element-wise ℓ_1 -norm along with the negative log likelihood:

$$\frac{1}{T} \sum_{t=0}^{T-1} \sum_{m=1}^M \left(\exp(\bar{\mu}_m - W_m^T x_t) - W_m^T x_t x_{m,t+1} \right) + \lambda \|W\|_1. \quad (5.2)$$

The decomposability of the norm means that we have the property that

$$\|W\|_1 = \|W_{\mathcal{S}}\|_1 + \|W_{\mathcal{S}^C}\|_1$$

where \mathcal{S} is some set of indices of W and \mathcal{S}^C the complimentary set. For the rest of the chapter we will assume that $\mathcal{S} = \{(i, j) : W_{i,j} \neq 0\}$.

5.3 Stationary Distribution and Mixing Time

The stationarity of processes are heavily studied particularly in many sequential data settings. One reason that studying stationary distributions is important is that by knowing which states are more likely than others, we can learn about the preferential states of a system, which may help to understand about the underlying mechanisms. In this section we assume that $W = W^T$ which ensures reversibility of the Markov chain described by the process defined by $x_{t+1} \sim \text{Poisson}(e^{\bar{\mu}-Wx_t})$. Notice that we have ensured stability of the system by assuming that every element of $W \geq 0$. First we derive the stationary distribution $\pi(x)$ where $x \in \mathbb{N}^M$, and then establish bounds on the mixing time. Note that this is a countable Markov chain with transition kernel:

$$P(x, y) = \mathbb{P}(x_{t+1} = y | x_t = x) = \prod_{m=1}^M \frac{1}{y_m!} \exp(-e^{\bar{\mu}_m - W_m^T x}) \exp((\bar{\mu}_m - W_m^T x) y_m).$$

Lemma 1. The Markov chain $x_{t+1} \sim \text{Poisson}(e^{\bar{\mu}-Wx_t})$ with $W_{i,j} \geq 0$ for all $1 \leq i, j \leq M$ is a reversible Markov chain with stationary distribution:

$$\pi(x) = \frac{1}{Z \prod_{m=1}^M x_m!} \exp\left(\bar{\mu}^T x + \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x}\right),$$

where $Z = \sum_x \frac{1}{\prod_{m=1}^M x_m!} \exp\left(\bar{\mu}^T x + \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x}\right)$. Further, for any $y \in \mathbb{Z}_+^M$, if $\bar{\mu}_m \leq \mu_{\max}$ for all $1 \leq m \leq M$ and for some $\mu_{\max} < \infty$,

$$\|P^t(y, \cdot) - \pi(\cdot)\|_{TV} \leq \left(1 - \frac{1}{e^{2Me^{\mu_{\max}}}}\right)^t.$$

Proof. To prove the form of the stationary distribution we show that

$$\pi(y) = \sum_x \pi(x) P(x, y),$$

where

$$P(x, y) = \left(\prod_{m=1}^M y_m! \right)^{-1} \exp \left(\bar{\mu}^T y - \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x} - x^T W y \right).$$

Plugging in $\pi(x)$ as specified,

$$\begin{aligned} & \sum_x \pi(x) P(x, y) \\ &= Z^{-1} \sum_x \left(\prod_{m=1}^M x_m! y_m! \right)^{-1} \exp \left(\bar{\mu}^T x + \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x} + \bar{\mu}^T y - \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x} - x^T W y \right) \\ &= Z^{-1} \sum_x \left(\prod_{m=1}^M x_m! y_m! \right)^{-1} \exp(\bar{\mu}^T(x + y) - x^T W y) \\ &= \frac{1}{Z \prod_{m=1}^M y_m!} \exp \left(\bar{\mu}^T y + \sum_{m=1}^M e^{\bar{\mu}_m - W_m^T y} \right) = \pi(y). \end{aligned}$$

The second to last equality is due to factoring out terms that do not depend on x and then the definition of the Taylor expansion of e^x as well as using the assumption that $W^T = W$.

To prove the upper bound on total variation distance, we define two chains, one chain y_t begins at the stationary distribution and the other independent chain starts at x_t begins at some arbitrary random state x , both with transition kernel P . These two chains are said to be coupled if they are run independently until the first time where the states are equal, then are equal for the rest of the trial. The notation $P^t(x, y)$ denotes the probability of transitioning from state y to state x in exactly t steps. Theorem 5.2 of [135] asserts that:

$$\|P^t(x, \cdot) - \pi(\cdot)\|_{TV} \leq \mathbb{P}_x(\tau_{couple} > t),$$

where $\tau_{couple} := \left\{ \min_{t>0} : x_t = y_t \right\}$. Note first that $\mathbb{P}(\tau_{couple} > t) \leq \prod_{\tau=0}^t (1 - \mathbb{P}(x_\tau = y_\tau = 0))$. Since the chains are independent until τ_{couple} , $\mathbb{P}(x_\tau = y_\tau = 0) = \mathbb{P}(x_\tau =$

0) $\mathbb{P}(y_\tau = 0)$. Note also that:

$$\mathbb{P}(x_\tau = 0 | x_{\tau-1} = x) = \exp\left(-\sum_{m=1}^M e^{\bar{\mu}_m - W_m^T x}\right) \geq \exp\left(-\sum_{m=1}^M e^{\mu_m}\right) \geq \exp(-M e^{\mu_{\max}}),$$

where the final inequality follows from the fact that $W_{i,j} \geq 0$. Hence $\mathbb{P}(\tau_{couple} > t) \leq \prod_{\tau=0}^t (1 - (1/\exp(2M e^{\mu_{\max}}))) = (1 - 1/\exp(2M e^{\mu_{\max}}))^t$. \square

5.4 Bounds for Estimation of Adjacency Matrix

In this section, we turn our attention to deriving bounds for $\|\widehat{W} - W\|_F^2$ using regularized maximum likelihood for large M , but under the assumption that W is sparse. We will first mention the important assumptions and facts about the process that we will need in order to prove the error bounds. It will be important in the derivation of the bounds that observations are bounded by a term that grows logarithmically in T , and additionally that a large fraction of the data is bounded by some constant. Intuitively, these bounds will be important because if the elements of x_t are very large, then the conditional expectation $\exp(\bar{\mu} - Wx_t)$ can be very small. When we know that x_{t+1} is being drawn from a Poisson distribution with mean essentially 0, then there's no way we can learn anything about the matrix W , which is basically the same as not receiving any data at all from that time index. While we can allow some time indices to be very large, it becomes a problem when it becomes too frequent. We use the assumption that all entries are non-negative to ensure stability and using this and the definition of the process we can prove that the process will have the desired bounds, as described in the following lemmas.

Lemma 2. If $\log MT \geq 1$, there exists constants C_1 and c_1 which depend on the value μ_{\max} , but are independent of T and M , such that $0 \leq x_{t,m} \leq C_1 \log(MT)$ with probability at least $1 - e^{-c_1 \log(MT)}$ for all $1 \leq t \leq T$ and $1 \leq m \leq M$.

Lemma 3. For any $\alpha \in (0, 1)$ such that αMT is an integer, there exist constants C_2 and c_2 which depend on the values of μ_{\max} and α , but independent of T and M , such that with probability at least $1 - e^{-c_2 MT}$, $0 \leq x_{t,m} \leq C_2$ for at least αMT of the indices.

In addition to needing bounds on the data we will need bounds on the inner products of the rows of the matrix W with the data. These inner products will be useful to upper and lower bound the conditional expectations of the data. Using the previous lemmas we have additional information about the range of the products $W_m^T x_t$, for all $1 \leq m \leq M, 1 \leq t \leq T$:

Corollary 1 (Bounded Entries). If $0 \leq x_{t,m} \leq C_1 \log MT$ for all (t, m) , then we also have the following bounds, using the largest element of W , W_{\max} and the largest amount of non-zeros in a row of W denoted by ρ .

$$0 \leq W_m^T x_t \leq \rho W_{\max} C_1 \log MT.$$

Additionally, if at least αMT observations satisfy $0 \leq x_{m,t} \leq C_2$ then, at least ξT columns of the rates are bounded element-wise by:

$$0 \leq W_m^T x_t \leq \rho W_{\max} C_2$$

for all $m = 1, \dots, M$, where $\xi = 1 - (1 - \alpha)M$. We will additionally assume that C_2 is large enough such that $\alpha > \frac{M-1}{M}$ and therefore $\xi \in (0, 1)$.

One last important fact about this process is that the smallest eigenvalue of $\Gamma_t \triangleq \mathbb{E}[x_t x_t^T | x_{t-1}]$ will be bounded strictly greater than 0 for most time indices. This is an important fact, similar to a restricted eigenvalue property, which ensures a diversity of measurements. This guarantee is proven in Lemma 4 and the remark that follows it. Finally, we come to our main theorem about the error bounds of inferring W using the regularized maximum likelihood estimator. In addition to

the previously mentioned observations and assumptions the proof of the theorem will rely heavily on some martingale concentration inequalities which are due to the dependence of the process from one time step to the next.

Theorem 1. If we assume $\lambda \geq \left\| \frac{2}{T} \sum_{t=0}^{T-1} (x_{t+1} - e^{\bar{\mu} - Wx_t}) x_t^T \right\|_\infty$, and let \widehat{W} be any solution to the regularized log-likelihood loss then,

$$\|\widehat{W} - W\|_F^2 \leq O(e^\rho s \lambda^2)$$

with probability at least $1 - 2 \exp(-\min(c_3 T / \rho^2 - c_4 s \log(2M), c_2 MT))$ where c_2, c_3 and c_4 are independent of M, T, ρ and s . Further $\left\| \frac{2}{T} \sum_{t=0}^{T-1} (x_{t+1} - e^{\bar{\mu} - Wx_t}) x_t^T \right\|_\infty \leq \frac{8C_1^2 e^{\mu_{\max}} \log^2(MT)}{\sqrt{T}}$ with high probability yielding the overall error rate of

$$\|\widehat{W} - W\|_F^2 \leq \tilde{O}\left(\frac{e^\rho s}{T}\right)$$

with probability at least $1 - \exp(-c_6 \min(T/\rho^2 - s \log(M), \log(MT)))$ for some c_6 independent of M, T, ρ and s .

Proof. We start the proof by making an important observation about the minimizer of Equation 5.2: this loss function can be completely decoupled by a sum of functions on rows of W . Therefore the RMLE is the matrix of the row-wise RMLEs. We can then use a standard method in empirical risk minimization and use the definition of the minimizer of the regularized likelihood for each row:

$$\begin{aligned} & \frac{1}{T} \sum_{t=0}^{T-1} e^{\bar{\mu}_m} \exp(-\widehat{W}_m^T x_t) + \widehat{W}_m^T x_t x_{t+1,m} + \lambda \|\widehat{W}_m\|_1 \\ & \leq \frac{1}{T} \sum_{t=0}^{T-1} e^{\bar{\mu}_m} \exp(-W_m^T x_t) + W_m^T x_t x_{t+1,m} + \lambda \|W_m\|_1. \end{aligned}$$

Using the model $x_{t+1,m} = \exp(\bar{\mu}_m - W_m^T x_t) + \epsilon_{t,m}$, where $\epsilon_{t,m} = x_{t+1,m} - \mathbb{E}[x_{t+1,m} | x_t]$

is a zero-mean random variable. Hence

$$\begin{aligned} & \frac{1}{T} \sum_{t=0}^{T-1} e^{\bar{\mu}_m} \exp(-\widehat{W}_m^T x_t) + \widehat{W}_m^T x_t (\exp(\mu_m - W_m^T x_t) + \epsilon_{t,m}) + \lambda \|\widehat{W}_m\|_1 \\ & \leq \frac{1}{T} \sum_{t=0}^{T-1} e^{\bar{\mu}_m} \exp(-W_m^T x_t) + W_m^T x_t (\exp(\mu_m - W_m^T x_t) + \epsilon_{t,m}) + \lambda \|W_m\|_1. \end{aligned}$$

From here we can use the definition of a Bregman divergence in order to lower bound the left hand side. An important property of Bregman divergences is that if the function they are induced by is strongly convex, then so is the Bregman divergence. As a result the Bregman can be lower bounded by the ℓ_2 norm, which is where our mean squared error term will come.

$$\begin{aligned} & \frac{1}{T} \sum_{t=0}^{T-1} e^{\bar{\mu}_m} \left(\exp(-\widehat{W}_m^T x_t) - \exp(-W_m^T x_t) - \exp(-W_m^T x_t) (-\widehat{W}_m^T x_t + W_m^T x_t) \right) \\ & \leq \left| \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{t,m} \Delta_m^T x_t \right| + \lambda (\|W_m\|_1 - \|\widehat{W}_m\|_1), \end{aligned}$$

where $\Delta_m = \widehat{W}_m - W_m$. Let $D(\cdot|\cdot)$ denote the Bregman divergence corresponding to $\exp(\cdot)$. Hence

$$\frac{e^{\mu_{\min}}}{T} \sum_{t=0}^{T-1} D(-\widehat{W}_m^T x_t | -W_m^T x_t) \leq \left| \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{t,m} \Delta_m^T x_t \right| + \lambda (\|W_m\|_1 - \|\widehat{W}_m\|_1).$$

Define the set $\mathcal{A}_T = \{t \in (0, 1, \dots, T-1) \mid \forall m \in (0, 1, \dots, M) W_m^T x_t \leq \rho W_{\max} C_2\}$.

Under Corollary 1, $|\mathcal{A}_T| \geq \xi T$ and on \mathcal{A}_T we have $\frac{\partial^2 \exp(x)}{\partial x^2} \geq \exp(-\rho W_{\max} C_2)$.

Therefore on \mathcal{A}_T it is true that $D(-\widehat{W}_m^T x_t | -W_m^T x_t) \geq \frac{\exp(-\rho W_{\max} C_2)}{2} (\Delta_m^T x_t)^2$ and

$D(-\widehat{W}_m^T x_t \| - W_m^T x_t) \geq 0$ on \mathcal{A}_T^c . Hence

$$\begin{aligned}
& \frac{\exp(\mu_{\min} - \rho W_{\max} C_2)}{2T} \sum_{t \in \mathcal{A}_T} (\Delta_m^T x_t)^2 \\
& \leq \left| \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{t,m} \Delta_m^T x_t \right| + \lambda (\|W_m\|_1 - \|\widehat{W}_m\|_1) \\
& = \left| \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{t,m} \Delta_m^T x_t \right| + \lambda (\|W_{m,\mathcal{S}}\|_1 - \|\widehat{W}_{m,\mathcal{S}}\|_1 - \|\widehat{W}_{m,\mathcal{S}^c}\|_1) \\
& \leq \left| \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{t,m} \Delta_m^T x_t \right| + \lambda \|\Delta_{m,\mathcal{S}}\|_1 - \lambda \|\Delta_{m,\mathcal{S}^c}\|_1 \\
& \leq \|\Delta_m\|_1 \left\| \frac{1}{T} \sum_{t=0}^{T-1} x_t \epsilon_{t,m} \right\|_{\infty} + \lambda \|\Delta_{m,\mathcal{S}}\|_1 - \lambda \|\Delta_{m,\mathcal{S}^c}\|_1.
\end{aligned}$$

In the above, we denote \mathcal{S} as the true support of W and have used the decomposability of the ℓ_1 norm. Note that $\left\| \frac{1}{T} \sum_{t=0}^{T-1} x_t \epsilon_{t,m} \right\|_{\infty} \leq \left\| \frac{1}{T} \sum_{t=0}^{T-1} x_t \epsilon_t^T \right\|_{\infty}$, meaning that the maximum absolute value of a column of a matrix is less than the maximum absolute value of the entire matrix. Under the assumption that $2 \left\| \frac{1}{T} \sum_{t=0}^{T-1} x_t \epsilon_t^T \right\|_{\infty} \leq \lambda$ and by the positivity of the left hand side of the inequality, we have that

$$0 \leq \frac{\lambda}{2} \|\Delta_m\|_1 + \lambda \|\Delta_{m,\mathcal{S}}\|_1 - \lambda \|\Delta_{m,\mathcal{S}^c}\|_1.$$

Using the decomposability of the ℓ_1 norm, this inequality implies that for all rows $1 \leq m \leq M$, we have that $\|\Delta_{m,\mathcal{S}^c}\|_1 \leq 3\|\Delta_{m,\mathcal{S}}\|_1$. Now, we sum the previous bound over all indices m . Define $\|W\|_T^2 = \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|W x_t\|_2^2$ for any $W \in \mathbb{R}^{M \times M}$ and define $C_3 = \exp(\mu_{\min} - \rho W_{\max} C_2)$, then we have the overall bound:

$$\frac{C_3}{2} \|\Delta\|_T^2 \leq \frac{\lambda}{2} \|\Delta\|_1 + \lambda \|\Delta_{\mathcal{S}}\|_1 - \lambda \|\Delta_{\mathcal{S}^c}\|_1 \leq \frac{3\lambda}{2} \|\Delta_{\mathcal{S}}\|_1.$$

Therefore we can define the cone on which the matrix Δ must be defined:

$$\mathcal{B}_{\mathcal{S}} := \{\Delta \in [-W_{\max}, W_{\max}]^{M \times M} \mid \|\Delta_{m,\mathcal{S}^c}\|_1 \leq 3\|\Delta_{m,\mathcal{S}}\|_1 \ \forall m \in [1, 2, \dots, M]\},$$

and restrict ourselves to studying properties of matrices in that set. Since $\|\Delta_S\|_1 \leq \sqrt{s}\|\Delta\|_F$ we have that

$$\|\Delta\|_T^2 \leq \frac{3}{C_3} \lambda \sqrt{s} \|\Delta\|_F = \delta_T \|\Delta\|_F, \quad (5.3)$$

where $\delta_T \triangleq \frac{3}{C_3} \lambda \sqrt{s}$. Now we consider three cases: if $\|\Delta\|_T \geq \|\Delta\|_F$, then $\max(\|\Delta\|_T, \|\Delta\|_F) \leq \delta_T$. On the other hand if $\|\Delta\|_T \leq \|\Delta\|_F$ and $\|\Delta\|_F \leq \delta_T$, then $\max(\|\Delta\|_T, \|\Delta\|_F) \leq \delta_T$.

Hence the final case we need to consider is $\|\Delta\|_T \leq \|\Delta\|_F$ and $\|\Delta\|_F \geq \delta_T$. Now we follow a similar proof technique to that used in Raskutti et al. [130], to understand this final scenario. Let us define the following set:

$$\begin{aligned} \mathcal{B}(\delta_T) &\triangleq \\ &\{\Delta \in [-W_{\max}, W_{\max}]^{M \times M} \|\Delta_{m, \mathcal{S}^c}\|_1 \leq 3\|\Delta_{m, \mathcal{S}}\|_1 \forall m, \|\Delta\|_T \leq \|\Delta\|_F, \|\Delta\|_F \geq \delta_T\}. \end{aligned}$$

Further, let us define the alternative set:

$$\begin{aligned} \mathcal{B}'(\delta_T) &\triangleq \\ &\{\Delta \in [-W_{\max}, W_{\max}]^{M \times M} \|\Delta_{m, \mathcal{S}^c}\|_1 \leq 3\|\Delta_{m, \mathcal{S}}\|_1 \forall m, \|\Delta\|_T \leq \|\Delta\|_F, \|\Delta\|_F = \delta_T\}. \end{aligned}$$

We wish to show that for $\Delta \in \mathcal{B}(\delta_T)$, we have $\|\Delta\|_T^2 \geq \kappa \|\Delta\|_F^2$ for some $\kappa \in (0, 1)$ with high probability, and therefore Equation 5.3 would imply that $\max(\|\Delta\|_T, \|\Delta\|_F) \leq \delta_T/\kappa$. We claim that it suffices to show that $\|\Delta\|_T^2 \geq \kappa \|\Delta\|_F^2$ is true on $\mathcal{B}'(\delta_T)$ with high probability. In particular, given an arbitrary non-zero $\Delta \in \mathcal{B}(\delta_T)$, consider the re-scaled matrix $\tilde{\Delta} = \frac{\delta_T}{\|\Delta\|_F} \Delta$. Since $\Delta \in \mathcal{B}(\delta_T)$ and $\mathcal{B}(\delta_T)$ is star-shaped, we have $\tilde{\Delta} \in \mathcal{B}(\delta_T)$ and $\|\tilde{\Delta}\|_F = \delta_T$ by construction. Therefore, if $\|\tilde{\Delta}\|_T^2 \geq \kappa \|\tilde{\Delta}\|_F^2$ is true, then $\|\Delta\|_T^2 \geq \kappa \|\Delta\|_F^2$ is also true. Alternatively if we define the random variable $\mathcal{Z}_T(\mathcal{B}') = \sup_{\Delta \in \mathcal{B}'(\delta_T)} \{\delta_T^2 - \|\Delta\|_T^2\}$, then it suffices to show that $\mathcal{Z}_T(\mathcal{B}') \leq (1 - \kappa)\delta_T^2$.

Recall that the empirical norm is defined as $\|\Delta\|_T^2 = \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta x_t\|_2^2$. Let $\mathcal{N} := N_{\text{pr}}(\gamma; \mathcal{B}'; \|\cdot\|_T)$ denote the proper covering number of \mathcal{B}' in $\|\cdot\|_T$ norm. Now let

$\Delta^1, \Delta^2, \dots, \Delta^{\mathcal{N}}$, be a minimal $\gamma\delta_T$ -proper covering of \mathcal{B}' , so that for all $\Delta \in \mathcal{B}'$, there is a Δ^k in our covering set such that $\|\Delta^k - \Delta\|_T \leq \gamma\delta_T$. We can write:

$$\delta_T^2 - \|\Delta\|_T^2 = \left(\delta_T^2 - \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta^k x_t\|_2^2 \right) + \left(\frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta^k x_t\|_2^2 - \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta x_t\|_2^2 \right)$$

for any $k \in \{1, 2, \dots, \mathcal{N}\}$. By the Cauchy-Schwarz inequality, we have

$$\begin{aligned} \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta^k x_t\|_2^2 - \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta x_t\|_2^2 &= \frac{1}{T} \sum_{t \in \mathcal{A}_T} x_t^T (\Delta^k - \Delta)^T (\Delta^k + \Delta) x_t \\ &\leq \|\Delta^k - \Delta\|_T \|\Delta^k + \Delta\|_T. \end{aligned}$$

By our choice of covering, we have $\|\Delta^k - \Delta\|_T \leq \gamma\delta_T$. On the other hand, by the definition of \mathcal{B}' we have $\|\Delta\|_T \leq \delta_T$ and $\|\Delta^k\|_T \leq \delta_T$ so

$$\|\Delta^k + \Delta\|_T \leq 2\delta_T.$$

Overall, we have established the upper bound $\frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta^k x_t\|_2^2 - \frac{1}{T} \sum_{t \in \mathcal{A}_T} \|\Delta x_t\|_2^2 \leq 2\gamma\delta_T^2$. Hence we have:

$$\mathcal{Z}_T(\mathcal{B}) \leq \max_{1 \leq k \leq \mathcal{N}} \{\delta_T^2 - \|\Delta^k\|_T^2\} + 2\gamma\delta_T^2,$$

where $\mathcal{N} = N_{\text{pr}}(\gamma\delta_T; \mathcal{B}'; \|\cdot\|_T)$. For any Δ^k in our covering set, we may use Lemma 5.

$$\mathbb{P}\left(\delta_T^2 - \|\Delta^k\|_T^2 > \left(1 - \frac{\xi}{2}\omega^2\right)\delta_T^2\right) \leq \exp\left(-\frac{c_3 T}{\rho^2}\right).$$

This line assumes $|\mathcal{A}_T| \geq \xi T$ which holds with probability at least $1 - 2e^{-c_2 M T}$. Now we can use a union bound,

$$\mathbb{P}\left(\max_{k=1,2,\dots,\mathcal{N}} \delta_T^2 - \|\Delta^k\|_T^2 > \left(1 - \frac{\xi}{2}\omega^2\right)\delta_T^2\right) \leq \exp(\log N_{\text{pr}}(\gamma\delta_T, \mathcal{B}', \|\cdot\|_T) - c_3 T/\rho^2).$$

What remains is to bound $\log N_{\text{pr}}(\gamma\delta_T, \mathcal{B}', \|\cdot\|_T)$. Since the proper covering entropy $\log N_{\text{pr}}(\gamma\delta_T, \mathcal{B}', \|\cdot\|_T)$ is upper bounded by the standard covering entropy

$\log N(\frac{\gamma\delta_T}{2}, \mathcal{B}', \|\cdot\|_T)$, it suffices to upper bound this quantity. Viewing the samples $\{x_t\}_{t=0}^T$ as fixed, let us define the zero-mean Gaussian process $\{G_\Delta\}_{\Delta \in \mathcal{B}}$ via $G_\Delta = \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}} \sum_{m=1}^M w_{t,m} \Delta_m^T x_t$ where $\{w_{t,m}\}$ are i.i.d. standard Gaussian random variables. By construction, we have $\text{var}[(G_\Delta - G_{\Delta'})] = \|\Delta - \Delta'\|_T^2$. By the Sudakov minoration [136], for all $\gamma, \delta_T > 0$ we have $(\gamma\delta_T/2)\sqrt{\log N(\gamma\delta_T/2, \mathcal{B}', \|\cdot\|_T)} \leq 4\mathbb{E}_w[\sup_{\Delta \in \mathcal{B}'} W_\Delta]$. Thus, we obtain the upper bound:

$$\sqrt{\log N(\gamma\delta_T/2, \mathcal{B}', \|\cdot\|_T)} \leq \frac{8}{\gamma\delta_T} \mathbb{E}_w \left[\sup_{\Delta \in \mathcal{B}'} \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}} \sum_{m=1}^M w_{t,m} \Delta_m^T x_t \right].$$

The final step is to upper bound the Gaussian complexity $\mathbb{E}_w[\sup_{\Delta \in \mathcal{B}'} \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}} \sum_{m=1}^M w_{t,m} \Delta_m^T x_t]$. Clearly:

$$\begin{aligned} \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}} \sum_{m=1}^M w_{t,m} \Delta_m^T x_t &= \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}} \text{trace}(x_t w_t^T \Delta) \\ &\leq \left\| \frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}_T} x_t w_t^T \right\|_\infty \|\Delta\|_1. \end{aligned}$$

Because $\Delta \in \mathcal{B}'(\delta_n)$ we have $\|\Delta\|_1 = \|\Delta_S\|_1 + \|\Delta_{S^c}\|_1 \leq 4\|\Delta_S\|_1$ and $\|\Delta_S\|_1 \leq \sqrt{s}\|\Delta_S\|_2 \leq \sqrt{s}\|\Delta\|_F = \sqrt{s}\delta_T$. Using this and the results of Lemma 6

$$\begin{aligned} \log N(\gamma\delta_T/2, \mathcal{B}', \|\cdot\|_T) &\leq \frac{2048}{\gamma^2 \delta_T^2} C_2^2 \log(2M) \|\Delta_S\|_1^2 \\ &\leq \frac{2048}{\gamma^2} C_2^2 s \log(2M) \end{aligned}$$

and therefore

$$\mathbb{P} \left(\max_{k=1,2,\dots,N} \delta_T^2 - \|\Delta^k\|_T^2 > \left(1 - \frac{\xi\omega^2}{2}\right) \delta_T^2 \right) \leq \exp \left(\left(\frac{2048 C_2^2 s}{\gamma^2} \right) \log(2M) - \frac{c_3 T}{\rho^2} \right).$$

By setting $\gamma = \frac{\xi\omega^2}{8}$ gives us the desired bound on $\mathcal{B}'(\delta_T)$

$$\begin{aligned} \mathbb{P}\left(\mathcal{Z}_T(\mathcal{B}) > \left(1 - \frac{\xi}{4}\omega^2\right)\delta_T^2\right) &\leq \exp\left(\left(\frac{2^{17}C_2s}{\xi^2\omega^4}\right)\log(2M) - \frac{c_3T}{\rho^2}\right) \\ &= \exp\left(c_4s\log(2M) - \frac{c_3T}{\rho^2}\right). \end{aligned}$$

Overall this tells us that on the set $\mathcal{B}'(\delta_T)$ we have that $\|\Delta\|_T^2 \geq \frac{\xi\omega^2}{4}\|\Delta\|_F^2$ with high probability. Now we return to the main proof. After considering all three cases that can follow from 5.3, we have

$$\max(\|\Delta\|_F^2, \|\Delta\|_T^2) \leq \frac{144}{C_3^2\omega^4\xi^2}s\lambda^2 = O(e^\rho s\lambda)$$

with high probability on the event described in Lemma 3. Combining these probability statements shows that this bound holds with probability at least $1 - 2\exp(-\min(c_3T/\rho^2 - c_4s\log(2M), c_2MT))$.

The final part of the proof, is to choose how to set the regularization parameter λ . We have to satisfy the condition that $\lambda \geq 2\|\frac{1}{T}\sum_{t=0}^{T-1}x_t\epsilon_t^T\|_\infty$ which is $O(\log^3(MT)/\sqrt{T})$ with probability at least $1 - 2\exp(-\min(c_1, c_5)\log(MT))$ by Lemmas 2 and 7. Overall, we then have the main result:

$$\max(\|\Delta\|_F^2, \|\Delta\|_T^2) \leq O\left(e^\rho \frac{s}{T} \log^6(MT)\right)$$

with probability at least $1 - \exp(-c_6\min(T/\rho^2 - s\log(M), \log(MT)))$ for some c_6 independent of M, T, ρ and s . \square

These error bounds provide several important facts about the inference process. The first is that by looking at the high probability statement, we see that we require $T \geq \rho^2s\log(M)$, up to constant terms. If ρ is fixed as a constant for increasing M , this tells us that T needs to be on the order of $s\log(M)$, which is significantly less than the total M^2 parameters which are being estimated, and therefore including

the sparsity assumption has led to a significant gain. The next bit of information from the risk bound is that it provides guidance in the setting of the regularization parameter. We see that we would like to set λ generally as small as possible, since the error ends up scaling approximately like λ^2 , but that it also needs to be at least as large as $\tilde{O}(1/\sqrt{T})$ for the bounds to hold. The balance between setting λ small enough to have low error, while maintaining that it is large enough is an equivalent argument to needing to set λ large enough for it to induce sparsity, but not too large to cause almost all elements to be fixed to 0. The next important fact about the error is that it scales like T^{-1} , which tells us how long we need to observe the process in order to achieve a desired level of accuracy. Finally, we notice that the error scales linearly with the sparsity level s but only logarithmically with the dimension M in order to estimate M^2 parameters. This fact enforces the idea that doing inference in sparse settings can greatly reduce the needed amount of sensing time especially when $s \ll M^2$.

5.5 Experimental Results

In this section we validate our theoretical results with experimental results performed on synthetically generated data. We do this by generating many trials of synthetic data with known generating parameters and then compare the estimated values. For all trials the constant offset vector, $\bar{\mu}$ was set to 0, and the 20×20 adjacency matrices, W , were set such that s randomly assigned indices were given values in the range $[0, 1]$. Data was then generated according the process described in Equation 5.1. An initial burn in period of 100,000 time steps was performed in order to assure that the process had time to sufficiently mix. Then the next T data points were taken and used to perform the estimation. The values of s and T were then varied over a wide range of values. For each (s, T) pair 100 trials were performed and the

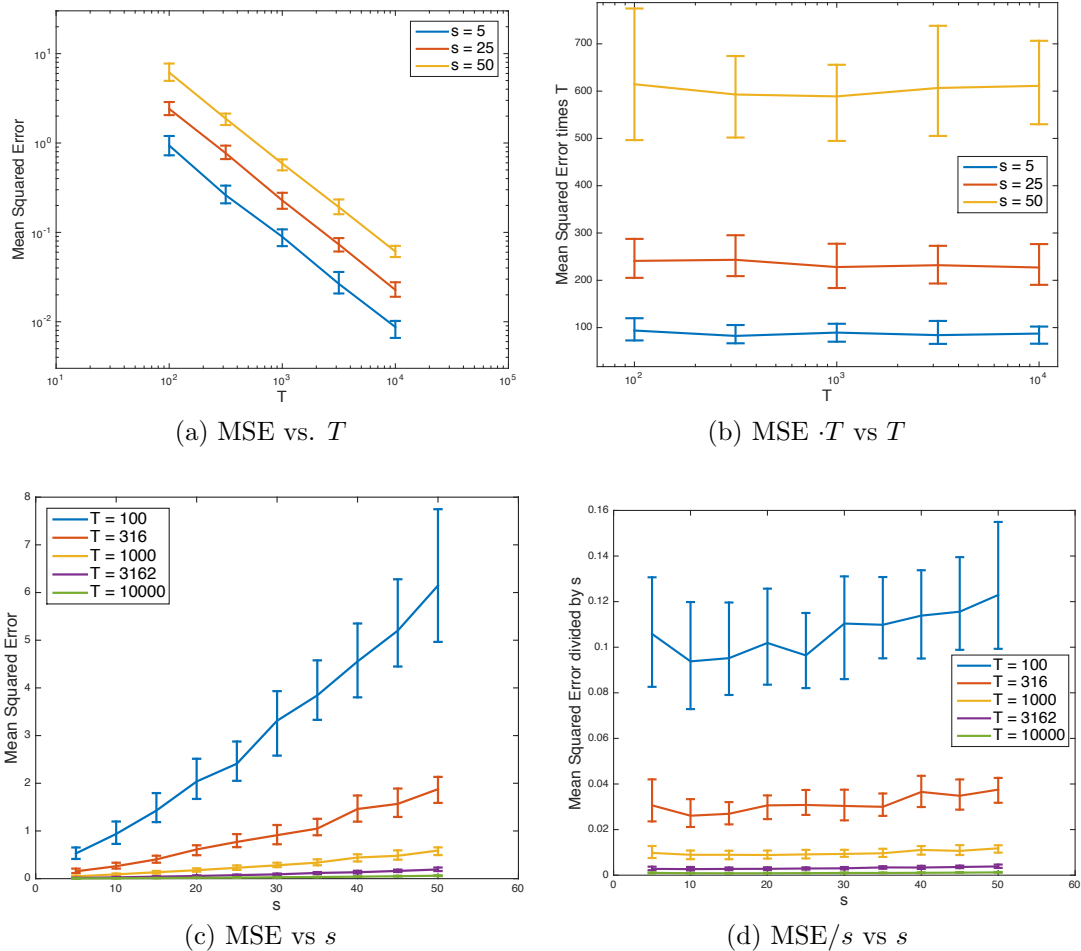


FIGURE 5.1: The top row of plots shows the MSE behavior over a widely varying range of T values, from 100 to 10000, where (a) is the MSE and (b) is the MSE multiplied by T to show that the MSE is behaving as $1/T$. The bottom row shows the MSE behavior over a range of s values, where (c) shows MSE and (d) shows MSE divided by s to show that the MSE is linear in s . In all plots the median value of 100 trials is shown, with error bars denoting the middle 50 percentile.

regularized maximum likelihood estimate \widehat{W} was calculated with $\lambda = 0.1/\sqrt{T}$ and the MSE was recorded. The MSE curves are shown in Figure 5.1. We have shown a series of plots which compare the MSE versus increasing behavior of T and s , as well as comparing the behavior of $\text{MSE} \cdot T$ and of MSE/s . What is plotted in the figure is the median of 100 trials for each (s, T) pair, with error bars denoting the middle 50 percentile. These plots show that indeed that setting λ proportional to $1/\sqrt{T}$ gives

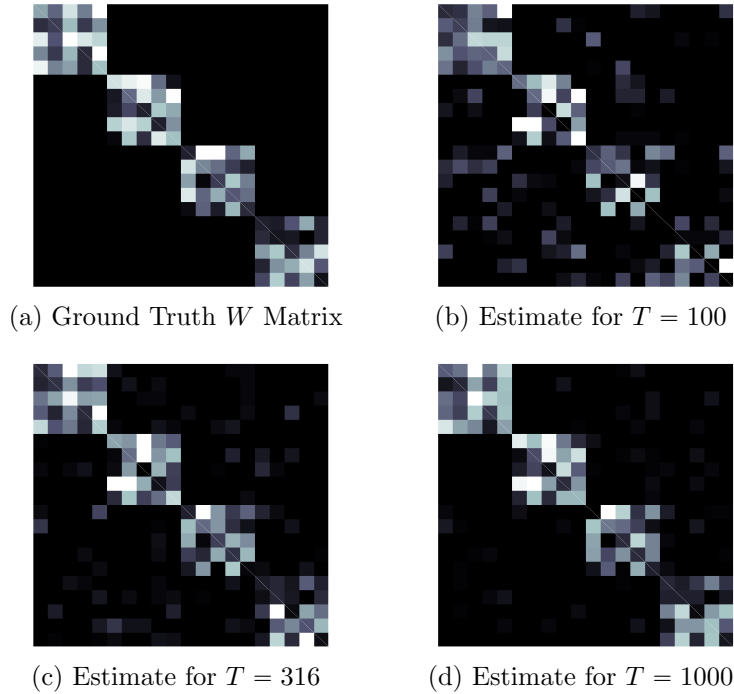


FIGURE 5.2: These images show the ground truth W matrix (a) and 3 different estimates of the matrix created using increasing amounts of data. We observe that even for a relatively low amount of data we have picked out most of the support but with several spurious artifacts. As the amount of data increases, fewer of the erroneous elements are estimated. All images are scaled from 0 (dark) to 1 (bright).

us the desired T^{-1} error decay rate. Additionally, we also see that the error increases approximately linearly in the sparsity level s , as predicted by the theory. Finally, in Figure 5.2 we show one specific example process and the estimates produced. First we show the ground truth matrix, generated to be block diagonal, in order to more easily visualize support structure whereas in the first experiment the support was chosen uniformly at random. One set of data was generated using this matrix, and then estimates were constructed using the first $T = 100, 316$ and 1000 data points. The figure shows how with more data, the estimates become closer to the original, where much of the error comes from including elements off the support of the true matrix.

5.6 Conclusion

The log-linear Poisson autoregressive process has been used successfully in many settings to learn network structure, for instance in neurology [85; 88] they use this model to learn about connections in the brain from firing patterns. However, this model is often used without any guarantees about accuracy. In this chapter we have shown important properties of the Regularized Maximum Likelihood Estimator of the PAR process under a sparsity assumption. First, we have derived a closed form equation for the stationary distribution of the process under the assumption of a symmetric network. Secondly, we have proven bounds on the MSE of the estimator as a function of sparsity, maximum degree of a node, size of network and sensing time. In order to prove this risk bound, we have incorporated many recently developed tools of statistical learning, including concentration bounds for non-Gaussian, dependent random variables. Our results on the MSE show that by incorporating sparsity the error rate scales with the sparsity of the matrix, and not the ambient dimension, which can be a huge saving. Additionally, we see that the error scales like $\frac{1}{T}$ up to log terms, which matches the rate in the Gaussian case.

5.7 Supplementary Lemmas

5.7.1 Proof of Lemma 2

Proof. For all $1 \leq t \leq T$ and $1 \leq m \leq M$, $x_{t,m}|x_{t-1}$ is drawn from a Poisson distribution with mean $e^{\bar{\mu}_m - W_m^T x_{t-1}}$. Because W is assumed to have entries in the range of $[0, W_{\max}]$ we know that $e^{\bar{\mu}_m - W_m^T x_{t-1}} \leq e^{\bar{\mu}_{\max}}$ where $\bar{\mu}_m \leq \mu_{\max}$ for some $\mu_{\max} < \infty$ for all m . Therefore we know that

$$\mathbb{P}(x_{t,m} \geq \eta + e^{\mu_{\max}} | x_{t-1}) \leq \mathbb{P}(Y \geq \eta + e^{\mu_{\max}})$$

where Y is a Poisson random variable with mean $e^{\mu_{\max}}$. To bound this quantity we can use the result in Lemma 8 in the Appendix,

$$\mathbb{P}(Y > \eta + e^{\mu_{\max}}) \leq \exp\left(-\frac{\eta}{4} \log\left(1 + \frac{\eta}{2e^{\mu_{\max}}}\right)\right).$$

Setting $\eta = C_1 \log MT - e^{\mu_{\max}}$,

$$\begin{aligned} \mathbb{P}(Y > C_1 \log MT) &\leq \exp\left(-\frac{C_1 \log MT - e^{\mu_{\max}}}{4} \log\left(1 + \frac{C_1 \log MT - e^{\mu_{\max}}}{2e^{\mu_{\max}}}\right)\right) \\ &\leq \exp\left(-\frac{C_1 \log MT - e^{\mu_{\max}}}{4}\right). \end{aligned}$$

Here, we have assumed that $C_1 \geq e^{\mu_{\max}}(2e - 1)$ and $\log MT \geq 1$. This upper bound is not dependent on the value of x_{t-1} , so this quantity is also an upper bound for the unconditional probability of $x_{t,m} \geq C_1 \log MT$. Using this for a single index t, m of our data x , and taking a union bound over all possible indices $1 \leq m \leq M, 1 \leq t \leq T$ gives

$$\begin{aligned} \mathbb{P}\left(\max_{1 \leq m \leq M, 1 \leq t \leq T} x_{t,m} > C_1 \log MT\right) &\leq \exp\left(\log MT - \frac{C_1 \log MT - e^{\mu_{\max}}}{4}\right) \\ &\leq \exp(-c_1 \log MT) \end{aligned}$$

for $c_1 \leq \frac{C_1 - e^{\mu_{\max}}}{4} - 1$. Thus if $C_1 > \max(e^{\mu_{\max}}(2e - 1), 4 + e^{\mu_{\max}})$, then $c_1 > 0$, and the bound is valid. \square

5.7.2 Proof of Lemma 3

Proof. We are interested in bounding the number of elements in x that are above the value C_2 . Saying at least $j \triangleq \alpha MT$ elements of x are less than a certain value, is equivalent to saying that the j^{th} smallest element is less than that value. Therefore

we can say

$$\begin{aligned} \mathbb{P}(j^{\text{th}} \text{ smallest element of } x > C_2) &= \mathbb{P}\left(\sum_{t=1}^T \sum_{m=1}^M Y_{t,m} \leq j-1\right) \\ &= \sum_{\ell=0}^{j-1} \mathbb{P}\left(\sum_{t=1}^T \sum_{m=1}^M Y_{t,m} = \ell\right) \leq \sum_{\ell=0}^j \sum_{y \in \mathcal{Y}^\ell} \mathbb{P}(Y = y). \end{aligned}$$

Here we define $Y_{t,m} \triangleq \mathbf{1}\{x_{t,m} \leq C_2\}$, and $\mathcal{Y}^\ell = \{y \in \{0, 1\}^{M \times T} \mid \sum_{t=1}^T \sum_{m=1}^M y_{t,m} = \ell\}$. We then can condition the values of Y_t on all previous values of Y and then understand this as a marginal of the joint distribution over Y_t and x_{t-1} . Below we use the notation $Y_{1:t}$ to denote all the time indices of Y from 1 to t , and similarly for y .

$$\begin{aligned} \mathbb{P}(Y = y) &= \prod_{t=1}^T \mathbb{P}(Y_t = y_t \mid Y_{1:t-1} = y_{1:t-1}) \\ &= \prod_{t=1}^T \sum_x \mathbb{P}(Y_t = y_t \mid Y_{1:t-1} = y_{1:t-1}, x_{t-1} = x) \mathbb{P}(x_{t-1} = x \mid Y_{1:t-1} = y_{1:t-1}) \\ &= \prod_{t=1}^T \sum_x \left(\prod_{m=1}^M \mathbb{P}(Y_{t,m} = y_{t,m} \mid x_{t-1} = x) \right) \mathbb{P}(x_{t-1} = x \mid Y_{1:t-1} = y_{1:t-1}) \end{aligned}$$

In the last line we have used the fact that conditioned on x_{t-1} , Y_t is independent across dimensions m , and independent of previous values $Y_{1:t-1}$. We can now make the observation that $\mathbb{P}(x_{t,m} > C_2 \mid x_{t-1} = x_{t-1})$ is exactly the probability that a Poisson random variable with rate $\exp(\bar{\mu}_m - W_m^T x_{t-1})$ is greater than C_2 , which can be upper-bounded by the probability that a Poisson random variable with rate $\exp(\mu_{\max})$ is greater than C_2 because we have assumed all values of W are non-negative. Call this probability $p_{\mu_{\max}}$. Thus we have $\mathbb{P}(Y = y) \leq p_{\mu_{\max}}^{MT - \sum_{t=1}^T \sum_{m=1}^M y_{t,m}}$. Therefore,

$$\begin{aligned}
\mathbb{P}\left(\sum_{t=1}^T \sum_{m=1}^M Y_{t,m} \leq j-1\right) &\leq \sum_{\ell=0}^j \binom{MT}{\ell} p_{\mu_{\max}}^{MT-\ell} = (1+p_{\mu_{\max}})^{MT} - \sum_{\ell=0}^{MT-j-1} \binom{MT}{\ell} p_{\mu_{\max}}^{\ell} \\
&\leq \binom{MT}{MT-j} (1+p_{\mu_{\max}})^j p_{\mu_{\max}}^{MT-j} \\
&\leq \frac{MT e^{MT-j}}{MT-j} (1+p_{\mu_{\max}})^j p_{\mu_{\max}}^{MT-j}.
\end{aligned}$$

The second inequality is from the application of Taylor's Remainder Theorem, and the third is from the fact that $\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$. Now use the fact that $j = \alpha MT$ as stated in the Lemma, to give

$$\begin{aligned}
\mathbb{P}\left(\sum_{t=1}^T \sum_{m=1}^M Y_{t,m} \leq j-1\right) &\leq \left(\frac{p_{\mu_{\max}} e}{1-\alpha}\right)^{(1-\alpha)MT} (1+p_{\mu_{\max}})^{\alpha MT} \\
&\leq \left[\left(\frac{p_{\mu_{\max}} e}{1-\alpha}\right)^{1-\alpha} 2^{\alpha}\right]^{MT}.
\end{aligned}$$

By using Lemma 8 in a similar way as was used in the proof of Lemma 2, $p_{\mu_{\max}}$ can be controlled by C_2 in the following way,

$$\begin{aligned}
p_{\mu_{\max}} = \mathbb{P}(x > C_2) &\leq \exp\left(-\frac{C_2 - e^{\mu_{\max}}}{4} \log\left(1 + \frac{C_2 - e^{\mu_{\max}}}{2e^{\mu_{\max}}}\right)\right) \\
&\leq \exp\left(-\frac{C_2 - e^{\mu_{\max}}}{4}\right),
\end{aligned}$$

when $C_2 \geq e^{\mu_{\max}}(2e-1)$. Plugging the result back into the bound gives

$$\mathbb{P}\left(\sum_{t=1}^T \sum_{m=1}^M Y_{t,m} \leq j-1\right) \leq \left[\left(\frac{\exp(1 - (C_2 - e^{\mu_{\max}})/4)}{1-\alpha}\right)^{1-\alpha} 2^{\alpha}\right]^{MT}.$$

When $C_2 > 4 + e^{\mu_{\max}} + \frac{4\alpha \log(2)}{1-\alpha} - 4 \log(1-\alpha)$ and additionally greater than $e^{\mu_{\max}}(2e-1)$ the condition from above, then the probability of this event is decaying in M and

T . Therefore for $c_2 = -\left(1 - \frac{e^{\mu_{\max}} - C_2}{4} - \log(1 - \alpha)\right)(1 - \alpha) - \alpha \log(2)$, we have the inequality

$$\mathbb{P}(\text{at least } \alpha MT \text{ elements of } x_{t,m} \leq C_2) \geq 1 - e^{-c_2 MT}$$

□

Lemma 4. Define the matrix $\Gamma_t \in \mathbb{R}^{M \times M} = \mathbb{E}[x_t x_t^T]$, where the x_t are generated using the matrix W with all non-negative elements and the vector $\bar{\mu}$. Then the smallest eigenvalue of Γ_t is lower bounded by $\exp(\mu_{\min} - \rho W_{\max} e^{\mu_{\max}})$.

Proof. Notice that Γ_t can be expressed as

$$\begin{aligned} \Gamma_t &= \mathbb{E}[x_t x_t^T] = \sum_{x_{t-1}} \mathbb{E}[x_t x_t^T | x_{t-1}] p(x_{t-1}) \\ &= \mathbb{E}[\exp(\mu - W x_{t-1}) \exp(\mu - W x_{t-1})^T + \text{Diag}(x_t)] \end{aligned}$$

Thus Γ_t will have two parts, one that comes from the expected value of an outer-product of a vector with itself, and one from the diagonal matrix formed by the expected value of x_t . We know that a positively weighted sum of outer-products will be a positive semi-definite matrix, and therefore have a non-negative minimum eigenvalue. The second part is a diagonal matrix, and therefore the smallest eigenvalue of the diagonal matrix is the smallest element on the diagonal. Therefore if we can lower bound the smallest element of the expected value of x_t , then that will be a lower bound on the smallest eigenvalue of Γ_t , because we are taking a positive sum of the two matrices.

$$\mathbb{E}[x_{t,m}] = \mathbb{E}[\exp(\bar{\mu}_m - W_m^T x_{t-1})] \geq e^{\mu_{\min}} \exp(-W_m^T \mathbb{E}[x_{t-1}])$$

This line is an application of Jensen's inequality. To then upper bound the elements of $\mathbb{E}[x_{t-1}]$, we note that since all the elements of W are positive, $x_{t-1,m}$ will be conditionally Poisson with rate less than e^{μ_m} , so no matter the value of x_{t-2} , we have $\mathbb{E}[x_{t-1,m}] \leq e^{\mu_{\max}}$. Therefore,

$$\mathbb{E}[x_{t,m}] \geq \exp(\mu_{\min} - \rho W_{\max} e^{\mu_{\max}}),$$

where ρ is the maximum number of non-zero elements in any row of W , and W_{\max} is the largest value of W . This shows that the minimum eigenvalue of $\Gamma_t > 0$. \square

Remark: While the previous Lemma shows that the matrix $\mathbb{E}[x_t x_t]$ has a smallest eigenvalue which can be lower bounded strictly greater than 0, the same proof can be used to show that $\mathbb{E}[x_t x_t | x_{t-k}]$ for $k \geq 2$ will also have a smallest eigenvalue lower bounded with the same bound. If $k = 1$, then a similar argument holds to show that the smallest eigenvalue of Γ_t is lower bounded by $\mathbb{E}[x_{t,m}] \geq \exp(\mu_{\min} - W_m^T x_{t-1}) \geq \exp(\mu_{\min} - \rho W_{\max} C_2)$ if the elements of x_{t-1} are bounded by C_2 , as will be the case in the setting of Lemma 5. Therefore, we still have a lower bound on the smallest eigenvalue of Γ_t which is strictly larger than 0.

Lemma 5. For any $\Delta \in \mathcal{B}(\delta_T)$ let the sequence $x_{a_1}, x_{a_2}, \dots, x_{a_N}$ where $\xi T \leq N = |\mathcal{A}_T| \leq T$, be randomly drawn according to the Poisson autoregressive model with the additional constraint that each element x_i has support upper bounded by C_2 as in the second part of Corollary 1, then we have

$$\|\Delta\|_T^2 \geq \frac{|\mathcal{A}_T|}{2T} \omega^2 \delta_T^2$$

with probability at least $1 - \exp(-c_3 T / \rho^2)$ for some $c_3 > 0$ which is independent of M, T and s , where ω^2 is the smallest eigenvalue of $\mathbb{E}[x_{a_t} x_{a_t}^T | x_{a_{t-1}}]$.

Proof. Define the sequence a_1, a_2, \dots as the elements of the set \mathcal{A}_T where $\xi T \leq |\mathcal{A}_T| \leq T$. Then define the sequence $(Y_n, n \in \mathbb{N})$ as

$$Y_n \triangleq \frac{1}{T} \sum_{t=1}^n \mathbb{E}[\|\Delta x_{a_t}\|_2^2 | x_{a_{t-1}}] - \frac{1}{T} \sum_{t=1}^n \|\Delta x_{a_t}\|_2^2.$$

It is important to note that even though we are skipping time indices by only focusing on \mathcal{A}_T , we still have a Markov chain where $p(x_{a_i} | x_{a_1}, x_{a_2}, \dots, x_{a_{i-1}}) = p(x_{a_i} | x_{a_{i-1}})$.

Notice the following values:

$$Y_n - Y_{n-1} = \frac{1}{T} \mathbb{E} [\|\Delta x_{a_n}\|_2^2 | x_{a_{n-1}}] - \frac{1}{T} \|\Delta x_{a_n}\|_2^2$$

$$M_n^k = \sum_{i=1}^n \mathbb{E} \left[\left(\frac{1}{T} \mathbb{E} [\|\Delta x_{a_n}\|_2^2 | x_{a_{n-1}}] - \frac{1}{T} \|\Delta x_{a_n}\|_2^2 \right)^k \mid x_{a_1}, \dots, x_{a_{n-1}} \right].$$

The first value shows that $\mathbb{E}[Y_n - Y_{n-1} | x_{a_1}, \dots, x_{a_{n-1}}] = 0$ and therefore Y_n is a martingale. Additionally, on \mathcal{A}_T , we have $0 \leq \|\Delta x_t\|_2^2 \leq \sum_{m=1}^M \|\Delta_m\|_1^2 \|x_t\|_\infty^2 \leq C_2^2 \sum_{m=1}^M \|\Delta_m\|_1^2$. Because $\Delta \in \mathcal{B}'(\delta_T)$, it is true that $\|\Delta_m\|_1 \leq 4\|\Delta_{m,S}\|_1$. We then use the relationship between the ℓ_1 and ℓ_2 norms to say $\|\Delta_{m,S}\|_1 \leq \sqrt{\rho} \|\Delta_{m,S}\|_2 \leq \sqrt{\rho} \|\Delta_m\|_2$ where ρ is the maximum number of non-zeros per row of the true matrix W . Putting these together means $\|\Delta x_t\|_2^2 \leq 16C_2^2 \rho \delta_T^2$. We then define $B \triangleq 16C_2^2 \rho \delta_T^2$. Therefore $|Y_n - Y_{n-1}| \leq \frac{B}{T}$ and the following is true:

$$M_n^k = \sum_{i=1}^n \mathbb{E} \left[\left(\frac{1}{T} \mathbb{E} [\|\Delta x_{a_n}\|_2^2 | x_{a_{n-1}}] - \frac{1}{T} \|\Delta x_{a_n}\|_2^2 \right)^k \mid x_{a_0}, \dots, x_{a_{n-1}} \right] \leq n \left(\frac{B}{T} \right)^k.$$

We can then get a bound on the summation term used in Lemma 9.

$$Z_n \triangleq \sum_{k \geq 2} \frac{\eta^k}{k!} M_n^k \leq n \sum_{k \geq 2} \frac{\eta^k B^k}{T^k k!} = n \left(e^{\eta B/T} - 1 - \frac{\eta B}{T} \right) \triangleq \hat{Z}_n$$

Now we can use Markov's inequality to get a bound on the desired quantity.

$$\mathbb{P}(Y_n \geq y) \leq \mathbb{E}[e^{\eta Y_n}] e^{-\eta y} = \mathbb{E}[e^{\eta Y_n - Z_n + Z_n}] e^{-\eta y} \leq \mathbb{E}[e^{\eta Y_n - Z_n}] e^{\hat{Z}_n - \eta y} \leq e^{\hat{Z}_n - \eta y}$$

The final inequality comes from the use of Lemma 9, which states that the given terms are supermartingales with initial term equal to 1, so the entire expectation is less than or equal to 1. The next step of the proof is to find the optimal value of η to minimize this upper bound.

$$\mathbb{P}(Y_n \geq y) \leq \exp(\hat{Z}_n - \eta y) = \exp \left(n \left(e^{\eta B/T} - 1 - \frac{\eta B}{T} \right) - \eta y \right)$$

Setting $\eta = \frac{T}{B} \log\left(\frac{Ty}{nB} + 1\right)$ yields the lowest such bound, giving

$$\begin{aligned} \mathbb{P}(Y_n \geq y) &\leq \exp\left(n\left(\frac{Ty}{nB} - \log\left(\frac{Ty}{nB} + 1\right)\right) - \frac{Ty}{B} \log\left(\frac{Ty}{nB} + 1\right)\right) \\ &= \exp\left(-nH\left(\frac{Ty}{nB}\right)\right) \end{aligned}$$

where $H(x) = (1+x)\log(1+x) - x$. We can use the fact that $H(x) \geq \frac{3x^2}{2(x+3)}$ for $x \geq 0$ to further simplify the bound.

$$\mathbb{P}(Y_n \geq y) \leq \exp\left(\frac{-3T^2y^2}{2BTy + 6nB^2}\right)$$

From here we, use the fact that we plug in our value of B and we look specifically at the case where $n = |\mathcal{A}_T| \leq T$ and therefore

$$\begin{aligned} \mathbb{P}\left(\frac{1}{T} \sum_{t=1}^{|\mathcal{A}_T|} \mathbb{E}[\|\Delta x_{a_t}\|_2^2 | x_{a_{t-1}}] - \frac{1}{T} \sum_{t=1}^{|\mathcal{A}_T|} \|\Delta x_{a_t}\|_2^2 \geq y\right) \\ \leq \exp\left(-\frac{3Ty^2}{2^5 C_2^2 \rho \delta_T^2 y + 3 \cdot 2^9 C_2^4 \rho^2 \delta_T^4}\right). \end{aligned}$$

To get the final form of the Lemma, we make the following observation which uses the result of Lemma 4

$$\frac{1}{T} \sum_{t=0}^{|\mathcal{A}_T|} \mathbb{E}[\|\Delta x_{a_t}\|_2^2 | x_{a_{t-1}}] \geq \frac{|\mathcal{A}_T|}{T} \omega^2 \|\Delta\|_F^2.$$

Combining the two statements shows that

$$\|\Delta\|_T^2 \geq \frac{|\mathcal{A}_T|}{2T} \omega^2 \|\Delta\|_F^2 = \frac{|\mathcal{A}_T|}{2T} \omega^2 \delta_T^2$$

with probability at least $1 - \exp(-c_3 T/\rho^2)$ where $c_3 \leq \left(\frac{3\xi^2 \omega^4}{2^6 C_2^2 \omega^2 + 3 \cdot 2^{11} C_2^4}\right)$. \square

Lemma 6. Let $(w_t)_{t=0}^T$ be i.i.d. $\mathcal{N}(0, I_{M \times M})$ random vectors which are also independent of $(x_t)_{t=0}^T$, where $x_t \in [0, C_2]^M$. Then

$$\mathbb{E}_w \left[\max_{1 \leq \ell, m \leq M} \left| \left(\frac{1}{\sqrt{T}} \sum_{t \in \mathcal{A}_T} x_t w_t^T \right)_{\ell, m} \right| \right] \leq C_2 \sqrt{2 \log(2M)}.$$

Proof. We first note that

$$\mathbb{E}_w \left[\max_{1 \leq \ell, m \leq M} \left| \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right| \right] = \max_{1 \leq \ell \leq M} \mathbb{E}_w \left[\max_{1 \leq m \leq M} \left| \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right| \right]$$

and additionally that $\sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m}$ is a normal random variable with mean 0 and variance $\sum_{t \in \mathcal{A}_T} x_{t,\ell}^2$. We can use the following standard technique to bound the expected value of the maximum of a set of random variables,

$$\begin{aligned} \exp \left(j \mathbb{E}_w \left[\max_{1 \leq m \leq M} \left| \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right| \right] \right) &\leq \mathbb{E}_w \left[\max_{1 \leq m \leq M} \exp \left(j \left| \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right| \right) \right] \\ &\leq \sum_{m=1}^M \mathbb{E}_w \left[\exp \left(j \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right) + \exp \left(-j \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right) \right] \\ &= 2M \exp \left(\frac{j^2 \sum_{t \in \mathcal{A}_T} x_{t,\ell}^2}{2} \right). \end{aligned}$$

These lines in order use Jensen's inequality, the monotonicity of the exponential function, the sum of positive numbers being larger than the maximum, the fact that $e^{|x|} \leq e^{-x} + e^x$, and the moment generating function of a normal random variable.

By taking logs and setting $j = \sqrt{2 \log(2M) / \sum_{t \in \mathcal{A}_T} x_{t,\ell}^2}$, we come to

$$\mathbb{E}_w \left[\max_{1 \leq m \leq M} \left| \sum_{t \in \mathcal{A}_T} x_{t,\ell} w_{t,m} \right| \right] \leq \sqrt{2 \log(2M) \sum_{t \in \mathcal{A}_T} x_{t,\ell}^2} \leq C_2 \sqrt{2T \log(2M)}$$

Noticing that this bound is independent of the index of the x sequence and dividing both sides by \sqrt{T} proves the stated result. \square

Lemma 7. Assuming all of the entries of W are non-negative, $x_{t,m} \leq C_1 \log(MT)$ for all $1 \leq m \leq M$ and $1 \leq t \leq T$ and that $T \geq 2$ and $\log(MT) \geq 1$, then

$$\max_{i,j} \frac{1}{T} \left| \sum_{t=0}^{T-1} x_{t-1,i} (x_{t,j} - \mathbb{E}[x_{t,j} | x_{t-1}]) \right| \leq 4C_1^2 e^{\mu_{\max}} \frac{\log^3(MT)}{\sqrt{T}}$$

with probability at least $1 - \exp(-c_5 \log(MT))$ where $c_5 = \frac{24C_1^2 e^{\mu_{\max}} - 8C_1^2 - 6}{4C_1^2 + 3}$.

Proof. This proof follows the same basic structure as that of Lemma 5. In order to prove this Lemma, we make use of Markov's inequality and Lemma 9 as they pertain specifically to our problem. Define the sequence $(Y_n, n \in \mathbb{N})$ as

$$Y_n \triangleq \frac{1}{T} \sum_{t=0}^{n-1} x_{t,m} (x_{t+1,\ell} - \mathbb{E}[x_{t+1,\ell}|x_t]).$$

Notice the following values:

$$Y_n - Y_{n-1} = \frac{x_{n-1,m}}{T} (x_{n,\ell} - \mathbb{E}[x_{n,\ell}|x_{n-1}])$$

$$M_n^k = \sum_{i=1}^n \mathbb{E} \left[\left(\frac{x_{i-1,m}}{T} (x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}]) \right)^k \mid x_1, \dots, x_{i-1} \right].$$

The first value shows clearly that $\mathbb{E}[Y_n - Y_{n-1} | x_1, \dots, x_{n-1}] = 0$ and therefore Y_n (and the negative of the sequence, $-Y_n$) is a martingale. Secondly, define Ω as the event where $|x_{m,i}| \leq C_1 \log MT, \forall m \in \{1, 2, \dots, M\}, i \in \{1, \dots, T\}$. On this event we know $|Y_n - Y_{n-1}| \leq \frac{C_1^2 \log^2(MT)}{T} \triangleq B$. Additionally, on Ω the following is true:

$$M_n^2 = \sum_{i=1}^n \mathbb{E} \left[\left(\frac{x_{i-1,m}}{T} (x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}]) \right)^2 \mid x_1, \dots, x_{i-1} \right]$$

$$= \frac{1}{T^2} \sum_{i=1}^n x_{i-1,m}^2 \mathbb{E}[(x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}])^2 | x_{i-1}]$$

$$= \frac{1}{T^2} \sum_{i=1}^n x_{i-1,m}^2 \exp(\mu_\ell - A_\ell^T x_{i-1}) \leq \frac{nC_1^2 \log^2(MT) e^{\mu_{\max}}}{T^2} \triangleq \widehat{M}_n^2$$

where the last step is because $x_{\ell,i} | x_{i-1} \sim \text{Poisson}(\exp(\bar{\mu}_\ell - W_\ell^T x_{i-1}))$ and the mean and variance of a Poisson random variable are equal. The final line uses Assumption

1 on the event Ω . We will also need to bound M_n^k as follows:

$$\begin{aligned}
M_n^k &= \sum_{i=1}^n \mathbb{E} \left[\left(\frac{x_{i-1,m}}{T^2} (x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}]) \right)^k \mid x_1, \dots, x_{i-1} \right] \\
&= \sum_{i=1}^n \mathbb{E} \left[\left(\frac{x_{i-1,m}}{T^2} (x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}]) \right)^2 \left(\frac{x_{i-1,m}}{T^2} (x_{i,\ell} - \mathbb{E}[x_{i,\ell}|x_{i-1}]) \right)^{k-2} \mid x_{i-1} \right] \\
&\leq B^{k-2} M_n^2
\end{aligned}$$

We need to use these values to get a bound on the summation term used in Lemma 9.

$$\begin{aligned}
Z_n &\triangleq \sum_{k \geq 2} \frac{\eta^k}{k!} M_n^k \leq \sum_{k \geq 2} \frac{\eta^k B^{k-2} M_n^2}{k!} \leq \frac{\widehat{M}_n^2}{B^2} \sum_{k \geq 2} \frac{(\eta B)^k}{k!} \triangleq \widehat{Z}_n \\
\widetilde{Z}_n &\triangleq \sum_{k \geq 2} \frac{\eta^k}{k!} (-1)^k M_n^k \leq \widehat{Z}_n
\end{aligned}$$

In the above \widetilde{Z}_n corresponds to the sum corresponding to the negative sequence $-Y_0, -Y_1, \dots$ which we will also need to obtain the desired bound. Now we are able to use a variant of Markov's inequality to get a bound on the desired quantity.

$$\begin{aligned}
\mathbb{P}(|Y_n| \geq y) &= \mathbb{P}(Y_n \geq y) + \mathbb{P}(-Y_n \geq y) \leq \mathbb{E}[e^{\eta Y_n}] e^{-\eta y} + \mathbb{E}[e^{\eta(-Y_n)}] e^{-\eta y} \\
&= \mathbb{E}[e^{\eta Y_n - Z_n + Z_n}] e^{-\eta y} + \mathbb{E}[e^{\eta(-Y_n) - \widetilde{Z}_n + \widetilde{Z}_n}] e^{-\eta y} \\
&\leq \mathbb{E}[e^{\eta Y_n - Z_n}] e^{\widehat{Z}_n - \eta y} + \mathbb{E}[e^{\eta(-Y_n) - \widetilde{Z}_n}] e^{\widehat{Z}_n - \eta y} \leq 2e^{\widehat{Z}_n - \eta y}
\end{aligned}$$

The final inequality here comes from the use of Lemma 9, which states that the given terms are supermartingales with initial term equal to 1, so the entire expectation is less than or equal to 1. The final step of the proof is to find the optimal value of η to minimize this upper bound.

$$\mathbb{P}(|Y_n| \geq y) \leq 2 \exp(\widehat{Z}_n - \eta y) = 2 \exp \left(\frac{\widehat{M}_n^2}{B^2} (e^{\eta B} - 1 - \eta B) - \eta y \right)$$

Setting $\eta = \frac{1}{B} \log \left(\frac{yB}{\widehat{M}_n^2} + 1 \right)$ yields the lowest such bound, giving

$$\begin{aligned} \mathbb{P}(|Y_n| \geq y) &\leq 2 \exp \left(\frac{\widehat{M}_n^2}{B^2} \left(\frac{yB}{\widehat{M}_n^2} - \log \left(\frac{yB}{\widehat{M}_n^2} + 1 \right) \right) - \frac{y}{B} \log \left(\frac{yB}{\widehat{M}_n^2} + 1 \right) \right) \\ &= 2 \exp \left(-\frac{\widehat{M}_n^2}{B^2} H \left(\frac{yB}{\widehat{M}_n^2} \right) \right) \end{aligned}$$

where $H(x) = (1+x) \log(1+x) - x$. We can use the fact that $H(x) \geq \frac{3x^2}{2(x+3)}$ for $x \geq 0$ to further simplify the bound.

$$\mathbb{P}(|Y_n| \geq y) \leq 2 \exp \left(\frac{-3y^2}{2yB + 6\widehat{M}_n^2} \right) = 2 \exp \left(-\frac{3y^2 T^2}{2C_1^2(Ty + 3ne^{\mu_{\max}}) \log^2(MT)} \right)$$

To prove the Lemma, we take the case where $n = T$ and take a union bound over all indices because Y_T considered specific indices m and ℓ , which gives the bound

$$\begin{aligned} &\mathbb{P} \left(\max_{i,j} \frac{1}{T} \left| \sum_{t=0}^{T-1} x_{t-1,i} (x_{t,j} - \mathbb{E}[x_{t,j}|x_{t-1}]) \right| \geq 4C_1^2 e^{\mu_{\max}} \frac{\log^3(MT)}{\sqrt{T}} \right) \\ &\leq \exp \left(\log(2M^2) - \frac{48C_1^4 \exp^{2\mu_{\max}} \log^4(MT)}{8C_1^4 e^{\mu_{\max}} \log^3(MT)/\sqrt{T} + 6C_1^2 e^{\mu_{\max}}} \right) \\ &\leq \exp \left(2 \log(MT) - \frac{24C_1^2 e^{\mu_{\max}} \log(MT)}{4C_1^2/\sqrt{T} + 3} \right) \\ &\leq \exp(-c_5 \log(MT)) \end{aligned}$$

where $c_5 = \frac{24C_1^2 e^{\mu_{\max}} - 8C_1^2 - 6}{4C_1^2 + 3}$ which is positive for sufficiently large C_1 . Here we have additionally assumed that $T \geq 2$ and that $\log(MT) \geq 1$. \square

5.8 Appendix for Chapter 5

5.8.1 Poisson Concentration Bounds

Using Bobkov and Ledoux [137], we have the following concentration bound for Poisson random variables.

Lemma 8. If $x \sim \text{Poisson}(\lambda)$:

$$\mathbb{P}(x - \lambda > t) \leq \exp\left(-\frac{t}{4} \log\left(1 + \frac{t}{2\lambda}\right)\right).$$

5.8.2 Exponential Martingale

We make use of Lemma 3.3 from Houdré and Reynaud-Bouret [138].

Lemma 9. Let $(Y_n, n \in \mathbb{N})$ be a martingale. For all $k \geq 2$, let

$$M_n^k \triangleq \sum_{i=1}^n \mathbb{E}[(Y_i - Y_{i-1})^k | \mathcal{F}_{i-1}].$$

Then for all integers $n \geq 1$ and for all η such that for all $i \leq n$, $\mathbb{E}[\exp(|\eta(Y_i - Y_{i-1})|)] \leq \infty$,

$$\varepsilon_n \triangleq \exp\left(\eta Y_n - \sum_{k \geq 2} \frac{\eta^k}{k!} M_n^k\right)$$

is a super-martingale. Additionally, if $Y_0 = 0$, then $\mathbb{E}[\varepsilon_n] \leq 1$.

Conclusion and Future Directions

In this thesis we have explored problems related to sequential and dependent data, under dynamically changing environments. These modalities present several challenges that are not modeled by the classical independent, identically distributed data setting. Modeling these dependencies and time variability are important, as they are properties demonstrated in many real world settings from neurology, astronomy, microscopy, social networks and seismology. We address the question of sequential data in changing environments by proposing online optimization routines which incorporate dynamical models. By incorporating dynamical models we are able to learn both the state and dynamics of a system simultaneously while making far fewer assumptions than methods in sequential filtering. Additionally, by incorporating these dynamical models, we allow for analysis of dependence between observations by allowing these models to take autoregressive forms. Therefore we can analyze point processes which model complex relationships between nodes in a network, which would be impossible using classical methods.

In our analysis of our proposed online optimization routines, we have provided novel regret bounds and in our analysis of point processes we have provided previ-

ously unknown sample complexity bounds. Both of these quantities are aimed at addressing the question of how much data is necessary to achieve a certain level of performance. By modifying the notion of regret to tracking regret in our scenario, we can characterize how much is lost by doing learning in a streaming setting compared to computationally expensive batch processing, in a dynamic environment. Our sample complexity bounds show how much data is required to do accurate inference in the scenario where there is a large amount of dependence between consecutive observations, but with full processing power. Both of these bounds required using modern statistical learning techniques. Additionally, both of these methods incorporate low-dimensional structure in high-dimensional settings to improve accuracy.

Our work leads to many future directions and open questions:

- Can we generalize methods to learn the dynamical models from broader classes in online learning? Our work offers solutions to learning from a finite collection of models, or to learn a parametric model when the functional form of the loss and dynamics work well together. Is there anything we could do more generally to learn both efficiently?
- How can we modify our online algorithm to learn not only the adjacency matrix in a Hawkes process, but also learn the influence function? We have shown that our method is robust to misspecified knowledge of the influence function, but is there a way to possibly learn it, without storing a large buffer of data? The non-convexity of this problem would require novel algorithms and analyses, but would be a large forward step in streaming analysis of network data.
- Can we modify and generalize our analysis of the log-linear Poisson autoregressive beyond generalized linear models? For instance can we get similar sample complexity bounds for learning an adjacency matrix W , when data are generated as $x_{t+1} \sim \text{Poisson}(g(\bar{\mu} + Wx_t))$ for different functions $g(\cdot)$? For

instance, neurologist have hypothesized that the true underlying model might have g not be the exponential function, but instead be a clipped version i.e. $g(x) = \max(B, \exp(x))$ for some B .

- Can we extend the analysis of PAR to higher order autoregressive models with more memory? Taken to the extreme, could these analyses be used to prove sample complexity bounds for continuous time processes like the Hawkes process? The current analysis of the Hawkes process requires large observation time and tends to only work in the asymptotic regime, where as our results on the PAR hold in the small T regime.

- [11] A. G. Hawkes, “Point spectra of some self-exciting and mutually-exciting point processes,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, pp. 83–90, 1971.
- [12] A. G. Hawkes, “Point spectra of some mutually-exciting point processes,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 33, no. 3, pp. 438–443, 1971.
- [13] D. J. Daley and D. Vere-Jones, *An introduction to the theory of point processes, Vol. I: Probability and its Applications*, Springer-Verlag, New York, second edition, 2003.
- [14] Konstantinos Fokianos, Anders Rahbek, and Dag Tjøstheim, “Poisson autoregression,” *Journal of the American Statistical Association*, vol. 104, no. 488, pp. 1430–1439, 2009.
- [15] Fukang Zhu and Dehui Wang, “Estimation and testing for a poisson autoregressive model,” *Metrika*, vol. 73, no. 2, pp. 211–230, 2011.
- [16] Konstantinos Fokianos and Dag Tjøstheim, “Log-linear poisson autoregression,” *Journal of Multivariate Analysis*, vol. 102, no. 3, pp. 563–578, 2011.
- [17] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization*, John Wiley & Sons, New York, 1983.
- [18] A. Beck and M. Teboulle, “Mirror descent and nonlinear projected subgradient methods for convex programming,” *Operations Research Letters*, vol. 31, pp. 167–175, 2003.
- [19] M. Zinkevich, “Online convex programming and generalized infinitesimal gradient descent,” in *Proc. Int. Conf. on Machine Learning (ICML)*, 2003, pp. 928–936.
- [20] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning and Games*, Cambridge University Press, New York, 2006.
- [21] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari, “Composite objective mirror descent,” in *Conf. on Learning Theory (COLT)*, 2010.
- [22] A. Bain and D. Crisan, *Fundamentals of Stochastic Filtering*, Springer, 2009.
- [23] L. Xie, Y. C. Soh, and C. E. de Souza, “Robust Kalman filtering for uncertain discrete-time systems,” *IEEE Trans. Autom. Control*, vol. 39, pp. 1310–1314, 1994.
- [24] Y. Theodor and U. Shaked, “Robust discrete-time minimum-variance filtering,” *IEEE Trans. Sig. Proc.*, vol. 44(2), pp. 181–189, 1996.

- [25] N. Merhav and M. Feder, “Universal prediction,” *IEEE Trans. Info. Th.*, vol. 44, no. 6, pp. 2124–2147, October 1998.
- [26] D. Angelosante, J. A. Bazerque, and G. B. Giannakis, “Online adaptive estimation of sparse signals: Where rls meets the 11-norm,” *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3436–3447, 2010.
- [27] M. Herbster and M. K. Warmuth, “Tracking the best linear predictor,” *Journal of Machine Learning Research*, vol. 35, no. 3, pp. 281–309, 2001.
- [28] N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz, “A new look at shifting regret,” arXiv:1202.3323, 2012.
- [29] N. Littlestone and M. K. Warmuth, “The weighted majority algorithm,” *Inf. Comput.*, vol. 108, no. 2, pp. 212–261, 1994.
- [30] E. Hazan and C. Seshadhri, “Efficient learning algorithms for changing environments,” in *Proc. Int. Conf on Machine Learning (ICML)*, 2009, pp. 393–400.
- [31] L. M. Bregman, “The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming,” *Comput. Mathematics and Math. Phys.*, vol. 7, pp. 200–217, 1967.
- [32] Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms and Applications*, Oxford Univ. Press, Oxford, UK, 1997.
- [33] L. Xiao, “Dual averaging methods for regularized stochastic learning and online optimization,” *J. Mach. Learn. Res.*, vol. 11, pp. 2543–2596, 2010.
- [34] J. Langford, L. Li, and T. Zhang, “Sparse online learning via truncated gradient,” *J. Mach. Learn. Res.*, vol. 10, pp. 777–801, 2009.
- [35] B. McMahan, “A unified view of regularized dual averaging and mirror descent with implicit updates,” arXiv:1009.3240v2, 2011.
- [36] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, New Jersey, 2002.
- [37] A. Rakhlin and K. Sridharan, “Online learning with predictable sequences,” arXiv:1208.3728, 2012.
- [38] M. Herbster and M. K. Warmuth, “Tracking the best expert,” *Machine Learning*, vol. 32, pp. 151–178, 1998.
- [39] W.M. Koolen and S. de Rooij, “Combining expert advice efficiently,” in *Proceedings of the 21st Annual Conference on Learning Theory (COLT 2008)*, 2008, pp. 275–286.

- [40] D. Adamskiy, W. M. Koolen, A. Chernov, and V. Vovk, “A closer look at adaptive regret,” in *Proceedings of the 23rd international conference on Algorithmic Learning Theory*, 2012, ALT’12, pp. 290–304.
- [41] A. Gyorgy, T. Linder, and G. Lugosi, “Efficient tracking of large classes of experts,” *IEEE Transaction on Information Theory*, vol. 58, pp. 6709–6725, November 2012.
- [42] V. Vovk, “Aggregating algorithms,” *Conf. on Learning Theory (COLT)*, 1990.
- [43] C.R. Shalizi, A.Z. Jacobs, K.L. Kликner, and A. Clauset, “Adapting to non-stationarity with growing expert ensembles,” arXiv:1103:0949, 2011.
- [44] S. Amari and H. Nagaoka, *Methods of Information Geometry*, American Mathematical Society, Providence, 2000.
- [45] M. J. Wainwright and M. I. Jordan, “Graphical models, exponential families, and variational inference,” *Foundations and Trends in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, December 2008.
- [46] K. S. Azoury and M. K. Warmuth, “Relative loss bounds for on-line density estimation with the exponential family of distributions,” *Machine Learning*, vol. 43, pp. 211–246, 2001.
- [47] M. Raginsky, R. Willett, C. Horn, J. Silva, and R. Marcia, “Sequential anomaly detection in the presence of noise and limited feedback,” *IEEE Transactions on Information Theory*, vol. 58, no. 8, pp. 5544–5562, 2012.
- [48] J.-B. Hiriart-Urruty and C. Lemaréchal, *Fundamentals of Convex Analysis*, Springer, Berlin, 2001.
- [49] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge Univ. Press, Cambridge, UK, 2004.
- [50] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, “Robust stochastic approximation approach to stochastic programming,” *SIAM J. Optim.*, vol. 19, no. 4, pp. 1574–1609, 2009.
- [51] J. Michael Steele, *The Cauchy-Schwarz master class : an introduction to the art of mathematical inequalities*, Cambridge University Press, Cambridge, New York, NY, Port Melbourne, 2010.
- [52] M. Szummer and R. W. Picard, “Temporal texture modeling,” in *Proceedings of International Conference on Image Processing (ICIP)*, 1996.
- [53] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto, “Dynamic textures,” *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

- [54] Avinash Ravichandran, Rizwan Chaudhry, and Rene Vidal, “Dynamic texture toolbox,” 2011, <http://www.vision.jhu.edu>.
- [55] Vincent Studer, Jérôme Bobin, Makhlad Chahid, Hamed Shams Mousavi, Emmanuel Candes, and Maxime Dahan, “Compressive fluorescence microscopy for biological and hyperspectral imaging,” *Proceedings of the National Academy of Sciences*, vol. 109, no. 26, pp. E1679–E1687, 2012.
- [56] R. Marcia and R. Willett, “Compressive coded aperture video reconstruction,” in *Proc. European Signal Processing Conference EUSIPCO*, 2008.
- [57] J. Y. Park and M. B. Wakin, “A multiscale framework for compressive sensing of video,” in *Picture Coding Symposium (PCS)*, Chicago, IL, May 2009.
- [58] Aswin C. Sankaranarayanan, Christoph Studer, and Richard G. Baraniuk, “CS-MUVI: video compressive sensing for spatial-multiplexing cameras,” in *2012 IEEE International Conference on Computational Photography (ICCP)*, Apr. 2012, pp. 1–10.
- [59] D. Angelosante, G. B. Giannakis, and E. Grossi, “Compressed sensing of time-varying signals,” in *Intl. Conf. on Dig. Sig. Proc.*, 2009.
- [60] N. Vaswani and W. Lu, “Modified-CS: Modifying compressive sensing for problems with partially known support,” *IEEE Trans. Sig. Proc.*, vol. 58, pp. 4595–4607, 2010.
- [61] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, “Single pixel imaging via compressive sampling,” *IEEE Sig. Proc. Mag.*, vol. 25, no. 2, pp. 83–91, 2008.
- [62] Z. Lin, M. Chen, and Y. Ma, “The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices,” arXiv:1009:5055, 2010.
- [63] C. Blundell, K. A. Heller, and J. M. Beck, “Modelling reciprocating relationships with hawkes processes,” in *Proc. NIPS*, 2012.
- [64] Scott W. Linderman and Ryan P. Adams, “Discovering latent network structure in point process data,” arXiv:1402.0914, 2014.
- [65] A. Stomakhin, M. B. Short, and A. Bertozzi, “Reconstruction of missing data in social networks based on temporal patterns of interactions,” *Inverse Problems*, vol. 27, no. 11, 2011.
- [66] E. N. Brown, R. E. Kass, and P. P. Mitra, “Multiple neural spike train data analysis: state-of-the-art and future challenges,” *Nature Neuroscience*, vol. 7, no. 5, pp. 456–461, 2004.

- [67] B. Klimt and Y. Yang, “The Enron corpus: a new dataset for e-mail classification research,” in *Proceedings of the European Conference on Machine Learning (ECML)*, 2004, pp. 217–226.
- [68] “<http://www.pbs.org/wgbh/pages/frontline/shows/blackout/california/timeline.html>,” Retrieved June 3, 2010.
- [69] R. Smith and J.R. Emswhiller, “Enron faces collapse as dynege bolts and stock price, credit standing dive,” *The Wall Street Journal*, 2001.
- [70] D. Snyder, *Random Point Processes*, Wiley-Interscience, New York, NY, 1975.
- [71] M. Piccardi, “Background subtraction techniques: a review,” in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, Oct. 2004, vol. 4, pp. 3099 – 3104.
- [72] C. Stauffer and W.E.L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 1999, vol. 2.
- [73] L. Balzano, R. D. Nowak, and B. Recht, “Online identification and tracking of subspaces from highly incomplete information,” in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE, 2010, pp. 704–711.
- [74] F. J. Anscombe, “The transformation of Poisson, binomial and negative-binomial data,” *Biometrika*, vol. 35, pp. 246–254, 1948.
- [75] M. Mäkitalo and A. Foi, “Optimal inversion of the Anscombe transformation in low-count Poisson image denoising,” *IEEE Trans. Image Process.*, vol. 20, no. 1, pp. 99–109, 2011.
- [76] M. Fisz, “The limiting distribution of a function of two independent random variables and its statistical application,” *Colloquium Mathematicum*, vol. 3, pp. 138–146, 1955.
- [77] P. Fryźlewicz and G. P. Nason, “Poisson intensity estimation using wavelets and the Fisz transformation,” Tech. Rep., Department of Mathematics, University of Bristol, United Kingdom, 2001.
- [78] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J-B. Sibarita, and J. Salamero, “Patch-based nonlocal functional for denoising fluorescence microscopy image sequences,” *IEEE Trans. Med. Imag.*, vol. 29, no. 2, pp. 442–454, 2010.
- [79] B. Zhang, J. Fadili, and J-L. Starck, “Wavelets, ridgelets, and curvelets for Poisson noise removal,” *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1093–1108, 2008.
- [80] T. Chan and J. Shen, *Image Processing And Analysis: Variational, PDE, Wavelet, And Stochastic Methods*, Society for Industrial and Applied Mathematics, 2005.

- [81] A. Beck and M. Teboulle, “Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems,” *IEEE Transactions Image Processing*, vol. 18, no. 11, pp. 2419–34, 2009.
- [82] J. Silva and R. Willett, “Hypergraph-based anomaly detection in very large networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 563–569, 2009, doi:10.1109/TPAMI.2008.232.
- [83] K. Zhou, H. Zha, and L. Song, “Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes,” in *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013.
- [84] T. P. Coleman and S. Sarma, “Using convex optimization for nonparametric statistical analysis of point processes,” in *Proc. ISIT*, 2007.
- [85] A. C. Smith and E. N. Brown, “Estimating a state-space model from point process observations,” *Neural Computation*, vol. 15, pp. 965–991, 2003.
- [86] M. Hinne, T. Heskes, and M. A. J. van Gerven, “Bayesian inference of whole-brain networks,” *arXiv:1202.1696 [q-bio.NC]*, 2012.
- [87] M. Ding, CE Schroeder, and X. Wen, “Analyzing coherent brain networks with Granger causality,” in *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, 2011, pp. 5916–8.
- [88] J. W. Pillow, J. Shlens, L. Paninski, A. Sher, A. M. Litke, E. J. Chichilnisky, and E. P. Simoncelli, “Spatio-temporal correlations and visual signalling in a complete neuronal population,” *Nature*, vol. 454, pp. 995–999, 2008.
- [89] M. S. Masud and R. Borisyuk, “Statistical technique for analysing functional connectivity of multiple spike trains,” *Journal of Neuroscience Methods*, vol. 196, no. 1, pp. 201–219, 2011.
- [90] Y. Ait-Sahalia, J. Cacho-Diaz, and R. J. A. Laeven, “Modeling financial contagion using mutually exciting jump processes,” Tech. Rep., National Bureau of Economic Research, 2010.
- [91] A Colin Cameron and Pravin K Trivedi, *Regression analysis of count data*, vol. 53, Cambridge university press, 2013.
- [92] Robert Engle, “Garch 101: The use of arch/garch models in applied econometrics,” *Journal of economic perspectives*, pp. 157–168, 2001.
- [93] M. Kuperman and G. Abramson, “Small world effect in an epidemiological model,” *Physical Review Letters*, vol. 86, no. 13, pp. 2909, 2001.
- [94] D. Vere-Jones and T. Ozaki, “Some examples of statistical estimation applied to earthquake data,” *Ann. Inst. Statist. Math.*, vol. 34, pp. 189–207, 1982.
- [95] Y. Ogata, “Seismicity analysis through point-process modeling: A review,” *Pure and Applied Geophysics*, vol. 155, no. 2-4, pp. 471–507, 1999.

- [96] F.P. Schoenberg, “Facilitated estimation of etas,” *Bulletin of the Seismological Society of America*, vol. 103, pp. 601 – 605, 2013.
- [97] A. Simma and M.I. Jordan, “Modeling events with cascades of poisson processes,” *Proceedings of the Twenty-Sixth Conference of Uncertainty in Artificial Intelligence (UAI2010)*, 2010.
- [98] G. Mohler, “Modeling and estimation of multi-source clustering in crime and security data,” *Annals of Applied Statistics*, vol. 7, pp. 1525 – 1539, 2013.
- [99] D. Sornette and S. Utkin, “Limits of declustering methods for disentangling exogenous from endogenous events in time series with foreshocks, main shocks and aftershocks,” *Physical Review E*, 2009.
- [100] A. Veen and F.P. Schoenberg, “Estimation of space-time branching process models in seismology using an em-type algorithm,” *Journal of the American Statistical Association*, 2008.
- [101] S.W. Linderman and R.P. Adams, “Scalable bayesian inference for excitatory point process networks,” arXiv: 1507.03228v1, 2015.
- [102] P. Reynaud-Bouret and S. Schbath, “Adaptive estimation for Hawkes processes; application to genome analysis,” *Annals of Statistics*, vol. 38, no. 5, pp. 2781–2822, 2010.
- [103] N. R. Hansen, P. Reynaud-Bouret, and V. Rivoirard, “LASSO and probabilistic inequalities for multivariate point processes,” *arXiv preprint arXiv:1208.0570*, 2012.
- [104] S. A. Pasha and V. Solo, “Hawkes-Laguerre reduced rank model for point processes,” in *ICASSP*, 2013.
- [105] Angelos Dassios and Hongbiao Zhao, “Exact simulation of hawkes process with exponentially decaying intensity,” *Electronic Communications in Probability*, vol. 18, pp. no. 62, 1–13, 2013.
- [106] E. Hall and R. Willett, “Dynamical models and tracking regret in online convex programming,” in *Proc. International Conference on Machine Learning (ICML)*, 2013.
- [107] J. Leskovec, L. Backstrom, and J. Kleinberg, “Meme-tracking and the dynamics of the news cycle,” *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2009.
- [108] Per Johansson, “Speed limitation and motorway casualties: a time series count data regression approach,” *Accident Analysis & Prevention*, vol. 28, no. 1, pp. 73–87, 1996.
- [109] David S Matteson, Mathew W McLean, Dawn B Woodard, and Shane G Henderson, “Forecasting emergency medical service call arrival rates,” *The Annals of Applied Statistics*, pp. 1379–1406, 2011.

- [110] Tina Hviid Rydberg and Neil Shephard, “A modelling framework for the prices and times of trades made on the new york stock exchange,” 1999.
- [111] V. Chavez-Demoulin and J. A. McGill, “High-frequency financial data modeling using hawkes processes,” *Journal of Banking & Finance*, vol. 36, no. 12, pp. 3415–3426, 2012.
- [112] Kurt Brännäs and Per Johansson, “Time series count data regression,” *Communications in Statistics-Theory and Methods*, vol. 23, no. 10, pp. 2907–2925, 1994.
- [113] Iain L MacDonald and Walter Zucchini, *Hidden Markov and other models for discrete-valued time series*, vol. 110, CRC Press, 1997.
- [114] Scott L Zeger, “A regression model for time series of counts,” *Biometrika*, vol. 75, no. 4, pp. 621–629, 1988.
- [115] Bent Jørgensen, Soren Lundbye-Christensen, PX-K Song, and Li Sun, “A state space model for multivariate longitudinal count data,” *Biometrika*, vol. 86, no. 1, pp. 169–181, 1999.
- [116] Ludwig Fahrmeir and Gerhard Tutz, *Multivariate statistical modelling based on generalized linear models*, Springer Science & Business Media, 2013.
- [117] Andréas Heinen, “Modelling time series count data: an autoregressive conditional poisson model,” *Available at SSRN 1117187*, 2003.
- [118] Fukang Zhu, “Modeling time series of counts with com-poisson ingarch models,” *Mathematical and Computer Modelling*, vol. 56, no. 9, pp. 191–203, 2012.
- [119] Fukang Zhu, “Modeling overdispersed or underdispersed count data with generalized poisson integer-valued garch models,” *Journal of Mathematical Analysis and Applications*, vol. 389, no. 1, pp. 58–71, 2012.
- [120] R. Willett and R. Nowak, “Multiscale Poisson intensity and density estimation,” *IEEE Transactions on Information Theory*, vol. 53, no. 9, pp. 3171–3187, 2007, doi:10.1109/TIT.2007.903139.
- [121] M. Raginsky, R. Willett, Z. Harmany, and R. Marcia, “Compressed sensing performance bounds under Poisson noise,” *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 3990–4002, 2010, arXiv:0910.5146.
- [122] M. Raginsky, S. Jafarpour, Z. Harmany, R. Marcia, R. Willett, and R. Calderbank, “Performance bounds for expander-based compressed sensing in Poisson noise,” *IEEE Transactions on Signal Processing*, vol. 59, no. 9, 2011, arXiv:1007.2377.
- [123] X. Jiang, R. Willett, and G. Raskutti, “Minimax rates of estimation for high-dimensional linear regression over ℓ_q -balls,” *IEEE Transactions on Information Theory*, vol. 61, pp. 4458–4474, 2015.

- [124] P. Bühlmann and S. van de Geer, *Statistics for High-Dimensional Data: Methods, Theory and Applications*, Springer, 2011.
- [125] S. van de Geer, “High-dimensional generalized linear models and the lasso,” *Annals of Statistics*, vol. 36, pp. 614–636, 2008.
- [126] V. Koltchinskii and M. Yuan, “Sparse recovery in large ensembles of kernel machines,” in *Proceedings of COLT*, 2008.
- [127] L. Meier, S. van de Geer, and P. Bühlmann, “High-dimensional additive modeling,” *Annals of Statistics*, vol. 37, pp. 3779–3821, 2009.
- [128] S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu, “A unified framework for high-dimensional analysis of m -estimators with decomposable regularizers,” *Statistical Science*, vol. 27, no. 4, pp. 538–557, 2010.
- [129] G. Raskutti, M. J. Wainwright, and B. Yu, “Minimax rates of estimation for high-dimensional linear regression over ℓ_q -balls,” *IEEE Transactions on Information Theory*, vol. 57, pp. 6976–6994, 2011.
- [130] G. Raskutti, M. J. Wainwright, and B. Yu, “Minimax-optimal rates for sparse additive models over kernel classes via convex programming,” *Journal of Machine Learning Research*, vol. 13, pp. 398–427, 2012.
- [131] P. Zhao and B. Yu, “On model selection consistency of Lasso,” *Journal of Machine Learning Research*, vol. 7, pp. 2541–2567, 2006.
- [132] X. Jiang, P. Reynaud-Bouret, V. Rivoirard, L. Sansonnet, and R. Willett, “A data-dependent weighted lasso under poisson noise,” *arXiv preprint arXiv:1509.08892*, 2015.
- [133] S. Basu and G. Michailidis, “Regularized estimation in sparse high-dimensional time series models,” *Annals of Statistics*, vol. 43, no. 4, pp. 1535–1567, 2015.
- [134] Niels Richard Hansen, Patricia Reynaud-Bouret, and Vincent Rivoirard, “Lasso and probabilistic inequalities for multivariate point processes,” *Bernoulli*, vol. 21, no. 1, pp. 83–143, 02 2015.
- [135] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*, American Mathematical Society, 2008.
- [136] G. Pisier, *The Volume of Convex Bodies and Banach Space Geometry*, vol. 94 of *Cambridge Tracts in Mathematics*, Cambridge University Press, Cambridge, UK, 1989.
- [137] S. G. Bobkov and M. Ledoux, “On modified logarithmic sobolev inequalities for bernoulli and poisson measures,” *Journal of Functional Analysis*, vol. 156, pp. 347–365, 1998.

- [138] Christian Houdré and Patricia Reynaud-Bouret, “Exponential inequalities, with constants, for u-statistics of order two,” in *Stochastic inequalities and applications*, pp. 55–69. Springer, 2003.

Biography

Eric Calvin Hall was born in Atlanta, GA on November 6th, 1987 to parents Steven and Deborah Hall. He spent his early childhood years in the Atlanta suburb of Sandy Springs, before his father's job took him to Limerick, Ireland and Oslo, Norway for the majority of his elementary schooling. After this European interlude, he moved back to Sandy Springs and soon enrolled at the Westminster Schools, where he graduated from the upper school in 2006.

For his undergraduate studies, he enrolled at the Pratt School of Engineering at Duke University where he focused on signal processing and applied mathematics. In 2010 he received the bachelor's of science in engineering degree in electrical and computer engineering with a second major in mathematics. In 2009 he was inducted into Eta Kappa Nu, the international electrical and computer engineering honor society.

In the fall of 2010, he continued his education by pursuing his graduate degree in electrical and computer engineering at Duke University, under the supervision of Professor Rebecca Willett. He received the masters of science degree from Duke in 2013, before moving to the University of Wisconsin-Madison, where he worked as a visiting researcher while finishing his P.h.D research. He will receive the Ph.D. in electrical and computer engineering from Duke in December 2015. His research interests include machine learning, optimization, statistical signal processing and image processing.