UNIVERSITY OF CALIFORNIA,
IRVINE

Generative Probabilistic Models for Analysis of Communication Event Data with Applications to Email Behavior

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Nicholas Martin Navaroli

Dissertation Committee:
Professor Padhraic Smyth, Chair
Professor Alexander Ihler
Professor Amelia Regan

2014

DEDICATION

To my incredible wife and inspiring parents.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Padhraic Smyth, for making my research career at UCI a very rewarding experience that will leave a lasting impact on my life. I am extremely grateful to have had Dr. Smyth as my advisor, mentor, and teacher. Your interest and dedication to your students was made apparent even before I accepted admission to UCI, back when I was visiting the research lab as a undergraduate. I am thankful for having the opportunity to independently explore new avenues of research topics while simultaneously being taught proper research methods and the types of research questions that need answering via thought-provoking queries, presenting to other students, and our weekly meetings. The weekly meetings we've had, the constructive (but always great) criticism you've given me on papers and presentations, and the high standard of research quality have contributed so much to my education in machine learning (and general research).

I would like to thank Dr. Alexander Ihler and Dr. Amelia Regan, my two other committee members, both whom have had a significant impact on me and my research. I originally met both while taking their graduate courses, and have learned a tremendous amount from them as teachers. I am grateful to have had the opportunity to collaborate on research with Dr. Ihler, where I enjoyed learning and performing research regarding graphical models in a manner outside the classroom. With this research opportunity, I learned a great deal about experimental procedures and the types of questions I should be thinking and answering when comparing different models and procedures. Although I have not had the opportunity to perform research with Dr. Regan, our discussions (both as part of my advancement committee and in outside meetings) have been very helpful in motivation and putting my research into perspective. These contributions, along with those from Dr. Smyth, have allowed me to develop a research framework I am both efficient and confident in, which I will always have to utilize.

My experience at UCI would not have been nearly as rewarding or exciting without the current and past members of the UCI Datalab research group. In particular, I want to thank Jimmy Foulds, who has been both a great friend and research collaborator. Thank you to Chris DuBois, whom I have also gotten to know and collaborated significantly with. I am grateful for the many conversations I've had and time spent with Moshe Lichman, both regarding research ideas and more casual or personal matters. I would like to thank Jon Hutchins, whom I collaborated with during my beginning years at UCI, and later as my mentor at two Google internships. I am extremely thankful for the opportunities you gave me at Google, leading to my upcoming employment there. These people, in addition to America Chambers, Drew Frank, Kevin Bache, Tracy Holsclaw, Ralf Krestel, Eric Nalisnick, Zachary Butler, Noble Kennamer, Geng Ji, Scott Triglia, Corey Shaninger, Scott Crawford, and Arthur Asuncion, were not only great colleagues, but also a major part of my social life at UCI. Thank you for all the fun times we've had.

During my time at UCI, I was extremely fortunate and am grateful to have met my wife, Danielle Davis, and become a close part of her family. I want to thank Danielle for the unconditional love she has given me, and for being there for me as my limits – both aca-

demically and personally – were pushed. She has been very understanding and supportive of my dedication to school, both as a wife and friend, which has meant the world to me. I also want to thank my parents-in-law, Faith and Robert Bolton, for accepting me as part of their family, their love, and their ability to motivate me during rough times.

I would like to thank my family — I would never have dreamed to accomplish such an achievement without their love, support, and motivation. My parents, Martin Navaroli Sr. and Cindy Byerrum, have taught me from the early days of elementary school the value of pursuing knowledge and utilizing that knowledge not only for success, but for truly enjoying what I do as a career. My grandparents, Mary and Albert Navaroli, have always been a significant part of my life and motivated me to work hard and enjoy what I put so much time and dedication to. Thank you to my brother and his wife, Marty Jr. and Ryanna Navaroli, for their friendship and love over the years.

While at UCI, I have had the pleasure of becoming part of two new step-families; I would like thank them for their encouragement and love. In particular, to Jim Byerrum and Elizabeth Hull for including me in their families, their support, and incredible food. Although I do not get to see them much, I am always thankful to have the love and support of my out-of-state aunts, cousins, and grandparents. I also want to thank Leslie Mace, Drew Monohan, Robert Roach, Chris Darr, Sairah Jahangir, and their families — we have become family; I would not be anywhere near where I am now without your friendships and encouragement.

This dissertation was supported by the Department of Defense under the National Defense Science and Engineering Graduate (NDSEG) Fellowship program, UC Irvine under its ICS Fellowship program, and the Office of Naval Research/Multidisciplinary University Research Initiative under grant number N00014-08-1-1015.

CURRICULUM VITAE

Nicholas Martin Navaroli

EDUCATION

Doctor of Philosophy in Computer Science **2014**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2011**
University of California, Irvine *Irvine, California*

Bachelor of Science in Computer Science **2009**
Minor in Mathematics; Outstanding Undergraduate of 2009
California State University, San Bernardino *San Bernardino, California*

FELLOWSHIPS AND AWARDS

National Defense Science and Engineering Graduate Fellowship 2011 – 2014
United States Department of Defense

REFEREED JOURNAL PUBLICATIONS

Nicholas Navaroli, Christopher DuBois, and Padhraic Smyth. **Modeling Individual Email Patterns over time with Latent Variable Models**. *Machine Learning*, 2013.

REFEREED CONFERENCE PUBLICATIONS

Nicholas Navaroli, Christopher DuBois, and Padhraic Smyth. **Statistical Models for Exploring Individual Email Communication Behavior**. *Fourth Asian Conference on Machine Learning*, 2012.

James Foulds, Alexander Ihler, Nicholas Navaroli, and Padhraic Smyth. **Revisiting MAP Estimation, Message Passing and Perfect Graphs**. *Proceedings of the 14th International Conference on AI and Statistics*, 2011.

Nicholas Navaroli, David Turner, Author Concepcion, and Robert Lynch. **Performance Comparison of ADRS and PCA as a Preprocessor to ANN for Data Mining**. *Proceedings of the 8th International Conference on Intelligent Systems Design and Applications*, 2008.

RESEARCH EXPERIENCE

Graduate Research Assistant **2009–2014**
University of California, Irvine *Irvine, California*

Undergraduate Research Assistant **2007-2008**
California State University, San Bernardino *San Bernardino, California*

TEACHING EXPERIENCE

Instructional Student Assistant **2009**
California State University, San Bernardino *San Bernardino, California*

Instructional Student Assistant **2008**
California State University, San Bernardino *San Bernardino, California*

PROFESSIONAL EXPERIENCE

Software Engineering / Research Intern **2013**
Google *Irvine, CA*

Software Engineering / Research Intern **2011**
Google *Irvine, CA*

Research Intern **2008**
Mathematics Department, CSUSB *San Bernardino, CA*

Research Intern **2007**
Center for Bio-Image Informatics, UCSB *Santa Barbara, CA*

ABSTRACT OF THE DISSERTATION

Generative Probabilistic Models for Analysis of Communication Event Data with Applications to Email Behavior

By

Nicholas Martin Navaroli

Doctor of Philosophy in Computer Science

University of California, Irvine, 2014

Professor Padhraic Smyth, Chair

Our daily lives increasingly involve interactions with others via different communication channels, such as email, text messaging, and social media. In this context, the ability to analyze and understand our communication patterns is becoming increasingly important. This dissertation focuses on generative probabilistic models for describing different characteristics of communication behavior, focusing primarily on email communication.

First, we present a two-parameter kernel density estimator for estimating the probability density over recipients of an email (or, more generally, items which appear in an itemset). A stochastic gradient method is proposed for efficiently inferring the kernel parameters given a continuous stream of data. Next, we apply the kernel model and the Bernoulli mixture model to two important prediction tasks: given a partially completed email recipient list, 1) predict which others will be included in the email, and 2) rank potential recipients based on their likelihood to be added to the email. Such predictions are useful in suggesting future actions to the user (i.e. which person to add to an email) based on their previous actions. We then investigate a piecewise-constant Poisson process model for describing the time-varying communication rate between an individual and several groups of their contacts, where changes in the Poisson rate are modeled as latent state changes within a hidden Markov model.

We next focus on the time it takes for an individual to respond to an event, such as receiving an email. We show that this response time depends heavily on the individual's typical daily and weekly patterns — patterns not adequately captured in standard models of response time (e.g. the Gamma distribution or Hawkes processes). A time-warping mechanism is introduced where the absolute response time is modeled as a transformation of effective response time, relative to the daily and weekly patterns of the individual. The usefulness of applying the time-warping mechanism to standard models of response time, both in terms of log-likelihood and accuracy in predicting which events will be quickly responded to, is illustrated over several individual email histories.

Chapter 1

Introduction

Current technology allows us to collect large quantities of time-stamped individual-level event data characterizing our "digital behavior" in contexts such as texting, using email, microblogging, social media interactions, and more — and the volume and variety of this type of data is continually increasing. The resulting time-series of events are rich in behavioral information about our daily lives. Tools for obtaining and visualizing such information are becoming increasingly popular, such as the ability to download your entire email history for mail applications such as Gmail, and various software packages for tracking personal fitness using data from devices such as Fitbit. Examples of such visualization tools are given in Figures 1.1, 1.2, and 1.3. In these figures, summary statistics are calculated and displayed based on several years of my personal email communication patterns.

In this dissertation, we focus on characterizing an individual's communication behavior using generative probabilistic models, a powerful statistical framework for capturing patterns and dependencies between random variables. Such models are advantageous over calculating summary statistics (as in Figures 1.1 to 1.3) in the following ways:

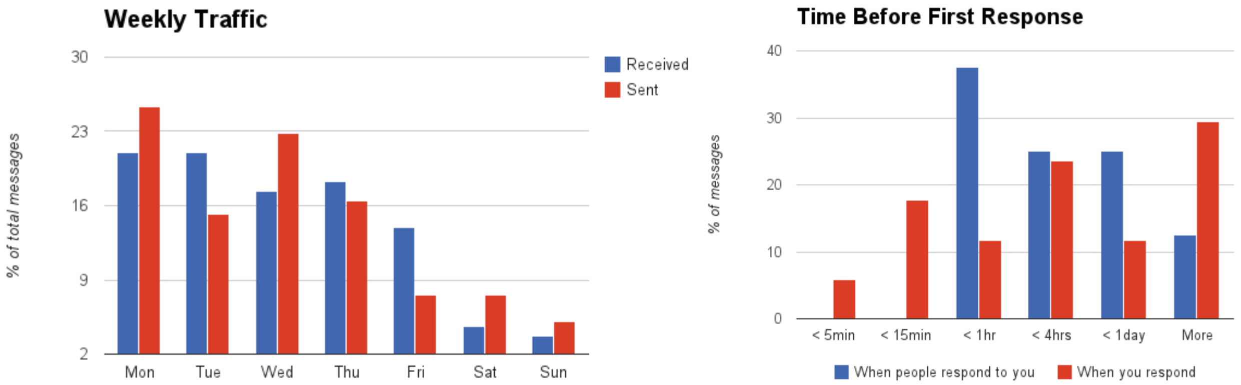


Figure 1.1: Sample charts displaying properties of email behavior, generated from the Gmail meter email analysis tool at `http://gmailmeter.com/`.

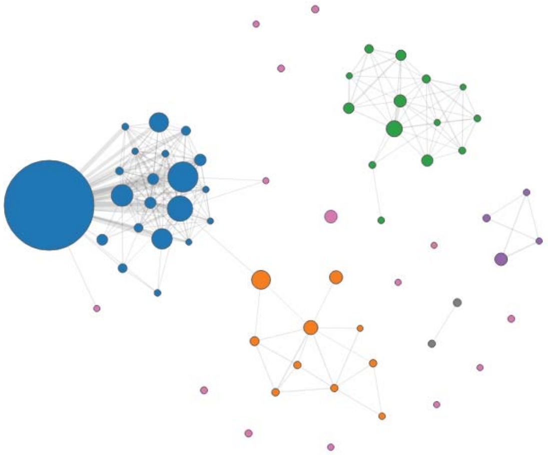
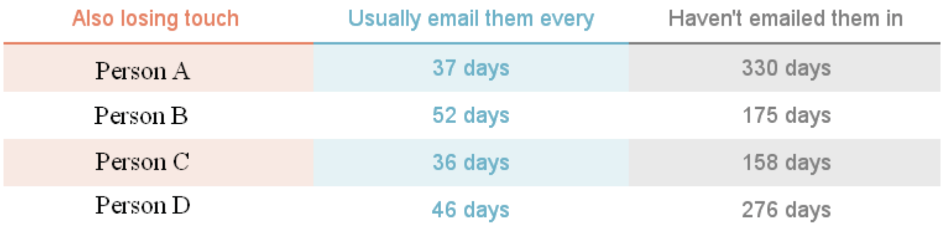


Figure 1.2: A co-appearance network aggregated across several years of email activity, using the Immersion email visualization tool at `https://immersion.media.mit.edu/`.



Also losing touch	Usually email them every	Haven't emailed them in
Person A	37 days	330 days
Person B	52 days	175 days
Person C	36 days	158 days
Person D	46 days	276 days

Figure 1.3: Sample table indicating which (anonymized) individuals have not been in contact for an unusually long period of time, using the email networking tool at `https://www.conspire.com/`.

- The underlying nature of communication behavior can be summarized with model parameters which are easy to interpret.
- It is possible to infer quantities that cannot be easily calculated from raw data, e.g. latent or unobserved groups of contacts the individual typically communicates with.
- Probabilistic models provide a natural framework for making future predictions of an individual's behavior.

We focus on applying probabilistic models for the purpose of extracting useful information from large egocentric histories of time-stamped event data. While we use email as the primary focus in this paper, the techniques and algorithms proposed are also applicable to other types of individual-level event data, such as text messaging or social networking. While email exchanges are one of the original and slower ways to communicate, they remain one of the richest means of digital communication in terms of metadata and information exchanged.

Another motivation for studying email is the broad consensus that current approaches to personal email management are inadequate (Fisher et al., 2006; Mark et al., 2012; Wainer et al., 2011; Whittaker et al., 2011). This has motivated the development of a variety of visualization and clustering techniques as the basis for automated email management tools, intended to help individuals better understand and manage their email (Al-Alwani, 2014; Dredze et al., 2009b,a; Fisher, 2005; Koren et al., 2011; MacLean et al., 2011; Zhou et al., 2012). Examples of such tools were given in Figures 1.1 to 1.3 (Devkar and McReynolds, 2013; ShuttleCloud Corp, 2014; Smilkov et al., 2013).

By accurately capturing the email communication patterns of an individual within a statistical model, future predictions can be made about the individual's behavior, allowing useful recommendations to be presented to an email user regarding email management. As an example, a relatively new feature in the Gmail client is the “Got the wrong Bob?” feature (Roth et al., 2010), where additional recipients to include in an email are suggested to the user based on historical co-appearance patterns and those already included in the email. Such models for communication behavior are also of interest outside of email management, such as the social sciences. For example, there is significant interest in analyzing digital human communication data as a mechanism for investigating social theories about human behavior and interaction (Bryant et al., 2006; Butts, 2008; de Nooy, 2011; Diesner et al., 2005; Garton et al., 1997; Karsai et al., 2014; Miritello et al., 2013; Zenk et al., 2010).

In the remainder of this chapter, we summarize the different email datasets used throughout the dissertation. The chapter concludes with an outline of the remaining dissertation chapters, summarizing the contributions contained within each chapter.

1.1 Email Corpora Used in Experiments

The models and algorithms presented throughout this dissertation are evaluated using four different collections of email inbox histories. Each of the four collections is referred to as an email *corpus*, with each individual inbox contained within a corpus referred to as an email *dataset*. The rest of the subsection describes each of the four email corpora in general, saving details on their collection, parsing, and preprocessing for Appendix A.

Personal Gmail Corpus

The Gmail corpus contains the complete email activity of four different individuals associated with the UCI Datalab research group. We developed a Python script to parse and anonymize all email header information (ignoring any text) from an individual's Gmail account[1]. Summary statistics of the different individuals are given in Table 1.1. These statistics do not

[1] See Appendix A for details and a link to the publicly-available code.

Individual	**First Email**	**Last Email**	**# Received**	**# Sent**	**# Contacts**
Anonymous A	Nov 2007	Jul 2014	24989	8855	922
Anonymous B	Nov 2007	Jun 2014	8614	3296	289
Anonymous C	Aug 2006	May 2014	151064	42929	3798
Anonymous D[2]	Jul 2004	Jan 2013	4803	15924	425

Table 1.1: Summary statistics for email datasets in Gmail corpus.

account for emails deleted by the individual — the python script does not have access to deleted emails.

The email activity for each individual spans multiple years, with a rich history containing thousands of emails sent between the individual and his/her contacts[3]. Unlike the other email corpora, the email datasets contained within this corpus span a long enough time interval to capture long-term changes in behavior (i.e. moving between countries, changing universities, etc).

Enron Corpus

The Enron email corpus refers to the email histories of approximately 150 Enron employees, made publicly available as a result of the investigation by the Federal Energy Regulatory Commission (Klimt and Yang, 2004; Shetty and Adibi, 2004)[4]. Like the Gmail corpus, all email activity for the employees was recovered over the course of approximately one to two years. The Enron corpus is unique in that it is one of the few available email datasets that provides complete data — from the original email headers to the text and attachments within the emails. As such, it has become a frequently-used testbed for evaluating and comparing models in the fields of email communication and social networking.

[2] This individual's activity was downloaded using an early version of the Python script, primarily for the experiments in Chapter 5. Due to an inability to re-download the email activity, results regarding this individual is unavailable for the experiments regarding response models in Chapter 6.

[3] The "# Contacts" column in Table 1.1 only considers contacts that appear in 5+ emails.

[4] Enron corpus and additional details available as of this writing at `http://www.cs.cmu.edu/~./enron/`.

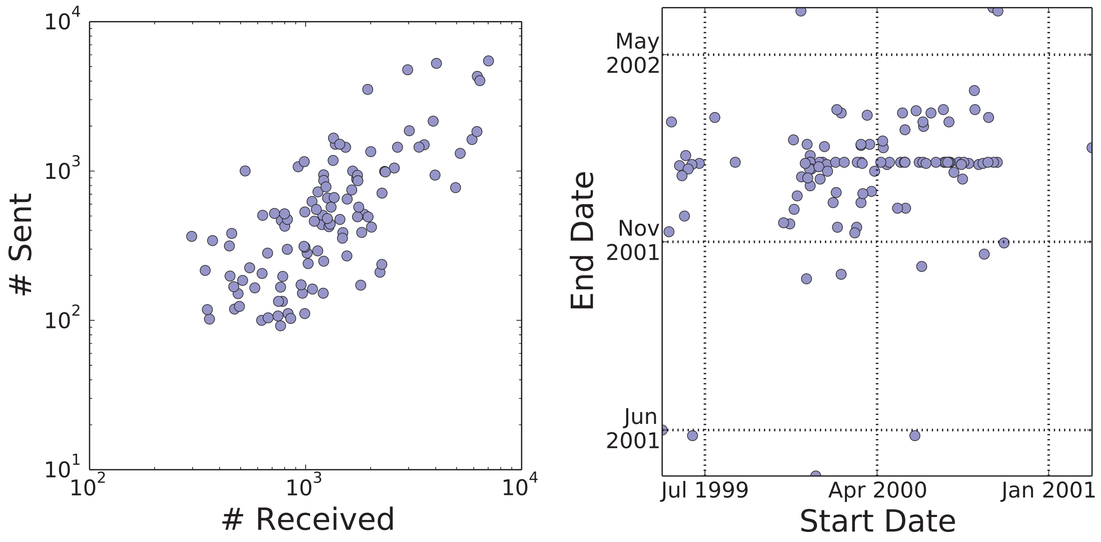


Figure 1.4: Summary statistics for email datasets in the Enron corpus. Left: number of emails received and sent (on log-log scale). Right: start and end dates of the datasets.

After preprocessing the email datasets to remove unusual accounts (explained in detail in Appendix A), 109 Enron employees remained in the corpus. Statistics over the number of emails received/sent and the time intervals of each inbox are shown in Figure 1.4.

European University Corpus

The email datasets contained within this corpus were collected by Eckmann et al. (2004) using the log files of an email server from an unknown European university. We will refer to this corpus not only as the European University corpus, but also the "Eckmann" email corpus.

Email activity in the form of {timestamp, sender, recipients} observations from the email server over the course of 83 days was collected, between the months of March to June — presumably the duration of the Spring quarter for the university. We believe the collected data is from 2003. While the complete activity consists of email interactions between thousands of students and thousands more outside the university, many individuals experienced unusual

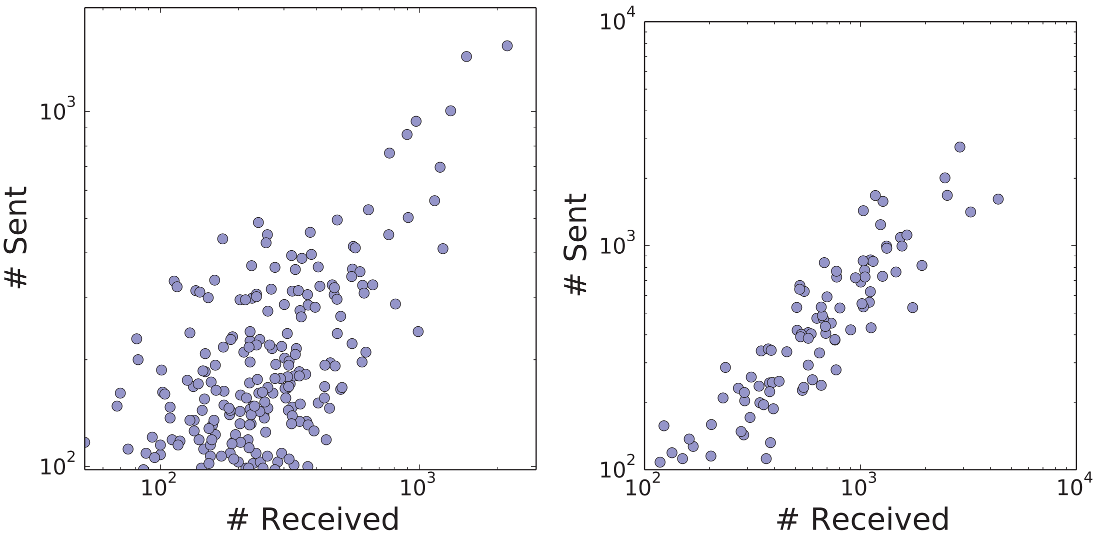


Figure 1.5: Number of emails received and sent (on log-log scale) for email datasets in the Eckmann (left) and Manufacturing (right) corpora.

activity (e.g. accounts which only sent or received emails, such as mailing lists) and/or had insufficient data[5]. After the preprocessing procedure discussed in Appendix A, 213 email datasets remain. The left plot of Figure 1.5 shows the number of emails that were received and sent for each of the 213 email datasets.

Manufacturing Company Corpus

The last email corpus we examine is referred to as the Manufacturing corpus, and contains email activity across 167 employees at a manufacturing company between the months of January and September of 2010 (Michalski et al., 2011)[6]. As with the Eckmann corpus, data was collected in the form of {timestamp, sender, recipients} observations — no subject or body text was collected. After preprocessing the email datasets, the email activity of 93 individuals remain. The right plot of Figure 1.5 shows the number of emails that were received and sent for each individual.

[5] A more precise definition of “unusual” and “insufficient” activity is given in Section A.2.

[6] Data available, as of this writing, at `http://konect.uni-koblenz.de/networks/radoslaw_email`

1.2 Dissertation Outline and Contributions

The structure of the dissertation, including novel contributions, is as follows:

Chapter 2 provides general background information relevant to the models and applications described throughout the dissertation. Generative probabilistic models such as the Bernoulli mixture model (BMM) and the Poisson process are discussed, and different methods of inferring parameter values from observed data are described.

Chapter 3 introduces a two-parameter kernel density estimator over multidimensional binary data. A stochastic gradient approach for estimating the bandwidth parameters of the kernel in the presence of a streaming source of data is proposed.

Contributions of Chapter 3 include:

- A two-parameter extension of the kernel density estimator proposed in Aitchison and Aitken (1976) is provided. It is proven that the two-parameter variant provides a better fit to binary data, with respect to its sparsity structure, both theoretically and empirically in terms of log-likelihood on held-out data.
- A parameterization of the kernel model, which can be viewed as a special case of the Bernoulli mixture model, is provided. Empirical results show that the kernel model provides a better fit to binary data compared to traditional BMMs.
- A stochastic gradient approach for estimating the bandwidth parameters is derived. A comprehensive experiment over many email datasets shows that the quality of parameter estimates is as good as, if not better than, traditional gradient and cross-validation approaches. The stochastic gradient approach also has the advantage of having a significantly reduced time complexity.

Chapter 4 applies both the Bernoulli mixture model and the kernel model of Chapter 3 to the task of itemset completion, where the goal is to predict which additional items will be added to a partially observed itemset. Two approaches to solving this problem are discussed — finding the most-probable configuration of items to be added to the itemset, and ranking the possible items based on their likelihood of being included in the itemset.

Contributions of Chapter 4 include:

- A method for analytically reducing the dimensionality of the problem, by approximating the values of dimensions satisfying certain criteria derived from the form of the Bernoulli mixture model. Results over many email datasets show that a significant fraction of dimensions can be analytically removed, dramatically reducing the problem's dimensionality.
- A comparison between various approximation methods for the most-probable configuration is given. Results show that similar estimates, if not better, are achieved when applying the methods in the reduced dimensionality after certain dimensions are analytically approximated. Results also give insight to the shape and structure the probability density of the Bernoulli mixture model.
- A ranking experiment is applied to the BMM, kernel model, and a baseline ranking model, where the task is to predict the next recipient to be added to an email. Results indicates that both the kernel model and BMM produce more accurate rankings than the baseline model, with the kernel model achieving slightly higher accuracy.

Chapter 5 addresses the problem of clustering recipients of an individual's sent emails, taking into account the time-varying behavior between the individual and the different clusters or groups of people. A joint model is specific over 1) the grouping of recipients, and 2) the frequency of emails sent from the individual to the different groups. Both aspects of the model are simultaneously learned using MCMC methods over observed data.

Contributions of Chapter 5 include:

- A model where changepoints (i.e. points in time when a behavioral change occurs) are modeled as latent state transitions of a hidden Markov model. This model is a non-parametric version of that proposed by Chib (1998), allowing for an arbitrary number of changepoints to be found.
- A model that accounts for the user's email activity towards a group of people as a piecewise-constant Poisson process, modulated by the changepoint model.
- Models of groups of email recipients using independent Bernoulli models. The groups are connected together using a mixed membership model, dependent on the Poisson process rates associated with each group.
- An MCMC approach for approximating the posterior distribution of model parameters, given observed data. Results using simulated data show a high accuracy in the estimation of model parameters, with a qualitative analysis over real email datasets revealing intuitive interpretations of estimated parameters.
- Quantitative experiments over email datasets that span multiple years indicate that the model is able to find email recipient groups which are more coherent than those found by baseline models. Additionally, the proposed model has superior predictive power to simpler baseline models.

Chapter 6 describes a method for modeling the time it takes for an individual to respond to a general communication event (e.g. a text message or email), while taking into account the strong circadian and weekly patterns the individual experiences. Standard probability distributions, in addition to Hawkes processes, are considered for modeling the response time. Additionally, the number of responses to the general communication event is modeled.

Contributions of Chapter 6 include:

- The concept of modeling the response time to an event as a transformation of *effective response time* is introduced. We define the effective response time as the integration between two points in time with respect to an *activity function* which parameterizes the individual's circadian and weekly patterns. The method is general in that it can be applied to any model or distribution over event response times.

- The time transformation technique is applied to two different models of response times: direct response time models and Hawkes processes, with different distributional forms of each. Results across 26 email datasets show a significant increase in both held-out log-likelihood and predictive power when directly accounting for circadian and weekly patterns via time transformation.

- We model the number of responses to an event as a mixture model between a logistic regression model and a geometric distribution. Results across 26 email datasets show that the logistic regression model is able to accurately predict which events will elicit a response based on metadata (e.g. the sender of the email, the number of recipients, etc), and that the geometric distribution is an appropriate model when conditioned on at least one response to the event.

Chapter 7 concludes the dissertation with a short discussion.

Chapter 2

Background

In this chapter, a brief introduction to the theory relevant to the work in this dissertation is provided. We review generative probabilistic models, identifying specific examples such as (mixtures of) independent Bernoulli distributions and Poisson processes. We then discuss some standard methods for inferring model parameters from observed data.

2.1 Generative Probabilistic Models

A *random variable* is a measure of some quantity of interest that varies according to a (usually unknown and complicated) non-deterministic process. A simple example is a coin flip; many factors affect its outcome (physical force of the flip, the type of ground the coin lands on, imperfections of the coin changing its weight and shape, etc). Probabilistic models associate each possible outcome with a nonnegative probability of that outcome occurring, thus approximating the unknown process that generates that outcome.

Let X be a set of random variables we are interested in modeling, $\mathbb{X}$ the support of X, and $x \in \mathbb{X}$ an instantiation of those random variables. A probabilistic model of X with parameters Ω places a nonnegative probability of observing $X = x$ for all $x \in \mathbb{X}$, such that $\int_{x \in \mathbb{X}} P(X = x|\Omega) = 1$. The model is considered to be a *generative* model if it is also able to produce random samples of $x \in \mathbb{X}$, thus describing a mathematical process of generating x.

As an example, consider the following scenario. There are two bags, bag one containing n_r red marbles and n_g green marbles, and bag two containing m_r red marbles and m_g green marbles. A coin is flipped; if the coin lands on heads (resp. tails), a single marble is blindly chosen from bag one (resp. two). Let F represent the coin flip ($f \in \{\text{heads, tails}\}$), and C represent the color of the chosen marble ($c \in \{\text{red, green}\}$). A potential probabilistic model describing the outcome $\{f, c\}$ may have the following form:

$$P(f = \text{heads}|\Omega) = \rho \qquad\qquad P(c = \text{red}|f, \Omega) = \begin{cases} \frac{n_r}{n_r+n_g} & \text{if } f = \text{heads} \\ \frac{m_r}{m_r+m_g} & \text{if } f = \text{tails} \end{cases}$$

The parameters of such a model are $\Omega = \{\rho, n_r, n_g, m_r, m_g\}$. This model uses the chain rule to rewrite $P(F = f, C = c|\Omega)$ as $P(F = f|\Omega)P(C = c|F = f, \Omega)$ and places a functional form over each of the two components. The model is generative in that the outcomes of F and C can be repeatedly simulated. First, F is simulated by drawing from a Bernoulli distribution with parameter ρ, then C is simulated by drawing from a Bernoulli distribution with parameter $\frac{n_r}{n_r+n_g}$ or $\frac{m_r}{m_r+m_g}$ (depending on the coin flip).

2.1.1 Independent Bernoulli Distribution

The first model discussed is an independent Bernoulli distribution. Let $\boldsymbol{x} \equiv \{x_i : 1 \leq i \leq N\}$, where $x_i \in \{0, 1\}^D$ is a D-dimensional binary vector and $x_{id} \in \{0, 1\}$ the value of the dth dimension in x_i. In this section, we will focus on the modeling of x_i. The interpretation of x_i is typically a collection of items, or an *itemset*, where $x_{id} = 1$ if and only if item d is part of the collection. Some examples of itemsets are shown in Table 2.1.

Data Type	**What is an itemset**	**What $x_{id} = 1$ implies**
Email	Recipients of an email	Person d received email i
Medical Diagnosis	Collection of symptoms	Patient i experienced symptom d
Consumer Spending	Items purchased	Customer i purchased item d
Attendance	People attending an event	Person d went to event i
Cinema	Actors in a film	Person d appeared in film i
Multi-labeled Documents	Labels of a document	Document i contains label d

Table 2.1: Examples of itemsets.

There are 2^D possible values of x_i, making it infeasible to manually specify $P(x_i|\Omega)$ for each value. The independent Bernoulli distribution factorizes $P(x_i|\Omega)$ in the following way:

$$P(x_i|\Omega) = \prod_{d=1}^{D} P(x_{id}|\theta_d) = \prod_{d=1}^{D} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \tag{2.1}$$

where $\Omega = \{\theta\}$ and $\theta \in [0,1]^D$ is a vector where θ_d is the probability that $x_{id} = 1$. This is referred to as an independent Bernoulli model because, under the generative process, $x_{id}|\theta_d \sim \text{Bernoulli}(\theta_d)$ and each x_{id} is generated independently of $\{x_{id'} : d' \neq d\}$.

Marginal and Conditional Distributions

Let $A \in \{1, \cdots, D\}$ be a set of indices, and $x_{iA} = \{x_{id} : d \in A\}$ a $|A|$-dimensional vector representing the corresponding dimensions in x_i. Additionally, let $\tilde{A} \equiv \{1, \cdots, D\} \setminus A$ represent the indices that are not in A, such that $x_i = x_{iA} \cup x_{i\tilde{A}}$. The $|A|$-variate marginal distribution of x_{iA} is defined as

$$\begin{aligned} P(x_{iA}|\theta) &= \sum_{x_{i\tilde{A}}} P(x_i|\theta) = \sum_{x_{i\tilde{A}}} \left[\prod_{d=1}^{D} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \right] \\ &= \left[\sum_{x_{i\tilde{A}}} \prod_{d\in\tilde{A}} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \right] \prod_{d\in A} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \\ &= \prod_{d\in A} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \end{aligned} \tag{2.2}$$

where we make use of the fact that $\sum_{x_{i\tilde{A}}} \prod_{d \in \tilde{A}} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} = 1$.

The $|\tilde{A}|$-variate conditional distribution of $x_{i\tilde{A}}$ given x_{iA} is defined as

$$\begin{aligned} P(x_{i\tilde{A}}|x_{iA}, \theta) &= \frac{P(x_{i\tilde{A}} \cup x_{iA}|\theta)}{P(x_{iA}|\theta)} = \frac{\prod_{d=1}^{D} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}}}{\prod_{d \in A} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}}} = \prod_{d \in \tilde{A}} {\theta_d}^{x_{id}}(1-\theta_d)^{1-x_{id}} \\ &= P(x_{i\tilde{A}}|\theta) \end{aligned} \tag{2.3}$$

which, as the name of the model suggests, is independent of x_{iA}.

2.1.2 Bernoulli Mixture Models

The independence assumptions of the model in Section 2.1.1 are very strong — typically the dimensions of observed binary vectors exhibit significant dependencies or correlations. For instance, the recipients of an email are usually related in some way with respect to the email sender (e.g. coworkers, family members), or certain symptoms of diseases typically co-appear together (e.g. coughing and sneezing). Suppose we are interested in answering queries of the form "given $x_{id} = 1$, what is the probability that $x_{id'} = 1$ for some $d' \neq d$?" We would expect the probability of $x_{id'} = 1$ to change based on the relationship between dimensions d' and d. For example, if an individual sent an email to a family member, it is reasonable to assume that other recipients of the email would include other family members, as opposed to co-workers.

The independent Bernoulli distribution is unable to capture such dependencies, as shown in its conditional distribution in Equation 2.3. One model that is able to introduce such dependencies is the Bernoulli mixture model (also referred to as the "BMM"), first introduced by Duda and Hart (1973). The BMM has the following probability density over values of x_i:

$$P(x_i|\Omega) = P(x_i|\pi, \theta) = \sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} \theta_{kd}{}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \tag{2.4}$$

where $\Omega = \{\pi, \theta\}$, $\pi \in [0,1]^K$ are the *mixing proportions* (with $\sum_{k=1}^{K} \pi_k = 1$), and $\theta \in [0,1]^{K \times R}$ is now a matrix of Bernoulli probabilities. The generative process associated with this model assumes that x is generated from exactly one of the K *mixing components*. Let $z_i \in \{1, \cdots, K\}$ represent this component, thus π_k is interpreted as the probability that $z_i = k$. If z_i was observed with x_i, the joint probability over (z_i, x_i) would be

$$P(z_i = k, x_i|\pi, \theta) = P(z_i = k|\pi) P(x_i|z_i = k, \theta) = \pi_k \prod_{d=1}^{D} \theta_{kd}{}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}}$$

This joint density is the probability that $z_i = k$ times the probability of x_i under the kth component. However, the variable z_i is *latent* or unobserved, thus we must marginalize over all possible values of z_i to derive the probability distribution in Equation 2.4:

$$P(x_i|\pi, \theta) = \sum_{k=1}^{K} P(z_i = k, x_i|\pi, \theta) = \sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} \theta_{kd}{}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}}$$

Figure 2.1 illustrates an example of what θ may look like if the BMM is modeling recipients of an email sent by an individual (assuming $K = 3$ mixtures or groups). For this application, θ_{kd} is the probability that person d receives the email, assuming it was sent to group k. The three groups portrayed in Figure 2.1 have clear interpretations: they represent the individual's family members (top plot), coworkers (middle plot), and friends (bottom plot). Note that the groups in a BMM do not have to partition the possible email recipients; this model is a *mixed membership model* where individuals are allowed to be in multiple groups (e.g. a friend may also be a coworker, such as the left-most friend in Figure 2.1).

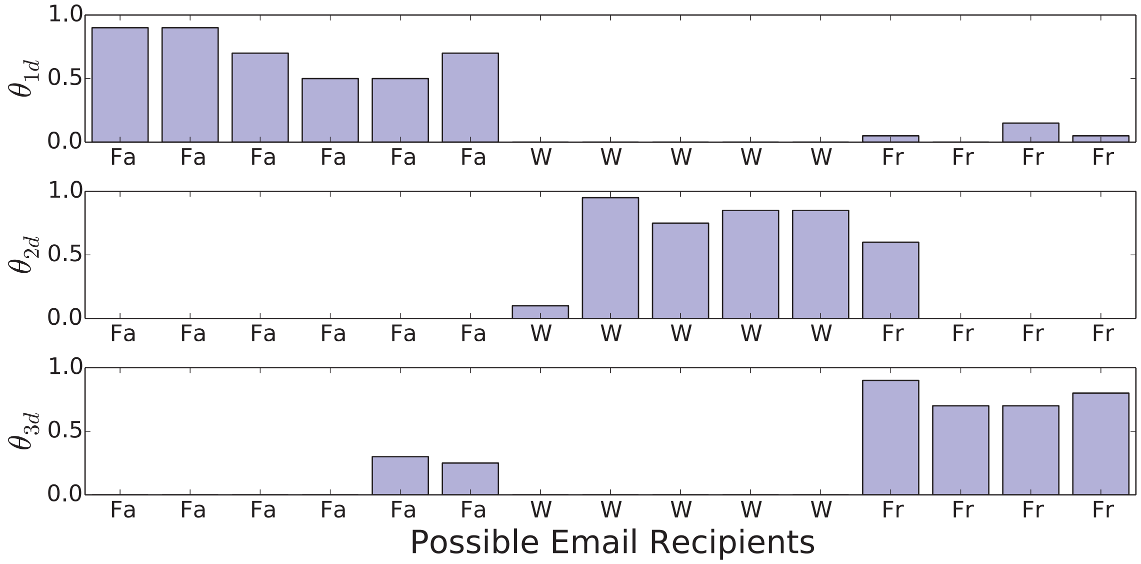


Figure 2.1: An example parameterization of θ for the Bernoulli mixture model. The x-axis represents people (Fa = family, W = work, Fr = friend), and y-axis represents the value of θ_{kd} for each person within each group.

Suppose that, in addition to θ defined as in Figure 2.1, the parameter $\pi = \{0.5, 0.4, 0.1\}$. Under the generative process, this means that 50% of emails are sent to family members, 40% to coworkers, and 10% to friends. Then, conditioned on the group k that will receive an email, each person d is added to that email with probability θ_{kd}.

Although the dimensions of x_i are independent within each of the K clusters, they become dependent once the latent variable z_i is marginalized (as will be shown in its conditional distribution). A common application of the Bernoulli mixture model is to have each component represent a cohesive group of items that often co-occur together, as shown in Figure 2.1. The BMM has been highly effective at modeling groups of email recipients (Navaroli et al., 2013), topics within text data (Juan and Vidal, 2002), DNA amplification patterns (Tikka et al., 2007), and digits in binary images (Juan and Vidal, 2004).

Marginal and Conditional Distributions

The $|A|$-variate marginal distribution $P(x_{iA}|\pi, \theta)$ of a BMM is

$$\begin{aligned}
P(x_{iA}|\pi, \theta) &= \sum_{x_{i\tilde{A}}} P(x_i|\pi, \theta) = \sum_{x_{i\tilde{A}}} \sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} {\theta_d}^{x_{id}} (1 - \theta_d)^{1-x_{id}} \\
&= \sum_{k=1}^{K} \pi_k \prod_{d \in A} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \left[\sum_{x_{i\tilde{A}}} \prod_{d \in \tilde{A}} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \right] \\
&= \sum_{k=1}^{K} \pi_k \prod_{d \in A} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \qquad (2.5)
\end{aligned}$$

which is a BMM where the mixing proportions π are unchanged, and the columns in θ corresponding to $\tilde{A}$ are ignored.

The $|\tilde{A}|$-variate conditional distribution $P(x_{i\tilde{A}}|x_{iA}, \pi, \theta)$ of a BMM is

$$\begin{aligned}
P(x_{i\tilde{A}}|x_{iA}, \pi, \theta) &= \frac{P(x_{i\tilde{A}} \cup x_{iA}|\pi, \theta)}{P(x_{iA}|\pi, \theta)} = \frac{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}}}{\sum_{k'=1}^{K} \pi_{k'} \prod_{d \in A} {\theta_{k'd}}^{x_{id}} (1 - \theta_{k'd})^{1-x_{id}}} \\
&= \sum_{k=1}^{K} \left[\frac{\pi_k \prod_{d \in A} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}}}{\sum_{k'=1}^{K} \pi_{k'} \prod_{d \in A} {\theta_{k'd}}^{x_{id}} (1 - \theta_{k'd})^{1-x_{id}}} \right] \prod_{d \in \tilde{A}} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \\
&= \sum_{k=1}^{K} \pi_k^{'} \prod_{d \in \tilde{A}} {\theta_{kd}}^{x_{id}} (1 - \theta_{kd})^{1-x_{id}} \qquad (2.6)
\end{aligned}$$

where $\pi_k^{'} \equiv \frac{\pi_k \prod_{d \in A} {\theta_{kd}}^{x_{id}} (1-\theta_{kd})^{1-x_{id}}}{\sum_{k'=1}^{K} \pi_{k'} \prod_{d \in A} {\theta_{k'd}}^{x_{id}} (1-\theta_{k'd})^{1-x_{id}}} = P(z_i = k|x_{iA}, \pi, \theta)$ is the posterior probability of $z_i = k$ having observed x_{iA}. Under the conditional distribution, the mixing proportions are reweighted such that more weight is given to clusters that “agree” with the configuration of x_{iA}. Thus, we can rewrite $P(x_{i\tilde{A}}|x_{iA}, \pi, \theta)$ as $P(x_{i\tilde{A}}|\pi', \theta')$, a BMM with mixing proportions $\pi^{'}$ and probability matrix θ' (equivalent to θ, with the columns corresponding to A ignored).

Figure 2.2 shows an example of how the mixing proportions may change when conditioning on x_{iA}, assuming θ is defined as in Figure 2.1 and $\pi = \{0.50, 0.40, 0.10\}$ (in blue). Suppose

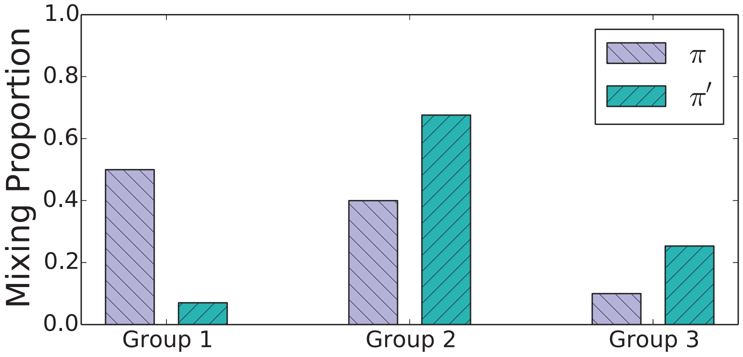


Figure 2.2: Reweighted mixing proportions π' (green, with up-right crosshatches) after conditioning on the fact that the left-most friend from Figure 2.1 received an email. The original proportions π are shown in blue, with down-right crosshatches.

we condition on $x_{id} = 1$, where person d represents the left-most friend in Figure 2.1. Under this condition, the mixing proportion for Group 1 (the family group) significantly decreases. This is because the probability that the person d received the email, conditioned on the email being sent to family, is close to 0 (as seen in Figure 2.1). Therefore, the email is most likely not being sent to family. The other two groups have their mixing proportions increased, with the increase for Group 2 (the coworker group) larger due to the relative values of π_k and θ_{kd} compared to Group 3 (the friend group).

Although the Bernoulli probabilities in θ are unchanged when conditioning on x_{id}, the probability that $x_{id'} = 1$ for $d' \neq d$ under the conditional distribution does change. To see this, consider the conditional probability $P(x_{id'} = 1|x_{id} = 1, \pi, \theta)$:

$$\begin{aligned} P(x_{id'} = 1|x_{id} = 1, \pi, \theta) = \frac{P(x_{id'} = 1, x_{id} = 1|\pi, \theta)}{P(x_{id} = 1|\pi, \theta)} &= \frac{\sum_{k=1}^{K} \pi_k \theta_{kd'} \theta_{kd}}{\sum_{k'=1}^{K} \pi_{k'} \theta_{k'd}} \\ &= \sum_{k=1}^{K} \left[\frac{\pi_k \theta_{kd}}{\sum_{k'=1}^{K} \pi_{k'} \theta_{k'd}} \right] \theta_{kd'} \end{aligned}$$

If the value of $x_{id'}$ was independent of x_{id}, then $P(x_{id'} = 1|x_{id} = 1, \pi, \theta)$ would be equal to $P(x_{id'} = 1|\pi, \theta)$. However, using the form of the marginal distribution in Equation 2.5, it is straightforward to show that this is generally not the case:

$$P(x_{id'} = 1|\pi, \theta) = \sum_{k=1}^{K} \pi_k \theta_{kd'} \neq \sum_{k=1}^{K} \left[\frac{\pi_k \theta_{kd}}{\sum_{k'=1}^{K} \pi_{k'} \theta_{k'd}} \right] \theta_{kd'} = P(x_{id'} = 1|x_{id} = 1, \pi, \theta)$$

Thus, under the Bernoulli mixture model, the values of the variates in x_i are dependent of one another.

2.1.3 Poisson Processes

The Poisson process is a stochastic process that models the rate at which events or observations occur over time. Unlike the Bernoulli mixture model, this is a model over positive values representing the timestamps of events. For example, the Poisson process may describe the rate at which

- customers arrive at or leave from a store
- Internet packets are processed on a server
- emails are sent from an individual to others

Here we give a brief introduction to the formulation of a Poisson process. For an in-depth introduction and overview of Poisson process, see Ross (2006, Sections 5.3 and 5.4).

Let $\boldsymbol{t} = \{t_i : 1 \leq i \leq M\}$ represent a sorted sequence of timestamps, where $t_i \geq 0$ is the timestamp the ith event occurred. The corresponding count process is defined as a function $N : \mathbb{R}^+ \rightarrow \mathbb{N}$, where $N(t)$ represents the number of events that have occurred up to time t. Formally, $N(t) = |\{t_i : t_i \leq t\}|$, with $N(0)$ defined as 0. The quantity $N(t) - N(s)$ for $t > s \geq 0$ is the number of events that occurred on the time interval $[s, t]$. If $N(t)$ is modeled as a Poisson process, then the probability that a single event occurs in an infinitely small window of time δt is defined as

$$P(N(t + \delta t) - N(t) = 1) = \lambda(t)\delta t$$

where $\lambda(t)$ is the parameter of the Poisson process, describing the rate at which an event will occur at time t (Daley and Vere-Jones, 2003). The exact form of $\lambda(t)$ can vary based on its application. In the simplest case, $\lambda(t) = \lambda$ is a constant value; such a model is referred to as a *homogeneous* Poisson process. For the purposes of this dissertation, we will only consider *non-homogeneous* Poisson processes where $\lambda(t)$ is a function of time. Most commonly, $\lambda(t)$ is either modeled as piecewise-constant or as a continuous function over time. By allowing the event rate to change over time, temporal changes in the nature of the event-generating mechanism can be modeled. For example, if modeling the rate at which one sends emails, a reasonable event rate $\lambda(t)$ would be one where the rate is higher during the daytime hours, where the person is presumably more likely to write an email.

For a Poisson process, the number of events that occur within the time interval $[s, t]$ where $t > s \geq 0$ is modeled as

$$N(t) - N(s) \sim \text{Poisson}\left(\int_s^t \lambda(u)du\right)$$

where $\int_s^t \lambda(u)du$ is referred to as the *mean value function* of the Poisson process across the interval $[s, t]$ (Ross, 2006, Section 5.4).

The probability of observing $\boldsymbol{t}$, the timestamps of all events, for a Poisson process is

$$P(\boldsymbol{t}|\lambda) = \left[\prod_{i=1}^{M} \lambda(t_i)\right] \exp\left(-\int_0^T \lambda(u)du\right)$$

where it is assumed that the events in $\boldsymbol{t}$ are observed over the time interval $[0, T]$ (Daley and Vere-Jones, 2003).

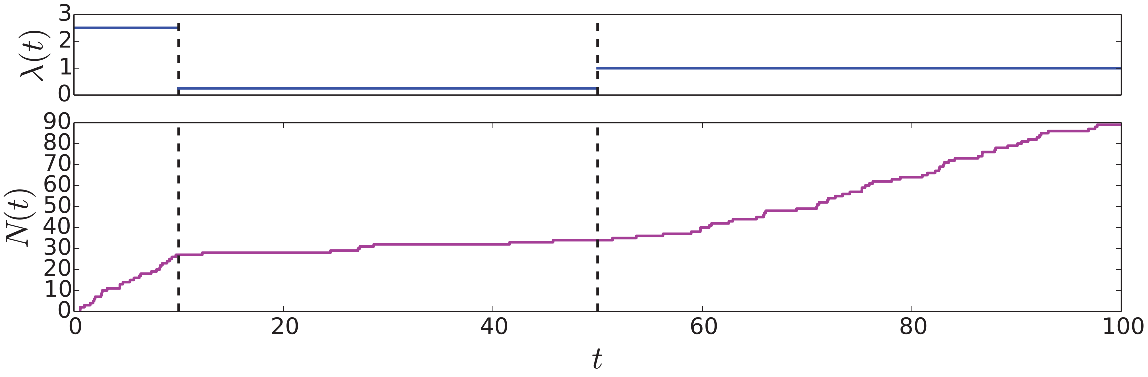


Figure 2.3: Example of a non-homogeneous Poisson process with a piecewise-constant rate.

The Poisson process is a generative model in that the timestamps of events $\boldsymbol{t}$ can be simulated given a functional form of $\lambda(t)$. This is typically accomplished by sampling the *inter-event times* between consecutive events (e.g. $t_k - t_{k-1}$). For homogeneous Poisson processes where $\lambda(t) = \lambda$, inter-event times follow an exponential(λ) distribution, thus they can be simulated by drawing from this distribution. For the general case where $\lambda(t)$ varies over time, it is straightforward to simulate inter-event times by time sampling a homogeneous Poisson process where $\lambda \geq \max_t \lambda(t)$ (Ross, 2006, Section 5.4). An example of what a simulated count process $N(t)$ may look like given a simple piecewise-constant parameterization of $\lambda(t)$ is given in Figure 2.3.

2.2 Parameter Inference: Point Estimates

Suppose we have specified a probabilistic model with parameters Ω that describe the characteristics of a random variable X. Additionally, we observe a set of N instantiations of X: $\boldsymbol{x} \equiv \{x_1, x_2, \cdots, x_N\}$, and that each x_i is *independent and identically distributed* (iid). This means that the value of x_i is independent of the other values $\{x_j : j \neq i\}$, and that all values were generated from the same model with the same parameter value of Ω. The likelihood of data $\boldsymbol{x}$ under the model parameterized by Ω is

$$P(\boldsymbol{x}|\Omega) = \prod_{i=1}^{N} P(x_i|\Omega)$$

This section focuses on producing *point estimates* for the parameters in Ω that explain the observed data $\boldsymbol{x}$. That is, a single value is assigned to each parameter, usually optimizing an objective function (e.g. the log of the above likelihood).

2.2.1 Maximum Likelihood Estimation

Maximum-likelihood (ML) estimation is an inference procedure that, given observed data $\boldsymbol{x}$, produces an estimate of parameter values $\hat{\Omega}$ that maximizes the observed log-likelihood:

$$\hat{\Omega} = \arg\max_{\Omega} \log P(\boldsymbol{x}|\Omega) = \arg\max_{\Omega} \sum_{i=1}^{N} \log P(x_i|\Omega)$$

As an example, suppose we are modeling a positive number $x > 0$ with an exponential distribution, which takes the following form:

$$P(x|\Omega) = \lambda e^{-\lambda x}$$

where $\Omega = \{\lambda\}$, with $\lambda > 0$ a model parameter. The log-likelihood of data $\boldsymbol{x}$ given a value of λ can be written as

$$\log P(\boldsymbol{x}|\Omega) = \sum_{i=1}^{N} \log P(x_i|\Omega) = \sum_{i=1}^{N} (\log \lambda - \lambda x_i) = N \log \lambda - \lambda \sum_{i=1}^{N} x_i$$

The ML estimate of λ can be found by setting the derivative of the above equation to zero and solving for λ. The derivative is

$$\frac{\partial \log P(\boldsymbol{x}|\Omega)}{\partial \lambda} = \frac{N}{\lambda} - \sum_{i=1}^{N} x_i$$

and setting it to zero leads to the ML estimate $\hat{\lambda} = \frac{N}{\sum_{i=1}^{N} x_i}$, the inverse of the empirical average. This is guaranteed to be the global maximum, as the second derivative is negative:

$$\frac{\partial^2 \log P(\boldsymbol{x}|\Omega)}{\partial \lambda^2} = -\frac{N}{\lambda^2} < 0$$

The maximum-likelihood technique can be applied to a wide range of distributions over $\boldsymbol{x}$ and its parameters Ω. In some cases, while the derivative may have an analytical form, there may not be a closed-form solution when setting the derivative to zero and solving for Ω (as is the case for ML estimation of parameters for the Gamma distribution). In these cases, numerical methods (e.g. Newton's method, gradient ascent, etc) can be applied in order to find the (potentially local) optimum value of Ω.

For the Bernoulli mixture model in Section 2.1.2, ML estimates cannot be directly computed; the derivative of the log-likelihood is complex due to a log function over a summation of terms. This is a common problem for mixture models as the latent variable (the component assignment for a BMM) is marginalized. One method for obtaining ML estimates for these types of models is the EM algorithm (Dempster et al., 1977), an iterative algorithm which maximizes a lower-bound of the log-likelihood. This algorithm is guaranteed to find a configuration of model parameters that locally maximizes the log-likelihood function (Bishop, 2006, Section 9.3).

2.2.2 Maximum a Posteriori Estimation

A potential problem with the maximum-likelihood estimation procedure is that it can produce unreliable estimates when there are few data. A solution to this is to take a Bayesian approach and place a *prior distribution* over the possible values of Ω. The *posterior distribution* of Ω given the observed data $\boldsymbol{x}$ is defined as

$$P(\Omega|\boldsymbol{x}) = \frac{P(\Omega, \boldsymbol{x})}{P(\boldsymbol{x})} = \frac{P(\Omega)P(\boldsymbol{x}|\Omega)}{\int_\Omega P(\Omega)P(\boldsymbol{x}|\Omega)d\Omega} \tag{2.7}$$

The maximum a posteriori (MAP) estimate of Ω is defined as the value of Ω which maximizes $P(\Omega|\boldsymbol{x})$, or equivalently $P(\Omega)P(\boldsymbol{x}|\Omega)$ (since $P(\boldsymbol{x})$ is not a function of Ω):

$$\hat{\Omega} = \arg\max_\Omega \log P(\Omega|\boldsymbol{x}) = \arg\max_\Omega \left(\log P(\Omega) + \sum_{i=1}^N \log P(x_i|\Omega) \right)$$

Going back to the exponential distribution in the previous section, suppose we place a Gamma prior distribution over the values of λ:

$$P(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda}$$

where $\alpha > 0$ and $\beta > 0$ are parameters of the Gamma distribution, and Γ the Gamma function. The MAP estimate of λ will then maximize

$$\begin{aligned} \log P(\Omega|\boldsymbol{x}) &\propto \log P(\lambda) + \sum_{i=1}^N \log P(x_i|\lambda) \\ &\propto (\alpha - 1)\log\lambda - \lambda\beta + N\log\lambda - \lambda\sum_{i=1}^N x_i \end{aligned}$$

As with the ML estimate, the MAP estimate can be solved by taking the derivative of the above equation, setting it to zero, and solving for λ. In this case, the MAP estimate is

$$\hat{\lambda} = \frac{N + \alpha - 1}{\sum_{i=1}^{N} x_i + \beta}, \tag{2.8}$$

which looks like the ML estimate, but "smoothed" by the values of α and β. The strength or influence the prior distribution has on $\hat{\lambda}$ depends on the relative values of α, β, and N. Figure 2.4 shows what the posterior distribution $P(\lambda|\boldsymbol{x})$ may look like under different scenarios.

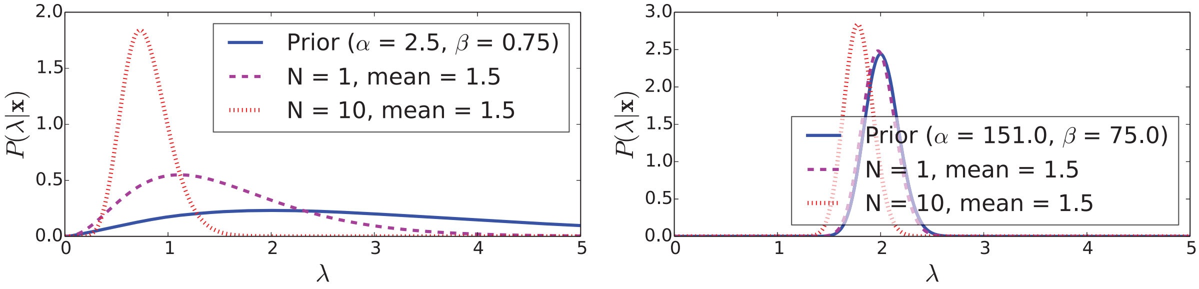


Figure 2.4: Posterior distributions for the exponential parameter λ, using two different Gamma priors (each with a mode at $\lambda = 2.0$).

In this figure, two different Gamma priors over λ are considered, each with a mode at $\lambda = \frac{\alpha-1}{\beta} = 2.0$. If $N = 0$ (no data is observed), then the MAP estimate of λ becomes the mode of the prior distribution (the peak of the blue curve at $\lambda = 2.0$). Suppose $N = 1$ with $x_1 = 1.5$; the ML estimate would be $\hat{\lambda} = \frac{1.0}{1.5}$. However, the MAP estimate from Equation 2.8 (the peaks of the magenta dashed curve) is biased towards 2.0. How much bias occurs depends on the values of α and β. When α and β are small (left plot), the posterior distribution is largely affected by x_1, pushing its mode closer towards $\frac{1.0}{1.5}$. This is considered a "weak" prior as only a few datapoints are needed to significantly change the shape of the posterior distribution. In contrast, when α and β are large (right plot), the posterior distribution is very similar to the prior — in this case, the MAP estimate is very close to mode of the prior at 2.0.

The dotted red curve in Figure 2.4 shows the posterior distribution $P(\lambda|\boldsymbol{x})$ when $N = 10$ and the empirical mean $\frac{1}{N}\sum_{i=1}^{N} x_i = 1.5$. The same ML estimate occurs at $\hat{\lambda} = \frac{1.0}{1.5}$. For the MAP estimate, the greater number of observed datapoints decreases the strength of the prior, allowing for estimates that are closer to the ML estimate.

This method of MAP estimation can be applied to any distribution over x where ML estimates can be produced, as long as a prior distribution over the values of Ω is specified. As with the Gamma prior in the previous example, most prior distributions have a relative "strength" determined by the values of their parameters. An appropriate strength of the prior to use during parameter inference often varies between applications.

2.3 Parameter Inference: Posterior Distributions

The inference procedures from Section 2.2 produce a single point estimate over the model parameters Ω. It is often desirable to instead analyze the full posterior distribution $P(\Omega|\boldsymbol{x})$, as opposed to a single quantity such as its mean or mode (e.g. MAP estimation). For example, properties such as the posterior variance and skewness (or higher moments of the distribution) cannot be obtained using the MAP estimation procedure — the entire distributional form of the posterior is needed.

2.3.1 Analytical form of Posterior Distribution

In some cases, the form of the posterior distribution $P(\Omega|\boldsymbol{x})$ can be calculated analytically. For example, when modeling x with the exponential distribution and a Gamma prior is placed over λ, the posterior distribution $P(\lambda|\boldsymbol{x})$ (illustrated in Figure 2.4) has the following form:

$$P(\lambda|\boldsymbol{x}) \propto P(\boldsymbol{x}|\lambda)P(\lambda) \propto \lambda^N \exp\left(-\lambda\sum_{i=1}^{N} x_i\right)\lambda^{\alpha-1}\exp\left(-\lambda\beta\right)$$
$$= \lambda^{N+\alpha-1}\exp\left(-\lambda\left(\sum_{i=1}^{N} x_i + \beta\right)\right)$$

which looks like a Gamma distribution with parameter values $\alpha' = N+\alpha$ and $\beta' = \sum_{i=1}^{N} x_i+\beta$. Indeed, when the proportionality constants and $P(\boldsymbol{x})$ (see Equation 2.7) are accounted for, the form of $P(\lambda|\boldsymbol{x})$ is exactly a Gamma distribution with parameters α' and β' — parameters which are a function of the prior parameters and statistics of the empirical data. In such cases, the prior is referred to as a *conjugate prior*, as the posterior distribution takes the same form as the prior distribution.

2.3.2 Approximating the Posterior Distribution with Samples

In some cases, the posterior distribution $P(\Omega|\boldsymbol{x})$ may not have an analytical form, or may only be known up to a normalization constant, e.g. $P(\Omega)P(\boldsymbol{x}|\Omega)$. The most typical case where this occurs is when using a non-conjugate prior over Ω, where $P(\Omega)P(\boldsymbol{x}|\Omega)$ can be evaluated, but the exact form of $P(\Omega|\boldsymbol{x})$ is unknown (as the quantity $P(\boldsymbol{x})$ from Equation 2.7 usually cannot be calculated in closed form).

When the posterior distribution cannot be analytically calculated, it is often approximated with a finite set of samples $S = \{s_i : 1 \leq i \leq M\}$, where s_i is an instantiation of Ω assumed to be simulated from $P(\Omega|\boldsymbol{x})$. Given the samples S, the expectation of any quantity $f(\Omega)$ under the posterior distribution $P(\Omega|\boldsymbol{x})$ can be estimated by using Monte Carlo integration:

$$E_{\Omega|\boldsymbol{x}}[f(\Omega)] = \int f(\Omega)P(\Omega|\boldsymbol{x})d\Omega \approx \frac{1}{M}\sum_{i=1}^{M} f(s_i)$$

For instance, if the goal is to estimate the model parameters using the mean of the posterior distribution, then $\hat{\Omega} = \frac{1}{M} \sum_{i=1}^{M} s_i$, the mean of the collected samples.

Samples from the posterior distribution can be obtained using a class of algorithms called Markov Chain Monte Carlo (MCMC) sampling (Hastings, 1970). These algorithms produce samples by iterating through a "random walk" over parameter space via a Markov chain whose stationary distribution is exactly the posterior distribution $P(\Omega|\boldsymbol{x})$. A plethora of algorithms exist for accomplishing this task, with different algorithms tailored to posterior distributions with different characteristics. Popular MCMC algorithms include the Metropolis-Hasting (Metropolis et al., 1953), Gibbs sampling (Geman and Geman, 1984), and Slice sampling (Neal, 2003) algorithms. Andrieu et al. (2003) provides a more in-depth tutorial on MCMC sampling algorithms.

Chapter 3

Kernel Density Estimators for Multidimensional Binary Data

Multidimensional binary data can be used to describe a vast range of applications. Often, such data is used to represent a collection of items, referred to as an *itemset*. As in Chapter 2, let $x_i \in \{0,1\}^D$ be a D-dimensional binary vector, where each dimension represents one of the D possible items available. Typically, a single itemset is represented with x_i, where $x_{id} = 1$ if and only if item d appears in itemset i. Some examples of itemsets that x_i can model were provided in Table 2.1.

In this chapter, we present a novel nonparametric probabilistic method for describing the patterns found within a collection of itemsets. The method uses kernel density estimation to generate a probability distribution over an observed collection of N itemsets, represented as $x = \{x_{id}\}$, with $1 \leq i \leq N$ and $1 \leq d \leq D$. Under this notation, $x_i \in \{0,1\}^D$ is a D-dimensional binary vector with $x_{id} \in \{0,1\}$ representing whether or not item d appeared in itemset i. The proposed kernel density estimator uses two bandwidth parameters for determining how probability mass is spread throughout the D-dimensional binary hypercube. We then review several inference procedures for estimating the bandwidth parameters, including a stochastic gradient method which can be applied when new data is observed in an online fashion (e.g. as emails are generated over time).

3.1 Background

This section gives an overview of previous work regarding kernel density estimation over multidimensional binary data, along with parameter inference procedures.

3.1.1 Discrete Kernel Densities

The idea of applying kernel density estimation over multidimensional binary data was first popularized by Aitchison and Aitken (1976), who proposed a probability distribution over $y \in \{0,1\}^D$ given historical data x as follows:

$$P(y|x,\lambda) \equiv \frac{1}{N}\sum_{i=1}^{N} K(y|x_i,\lambda) \tag{3.1}$$

$$K(y|x_i,\lambda) \equiv \lambda^{D-(x_i-y)^T(x_i-y)}(1-\lambda)^{(x_i-y)^T(x_i-y)} \tag{3.2}$$

where $\lambda \in [0,1]$ is the bandwidth parameter, and $(x_i - y)^T(x_i - y)$ is the Hamming distance between x_i and y[1]. Given x and λ, one can think of this model in a generative manner as follows:

1. Select an itemset x_i uniformly at random from the dataset.

[1] As the values of x_i are discrete, kernel models estimate a probability mass function, not a probability density. However, following the language in Aitchison and Aitken (1976) and general literature in kernel estimation, we will continue to refer to distributions over x_i as densities.

2. For each dimension d, set

$$y_d = \begin{cases} x_{id} & \text{with probability } \lambda \\ 1 - x_{id} & \text{with probability } 1 - \lambda \end{cases}$$

Thus, y can be interpreted as a noisy observation of one of the x_i, where each dimension (or "bit") in x_i is inverted with probability $1 - \lambda$. It is straightforward to show that the second step in the generative procedure is equivalent to the kernel function of Equation 3.2:

$$\begin{aligned} P(y|x_i, \lambda) &= \prod_{d=1}^{D} P(y_d|x_{id}, \lambda) = \prod_{d=1}^{D} \lambda^{|x_{id}-y_d|}(1-\lambda)^{1-|x_{id}-y_d|} \\ &= \lambda^{D-\sum_{d=1}^{D}|x_{id}-y_d|}(1-\lambda)^{\sum_{d=1}^{D}|x_{id}-y_d|} \\ &= K(y|x_i, \lambda) \end{aligned}$$

where $\sum_{d=1}^{D} |x_{id} - y_d| = (x_i - y)^T(x_i - y)$.

The particular x_i that y is a noisy observation of is unobserved, thus we must marginalize over all possible values of i:

$$P(y|x, \lambda) = \sum_{i=1}^{N} P(z = i)P(y|z = i, x, \lambda) = \sum_{i=1}^{N} \frac{1}{N} P(y|x_i, \lambda) = \frac{1}{N} \sum_{i=1}^{N} K(y|x_i, \lambda)$$

where $1 \leq z \leq N$ indicates the index of the datapoint used to generate y. This probability is equivalent to Equation 3.1.

When $\lambda = 1$, Equation 3.2 becomes a point mass at $y = x_i$, making the probability distribution in Equation 3.1 equivalent to a histogram over the unique itemsets in x. When $\lambda = \frac{1}{2}$, both Equations 3.1 and 3.2 become uniform distributions over $\{0, 1\}^D$. From a kernel density viewpoint, values of $\lambda < \frac{1}{2}$ are nonsensical; the mode of $K(y|x_i, \lambda)$ becomes such that $y_d = 1 - x_{id}$, the "inverted" value of x_i. Unless stated otherwise, it will be assumed from here onwards that $\lambda \geq \frac{1}{2}$.

As with traditional Gaussian kernels, the bandwidth parameter λ controls the variance of the kernel function in Equation 3.2. Due to the independence assumptions in the kernel function, the distribution of y_d given x_i is a Bernoulli distribution with parameter $\rho \equiv \lambda^{x_{id}}(1-\lambda)^{1-x_{id}}$ and variance $\rho(1-\rho) = \lambda(1-\lambda)$. As λ increases, the variance becomes smaller, and becomes 0 when $\lambda = 1$. This does not mean that the variance of y_d given *all* $\{x_i\}$ (as per Equation 3.1) is zero, only the variance of $P(y_d|x_i, \lambda)$ under the kernel function. In contrast, the variance is maximized when $\lambda = \frac{1}{2}$.

A variant of the kernel presented by Aitchison and Aitken (1976) is to have $\boldsymbol{\lambda} \in [0,1]^D$ be a D-dimensional vector, such that

$$K(y|x_i, \boldsymbol{\lambda}) \equiv \prod_{d=1}^{D} \lambda_d^{1-|x_{id}-y_d|}(1-\lambda_d)^{|x_{id}-y_d|} \tag{3.3}$$

An advantage of allowing $\boldsymbol{\lambda}$ to have a different value for each dimension is that different variances across dimensions can be accounted for (Titterington, 1980; Tutz and Grob, 1995). For example, the variance of $P(y_d|x_i, \boldsymbol{\lambda})$ becomes $\lambda_d(1-\lambda_d)$, independent of other dimensions. When the values of λ_d are equivalent for all d, Equation 3.3 reduces to Equation 3.2. A significant disadvantage of this variant is that parameter inference becomes much more difficult (discussed in detail in Section 3.4).

The kernel models defined by these equations have in the past often been used for classification purposes, where each itemset x_i is labeled with a class $c_i \in \{1, 2, \cdots, C\}$. The task is then to predict which of the C classes a new itemset y belongs to. This is typically done by first estimating C separate kernel densities for each class, then comparing the likelihood of y under each estimate (Aitchison and Aitken, 1976; Brown and Rundell, 1985; Hall, 1981; Tutz, 1986). Here we focus on the more common scenario where we have *unlabeled* itemsets (they do not originate from different classes), and the goal is to use the predictive density of the model to place high probability over the (potentially new) itemsets we may expect to see.

3.1.2 Selection of Bandwidth Parameter

Traditionally, the λ parameter for the kernel of Aitchison and Aitken (1976) has been estimated using cross-validation techniques, where the parameter is estimated by a value that maximizes an objective function with respect to a set of training data. The objective function originally used by Aitchison and Aitken (1976) was leave-one-out cross-validation log-likelihood, $\sum_{i=1}^{N} \log P(x_i|x_{-i}, \lambda)$, where $x_{-i} \equiv \{x_j : j \neq i\}$. Other objective functions that have been used include K-fold cross-validated log-likelihood (Bowman et al., 1984; Grund, 1993) and those relating to mean-squared error (Brown and Rundell, 1985; Hall, 1981).

In cases where datapoints are labeled under different classes, a separate kernel may be trained for each class and the objective function becomes a measure of how "separable" the different kernels are, in terms of classification (Tutz, 1986; Tutz and Grob, 1995). While multivariate extensions of these methods can be applied (e.g. for the D-parameter kernel function defined by Equation 3.3), it is computationally more efficient to cross-validate each parameter independently (Titterington, 1980). However, such estimations are likely to be inaccurate as the (potentially strong) dependency structure between parameters is ignored. A more in-depth review of the different objective functions used for parameter estimation are provided in Titterington (1980); Tutz and Grob (1995).

When the data is assumed to be observed as a continuous data stream, estimating kernel parameters can be made using online methods. Work in this vein focused primarily on kernels over continuous data, although they can theoretically be applied to the discrete case. In Heinz and Seeger (2006) and Zhou et al. (2003), online inference is done using a *M-kernel* model, where the summation over N terms in Equation 3.1 is replaced with a finite sum over M datapoints or centroids. In Aggarwal (2003), the summation is instead replaced with a summation over points in a finite grid. By replacing the growing sum in the kernel density to that over a fixed and finite amount of terms, efficient density evaluations and bandwidth selection can be made. A different approach, proposed by Mokkadem et al. (2009), preserves the structure of the kernel in Equation 3.1 and estimates parameters in an online fashion by minimizing mean squared error via stochastic gradient (Robbins and Monro, 1951).

3.1.3 Differences Between Previous Work and Proposed Model

There are three main differences between the work proposed in this chapter and the previous work discussed thus far in the section:

1. We propose a **two-parameter variant** of the kernel model of Equation 3.1.

2. Previous approaches are generally inefficient in scenarios where data is observed incrementally in a sequential manner. The structure of the objective function changes with the presence of new data, requiring re-optimization of the parameters over the entire historical data. We optimize a slightly different objective function suited for such applications, referred to as the **sequential log-likelihood**.

3. Motivated by the sequential setting where new data is continually observed, we propose a **stochastic gradient** approach for optimizing the sequential log-likelihood with respect to the kernel parameters. While similar to the work of Mokkadem et al. (2009), the approach of this chapter differs in that the optimization is over the proposed sequential log-likelihood and is applied to a kernel model over discrete data.

3.2 A Two-Parameter Kernel Density

The model presented in this chapter also takes the form of a kernel density estimator, and can be viewed as a generalization of Aitchison and Aitken (1976). Under this model, the probability of an itemset y given previously observed data x is

$$P(y|x, \lambda_0, \lambda_1) \equiv \frac{1}{N} \sum_{i=1}^{N} K(y|x_i, \lambda_0, \lambda_1) \tag{3.4}$$

$$K(y|x_i, \lambda_0, \lambda_1) \equiv \prod_{d=1}^{D} \lambda_0^{(1-x_{id})(1-y_d)} (1-\lambda_0)^{(1-x_{id})y_d} \lambda_1^{x_{id}y_d} (1-\lambda_1)^{x_{id}(1-y_d)} \tag{3.5}$$

where $\lambda_0, \lambda_1 \in [0, 1]$ are both bandwidth parameters. Given x, λ_0, and λ_1, one can also think of this model in a generative manner as follows:

1. Select an itemset x_i uniformly at random from the dataset.
2. For each dimension d, set

$$y_d = \begin{cases} 0 & \text{with probability } \lambda_0^{1-x_{id}}(1-\lambda_1)^{x_{id}} \\ 1 & \text{with probability } (1-\lambda_0)^{1-x_{id}}\lambda_1^{x_{id}} \end{cases}$$

Under this procedure, λ_0 can be interpreted as $P(y_d = 0|x_{id} = 0)$, and λ_1 as $P(y_d = 1|x_{id} = 1)$. A similar derivation to the one in Section 3.1.1 can be used to show that, after marginalizing over which x_i that y is a noisy observation of, the likelihood of y under this procedure is equivalent to the kernel probability density in Equation 3.4. As with the original kernel, we will further assume that $\lambda_0 \geq \frac{1}{2}$ and $\lambda_1 \geq \frac{1}{2}$ unless stated otherwise.

By introducing a second parameter to the model, the asymmetry in the behavior of the 0 and 1 values is captured. This increases the flexibility of the model, compared to the original model in Equation 3.1 and the D-dimensional variant in Equation 3.3. As an example, consider the Bernoulli probability distribution $P(y_d = 1|x_{id})$ under the respective generative procedures of the kernel models. The Bernoulli parameter, equivalent to the probability that $y_d = 1$, can take one of two different values depending on the value of x_{id}. The parameter values under all three kernel models is shown in Table 3.1.

Model	$P(y_d = 1\|x_{id} = 0)$	$P(y_d = 1\|x_{id} = 1)$
One-Parameter Kernel (Equation 3.2)	$1 - \lambda$	λ
D-Parameter Kernel (Equation 3.3)	$1 - \lambda_d$	λ_d
Two-Parameter Kernel (Equation 3.4)	$1 - \lambda_0$	λ_1

Table 3.1: Probability $P(y_d = 1|x_{id})$ under different kernel models.

The probability $P(y_d = 1|x_{id})$ is equivalent across all dimensions for the one-parameter kernel, which can be problematic when the variance in the value of y_d fluctuates across dimensions. While the D-parameter variant overcomes this problem by modeling a separate probability across the dimensions, for a single dimension d the variance of the distribution is invariant to the value of x_{id} and is always $\lambda_d(1 - \lambda_d)$. This is because, by construction, the probability $P(y_d = 1|x_{id} = 0)$ is forced to be equal to $1 - P(y_d = 1|x_{id} = 1)$, limiting the form of the two different Bernoulli distributions. In contrast, the two probabilities $P(y_d = 1|x_{id} = 0)$ and $P(y_d = 1|x_{id} = 1)$ are independent for the two-parameter kernel, allowing the variance of y_d to vary across dimensions based on the values within x_{id}.

3.2.1 Accounting for the Sparsity of the Data

The main advantage of the kernel introduced in Section 3.2 is its flexibility to model the asymmetry between the 0 and 1 values found within a dataset. Typically, multivariate binary data tends to be *sparse*, where itemsets contain only a few of the possible items. For example, customers usually buy a few items out of the thousands of available items, and emails are mostly sent to a limited number of recipients. As an empirical example, Figure 3.1 shows the size of recipient lists across all emails for two email datasets from the Gmail corpus in

Section 1.1. The number of email recipients (size of the itemset) is typically 1 or 2, clearly much smaller than the hundreds or thousands of possible recipients.

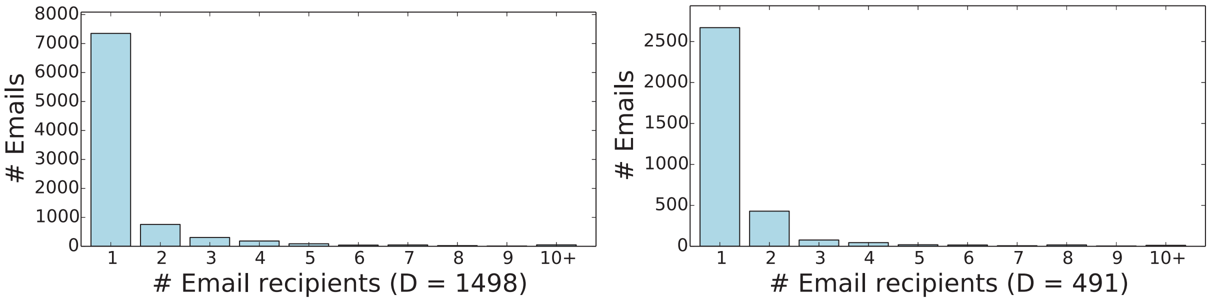


Figure 3.1: The size of email recipient lists across all sent emails, for two individuals from the Gmail email corpus.

Being able to correctly model the sparsity of a dataset is crucial for making future predictions over the types of itemsets we expect to encounter. If the expected size of an itemset under the model is significantly different from that found within the data, then a disproportionate amount of probability mass is "spread" away from what is observed within the data. As a result, the model will assign lower predictive likelihoods for new itemsets y (assuming the unknown process that generated y is similar to that of x).

Consider the quantity $\sum_{d=1}^{D} y_d$, the number of dimensions in y with a value of 1 (also referred to as the size of the itemset). Let $n_{i1} \equiv \sum_{d=1}^{D} x_{id}$ be the number of dimensions in x_i with a value of 1, and $n_{i0} \equiv D - n_{i1}$ the number of dimensions with a value of 0. The expected size of an itemset with respect to the empirical dataset x is

$$E_x\left[\sum_{d=1}^{D} y_d\right] = \frac{1}{N}\sum_{i=1}^{N} n_{i1}$$

To see why the one-parameter kernel from Section 3.1.1 is unable to model the sparsity of the data, we calculate the expected size of an itemset with respect to the density $P(y|x, \lambda)$

defined in Equation 3.1:

$$\begin{aligned}
E_K\left[\sum_{d=1}^{D} y_d\right] &= \sum_{d=1}^{D} P(y_d = 1|x,\lambda) = \sum_{d=1}^{D}\left(\frac{1}{N}\sum_{i=1}^{N}\lambda^{x_{id}}(1-\lambda)^{1-x_{id}}\right) \\
&= \frac{1}{N}\left(\sum_{i=1}^{N} n_{i1}\right)\lambda + \frac{1}{N}\left(\sum_{i=1}^{N} n_{i0}\right)(1-\lambda)
\end{aligned}$$

where we made use of the independence assumptions across dimensions in y to derive the form of $P(y_d = 1|x,\lambda)$, described in detail in Section 3.2.2.

Suppose that $\sum_{d=1}^{D} x_{id} \leq \frac{D}{2}$ for all i, as is generally the case for sparse data. It follows that $n_{i0} \geq n_{i1}$. Using this inequality, the above expectation becomes

$$E_K\left[\sum_{d=1}^{D} y_d\right] \geq \frac{1}{N}\left(\sum_{i=1}^{N} n_{i1}\right)\lambda + \frac{1}{N}\left(\sum_{i=1}^{N} n_{i1}\right)(1-\lambda) = \frac{1}{N}\sum_{i=1}^{N} n_{i1} = E_x\left[\sum_{d=1}^{D} y_d\right]$$

Thus, the kernel defined by Equation 3.1 is guaranteed to generate itemsets whose expected size is greater than that found within the empirical data. In Section 3.5.5, we will show that this property significantly reduces the predictive power of the one-parameter kernel.

In contrast, the expected size of an itemset with respect to the density $P(y|x,\lambda_0,\lambda_1)$ in Equation 3.4 is

$$\begin{aligned}
E_K\left[\sum_{d=1}^{D} y_d\right] &= \frac{1}{N}\left(\sum_{i=1}^{N} n_{i1}\right)\lambda_1 + \frac{1}{N}\left(\sum_{i=1}^{N} n_{i0}\right)(1-\lambda_0) \\
&= D(1-\lambda_0) - \frac{1}{N}\left(\sum_{i=1}^{N} n_{i1}\right)(1-\lambda_0-\lambda_1) \qquad (3.6)
\end{aligned}$$

where we made use of the property $\sum_{i=1}^{N} n_{i0} = DN - \sum_{i=1}^{N} n_{i1}$.

Comparing this quantity to $E_x\left[\sum_{d=1}^{D} y_d\right]$, we get

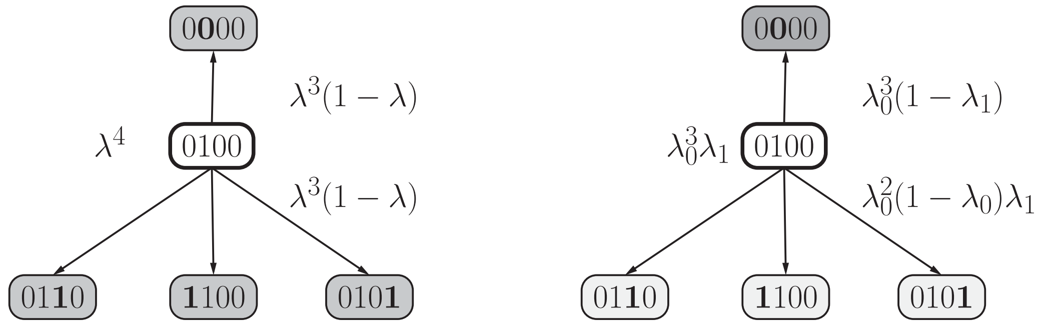
$$\begin{aligned} D(1-\lambda_0) - E_x\left[\sum_{d=1}^{D} y_d\right](1-\lambda_0-\lambda_1) &\lesseqgtr E_x\left[\sum_{d=1}^{D} y_d\right] \\ D\left(\frac{1-\lambda_0}{2-\lambda_0-\lambda_1}\right) &\lesseqgtr E_x\left[\sum_{d=1}^{D} y_d\right] \end{aligned}$$

Thus, if $D\left(\frac{1-\lambda_0}{2-\lambda_0-\lambda_1}\right)$ is less than (greater than) $E_x\left[\sum_{d=1}^{D} y_d\right]$, then y is expected to be more (less) sparse than the empirical data. The ability to model the sparsity in the data comes from the property that the direction of the inequality can be controlled using different values of λ_0 and λ_1. While it is possible in theory to set one parameter as a function of the other in order to turn the above inequality into an equality, violations in compatible parameter values may occur (i.e. λ_0 or λ_1 may become greater than 1 or less than 0). We empirically found that data sparsity is accurately modeled as $E_K\left[\sum_{d=1}^{D} y_d\right] \approx E_x\left[\sum_{d=1}^{D} y_d\right]$ across different datasets (see Figure 3.18 in the experimental results of Section 3.5.5), given the proposed estimation method in Section 3.4.

The D-dimensional kernel model defined by Equation 3.3 is similar to our model in that the expected size of an itemset can be made either smaller or larger than that found in the data. Using a similar approach as the previous models, the expected size of an itemset under this model is

$$E_K\left[\sum_{d=1}^{D} y_d\right] = \frac{1}{N}\sum_{d=1}^{D}\left(\sum_{i=1}^{N} x_{id}\right)\lambda_d + \frac{1}{N}\sum_{d=1}^{D}\left(N - \sum_{i=1}^{N} x_{id}\right)(1-\lambda_d) \lesseqgtr E_x\left[\sum_{d=1}^{D} y_d\right]$$

where the direction of the inequality depends on the values of $\{\lambda_d\}$. While this model is able to control the sparsity of y relative to the empirical data, the model must use D parameters for this property to hold, whereas our proposed model only needs two.



(a) One-parameter model (Equation 3.1). (b) Two-parameter model (Equation 3.4).

Figure 3.2: Example of how the two different kernels spread probability mass throughout the binary hypercube (assuming $x_i = [0100]$).

An alternative illustration of how the one- and two-parameter kernels spread probability mass throughout the binary hypercube via its kernel function is given in Figure 3.2. Assume that, following the generative interpretation of the model, y will be a noisy observation of $x_i = [0100]$. Both kernels have their modes for the kernel function $K(y|x_i, \lambda)$ at $y = [0100]$. Looking at the four possible values of y that differ from x_i by one dimension ($[0000], [1100], [0110]$, and $[0101]$), the probability of each under the one-parameter kernel function defined by Equation 3.2 is $\lambda^3(1-\lambda)$. Because probability mass is spread out evenly across the four, y is expected to be less sparse than x_i (since three of the four possible values have more 1's than x_i). In contrast, probability mass is *not* spread out evenly across the four outcomes for the two-parameter kernel function, defined by Equation 3.5. A probability of $\lambda_0^3(1-\lambda_1)$ is assigned to $[0000]$, while the other three values each have a probability $\lambda_0^2(1-\lambda_0)\lambda_1$. The values of λ_0 and λ_1 decide whether or not y is expected to be more or less sparse than x_i.

3.2.2 Relation to Bernoulli Mixture Models

A useful interpretation of the kernel models presented in this chapter is that of the Bernoulli mixture model (BMM) from Section 2.1.2. Here we will focus on the two-parameter model, although similar derivations can be made for the one-parameter and D-dimensional variants.

The probability density for a BMM was given in Equation 2.4:

$$P(y|\pi,\theta) = \sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} {\theta_d}^{y_d}(1-\theta_d)^{1-y_d}$$

The casting of the two-parameter kernel model into a BMM assumes there are $K = N$ components (one for each data point), each with an equal mixing weight $\pi_k = \frac{1}{N}$. Additionally, θ is a N-by-D matrix of Bernoulli probabilities, with the following structure:

$$\theta_{kd} \equiv P(y_d = 1|x_k) = (1-\lambda_0)^{1-x_{kd}}\lambda_1^{x_{kd}}$$

where k is an index into both component and data point. Plugging these values of π and θ into the probability density of the BMM, we get

$$\begin{aligned} P(y|\pi,\theta) &= \sum_{k=1}^{N} \pi_k \prod_{d=1}^{D} \theta_{kd}^{y_d}(1-\theta_{kd})^{1-y_d} \\ &= \frac{1}{N} \sum_{k=1}^{N} \prod_{d=1}^{D} \left[(1-\lambda_0)^{(1-x_{kd})y_d}\lambda_1^{x_{kd}y_d}\right]\left[\lambda_0^{(1-x_{kd})(1-y_d)}(1-\lambda_1)^{x_{kd}(1-y_d)}\right] \end{aligned}$$

where we make use of the fact that $1-\theta_{kd} = \lambda_0^{1-x_{kd}}(1-\lambda_1)^{x_{kd}}$. This is equivalent to the kernel density from Equation 3.4. Note that, although the kernel model is a special form of a Bernoulli mixture model, a BMM generally does not specify a kernel over multidimensional binary vectors.

If the same itemsets are found repeatedly within the data (as is the case with email data), an equivalent and potentially more useful BMM parameterization may be

$$P(y|\pi,\theta) = \sum_{v\in\{0,1\}^D} \left(\frac{n_v}{N}\right) \prod_{d=1}^{D} \theta_{vd}^{y_d}(1-\theta_{vd})^{1-y_d}$$

where the mixing proportions, $\frac{n_v}{N}$, are the fraction of datapoints equal to $v \in \{0,1\}^D$, and $\theta_{vd} \equiv (1-\lambda_0)^{1-v_d}\lambda_1^{v_d}$.

The relation between the kernel model and a Bernoulli mixture model is useful in that any theory regarding a BMM can be directly applied to the kernel model in Equation 3.4. For example, let $A \subset \{1, 2, \cdots, D\}$ be a subset of indices, with $y_A = \{y_d : d \in A\}$ a $|A|$-dimensional binary vector. The following properties of a BMM also apply to kernel models:

- Marginal kernel distributions of the form $P(y_A|x, \lambda_0, \lambda_1)$ are equivalent to the marginal BMM from Equation 2.5:

  $$P(y_A|x,\lambda_0,\lambda_1) = \frac{1}{N}\sum_{i=1}^{N}\prod_{d\in A} P(y_d|x_{id},\lambda_0,\lambda_1)$$

  The resulting distribution is also a kernel distribution over the $|A|$-dimensional marginalized binary space.

- Conditional kernel distributions of the form $P(y_{\tilde{A}}|y_A, x, \lambda_0, \lambda_1)$ are equivalent to the conditional BMM from Equation 2.6:

  $$P(y_{\tilde{A}}|y_A,x,\lambda_0,\lambda_1) = \sum_{i=1}^{N}\pi'_i\prod_{d\in\tilde{A}} P(y_d|x_{id},\lambda_0,\lambda_1)$$
  $$\pi'_i \propto \frac{\prod_{d\in A} P(y_d|x_{id},\lambda_0,\lambda_1)}{N}$$

In the resulting $(D-|A|)$-dimensional distribution, the mixing proportions $\{\pi'_i\}$ are no longer equal to $\frac{1}{N}$ as in a traditional kernel model, but are instead weighted according to the values of y_A.

3.3 Sequential Log-Likelihood

After defining the kernel in Section 3.2, the next task is to estimate the values of λ_0 and λ_1 given some data. The previous inference methods for the kernel proposed by Aitchison and Aitken (1976) from Section 3.1.2 can easily be extended to our two-parameter kernel. These methods involve selecting parameter values that optimize an objective function, such as a cross-validated log-likelihood. In this section, we propose a different objective function to optimize, referred to as the *sequential log-likelihood*. This objective function is motivated by the application where datapoints are observed one-by-one in a sequential manner, and has the following form:

$$\begin{aligned} LL^{\text{seq}}(x;\lambda_0,\lambda_1) &\equiv \sum_{i=2}^{N} \log P^{(i)}(x_i|\{x_j : j<i\},\lambda_0,\lambda_1) \\ &= \sum_{i=2}^{N} \log\left(\frac{1}{i-1}\sum_{j<i} K(x_i|x_j,\lambda_0,\lambda_1)\right) \qquad (3.7) \end{aligned}$$

with $K(x_i|x_j,\lambda_0,\lambda_1)$ defined by Equation 3.5 (although any kernel function can be applied here). This is the sum of the log-likelihood for each datapoint, under the kernel created using only the previous datapoints. Note that the summation starts at $i=2$; a probability of 1 is arbitrarily assigned to x_1 due to the kernel not having any data associated with it.

We use the notation $P^{(i)}$ to emphasize that the probability distribution for x_i is different from the distribution for x_j, $j \neq i$; if the probability distribution remained unchanged, the sequential log-likelihood above would reduce to the joint log-likelihood $P(x|\lambda_0,\lambda_1)$. The differences between distributions $P^{(i)}$ and $P^{(j)}$ are not in the values of λ_0 or λ_1 (which remain fixed across all $P^{(i)}$), but rather in the *support* of the kernel; the distribution $P^{(i)}$ contains a summation over $i - 1$ terms, whereas the distribution $P^{(j)}$ contains a summation over $j - 1 \neq i - 1$ terms. As the name suggests, this objective function is applicable only for data with a natural ordering. For our primary application of email, a natural ordering arises from timestamp information: if $i < j$, then it is assumed that email x_i was sent before email x_j.

The sequential log-likelihood in Equation 3.7 is closely related to the leave-one-out cross-validation log-likelihood $\sum_{i=1}^{N} \log P(x_i|x_{-i}, \lambda_0, \lambda_1)$. Whereas the leave-one-out log-likelihood considers the distribution of x_i given all other data x_{-i}, the sequential log-likelihood considers the distribution of x_i given only previous data $\{x_j : j < i\}$. As $i \uparrow N$, the distributions under consideration between the two become nearly identical; the main differences are due to the distributions of earlier datapoints. In fact, the sequential log-likelihood is an example of the prequential approach as proposed by Dawid (1992), applied to kernel density estimation.

When optimizing the sequential log-likelihood, the optimal value of λ_0 and λ_1 will generally increase as the number of datapoints N increases. This property was originally proven for the one-parameter kernel using the leave-one-out cross-validation objective function in Aitchison and Aitken (1976), however the underlying idea of the proof can also be applied here. As more data is observed, the true and unknown probability density $\tilde{P}(x)$, assumed to have generated the data, becomes better represented with those observations. Thus, the most accurate representation of $\tilde{P}(x)$ using the kernel model in Equation 3.5 is achieved with a lower variance in the kernel function (e.g. values of λ_0 and λ_1 closer to 1). It is important to remember that, although the optimal values of λ_0 and λ_1 (and, equivalently, the resulting kernel density) changes with N, the distribution $\tilde{P}(x)$ assumed to have originally generated the data is assumed to be fixed and not changing.

Algorithm 1 Gridsearch Method for Parameter Inference

```
Input: training data $x = \{x_i : 1 \leq i \leq N\}$, objective function $f(x; \lambda_0, \lambda_1)$, search space
    $\Lambda \equiv \{\lambda_0^{(s)}, \lambda_1^{(s)} : 1 \leq s \leq S\}$
Output: a value $\{\lambda_0, \lambda_1\} \in \Lambda$ which maximizes $f(x; \lambda_0, \lambda_1)$
    return $\arg \max_{\{\lambda_0, \lambda_1\} \in \Lambda} f(x; \lambda_0, \lambda_1)$
```

3.4 Inference of Kernel Parameters

In this section, we discuss different approaches to estimating kernel parameters which optimize an objective function. We will focus on the sequential log-likelihood from Section 3.3 as the objective function, although these methods apply to similar objective functions (e.g. cross-validated log-likelihood).

3.4.1 Gridsearch Methods

Perhaps the simplest approach to performing the cross-validation methods discussed in Section 3.1.2 is to apply a gridsearch over a predefined search space of possible parameter values. For each possible parameter configuration, the objective function (e.g. cross-validated log-likelihood, or the sequential log-likelihood of Section 3.3) is evaluated using a set of training datapoints. After evaluating the objective function over all predefined values, parameter estimates are obtained using the configuration yielding the optimal value in the objective function. An outline of the gridsearch method for the two-parameter kernel is given in Algorithm 1.

The gridsearch method as described in Algorithm 1 is straightforward to perform, but has some disadvantages. First, the quality of parameter estimates are largely dependent on both the range and resolution of the search space. Second, a single evaluation of the objective functions discussed in this chapter takes $O(N^2)$ time to compute; for a search space containing S parameter configurations, finding the optimal parameter configuration takes $O(SN^2)$

time[2]. Furthermore, the value of S typically grows exponentially in the number of model parameters — for a model with B parameters ($B = 2$ for the kernel model in Section 3.2) and R search points along each dimension, $S = R^B$. This can be prohibitively slow for even moderate N, B, and/or R.

While the gridsearch method has its disadvantages, it is useful under certain scenarios. For instance, a gridsearch over a coarse search space can be used as an initialization method for standard gradient methods (Silverman, 1986, Section 3.5). This would initialize the gradient method to a "high-quality" estimate (with respect to the objective function), allowing the method to quickly converge to a local maxima likely to be near the global maxima. Additionally, a golden section search (Kiefer, 1953) can be performed, which iteratively refines the search space based on the geometry of the objective function at previously evaluated points. Analogous to a binary search, at each iteration a range of possible parameter values is eliminated from the search space. For the experiments in Section 3.5.3, we will focus on the gridsearch method as defined by Algorithm 1.

3.4.2 Gradient Methods

An alternative inference procedure is to optimize the objective function numerically making use of gradient information. Gradient methods date back to Cauchy (1847), and have become a powerful and standard tool in numerical optimization. As such, they are natural methods to use in kernel bandwidth selection (Sheather and Jones, 1991; Silverman, 1986). In this section, we give a brief review of general gradient methods, in the context of estimating the parameters of the kernel model in Section 3.2 which maximize the sequential log-likelihood.

A local maximum of any objective function $f(x; \lambda)$ with respect to λ can be found by first establishing an initial parameter estimate $\lambda^{(0)}$, then iteratively updating the parameter via

[2] While the time complexity theoretically grows with S, in practice it is possible to take advantage of the fixed search space by sharing computation and vectorizing its implementation.

Algorithm 2 Gradient Method for Parameter Inference

```
Input: training data $x = \{x_i : 1 \leq i \leq N\}$, objective function $f(x; \lambda_0, \lambda_1)$, stepsize schedule
    $\boldsymbol{\gamma} \equiv \{\gamma^{(t)}, 1 \leq t \leq \infty\}$, initial parameter estimates $\lambda_0^{(0)}, \lambda_1^{(0)}$
Output: a value $\{\lambda_0, \lambda_1\}$ which locally maximizes $f(x; \lambda_0, \lambda_1)$
  for $t = 1$ to $\infty$ do
    $\lambda_0^{(t)} \leftarrow \lambda_0^{(t-1)} + \gamma^{(t)} \frac{\partial}{\partial \lambda_0} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$
    $\lambda_1^{(t)} \leftarrow \lambda_1^{(t-1)} + \gamma^{(t)} \frac{\partial}{\partial \lambda_1} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$
    if convergence criteria is met then
      return $\{\lambda_0^{(t)}, \lambda_1^{(t)}\}$
    end if
  end for
```

gradient ascent:

$$\lambda^{(t)} \leftarrow \lambda^{(t-1)} + \gamma^{(t)} \nabla f(x; \lambda^{(t-1)}) \tag{3.8}$$

where t is the iteration number, $\nabla f(x; \lambda)$ is the gradient of the objective function for training data x with respect to λ, and $\gamma^{(t)}$ is known as the *stepsize*. There are many different ways to determine the stepsize $\gamma^{(t)}$; for the experiments of this chapter, the optimization of Equation 3.8 is performed using the BFGS algorithm (Byrd et al., 1995), where a line search is performed to determine $\gamma^{(t)}$. The dimensionality of λ and $\nabla f(x; \lambda)$ are the same as the dimensionality of the kernel parameters, namely, one for the kernel of Aitchison and Aitken (1976), and two for the kernel proposed in Section 3.2. The gradient ascent method can be proven to converge to a local maximum of the objective function (Nocedal and Wright, 1999, Section 3.3)) under broad conditions[3]. An outline of the gradient method, applied to the two-parameter kernel, is shown in Algorithm 2[4].

[3] Convergence is contingent on an appropriate *stepsize schedule* $\boldsymbol{\gamma} \equiv \{\gamma^{(t)}, 1 \leq t \leq \infty\}$, with $\gamma^{(\infty)} = 0$. Also, while the method is referred to as gradient ascent, an increase in the objective function is only guaranteed for sufficiently small stepsizes $\gamma^{(t)}$.

[4] The convergence criterion is usually defined as a threshold in the difference of estimated parameters between two consecutive iterations, or a maximum number of iterations.

When optimizing the sequential log-likelihood for the two-parameter kernel[5], the objective function $f(x;\lambda) \equiv LL^{\text{seq}}(x;\lambda_0,\lambda_1)$ and x is the training dataset. The gradient is defined as

$$\nabla LL^{\text{seq}}(x;\lambda_0,\lambda_1) = \left[\frac{\partial LL^{\text{seq}}(x;\lambda_0,\lambda_1)}{\partial \lambda_0}, \frac{\partial LL^{\text{seq}}(x;\lambda_0,\lambda_1)}{\partial \lambda_1}\right]^\top$$

Taking the derivative of the sequential log-likelihood in Equation 3.7 with respect to λ_0 (a similar derivation exists for λ_1), we have

$$\begin{aligned}\frac{\partial LL^{\text{seq}}(x;\lambda_0,\lambda_1)}{\partial \lambda_0} &= \sum_{i=2}^{N} \frac{\partial}{\partial \lambda_0} \log\left(\frac{1}{i-1}\sum_{j<i} K(x_i|x_j,\lambda_0,\lambda_1)\right) &(3.9)\\ &= \sum_{i=2}^{N}\left(\frac{\sum_{j<i}\frac{\partial}{\partial \lambda_0}K(x_i|x_j,\lambda_0,\lambda_1)}{\sum_{j<i}K(x_i|x_j,\lambda_0,\lambda_1)}\right) &(3.10)\end{aligned}$$

where the derivative of the kernel function (in Equation 3.5) is

$$\begin{aligned}\frac{\partial}{\partial \lambda_0}K(x_i|x_j,\lambda_0,\lambda_1) &= \frac{\partial}{\partial \lambda_0}\left(\lambda_0^{m_{00}}(1-\lambda_0)^{m_{01}}\lambda_1^{m_{11}}(1-\lambda_1)^{m_{10}}\right)\\ m_{ab} &\equiv \sum_{d=1}^{D}\delta(x_{id}=a)\delta(x_{jd}=b)\end{aligned}$$

While the derivatives used in the gradient are straightforward in form, we found that its implementation required a significant increase in the complexity of the code. For instance, calculating $\frac{\partial}{\partial \lambda_0}LL^{\text{seq}}(x;\lambda_0,\lambda_1)$ using Equation 3.10 requires evaluating $O(N^2)$ derivatives of the form $\frac{\partial}{\partial \lambda_0}K(x_i|x_j,\lambda_0,\lambda_1)$ (the numerator terms), in addition to $O(N^2)$ evaluations of $K(x_i|x_j,\lambda_0,\lambda_1)$ (the denominator terms). In favor of code simplicity, we decided to approximate it in our experiments and avoid calculating the gradient exactly. This is done by approximating $\frac{\partial}{\partial \lambda_0}LL^{\text{seq}}(x;\lambda_0,\lambda_1)$ using the definition of a derivative:

$$\frac{\partial LL^{\text{seq}}(x;\lambda_0,\lambda_1)}{\partial \lambda_0} \approx \frac{LL^{\text{seq}}(x;\lambda_0+\epsilon,\lambda_1) - LL^{\text{seq}}(x;\lambda_0,\lambda_1)}{\epsilon}$$

[5] Similar derivations apply for cross-validated log-likelihoods and one-parameter kernel.

where we chose $\epsilon \equiv$ 1e-5 in our experiments — we found that the level of approximation in the derivatives using values of ϵ in this magnitude to be negligible. A similar approximation is made for $\frac{\partial}{\partial \lambda_1} LL^{\text{seq}}(x; \lambda_0, \lambda_1)$. This approximation has the same $O(N^2)$ time complexity as the exact calculation of the derivative, but is much simpler to implement in code.

When using the gradient ascent method for obtaining parameter estimates, the overall time complexity is $O(TN^2)$, where T is the number of iterations before a local maximum is reached. The number of iterations T is a function of 1) the starting parameter estimates, 2) the choice of stepsizes $\{\gamma^{(t)}\}$, and 3) the convergence criterion. In the experiments of Section 3.5.4, we show that the number of iterations T is typically between 30–50, and show its runtime is significantly less than the previously discussed gridsearch method. An additional advantage of gradient methods is that its search space is the continuous range of all possible parameter values — no discretized search spaces have to be defined.

3.4.3 Stochastic Gradient Methods

A disadvantage of the gradient method in Section 3.4.2 is that a single iteration takes $O(N^2)$ time, as the (derivative of) the objective function with respect to the entire training dataset must be evaluated. This can be prohibitively slow for large training datasets (e.g. tens of thousands of emails over the course of many years). As an alternative, in this section we explore *stochastic gradient* (SG) methods for efficiently obtaining parameter estimates (Robbins and Monro, 1951).

The stochastic gradient approach is very similar to the gradient method, however the gradient in Equation 3.8 is approximated at each iteration by only considering a subset of all datapoints in its calculation. Specifically, the derivatives of the gradient are approximated using with the following procedure:

Algorithm 3 Stochastic Gradient Method for Parameter Inference

```
Input: training data $x = \{x_i : 1 \leq i \leq N\}$, objective function $f(x; \lambda_0, \lambda_1)$, stepsize schedule
  $\boldsymbol{\gamma} \equiv \{\gamma^{(t)}, 1 \leq t \leq \infty\}$, initial parameter estimates $\lambda_0^{(0)}, \lambda_1^{(0)}$
Output: a value $\{\lambda_0, \lambda_1\}$ which locally maximizes $f(x; \lambda_0, \lambda_1)$
  for $t = 1$ to $\infty$ do
    Create minibatch $m_t$ from training dataset
    Approximate $\frac{\partial}{\partial \lambda_0} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$ using minibatch $m_t$ (e.g. Equation 3.11)
    Similarly, approximate $\frac{\partial}{\partial \lambda_1} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$ using minibatch $m_t$
    $\lambda_0^{(t)} \leftarrow \lambda_0^{(t-1)} + \gamma^{(t)} \frac{\partial}{\partial \lambda_0} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$
    $\lambda_1^{(t)} \leftarrow \lambda_1^{(t-1)} + \gamma^{(t)} \frac{\partial}{\partial \lambda_1} f(x; \lambda_0^{(t-1)}, \lambda_1^{(t-1)})$
    if convergence criteria is met then
      return $\{\lambda_0^{(t)}, \lambda_1^{(t)}\}$
    end if
  end for
```

1. Subsample $M \geq 1$ of the N training datapoints at random, creating a "minibatch". Let $m \subset \{1, 2, \cdots, N\}$ be an index set representing which datapoints are in the minibatch.

2. Replace the derivative's first summation over all training datapoints with a summation only over the minibatch. For example, $\frac{\partial}{\partial \lambda_0} LL^{\text{seq}}(x; \lambda_0, \lambda_1)$ is approximated as

$$\frac{\partial LL^{\text{seq}}(x; \lambda_0, \lambda_1)}{\partial \lambda_0} \approx \sum_{i \in m} \frac{\partial}{\partial \lambda_0} \log \left(\frac{1}{i-1} \sum_{j<i} K(x_i | x_j, \lambda_0, \lambda_1) \right) \tag{3.11}$$

An outline of the stochastic gradient method, applied to the two-parameter kernel, is shown in Algorithm 3. By subsampling M datapoints to approximate the derivatives, the time complexity of a single iteration is reduced from $O(N^2)$ to $O(MN)$. Similar to the gridsearch method of Section 3.4.1, there is a trade-off between accuracy and time complexity. In the limit where $M = N$, the SG and gradient methods become equivalent, thus requiring $O(N^2)$ time for a single iteration. On the other hand, when $M = 1$ the SG approach crudely approximates the gradient, requiring only $O(N)$ time per iteration. As with gradient methods, typically many iterations of parameter updates are required — at each iteration the gradient is approximated with Equation 3.11 using a different minibatch of M datapoints.

Obtaining parameter estimates using the stochastic gradient approach takes $O(TN)$ time, where T is the number of iterations before a convergence criteria is reached. While this may seem more efficient than the $O(TN^2)$ complexity of the gradient method, the number of iterations required before convergence varies greatly between the two methods. As each stochastic gradient iteration approximates the gradient using a small minibatch of datapoints, many more iterations may be needed to converge to a local maximum. As we discuss in Section 3.4.4 and show empirically in Section 3.5.4, reliable parameter estimates can be obtained by applying stochastic gradient to a single pass through the training data[6], resulting in an overall $O(N^2)$ time complexity.

Determining the Stepsize $\gamma^{(t)}$

While the approximated derivatives may be unreliable (potentially *decreasing* the value of the objective function after an iteration), convergence is guaranteed for any value of M over a sufficient number of iterations and appropriate stepsizes $\{\gamma^{(t)}\}$ (Bottou, 1998). One way to determine the stepsize parameter, suggested by Bottou (2012), is to have $\gamma^{(t)} \equiv (1 + \alpha t)^{-1}$, where $\alpha > 0$ is a parameter determining how quickly the stepsize approaches zero over iterations. However, preliminary experiments suggested that the optimal value of α can vary significantly across different email datasets.

In the experiments of this chapter, we apply the Adadelta algorithm of Zeiler (2012) in order to determine the stepsize $\gamma^{(t)}$ at each iteration. Instead of defining $\gamma^{(t)}$ to be a function of t and α, it is defined using only the gradient information and magnitude of changes to parameters from the previous iterations. Let $\Delta\lambda_t$ and g_t be the change in parameters and

[6] The reliability of the parameter estimates increases with the size of the training data. The experimental results in Section 3.5 suggest that most email datasets, including those which span a range of only a few months, had sufficient data for reliable estimation.

the (approximated) gradient at iteration t, respectively. The stepsize at iteration t is

$$\gamma^{(t)} \equiv \frac{\sqrt{E[\Delta\lambda^2] + \epsilon}}{\sqrt{E[g^2] + \epsilon}}$$

where

$$E[\Delta\lambda^2] \equiv \sum_{i=1}^{t-1} \rho^{(t-1)-i}(1-\rho)\Delta\lambda_i^2 \qquad\qquad E[g^2] \equiv \sum_{i=1}^{t} \rho^{t-i}(1-\rho)g_i^2$$

and $\{\rho, \epsilon\}$ are hyperparameters. The numerator term acts as a momentum term, where large (small) changes in parameter values will allow for larger (smaller) stepsizes in the following iterations. The denominator term uses the gradient information to smooth out future updates. While this definition requires two hyperparameters, preliminary experiments agreed with the findings of Zeiler (2012) that the performance of the stochastic gradient method is much less sensitive to ρ and ϵ than they were on α. We set $\rho = 0.99$ and $\epsilon = 0.0001$ for the upcoming experiments.

3.4.4 Optimization of Parameters for Sequential Data

The inference methods discussed thus far have assumed the existence of a large training dataset x, which the objective function is maximized over. As mentioned in Section 3.3, we are interested in applications where data is observed incrementally in a sequential manner, where models must repeatedly make efficient parameter updates as new datapoints are observed. Gridsearch and gradient methods do not adapt well to the presence of new data, as the shape (and potentially the maxima) of the objective function changes. This requires re-evaluating the objective function across the entire search space for gridsearch methods, or re-applying the complete gradient ascent method to discover the new maximum.

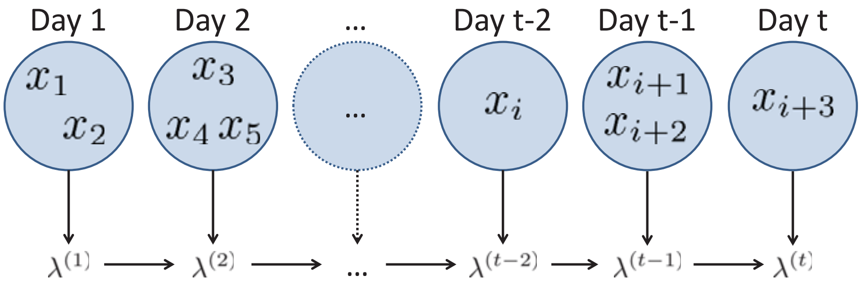


Figure 3.3: Illustration of the stochastic gradient method, where a minibatch of data m_t corresponds to itemsets which were observed on day t.

The stochastic gradient method of Section 3.4.3 is a natural inference method for iteratively optimizing the objective function when data is observed sequentially. Each time a new datapoint (or minibatch of datapoints) is observed, parameters can be updated using the gradient update of Equation 3.8 with the minibatch approximation to the derivative via Equation 3.11. This is a slightly different approach to traditional stochastic gradient discussed in Section 3.4.3, where minibatches were created randomly over historical data. It is possible to use the randomized method for creating minibatches over the (continually increasing) historical data, however defining minibatches sequentially is a natural choice given the nature of the sequential data. For the experiments of this chapter, a minibatch is defined to be the itemsets which were observed over a continuous 24-hour window of time[7]. An illustration of the stochastic gradient process under this setup is given in Figure 3.3. One quality of this approach is that the size of the minibatch varies across days.

An interesting note is that this method of updating kernel parameters after each minibatch is processed can be seen as a special type of "probability forecasting system" (Dawid, 1992). Such systems are designed to iteratively take actions that minimize local expected loss functions with respect to historical data, in order to minimize a global loss function over the joint data. Under this context, the "action" that the system (kernel model) would make is changing the value of the λ parameter. The local loss function would be the negative

[7] This assumes the itemset data is timestamped, which is the case for the email datasets used in the experiments.

log-likelihood of a single datapoint x_i, and the global loss function would be the negative sequential log-likelihood, referred to as the *prequential log-likelihood* in Dawid (1992).

3.5 Experiments

We begin the experiments section by first describing the datasets of interest in Section 3.5.1 and a useful transformation of kernel parameters (and priors of) in Section 3.5.2. The experiments in Section 3.5.3 compare the optimization of different objective functions for parameter inference, using both the one- and two-parameter kernels. In Section 3.5.4, the different inference methods (gridsearch, gradient, SG) are compared in terms of accuracy and time. Lastly, we evaluate in Section 3.5.5 the fitting of the one- and two-parameter kernels in the setting where new data is sequentially observed, compared to Bernoulli mixture models.

3.5.1 Description of Datasets

For these experiments, we focus on the emails sent by inbox owners in the email datasets described in Section 1.1. Each individual with an email inbox corresponds to a separate dataset; the experimental procedures described in this section are independently run for each individual. This setup simulates the scenario where the models are treated as extensions to an individual's email client, having access to only that individual's email activity.

Suppose an individual sent N emails to D different people; their dataset is represented as $x = \{x_{id} : 1 \leq i \leq N, 1 \leq d \leq D\}$, where $x_{id} = 1$ if and only if the individual sent email i to person d. We do not distinguish whether or not the person appeared in the "to", "cc", or "bcc" field. The emails are sorted such that $i < j$ implies email i was sent before email j.

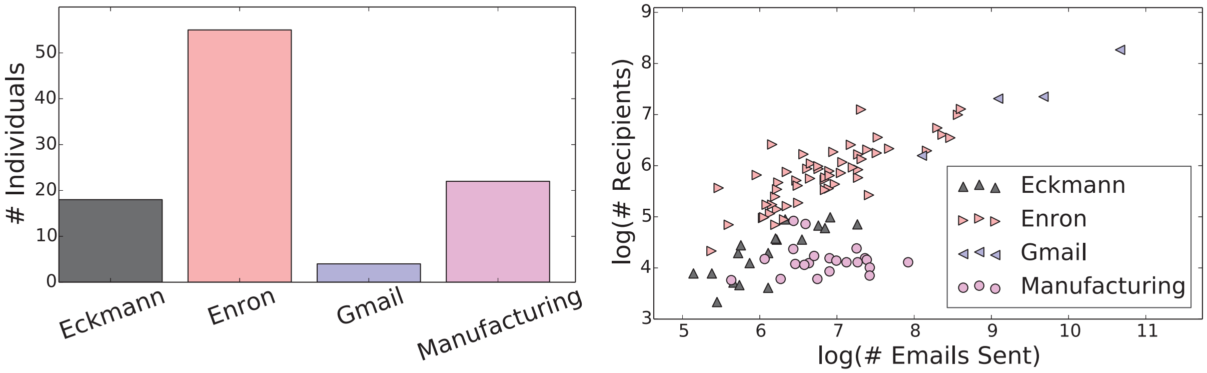


Figure 3.4: Summary statistics for the email datasets used in the experiments. Left: the number of individuals considered from each email corpus. Right: for each individual, the number of emails they sent versus the number of unique recipients they sent emails to.

To ensure the models have sufficient data for parameter inference, only individuals whose email activities meet the following criteria were considered:

1. The individual sent 100 or more emails that included more than one recipient.

2. At least 10 different people received 5 or more emails from the individual.

Summary statistics for the individuals that meet this criteria are displayed in Figure 3.4; a total of 99 email datasets are considered in these experiments. The number of sent emails and recipients varies greatly between the individuals; some individuals send a lot of emails to relatively few people, while others send few emails to many people. The following experiments will show that, in general, the kernel methods have the flexibility to model this wide range of behavior.

3.5.2 Priors and Log-Odds of Kernel Parameters

Before discussing the experiments, we first explain two important details in the estimation of kernel parameters which optimize an objective function.

Parameter Priors

Motivated by Bayesian ideas, prior distributions are placed over the kernel parameters (λ for the one-parameter kernel, and $\{\lambda_0, \lambda_1\}$ for the two-parameter kernel). By placing priors over the parameters, the objective function to optimize becomes (for the two-parameter kernel)

$$\log P(\lambda_0) + \log P(\lambda_1) + f(x; \lambda_0, \lambda_1)$$

where $f(x; \lambda_0, \lambda_1)$ is a log-likelihood term (e.g. the sequential log-likelihood). Unlike a typical Bayesian setup, this objective function is not a proper posterior distribution over the kernel parameters, as the $f(x; \lambda_0, \lambda_1)$ term is not a joint distribution over x (see Section 3.3; a similar explanation applies to cross-validated log-likelihoods). Instead, the prior distributions act as a *regularization term*[8] over the possible kernel parameters.

For the gridsearch and gradient inference methods, inclusion of the prior distributions is trivial as the value of the prior (and its derivative) can be easily computed. In order to include the prior distributions in the SG approach, the priors must be scaled as to not dominate the approximate derivative of the log-likelihood term. For example, if the objective function is the sequential log-likelihood regularized by parameter priors, the SG approximation to the derivative (Equation 3.11) becomes

$$\frac{\partial LL^{\text{seq}}(x; \lambda_0, \lambda_1)}{\partial \lambda_b} \approx \frac{|m|}{N} \left(\frac{\partial}{\partial \lambda_b} \log P(\lambda_b) \right) + \sum_{i \in m} \frac{\partial}{\partial \lambda_b} \log \left(\frac{1}{i-1} \sum_{j<i} K(x_i | x_j, \lambda_0, \lambda_1) \right)$$

where $b \in \{0, 1\}$ and m represents the minibatch used for the parameter update.

[8] It is common to have a regularization parameter β determining the strength of the regularization during optimization. Here we implicitly assume that $\beta = 1$. While we could change the strength of regularization by changing the value of β, an equivalent effect can be achieved by changing the strength of the prior distribution (see Section 2.2.2 for details).

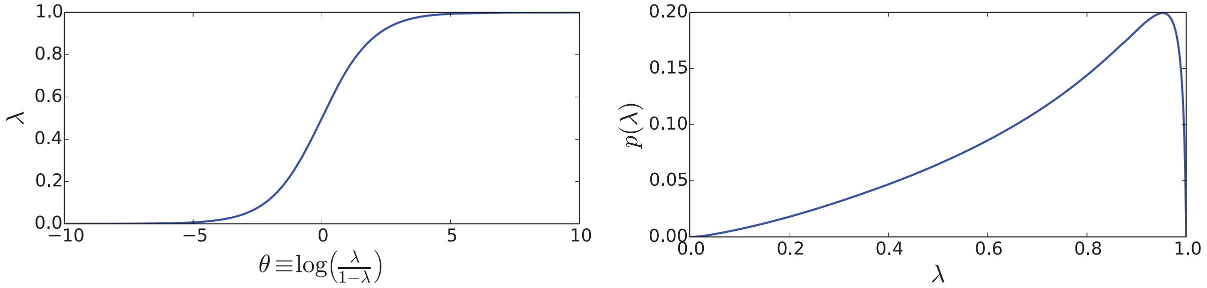


Figure 3.5: Relationship between λ and θ. Left: the one-to-one mapping between λ and θ. Right: Prior over λ taken by transforming the Gaussian$(3, 2)$ prior over θ.

Log-Odds Representation of Kernel Parameters

A potential problem with the gradient and SG inference methods is that the possible values of the parameters (λ, λ_0, or λ_1) are bounded on the interval $[0, 1]$. In order to make these algorithms robust to boundary conditions, the following transformation of parameters is taken prior to optimization:

$$\theta \equiv \log\left(\frac{\lambda}{1-\lambda}\right) \qquad \theta_0 \equiv \log\left(\frac{\lambda_0}{1-\lambda_0}\right) \qquad \theta_1 \equiv \log\left(\frac{\lambda_1}{1-\lambda_1}\right)$$

where $-\infty \leq \theta \leq \infty$ is known as the *log-odds* of λ (similarly for θ_0 and θ_1). The different inference methods then optimize the objective function with respect to the log-odds of parameters, transforming the estimations back to their original form afterwards.

A Gaussian$(\mu = 3, \sigma^2 = 2)$ prior is placed over θ, θ_0, and θ_1. How this prior translates into a prior over the original kernel parameters, along with the mapping between λ and θ, is shown in Figure 3.5. As the experimental results will show, this is a relatively weak prior that quickly becomes dominated by the observed data.

3.5.3 Optimization of Different Objective Functions

The first experiment compares the values in estimated parameters for both the one- and two-parameter kernels when optimizing the following objective functions (regularized via prior distributions over kernel parameters):

1. leave-one-out cross-validated log-likelihood
2. 10-fold cross-validated log-likelihood
3. sequential log-likelihood of Section 3.3

Each objective function is maximized using only the gridsearch procedure of Section 3.4.1. Comparisons between the different inference methods are saved for the experiments in Section 3.5.4 and between the two kernel models in Section 3.5.5.

Experimental Setup

For a single email dataset (individual's inbox), the dataset is split chronologically into 80% training and 20% test. A separate one- and two-parameter kernel is trained for each objective function to maximize. These objective functions are with respect to only the training dataset — the test data is ignored during parameter inference. Once the kernel models are trained, their parameters become fixed as the models are evaluated over the test data. Results are then aggregated across all separate individual datasets.

Gridsearch Inference Procedure

The different objective functions are maximized using the gridsearch inference procedure described in Section 3.4.1. For our two-parameter kernel, the grid search is over 50 values of $2 \leq \theta_0 \leq 10$ (equivalently, $0.88 \leq \lambda_0 \leq 1 - 4.5 \times 10^{-5}$), and 50 values of $-3 \leq \theta_1 \leq 5$ (equivalently, $0.05 \leq \lambda_1 \leq 0.99$), for a total of 2500 parameter configurations. Similarly, for the kernel in Aitchison and Aitken (1976), a grid search over 200 values of $2 \leq \theta \leq 12$ (equivalently, $0.88 \leq \lambda \leq 1 - 6.1 \times 10^{-6}$) is performed. These boundaries were chosen after preliminary experiments to ensure that the optimal parameter values across datasets lie within this range.

Evaluation Metric: Test Log-Likelihood

Once a kernel's parameters are learned from the 80% training data, that kernel is evaluated by calculating the log-likelihood of the 20% test data conditioned on the training data and kernel parameters. Let $x = \{x_i : 1 \leq i \leq N^{\text{train}}\}$ represent the training data, and $y = \{y_j : 1 \leq j \leq N^{\text{test}}\}$ the test data. The test log-likelihood (for the two-parameter kernel) is defined as

$$\frac{\log P(y|x, \lambda_0, \lambda_1)}{N^{\text{test}}} = \frac{1}{N^{\text{test}}} \sum_{j=1}^{N^{\text{test}}} \log \left(\frac{1}{N^{\text{train}}} \sum_{i=1}^{N^{\text{train}}} K(y_j|x_i, \lambda_0, \lambda_1) \right) \tag{3.12}$$

which is the average log-likelihood across test datapoints under the kernel model trained with the training data. In this calculation, the kernel is not updated after evaluating the log-likelihood of test data point y_j — the parameters remain fixed to their values estimated from the training data to ensure that all methods are being evaluated on the same basis.

Results

We focus first on results regarding the one-parameter kernel of Aitchison and Aitken (1976), then discuss the results with respect to our two-parameter kernel. The results in this section across the two kernels are similar (in terms of relative performance between the different

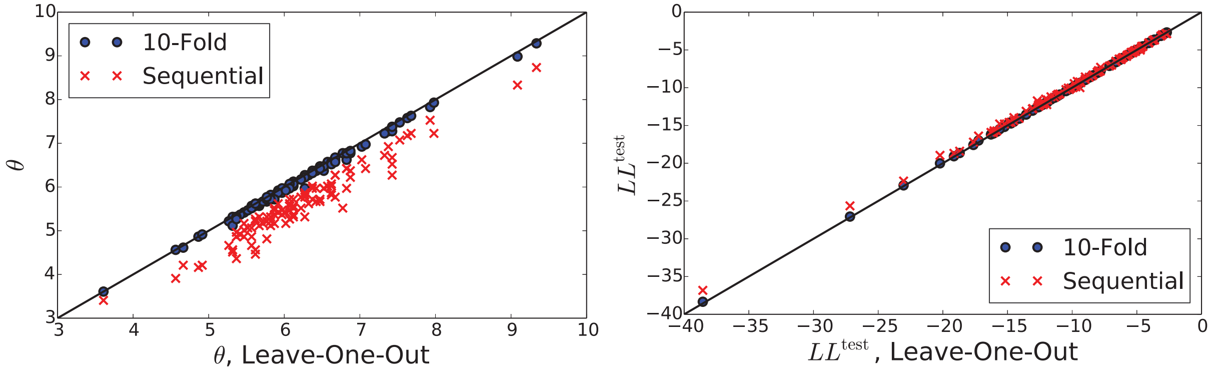


Figure 3.6: The estimated log-odds of the kernel parameter θ (left) and test log-likelihood (right) for the one-parameter kernel, across all email datasets. The x-axis represents inference via maximizing the leave-one-out objective function, with the y-axis representing both the 10-fold (blue circles) and sequential (red x's) objective functions.

objective functions) — the discussion regarding the one-parameter kernel directly applies to the two-parameter version.

Figure 3.6 shows the estimated log-odds of the kernel parameter (θ) and the test log-likelihood across all email datasets. The left plot compares the estimated values of θ when optimizing the leave-one-out log-likelihood versus both the 10-fold and sequential log-likelihoods. Compared to the estimated values of θ when maximizing the leave-one-out log-likelihood, a minor but systematic decrease in parameter estimates occurs when maximizing the 10-fold log-likelihood. The systematic decrease becomes more significant when the sequential log-likelihood is maximized. However, as shown in the right plot of Figure 3.6, the decreased parameter estimates do not significantly affect the kernel's performance in terms of test log-likelihood — in some cases the test log-likelihood is increased.

An illustration of this phenomenon is shown in Figure 3.7, where the shape of the different objective functions and the test log-likelihood are shown as a function of θ for two separate individuals (datasets). The leave-one-out and 10-fold log-likelihoods over the training data (top row) are very similar in shape, while the the form of the sequential log-likelihood is systematically smaller with its mode at a smaller value of θ. The test log-likelihood (bottom

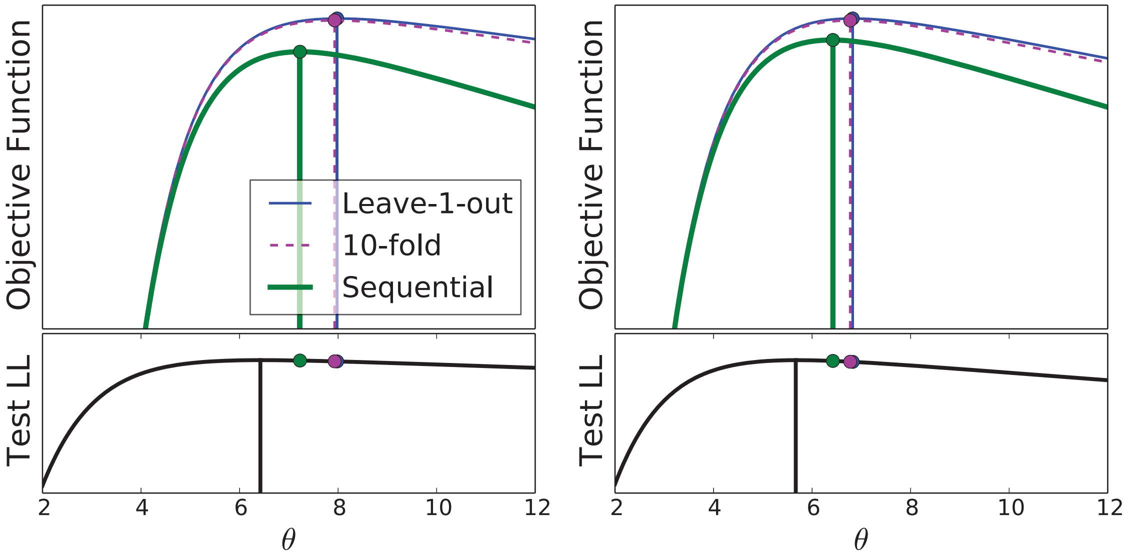


Figure 3.7: Values of the different objective functions (top row) and test log-likelihood (bottom row) over the range of θ parameters explored via gridsearch. Each column represents a different individual dataset. Vertical lines represent the maximum value of each function.

row) is similar in shape to these objective functions[9]. The similar performance in terms of test log-likelihood comes from the fact that the modes of all objective functions lie in a regime of the test log-likelihood that changes very slowly as a function of θ (the values of θ in this range translate to near-equivalent values of $\lambda \approx 0.997$). Although this example focuses on only two email datasets, similar patterns are found across all other datasets.

The systematic differences between parameter estimates comes from the asymptotic properties of optimal parameter values (Aitchison and Aitken, 1976), previously discussed in Section 3.3. When conditioning on a larger training dataset, the optimal value of λ (equivalently, θ) in terms of predictive likelihood becomes closer to $\lambda = 1$ ($\theta = \infty$). While the size of the training dataset stays the same when optimizing the different objective functions, the number of datapoints conditioned on when evaluating datapoint-wise log-likelihoods within the objective functions are different. Table 3.2 shows the number of training datapoints

[9] We only show the form of the test log-likelihood for illustrative purposes, as one would not directly maximize this function since the test data is usually removed during inference.

conditioned on when evaluating the contribution of training datapoint x_i to each objective function.

Log-likelihood Function	**# datapoints conditioned on with respect to x_i**
Leave-one-out	$N-1$, as $P(x_i \mid x_{-i}, \lambda)$ is calculated.
10-fold	$\approx 0.9N$, as $P(x_i \mid \{\text{data from other 9 folds}\}, \lambda)$ is calculated.
Sequential	$i-1$, as $P(x_i \mid \{x_j : j < i\}, \lambda)$ is calculated.

Table 3.2: Number of training datapoints conditioned on when evaluating the contribution of datapoint x_i to each of the log-likelihood objective functions.

Each objective function contains a sum over log-probabilities across all training datapoints, conditioned on different subsets of other datapoints. As the size of those subsets becomes larger, so does the optimal parameter value maximizing the objective function. The 10-fold log-likelihood conditions on a slightly smaller set of datapoints compared to the leave-one-out log-likelihood (the training data in the 9 other validation folds, versus all $N-1$ other training datapoints). In contrast, the sequential log-likelihood conditions on a much smaller subset of datapoints — only the $i-1$ datapoints that come before x_i. As a result of the asymptotic properties described in Aitchison and Aitken (1976), the optimal value of θ will tend to be smaller when optimizing the sequential log-likelihood compared to cross-validated log-likelihoods. This is precisely what is portrayed in the results of Figure 3.6.

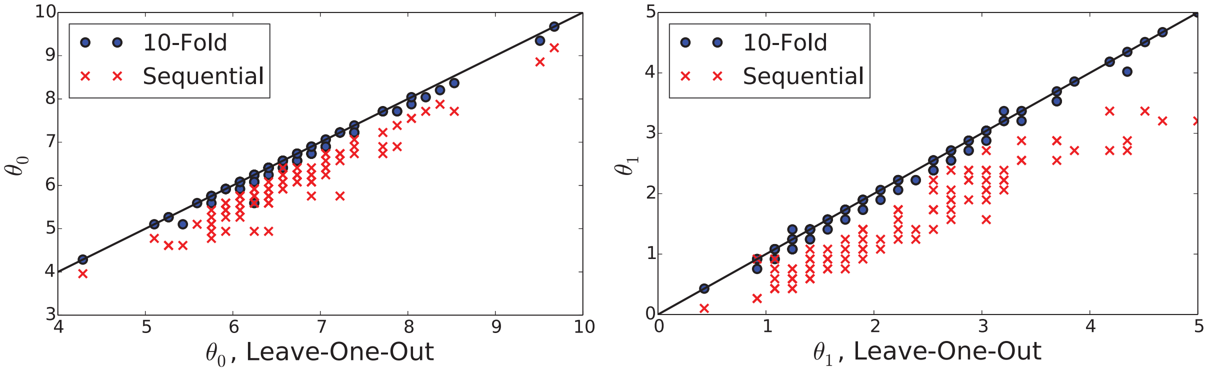


Figure 3.8: The estimated log-odds of the kernel parameters θ_0 (left) and θ_1 (right) for the two-parameter kernel, across all email datasets. The x-axis represents inference via maximizing the leave-one-out objective function, with the y-axis representing both the 10-fold (blue circles) and sequential (red x's) objective functions.

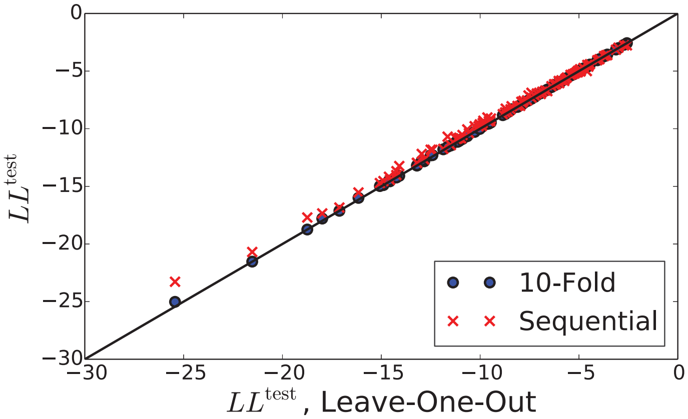


Figure 3.9: The test log-likelihood for the two-parameter kernel, across all email datasets. The x-axis represents inference via maximizing the leave-one-out objective function, with the y-axis representing both the 10-fold (blue circles) and sequential (red x's) objective functions.

Focusing now on the two-parameter kernel, Figures 3.8 and 3.9 compare the estimated values of $\{\theta_0, \theta_1\}$ and the resulting test log-likelihood when maximizing each objective function across all email datasets, respectively. The relative performance between the different objective functions is similar to the one-parameter kernel; the estimated values of θ_0 and θ_1 are systematically smaller when maximizing the sequential log-likelihood as opposed to cross-validated log-likelihoods. The difference in test log-likelihood as a result of the difference in parameter estimates is generally negligible, however a small improvement in test log-likelihood is achieved when maximizing the sequential log-likelihood for some datasets (similarly for the one-parameter kernel in Figure 3.6). This is because the smaller values of θ_0 and θ_1 allow for more smoothing (larger variance) of the kernel function, which can allow for a better fit to test data if there there are differences between the datapoints within the training and test datasets.

3.5.4 Comparison of Different Inference Methods

The previous experiment compared the cross-validated and sequential log-likelihoods as objective functions to optimize for the purpose of parameter inference. In this section, we focus on the performance of gridsearch, gradient, and SG inference methods in terms of estimation accuracy and time to compute the estimates. We focus on the optimization of the sequential log-likelihood, although similar results occur for cross-validated log-likelihoods. A direct comparison between the one- and two-parameter kernels is saved for Section 3.5.5.

Experimental Setup

The experimental setup is identical to the previous experiments in Section 3.5.3. Each dataset is split into 80% training and 20% test data, where the kernel parameters (for both one- and two-parameter variants) are inferred using each of the different inference methods and using only the training data. After the parameters are estimated, they remain fixed as the log-likelihood of the test data conditioned on the training data and kernel parameters is calculated (via Equation 3.12).

Inference Procedures

A separate kernel is independently trained for each inference procedure (gridsearch, gradient, SG) that optimizes the sequential log-likelihood, regularized by parameter priors, over training data. For the gridsearch method, the range of possible parameter values explored is identical to that of the experiments in Section 3.5.3.

The gradient method follows the procedure described in Section 3.4.2 to iteratively maximize the sequential log-likelihood. Initial parameter estimates are set by setting

$$\lambda = \lambda_0 = \lambda_1 = 1 - \frac{1}{D}$$

and setting the equivalent log-odds parameters (θ, θ_0, and θ_1 of Section 3.5.2) appropriately. Under this parameterization, the expected number of differing dimensions between y and x_i

under the kernel functions defined by Equations 3.2 and 3.5 is exactly 1. We found that the performance of the gradient method is robust to this initial estimate. The implementation of the gradient method used is that found in the scientific Python package SciPy[10].

For the stochastic gradient method from Section 3.4.3, the training dataset is treated as though it were a continuous data stream. The training data is partitioned into minibatches, with each minibatch containing emails over a single 24-hour period (which may vary in size — see Figure 3.3). Kernel parameters are then updated after each minibatch is processed, simulating the scenario where model parameters are updated on a daily basis. The same initialization of parameters for the gradient method is also used for the stochastic gradient method.

For m_t, the tth minibatch, the kernel parameters are updated by applying the SG update of Equation 3.8 with a stepsize schedule according to the Adadelta algorithm (Zeiler, 2012). After the SG method passes through the minibatches of streaming data a single time, final parameter estimates (with which to evaluate the log-likelihood of test datapoints) are taken by averaging over the estimated parameter values across the last 10 minibatches. This averaging helps to eliminate noise introduced in the updating of parameter values due to the stochastic approximation of derivatives.

Evaluation Metrics: Test Log-Likelihood, Inference Time

For these experiments, three separate evaluation metrics are used. The first is the log-likelihood of test datapoints as defined by Equation 3.12. Second, the total time taken to produce parameter estimates is compared across the different inference methods. The last metric is the total number of objective function evaluations needed in order to produce parameter estimates. Although related to total running time, the number of function eval-

[10] See the "minimize" function of the **scipy.optimization** package in SciPy version 0.14.0. The minimization is over the negative objective function, equivalent to maximizing the original objective function.

uations helps to give perspective on the theoretical time it takes for the inference methods to run, removing biases introduced in the implementations across different methods.

The total number of function evaluations is constant for the gridsearch and SG inference methods. For gridsearch methods, this number is the size of the search space. It is important to note that this number is arbitrary, as there is a trade-off between time and accuracy (see Section 3.4.1). However, a high-resolution search space is necessary in order to ensure accurate parameter estimations. While the SG method never directly evaluates the objective function, it has the equivalent complexity of calculating the objective function two (three) times for the one-parameter (two-parameter) kernel, due to the numerical approximation of the derivative(s) and the single pass through the training data. It is possible to pass through the training data multiple times under the SG framework, however we will show that a single pass is sufficient to produce estimates similar to those obtained from other methods. Only the gradient method has a variable number of objective function evaluations.

Results

We begin by comparing the differences between the estimated parameter values when using the gridsearch, gradient, and stochastic gradient methods for maximizing the sequential log-likelihood. These differences can be seen for the one-parameter kernel in Figure 3.10, and for the two-parameter kernel in Figure 3.11, where the differences are relative to the estimated parameters from the gridsearch method. As described in the previous section, the recorded parameter estimates using the stochastic gradient method is the average of the parameter estimate across the last 10 minibatches.

As expected, Figures 3.10 and 3.11 show that the estimated parameters using the gradient method are almost equivalent to that when using gridsearch. The differences shown are due mostly to the discretization of the search space in gridsearch methods — given a search

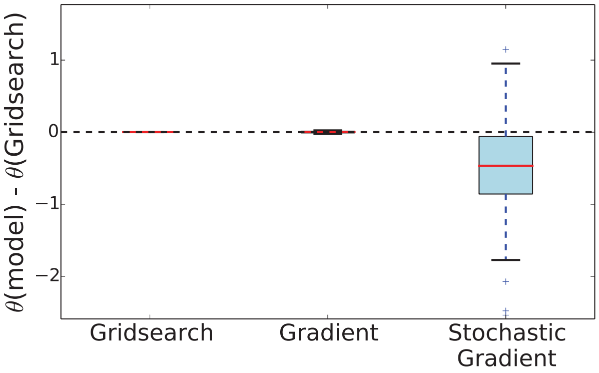


Figure 3.10: Differences in the estimated value of θ between the different inference methods and the gridsearch method, for the one-parameter kernel, across email datasets.

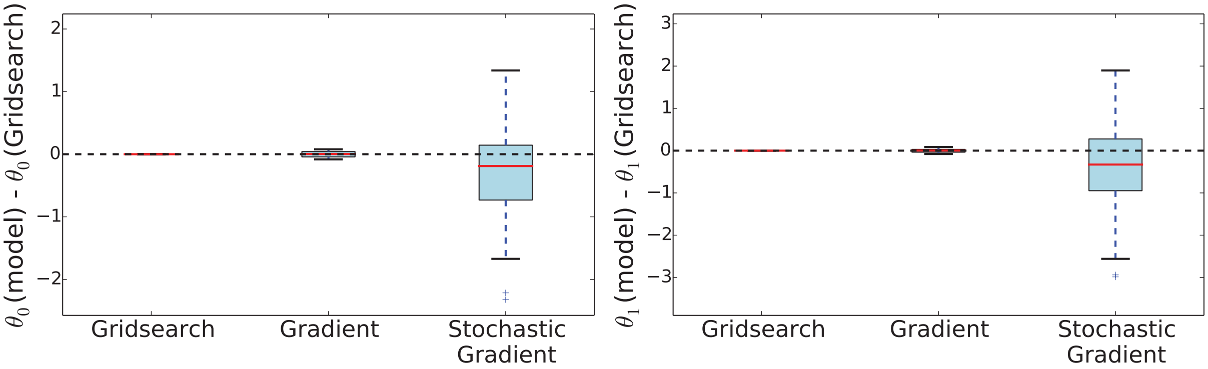


Figure 3.11: Differences in the estimated values of θ_0 (left) and θ_1 (right) between the different inference methods and the gridsearch method, for the two-parameter kernel, across email datasets.

space with an infinite resolution, the gridsearch and gradient methods will in theory find the same optimal parameter values. Additional minor differences can come from the convergence criterion of gradient methods; here the gradient method terminated when the difference in updated parameter values falls below 0.001.

For the stochastic gradient method, the accuracy of the estimated parameter values (relative to those found by the gridsearch method) varies quite a bit. An illustration for why this is shown in Figure 3.12, where the estimated value of θ is shown across minibatches, for two individual email datasets. Although this figure shows the estimated parameter for only the one-parameter kernel, similar behavior occurs for both θ_0 and θ_1 of the two-parameter kernel. The stochastic gradient method tends towards smaller values of θ in the beginning

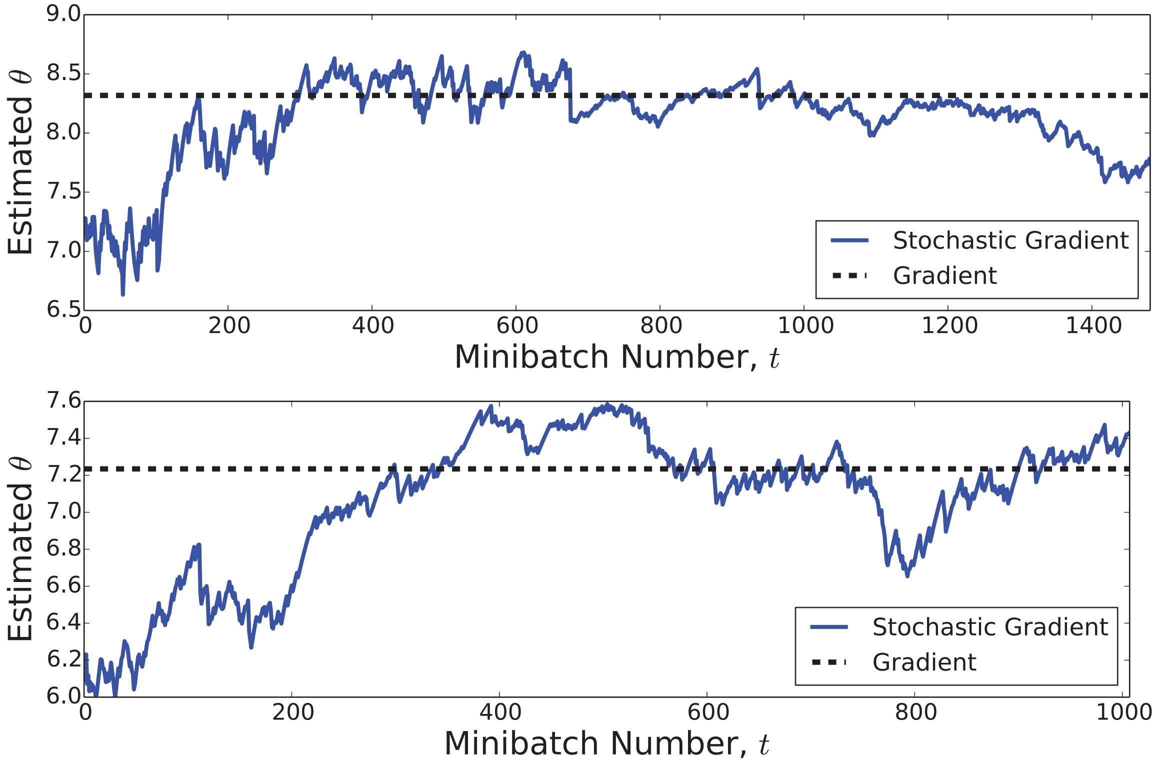


Figure 3.12: Estimated values of θ for the one-parameter kernel over minibatches when using the stochastic gradient inference method, for two individual datasets.

(for reasons explained earlier), then slowly approaches the gradient's solution. However, due to the stochastic nature of the gradient updates, the estimated value of θ can temporarily deviate from the solution of the gradient method. As final parameter estimates are taken by averaging across estimates over the last 10 minibatches, the estimation can differ significantly from the gradient's solution.

In Figure 3.13, the test log-likelihood when using parameter estimates inferred by the different methods are compared, for both the one- and two-parameter kernel. As with Figures 3.10 and 3.11, the differences shown are with respect to the gridsearch method; a large positive difference represents a higher test log-likelihood compared to the gridsearch method. The difference in test log-likelihood between the gridsearch and gradient methods are minimal.

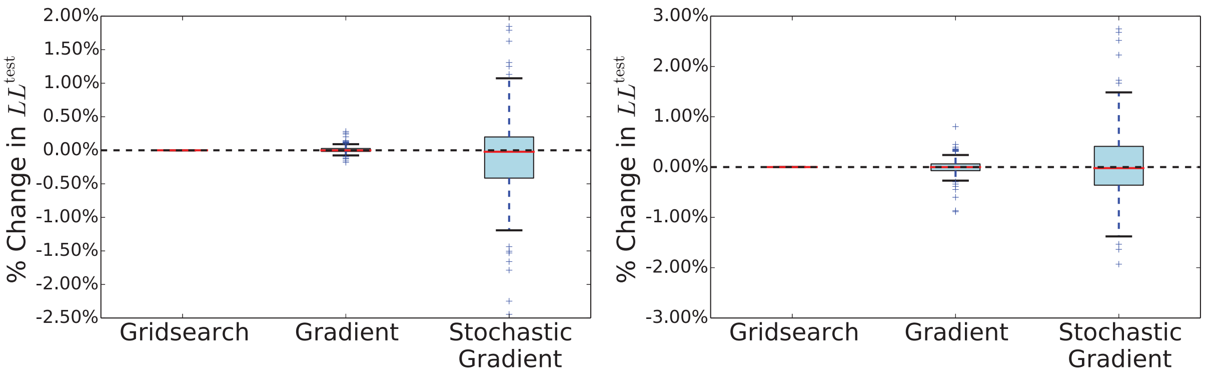


Figure 3.13: Differences of the test log-likelihood between the different inference methods and the gradient method, for both the one-parameter (left) and two-parameter (right) kernels, across email datasets.

While the differences in the stochastic gradient method are larger, the overall difference is still minor — the largest change in log-likelihood is at most a few percent[11].

The fact that similar log-likelihoods can be achieved with large differences in parameters is due to the nature of the test log-likelihood, examples of which were shown in the bottom row of Figure 3.7. For large values of θ, the test log-likelihood decreases slowly as a function of θ. The reason for this is that large values of θ map into similar values of λ (see the left plot of Figure 3.5). Although only relative differences in parameters were shown in Figures 3.10 and 3.11, the magnitude of the parameters lie within this regime of the test log-likelihood.

While the differences in test log-likelihood may be minimal, the time it takes to calculate those parameter estimates varies greatly across the different methods. In Figure 3.14, the ratio of the inference time of the different methods to the stochastic gradient method is shown across all datasets. For the one-parameter kernel in the left plot, both the gridsearch and gradient methods took roughly the same time and are approximately 6–7 times slower than the stochastic gradient method on average. In Figure 3.15, the dataset-wise inference times between the gradient and stochastic gradient methods for both one- and two-parameter ker-

[11] Due to the discrete nature of itemset data, the optimal test log-likelihood is bounded above by 0. Thus, a p-percent increase in log-likelihood represents a p-percent improvement towards the optimal log-likelihood.

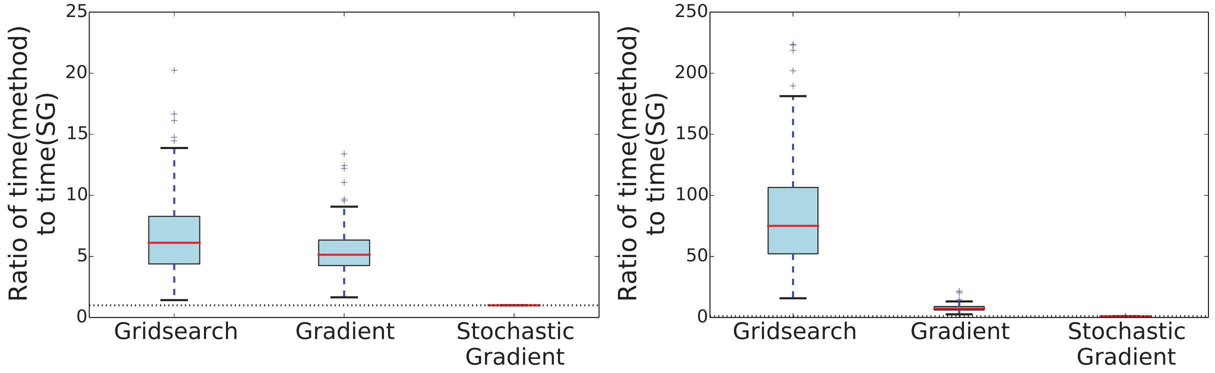


Figure 3.14: Inference times for the gridsearch, gradient, and stochastic gradient methods. Only the ratio of inference times between the methods and the stochastic gradient method are shown. Left: one-parameter kernel, Right: two-parameter kernel.

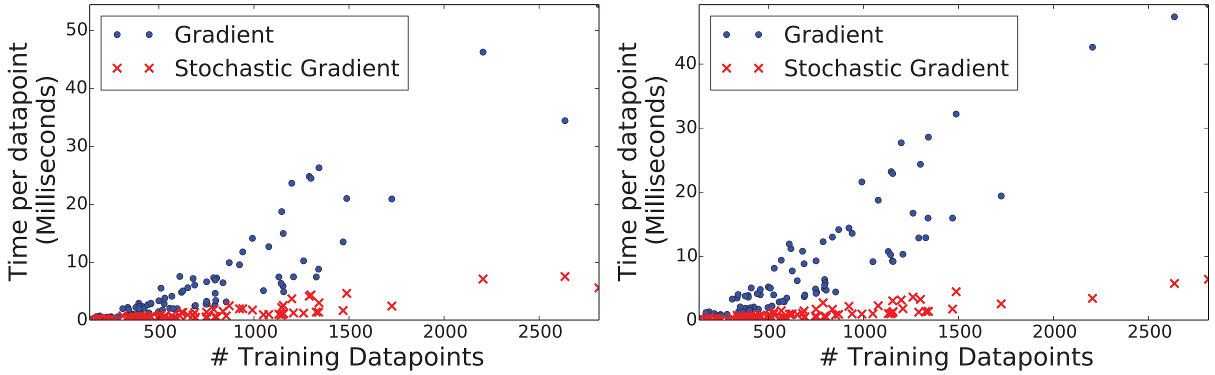


Figure 3.15: Inference times for the gradient and stochastic gradient methods for the one-parameter (left) and two-parameter (right) kernels, per dataset. The x-axis represents the number of datapoints in the training dataset, and the y-axis is the time-per-datapoint (in milliseconds) the method took to produce parameter estimates.

nels are shown. The $O(N^2)$ time complexity can be seen, with the runtime of the stochastic gradient method significantly smaller than the gradient method.

There are two reasons for why the gridsearch and gradient methods have similar computation times, given that the gradient method theoretically has a lower time complexity. First, the predefined search space for the gridsearch method can be used to optimize its implementation via vectorized code. Because the points where gradients are evaluated are unknown in gradient-based methods, such optimization is not possible. Second, the predefined search space is arbitrary in size, thus the runtime can be set arbitrarily large or small. For example,

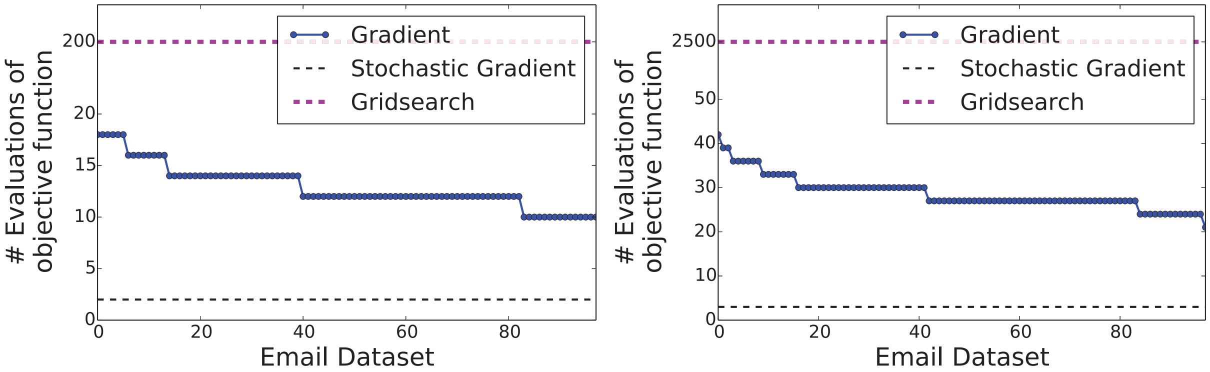


Figure 3.16: Number of evaluations of the objective function for each inference method (y-axis), across all email datasets (x-axis). Left: one-parameter kernel, Right: two-parameter kernel.

a search space of 2500 configurations was used for the two-parameter kernel, and 200 for the one-parameter kernel (as described in Section 3.5.3). For the two-parameter kernel, the increased search space resulted in the method being approximately 7 times slower than the gradient method, and 70 times slower than the stochastic gradient method.

To give a more theoretical and unbiased (in terms of implementation) comparision, in Figure 3.16 the number of times the (gradient of the) sequential log-likelihood was evaluated across the different methods is shown. As discussed earlier in this section, both gridsearch and stochastic gradient methods require a fixed number of evaluations, shown as the dashed horizontal lines in the figure. Only the gradient method varies in the number of evaluations; the x-axis in the plots represent the different datasets, and are sorted in descending order of evaluations. For both kernels, the number of evaluations required by the gradient method is much smaller than the gridsearch method (given the previously-discussed search spaces) but larger than the stochastic gradient method.

The main result of the experiments in this section is that, while the parameter estimates across all methods are similar and produce similar test log-likelihoods, the stochastic gradient method can obtain such estimates in a fraction of the time by taking a single pass through the training dataset.

3.5.5 Fitting of the One- and Two-Parameter Kernels

The experiments in this section explore how well the one- and two-parameter kernels are able to accurately fit the data. Motivated by the results from previous sections, we will use only the stochastic gradient inference method for estimating parameter values which optimize the sequential log-likelihood. By fixing the objective function and optimization method, a fair comparison between the kernel models is provided.

Experimental Setup

The setup of this experiment differs from previous experiments in this chapter in that a training or test dataset is not established. Instead, we treat the entire data as originating from a data stream where new data is continually being observed. This is similar to how the stochastic gradient method treated the training data from Section 3.5.4, and is somewhat analogous to leave-m-out cross validation applied in a sequential manner. Most importantly, this setup explores the model's capability of fitting to new data in real time.

Each of the 99 email datasets are split chronologically into 10% "initial" and 90% "streaming" data. The initial dataset is used by the models for initializing their parameter values, treating it as a training dataset. The streaming data is then partitioned into minibatches, with each minibatch covering emails across a 24-hour period. Let m_t represent the tth minibatch, with m_0 the minibatch of initial data. Once the models are initialized using the initial data m_0, each minibatch $\{m_t : t \geq 1\}$ is processed in sequence by doing the following:

1. The evaluation metric with respect to m_t is calculated (i.e. the log-likelihood of each datapoint in m_t under the model, with parameters estimated from data up to, but not including, m_t).

2. The model considers m_t as part of historical data, and updates parameters accordingly.

This procedure simulates a scenario where the model is updated daily, and the evaluation metric a measure of how well the model is able to describe new data as they arrive.

Stochastic Gradient Inference

The parameters of both the one- and two-parameter kernel models are estimated using the stochastic gradient method from Section 3.4.3. As in the experiments of Section 3.5.4, the stepsizes $\{\gamma^{(t)}\}$ used across gradient updates is determined using the Adadelta algorithm. The initialization of the kernel parameter(s) are set as in the previous experiment.

Baseline Model: Bernoulli mixture models

In addition to comparing the two kernel models, their performances are compared to the Bernoulli mixture model described in Section 2.1.2. We chose to compare against Bernoulli mixture models due to the similarities between the two described in Section 3.2.2 (i.e. the kernel models can be thought of as a nonparametric BMM).

A separate BMM is trained for each of $K = 5, 10, 30$, and 50 mixture components. First, each model is initialized via the EM algorithm over the 10% initial data (Dempster et al., 1977). This is done by running the EM algorithm ten separate times, each time with a random initialization. The EM algorithm terminates either after 100 iterations, or after the difference in parameter updates falls below 0.001, whichever comes first. The BMM initializes its parameters based on the run that resulted in the largest log-likelihood over the 10% initial data. These parameters will then be iteratively updated as each minibatch of streaming data is observed.

We use two different approaches for updating the BMM parameters in the presence of a new minibatch of data. The first approach is to restart the EM algorithm and apply it to all historical data, running until the same convergence criteria as with the initial data is reached (100 iterations, or an updated parameter difference below 0.001). The EM algorithm is run a single time, and is initialized to the estimated parameter values from the previous run.

The second approach is to use an online variant of EM, such as stepsize EM (Liang and Klein, 2009), that is designed to handle streaming data. Under this approach, the model parameters are updated with a single iteration of E and M steps once a new minibatch of data is observed. The M-step of stepsize EM is identical to that of traditional EM: parameter estimates are functions of the expected *sufficient statistics* for the mixture components, maximizing the log-likelihood of the data assumed to have generated such statistics. The difference in stepwise EM is within the E-step, where the expected sufficient statistics are generated not from the entire historical data, but from a convex combination of the last calculated sufficient statistics ($s^{(t-1)}$) and those from m_t ($s^{(t)}$):

$$s_{kl} \leftarrow \gamma_t s_{kl}^{(t)} + (1 - \gamma_t) s_{kl}^{(t-1)}$$

where γ_t is the "stepsize schedule", similar to that in stochastic gradient updates, and takes the form of $(1+t)^{-\alpha}$ (Liang and Klein, 2009). The subscript kl represents the lth expected sufficient statistic for component k. For the BMM, the expected sufficient statistics are $s_{kd} \equiv \sum_{i=1}^{N} z'_{ki} x_{id}$ for each dimension d, where $z'_{ki} \equiv P(\text{component } k | x_i, \text{current parameters})$. For each value of K, three separate stepwise EM procedures are performed, one for each stepsize parameter $\alpha \in \{0.25, 0.50, 0.75\}$.

Evaluation Metric: Minibatch Sequential Log-Likelihood

As the models process each minibatch of streaming data in turn, the log-likelihood of each datapoint x_i is calculated, given the model trained on minibatches before x_i. The evaluation metric is defined to be the *minibatch sequential log-likelihood*, the averaged log-likelihood across all datapoints in all minibatches, and has the following form:

$$\frac{1}{N - |m_0|} \sum_{t=1}^{T} \sum_{i \in m_t} \log P^{(t)} \left(x_i \middle| \bigcup_{t'=0}^{t-1} m_{t'}, \Omega \right) \tag{3.13}$$

where m_0 represents the initial data, T the number of minibatches, m_t the tth minibatch, and $P^{(t)}$ the probability distribution given estimated model parameters Ω from data up to, but not including, minibatch m_t. This is very similar in structure to the sequential log-likelihood from Section 3.3, except for the averaging and accounting for minibatches.

An alternative evaluation metric would be to follow the previous experiments in this chapter and evaluate the test log-likelihood given a model trained over training data. While we expect the results to be similar for this case, we chose the above evaluation metric and experimental setup as it is more suitable to real-world datasets where datapoints arrive sequentially (e.g. as the email inbox owner composes additional emails).

Results

The minibatch sequential log-likelihood defined by Equation 3.13 for each model is displayed in Figure 3.17. Specifically, the recorded values are the percent differences in minibatch sequential log-likelihood between the different models and the one-parameter kernel. Thus, larger positive values represent better adaptation of the model to the streaming data, compared to the one-parameter kernel. For each Bernoulli mixture model trained, only the model yielding the largest log-likelihood across the batch and stepsize EM inference methods for a specific value of K is recorded. The results in Figure 3.17 indicate that a significant increase in log-likelihood is achieved when using the two-parameter kernel to model email recipients, compared to both the one-parameter kernel and Bernoulli mixture models.

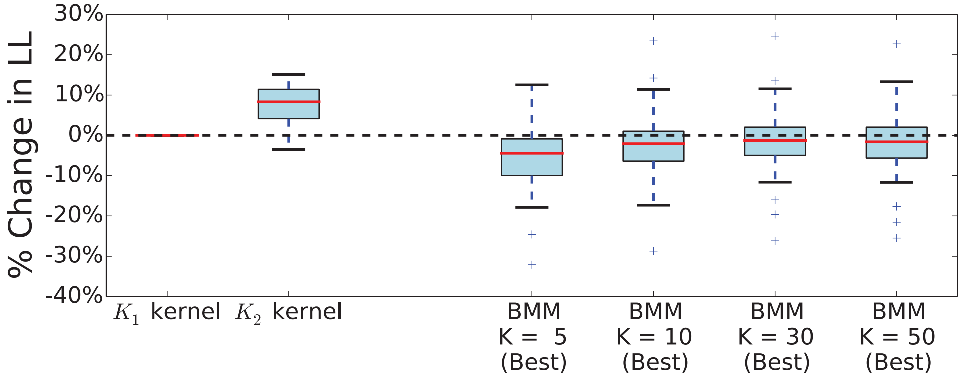


Figure 3.17: Difference in minibatch sequential log-likelihood between the models and the one-parameter kernel with optimal performance (the "K_1 (best)" model), across all datasets. For the Bernoulli mixture models, only the inference method (e.g. batch versus stepwise EM) yielding the optimal minibatch sequential log-likelihood is shown.

We give intuition behind these results for each model in turn. First, the minibatch sequential log-likelihood of the two-parameter kernel is significantly larger than the Bernoulli mixture model with any value of K. This result is expected, as the kernel is a nonparametric model that makes complete use of the historical data to shape its probability distribution. In addition to the general increase in log-likelihood, choosing to model the data with the kernel model has certain advantages over Bernoulli mixture models. The major limitation in a BMM is the choice of K; a poorly chosen value of K can easily lead to reduced predictive power. This, in part, explains the large variances seen for the Bernoulli mixture models in Figure 3.17. One could mitigate this problem by allowing the BMM to have a potentially infinite number of components (e.g. using a Chinese restaurant process as a prior over mixing proportions (Aldous, 1985)). However, doing so transforms the BMM into a nonparametric model, similar to kernel model, with the disadvantage of a significantly more complex inference method (e.g. Gibbs sampling over component assignments of historical datapoints, initializing parameters for "new" components, etc).

There are some advantages of the BMM over kernel models, however. When using stepsize EM to infer the parameters for a Bernoulli mixture model, parameter updates given a minibatch of size M takes $O(MK)$ time; the log-likelihood of each of the M datapoints under

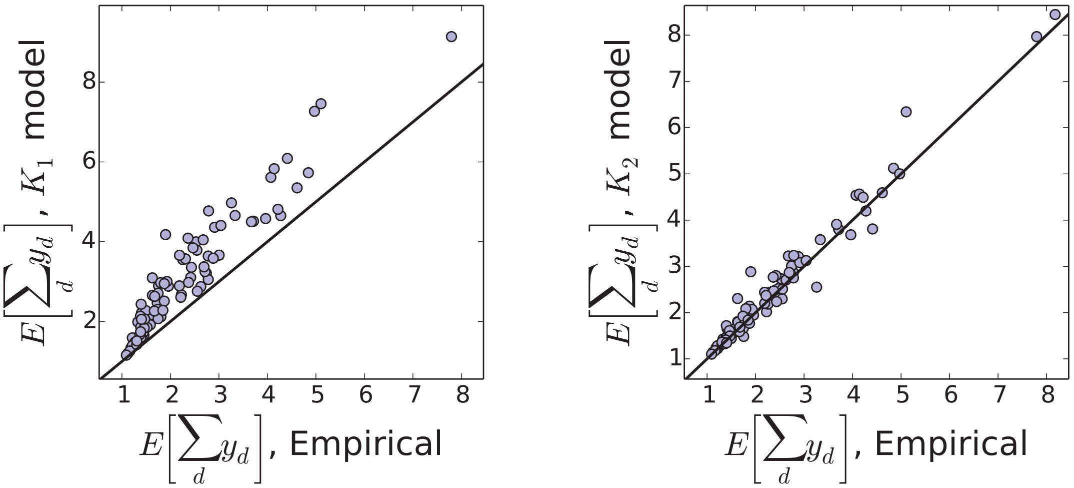


Figure 3.18: Expected density of $y \in \{0,1\}^D$, both empirically (x-axis) and under the kernel models (y-axis). Each point represents a different dataset. Left: one-parameter kernel. Right: two-parameter kernel.

each of the K components must be calculated. This is a more efficient update than the kernel's $O(MN)$ complexity of a stochastic gradient update. Additionally, BMMs require only $O(KD)$ space to store its parameters; the historical datapoints can be forgotten once the model is updated (assuming the inference method is stepwise EM).

Comparing the minibatch sequential log-likelihood of our kernel to the one-parameter kernel in Figure 3.17, the two-parameter kernel is generally a better fit to the streaming data. This general increase is due largely to the flexibility of the two-parameter model in accounting for the sparsity of the data. To illustrate this, we refer back to the discussion of data sparsity from Section 3.2.1. Let the *density* of a binary vector y be defined as $\sum_{d=1}^{D} y_d$. The expected value of this quantity under the two-parameter model (and similarly the one-parameter model by setting $\lambda_0 = \lambda_1$) was given in Equation 3.6. Figure 3.18 compares this expectation with that under the empirical dataset: $\frac{1}{N}\sum_{i=1}^{N}\sum_{d=1}^{D} x_{id}$. The values of λ, λ_0, and λ_1 used in these calculations are derived by taking the transformation of the estimated values of θ, θ_0, and θ_1 with the "optimal" inference procedure (in terms of minibatch sequential log-likelihood), averaged across the last 20 minibatches.

The left plot in Figure 3.18 compares the expected density of y under the one-parameter kernel against the empirical data. This expectation under the model is systematically larger than that found empirically, for reasons described in Section 3.2.1. As a result, the one-parameter kernel places a disproportionate amount of probability mass over binary vectors that are more dense than is typically found in the data, lowering the log-likelihood of new datapoints with the same density as the historical data. In contrast, the right plot in Figure 3.18 shows that our two-parameter model is able to accurately reflect the sparsity of the different datasets in general. This allows the model to place more probability mass over binary vectors of similar sparsity as the data, leading to higher log-likelihoods of streaming datapoints that are similar in sparsity to the previously seen data.

3.6 Discussion and Future Work

In this chapter, we explored nonparametric probability distributions over multidimensional binary data via kernel densities. The idea of applying a kernel density over binary data is largely credited to Aitchison and Aitken (1976), who introduced a kernel with a single bandwidth parameter. We showed that the expected sparsity of a vector under this parameterization of the kernel is guaranteed be less than the datapoints represented by the kernel, resulting in probability mass being placed over non-representative values. We expanded upon the work in Aitchison and Aitken (1976) by introducing a two-parameter variant that is able to match the expected sparsity as these datapoints, allowing for a better fit to data.

For parameter inference, an objective function measuring the quality of parameter values was proposed, referred to as the sequential log-likelihood. A general stochastic gradient method for estimating the kernel's parameters was then derived based on optimization of this objective function. The procedure is general in that it can be applied to other objective functions (e.g. cross-validated log-likelihoods) and to kernel densities over any data type, not just those over binary vectors. We showed that this method is able to produce parameter estimates that are similar in value to estimates using traditional cross-validation methods, across a much shorter period of time.

While we introduced a stochastic gradient procedure for estimating the kernel parameters, there are some open problems regarding its application. Most importantly, a single update in the stochastic gradient framework requires $O(N)$ time, resulting in a $O(N^2)$ overall complexity when applied to an entire dataset as proposed in Section 3.5. One possible solution is to apply a second approximation to the objective function's gradient, in addition to that already introduced by stochastic gradient. For example, one can subsample the datapoints represented by the second sum in Equation 3.11. In the extreme case, constant-time updates can be achieved by subsampling a small number of $Q << N$ datapoints in this sum. However, we found that for even moderately-sized $Q \approx 100$, the approximation results in an extreme bias in parameter estimates, for reasons regarding the asymptotic properties of the kernel discussed in Section 3.5.3. An interesting direction for future work would be to develop techniques that could correct for this bias, as a function of Q and N, leading to a reliable iterative inference algorithm with constant-time updates.

A different direction of future work would be to change the inference procedure altogether, using methods similar to those for Gaussian or continuous kernel density estimation. Many approaches to bandwidth selection have been explored for such kernels, and are discussed in detail in Jones et al. (1996); Zambom and Dias (2012). For example, there are plug-in or "pilot" estimators which select bandwidth parameters optimizing an approximation of the theoretical mean squared error, where the approximation is built using an initial or "pilot" density estimate over data (Woodroofe, 1970; Sheather and Jones, 1991). Other methods give the bandwidth parameter the flexibility to vary across either datapoints (Breiman et al., 1977; Sain and Scott, 1996) or the parameter space itself (Loftsgaarden and Quesenberry, 1965). Each of these methods, if able to be adapted to multidimensional binary vector space, could result in much more efficient (e.g. closed-form) and flexible parameter updates.

3.7 Summary of Contributions

The primary contributions of this chapter include the following:

- We introduced a two-parameter variant of the multidimensional binary kernel in Aitchison and Aitken (1976) that is able to accurately model the sparsity of the data represented by the kernel.
- We explored an objective function (an alternative to traditional cross-validated log-likelihoods) that can be optimized in order to estimate kernel parameters: the sequential log-likelihood.
- We developed an efficient inference procedure based on stochastic gradient for estimating kernel parameters that optimize the sequential log-likelihood.
- We evaluated our two-parameter kernel against the one-parameter kernel and Bernoulli mixture models on 99 different email datasets. Results showed that generally

    a) optimizing the sequential log-likelihood to infer kernel parameters leads to similar estimates and test log-likelihood when optimizing cross-validated log-likelihoods.

    b) the stochastic gradient inference method produced similar quality parameter estimates to gridsearch and gradient methods, with the benefit of significantly reduced time complexity.

    c) our model generally held more predictive power (measured by log-likelihood of unseen datapoints) compared to the other models.

Chapter 4

Probabilistic Completion of Itemsets with Bernoulli Mixture Models

The previous chapter focused on the modeling of D-dimensional binary data (or "itemsets") using nonparametric probabilistic models. In this chapter, the problem of *itemset completion* is discussed — given a partially observed itemset $x_O \equiv \{x_d = 1 : d \in O\}$ for some subset of items $O \subset \{1, \cdots, D\}$, is it possible to infer which dimensions in $x_{\tilde{O}}$, where $\tilde{O} = \{1, \cdots, D\} \setminus O$, will also have a value of 1 (i.e. the remaining items in that itemset)[1]? Figure 4.1 illustrates the itemset completion problem – some dimensions are observed to have a value of 1, which make up x_O, and the task is to infer which unobserved dimensions in $x_{\tilde{O}}$ (marked with "?"), if any, will also have a value of 1.

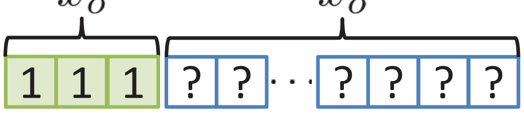


Figure 4.1: Example of the itemset completion problem. Given the observed x_O, the problem is to infer which dimensions in $x_{\tilde{O}}$ will also have a value of 1.

[1] While it is possible for values in x_O to be 0 (i.e. it is observed that an item is *not* in the itemset), such scenarios are uncommon. Thus, we focus on the assumption that all values in x_O are observed to be 1.

Being able to predict which additional items will be included with a partially observed itemset can be critical for some applications. For example, it is useful to predict additional symptoms a patient will endure based on their diagnosis of previous symptoms so preventive treatment can be provided. For online shopping websites, the past purchases of a customer can be useful in suggesting additional items the shopper is likely to buy.

The itemset completion problem also has application to email management systems. As the user is composing an email, recipients of that email are typically entered one-by-one. These recipients could be used to recommend additional, appropriate recipients to include in the email. Having such a recommendation system is valuable for preventing two types of mistakes, both potentially costly to the user: occlusion of relevant recipients (e.g. forgetting to add a coworker when discussing a project), and inclusion of irrelevant recipients (e.g. adding friend "Bob Smith" instead of boss "Bob Stevenson" for a status report). Being able to reduce the frequency of these errors can greatly enhance the user's experience and sense of security (with respect to their outgoing messages) with the email client, as observed by a large-scale experiment using the Gmail client (Roth et al., 2010).

In this chapter, we focus on the itemset completion problem in applications where x is modeled using the Bernoulli mixture model described in Section 2.1.2. The theory and application discussed are also directly applicable to the kernel models from Chapter 3, as they are a special form of a BMM (detailed in Section 3.2.2). Given an instantiation of a BMM with mixing proportions π and Bernoulli probabilities θ, predictions over which items are likely to be added to a partially observed itemset x_O can be derived from the form of the conditional distribution $P(x_{\tilde{O}}|x_O, \pi, \theta)$.

Two different approaches to the itemset completion problem are investigated, both of which make use of the conditional distribution of a Bernoulli mixture model. In the first approach, the value of the $(D - |O|)$-dimensional vector $x_{\tilde{O}}$ is estimated directly by calculating the *maximum-a-posteriori* (MAP) instantiation[2], defined as $\hat{x}_{\tilde{O}} \equiv \arg\max_{x_{\tilde{O}}} P(x_{\tilde{O}}|x_O, \pi, \theta)$. The second approach, instead of partitioning the $D - |O|$ possible items based on an estimated value of $x_{\tilde{O}}$, ranks the possible items by their likelihood of being included with the partially observed itemset.

The rest of the chapter is structured as follows. We begin with an overview of previous methods that address the itemset completion problem (focusing on the task of email recipient recommendation) in Section 4.1. Section 4.2 discusses the MAP approach to itemset completion, while Section 4.3 investigates the ranking approach. In Section 4.4, we measure the effectiveness of both approaches using the BMM and kernel models. The chapter concludes with a discussion and summary of contributions.

4.1 Related Work

Perhaps the most common approach to the itemset completion problem is *ranking*, where unobserved items are ranked according to their likelihood of being included in a partially observed itemset. A significantly large literature exists regarding the general problem of ranking — here we will review work specific to the task of itemset completion.

The work of Rudin et al. (2011) uses a *sequential event prediction* framework for ranking, where predictions over the next item to be observed in a partial itemset is based on a ranking of association rules (derived from co-appearance patterns of historical data). This framework was used for the task of online shopping cart recommendations. Letham et al. (2013) expand upon this work by using an optimization-based approach, minimizing a metric referred to as empirical risk or error. Here, the framework was applied in the contexts of patient symptom diagnosis and email recipient suggestions. A related strand of work is that of Grob et al.

[2] We follow the notation commonly used in machine-learning oriented literature, such as Liu and Ihler (2013). In other literature, particularly for graphical models, the problem is referred to as the *most-probable-explanation* (MPE) (Darwiche, 2009, Section 10.1).

(2009), where overlapping clusterings of an interaction graph are discovered and used in the contexts of recipient recommendation for mobile text messaging and Facebook event invites. These methods are similar in that rankings are based on measures derived from the empirical statistics of historical data — a probabilistic approach is not used.

Specific to the context of email recipient recommendation, Roth et al. (2010) ranks potential email recipients using only ego-centric interaction networks derived from historical interactions between an individual and different sets of contacts. The work of Pal and McCallum (2006) also addresses recipient recommendation by using a combination of the ego-centric interaction network and subject/body text. When taking into account the aggregated interaction network across multiple accounts (e.g. emails across a business organization) in addition to email text, more sophisticated ranking algorithms for the purpose of recipient recommendation (e.g. recommending "expert" recipients that can answer the email's subject matter) can be made (Bar-Yossef et al., 2008; Carvalho and Cohen, 2008; Graus et al., 2014). While some of these works used a probabilistic approach for recommending recipients, they differ from the approaches explored in this chapter in that the modeling of recipients was based on text and other metadata included in the email.

In this work, rankings are based on a probabilistic interpretation of how itemsets are generated, as opposed to using empirical co-appearance patterns or joint probabilistic models including the email's metadata. In particular, future items are ranked based on their marginal probabilities in the conditional distribution induced by a partially observed itemset. This could be viewed as a special form of item-based collaborative filtering (Sarwar et al., 2001), where future items are ranked based on their similarity (as per a similarity measure) to the currently observed items. Under this framework, the similarity measure would be the marginal probabilities under the conditional distribution. Using the Bernoulli mixture model to derive the conditional distribution and generate rankings is also analogous to cluster-based models of collaborative filtering (Breese et al., 1998), which use latent clusterings of itemsets to derive scores used in ranking.

As an alternative to the ranking task, we investigate a second approach to itemset completion — finding the most-probable configuration of unobserved items given the partially observed itemset. This approach instead *partitions* the possible items into those likely to be added to the itemset and those unlikely. Such a configuration of variables is referred to as the maximum-a-posteriori (MAP) instantiation, and is a well-studied problem in general (Darwiche, 2009; Liu and Ihler, 2013; Park and Darwiche, 2004). In this chapter, several approximation algorithms are applied to estimate the MAP instantiation.

In addition, the structure of the BMM is exploited to analytically solve for a significant proportion of dimensions, reducing the dimensionality prior to estimation. This is done using *dead-end elimination* (Desmet et al., 1992), which reduces the dimensionality of an objective function by finding optimal configurations of subsets of dimensions. The dead-end elimination approach is typically used in the context of protein sequencing; here, it is used to reduce the dimensionality of the MAP instantiation problem.

4.2 Itemset Completion via MAP Estimation

The first approach to itemset completion we discuss is finding the MAP instantiation with respect to a probability distribution over x, defined as

$$\hat{x}_{\tilde{O}} \equiv \arg\max_{x_{\tilde{O}}} P(x_{\tilde{O}} | x_O, \Omega) \tag{4.1}$$

where x_O is the partially observed itemset, $x_{\tilde{O}}$ the $(D-|O|)$-dimensional vector of unobserved dimensions, and Ω the model parameters (assumed to be known or estimated from a set of training data). By calculating the value of $\hat{x}_{\tilde{O}}$, the unobserved items are effectively partitioned into ones likely to be in the completed itemset (values of $d \in \tilde{O}$ where $\hat{x}_d = 1$) and those not likely to be added (values of $d \in \tilde{O}$ where $\hat{x}_d = 0$). In the context of email, $\hat{x}_{\tilde{O}}$ represents the most-likely configuration of additional recipients to include in an email, given its current list of recipients.

When modeling x with a Bernoulli mixture model, calculating conditional distributions of the form $P(x_{\tilde{O}}|x_O, \pi, \theta)$ is straightforward — recall from Section 2.1.2 that the conditional distribution of a BMM is defined as

$$P(x_{\tilde{O}}|x_O, \pi, \theta) = \sum_{k=1}^{K} \left[\frac{\pi_k P(x_O|\theta_k)}{\sum_{k'=1}^{k} \pi_{k'} P(x_O|\theta_{k'})} \right] P(x_{\tilde{O}}|\theta_k) = \sum_{k=1}^{K} \pi'_k P(x_{\tilde{O}}|\theta_k) = P(x_{\tilde{O}}|\pi', \theta')$$

which is also a BMM where π' are the mixing proportions re-weighted by the likelihood of x_O under each component, and θ' equivalent to θ with the columns corresponding to x_O removed. Thus, the MAP instantiation $\hat{x}_{\tilde{O}}$ in Equation 4.1 for a BMM is equivalent to

$$\begin{aligned} \hat{x}_{\tilde{O}} &\equiv \arg\max_{x_{\tilde{O}}} P(x_{\tilde{O}}|\pi', \theta') \\ &= \arg\max_{x_{\tilde{O}}} \sum_{k=1}^{K} P(x_{\tilde{O}}, z = k|\pi', \theta') = \arg\max_{x_{\tilde{O}}} \sum_{k=1}^{K} \pi'_k \prod_{d \in \tilde{O}} {\theta'_{kd}}^{x_d} (1 - \theta'_{kd})^{1-x_d} \end{aligned} \tag{4.2}$$

where z is a latent variable indicating which of the K "components" $x_{\tilde{O}}$ is assumed to be generated from. If z was known, for example $z = k$, then the MAP problem would reduce to

$$\hat{x}_{\tilde{O}} = \arg\max_{x_{\tilde{O}}} \pi'_k \prod_{d \in \tilde{O}} {\theta'_{kd}}^{x_d} (1 - \theta'_{kd})^{1-x_d} = \arg\max_{x_{\tilde{O}}} \prod_{d \in \tilde{O}} f(x_d)$$

where $f(x_d) \equiv {\theta'_{kd}}^{x_d} (1 - \theta'_{kd})^{1-x_d}$ and π'_k can be ignored because it is constant with respect to $x_{\tilde{O}}$. This problem can be easily solved in $O(|\tilde{O}|)$ time by maximizing each $f(x_d)$ independently. However, the value of z is unknown, thus the probability density of the BMM marginalizes over all possible values of z. Because of the marginalization over a latent variable, the problem of calculating $\hat{x}_{\tilde{O}}$ is referred to as the *marginalized* MAP problem (Liu and Ihler, 2013)[3], a known $\mathrm{NP}^{\mathrm{PP}}$-complete problem (Park and Darwiche, 2004).

The marginalization of z significantly increases the complexity of solving for $\hat{x}_{\tilde{O}}$ as it removes the ability to factorize the likelihood into functions over small subsets of variables that can be independently solved. Thus, for the BMM, directly computing $\hat{x}_{\tilde{O}}$ requires an enumeration over all $2^{|\tilde{O}|}$ possible values of $x_{\tilde{O}}$. This time complexity quickly becomes intractable for even moderate values of $|\tilde{O}|$. Fortunately, the form of the probability density in a Bernoulli mixture model allows for some dimensions in $\hat{x}_{\tilde{O}}$ to be analytically solved efficiently, and is discussed in Section 4.2.1. Efficient algorithms that approximate the value of $\hat{x}_{\tilde{O}}$ for dimensions that cannot be analytically solved are then provided in Section 4.2.2.

As the conditional distribution of a BMM is itself a BMM, Sections 4.2.1 and 4.2.2 will address the issues of solving for the more general problem $\hat{x} = \arg\max_x P(x|\pi, \theta)$. The problem can then be translated back into the original MAP problem of Equation 4.2 by replacing $P(x|\pi, \theta)$ with $P(x_{\tilde{O}}|\pi', \theta')$.

4.2.1 Dimensionality Reduction

Let $S \subset \{1, \cdots, D\}$ be a set of indices and $U \equiv \{1, \cdots, D\} \setminus S$, such that $x = x_S \cup x_U$. In this section, the structure of the Bernoulli mixture model is exploited to analytically solve for $\hat{x}_S \subset \hat{x}$, the values of x_S under the MAP instantiation $\hat{x} = \arg\max_x P(x|\pi, \theta)$. By analytically solving for $\hat{x}_S$, the dimensionality of the MAP problem reduces to $|U| \leq D$, where "U" indicates unsolved. The reduced dimensionality allows for the MAP approximation algorithms of Section 4.2.2 to operate both more efficiently and accurately (as fewer dimensions have to be approximated).

[3] In literature which refers to the MAP problem as the MPE problem, the marginalized MAP problem is referred to as the MAP problem (Darwiche, 2009, Section 5.2.4).

π_1	π_2	π_3
0.60	0.38	0.02

θ_1	0.72	0.98	0.33	0.21
θ_2	0.68	0.87	0.24	0.12
θ_3	0.33	0.72	0.54	0.09

Table 4.1: Example parameters for a Bernoulli mixture model

The idea behind the dimensionality reduction is straightforward: we identify dimensions where the probability of x according to the BMM (Equation 2.4) is always higher when set to a specific configuration, regardless of the values of other dimensions. This process of identifying optimal configurations of a subset of dimensions in x is an application of *dead-end elimination* (Desmet et al., 1992).

As an example, consider the Bernoulli mixture model portrayed in Table 4.1. The Bernoulli probabilities for x_2 (the second column of θ) are all greater than $\frac{1}{2}$, thus the probability of x under the BMM (Equation 2.4) is always larger when $x_2 = 1$:

$$\begin{aligned} P(x_{-2}, x_2 = 1|\pi, \theta) &= \sum_{k=1}^{K} \pi_k \theta_{k2} \prod_{d \neq 2} P(x_d|\theta_k) \\ &> \sum_{k=1}^{K} \pi_k (1 - \theta_{k2}) \prod_{d \neq 2} P(x_d|\theta_k) = P(x_{-2}, x_2 = 0|\pi, \theta) \end{aligned}$$

Thus, it must be the case that $\hat{x}_2$, the value of x_2 under the MAP instantiation $\hat{x}$, is equal to 1. A similar approach can be applied to show that $\hat{x}_4 = 0$, as the Bernoulli probabilities for x_4 is always less than $\frac{1}{2}$.

The examples using x_2 and x_4 in Table 4.1 are trivial in that the Bernoulli probabilities were always less than or greater than $\frac{1}{2}$. There may be additional dimensions lacking this property in which the probability is always larger when set to a particular value — x_3 from

Table 4.1 is such an example. In general, for some dimension d, we can set

$$\hat{x}_d = \begin{cases} 0 & \text{if } \min_{x_{-d}} \Big[P(x_d = 0, x_{-d} | \pi, \theta) - P(x_d = 1, x_{-d} | \pi, \theta) \Big] \geq 0 \\ 1 & \text{if } \min_{x_{-d}} \Big[P(x_d = 1, x_{-d} | \pi, \theta) - P(x_d = 0, x_{-d} | \pi, \theta) \Big] \geq 0 \\ \text{inconclusive} & \text{otherwise} \end{cases}$$

For the Bernoulli mixture model, the above conditions can be checked using the following inequalities (derivations of which are shown in Appendix B):

$$\hat{x}_d = \begin{cases} 0 & \text{if } \sum_{k:\theta_{kd} \leq \frac{1}{2}} \pi_k (1 - 2\theta_{kd}) \alpha_{kd} + \sum_{k:\theta_{kd} > \frac{1}{2}} \pi_k (1 - 2\theta_{kd}) \beta_{kd} \geq 0 \quad \text{(4.3a)} \\ 1 & \text{if } \sum_{k:\theta_{kd} \leq \frac{1}{2}} \pi_k (2\theta_{kd} - 1) \beta_{kd} + \sum_{k:\theta_{kd} > \frac{1}{2}} \pi_k (2\theta_{kd} - 1) \alpha_{kd} \geq 0 \quad \text{(4.3b)} \\ \text{inconclusive} & \text{otherwise} \end{cases}$$

where

$$\alpha_{kd} \equiv \min_{x_{-d}} \prod_{d' \neq d} {\theta_{kd'}}^{x_{d'}} (1 - \theta_{kd'})^{1 - x_{d'}} \qquad \beta_{kd} \equiv \max_{x_{-d}} \prod_{d' \neq d} {\theta_{kd'}}^{x_{d'}} (1 - \theta_{kd'})^{1 - x_{d'}}$$

Thus, a single dimension d that satisfies either Equation 4.3a or Equation 4.3b is considered solved, i.e. $d \in S$. This inequality test can be applied to each dimension d independently to construct the index sets S and U in $O(KD)$ time.

A potential issue with Equations 4.3a and 4.3b is that the inequalities are too restrictive. For example, the probability of x under the BMM parameters of Table 4.1 is always greater when $x_1 = 1$, as opposed to $x_1 = 0$. However, the lower-bound of the difference in probability (the left-hand-side of Equation 4.3b, derived in Appendix B) is ≈ -0.0001. Thus, x_1 will not be identified as a solved dimension in the MAP instantiation. A similar problem arises for dimensions where the probability is generally increased when set to a specific configuration, but slightly decreases for very rare values in other dimensions of x.

To prevent this problem, we propose to instead use the following set of inequalities for solving dimensions in $\hat{x}$:

$$\hat{x}_d \leftarrow \begin{cases} 0 & \text{if} \sum_{k:\theta_{kd} \leq \frac{1}{2}} \pi_k(1-2\theta_{kd})\alpha_{kd} + \sum_{k:\theta_{kd} > \frac{1}{2}} \pi_k(1-2\theta_{kd})\beta_{kd} \geq -\epsilon \quad (4.4\text{a}) \\ 1 & \text{if} \sum_{k:\theta_{kd} \leq \frac{1}{2}} \pi_k(2\theta_{kd}-1)\beta_{kd} + \sum_{k:\theta_{kd} > \frac{1}{2}} \pi_k(2\theta_{kd}-1)\alpha_{kd} \geq -\epsilon \quad (4.4\text{b}) \\ \text{inconclusive} & \text{otherwise} \end{cases}$$

where ϵ is a small positive number. A dimension d that satisfies Equation 4.4a (similarly for Equation 4.4b) will have $\hat{x}_d = 0$ with high probability — if it is the case that $\hat{x}_d = 1$, then the MAP instantiation with $\hat{x}_d$ mistakenly set to 0 will be within ϵ probability of the true MAP instantiation[4]. In the experiments of Section 4.4, we set $\epsilon = 0.005$; we empirically found that this value allows for solving a significant proportion of dimensions in $\hat{x}$ with a low error rate (as suggested by the experimental results). Because the values of dimensions that satisfy Equation 4.4a or Equation 4.4b are now approximated, we will refer to them as analytically *removed* instead of solved.

Once $\hat{x}_S$ is analytically removed, the MAP problem of Equation 4.2 reduces to solving

$$\begin{aligned} \hat{x}_U &\equiv \arg\max_{x_U} P(x_U, x_S = \hat{x}_S|\pi,\theta) = \arg\max_{x_U} P(x_S = \hat{x}_S|\pi,\theta)P(x_U|x_S = \hat{x}_S,\pi,\theta) \\ &= \arg\max_{x_U} P(x_U|x_S = \hat{x}_S,\pi,\theta) = \arg\max_{x_U} P(x_U|\pi',\theta') \end{aligned}$$

where the conditional distribution in the last equality again takes the form of a Bernoulli mixture model with updated parameters that condition on $x_S = \hat{x}_S$. Thus, the original MAP problem over D-dimensional x can be solved by 1) finding $\hat{x}_S$, 2) conditioning on $\hat{x}_S$, then 3) solving the $|U|$-dimensional MAP problem over this conditional distribution to find $\hat{x}_U$.

In the worst case, $S = \emptyset$ and $x_U = x$, with no reduction in dimensionality. However, itemset data is typically *sparse* in that the size of an itemset is typically much smaller then the set

[4] Assuming the other $D-1$ dimensions are correctly set.

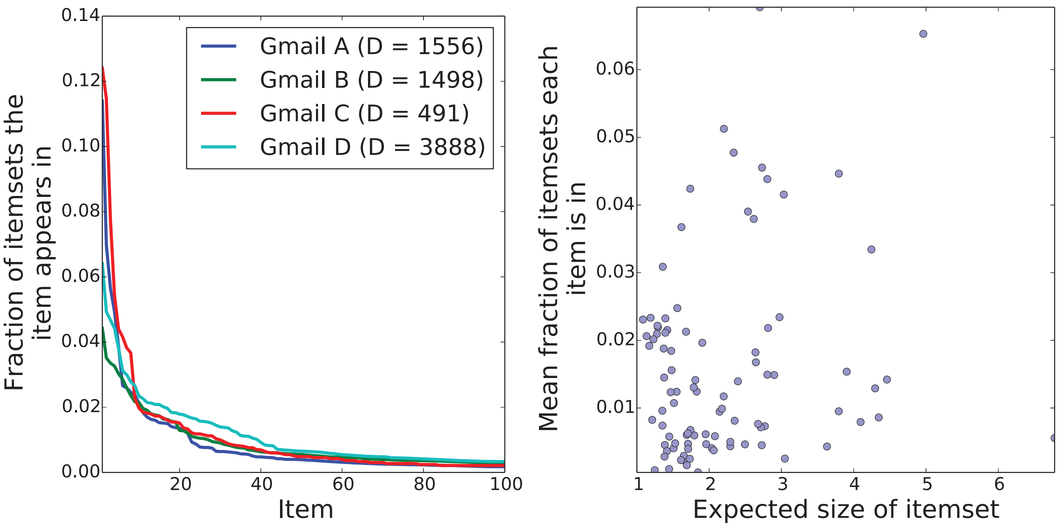


Figure 4.2: Sparsity of itemset data across different email datasets, described in Section 4.4. Left: Gmail datasets, Right: all datasets.

of all possible items. Additionally, only a small number of items are commonly seen across multiple itemsets, with the large majority of other items rarely seen. Figure 4.2 illustrates these sparsity properties for the email datasets described in the experiments of Section 4.4. The left plot shows the fraction of itemsets each item appears in, for each of the Gmail datasets. Only the 100 most frequent items are shown — the hundreds to thousands of items not plotted for each dataset appear in less than 1% of observed itemsets. The right plot shows the expected size of an itemset (x-axis) versus the averaged fraction of itemsets that items appear in (y-axis), across all email datasets.

The datasets shown in Figure 4.2 are extremely sparse, allowing for a significant proportion of dimensions to be analytically removed. In practice, Equation 4.4a often selects a large proportion of items that rarely appear, setting their optimal values in $\hat{x}$ to 0. Equation 4.4b, while less effective in reducing dimensionality, is useful in cases where multiple items frequently co-appear in the same itemset. For example, suppose that $P(x_e = 1|x_d = 1, \pi, \theta) \approx 1$. If $x_d \in x_O$ (the partially observed itemset), then it is very likely that item e will be in the itemset as well. Equation 4.3b will identify this dimension in the conditional distribution and set $\hat{x}_e = 1$.

4.2.2 Approximation Strategies

Even with reducing the dimensionality of the MAP problem in Section 4.2.1, the number of remaining dimensions to solve for is typically too large to compute exactly, as the computation takes $O(2^{|U|})$ time. In this section, we give a brief overview of some methods which approximate the MAP instantiation defined by Equation 4.2, focusing on the general MAP problem of approximating $\hat{x}$ given the conditional BMM parameters. The same approach can be directly applied to the case of solving for $\hat{x}_U$ in Section 4.2.1, after conditioned on $\hat{x}_S$ (as the conditional distribution is also a BMM).

Monte Carlo Estimation

Perhaps the most straightforward method for estimating the MAP instantiation is to use a Monte Carlo search, where the probability density $P(x|\pi, \theta)$ is approximated with many samples from the distribution. Let $Y = \{y_1, y_2, \cdots, y_M\}$ represent M samples generated from the BMM, where $y_m \in \{0, 1\}^D$. Given the samples Y, a Monte Carlo estimate of $\hat{x}$ can be made using the sample yielding the highest probability:

$$\hat{x} \approx \arg\max_{y_m} P(y_m|\pi, \theta)$$

Generating a single sample takes $O(D)$ time, according to the generative procedure of a BMM. Thus, this approximation method takes a total of $O(MD)$ time. As M increases, so does the quality of the approximation — in the limit where all 2^D unique values of x are sampled, the approximation will become exact. However, the value of M is typically much smaller than 2^D, thus only a small fraction of the probability space is explored. For the experimental results in Section 4.4.3, the Monte Carlo estimation method is applied twice — once where $M = 100$ samples, and another where $M = 100000$ samples.

Marginal method: thresholding marginal probabilities

The Monte Carlo estimation method is non-deterministic — different estimations are provided each time the procedure is run. A deterministic method for estimating $\hat{x}$ is to solve for the univariate MAP instantiation $\hat{x}_d$ across each dimension d independently. That is, set

$$\hat{x}_d \leftarrow \begin{cases} 0 & \text{if } P(x_d = 1|\pi, \theta) = \sum_{k=1}^{K} \pi_k \theta_{kd} \leq \frac{1}{2} \\ 1 & \text{if } P(x_d = 1|\pi, \theta) = \sum_{k=1}^{K} \pi_k \theta_{kd} > \frac{1}{2} \end{cases}$$

Estimating $\hat{x}$ according to this method can be calculated in $O(KD)$ time. This is advantageous over the Monte Carlo method in that the computation time is fixed — it depends only on properties of the Bernoulli mixture model, not a variable number of simulated samples. However, this method is still clearly an approximation, as the dependencies induced by the marginalization of z in the BMM probability density are ignored.

Joint MAP estimation of (x, z)

Another deterministic method of estimating $\hat{x}$ is to not marginalize over the latent variable z, and instead find the joint MAP instantiation of x and z:

$$\hat{x}, \hat{z} = \arg\max_{\hat{x},\hat{z}} P(x, z|\pi, \theta) = \arg\max_{\hat{x},k} \pi_k \prod_{d=1}^{D} {\theta_{kd}}^{x_d} (1 - \theta_{kd})^{1-x_d}$$

The value of $\hat{x}$ is then used to approximate the MAP instantiation of Equation 4.2, ignoring the value of $\hat{z}$. Finding the joint MAP instantiation $(\hat{x}, \hat{z})$ is straightforward to compute (requiring $O(KD)$ time), as the above objective function to maximize is the product of $D+1$ univariate functions. This is an approximation to the true value of $\hat{x}$, as the value of $\hat{x}$ in the joint MAP instantiation is generally not consistent with the solution of Equation 4.2 (Darwiche, 2009, Section 10.1). One advantage of this method over the previous marginal method is that the estimated value of $\hat{x}$ is guaranteed to be the mode of one of the K mixture components of the BMM. Thus, the local structure of the BMM components are respected.

EM Algorithm

For Bernoulli mixture models, the EM algorithm (Bishop, 2006, Section 9.3.3) is an effective procedure for finding maximum-likelihood estimates of π and θ given data x (Juan and Vidal, 2002, 2004). The EM algorithm can also be used to provide "maximum-likelihood" estimates of $\hat{x}$ given the model parameters $\{\pi, \theta\}$ which (locally) maximize $P(x|\pi, \theta)$. Thus, the EM algorithm is an appropriate method for estimating the MAP instantiation.

The EM algorithm starts with an initial estimate of $\hat{x}$, and iterates between the following two steps until convergence (e.g. until the estimate of $\hat{x}$ is unchanged between two iterations):

E step: The expectation of the latent variable z is calculated with respect to the current estimate of $\hat{x}$. That is, $\gamma_k \equiv P(z = k|\hat{x}, \pi, \theta)$ is calculated for each value of $1 \leq z \leq K$.

M step: $\hat{x}$ is estimated such that the expected complete log-likelihood of the BMM, defined as $\sum_{k=1}^{K} \gamma_k \sum_{d=1}^{D} (\hat{x}_k \log \theta_{kd} + (1 - \hat{x}_k) \log(1 - \theta_{kd}))$, is maximized. The values of $\hat{x}$ are set such that

$$\hat{x}_d = \begin{cases} 0 & \text{if } \sum_{k=1}^{K} \gamma_k \log\left(\frac{\theta_{kd}}{1-\theta_{kd}}\right) \leq 0 \\ 1 & \text{if } \sum_{k=1}^{K} \gamma_k \log\left(\frac{\theta_{kd}}{1-\theta_{kd}}\right) > 0 \end{cases}$$

A single iteration of the EM algorithm described above takes $O(KD)$ time. Empirical findings showed that only a few iterations are needed before the algorithm converges on a value of $\hat{x}$. Two common issues inherent to the EM algorithm are 1) the initial estimate of $\hat{x}$ often affects the quality of the final estimate, and 2) the final estimate is guaranteed to *locally* maximize the likelihood function. We mitigate these two issues by running the EM algorithm 5 separate times in our experiments, initializing $\hat{x}$ in each run with a different sample from the BMM. The overall estimate of the MAP instantiation is then the EM instance that produced the most-probable final estimate $\hat{x}$. Another approach for initializing the EM algorithm is to use the output of the Monte Carlo method previously discussed — properties of this approach are explored in the experimental results.

Variational Message Passing

The last approach we explore for estimating the MAP instantiation is the variational message passing approach of Liu and Ihler (2013). This approach applies a variational approximation to the dual of the optimization problem in Equation 4.2, which can be solved using an iterative message passing algorithm. The algorithm uses the factor graph representation of the Bernoulli mixture model, defined as

$$q(x|\pi, \theta) \propto \sum_{z} f(z) \prod_{d=1}^{D} f(x_d, z) f(x_d) \tag{4.5}$$

Under this representation, the factors $f(z)$ and $f(x_d, z)$ map directly to the mixing proportions π and Bernoulli probabilities θ respectively. The $f(x_d)$ factors are the marginal beliefs in the possible values of x_d, and are updated in an iterative manner. The factor graph representation of the Bernoulli mixture model is shown in Figure 4.3.

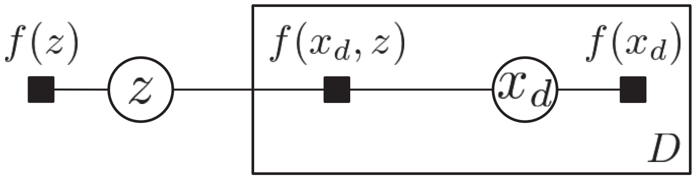


Figure 4.3: A factor graph representation of the Bernoulli mixture model.

We use the proximal message passing method of Liu and Ihler (2013), which formulates the marginal MAP problem as a sequence of marginalization problems, and is given in Algo-

Algorithm 4 Variational Message Passing

```
Input: BMM mixing proportions $\pi$, BMM Bernoulli probabilities $\theta$
Output: an estimated value of $\hat{x} = \arg\max_x P(x|\pi, \theta)$
  $f(z) \leftarrow \pi_z$
  $f(x_d, z) \leftarrow \theta_{zd}$
  $f^0(x_d) \leftarrow \sum_{k=1}^{K} f(z) f(x_d|z)$
  for $t = 1$ to $\infty$ do
    $m_{x_d \to z}(z) \leftarrow \sum_{x_d \in \{0,1\}} f^{t-1}(x_d) f(z, x_d)$
    $m_{z \to x_d}(x_d) \leftarrow \sum_{z=1}^{K} f(z, x_d) \frac{1}{m_{x_d \to z}(z)} \left( f(z) \prod_{d'=1}^{D} m_{x_{d'} \to z}(z) \right)$
    $f^t(x_d) \propto f^{t-1}(x_d) m_{z \to x_d}(x_d)$
    if convergence criteria is met then
      return $\hat{x} = \{x_d = \arg\max_{x_d} f^t(x_d) : 1 \leq d \leq D\}$
    end if
  end for
```

rithm 4. At the tth iteration, messages are passed between the z and $\{x_d\}$ variables, and the marginal factors $f^t(x_d)$ are updated based on the messages sent from z and the marginal factors $f^{t-1}(x_d)$ from the previous iteration. A single iteration of the message passing algorithm takes $O(KD)$ time, and is terminated when the maximum difference between marginal factors $f^t(x_d)$ and $f^{t-1}(x_d)$ is below a threshold (set to 0.01 in the following experiments).

4.3 Ranking Future Items

The task of finding the MAP instantiation $\hat{x}_{\tilde{O}}$ of the conditional distribution $P(x_{\tilde{O}}|x_O, \pi, \theta)$ is useful for predicting the outcome of $x_{\tilde{O}}$. By estimating $\hat{x}_{\tilde{O}}$, the unobserved items are effectively partitioned into items likely to be part of the completed itemset ($\hat{x}_d = 1$), and those unlikely ($\hat{x}_d = 0$).

A different and unrelated approach to the itemset completion problem is to instead *rank* the $D - |O|$ items not observed in x_O, such that higher-ranked items are more likely to be added to the partial itemset x_O than lower-ranked items. Such a ranking task is useful in scenarios where only the next item that will be included in the partial itemset needs to be predicted, instead of all remaining items at once (e.g. via finding the MAP instantiation). For example, in email recipient recommendation, it is useful to display to the end user an ordered list of the top M people the user is most likely to add to the email, based on those already added (Roth et al., 2010).

When modeling x with a Bernoulli mixture model, a natural ranking amongst the potential items to be added exists; such items can be ranked based on the following quantity:

$$s_d \equiv P(x_d = 1|x_O, \pi, \theta) = \sum_{k=1}^{K} \pi'_k \theta_{kd} \tag{4.6}$$

where $\{\pi'_k\}$ are the reweighted mixing proportions after conditioning on x_O, and s_d referred to as the *score* of item d when conditioned on the partially observed itemset. The score is defined as the marginal probability that $x_d = 1$ under the conditional distribution $P(x_{\tilde{O}}|x_O, \pi, \theta)$, thus ranking in descending order of score ranks the items by their marginal probabilities of being included in the partial itemset.

The time complexity to compute a ranking of items is $O(KD + D \log D)$, and is exact in the sense that the produced ranking is "optimal" given the BMM parameters. The $O(KD)$ time complexity is the time required to compute the scores across all $O(D)$ unobserved dimensions (each score taking $O(K)$ time to compute via Equation 4.6). The $O(D \log D)$ time complexity is due to the sorting of the scores to produce the ranking.

4.4 Experiments

We begin the experiments by discussing the email datasets used and the experimental setup in Sections 4.4.1 and 4.4.2 respectively. The experiments in Section 4.4.3 explore 1) the advantages gained when analytically removing dimensions in the MAP instantiation, and

2) the difference in performance between the different MAP estimation methods of Section 4.2.2. Lastly, Section 4.4.4 explores the quality of the ranking of items (as per Section 4.3) compared to a baseline ranking method.

4.4.1 Datasets

The email datasets used for these experiments are identical to those used in the experiments with kernels over multidimensional binary data in Section 3.5.1 — 99 email datasets are included across the Eckmann, Enron, Gmail, and Manufacturing email corpora described in Section 1.1. As with the experiments from Chapter 3, the experiments are run independently on each of the 99 email datasets.

4.4.2 Experimental Setup

The experiments of Sections 4.4.3 and 4.4.4 use the same experimental setup. For a single email dataset (one individual's inbox history), the data is split chronologically into 75% training and 25% test. The parameters of the Bernoulli mixture models and the one- and two-parameter kernel models of Chapter 3 are then estimated from the training data. The π and θ parameters of the BMM are estimated using the EM algorithm (Juan et al., 2004), with the value of K (the number of components) chosen via 10-fold cross-validation. Values of $K \in \{10, 20, 30, 50, 100\}$ were considered for the BMM. Very weak priors were placed over the BMM parameters — a symmetrical Dirichlet(1.01) prior over π and a Beta$(1.01, 1.01)$ prior over each element in θ. For the kernel models, the bandwidth parameter(s) are estimated by taking a single pass through the training data and using stochastic gradient ascent updates, as described in Sections 3.4.3 and 3.5.2.

Once the different models are trained, their parameters become fixed as the models are evaluated over the test data (evaluation metrics are explained in the following subsections). As the data is split chronologically, there may be some items (e.g. email recipients) in the test data that do not appear in the training data — such items are ignored, focusing only on those appearing in the training data. Figure 4.4 shows the number of training and test itemsets across all 99 email datasets, after test itemsets containing test-only items are removed.

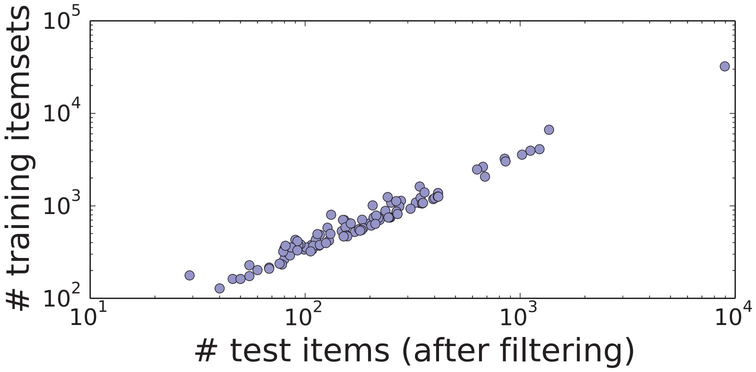


Figure 4.4: The number of test itemsets (after removing items that do not appear in the training data) versus the number of training itemsets, across all email datasets.

4.4.3 MAP Estimation

We first explore the utility of the dimensionality reduction described in Section 4.2.1 and the different MAP approximation algorithms of Section 4.2.2, when applied to test data. The quality of the MAP estimates is compared only between the different approximation methods — a comparison between the BMM and kernel models is discussed in Section 4.4.4.

Each itemset in the 25% test data is processed according to the following procedure:

1. The BMM and kernel models are conditioned on only the first item to appear in the itemset (e.g. the first of the email recipients). Thus, the partially observed itemset x_O contains only this item.

2. The $(D-1)$-dimensional MAP instantiation $\hat{x}_{\tilde{O}}$ is then estimated using the different approximation algorithms, with no reduction in dimensionality. The log-likelihood of $\hat{x}_{\tilde{O}}$ estimated under each model and algorithm is recorded.

3. Dimensionality reduction is applied by solving for a subset of dimensions $\hat{x}_S \subset \hat{x}_{\tilde{O}}$, as discussed in Section 4.2.1.

4. The $|U|$-dimensional MAP instantiation $\hat{x}_U \subset \hat{x}_{\tilde{O}}$ is estimated using the same approximation algorithms as before. The log-likelihood of $\hat{x}_{\tilde{O}} = \hat{x}_S \cup \hat{x}_U$ estimated under each model and algorithm is recorded.

We chose the following setup as it simulates the scenario where an email user is adding the first recipient to an email, and the email client is tasked to predict all other recipients of the email based on that first recipient.

Results

Figure 4.5 shows the percentage of dimensions that were analytically removed using Equations 4.4a and 4.4b, averaged across all test itemsets. The variances in the boxplots are over the different email datasets. For all models, a significant reduction in dimensionality occurred, with the BMM and two-parameter (K_2) kernel having at least 90% of dimensions removed on average. With the dimensionality of the MAP problem reduced by 10-fold, the approximation algorithms discussed in Section 4.2 become much more efficient by producing high-quality estimates in shorter amounts of time.

To see the advantage of dimensionality reduction, Figure 4.6 compares the quality of the MAP estimate when approximating $\hat{x}_{\tilde{O}}$ directly and first solving for $\hat{x}_S$ then approximating $\hat{x}_U$. Specifically, the mean difference in log-likelihood between the MAP estimates with and without dimensionality reduction is shown, where larger positive values indicate a higher-

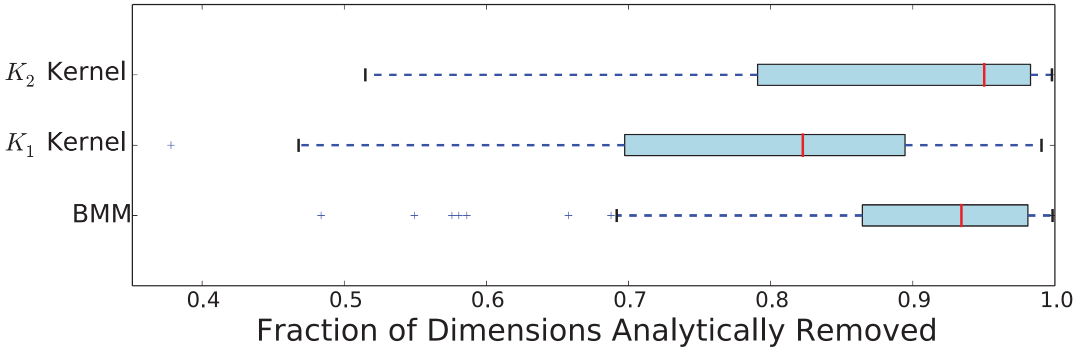


Figure 4.5: Fraction of dimensions analytically removed (x-axis) via Equations 4.4a and 4.4b, across different models (y-axis) and email datasets (plotted points).

quality MAP estimate when first solving for $\hat{x}_S \subset \hat{x}_{\tilde{O}}$. The boxplot variances are over the different email datasets. The abbreviated algorithm names are described in Table 4.2.

Abbreviated Name	**Estimation Method**
MC	Monte Carlo, using $M = 10000$ samples
M	Marginal
J	Joint
EM	EM algorithm, initialized with a random sample
MP	Variational message passing

Table 4.2: Abbreviation of MAP approximation algorithms.

For the Monte Carlo and joint estimation methods in Figure 4.6, the quality of the MAP estimate is largely unaffected by the dimensionality reduction. In select cases, the quality of the estimate is lowered after applying the dimensionality reduction. The message passing algorithm was also mostly unaffected, with a better estimate occasionally provided when applied to the reduced space. For the marginal and EM methods, the variance in the differences is much larger, with a slight positive trend across the email datasets.

The decreases in log-likelihood can be caused by a variety of reasons. For the Monte Carlo and EM algorithms, the differences can be explained by the quality of simulated samples used by the algorithm. If high-quality (resp. low-quality) samples in the complete (resp. reduced) dimensionality are produced, then it is unlikely that the algorithm in the reduced space will achieve a higher log-likelihood than in the complete space. Another explanation

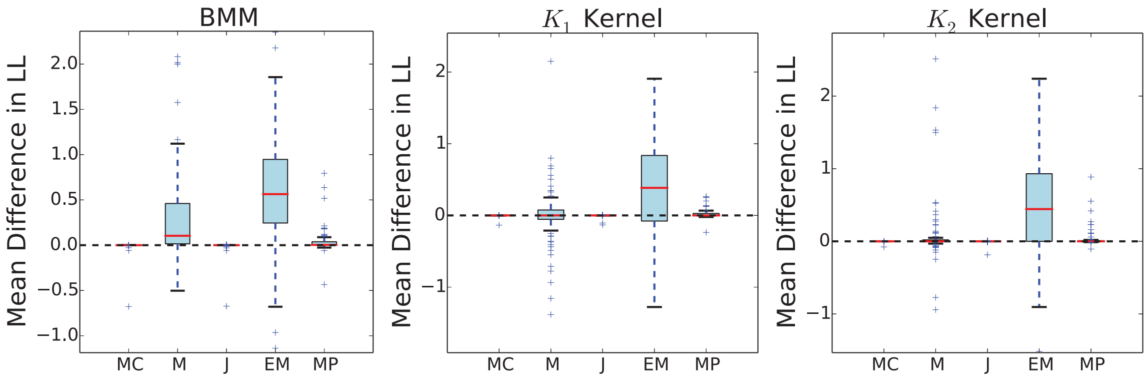


Figure 4.6: Differences in log-likelihood (y-axis) when applying each MAP approximation algorithm (x-axis) in the reduced- and full-dimensionality spaces. Larger positive values indicate higher log-likelihoods found after applying the dimensionality reduction. Direct comparison between the different algorithms is saved for Figure 4.8.

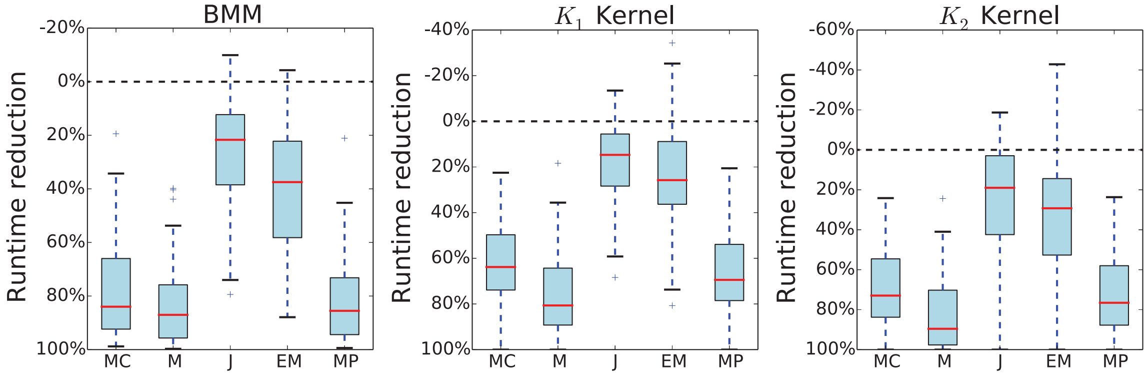


Figure 4.7: Percent decrease in running time when applying different MAP estimation algorithms after dimensionality reduction.

(applicable to all algorithms) is that some of the removed dimensions may be incorrect — Equations 4.4a and 4.4b find dimensions in the MAP instantiation that are set to a specific value *with high probability*. If such dimensions are incorrect, then the MAP estimate across other dimensions is based on conditioning over these incorrect values (see Section 4.2.1), potentially lowering the quality of the estimate.

While the quality of the MAP estimate stays the same on average, when applying the algorithms over a reduced space the time complexity significantly decreases. Figure 4.7 shows the percent decrease in running time for each algorithm and model when applied to the reduced space, averaged across all test itemsets. These approximation algorithms

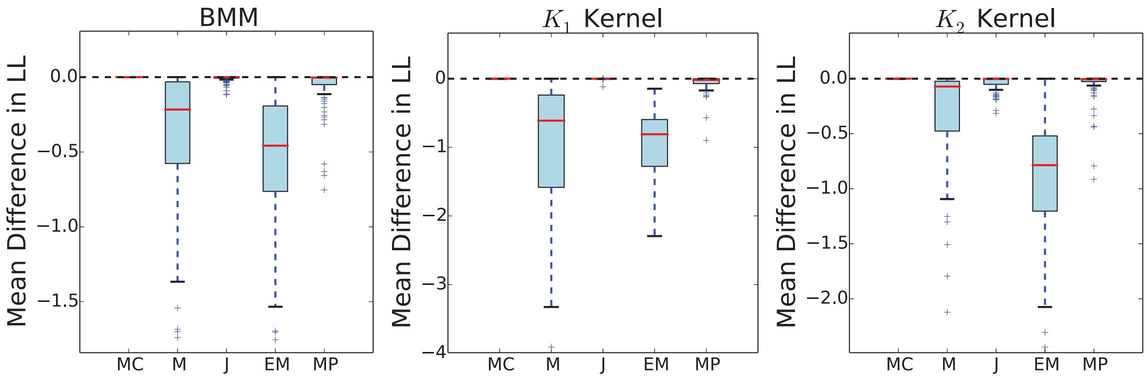


Figure 4.8: Difference in log-likelihood (y-axis) when estimating the MAP using the Monte Carlo method, compared to the other methods from Section 4.2.2. Negative values represent datasets where the Monte Carlo method achieved a higher log-likelihood in its estimate. All algorithms were run over the reduced dimensionality space, where $\hat{x}_S$ is analytically removed.

each have time complexity that is linear in D, thus the decrease in running time is linear. This is certainly seen within the Monte Carlo, marginal, and message passing algorithms, as the runtime was reduced by 60-90% on average (matching the percentage in dimensionality reduction shown in Figure 4.5). The reduction in runtime is not as significant for the joint or EM algorithms — in some cases, these algorithms took a longer amount of time. This is likely a result of their implementations, in additional to poor initial estimates for the EM algorithm (taking more iterations to converge). Lastly, it is worth noting that in extreme cases where the reduced dimensionality D' is very small (e.g. $D' \leq 20$), it becomes feasible to calculate the MAP instantiation exactly by brute force (taking $O(2^{D'})$ time).

We conclude by comparing the performance of the different MAP estimation methods from Section 4.2.2. Figure 4.8 shows the mean difference in log-likelihood between MAP estimates using the Monte Carlo method compared to other methods, after dimensionality reduction has been performed. Larger positive values indicate higher-quality estimates were produced relative to the Monte Carlo method. Very clear patterns are seen — the Monte Carlo method is the best performing, followed by the joint and message passing estimation methods. The marginal and EM algorithms consistently produce MAP estimates with lower likelihoods.

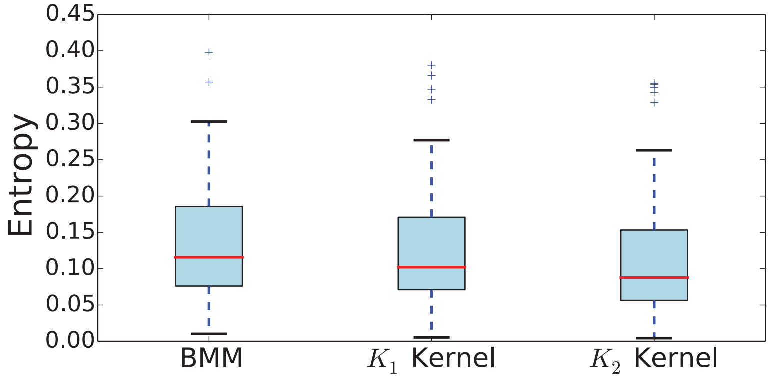


Figure 4.9: Upper-bound on the entropy of the conditional distributions, averaged over test datapoints. The entropy upper-bounds are in units of bits, normalized by the dimensionality of the conditional distributions.

The results in Figure 4.8 reveal more about the structure of the BMM and kernel models than the performance of the algorithms themselves. The fact that the best estimate can be achieved by simulating a small subset of samples indicates that the modes of these distributions are very peaked. A look at the entropy of these conditional distributions confirms that this is indeed the case. For each conditional distribution considered, an upper bound on its entropy is calculated from its univariate entropies (Cover and Thomas, 2006, Theorem 2.6.6). Figure 4.9 shows the average upper-bound, in units of bits normalized by the dimensionality of the conditional distribution, across all datasets and models. In most cases, a single dimension (or bit) contains less than 0.15 bits of information, indicating little uncertainty in the conditional probability densities.

Additional evidence of the peakedness in the conditional distributions is given in Figure 4.10. The left plot compares the log-likelihood of the MAP estimate against two versions of the Monte Carlo algorithm, one using $M = 10000$ samples and one using only $M = 100$ samples. For the large majority of email datasets, the same-quality MAP estimate can be found using only 100 simulated samples, suggesting that the mode(s) of the conditional distributions are large and easy to find. As another illustration, we initialized the EM algorithm to the output of the Monte Carlo method, and compared its log-likelihood to the Monte Carlo method in the right plot of Figure 4.10. In all cases, the EM algorithm immediately converges to the

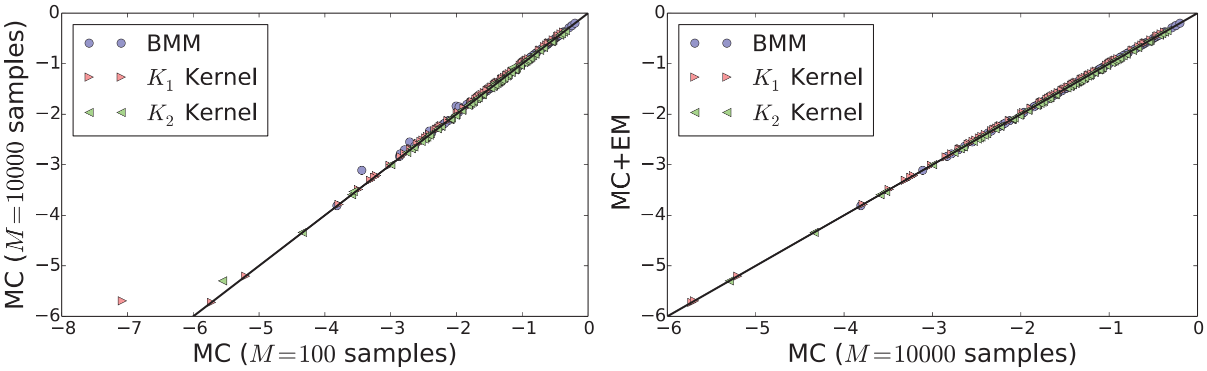


Figure 4.10: Comparison in log-likelihood of MAP estimation via Monte Carlo method, after applying dimensionality reduction. Left: Monte Carlo estimation using 100 samples versus 10000 samples. Right: Monte Carlo estimation using 10000 samples versus the EM algorithm initialized via Monte Carlo estimation.

original Monte Carlo estimate. This is evidence that the Monte Carlo method is indeed able to find a local maximum or mode using a limited number of samples.

Furthermore, the higher log-likelihoods of estimates using the joint method in Figure 4.8 suggests that these conditional modes tend to be consistent with those of the individual components of the models — a property not held in general by mixture models. The decrease in performance for the marginal method is expected, as the dependency structure between the dimensions is ignored when producing an estimate. The low performance of the EM algorithm suggests that its performance is heavily dependent on the initial estimate (evidence for this was shown in the right plot of Figure 4.10).

The structure of the BMM and kernel models reflected in these results is specific to the context of email recipient behavior. Typically, emails are sent to small groups of people, with strong co-appearance patterns amongst possible recipients. Thus, conditioning on the first person to receive an email reveals a lot of information regarding which other possible recipients are likely to be added. This does not necessarily suggest that the task of itemset completion is easy — a person can be an active member across multiple groups, thus conditioning on that person receiving the email can create ambiguity over which set of others will also receive the email. However, as the results in the following experiment will also show, it does suggest that the BMM and kernel models are effective at identifying the different groups of people an email user typically sends emails to.

4.4.4 Ranking Future Items

In this experiment, we focus on the approach of itemset completion via ranking. In particular, we evaluate the accuracy in ranking potential items to be added to a partially observed itemset when ranked by their marginal probability in the respective conditional distribution (as discussed in Section 4.3). Ranking accuracy between the Bernoulli mixture model, the kernel model, and a baseline ranking model are compared.

Evaluation Metric

The evaluation metric used in this experiment is the same as that used in the work of Letham et al. (2013). The metric is an error rate based on the number of "incorrect" items (e.g. recipients that will not be added to the email) which are ranked higher than "correct" items (e.g. those that are added to the email).

To formally define the metric, some notation is needed. Let $r_i \subset \{1, \cdots, D\}$ be an index set representing the items contained in itemset i, with $|r_i| = R_i$. Thus, the itemset represented as a D-dimensional binary vector x will have $x_d = 1$ if and only if $d \in r_i$. Additionally, let $r_{i,j}$ be the jth item to appear in the itemset (i.e. the jth person added to an email). Suppose we observe the first m items in the itemset (with $1 \leq m < R_i$), thus a ranking over the $D - m$ other items is required. A ranking is generated by sorting scores associated with each item in descending order. The error rate of such a ranking, with respect to the $R_i - m$

unobserved items in the itemset, is defined as

$$E_{i,m} \equiv \left(\frac{1}{|A_{i,m}||B_i|}\right) \sum_{a \in A_{i,m}} \sum_{b \in B_i} \delta(s_b \geq s_a) \tag{4.7}$$

where s_a is item a's score, and

$$A_{i,m} \equiv \bigcup_{j=m}^{R_i} r_{i,j} \qquad\qquad B_i \equiv \{1, \cdots, D\} \setminus r_i$$

are the remaining items in the itemset and all items not in the itemset, respectively. For the BMM and kernel models, an item's score is defined by Equation 4.6 — the marginal probably of the item being included in the itemset, conditioned on observing the first m items. An error rate of $E_{i,m} = 0$ indicates a perfect ranking where all $(R_i - m)$ unobserved items in the itemset are ranked higher than those not in the itemset. Similarly, an error rate of $E_{i,m} = 1$ indicates a ranking where its *reverse ordering* yields a perfect ranking.

Each itemset in the test data is processed independently. For a test itemset containing R_i items, the error rates $\{E_{i,m} : 1 \leq m < R_i\}$ are calculated. These reflect the error rates of an item recommendation system as items are added to the itemset one by one. The overall error across all test itemsets is defined to be the average error over all rankings:

$$E_{\text{test}} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \frac{1}{R_i} \sum_{m=1}^{R_i} E_{i,m} \tag{4.8}$$

Baseline Ranking Model

We compare the error rate of rankings generated by the BMM and kernel models to that of a baseline ranking model — the sequential event prediction (SEP) framework of Letham et al. (2013). Under this framework, item a's score s_a in Equation 4.7 is defined as

$$s_a \equiv \lambda_a + \sum_{j=1}^{m} \mu_{r_{i,m}} \hat{P}(a|r_{i,m}) \tag{4.9}$$

where λ_a is a bias term for item a, μ_b a "strength of influence" term for item b, and $\hat{P}(a|b)$ the empirical probably of item a being in an itemset that item b is in (estimated from training data)[5]. Larger scores in Equation 4.9 indicate items with stronger co-appearance patterns between the items and those already observed in the itemset.

The parameters of the SEP model are $\boldsymbol{\lambda} = \{\lambda_d : 1 \leq d \leq D\}$ and $\boldsymbol{\mu} = \{\mu_d : 1 \leq d \leq D\}$, and are learned from the training dataset by numerically minimizing (via gradient methods) the following quantity:

$$\begin{aligned}\tilde{E}_{\text{test}} &\equiv \beta \left(\|\boldsymbol{\lambda}\|_2^2 + \|\boldsymbol{\mu}\|_2^2 \right) + \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \frac{1}{R_i} \sum_{m=1}^{R_i} \tilde{E}_{i,m} \\ \tilde{E}_{i,m} &\equiv \left(\frac{1}{|A_{i,m}||B_i|} \right) \sum_{a \in A_{i,m}} \sum_{b \in B_i} e^{s_b - s_a}\end{aligned}$$

where $\tilde{E}_{i,m}$ is a continuous upper-bound of $E_{i,m}$ in Equation 4.7, and β a regularization parameter (Letham et al., 2013). The value of β is chosen via 10-fold cross-validation over the training dataset, where values of $\beta \in \{0, 0.01, 0.1, 1.0\}$ are considered.

Results

We begin by comparing the test errors (defined as E_{test} in Equation 4.8) when ranking items using the one-parameter (K_1) and two-parameter (K_2) kernel models. In Figure 4.11, the test errors between the two models are shown, where each point represents a separate email

[5] In the version presented by Letham et al. (2013), b can be generalized to a subset of items, capturing higher-order dependencies between subsets of items. We chose to model pairwise dependencies as introducing higher-order terms increases both the number of parameters to learn and the complication in defining appropriate subsets of items to model.

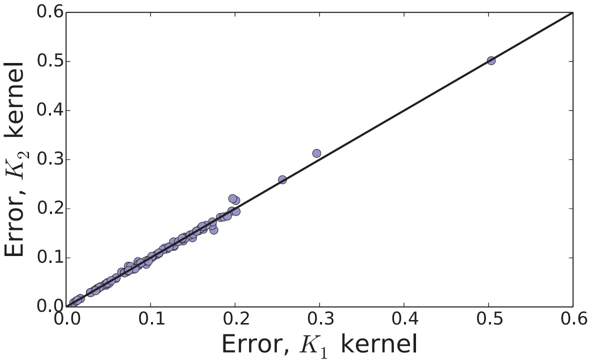


Figure 4.11: Test error between the kernel models, across all email datasets.

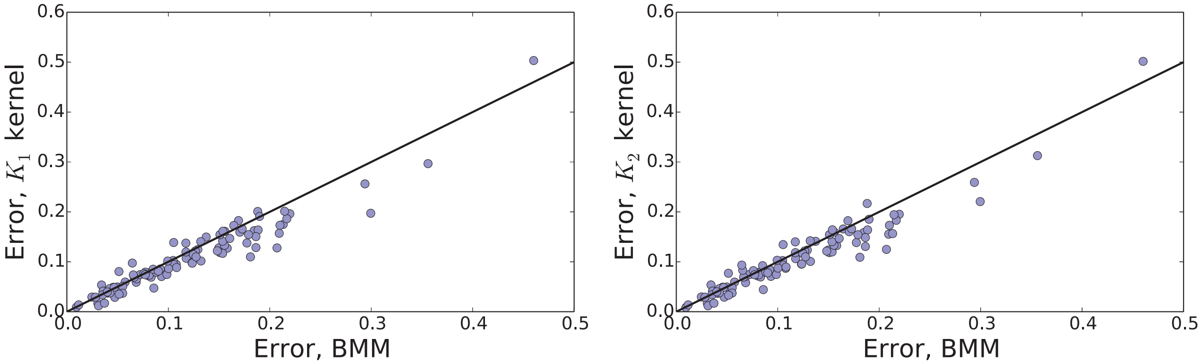


Figure 4.12: Test error between the BMM and the K_1 (left) and K_2 (right) kernel models.

dataset. For the majority of email datasets, similar test errors occur when ranking using either kernel model. This is due to the similar structure in the probability density posited by the kernels (with respect to their equivalent BMM parameters π and θ — see Section 3.2.2). When interpreted as a BMM, both kernels have equivalent mixing proportions π, and the Bernoulli probabilities in θ are close to 0 or 1 in the same indices. Thus, although the two-parameter kernel was shown to be a better fit to test data in Chapter 3, when used for ranking both kernels are able to identify which items are (not) likely to be added to a partially observed itemset.

Figure 4.12 compares the test error across all datasets when ranking items using the Bernoulli mixture model versus the two kernel models. In general, the error is lower when ranking

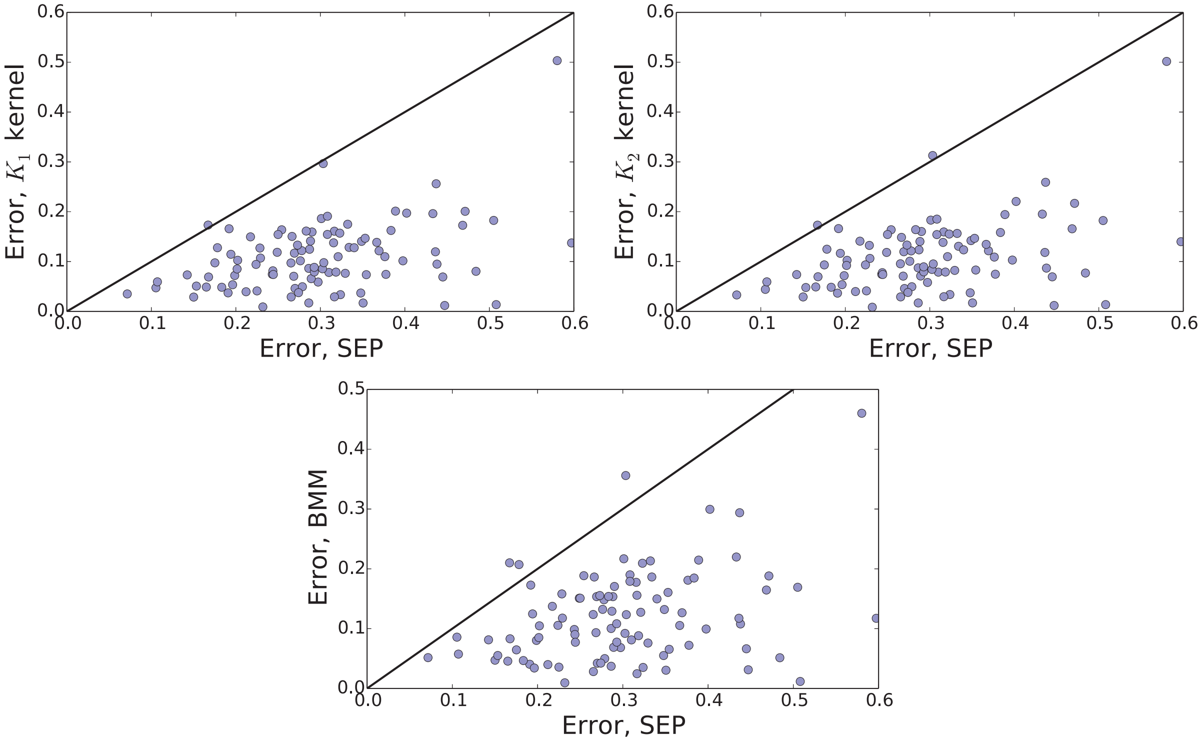


Figure 4.13: Test error between the SEP baseline (x-axis) and the BMM (y-axis; bottom plot) and kernel (y-axis; top plots) models.

using either of the kernel models, compared to the BMM model[6]. For one particular dataset, the kernel models failed to converge when estimating the bandwidth parameters, resulting in an unusually high test error. The decrease in test error for the majority of other datasets is due largely to the non-parametric nature of the kernel model, where dependency structures between the different items are better captured. For traditional Bernoulli mixture models, the limitation of modeling the data using a fixed number of components increases the approximation of such dependencies.

We conclude by comparing in Figure 4.13 the test error of both the BMM and kernel models to that of the baseline SEP ranking model as proposed by Letham et al. (2013). Both models produce significantly lower test errors than the SEP ranking model across most datasets. The main reason for this is that the BMM (and, equivalently, the kernel model) is able

[6] The pairwise differences in test errors are statistically significant (p-value less than 0.001) according to a signed-rank Wilcoxon test.

to account for higher-order dependencies in its conditional distribution than the pairwise influence weights of the SEP ranking model. While it is possible to include higher-order dependencies in the SEP ranking model, complications occur when doing so. For example, which higher-order dependencies are accounted for must be manually specified (or heuristically set based on the training dataset). Additionally, the empirical co-appearance patterns $\hat{P}(a|b)$ of Equation 4.9 are likely to be noisy, as the number of itemsets that items a or b appear in may be small. Both issues are accounted for when applying a probabilistic model over the joint behavior of all items, with appropriate priors placed over model parameters prior to training.

4.5 Discussion and Future Work

In this chapter, we addressed the problem of itemset completion — given a partially observed itemset, the goal is to predict which additional items will be added to that itemset. Two different approaches to the problem were explored, each based on conditional distributions of a Bernoulli mixture model describing the itemsets. The first approach directly estimates the remaining items to be added by approximating the MAP instantiation of the conditional distribution. We proposed a technique for analytically removing values of some dimensions of the MAP instantiation, allowing MAP approximation algorithms to work more efficiently in a reduced dimensionality space. The second approach, specifically designed for item recommendation, ranks items according to their likelihood of being included to the itemset. Although we focused on the problem of itemset completion with respect to email recipient lists, the proposed methods are applicable to any type of itemset data (e.g. Table 2.1).

Experimental results across 99 different email datasets showed that the quality of the MAP estimates (in terms of log-likelihood) were generally the same, if not better, when dimensionality reduction is applied. Results also showed a decrease in ranking error when ranking items according to the marginal probabilities of a BMM conditional distribution, compared to a baseline ranking model. These results were found not only for Bernoulli mixture models, but also for the kernel models of Chapter 3 (which are a special form of a BMM).

Potential future work to the approaches described here could include the use of additional information or metadata when estimating the MAP instantiation or ranking items. In particular, for modeling email behavior, the "back and forth" messaging that constitutes an email "thread" could be used. For example, the recipients of previous messages the individual has sent in an email thread could be used to refine predictions over who will receive an email later in the thread. Another strand of future work, specific to timestamped itemset data, is to have the BMM parameters change over time. This would allow the shape of the probability distribution over itemsets to adapt as item co-appearance patterns change, i.e. as the relationship between an individual's email contacts change over time.

4.6 Summary of Contributions

The primary contributions of this chapter include the following:

- Two different approaches to the itemset completion problem were addressed, each exploiting the structure of the conditional distribution of Bernoulli mixture models (and, equivalently, the kernel models of Chapter 3).
- A method was developed for analytically estimating the values of some dimensions in the MAP instantiation, consistent with the true value with high probability. Experiments over 99 different email datasets showed the majority of dimensions can be analytically removed, allowing for a significantly decreased time complexity in estimating the MAP instantiation over remaining dimensions.

- The performance of different MAP approximation algorithms are compared, in both the original and reduced dimensionality spaces. The results not only showed a clear ordering in terms of performance, but also revealed information about the general structure of a BMM when applied to email behavior.

- The BMM and kernel models were applied to the task of ranking which potential items are likely to be added to a partially observed itemset. Results indicated a significant improvement in predictive power when modeling itemsets with the BMM and kernel models, compared to a baseline ranking model.

Chapter 5

Modeling Itemset Patterns over time with Latent Variable Models

The modeling choices over itemset data in Chapters 3 and 4 were proven effective in capturing the co-appearance and higher-order dependency structure between items contained in an itemset. In this chapter, we incorporate the time dimension (i.e. when the itemset occurred) into the modeling of itemsets. We focus on analyzing itemset patterns over time, specifically in the context of egocentric communication data. Such data consists of a time series of counts of communication events between an individual and his or her social circle, where an itemset is considered to be the set of recipients of an event (e.g. those who receive an email).

As an example of the type of data of interest, Figure 5.1 shows the weekly email communication patterns from several years of email history of a Gmail account from Section 1.1. The left plot shows the weekly rate of the user sending emails over time, and the right plot shows when different email recipients (y-axis — each column representing an itemset) receive emails from that user over time (x-axis). The patterns of communication are clearly non-stationary. As the user transitioned through different universities, projects, and social

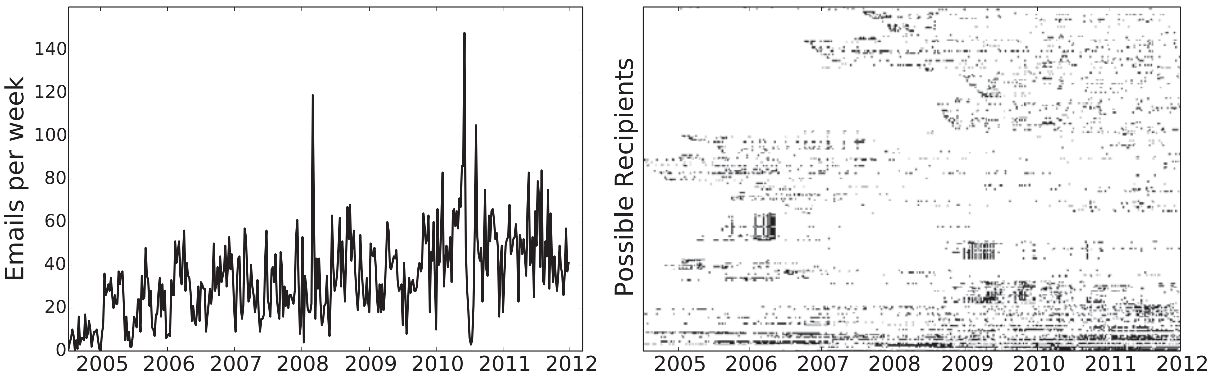


Figure 5.1: Personal email communication data for a single Gmail account.

activities, the recipient patterns and overall communication rate changed significantly over time. Describing such data statistically can be a difficult task, as there are multiple unobserved aspects of the data which are difficult to disentangle: at what rates does an individual communicate? who does he or she communicate with? are their prominent patterns among the recipients such as groups or clusters? and how do such patterns change over time?

For example, prior to 2008, a significant portion of the user's email contacts shown in Figure 5.1 were rarely present. In addition, during the middle of 2010 there is a large spike in email activity followed by a sharp drop in communication; it would be interesting to identify which recipients were associated with this change in behavior. We describe in this chapter an unsupervised statistical learning approach that can explain such variations, in terms of both who we communicate with and the rate at which we communicate.

There are several potential applications of this type of model. For example, there is a broad consensus that there is significant room for improvement over current approaches to personal email management (Fisher et al., 2006; Wainer et al., 2011; Whittaker et al., 2011). This has motivated the development of a variety of visualization and clustering techniques as the basis for automated email management tools to help individuals to better understand and manage their email (Dredze et al., 2009b,a; Fisher, 2005; Koren et al., 2011; MacLean et al., 2011). A simple example is the Gmail "Got the wrong Bob?" feature (Roth et al., 2010) or, similarly, the work presented in Chapter 4, where co-appearance patterns in email recipient lists are learned and used to recommend potential recipients to the email sender.

The proposed model in this chapter consists of two interacting components:

1. Group structure among recipients is modeled by a mixed membership model, similar to that used for social networks (Airoldi et al., 2008) and in topic modeling for text (Blei et al., 2003). This framework allows for modeling of recipients as members of multiple groups in a natural and parsimonious manner.

2. The daily number of emails sent over time is modeled via a set of independent piecewise-constant Poisson processes, one per group. The number of changepoints between Poisson segments for each group is handled by a non-parametric Dirichlet process, i.e., there are a potentially infinite number of changepoints and segments in the prior for the model, from which we infer a posterior distribution over a finite number of segments and changepoints given the observed data.

The rest of this chapter is structured as follows. In Section 5.1 we discuss previous work on modeling count time series in email and other communication data, where the primary focus has been on segmentation and changepoint detection but without group structure. Sections 5.2 and 5.3 outline the changepoint and grouping aspects of the model respectively. Parameter inference is detailed in Section 5.4, with an illustration of the fitting procedure to simulated data in Section 5.5. Section 5.6 applies the model to real-world email datasets, demonstrating how the model can be used to better understand such data. Section 5.7 describes a set of experiments where we compare the proposed model to various baselines in terms of 1) the quality of the groups learned by the model, and 2) the predictive accuracy of the model. The chapter concludes with a discussion and summary of contributions.

A published version of the work presented in this chapter is available in Navaroli et al. (2012, 2013).

5.1 Related Work

Prior work relevant to our proposed approach can be broadly categorized into three areas: (1) models of email communication data, (2) segmentation of time series of count data, and (3) identification of group structure in dynamic social network data.

Earlier work on analysis of email time series data has focused primarily on the modeling of overall communication rates. For example, the work of Malmgren et al. (2008, 2009) investigated a variety of bursty and non-homogeneous Poisson process models to capture the overall rate at which individuals send email. Earlier work in a similar vein applied Markov-modulated Poisson processes to telephone call and Web navigation data (Scott and Smyth, 2003; Scott, 2004). The Markov-modulated Poisson process has also been successful in modeling similar types of "rate" data outside of communication, such as highway traffic and building entrances (Ihler et al., 2007). Our approach also uses latent piecewise-constant Poisson processes for modeling temporal variations in an individual's communication rate. We differ from prior work in that the overall communication rate for an individual is modeled using a combination of (a) grouping patterns amongst recipients, and (b) time-varying rates across groups — prior work focused on modeling the overall rate for an individual, without recipient information.

In the broader context of segmentation of time series of count data, statistical learning approaches have been well studied. In Auger and Lawrence (1989), a dynamic programming approach is used to determine the location of changepoints (i.e. when a temporal change in behavior occurs) based on optimizing a measure of fit (e.g. log-likelihood) across segments.

A hierarchical approach to segmentation was introduced by Guralnik and Srivastava (1999), which iteratively splits existing segments into smaller segments until a stopping criteria is reached. In Fearnhead (2006), a probabilistic model is used to place priors over the number and location of changepoints, which is used to obtain posterior estimations. Another probabilistic approach is the work of Chib (1998), which models the time series with a finite-state hidden Markov model (HMM), such that changepoints are represented as latent state transitions within the HMM. The HMM is constrained such that, once a state is transitioned away from, that state cannot be visited again. Thus, for a HMM with K different states, at most $K - 1$ changepoints can be inferred.

Our approach is most similar to Chib (1998), but uses a Dirichlet process prior in order to have a potentially infinite number of latent states, allowing for an arbitrary number of changepoints to be inferred. A significant difference from previous work is that we do not detect changepoints in a single time series, but in the decomposition of the time series according to the (simultaneously learned) latent grouping structure. Each latent group is associated with its own time series and changepoints. Our approach is also inspired by non-parametric Bayesian methods applied to other types of human communication, such as detecting speaker changes in audio recordings of meetings (Fox et al., 2011). These methods use a HMM with a flexible number of latent states. In this work, we similarly use non-parametric techniques for segmenting the time series associated with each latent group.

The third relevant strand of prior work is learning latent group structure from dynamic social network data. A large literature exists on this topic, including techniques based on optimizing cost functions (Berger-Wolf and Saia, 2006) or using statistical model-based approaches. One distinction among these methods is whether individuals are allowed to belong to one or several groups. We take an approach allowing individuals to be members of multiple groups, akin to mixed membership models (Airoldi et al., 2008; Choi et al., 2012) and the work of McAuley and Leskovec (2012). In particular, we jointly model both the group memberships and the rate of events involving a particular group.

This is similar in some respects to prior work on dynamic topic models, except that here we use a rate-based changepoint model to handle temporal changes, wheras the typical approach is to model temporal changes as a smooth function of time (Blei and Lafferty, 2006; Wang et al., 2008). Also of relevance is prior work on community detection for dynamic social networks based on node clustering techniques, e.g., detecting clusters of nodes (communities) in a time-varying weighted graph. Such approaches include algorithms based on graph-coloring (Tantipathananandh et al., 2007) and clustering methods over "snapshots" of a dynamic network (Xu et al., 2011). While it is possible to use these approaches for the grouping component of our model, we use the mixed membership approach, allowing individuals to belong to multiple groups at once. The probabilistic semantics of such a model allows us to learn and reason about both groups and communication rates in a coherent fashion.

5.2 Modeling Communication Rates

We begin by describing our modeling approach to learning time-varying communication rates from a time series of count data using an infinite-state HMM. The modeling of communication rates is then coupled with the modeling of latent group structure in Section 5.3.

5.2.1 Modeling the Number of Emails Sent each Day

Let $N_t \geq 0$ represent the total number of emails the user sends on day t. The set of variables $\{N_t : 1 \leq t \leq T\}$ define a stochastic process. We model $N_t \sim \text{Poisson}(\lambda_t)$, where λ_t is the *communication rate* at which the user sends emails on day t. Because λ_t is allowed to change across days, this type of process is usually referred to as a non-homogeneous Poisson process.

Our model assumes that the user communicates with K potentially overlapping groups of people. It is further assumed that each email the user sends is sent to exactly one of the K groups. We model the rate at which emails are sent to each group as independent Poisson processes, i.e., a change in the rate at which emails are sent to one group does not affect the rate at which emails are sent to other groups. This assumption is clearly an approximation of what happens in practice — for instance there may be exogenous or external events, such as the user going on vacation, which affect the communication between the user and most or all groups simultaneously. Nonetheless, the independence assumption is a useful (and computationally efficient) modeling choice, allowing us to capture "first-order" group behavior — models allowing dependencies between groups and/or shared dependence on exogenous events would be of interest as extensions of the model proposed in this chapter.

Let $N_{k,t}$ represent the (unobserved) number of emails the user sends to group k on day t, such that $N_t \equiv \sum_{k=1}^{K} N_{k,t}$. The values of $\{N_{k,t} : 1 \leq t \leq T\}$ then define a stochastic process specific to the user's communication with group k. We model $N_{k,t} \sim \text{Poisson}\,(\lambda_{k,t})$, where $\lambda_{k,t}$ is defined as the communication rate at which the user sends emails to group k on day t. Because of our independence assumptions, it follows that N_t is modeled as the superposition of independent Poisson processes, where

$$N_t = \sum_{k=1}^{K} N_{k,t} \sim \text{Poisson}\,(\lambda_t) \qquad\qquad \lambda_t = \sum_{k=1}^{K} \lambda_{k,t} \tag{5.1}$$

5.2.2 The Poisson Process with Respect to a Single Group

In this subsection, we focus on the modeling of the non-homogeneous Poisson process with respect to a single group k, describing the time-varying communication rate of the user towards that group. The modeling of the groups themselves is discussed in Section 5.3.

We model the user's emailing rate to group k over time, $\{\lambda_{k,t} : 1 \leq t \leq T\}$, using a HMM. Let $s_{k,t}$ represent the latent state of the HMM associated with group k on day t. We model the HMM such that its unique latent states correspond to different values for emailing rates. Thus, the value of $\lambda_{k,t}$ is a function of the HMM latent state $s_{k,t}$ (defined later), and the value of $s_{k,t}$ dependent on the state of the previous day for that group, $s_{k,t-1}$.

We formally define a *changepoint* to be a time t where a transition between different latent states occur, e.g. values of t where $s_{k,t} \neq s_{k,t+1}$. Changepoints typically correspond to unobserved events in the user's history which changes their communication rate with the group — vacations, research deadlines, and changing between schools or locations are all examples of changepoint events. We define a single, contiguous interval of time between two adjacent changepoints to be a *segment*. Each segment of time represents a period of constant mean activity for the user with respect to a particular group.

Traditionally, a HMM has a finite number of states, limiting the number of modes of activity a user can have. Here we allow the HMM to have a countably infinite number of states, where only a finite subset of those states are ever seen given the observed data (similar to Beal et al. (2002)). As in Chib (1998), the HMM is constrained such that it cannot transition to previous states, ensuring that states span a single interval of time[1]. This is achieved by placing separate a symmetric Dirichlet prior over each row of the transition matrix. As the number of latent states tends to infinity, these priors converge in distribution to Dirichlet processes (Neal, 2000). By applying the Dirichlet prior and integrating out the parameters

[1] An alternative approach is to allow the state transition matrix to be unconstrained, i.e., allowing the HMM to return to earlier states (Fox et al., 2011). Such an approach is certainly feasible, and has the advantage that segments could share parameters by representing recurring states and rates. We did not pursue this approach primarily for computational reasons as inference in such a model is significantly more complex than in the proposed constrained model.

for the HMM transition matrix, the transition probabilities between states become:

$$P(s_{k,t}|s_{k,-t},\gamma,\kappa) = \begin{cases} \frac{V_t+\gamma}{V_t+\gamma+\kappa} & \text{if } s_{k,t} = s_{k,t-1} \\ \frac{\kappa}{V_t+\gamma+\kappa} & \text{if } s_{k,t} \text{ is a new state }, \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

where

- γ and κ are adjustable hyperparameters, affecting the state transition behavior of the HMM (discussed in detail in Appendix C).
- $s_{k,-t} \equiv \{s_{k,t'} : t' \neq t\}$ is the set of all other HMM states (the integration of the transition matrix introduces dependencies between all latent states).
- $V_t \equiv \sum_{t'=2}^{t-1} \delta(s_{k,t-1} = s_{k,t'-1})\delta(s_{k,t-1} = s_{k,t'})$ is number of times the HMM has self-transitioned to state $s_{k,t-1}$ up to time t.

The next modeling choice we discuss is the value of $\lambda_{k,t}$, group k's rate at time t, with respect to the latent state $s_{k,t}$. We model the log-value of the rate as

$$\log \lambda_{k,t} = \beta_{k,s_{k,t}} + \alpha_{d(t)} \tag{5.3}$$

where

- $\beta_{k,s_{k,t}}$ is the contribution to the communication rate specific to the HMM latent state $s_{k,t}$. Thus, for each time segment defined by a unique HMM latent state m, $\beta_{k,m}$ determines the user's relative communication rate to that group during that time. Although defined in log-space, we will refer to these parameters are *group rate* parameters.
- $a_{d(t)}$ is group-independent and represents the daily effect on the communication rate for day $d(t) \in W$. Different parameterizations of $d(t)$ are possible; for instance we can

define $W = \{0, 1\}$ such that 0 represents weekdays and 1 represents weekends. Such a configuration allows the communication rate to vary across weekdays and weekends, keeping the relative rates across each group unchanged. In this chapter, we set $W = \{0, \cdots, 6\}$, allowing the communication rate to be modulated across each day of the week. We will refer to these parameters as *weekday rate* parameters.

The value of $\lambda_{k,t}$ defined by Equation 5.3 can be thought of as a special form of Poisson regression, i.e., $\log \lambda_{k,t} = Y_{k,t}^T \omega$, where $Y_{k,t}^T$ is a feature vector and $\omega = \{\{\beta_{k,m}\}, \{\alpha_{d(t)}\}\}$ a vector of regression parameters. The form of $Y_{k,t}^T$ under this setup would be a set of indicator variables indicating 1) the latent state $s_{k,t}$, and 2) the day-of-week value of $d(t)$.

Graphical Representation of the HMM Component of the Model

A graphical representation of the HMM model discussed in this section is given in Figure 5.2. To summarize, a separate HMM model is applied independently to each of the K groups. The HMM latent states $\{s_{k,t}\}$ depend on the transition matrices (represented by the "A" node), which are integrated out according to the Dirichlet process prior. We take a Bayesian approach and apply a Normal(μ, σ^2) prior independently over each dimension of the rate parameters α and β (in Equation 5.3). The value of N_t, shaded in gray as they are the only observed variables, is dependent on the values of $\{\lambda_{k,t} : 1 \leq k \leq K\}$ (in Equation 5.1). Since the value of $\lambda_{k,t}$ is a deterministic function of $\{s_{k,t}, \alpha, \beta\}$, they could be omitted from the depiction of the graphical model. However, they are included for clarity.

5.3 Modeling Email Recipient Groups

In this section, we address to modeling of the K separate recipient groups that the user interacts with, where a group is defined as a distribution over the possible sets of recipients

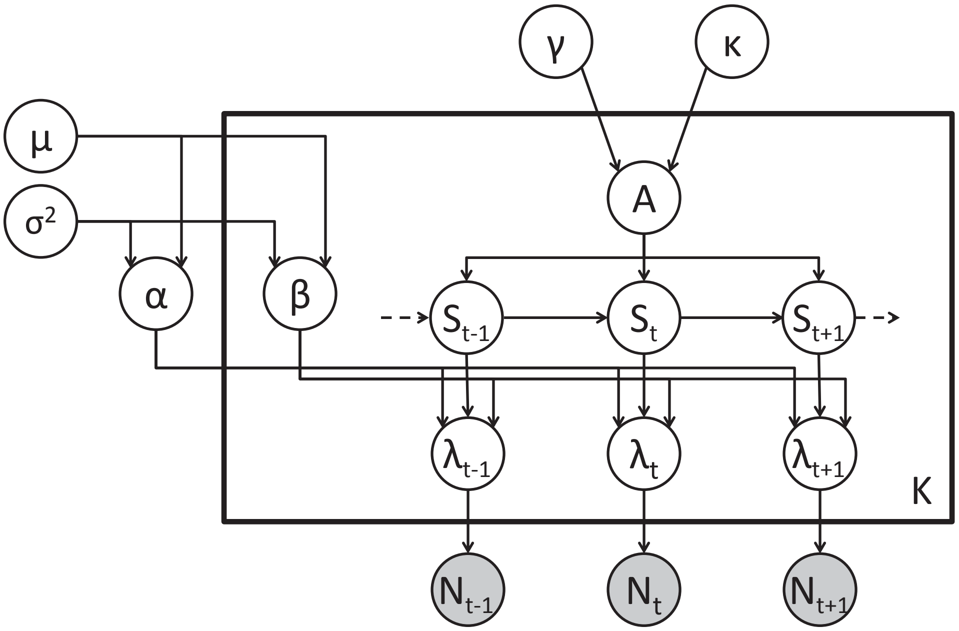


Figure 5.2: Graphical representation of the HMM component of the model, with a separate HMM modeled for each of the K groups. Here, A represents the HMM transition matrices.

for an email. Intuitively, a group should reflect a collection of email recipients, all of which share a common relationship to the user. Examples of such groups include family members, friends, organizational peers, members of a club the user participates in, etc.

We assume that each email the user sends is sent to a single group. Let $z_{t,n} \in \{1, \cdots, K\}$ represent the latent group that the nth email on day t was sent to, and $x_{t,n} \in [0,1]^R$ a R-dimensional binary vector indicating which of the R possible recipients are in the email. Suppose that $z_{t,n} = k$ — conditioned on this value, each possible recipient r is assumed to be independently added to the email with probability $\theta_{k,r}$. The generative model is

$$\theta_{k,r}|\alpha^{(z)}, \beta^{(z)} \sim \text{Beta}(\alpha^{(z)}, \beta^{(z)}) \qquad\qquad x_{t,n}|\theta, z_{t,n} \sim \prod_{r=1}^{R} \text{Bernoulli}(\theta_{z_{t,n},r}), \tag{5.4}$$

where $\alpha^{(z)}$ and $\beta^{(z)}$ are hyperparameters to the Beta prior over individual recipient probabilities. Under this model, the expected number of recipients for an email sent to group k is $\sum_{r=1}^{R} \theta_{k,r}$. The modeling choice of $x_{t,n}$ in Equation 5.4 is equivalent to the Bernoulli mixture model introduced in Section 2.1.2, with the latent variable z explicitly conditioned on (as opposed to marginalized over).

The modeling of the latent group indicator variables $\{z_{t,n}\}$ is a key aspect of the model — the distribution of $z_{t,n}$ introduces dependencies between the HMM model discussed in Section 5.2 and the generative model of email recipients in Equation 5.4. We model the discrete distribution over the latent variable $z_{t,n}$ as a function of the daily rates $\{\lambda_{k,t} : 1 \leq k \leq K\}$, the rate at which the user is sending emails to each group on day t. Because the emailing rates to each group are independently modeled with a Poisson process and the time between consecutive emails sent to a particular group follows an exponential distribution, it is straightforward to show (using standard properties of exponential random variables (Ross, 2006, Section 5.2)) that the probability of an email on day t is sent to group k can be written as

$$P(z_{t,n} = k | \{\lambda_{k',t} : 1 \leq k' \leq K\}) = \frac{\lambda_{k,t}}{\sum_{k'=1}^{K} \lambda_{k',t}} \tag{5.5}$$

Our approach to modeling email recipients is similar in some respects to the dynamic topic model approach of Blei and Lafferty (2006), where each "group" discussed here is analogous to a "topic". However, there are some significant differences between the work presented here and topic models. The data x is represented in this model as a multidimensional binary vector, not as a "bag of words" where repetitions are allowed. This allows the grouping structure to be defined as an independent Bernoulli distribution, as opposed to a multinomial distribution. Additionally, latent variables are used to define the groups associated with each datapoint x — in topic models, separate latent variables are applied to each dimension within x. Lastly, we use a discrete-state Markov process to model the changes in group behavior over time, instead of the autoregressive approach in Blei and Lafferty (2006).

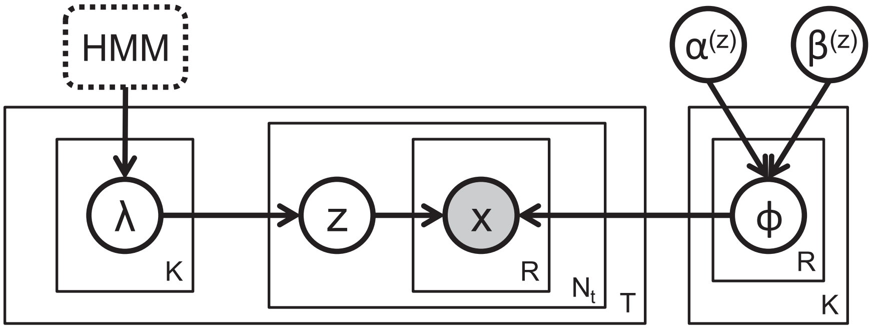


Figure 5.3: Graphical representation of the grouping component of the model. The "HMM" supernode is a compact representation of Figure 5.2.

Figure 5.3 shows a graphical representation of the modeling of email recipient groups, as described in this section. In the interest of interpretability, all the HMM variables described in Figure 5.2 (except for λ and N) are combined into a single supernode. In the graphical model, the "x" node is shaded in gray as it is the only variable observed.

5.4 Parameter Inference

Given an observed email dataset in the form $\{x_{t,n} : 1 \leq t \leq T, 1 \leq n \leq N_t\}$, the parameters of the model discussed in Sections 5.2 and 5.3 are inferred using Markov chain Monte Carlo (MCMC) techniques. In particular, we use Gibbs sampling to iteratively sample each model variable from their full conditional distributions (Geman and Geman, 1984). These conditional distributions can be derived using the graphical models in Figures 5.2 and 5.3, as the joint distribution over all parameters (which the conditional probabilities are proportional to) factors according to the graphical model. Here we outline the sampling equations for each variable in the following subsections — derivations of the sampling equations are provided in Appendix D. To simplify notation, variables without subscripts and in bold denote the set of all variables that can be indexed by it, e.g., $\boldsymbol{\lambda} \equiv \{\lambda_{k,t} : 1 \leq k \leq K, 1 \leq t \leq T\}$. We use Ω to denote the set of all model parameters.

Sampling the Latent Group Indicators z

By taking advantage of the conjugacy between the Beta and Bernoulli distributions, the group membership probabilities $\boldsymbol{\theta}$ can be integrated out of the conditional distribution analytically. The conditional distribution of $z_{t,n}$ given data and all other model variables is

$$\begin{aligned} P(z_{t,n} = k|\boldsymbol{x}, \Omega \setminus z_{t,n}) &\propto P(\boldsymbol{z}|\boldsymbol{\lambda}) \int P(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{\theta}|\alpha^{(z)}, \beta^{(z)}) d\boldsymbol{\theta} \\ &\propto \lambda_{k,t} \prod_{r=1}^{R} \left(\frac{(c_{1,k,r}^{-(t,n)} + \alpha^{(z)})^{x_{t,n,r}} (c_{0,k,r}^{-(t,n)} + \beta^{(z)})^{(1-x_{t,n,r})}}{c_{1,k,r}^{-(t,n)} + \alpha^{(z)} + c_{0,k,r}^{-(t,n)} + \beta^{(z)}} \right) \end{aligned} \tag{5.6}$$

where $x_{t,n,r} \in \{0, 1\}$ is the value of the rth dimension in $x_{t,n}$, and the joint distributions $P(\boldsymbol{z}|\boldsymbol{\lambda})$, $P(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})$, and $P(\boldsymbol{\theta}|\alpha^{(z)}, \beta^{(z)})$ are

$$\begin{aligned} P(\boldsymbol{z}|\boldsymbol{\lambda}) &= \prod_{t=1}^{T} \prod_{n=1}^{N_t} P(z_{t,n}|\{\lambda_{k,t} : 1 \leq k \leq K\}) \\ P(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}) &= \prod_{t=1}^{T} \prod_{n=1}^{N_t} P(x_{t,n}|\theta_{z_{t,n}}) \\ P(\boldsymbol{\theta}|\boldsymbol{\alpha^{(z)}}, \boldsymbol{\beta^{(z)}}) &= \prod_{k=1}^{K} \prod_{r=1}^{R} P(\theta_{k,r}|\alpha^{(z)}, \beta^{(z)}) \end{aligned}$$

and

$$c_{1,k,r}^{-(t,n)} \equiv \sum_{t' \neq t} \sum_{n' \neq n} \delta(z_{t',n'} = k) x_{t',n',r} \qquad c_{0,k,r}^{-(t,n)} \equiv \sum_{t' \neq t} \sum_{n' \neq n} \delta(z_{t',n'} = k)(1 - x_{t',n',r})$$

where $c_{1,k,r}^{-(t,n)}$ and $c_{0,k,r}^{-(t,n)}$ are the number of emails sent to group k that recipient r does and does not appear in, respectively, ignoring the nth email on day t. In the Gibbs sampler, a new value for each $z_{t,n}$ is sampled sequentially using Equation 5.6. Once a new value for $z_{t,n}$ is sampled, the statistics for how often recipients (do not) appear in emails sent to each group are updated. The sampling process for these variables, including the derivation of the conditional distributions, is similar to that of standard collapsed Gibbs sampling algorithms for latent variable models for discrete data, for example as used for Latent Dirichlet Allocation.

In the experimental section, we place a Beta($\alpha^{(z)} = 0.01, \beta^{(z)} = 0.01$) prior over each recipient probability $\theta_{k,r}$. This prior reflects our intuition that recipients which are active in a group should have high membership probability, whereas non-active recipients should have low membership probability.

Sampling the Communication Rate Parameters α and β

We place a Normal($\mu = 0$, $\sigma^2 = 1$) prior over each element in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, the weekday- and group-specific contributions to the user's communication rate in Equation 5.3, respectively. This choice of prior is non-conjugate, making it intractible to sample directly from the conditional distribution. However, a valid Gibbs sample can be produced by instead sampling from the log of the unnormalized conditional distribution, via a technique known as *slice sampling* (Neal, 2003). The log of the unnormalized conditional distributions used for sampling α_w and $\beta_{k,m}$ are

$$\begin{aligned}
\log P(\alpha_w | \boldsymbol{x}, \Omega \setminus \alpha_w) &\propto \log \left(P(\alpha_w | \mu, \sigma^2) P(\boldsymbol{N} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \right) \\
&\propto -\frac{(\alpha_w - \mu)^2}{2\sigma^2} + \alpha_w \left(\sum_{t:d(t)=w} N_t \right) - e^{\alpha_w} \left(\sum_{t:d(t)=w} \sum_{k=1}^{K} e^{\beta_{k,s_{k,t}}} \right) \\
\log P(\beta_{k,m} | \boldsymbol{x}, \Omega \setminus \beta_{k,m}) &\propto \log \left(P(\beta_{k,m} | \mu, \sigma^2) P(\boldsymbol{N} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) P(\boldsymbol{z} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \right) \\
&\propto -\frac{(\beta_{k,m} - \mu)^2}{2\sigma^2} - \sum_{t:s_{k,t}=m} e^{\alpha_{d(t)}} e^{\beta_{k,m}} + g_{k,m} \beta_{k,m},
\end{aligned}$$

where

$$P(\boldsymbol{N}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s}) = \prod_{t=1}^{T} P(N_t|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s}) = \prod_{t=1}^{T} P(N_t|\{\lambda_{k,t} : 1 \leq k \leq K\})$$

$$P(\boldsymbol{z}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s}) = P(\boldsymbol{z}|\boldsymbol{\lambda})$$

$$g_{k,m} \equiv \sum_{t=1}^{T}\sum_{n=1}^{N_t} \delta(z_{t,n} = k)\delta(s_{k,t} = m).$$

The value of $g_{k,m}$ represents the number of emails sent to group k while the HMM corresponding to that group was in segment m. Note that $\boldsymbol{\lambda}$ is a deterministic function of $\boldsymbol{s}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ (using Equation 5.3).

Sampling the HMM Hyperparameters γ and κ

Instead of fixing the Dirichlet process hyperparameters γ and κ, we place priors on them and learn them from the data. Priors are placed over the *ratio* $r \equiv \frac{\gamma}{\gamma+\kappa}$ and *magnitude* $m \equiv \gamma + \kappa$, as opposed to γ and κ directly. The ratio r is interpreted as the prior probability of staying in a newly visited state in the HMM (see Equation 5.2), and the magnitude m the strength of this prior. The priors used are

$$r \sim \text{Beta}(\alpha^{(r)} = 100, \beta^{(r)} = 1) \qquad\qquad m \sim \text{Gamma}(\kappa_m = 0.5, \theta_m = 20)$$

where intuition behind the expected values of r and m under the given priors is provided in Appendix C. As with the rate parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, these priors are non-conjugate, thus slice sampling over the log of the unnormalized conditional probability is used to produce Gibbs samples. First, the magnitude m is sampled, which deterministically updates γ and κ simultaneously (as the mapping between $\{r, m\}$ and $\{\kappa, \gamma\}$ are one-to-one). The ratio r is then sampled, updating γ and κ a second time. According to the graphical model, the conditional probabilities depend only on the priors and the HMM latent state probabilities:

$$P(m|\boldsymbol{x}, \Omega \setminus \{\gamma, \kappa\}) \propto P(m|k_g, \theta_g) P(\boldsymbol{s}|\gamma, \kappa)$$

$$P(r|\boldsymbol{x}, \Omega \setminus \{\gamma, \kappa\}) \propto P(r|\alpha^{(r)}, \beta^{(r)}) P(\boldsymbol{s}|\gamma, \kappa),$$

where

$$P(\boldsymbol{s}|\gamma, \kappa) \equiv \prod_{k=1}^{K} \prod_{t=1}^{T} P(s_{k,t}|\{s_{k,t'} : t' \neq t\}, \gamma, \kappa).$$

and is calculated using Equation 5.2.

Sampling the HMM Latent States s

For each day t and group k, we sample the latent state $s_{k,t}$ conditioned on (a) all other latent states for group k, (b) the latent states for other groups on day t, and (c) the number of emails sent on day t. Due to the restriction that the HMM cannot transition back to previous states (Section 5.2), the value of $s_{k,t}$ is only sampled in cases where $s_{k,t-1} \neq s_{k,t+1}$. When $s_{k,t}$ is sampled, its possible values are 1) the previous state $s_{k,t-1}$, 2) the next state $s_{k,t+1}$, or 3) a brand new state. The prior probability of having $s_{k,t}$ be a new state is proportional to the hyperparameter κ. The conditional probability for sampling $s_{k,t}$ is

$$P(s_{k,t}|\boldsymbol{x}, \Omega \setminus s_{k,t}) \propto P(s_{k,t}|s_{k,-t}, \gamma, \kappa) P(\{z_{t,n} : 1 \leq n \leq N_t\}|\boldsymbol{\lambda}_t) P(N_t|\boldsymbol{\lambda}_t)$$

where $\boldsymbol{\lambda}_t \equiv \{\lambda_{k,t} : 1 \leq k \leq K\}$. Thus, $s_{k,t}$ can be sampled by calculating the probability for all possible values of $s_{k,t}$, then normalizing to create a discrete distribution from which to sample $s_{k,t}$.

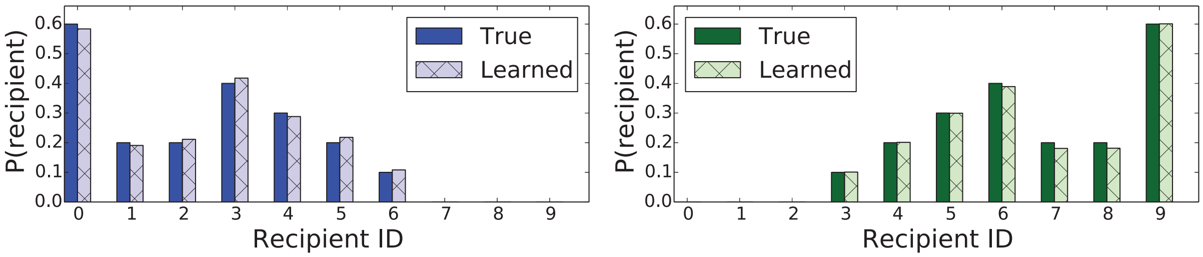
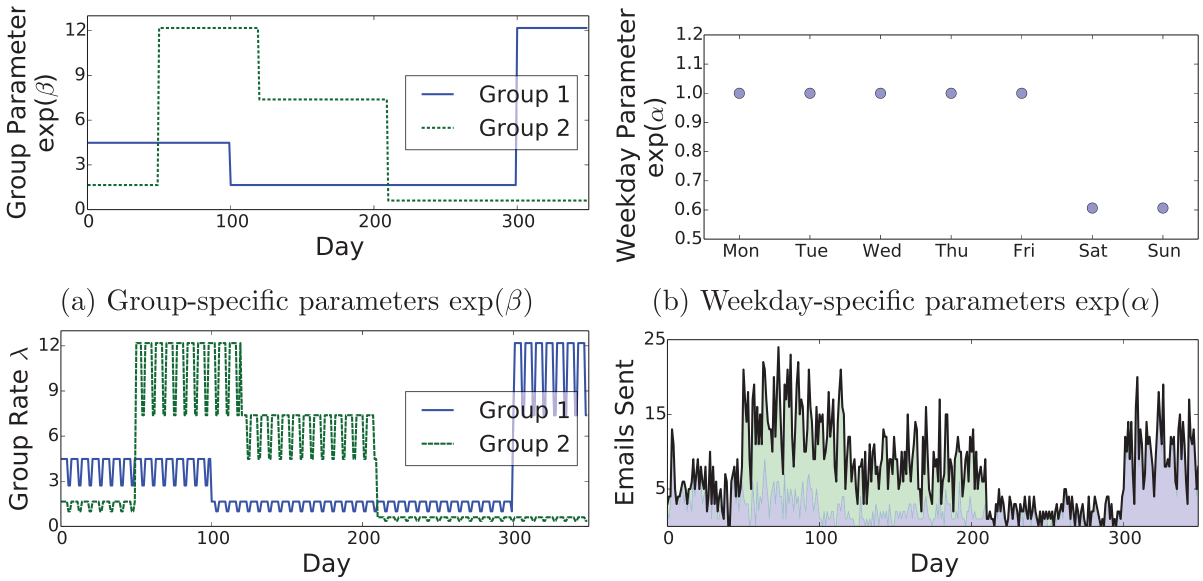


Figure 5.4: True (solid bars) and inferred (cross-hatched bars) membership probabilities for two simulated groups.

In order to calculate the probability of $s_{k,t}$ being a brand new state, we first need a new β parameter corresponding to that new state. A sample of $s_{k,t}$ is produced by first sampling a value of this new parameter from its prior distribution, then using its value in the above conditional distribution. This is an example of sampling using auxiliary variables (Neal, 2000), where a sample from $p(s_{k,t})$ is produced by sampling from a joint distribution $p(s_{k,t}, \xi)$ whose marginal distribution is $p(s_{k,t})$. The auxiliary variable ξ is then discarded. Here, ξ represents the newly sampled β parameter. If $s_{k,t}$ is a singleton state (e.g. $s_{k,t-1} \neq s_{k,t} \neq s_{k,t+1}$), that state will "disappear" if $s_{k,t}$ is sampled to be equal to $s_{k,t-1}$ or $s_{k,t+1}$. When this occurs, the corresponding β parameter is no longer associated with a latent state. As is common in the application of Dirichlet processes, such parameters are discarded.

5.5 An Illustrative Example Using Simulated Data

As an illustration of the fitting procedure, we simulated an email dataset where the user emails $K = 2$ different groups containing $R = 10$ possible recipients, over the course of $T = 350$ days. The solid bars in Figure 5.4 show the membership probabilities of the two groups; each group contains three recipients unique to that group, with the remaining four recipients common across both groups.



(a) Group-specific parameters $\exp(\beta)$

(b) Weekday-specific parameters $\exp(\alpha)$

(c) Communication rates λ

(d) Number of emails sent each day

Figure 5.5: Ground truth of rate parameters for the simulated email user.

The group-specific communication rates (the β parameters) are fixed, and are shown in Figure 5.5a. The rates are defined so that each group is dominant during different periods of time. The interaction with group 1 (solid blue line) changes on days 100 and 300, as indicated by the changing values of β. Similarly, the interaction with group 2 (dashed green line) changes on days 50, 120, and 210. The values for the weekday-specific communication rates (the α parameters) are also fixed, and are shown in Figure 5.5b. The values of α are set such that the user's general email activity on weekends reduces by 40%. The emailing rates to both groups, taking weekly patterns into effect, is seen in Figure 5.5c.

Given the model parameters, emails are simulated by first sampling the total number of emails sent for each day; $N_t \sim \text{Poisson}(\sum_{k=1}^{K} \lambda_{k,t})$. Figure 5.5d shows the sampled values for N_t across days. For each email sent, which latent group the email was sent to is set by sampling $z_{t,n} \sim \text{Categorical}(\{p_k : 1 \leq k \leq K\})$, where $p_k \propto \lambda_{k,t}$ (using Equation 5.5). Once the latent group is determined, the recipients of that email are selected by iterating through each possible recipient r, and including them in the recipient list with probability $\theta_{z_{t,n},r}$.

Estimation of Model Parameters

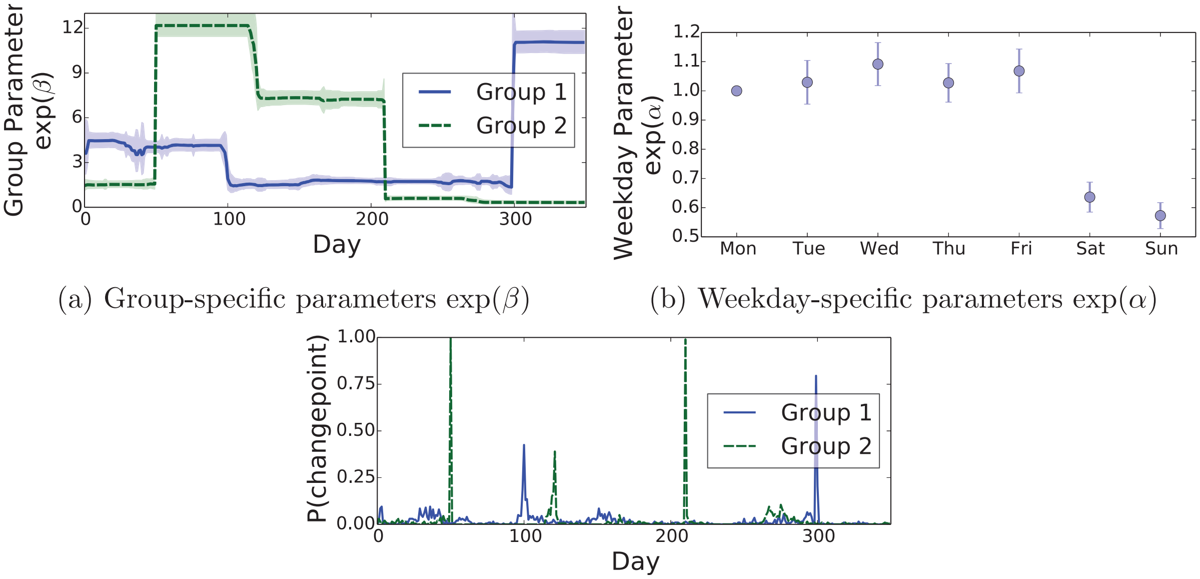
To learn the parameters of the model back from simulated data, we iteratively sample the parameters as described in Section 5.4. A total of 2100 samples are collected, with each sample containing a sampled value across all parameters. The first 100 samples are discarded as burn-in, and every tenth sample after that is kept — parameter estimates are then generated from the 200 remaining samples. The HMM latent states are initialized such that every $s_{k,t}$ represents its own unique state. Equivalently, the changepoints are initialized such that one occurs on each of the 350 simulated days. The rate and Dirichlet process parameters are initialized to samples from their respective prior distributions, discussed in Section 5.4. The latent group variables $z_{t,n}$ are initialized uniformly at random between the two groups.

The cross-hatched bars in Figure 5.4 show the learned groups estimated from the samples. As the membership probabilities θ are integrated out during sampling, they are estimated using the set of latent z variables from the sample that produced the largest log-likelihood $\log P(\boldsymbol{x}|\Omega)$, resulting in a maximum a posteriori (MAP) estimate of θ. The value of θ is calculated as follows:

$$\hat{\theta}_{k,r} = \frac{\alpha^{(z)} + \sum_{t=1}^{T} \sum_{n=1}^{N_t} \delta(z_{t,n} = k) x_{t,n,r}}{\alpha^{(z)} + \beta^{(z)} + \sum_{t=1}^{T} \sum_{n=1}^{N_t} \delta(z_{t,n} = k)},$$

where $\theta_{k,r}$ is the probability of recipient r appearing in an email sent to group k, and is estimated as the fraction of emails sent to group k in which recipient r was included (smoothed by $\alpha^{(z)}$ and $\beta^{(z)}$).

Figure 5.6a shows the estimated value of the β parameters by taking the average sampled value across the 200 samples, with the shaded regions indicating one standard deviation across the samples. Comparing this plot to Figure 5.5a, the model is able to learn the correct values of the parameters, even when the email rates are relatively small for both groups (e.g. days 210 to 300 in Figure 5.5d).



(a) Group-specific parameters $\exp(\beta)$

(b) Weekday-specific parameters $\exp(\alpha)$

(c) Posterior probability of changepoints.

Figure 5.6: Estimated parameters for the simulated email user.

The values of the α parameters were estimated such that each day of the week has a separate parameter. Figure 5.6b shows the average value (and one standard deviation) of each α parameter across the 200 samples. The estimated values are within one standard deviation of the true values of α from Figure 5.5b. Note that the α parameter for the first day of the week has zero variance — we arbitrarily fix this parameter, thus allowing the other α and β parameters to scale accordingly[2].

Figure 5.6c shows the posterior probability of a changepoint across days for the two groups. The posterior probability of a changepoint occurring on day t with respect to group k is estimated as the fraction of samples where $s_{k,t} \neq s_{k,t+1}$. For both groups, the peaks in the posterior probability correspond to the true changepoints in the simulated data. This indicates that the model is able to both decompose the overall email activity appropriately across the two groups, while simultaneously learning when user's behavior with respect to each group changes over time.

[2] An alternative approach, more akin to regression models, would be to instead model an intercept term in the definition of $\lambda_{k,t}$ in Equation 5.3

The presented results are intended to be illustrative and demonstrate that the learning algorithm for the model is behaving as expected — the primary interest is the interpretation of the model parameters when applied to real data, discussed in the following section.

5.6 Exploratory Analysis Using Real-World Email Data

In this section, we interpret the estimated model parameters when inferred from the email histories of two Gmail datasets and one Enron dataset, described in Section 1.1. For each dataset, the number of groups learned is heuristically set to $K = 2\sqrt{R}$ (rounded to the nearest integer), where R is the total number of unique recipients the user has emailed over the time period. We found this to be a simple and effective heuristic for automatically choosing the number of groups to learn. An extension to the model would be to learn K automatically from the data, e.g., using Bayesian non-parametric techniques[3].

We use the same initialization of model parameters and configurations of the hyperparameters as the simulated dataset in Section 5.5. A total of 21000 Gibbs samples are produced, discarding the first 1000 as burn-in, and keeping every 20^{th} sample after that (for a total of 1000 samples). Model parameters are estimated from these samples in a similar manner as described in Section 5.5.

Figure 5.7 shows the learned parameters of the model for the different email users, where columns represent users. The top row of plots consist of the number of emails sent each week by each user. The second row shows the time intervals for which there was significant activity between the user and each group, where activity is represented as horizontal bars along the x-axis. Such intervals were determined by thresholding the value of the β parameters for

[3] We chose not to follow this approach because of the additional complexity required to initialize a HMM for a new group, to average across samples for estimating emailing rates, etc. Additionally, standard CRP modeling techniques are not directly applicable here as the distribution of latent group variables depend on the emailing rates for each group — not a Dirichlet process prior.

each group. The third row of plots show the number of emails the user had sent to two particular groups over time. The last row shows the learned β parameters for these two groups, again across time. The posterior probability of changepoints is not shown in this figure, as the true changepoints are unknown. However, changepoints are clearly seen with respect to the two chosen groups by the sharp changes in the β parameters over time.

The left column of Figure 5.7 shows the estimated model parameters for one of the Gmail datasets. There is evidence in the group activity that a significant change occurred in this user's behavior, around the middle of 2009. This changepoint corresponds to the user transitioning between two academic institutions: connections between the user and individuals at the original institution faded as new connections were formed as a result of the move from one location to another. The bottom two plots of the first column show email activity between this user and two learned groups, each corresponding to a separate research project the user participated in. These two research projects were associated with a small number of common participants (e.g. the same faculty advisors), illustrating that the model can use co-appearance information to successfully disambiguate groups which share common subsets of recipients. Spikes in behavior between the user and two groups correspond to different major deadlines, e.g., progress reports, paper submissions, presentations, etc.

The center column of Figure 5.7 shows the estimated model parameters for a different Gmail user — the top two plots indicate that this user sends considerably more emails, and to more people, than the previous user. Email activity is low for the first year as the user was experimenting with new email client software, followed by a sudden change and increase in activity as the user switched all email activity to the new client. The second row shows a gradual accumulation of groups over time (e.g. more horizontal bars start to appear), containing both groups that the user communicates with on a regular basis as well as those which are active for only specific intervals of time.

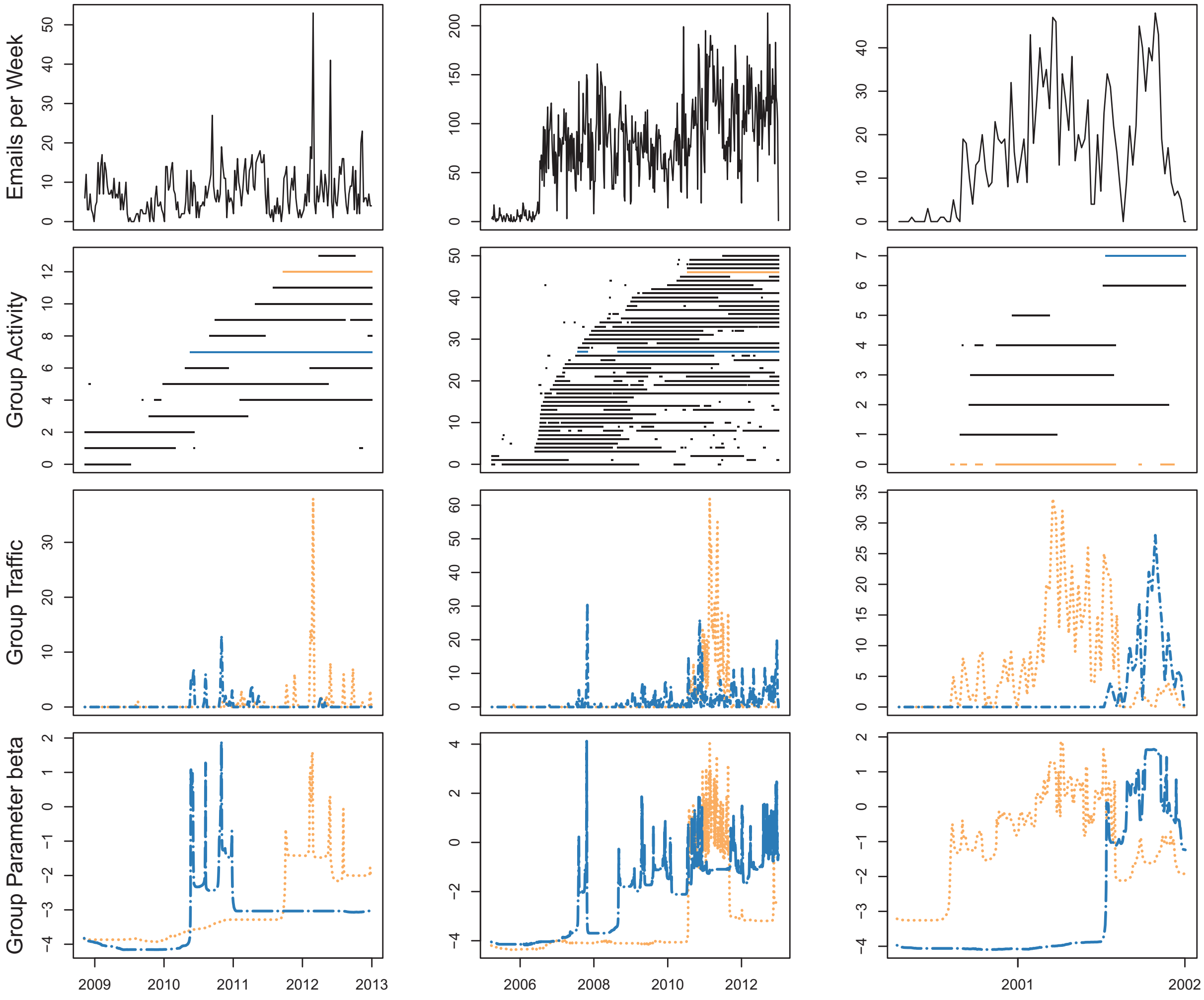


Figure 5.7: Exploratory analysis from fitting the model to real-world email data. Each column corresponds to the email history of a specific individual. First row: observed weekly email volume. Second row: greater than average β values for each learned group over time. Third row: number of emails per week assigned to two chosen groups. Fourth row: learned parameters β for the chosen groups. Figure generated by Chris DuBois and published in Navaroli et al. (2013).

The bottom two plots of this column show the email traffic and estimated β parameters between the user and two specific groups learned from the data. The dashed (blue) group corresponds to a project where the user is a principal investigator for a large proposal and the prominent group members are six other faculty members at a number of different institutions. There is increasing activity in mid-2007 due to proposal preparation, then a spike at the proposal deadline itself in late 2007, followed by a quiet period until early 2008 when the project was funded. The group activity is then characterized by relatively low levels of activity for the next few years as the project proceeded, punctuated by spikes of more intense activity once or twice a year around the times of project review meetings. The dotted (orange) line shows a different group corresponding to about 15 individuals involved in the organization of a large conference. The activity of this group ramped up in mid-2010 as the user became involved in conference organization, followed by roughly 12 months of relatively high activity until the actual conference in Summer 2011.

The third column in Figure 5.7 illustrates the estimated model parameters for an active user from the Enron email corpus. This user's email activity increased in early 2001 and there appears to have been a significant change in recipient groups near the middle of the year, as activity for some groups stop while new activity begins for others. The bottom two plots indicate the user's communication between two different groups, one before the apparant change and one afterwards. The significant change in behavior suggests that the Enron employee may have moved between office locations or to a different organizational branch of Enron around this time.

Similar interesting patterns were found for other email users and other groups with respect to the users detailed in this section. Most of the inferred groups were focused on small subsets of email recipients and were active during specific intervals of time. A few of the less-prominent groups consisted of "singleton" groups consisting of just a few emails sent to a single person, and "combined" groups which merged two clear disjoint subsets of contacts together (but appeared in similar parts over time). This is most likely due to the heuristic number of groups inferred being too few (for the joined groups) or many (for the singleton groups) for the user, thus in effect the model is under- or over-fitting to the data. Being able to set the number of inferred groups non-parametrically from the data would be a useful extension to the proposed model, and is an avenue of future work discussed in Section 5.8.

5.7 Experiments

In this section we describe two different sets of experiments. In Section 5.7.2 the quality of the learned groups produced by our model is explored, and in Section 5.7.3 the predictive performance of our model is compared to different baselines.

5.7.1 Email Datasets and Experimental Setup

The email accounts used in these experiments include:

- three users from the Gmail email corpus in Section 1.1[4].
- five Enron users which sent the most emails over the course of their email history[5].

The experiments were performed independently across all email datasets.

Priors over the model parameters, in addition to the estimation procedure of the parameters, are similar to that discussed in Sections 5.4 and 5.5 respectively. As in Section 5.6, the number of groups inferred for a particular user is heuristically set to $K = 2\sqrt{R}$. A total of

[4] Data for “Gmail A” from Table 1.1 was unavailable during the time the following results were compiled.

[5] While it is theoretically possible to apply the model to all email datasets described in Section 1.1, the time-varying nature of the model requires a substantially large dataset to infer meaningful parameter values.

1100 samples is collected, with the first 100 samples discarded for burn-in and every tenth sample kept after that, for a total of 100 samples used for parameter estimation.

5.7.2 Quality of Inferred Groups

In the first experiment, we quantitatively measure the quality of inferred groups of our model. We define a quality metric with respect to grouping parameters, and show that our model infers higher-quality groups than simpler baseline methods.

Evaluation Metric: Group Coherence

To measure the quality of the learned groups, we define a *coherence* metric which measures the co-appearance frequencies of recipients in a group. These frequencies are calculated with respect to the emails assigned to that particular group. For group k, the coherence measure is formally defined as

$$c_k \equiv \frac{1}{C_k} \left(\sum_{q=1}^{R} \sum_{r=1}^{R} P_k(q) P_k(r) P_k(q, r) \right)$$

where $C_k \equiv \sum_{q=1}^{R} \sum_{r=1}^{R} P_k(q) P_k(r)$ is a normalization constant, $P_k(r)$ the fraction of emails sent to group k that recipient r appears in, and $P_k(q, r)$ the fraction of emails sent to the group where recipients q and r both appear. The values of $P_k(q), P_k(r)$, and $P_k(q, r)$ are estimated based on the latent z variables from the highest log-likelihood sample obtained from the Gibbs sampler. For example, $P_k(q, r)$ is defined as

$$P_k(q, r) \equiv \frac{\sum_{t=1}^{T} \sum_{n=1}^{N_t} \delta(z_{t,n} = k) x_{t,n,q} x_{t,n,r}}{\sum_{t=1}^{T} \sum_{n=1}^{N_t} \delta(z_{t,n} = k)}$$

The intuition behind the coherence measure c_k is that it rewards placing pairs of individuals into the same group who often co-appear in the same emails — conversely, it penalizes putting pairs of recipients into the same group that rarely co-appear in an email. The coherence is maximized at $c_k = 1$ when all pairs of individuals in a group co-appear in all emails assigned to that group. As the co-appearance graph of recipients becomes less well-connected, the value of c_k decreases.

Consider for example a group containing four individuals labeled A, B, C, and D. If every email contains all group members, the coherence of that group is 1. A coherence score of less than 1 will be assigned if emails are only sent to pairs of recipients, e.g., involving $\{A, B\}$, $\{B, C\}$, $\{C, D\}$, etc. An even smaller coherence score is assigned to groups where subsets of individuals never co-appear in an email, e.g., if all emails sent to the group involved $\{A, B\}$ or $\{C, D\}$, but never $\{A, C\}$. The overall coherence score assigned to the model is a weighted average of the group-specific coherence scores, where weights are defined by the total number of emails assigned to that group.

Baseline Models

We compare the coherence of the groups inferred with our model to those learned in four baseline models:

- The first baseline is identical to our proposed model, except that a multinomial distribution is used to model the membership probabilities, where $\sum_r \theta_{k,r} = 1$ (as in Navaroli et al. (2012)). A multinomial likelihood is placed over email recipients for this baseline, as opposed to a product of independent Bernoulli trials.
- The *single segment* baseline is similar to our model, but restricted to contain only a single time segment for each group. Thus, there is no time variation in the relative communication rates of groups. This baseline can be viewed as a probabilistic clustering

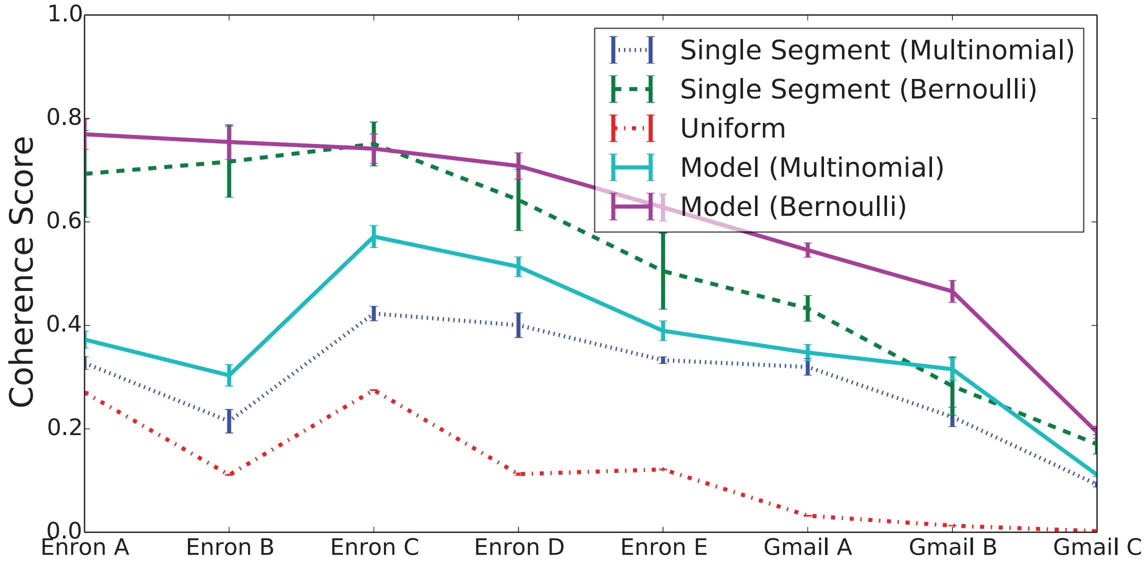


Figure 5.8: Coherence scores (y-axis) of the model and baseline methods, averaged across 20 random initializations of parameter values. The x-axis represents different email datasets.

of the recipients, where the (constant) rate of emails sent to each group determines the cluster mixing coefficients. We implemented both multinomial and independent Bernoulli versions of this baseline.

- The final baseline is a Uniform model, where the latent group assignments for emails $z_{t,n}$ are chosen uniformly at random among the K groups. Groups defined by this baseline model will be random, and are expected to have low coherence scores.

Results

To generate the coherence scores, the model and baselines were trained on the complete data (e.g. no test dataset) 20 separate times, each time randomly initializing the model parameters used in the Gibbs sampler. We then record the mean and standard deviation of the coherence score (averaged across weighted groups) across the 20 different sets of learned parameters.

Figure 5.8 shows the coherence scores for each email dataset (x-axis) and model. Values along the y-axis represent coherence scores for each model and baseline, with error bars indicating one standard deviation across different parameter estimates. For each user, the coherence of the groups learned by our model is higher than those learned by the baseline models. The increase in performance over the single segment baseline suggests that the dynamic nature of the user's interactions between the different groups can be exploited to further refine the inferred grouping structure.

It is interesting to compare the coherence results of Figure 5.8 between the multinomial and independent Bernoulli definitions for groups in our model (the turquoise and magenta solid lines, respectively). When using the multinomial distribution, the likelihood of email recipients is the product of membership probabilities for recipients appearing in the email. As this likelihood places probability only over recipients that appear in the email, the likelihood is not lowered if a high-probability recipient is absent in the email — groups can be easily created that consist of disjoint subsets of recipients which never co-appear in an email together. In contrast, the likelihood of an independent Bernoulli model over email recipients (Equation 5.4) is lowered if a high-probability recipient does not receive an email. This naturally penalizes groups combining disjoint subsets of email recipients together[6], as likelihoods for that group will always be low due to the lack of presence of at least one subset. The difference in modeling the recipients *not* receiving an email provides a plausible explanation for the observed increase in coherence score for the Bernoulli models compared to the Multinomial models in Figure 5.8.

5.7.3 Predictive Performance

In this section, we compare the model's predictive performance over held out test data against baseline models. In particular, we train the model and baselines over a training

[6] However, the merging of groups is still possible if the number of groups K is chosen to be too low.

dataset created by selecting 80% of the emails uniformly at random, treating the remaining 20% as unobserved test data[7]. We split the data randomly, as opposed to chronologically, primarily for evaluating the fitting of both grouping and time-varying rate parameters to test data occurring at different periods of time. As with the experiments of Section 5.7.2, the experimental procedure discussed here is applied to each email dataset independently.

Evaluation Metric: Test Log-Likelihood

To evaluate the fitting of the model and baselines to held out test data, we measure the *test log-likelihood* using parameters estimated from training data. The test log-likelihood is formally defined as

$$LL_{\text{test}} = \sum_{t=1}^{T} \sum_{n:\text{missing}} \log \left(\sum_{k=1}^{K} P(z_{t,n} = k | \lambda, \theta) P(x_{t,n} | \theta_k) \right),$$

where the second sum is over the emails on day t that are unobserved (i.e. in the test dataset).

Baseline Models

The predictive power of the proposed model is compared against four baseline approaches:

- The *uniform* baseline consists of a single group, whose Bernoulli membership probabilities $\{\theta_r : 1 \leq r \leq R\}$ are equal across all possible recipients and is set such that the expected number of recipients in an email ($\sum_r \theta_{k,r}$) matches that empirically found in the training dataset.

[7] One could also explicitly model the missing data by averaging over the missing information during MCMC sampling — however this would require a more complex sampling algorithm. Thus, we opted for the simpler approach of ignoring the missing data during training.

- The *single group* baseline corresponds to a maximum-likelihood estimate of an independent Bernoulli model over possible recipients (Equation 2.1), where θ_r is estimated as the fraction of emails that recipient r appears in.

- The *sliding window* baseline also consists of a single group, however the recipient probabilities on day t are estimated locally based over a window of time. This allows the membership probabilities to adapt to changes over time as the user's behavior towards different recipients change. Different sized windows up to two months were evaluated — only the optimal window in terms of test log-likelihood is reported here.

- The *single segment* baseline is identical to that from Section 5.7.2.

For the first three baselines, only a single group exists[8]. The test log-likelihood of these baselines reduce to

$$LL_{\text{test}} = \sum_{t=1}^{T} \sum_{n:\text{missing}} \log P(x_{t,n}|\theta).$$

Results

To evaluate the models, the experimental procedure was run 20 separate times for each email dataset. Each of the 20 experimental runs used a different, randomly generated, set of 80% observed data and 20% test data. For each training dataset, the model parameters are estimated using the method described in Section 5.5, obtaining estimates using Gibbs samples collected from that dataset. Test log-likelihoods are then calculated across the 20 test datasets, using model parameters estimated from their respective training datasets.

Figure 5.9 shows the test log-likelihood of the four baselines, relative to our model, across the Gmail and simulated datasets. Similarly, Figure 5.10 compares the test log-likelihood

[8] The *sliding window* baseline has membership probabilities changing over time, but uses a single group.

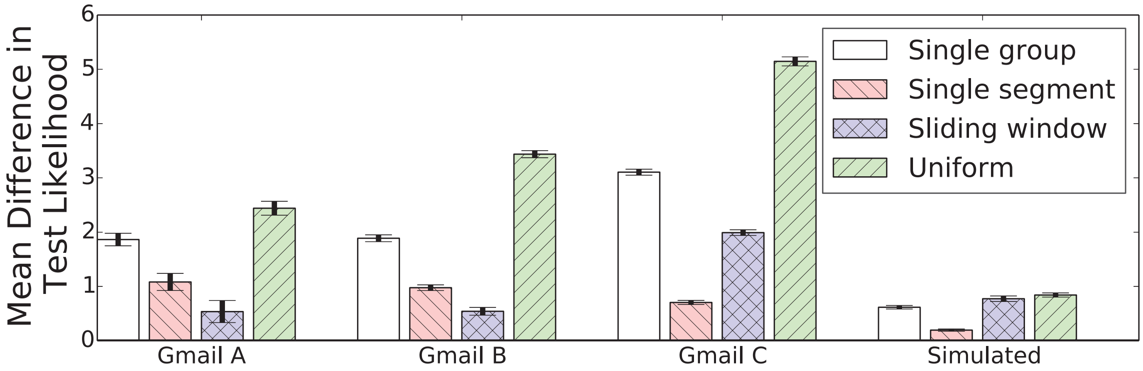


Figure 5.9: Differences in test log-likelihood between our model and baseline models, for different Gmail users (and the simulated user of Section 5.5).

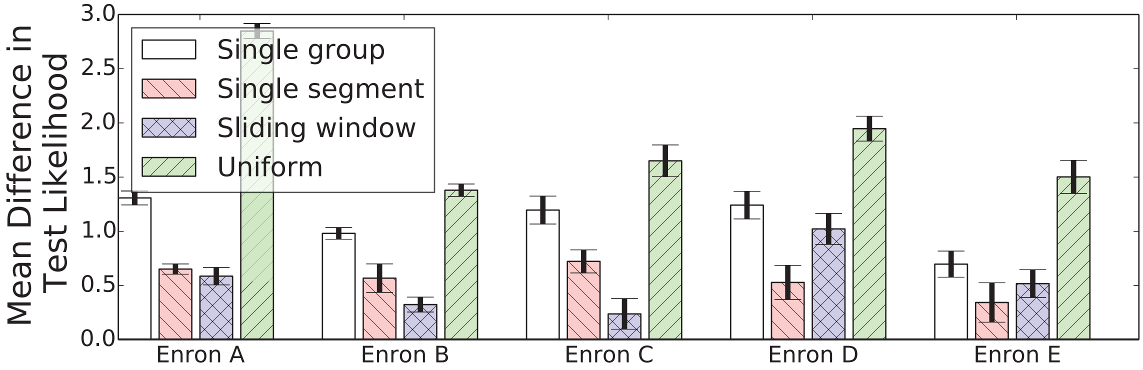


Figure 5.10: Differences in test log-likelihood between our model and baseline models, for different Enron users.

between the model and baselines for the Enron datasets. The y-axis is the mean difference in log-likelihood across emails in the test dataset between our proposed model and each baseline. The mean differences across the 20 test datasets are then averaged, with error bars showing one standard deviation. Larger positive differences indicate that our proposed model is a better fit to the test data, compared to the baseline. The "simulated" user makes use of the simulated dataset described in Section 5.5.

The results in Figures 5.9 and 5.10 show that our proposed model is systematically more accurate in test predictions than all of the baselines across the datasets. An interesting aspect of these results is the difference in the accuracies between the single segment and the sliding window baselines. For some users, the single segment baseline is more accurate than the sliding window baseline, as indicated by the lower relative differences compared to the proposed model along the y-axis. For these users, most of the predictive power is in the co-appearance structure in emails, e.g., the single segment baseline is able to effectively model several “active” groups on any given day. The “Gmail C” and simulated users are two examples of such users — the importance of co-appearance information was made apparent for the simulated user in Figures 5.4 and 5.5, where there is significant overlap in the group structure both in terms of membership and intervals of “active” time.

For users where the sliding window baseline is more accurate, most of the predictive power is in the appearance of recipients across emails locally in time. For example, the sliding window baseline is advantageous to use when there is at most a single group active on any day, which may change over time. The “Gmail A” and “Gmail B” users both achieve better predictive performance when using the sliding window baseline — the general activity of these users is that emails are typically sent to small subsets of people for some length of time, then they interact with different subsets of people as time progresses.

Regardless of whether the predictive power lies primarily in the co-appearance of recipients or when recipients receive emails, the results in Figures 5.9 and 5.10 show that our proposed model is more accurate in predicting unobserved test data. This illustrates that the model is able to capture useful information from the observed data, both in terms of temporal variation and group structure.

5.8 Discussion and Future Work

In this chapter, we have presented a statistical model for exploring and analyzing the communication behavior of a single user in an egocentric email network over time. This model can find interpretable groups of email recipients by leveraging the co-appearance patterns present in both email recipient lists and times where recipients receive emails. We illustrated the exploratory aspects of our approach by fitting the model to data from multiple real email accounts and interpreting the composition of the learned groups and the parameters governing their prevalence over time. Experiments over several email datasets, each spanning multiple years, indicated that the model yields improved predictive accuracy and coherence in group structure over a variety of baselines. While the model proposed in this chapter was described in the context of sending emails, it can be readily applied to broader types of multi-recipient directed communication data.

The model proposed in this chapter is a useful starting point for modeling data such as email histories, and there are a variety of potential extensions and generalizations that are worth exploring. For example, a more general Poisson regression framework could be employed, where additional exogenous covariates are incorporated into the model as well as detecting global segment boundaries that affect all groups simultaneously. Furthermore, real email data often exhibits intermittent bursts of activity "embedded" within longer sequences of lower-level activity, suggesting that a model allowing temporal bursts (e.g., as in Kleinberg (2003)), superposed on the segments, may be a useful avenue for further exploration.

A piecewise-constant framework is used by our model to define the group-dependent Poisson processes, where the emailing rates are constant between changepoints. While this is able to accurately reflect sudden changes in a user's communication behavior, it is clearly an approximation to real email activity. For instance, gradual changes in behavior over long periods of time (e.g. the slight increase in the daily number of emails sent over the years for the user in Figure 5.1) are poorly approximated. A natural extension to the model would be to augment the piecewise-constant framework to also incorporate such changes, as well as periodic trends such as seasons or holidays.

There are also numerous opportunities to extend the modeling choice of groups. For example, in the present work we fix the number of clusters or groups to learn, K. One could instead include a second non-parametric component to the group component of the model, allowing each email the opportunity to be sent to a newly created group of recipients. It would also be natural to allow groups to be related and dependent (e.g., a hierarchical clustering), as well as allowing the group membership probabilities to change over time.

5.9 Summary of Contributions

The primary contributions of this chapter include the following:

- We introduced a changepoint model for detecting changes in communication behavior based on latent state transitions of a hidden Markov model, which can be viewed as a non-parametric generalization of Chib (1998).
- We modeled the rate at which a user sends emails to different groups as piecewise-constant Poisson processes, with emailing rates defined as a function of the HMM latent state and the day of week.
- We introduced a mixed-membership model over email recipient groups, based on the independent Bernoulli distribution. We introduced dependencies between this model and the changepoint / Poisson process model by modeling the probability of the user sending an email to a specific group based on the Poisson process emailing rates.
- We derived a Gibbs sampling approach for simultaneously learning the model parameters describing 1) the common email recipient groups, 2) the user's emailing rates towards those groups, and 3) changepoints in activity between the user and each group.
- Results on simulated data show that the Gibbs sampling inference method is able to produce accurate estimates of parameters when conditioning on observed behavior.

- Results over histories of several email accounts spanning multiple years indicate that the inferred grouping structure of the model is superior to that of simpler baseline models, in terms of a coherence metric we defined. Additionally, experiments using held-out test data showed that the proposed model is a better fit to unobserved data compared to baseline models.

Chapter 6

Probabilistic Modeling of Communication Response Behavior

While previous chapters focused on the modeling of itemsets and using timestamp information to refine clusterings of items into coherent groups, the timestamp information was treated as metadata — their values were assumed fixed and conditioned on. In this chapter, we instead treat the itemsets as metadata and investigate the nature of the timestamps in the context of *event response* frameworks, where each {itemset, timestamp} pair is considered an event, potentially created in response to a previous event in time.

We define an event $e \equiv \{\tau, s, m\}$ to be an observable action where

- $\tau \in \mathbb{R}^+$ is the timestamp that the event occurred
- $s \in \{1, 2, \cdots, S\}$ is the actor who initiates the event
- m is the metadata describing the event

As an example, consider the event of a text message being sent at time τ. The actor would be the person that composed and sent the message, and metadata could include the content of the message, recipients of the message, and GPS coordinates of the actor's location. We will assume that the metadata additionally includes information regarding whether or not the event was a response to a previous event in time, i.e. if the text message was a reply to a previously received message. Other types of data that fall under the event response framework are described in Table 6.1.

Data Type	**Event**	**Actor**	**Metadata**
Email	Sending an email	Email Sender	Email Recipients
Instant Messaging	Sending a message	Message sender	GPS coordinates
Twitter	Posting a (re)tweet	Tweet sender	Hashtags
Wikipedia	Making an edit	The editor	Content changed

Table 6.1: Examples of event data.

Here we are interested in the task of modeling the behavior of a single individual (also referred to as the "ego") towards others. In particular, we investigate *if* and *when* the individual responds to previous events, such as replying to a previously received text or email. We consider events from an egocentric perspective, where all events involve the individual or ego. For example, in text messaging data the ego may be the owner of the phone receiving and sending text messages, and in email communication data the ego is the owner of the email inbox whose behavior towards others is being modeled. In this chapter, we will focus on the application of email communication[1], where metadata contains the recipient lists of emails (i.e. no subject or body text is utilized by the proposed models).

For such data, we distinguish between different types of events. First, *incoming* events are those that are directed towards the individual (e.g. a received email); for these events, the actor would be someone other than the individual. Second, *response* events are those where the ego takes some action in response to an incoming event (e.g. replying to an email); for

[1] The methods described in this chapter are also applicable to other types of event or communication data, such as those described in Table 6.1

these events, the actor will always be the ego. Lastly, there are events initiated by the ego that are not response events, such as the ego starting an email conversation. For the modeling purposes of this chapter, we focus on the response events from the ego towards incoming events.

Let $e_i = \{s_i, \tau_i, m_i\}$ represent the ith incoming event the individual receives. The first problem of interest is determining the number of replies $N_i \geq 0$ the individual will send in response to e_i. In communication data, the number of replies is typically 0 (no reply was given) or 1 (a reply was given). In rare occasions, $N_i > 1$, e.g. the ego sends follow-up replies to the same email. For predicting N_i, we will first predict the binary outcome of whether or not $N_i > 0$. Next, if conditioning on $N_i > 0$, we predict the value of N_i.

For an incoming event e_i where $N_i > 0$, let $r_{ij} = \{s_{ij} = \text{ego}, t_{ij} > \tau_i, m_{ij}\}$ represent the individual's jth response to that event. The next problem of interest is to predict the time of the response $t_{ij} > \tau_i$. In particular, we focus on probability distributions of the form $p(\Delta t_{ij}|\Omega)$, where $\Delta t_{ij} \equiv t_{ij} - \tau_i$ is referred to as the *response time*. An example sequence of events in the context of email communication, labeled under the event response framework described thus far, is provided in Table 6.2.

Event	**Time**	**Type**	**Notation**
Person A sends email to ego	τ_1	Incoming	$e_1 = \{m_1, \tau_1, s_1 = A\}$
Person B sends email to ego	τ_2	Incoming	$e_2 = \{m_2, \tau_2, s_2 = B\}$
Ego replies to Person A	$t_{11} > \tau_1$	Response	$r_{11} = \{m_{11}, t_{11}, s_{11} = \text{ego}\}$
Person B sends another email to ego	τ_3	Incoming	$e_3 = \{m_3, \tau_3, s_3 = B\}$
Ego sends follow-up to Person A	$t_{12} > \tau_1$	Response	$r_{12} = \{m_{12}, t_{12}, s_{12} = \text{ego}\}$
Ego replies to B's second email	$t_{31} > \tau_3$	Response	$r_{31} = \{m_{31}, t_{31}, s_{31} = \text{ego}\}$

Table 6.2: Sequence of labeled communication events. According to this table, $N_1 = 2$, $N_2 = 0$, and $N_3 = 1$.

In this chapter, we consider egocentric communication datasets (e.g. all emails in an individual's inbox), where $E = \{e_i : 1 \leq i \leq N\}$ denotes all incoming events the individual receives (e.g. received emails) and $R = \{r_{ij} : 1 \leq i \leq N, 1 \leq j \leq N_i\}$ all responses the individual sends to incoming events.

The rest of the chapter is structured as follows. We begin by describing related models of event/communication responses in Section 6.1. The problem of whether or not the individual will respond to an event is then discussed in Section 6.2, with corresponding experiments in Section 6.3. Section 6.4 discusses the difficulties in modeling event response times due to circadian rhythms, and a parametrization of such rhythms is introduced to measure *effective* response times. The modeling of effective response times is applied to two different models in Section 6.5, with Section 6.6 detailing an inference method for learning an ego's circadian rhythms. Experiments showing the benefits of modeling effective response times are shown in Section 6.7. We then conclude with a discussion and summary of contributions.

6.1 Related Work

The problem of modeling interaction and communication networks in general has been well-studied. A common task for example is identifying groups of entities within the interaction network that share common behavioral patterns. These methods are usually applied over the aggregate network, i.e. events aggregated over time (McAuley and Leskovec, 2012), or to a dynamic network changing over time (Navaroli et al., 2013; Tantipathananandh et al., 2007). Related problems specific to email communication include the identification of roles or relations between entities within these communities (Aliabadi et al., 2013), suggesting relevant recipients of an email being composed (Carvalho and Cohen, 2008), predicting whether or not an email will be responded to (Aberdeen et al., 2010; Dredze et al., 2005), and learning common topics discussed between individuals (McCallum et al., 2007). In this chapter, we focus on egocentric email networks and investigate the task of if and how long it takes for one to reply to another's email.

Strong circadian and weekly rhythms are usually present in an individual's response behavior, and are a contributing factor to the "bursty" behavior where responses are usually sent quickly or after long periods of inactivity (Barabási, 2005; Malmgren et al., 2009). In the context of modeling the time it takes for an individual to respond to an event (e.g. an email), previous work has either ignored periods of inactivity (Masuda et al., 2013) or modeled the bursty behavior using power-law (Barabási, 2005; Eckmann et al., 2004) and lognormal (Kaltenbrunner et al., 2008; Stouffer et al., 2006) distributions. The lognormal distribution has also been used to model the timing of "tweets" in Twitter data (Zaman et al., 2014). While these distributions model the bursty and long-tailed behavior in interaction and communication data, they do not directly model circadian and weekly rhythms.

Another class of models over temporal patterns of response behavior include point processes, which model the rate at which events (e.g. replies) occur. In particular, various forms of Poisson processes have been shown to be effective in capturing bursty communication behavior (Malmgren et al., 2008; Simma and Jordan, 2010). A special form of a non-homogeneous Poisson process is the Hawkes processes (Hawkes, 1971), where the overall rate of responding to *any* event is the superposition of many independent response rates, one for each event that occurs. In addition to modeling response times, Hawkes processes have been used in order to infer latent relationships between individuals in social networks (Blundell et al., 2012; Fox et al., 2013; Zipkin et al., 2014), along with modeling dyadic interactions (Halpin and De Boeck, 2013; Masuda et al., 2013). In this chapter, we will apply Hawkes processes to egocentric communication data, modeling the ego's response behavior towards others with respect to their circadian and weekly rhythms.

While the approaches above are able to model bursty behavior in communication data, the circadian patterns of the ego are not explicitly accounted for. For Poisson process models of communication behavior, circadian and daily patterns have been accounted for by decomposing the event rate $\lambda(t)$ into a product of terms based on differently-scaled time intervals (e.g. daily, hourly). Such a decomposition has been proven successful in describing circadian patterns of email communication (Malmgren et al., 2009), instant messaging (Pozdnoukhov and Walsh, 2010), and telephone calls (Scott, 2000).

The proposed modeling of circadian rhythms is most similar to the work of Jo et al. (2012), who investigated the rescaling of time via histograms to remove circadian and weekly patterns in exploratory data analysis of individual communication events. Our work is similarly motivated but extends this earlier work in several significant respects. First, we use smooth non-parametric kernels (rather than histograms) to model temporal patterns. Second, we demonstrate how the transformation of time can be effectively embedded in different statistical models such as Hawkes processes. Lastly, we quantify the improvements gained by transforming time with a series of systematic prediction experiments on out-of-sample data.

6.2 Modeling the Number of Responses to an Event

In this section we focus on modeling the number of responses $N_i \geq 0$ the individual or ego sends to an incoming event e_i. It is assumed that N_i is independent of the ego's circadian and weekly patterns, i.e. whether or not the ego will reply depends on the *content* and *intent* of the event, not when the event was received. Thus, we save details on such patterns for Section 6.4.

Consider the empirical distributions over the number of responses sent back to an email for individuals in the Gmail and Enron email corpora[2], shown in Figure 6.1.

Two clear patterns are seen universally across all individuals in Figure 6.1. First, the majority of emails received by an individual go unanswered (approximately 85–90%). Second, the number of emails receiving $R + 1$ replies is a small fraction of those that received R replies,

[2] The Eckmann and Manufacturing email corpora do not include the response structure of emails. Thus, they will not be included for analysis throughout the remainder of this chapter.

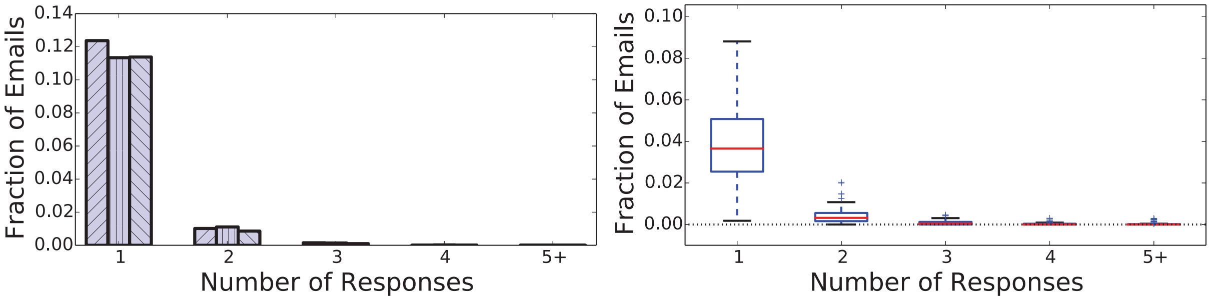


Figure 6.1: Number of responses to emails, across datasets in the Gmail (left) and Enron (right) email corpora.

for $R \geq 1$. This is not surprising — many emails do not require a response; examples include those that are irrelevant, concluding emails to earlier conversations, automated messages requiring no responses, spam, etc. For emails that do receive a response, a single response is by far the most common case. Multiple responses may occur in the case of follow-up emails being sent or corrections to a previous reply.

With these characteristics in mind, we propose the following distribution over $N_i \geq 0$:

$$N_i \sim (1 - \gamma_i)\delta(N_i = 0) + \gamma_i \text{Geometric}(\theta) \tag{6.1}$$

which is a mixture between a point mass at $N_i = 0$ and a Geometric distribution over positive values. Under this parameterization, γ_i is interpreted as the probability of sending *any* response to event e_i, regardless of how many (e.g. $P(N_i > 0)$).

Assuming the number of responses N_i sent to event i is independent of all over events, the likelihood of the model described by Equation 6.1 for $\{N_i : 1 \leq i \leq N\}$ is

$$P(\{N_i\}|\{e_i\}, \Omega) = \left(\prod_{i:N_i=0} (1 - \gamma_i) \right) \left(\prod_{i:N_i>0} (\gamma_i) \right) \left(\prod_{i:N_i>0} (1 - \theta)^{N_i - 1} \theta \right) \tag{6.2}$$

This likelihood conveniently decomposes into the product of two functions — one involving $\{\gamma_i\}$, and one involving θ. Thus, maximum-likelihood (ML) estimations of their values can be independently calculated. The ML estimation of θ is straightforward due to Geometric distribution form and has the value of:

$$\hat{\theta} = \frac{\sum_{i:N_i>0} 1}{\sum_{i:N_i>0} N_i}$$

The ML estimation of γ_i varies, depending on its form. If $\gamma_i = \gamma \in [0, 1]$ is a global Bernoulli probability of responding to an event, then the ML estimate of γ would be the empirical fraction of events that received a response. However, we instead follow a logistic regression approach and model γ_i as

$$\gamma_i = \frac{1}{1 + \exp\left(-X_i^T \beta\right)} \tag{6.3}$$

where X_i is a feature vector derived from the event's metadata m_i, and β a vector of regression coefficients. To obtain ML estimates of the regression coefficients β, we note that log of the likelihood in Equation 6.2 is proportional to

$$\begin{aligned} \log P(\{N_i\}|\{e_i\}, \Omega) &\propto \sum_{i:N_i=0} \log(1 - \gamma_i) + \sum_{i:N_i>0} \log(\gamma_i) \\ &= - \sum_{i:N_i=0} X_i^T \beta - \sum_{i=1}^{N} \log(1 + \exp\left(-X_i^T \beta\right)) \end{aligned}$$

While taking the derivative of the above log-likelihood is straightforward, a closed form solution does not exist for the ML estimate of β. However, estimates can be computed numerically using iterative Newton-Raphson updates (Bishop, 2006, Section 4.3.3).

6.2.1 Logistic Regression Features for Email Data

Before we describe the regression features X_i used for the experiments in Section 6.3.3, we define the *mode of reception* of an event e_i, specific to the context of email communication. This is a categorical variable, and has one of the following values according to Table 6.3.

Mode of Reception	Description
DIRECT	The email was sent to only one person — the ego.
TO	Email has multiple recipients; ego appears in “to” field.
OTHER	Email has multiple recipients; ego does not appear in “to” field.

Table 6.3: Different modes of reception for an email.

The “other” mode of reception can range from the individual appearing in the “cc” or “bcc” field, to not appearing at all (e.g., the received email was sent to a mailing list). The intuition behind this variable is that the individual’s probability of replying to a received email may depend on how it is received. For example, we would expect that an email sent directly to the individual has a higher probability of being responded to, compared to one where the individual is one of many that were cc’d. Figure 6.2 confirms this intuition by showing the fraction of emails replied to as a function of mode of reception, across all individuals from the Gmail and Enron corpora (see Section 1.1).

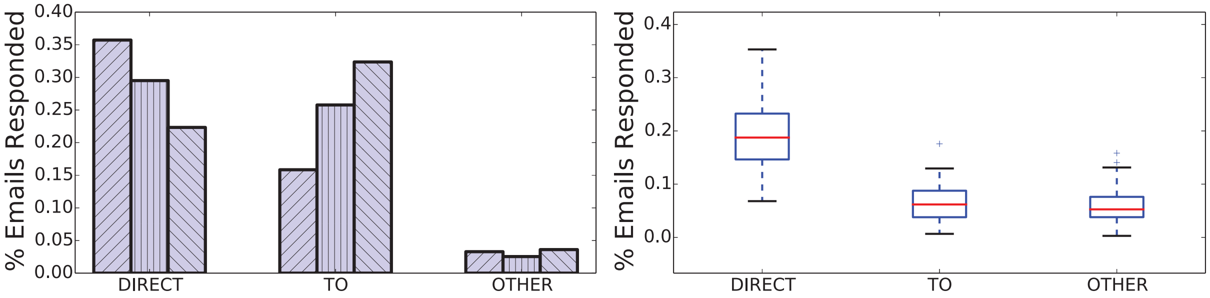


Figure 6.2: Fraction of emails responded to as a function of email reception (see Table 6.3) across datasets in the Gmail (left) and Enron (right) email corpora.

With this definition in mind, the regression features X_i used in Equation 6.3 (to predict whether or not email e_i will be replied to) are defined as

- Indicator variables representing the email’s mode of reception, with a regression coefficient defined for each mode. Motivation of this was shown in Figure 6.2.

- Indicator variables representing the sender of the email s_i, with a regression coefficient defined for each possible s_i. This models the intuition that the ego is more likely to respond to certain people, compared to others.

- The length of the corresponding email thread, in terms of email exchanges. A single regression coefficient is defined, and models how (un)likely the ego will respond to an email thread as it continues to grow.

- The number of people that received the email, minus one. By subtracting one, the value of the corresponding regression coefficient will not compete with that of the "direct" recipient mode (which ensures only one person received the email).

- An arbitrary "constant" feature set to 1, whose corresponding regression coefficient is interpreted as the intercept term.

6.3 Experiments over Number of Responses

In this section, the fitting of the model over the number of responses to an event is explored, as described in Section 6.2. A description of the datasets used in these experiments is given in Section 6.3.1, with the experimental procedure detailed in Section 6.3.2. Results are presented in Section 6.3.3.

6.3.1 Description of Datasets

The experiments of this chapter focus on the response behavior of email inbox owners in the Gmail and Enron email corpora described in Section 1.1[3]. The activity of each inbox owner or individual is considered a separate dataset; experiments are independently run across all

[3] The Eckmann and Manufacturing email datasets are not considered here, as the response structure (i.e. which emails are responses to which) is unknown

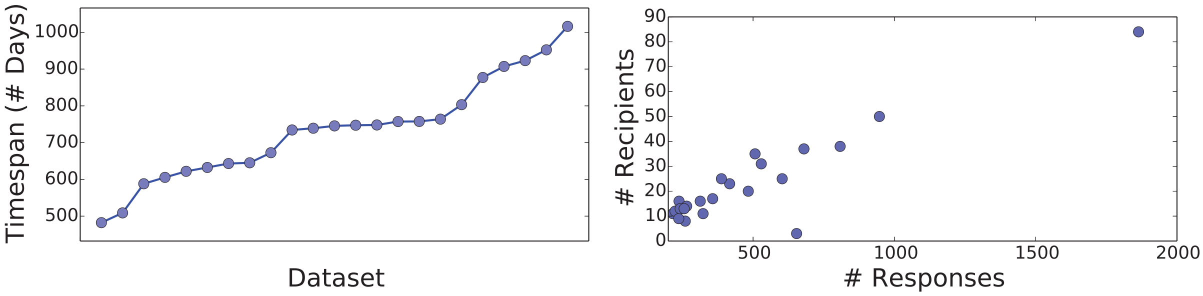


Figure 6.3: Summary statistics of the Enron datasets used in the experiments.

individuals. This setup is inspired by the practical scenario where the model is part of an individual's email client, having access to only that individual's response behavior.

To ensure that the models described in this chapter have sufficient data for reliable parameter inference, only inbox owners that have sent responses to more than 200 emails are considered. This threshold resulted in a total of 26 email datasets; three from the Gmail corpora (summary statistics shown in Table 6.4), and 23 from the Enron corpora (summary statistics shown in Figure 6.3)[4].

Ego	First Email	Last Email	# Responses	# Recipients
Gmail A	November 2007	July 2014	4769	211
Gmail B	November 2007	June 2014	1780	71
Gmail C	August 2006	May 2014	26104	678

Table 6.4: Summary statistics of the Gmail datasets used in the experiments.

6.3.2 Experimental Setup

All models are trained and evaluated in an online setting where new data is continually being observed. The first 20% (chronologically) of each individual's email inbox is used for initialization — initial estimates of model parameters are derived from this data. The remaining 80% of the data is then split into minibatches, with each minibatch containing all

[4] The "# Recipients" statistic counts the number of people that received 5+ responses from the individual

emails in the individual's inbox over a single 24-hour period. The minibatches are processed chronologically, according to the following procedure:

1. All log-likelihood and prediction metrics are calculated across the appropriate events in the minibatch, using the estimation of model parameters derived from all data up to, but not including, the current minibatch. The different metrics are discussed in detail in the appropriate subsections.

2. The estimation of the model parameters is updated to consider all events from the minibatch.

This processing of minibatches simulates the realistic scenario where the model is updated on a daily basis, and predictions for all emails on a particular day must be made with respect to estimated parameters derived from data up to the previous day.

6.3.3 Experimental Models and Results

The main model of interest is that proposed in Section 6.2, where the number of responses N_i the individual sends to event i is modeled as a mixture between a point mass at $N_i = 0$ and a Geometric distribution over values of $N_i > 0$. In this model, the mixing proportion $\gamma_i \equiv P(N_i > 0)$ (see Equation 6.1) is calculated via logistic regression. We compare this model to a baseline Geometric distribution over values of $N_i \geq 0$ (i.e. no logistic regression component).

Different variations of the logistic regression component described in Section 6.2.1 were also used in this experiment, with each variation using a different subset of the described features. We found that the quality of these variations were inbetween the baseline Geometric model and the "complete" logistic regression model using all features described in Section 6.2.1.

Thus, results regarding these variations are not shown, focusing on the comparison between the baseline and complete logistic regression models.

For the logistic regression model, estimated values of the regression coefficients β are regularized by optimizing the log-likelihood in addition to a L2 norm, preferring values of β with smaller magnitudes. We found that the estimated values of β were largely unaffected when using small to moderate values in the strength of the regularization. For the Geometric distributions (one over $N_i > 0$ as in Equation 6.1 and one over $N_i \geq 0$ as a baseline), a $\text{Beta}(9, 3)$ prior is placed over their parameter θ, resulting in maximum-a-posteriori (MAP) estimations. Relative to the size of the 20% initial data used for initial estimates of θ, we found this to be a weak prior.

Evaluation Metrics

For these experiments, we consider two evaluation metrics for comparing the fitting of the model and baseline to the data. The first is the average log-likelihood over the number of responses to an event (e.g. the first term in Equation 6.7, divided by the number of incoming events in the minibatches of data). The second metric is the area under the ROC curve (AUC), calculated by sorting the probabilities $\{\rho_i \equiv P(N_i > 0)\}$ and their corresponding outcomes via ground truth (e.g. whether or not each event was responded to). The probability ρ_i is calculated to be

$$\rho_i = \begin{cases} \gamma_i & \text{for the logistic regression model described in Section 6.2} \\ 1 - P(N_i = 0) & \text{for the Geometric baseline over values of } N_i \geq 0 \end{cases}$$

Note that both the log-likelihood of N_i and the probability ρ_i are calculated using parameters estimated from data up to the day before event i takes places, as described in Section 6.3.2.

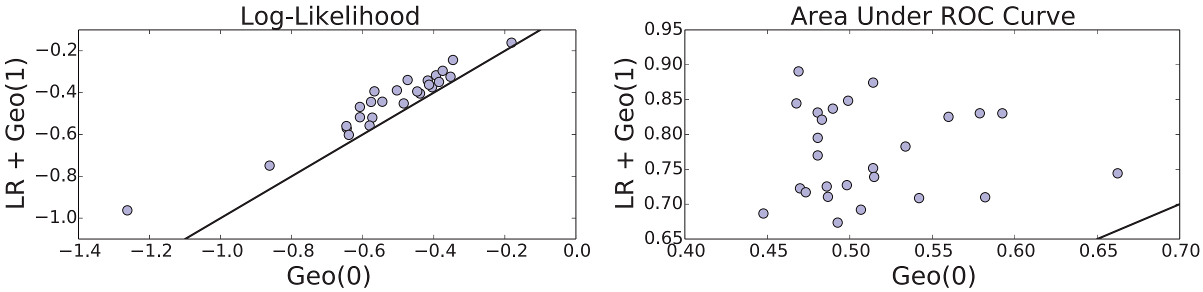


Figure 6.4: The log-likelihood (left plot) and AUC (right plot) for both the baseline Geometric model (x-axis) and the model defined by Equation 6.1 (y-axis) over $\{N_i\}$, across all email datasets.

Results

Figure 6.4 compares the average log-likelihood across all $\{N_i\}$ (left plot) and the AUC when predicting whether or not $N_i > 0$ (right plot) between the baseline model ("Geo(0)") and the model from Section 6.2 ("LR + Geo(1)"). Each point in the figure represents a separate email dataset. A systematic increase in both log-likelihood and AUC is seen across all 26 email datasets when modeling N_i as the mixture model in Equation 6.1 versus a standard Geometric distribution for values of $N_i \geq 0$. In particular, the increase in AUC is substantial across all email datasets.

This confirms the discussion from Section 6.2 that the probability of an individual replying to an email depends heavily on features derived from the metadata, in particular who sent the email and how the email was received (see Table 6.3). To better understand how the regression features affect the estimated probability of sending a response, Figure 6.5 shows the values of the different regression coefficients across email datasets. These values are those estimated after all minibatches of data are processed. The left plot illustrates that the individual's probability of replying to an email decreases as more people receive that email (the negative coefficient). This follows the intuition that emails sent to a large number of people are likely to be "announcement" emails that serve to portray information, not requiring a response from those which received it. Additionally, the positive coefficient over

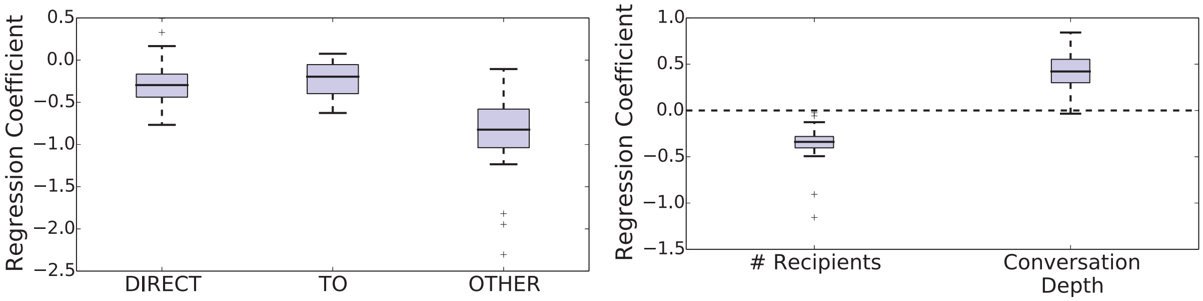


Figure 6.5: Values in estimated regression coefficients across email datasets. Left: coefficients related to the number of recipients of the email and the length of the email conversation thread. Right: coefficients for the different mode of reception (see Table 6.3).

conversation depth indicates that the probability of a response increases as the corresponding email "thread" continues to grow, as one would expect.

The right plot in Figure 6.5 shows how the probability of a response is affected by the way the individual received the email (as defined by Table 6.3). The values of the corresponding regression coefficients match the empirical statistics shown in Figure 6.2, where the probability of responding is greatest when the individual is the only recipient of the email (the "DIRECT" mode), and lowest when the individual is cc'd with others (the "OTHER" mode). Note that the "TO" mode of reception describes emails where the individual is one of many which appeared in the "to" field of the email header. Even though the corresponding coefficient is greater than the "DIRECT" mode, the overall probability of a response is lower due to the negative "# recipients" regression coefficient.

Although not shown in Figure 6.5, the regression coefficients varied greatly across the different email senders (e.g. who sent the email to the individual), ranging from large negative values to large positive values. This agrees with our intuition from Section 6.2.1 that the chances of the individual responding to an email varies based on sender; this pattern in coefficients was generally seen across all email datasets.

6.4 Circadian Rhythms in Event Response Times

For the remainder of the chapter, we will focus on the modeling of the time it takes for an individual to respond to an event e_i, conditioned on at least one response being sent (e.g. $N_i > 0$). We begin by discussing the time it takes for the individual to respond to an event with respect to their circadian rhythms.

In communication data, strong circadian (daily) and weekly patterns are commonly found in an individual's behavior. For example, in email communication in a professional environment, people are generally more active and likely to write an email during the daytime and on weekdays. Figure 6.6 shows the email usage of two different individuals[5] across multiple years, where the number of emails each has sent over every hour of the week is displayed. The circadian and weekly patterns are prevalent for both individuals; both typically send less emails over the weekends, with the activity of the individual represented by the bottom plot almost non-existent for Saturdays and Sundays. Nightly activity is also significantly less for both individuals, compared to their peak activity throughout the day. These strong daily and weekly patterns are present not only for email data, but for many types of communication data, such as mobile text messaging (Jo et al., 2012).

Suppose we are interested in defining a probability density over the time of a reply or response to an incoming event. If τ is the timestamp of an incoming event the individual received, the quantity of interest is $t > \tau$, the timestamp of the individual's reply (assuming a reply was sent). A straightforward approach is to model t as

$$t = \tau + \Delta t \qquad\qquad \Delta t \sim f^{+}$$

where $\Delta t \equiv t - \tau > 0$ is the *response time* (e.g. how long it takes for the individual to respond), and f^{+} a probability distribution over $\Delta t \in (0, \infty)$. One can then estimate the

[5] These are two individuals selected from the Gmail and Enron corpora described in Section 1.1.

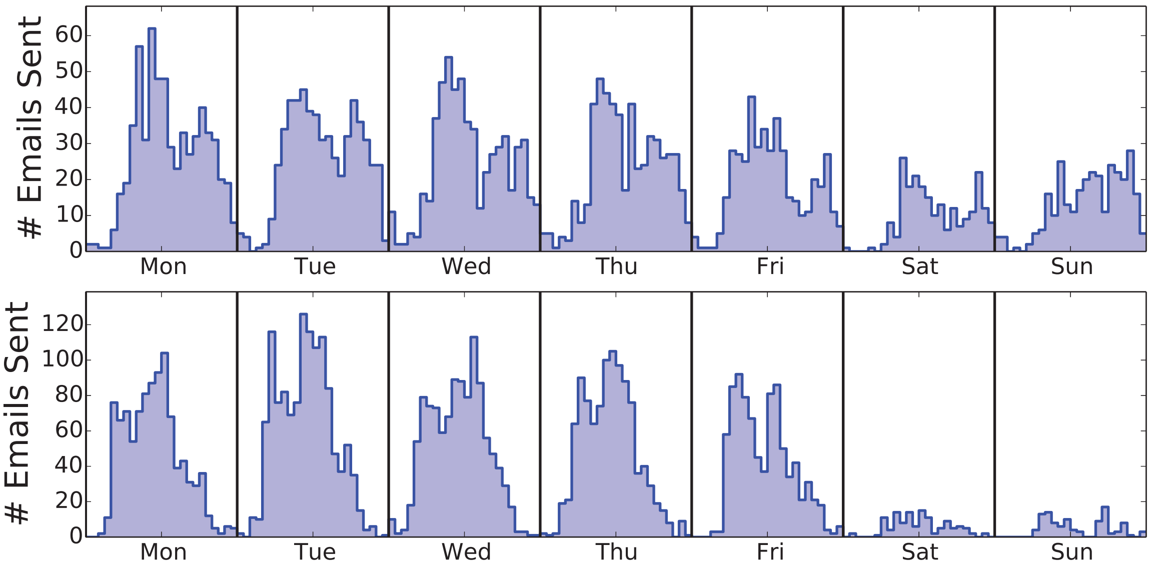


Figure 6.6: The number of emails sent as a function of time of week, for two separate individuals. Counts are aggregated throughout the individual's entire email history.

parameters of f^+ from the individual's historical data $\{\Delta t_{ij}\}$, where $\Delta t_{ij} = t_{ij} - \tau_i$ is the time it takes for the individual to send the jth response to the ith event.

The problem with this approach is that the time it takes for a person to reply can fluctuate with the time of day or week. Suppose an individual receives an incoming event during the middle of their typical sleep cycle. The response time Δt for that event is likely to be large as they may not have had a chance to become aware of the event, let alone respond to it, until they are awake hours later. In contrast, an event received in the beginning of the ego's daily routine may be responded to after a shorter amount of time. An example of this phenomenon is illustrated in Figure 6.7, where the average time to respond to an email is clearly dependent on when the email was received, for one particular individual. Unsurprisingly, the average response time is inversely related to the general activity of that individual during the time of the received email (bottom plot).

Because the quantity Δt is invariant to *when* the individual received the email, circadian rhythms are not accounted for in probability distributions over Δt. Thus, parameters that are estimated by averaging over $\{\Delta t_{ij}\}$ are likely to misrepresent the individual's typical

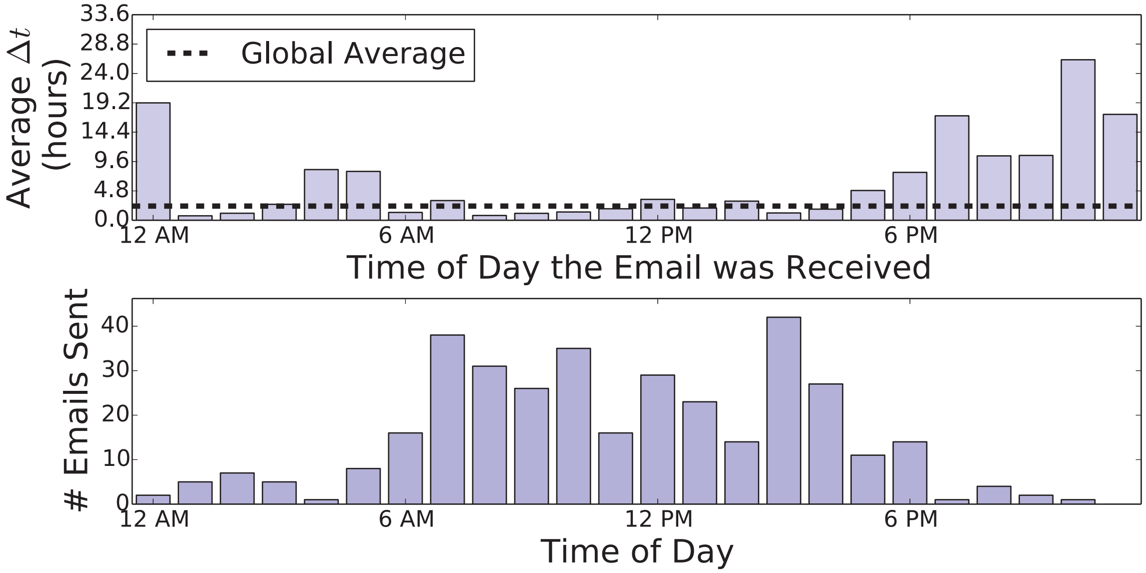


Figure 6.7: Average time it takes for a particular individual to respond to an email. Top: average response time across each hour of the day. Bottom: number of emails the individual sent each hour of the day, aggregated across their entire history.

activity (the dashed black line in Figure 6.7). Additionally, circadian rhythms are known to contribute to the large variances and heavy tails in estimated distributions over Δt (Barabási, 2005; Fox et al., 2013). Being able to effectively "remove" such patterns in a distribution over response time will both 1) allow the distribution to change as a function of time of day and week, and 2) reduce the variance in the resulting distribution.

We approach this problem by using an inhomogeneous model where the distribution over Δt depends on and changes with τ, the time of the received event. There are many ways to accomplish this, for example with a conditional density model allowing the model parameters Ω to be a function of time, or using separate models for different time intervals (e.g. every x hours or days). While such approaches are potentially useful, both have their drawbacks. The temporal dependence of parameters in the conditional density approach may be quite non-linear and difficult to capture, while the binning approach will result in partitioning the data across time intervals, which are likely to be sparse. With this in mind, the temporal transformation approach we discuss in Section 6.4.1 has the advantage of being straightforward to implement and, as will be shown in Section 6.7, quite effective.

6.4.1 Response Time as a Transformation of Effective Response Time

We address the problems regarding circadian and weekly rhythms by considering the ego's *effective response time* $\tilde{\Delta}t$ to an event relative to a non-linear time-warping mechanism (described in detail later in the section). The actual response time Δt to considered to be a transformation of $\tilde{\Delta}t$, where $\Delta t = f(\tilde{\Delta}t|\tau)$ for some function f dependent on τ. We consider probability distributions of the form $p(\tilde{\Delta}t|\Omega)$, which induces the following probability distribution over actual response time Δt (via transformation of random variables):

$$q(\Delta t|\tau,\Omega) = p\bigg(f^{-1}(\Delta t|\tau)\bigg|\Omega\bigg)\frac{\partial}{\partial\Delta t}f^{-1}(\Delta t|\tau) \tag{6.4}$$

where the distribution over effective response time $p(\tilde{\Delta}t|\Omega)$ is multiplied by the Jacobian of the inverse transformation f^{-1}. In principle we could use any transformation here[6], but a natural choice is to transform with respect to the individual's activity so that time is dilated at times of high activity and contracted at times of low activity (see Jo et al. (2012)). Thus, we define the transformation $\Delta t = f(\tilde{\Delta}t|\tau)$ such that

$$\tilde{\Delta}t = f^{-1}(\Delta t|\tau) \equiv \int_{\tau}^{\tau+\Delta t} a(u)du \tag{6.5}$$

where $a : \mathbb{R}^+ \to \mathbb{R}^+$ is a positive function, referred to as the *activity function* of the individual or ego. Plugging this inverse transformation into Equation 6.4, the distribution over Δt induced by the transformation using $\tilde{\Delta}t$ is

$$q(\Delta t|\tau,\Omega) = p\bigg(\int_{\tau}^{\tau+\Delta t} a(u)du\bigg|\Omega\bigg)a(\tau+\Delta t) \tag{6.6}$$

[6] As long as the transformation from $\tilde{\Delta}t$ to Δt is one-to-one

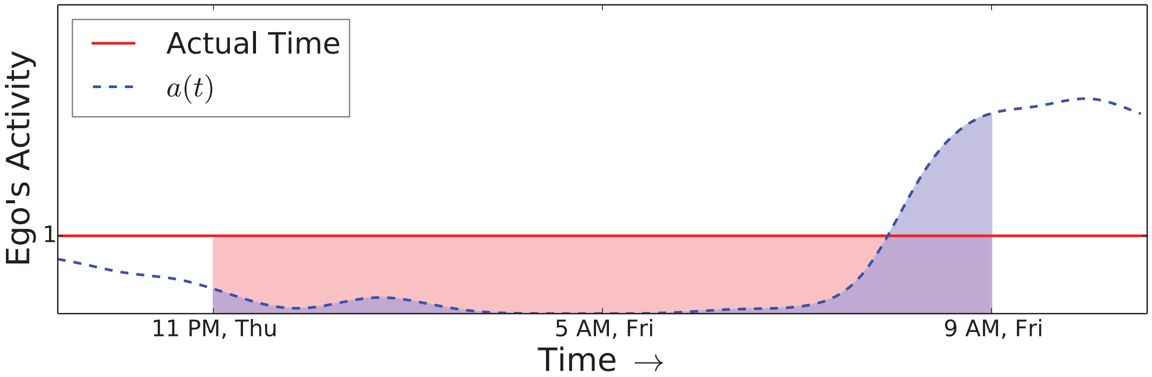


Figure 6.8: Example of an individual responding, at $t = 9$ AM Friday, to an incoming event received at $\tau = 11$ PM the previous night. The actual response time Δt (red shaded area) is 10 hours, while the effective response time $\tilde{\Delta} t$ (blue shaded area; determined by the activity function $a(t)$) is ≈ 4 hours.

When $a(u) = 1$ for all u (i.e. the individual's activity is constant over time), $\Delta t = \tilde{\Delta} t$ and $q(\Delta t|\tau, \Omega) = p(\Delta t|\Omega)$.

The activity function $a(t)$ is referred to as such because its value at any point in time can be interpreted as a relative "rate of activity" for the individual. We illustrate this interpretation with an example in Figure 6.8, where an individual receives an incoming event at $\tau = 11$ PM and responds to it at $t = 9$ AM the following morning. The actual response time would be $\Delta t = 10$ hours (the shaded area in red), clearly longer than what the individual's typical response time may be throughout the day; while it did take 10 hours to respond, the individual was not "active" throughout those 10 hours. With the activity function defined by the dashed blue line in Figure 6.8, the effective response time $\tilde{\Delta} t = \int_{11PM}^{9AM} a(t)dt \approx 4$ effective hours, much less than Δt. Thus, values of $a(t) < 1$ throughout this time interval suggests that the ego is asleep during the night, indicating that their rate of activity is lowered. Similarly, values of $a(t) > 1$ represent times where the individual is more active than average (e.g. throughout their typical work day).

As an example of what the transformed distribution $q(\Delta t|\tau, \Omega)$ defined by Equation 6.6 may look like, suppose $p(\tilde{\Delta} t|\Omega)$ is modeled a lognormal distribution with a mean time to respond

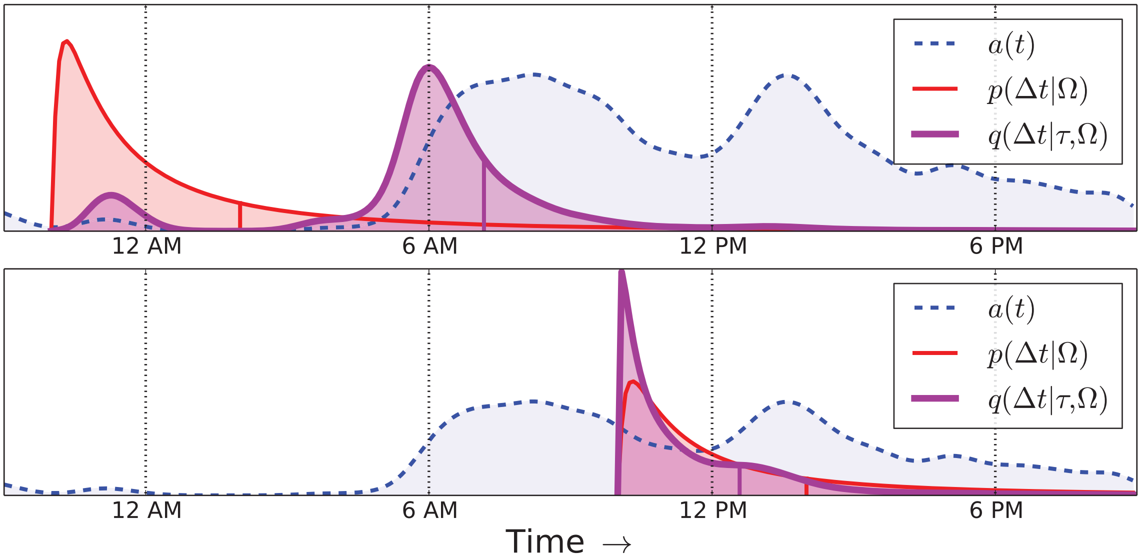


Figure 6.9: Examples of a probability distribution $q(\Delta t|\tau,\Omega)$ (in magenta) induced by a distribution over effective response time $\tilde{\Delta t}$. Unlike the distribution $p(\Delta t|\Omega)$ (in red), the shape of $q(\Delta t|\tau,\Omega)$ is dependent on τ and the activity function $a(t)$ (dashed blue line). Solid vertical lines represent the distribution means.

of 4 hours and variance 3 hours. Figure 6.9 shows what $q(\Delta t|\tau,\Omega)$ (thick magenta curve) would look like, compared to a lognormal distribution over actual response time $p(\Delta t|\Omega)$ (thin red curve), for responding to events received at $\tau = 10$ AM (top plot) and $\tau = 10$ PM (bottom plot).

In this example, we assume the activity function used to transform time according to Equation 6.5 is defined by the dashed blue line in Figure 6.9. The lognormal density $p(\Delta t|\Omega)$ is invariant to τ, thus has the same shape for both events. While this distribution may be sensible for the event received at 10 AM, it does not accurately describe the individual's behavior towards the event received at 10 PM. In this case, much of the probability mass is "wasted" in the late hours of the evening, where the individual's activity is low.

In contrast, the shape of the distribution $q(\Delta t|\tau,\Omega)$ in Figure 6.9 significantly changes between the two events. For the incoming event at $\tau = 10$ PM, most of the probability mass is shifted away from the night and to the early hours of the morning where the ego becomes

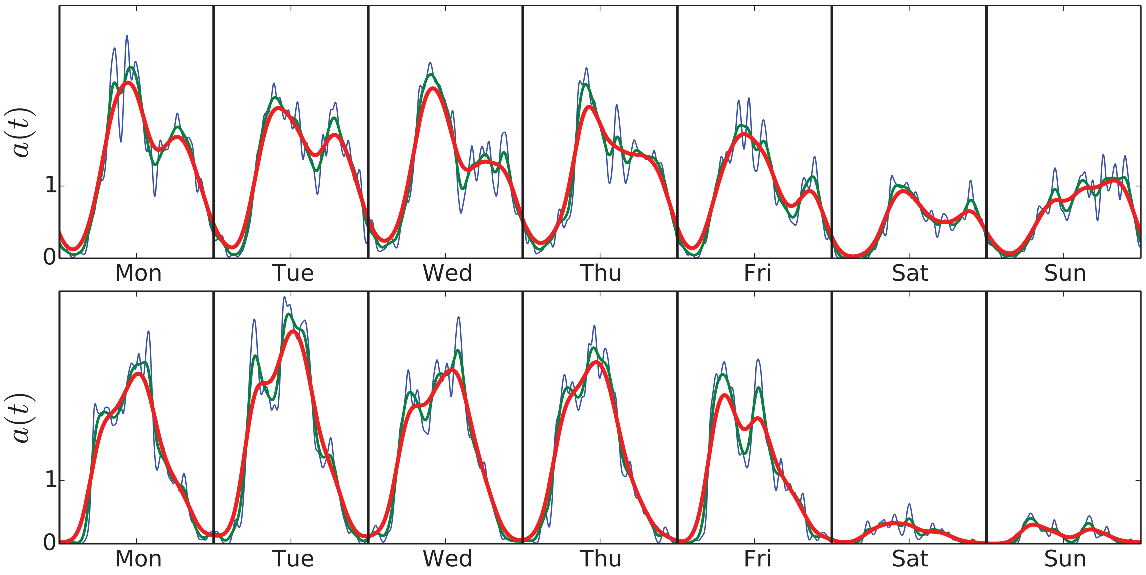


Figure 6.10: Estimated activity functions over the course of a week for two individuals (the same as in Figure 6.6). Different colored lines correspond to different smoothing parameters in the estimation of $a(t)$, discussed in detail in Section 6.6.

active; the mean time of a response is around 7 AM (9 hours later). This transformed distribution more accurately describes the individual's likelihood of responding in the early working hours, where they are likely to first notice the event, as opposed to the middle of the night. For the event received at $\tau = 10$ AM, the increased activity function (indicating the ego is active) causes $q(\Delta t|\tau, \Omega)$ to be more peaked than $p(\Delta t|\Omega)$, thus the event is likely to be responded to more quickly. In this case, the mean time of a response is around 12:30 PM (2.5 hours later). As these examples showed, the specification of the activity function $a(t)$ allows the distribution $q(\Delta t|\tau, \Omega)$ over response time to "warp" significantly in shape as a function of τ, minimizing the effects of the individual's circadian and weekly rhythms.

Figure 6.10 shows examples of what an activity function may look like over the course of a week for two different individuals' email behavior, where the estimated activity function $a(t)$ closely follows the histograms of observed activity shown in Figure 6.6. Clear circadian and weekly patterns are shown for both individuals. The estimation procedure of $a(t)$ from an individual's historical activity is described in detail in Section 6.6.

We conclude this section by noting that our overall approach in modeling response times can be viewed from two equivalent perspectives. The first is a "transformed time" approach where we first transform the time dimension with respect to $a(t)$, then use a simple distributional form (such as a lognormal) to model effective response times $p(\tilde{\Delta t}|\Omega)$. Alternatively, this approach can be viewed as a relatively complex model over actual response time, $q(\Delta t|\tau, \Omega)$ (as in Figure 6.9), whose shape is induced by the simpler model over effective time $p(\tilde{\Delta t}|\Omega)$ and the activity function $a(t)$.

6.5 Models over Effective Response Times

In this section, we apply the transformation of time using effective response times to two different models of response behavior, illustrating the beneficial effect of including the activity function $a(t)$ into each model. First, a direct model over the number of responses to an event and their respective response times is explored. The second model takes a stochastic process approach, modeling the timestamps of responses $t > \tau$ (and, implicitly, the number of responses) via time-varying response rates.

Here, we assume the form of the activity function has previously been estimated and is conditioned on (i.e. $a(t) \in \Omega$). We explore its estimation in Section 6.6.

6.5.1 Direct Modeling of Response Behavior

We first show how to use the transformation of time to directly model the individual's response time Δt to an event. Let e_i be the ith incoming event that the individual receives, occurring at time τ_i. Additionally, let Ω be the set of model parameters. The timestamps of the individual's N_i responses $\{t_{ij} : 1 \leq j \leq N_i\}$ to that event are assumed to be generated as follows:

1. Draw the number of responses $N_i \sim d(N_i|e_i, \Omega)$, where $d(N_i; e_i, \Omega)$ is the discrete distribution defined by Equation 6.1 from Section 6.2.
2. For each of the N_i responses, independently draw the jth response time $\Delta t_{ij} \sim q(\Delta t_{ij}|\tau_i, \Omega)$, where $q(\Delta t_{ij}|\tau_i, \Omega)$ is defined as the transformed distribution in Equation 6.6.

Any distribution over the effective response time defined by Equation 6.5 can be used to create the transformed distribution $q(\Delta t_{ij}|\tau_i, \Omega)$. Common examples in the context of email communication include the exponential (Fox et al., 2013; Malmgren et al., 2008), Gamma (Halpin and De Boeck, 2013), and lognormal (Stouffer et al., 2006) distributions; an example using the lognormal distribution was shown in Figure 6.9. A potentially useful extension to this model would be to allow for dependencies between the N_i response times, when $N_i > 1$; the independence assumption is assumed for simplicity.

The log-likelihood of the direct model over response behavior is

$$\log P(\{t_{ij}\}|\{e_i\}, \Omega) = \sum_{i=1}^{N} \log d(N_i|e_i, \Omega) + \sum_{i=1}^{N} \sum_{j=1}^{N_i} \log q(\Delta t_{ij}|\tau_i, \Omega) \tag{6.7}$$

The first term is independent of the modeling choice of $q(\Delta t_{ij}|\tau_i, \Omega)$, and can be independently optimized to infer the respective model parameters (as discussed in Section 6.2). Here, we focus on the second term: the log-likelihood of the response times.

Using the transformation via effective response time $\tilde{\Delta} t_{ij}$ and the form of $q(\Delta t_{ij}|\tau_i, \Omega)$ defined by Equations 6.5 and 6.6 respectively, the log-likelihood is proportional to

$$\log P(\{t_{ij}\}|\{e_i\}, \Omega) \propto \sum_{i=1}^{N} \sum_{j=1}^{N_i} \log p(\tilde{\Delta} t_{ij}|\Omega) + \sum_{i=1}^{N} \sum_{j=1}^{N_i} \log a(\tau_i + \Delta t_{ij})$$

The second term is constant with respect to model parameters specific to the distributional form of p, thus estimates for these parameters can be obtained by maximizing the first expression over effective response times $\{\tilde{\Delta} t_{ij}\}$. For example, suppose an exponential distribution is placed over effective response times, where $\lambda \subset \Omega$ is the exponential mean (in units of days) of the individual's effective response time to an event. Using the maximum-likelihood approach, the value of λ would be estimated to be the reciprocal of the empirical mean in the effective response times $\{\tilde{\Delta} t_{ij}\}$. Similar maximum-likelihood estimates can be straightforwardly derived for the Gamma and lognormal distributions.

The process of applying the time transformation to standard probability distributions of response time has a strong connection to survival modeling. In survival models, the response time Δt is the quantity of interest, and is modeled with a survival function

$$S(x) \equiv P(\Delta t > x) = \exp\left(-\int_0^x h(u)du\right)$$

where $h(u)$ is referred to as a *hazard function* (Aalen et al., 2008, Section 1.1.2). One way to model a hazard function is to have it be proportional to the activity function, $h(u) = \lambda a(u)$ for $\lambda > 0$. The activity function is a natural choice of hazard function as it can be interpreted as a rate of events occurring over time. When the hazard function is modeled in such as way, then the probabilities of the survival model become equivalent to that of an exponential distribution with parameter λ over the transformed time dimension. This can be shown using the cumulative density function $Q(\Delta t|\tau, \lambda)$ of an exponential distribution transformed according to Equation 6.6 (assuming the email was received at time $\tau = 0$):

$$q(\Delta t > x|\tau = 0, \lambda) = 1 - Q(x|\tau = 0, \lambda) = \exp\left(-\lambda \int_0^x a(u)du\right) = S(x)$$

6.5.2 Stochastic Process Models of Response Behavior

We next show how to use an individual's activity function to account for circadian and weekly patterns within a stochastic process model. In particular, we focus on Hawkes processes, commonly used to describe the "bursty" episodes of active behavior followed by long periods of inactivity (Hawkes, 1971). Such processes have successfully been used to model email communication patterns (Fox et al., 2013; Halpin and De Boeck, 2013; Simma and Jordan, 2010). Under this model, the timestamps of the individual's responses $\{t_{ij}\}$ are assumed to be generated from a global *response rate* $\lambda(t)$ of the following form:

$$\lambda(t) \equiv \lambda_0(t) + \sum_{i:\tau_i<t} g(t|\tau_i) \tag{6.8}$$

where $\lambda_0(t)$ is referred to as the *base rate* at which the individual initiates events (e.g. sending an email that is not a reply), and $g(t|\tau_i)$ are the *triggering functions* describing the rate at which the individual responds to event i. The form of $g(t|\tau_i)$ is usually

$$g(t|\tau_i) \equiv \nu p(\Delta t|\Omega)$$

where $p(\Delta t|\Omega)$ is a standard distribution (e.g. exponential) over response time $t - \tau_i$ and $\nu \in \Omega$ is interpreted as the expected number of replies to a single event (Fox et al., 2013). The response rate $\lambda(t)$ is thus the rate at which the individual creates an event, whether or not it was a response. An example of what $\lambda(t)$ may look like, assuming $\lambda_0(t) = \lambda_0$ is constant and using a Gamma triggering function, is illustrated in Figure 6.11.

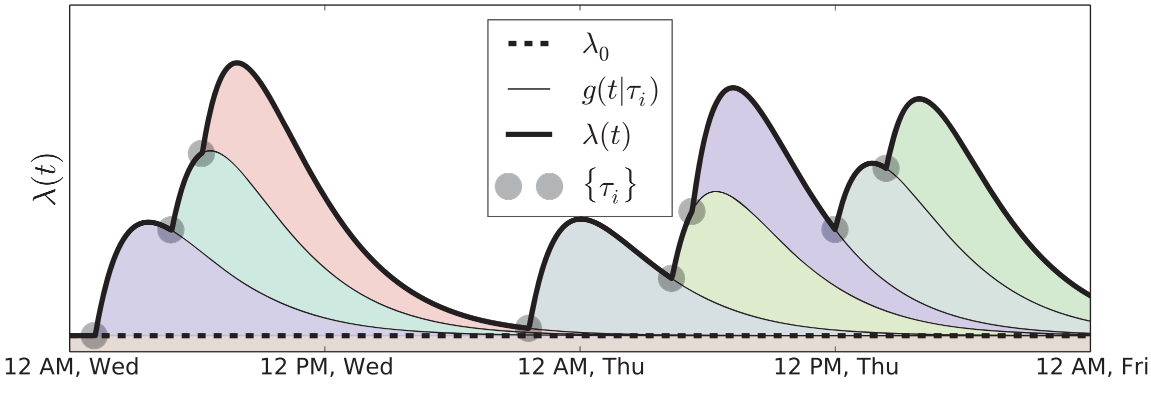


Figure 6.11: Example intensity function $\lambda(t)$ for a Hawkes process.

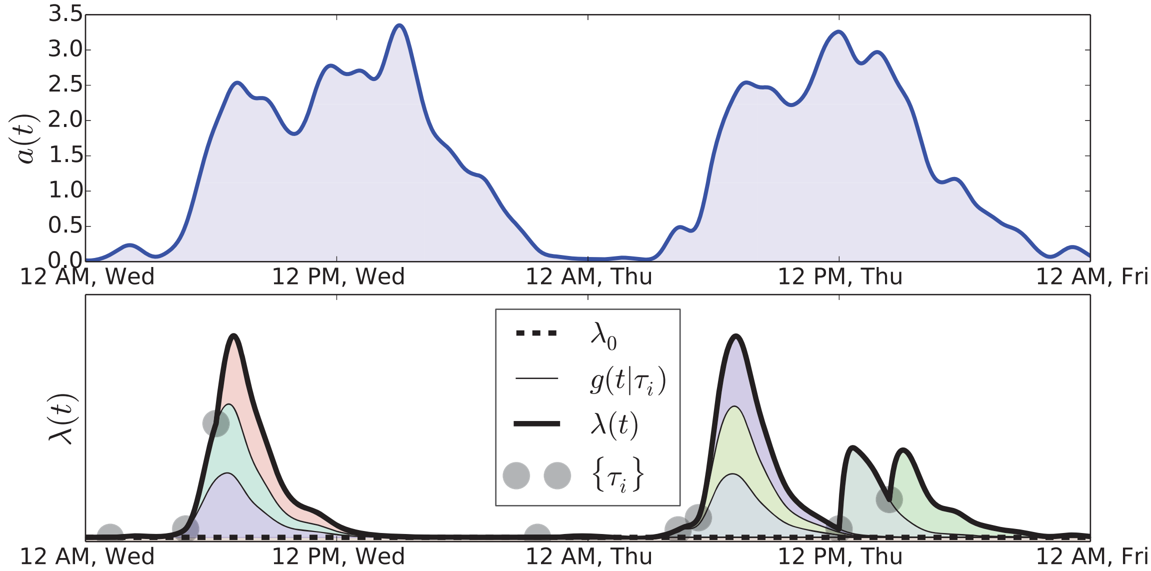


Figure 6.12: The same Hawkes process in Figure 6.11, but using triggering functions transformed by effective response times. The shape of the overall response rate $\lambda(t)$ (bottom plot) becomes dependent on the activity function (top plot).

The Hawkes process is useful in that the individual's response rate is a function of $\{\tau_i\}$ — a response is more likely to occur as additional incoming events are received. However, the triggering function $g(t|\tau_i)$ is a function of actual response time Δt, and is prone to the same problems posed by circadian and weekly patterns when modeling $p(\Delta t|\Omega)$. We propose to instead model the rate at which the individual responds to event i using the transformed time domain via activity function $a(t)$, where

$$g(t|\tau_i) \equiv \nu q(\Delta t|\tau_i, \Omega)$$

with $q(\Delta t|\tau_i, \Omega)$ defined by Equation 6.6. An example of what the response rate $\lambda(t)$ may look like under such a transformation is shown in Figure 6.12. As the triggering functions are now dependent on the timestamps of incoming events τ and the activity function $a(t)$, their shape varies based on the time of day or week. The resulting model is now one that can accurately describe the bursty response behavior of an individual with respect to their circadian and weekly rhythms.

The log-likelihood of response times according to the Hawkes process is

$$\log P(\{t_{ij}\}|\{e_i\}, \Omega) = \sum_{i=1}^{N} \sum_{j=1}^{N_i} \log \lambda(t_{ij}) - \int_0^T \lambda(u) du$$

where it is assumed that the observed dataset is over the time interval $[0, T]$ (Daley and Vere-Jones, 2003).

The log-likelihood contains a log function over summations of terms (with $\lambda(t)$ defined by Equation 6.8), which can make parameter inference intractable if the response structure (e.g. which incoming event the individual replies to) is unknown, thus requiring the marginalization of response rates over all incoming events to produce $\lambda(t)$ (Halpin and De Boeck, 2013; Hawkes and Oakes, 1974; Olson and Carley, 2013). However, in the case where the response structure is known (as is the case for the email datasets considered in Section 6.3.1), the response rate $\lambda(t)$ reduces to

$$\lambda(t) = \begin{cases} \nu g(t|\tau_i) & \text{if the individual replied to event } i \\ \lambda_0(t) & \text{if the individual initiated the event} \end{cases} \tag{6.9}$$

resulting in the following log-likelihood over response times:

$$\log P(\{t_{ij}\}|\{e_i\}, \Omega) = c + \sum_{i=1}^{N} \left(\sum_{j=1}^{N_i} \log(\nu q(\Delta t_{ij}|\tau_i, \Omega)) - \nu \int_{\tau_i}^{T} q(u|\tau_i, \Omega) du \right) \tag{6.10}$$

where c contains the summation of terms related to $\lambda_0(t)$, the rate at which non-response events initiated by the individual are produced.

Maximum-likelihood estimates of model parameters can be numerically calculated (no closed form exists due to the integral term of the log-likelihood) efficiently when using the parameterization of $\lambda(t)$ defined by Equation 6.9. When the response structure is unknown, it can be estimated jointly with the model parameters using an approach such as the EM algorithm (Halpin and De Boeck, 2013; Olson and Carley, 2013).

The modeling choice for $\lambda_0(t)$ is independent of the modeling of response times, as it describes the rate at which the individual initiates new events. As the activity function $a(t)$ from Section 6.4 can be interpreted as a relative activity rate of the individual, an appropriate modeling choice is $\lambda_0(t) = \rho \times a(t)$, where $\rho > 0$ is a scaling parameter that can be learned via maximum-likelihood.

6.6 Estimation of the Activity Function

In this section, a method for estimating the activity function $a(t)$ from historical data is explored, producing results such as those found in Figure 6.10. Recall that an estimate of $a(t)$ is required for the proposed time transformation approach described in Section 6.4.1. The proposed method can be applied in either a batch or online setting, allowing for efficient updates to the estimation of $a(t)$ in the presence of new activity from the individual.

6.6.1 Nonparametric Estimation via Weekly Timestamps

Suppose an individual's historical data consists of K events that they have created, regardless of whether or not they were responses to previous events. Let $\boldsymbol{t} = \{t_k : 1 \leq k \leq K\}$ represent the set of timestamps of such events (in units of days); under this notation, the set of timstamps for all the individual's responses $\{t_{ij}\} \subseteq \boldsymbol{t}$. In contrast to previous sections, we use the single subscript k in this section to describe a timestamp in $\boldsymbol{t}$. The data is assumed to be sorted such that, for $i < j$, $t_i < t_j$.

We define $\Delta_w(t_a, t_b) \in [0, 3.5]$ days to be the minimal *weekly time difference* between two timestamps t_a and t_b:

$$\Delta_w(t_a, t_b) \equiv \min\Big((|t_a - t_b| \bmod 7), 7 - (|t_a - t_b| \bmod 7)\Big) \tag{6.11}$$

Given the historical timestamps $\boldsymbol{t}$, the activity function $a(t)$ is estimated non-parametrically and has the following form:

$$a(t) \equiv \frac{1}{Z}\sum_{k=1}^{K} \exp\left(-\frac{\Delta_w(t, t_k)^2}{2h^2}\right) \tag{6.12}$$

where h is the *bandwidth parameter*, and Z a normalization constant such that $\int_t^{t+7} a(u)du = 7$. Note that, by construction of $\Delta_w(t_a, t_b)$, the estimated $a(t)$ has a periodicity of 7 days.

The structure of this estimate is very similar to that of kernel density estimation, but has some differences semantically. First, the domain of the estimation is over the weekly interval of timestamps, not the absolute value of timestamps themselves. Second, $a(t)$ is a function, not a probability distribution. Indeed, under the right normalization $\left(\int_t^{t+7} a(u)du = 1\right)$ the activity function can be interpreted as a probability distribution over the time of the week the individual will respond to an incoming event. However, the normalization we chose allows for the following interpretation of $a(t)$: if the actual response time to an event $\Delta t =$ 7 days, then the effective response time $\tilde{\Delta} t$ (defined by Equation 6.5) is also 7 days. Thus, the estimated activity function is modeling the fluctuation in the individual's activity via circadian and weekly patterns on the scale of hours and days, not weeks.

Figure 6.13 shows what the estimation of the activity function via Equation 6.12 looks like for a single individual's data. As expected, the non-parametric estimation captures the daily and weekly patterns of the individual (shown as the histogram in the figure). The figure also shows that the bandwidth parameter h has the same role to that in kernel density estimation:

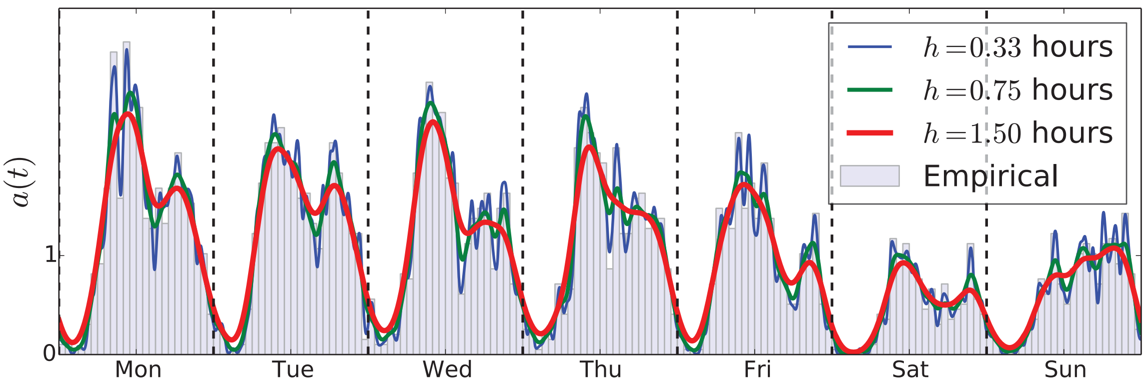


Figure 6.13: Non-parametric estimation of the activity function using Equation 6.12, applied to a single individual's email behavior.

the larger h is, the smoother the estimation of $a(t)$. The selection of the bandwidth parameter is discussed in Section 6.6.4. The estimation via Equation 6.12 was also used to generate the activity functions shown in Figure 6.10.

Calculating the Normalization Constant

In order to calculate the normalization constant Z, we must first look at the area under the curve for a single summand term in Equation 6.12 over the interval $[t, t+7]$. Using the form of $\Delta_w(t, t_k)$ in Equation 6.11, this area is equal to

$$\int_t^{t+7} \exp\left(-\frac{\Delta_w(u, t_k)^2}{2h^2}\right) du = 2\int_0^{3.5} \exp\left(-\frac{u^2}{2h^2}\right) du \approx \sqrt{2\pi h^2}$$

where the approximation becomes exact as $h \downarrow 0$. An illustration of this area under curve is shown in Figure 6.14. Thus, for a reasonable[7] bandwidth parameter h, the normalization constant Z can be calculated in closed form:

$$Z = \frac{K\sqrt{2\pi h^2}}{7}$$

[7] For the overly-smooth value of $h = 12$ hours, the approximation in the area under the curve is $<$ 1e-10

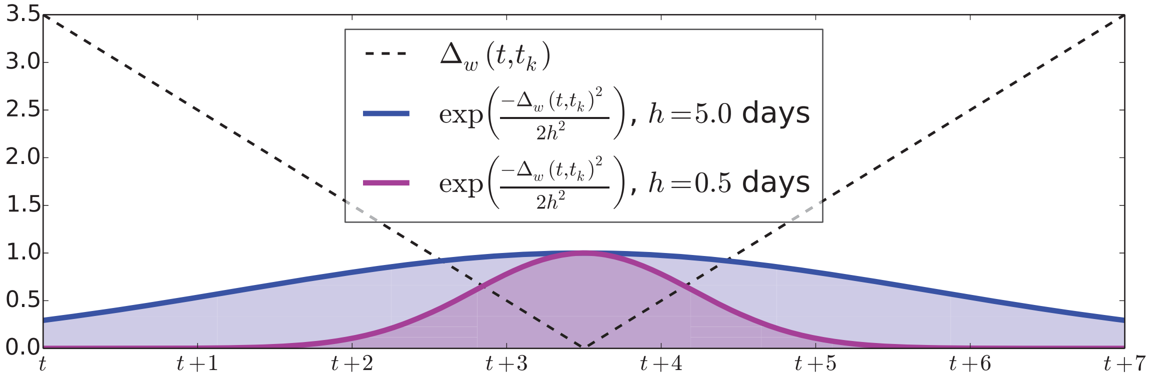


Figure 6.14: Area under the curve for a single summand term in Equation 6.12 over the interval $[t, t+7]$. The curve is centered such that the peak occurs at time $t + 3.5$.

Online Updates to the Estimation of the Activity Function

The non-parametric nature of the estimation of $a(t)$ allows it to be easily updated when new data is observed. Let $a^{(K)}(t)$ and $Z^{(K)}$ represent the estimated activity function and normalization constant over historical data $\{t_k : k \leq K\}$ respectively. Suppose the individual then produces a new event with timestamp t_{K+1}; a new estimate of $a(t)$ is created by appending the sum in the estimation to include t_{K+1} and updating the normalization constant:

$$a^{(K+1)}(t) \equiv \frac{1}{Z^{(K+1)}} \sum_{k=1}^{K+1} \exp\left(-\frac{\Delta_w(t, t_k)^2}{2h^2}\right)$$

Efficient Implementation of the Activity Function

A naive implementation of the estimated activity function in Equation 6.12 requires $O(K)$ space to store, with constant-time updates. More importantly, evaluations of $a(t)$ and $\int a(t)dt$ both take $O(K)$ time to calculate. This can be problematic when queries requiring the calculation of effective response times (e.g. probabilities involving the models discussed in Section 6.5) must be quickly answered.

The complexities of such queries can be reduced to constant-time at the cost of minimal approximation. This is done by discretizing the activity function into an array of size S, so that the sth element represents the value of $a(t)$ for any timestamp t such that $t \bmod 7 = \frac{7s}{S}$. In our work here we selected S = 604,800, the number of seconds in a week. Thus, the approximation induced by the discretization is minimal — smaller than the resolution of the timestamps (one second for email data). When calculating integrals of $a(t)$ on the order of hours or days, this approximation becomes negligible. Discretization of the activity function also has the benefit of requiring constant-time storage (albeit storing S elements).

Values of the elements in the discretized array are straightforward to update in the presence of new data by making use of the following recursive properties:

$$\begin{aligned} a^{(K+1)}(t) &= \frac{1}{Z^{(K+1)}} \left(Z^{(K)} a^{(K)}(t) + \exp\left(-\frac{\Delta_w(t, t_{K+1})^2}{2h^2} \right) \right) \\ Z^{(K+1)} &= Z^{(K)} + \frac{\sqrt{2\pi h^2}}{7} \end{aligned}$$

A minor disadvantage of discretizing the activity function is that an updates take $O(S)$ time, as each element in the array is updated. While this is theoretically a constant-time update with respect to the number of datapoints K, it is more expensive then simply appending the datum to the historical data previously observed.

6.6.2 Decreasing the Influence of Past Datapoints

A potential issue of the non-parametric estimation for the activity function (via Equation 6.12) is that each datapoint is given an equal amount of "influence." Suppose an individual suddenly experiences a significant change in their circadian and weekly patterns. For example, the individual may move to a location in a different timezone, or their weekly work schedule may change. When this happens, the estimation will average the two different pat-

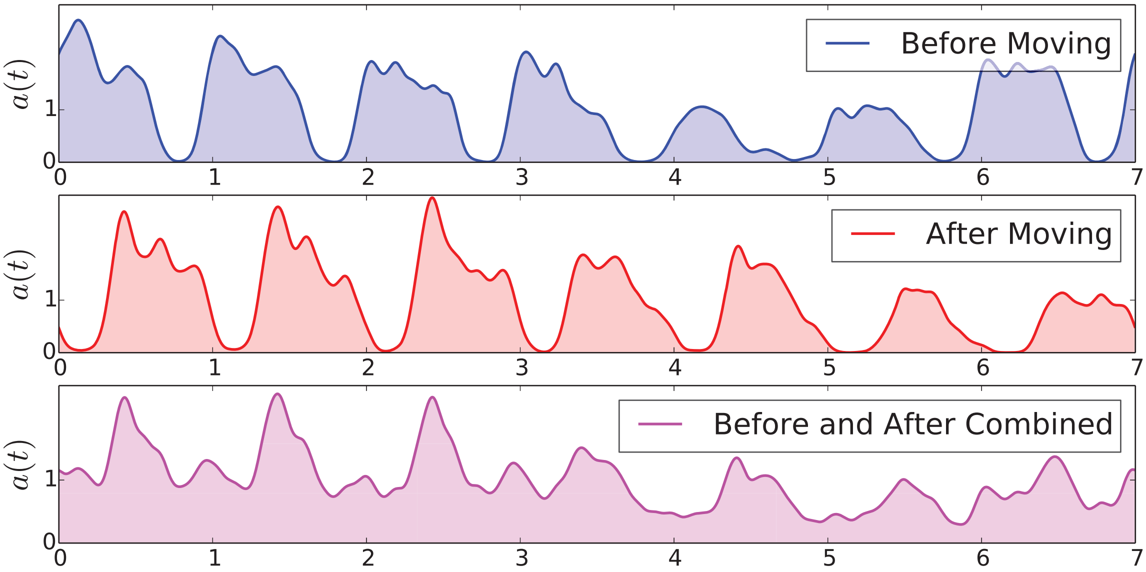


Figure 6.15: Example of estimated $a(t)$ for an individual that experienced a significant change of behavior due to moving between countries. Top: estimated $a(t)$ at the first location. Middle: estimated $a(t)$ at the second location. Bottom: estimated $a(t)$ considering all data.

terns together, resulting in an activity function that is non-representative of the individual's typical behavior.

An example of this phenomenon is shown in Figure 6.15, where an individual experienced a significant change in response behavior when they moved to a different country[8]. Two major changes occurred with this move: the timezone shifted by 10 hours, and their non-work days moved from Friday/Saturday to Saturday/Sunday. Both of these changes are seen when comparing the top and middle plots in Figure 6.15; these are estimates of the activity function using only datapoints from one of the two countries. The estimation of $a(t)$ in the bottom plot is the estimation across the entire time interval of the dataset, resulting in a clearly inaccurate representation of their daily and weekly activity.

A solution to this issue is to allow the weight or "influence" of a datapoint to decrease over time as newer datapoints are observed. Allowing the estimation method to "forget" older datapoints gives it the flexibility to change as incoming datapoints do. This can be

[8] As explained by the individual (they are part of the "Gmail" email corpus, described in Section 1.1).

accomplished by weighting each datapoint in Equation 6.12:

$$a(t) \equiv \frac{1}{Z} \sum_{k=1}^{K} w_k \exp\left(-\frac{\Delta_w(t, t_k)^2}{2h^2}\right) \tag{6.13}$$

$$w_k \equiv \exp\left(-\alpha(t_K - t_k)\right) \tag{6.14}$$

where $\alpha \geq 0$ is referred to as a *decay* parameter, controlling the rate at which datapoints lose influence. Under this formulation, the most recent datapoint is always given a weight of $w_K = 1$, and the time it takes for a datapoint to lose half of its influence (referred to as its "half-life") is $\frac{\log(2)}{\alpha}$ days. This idea of down-weighting datapoints over time is similar to that found in kernel density estimation over time series data (Grillenzoni, 2006), albeit with a different structure of weights and interpretation.

The normalization constant Z for Equation 6.13 can be calculated in closed-form:

$$Z = \frac{W\sqrt{2\pi h^2}}{7} \qquad\qquad W = \sum_{k=1}^{K} w_k$$

where the quantity W is referred to as the effective number of datapoints represented by the kernel estimation. In the case where $\alpha = 0$, we have $w_k = 1$ for all k, $W = K$, and Equation 6.13 equivalent to Equation 6.12.

Figure 6.16 shows what the estimated activity function looks like over time for the individual described in Figure 6.15. The left image shows, for each week over time, a smoothed histogram over the individual's email activity for that week. It is apparent that the individual's move between countries occurred in mid-2012. The bottom of the middle image shows the averaging effect between the two different behavior patterns when $\alpha = 0$ (i.e. estimating the activity function via Equation 6.12). In contrast, the right image shows the estimated activity over time when the half-life of influence weights is set to 60 days. The estima-

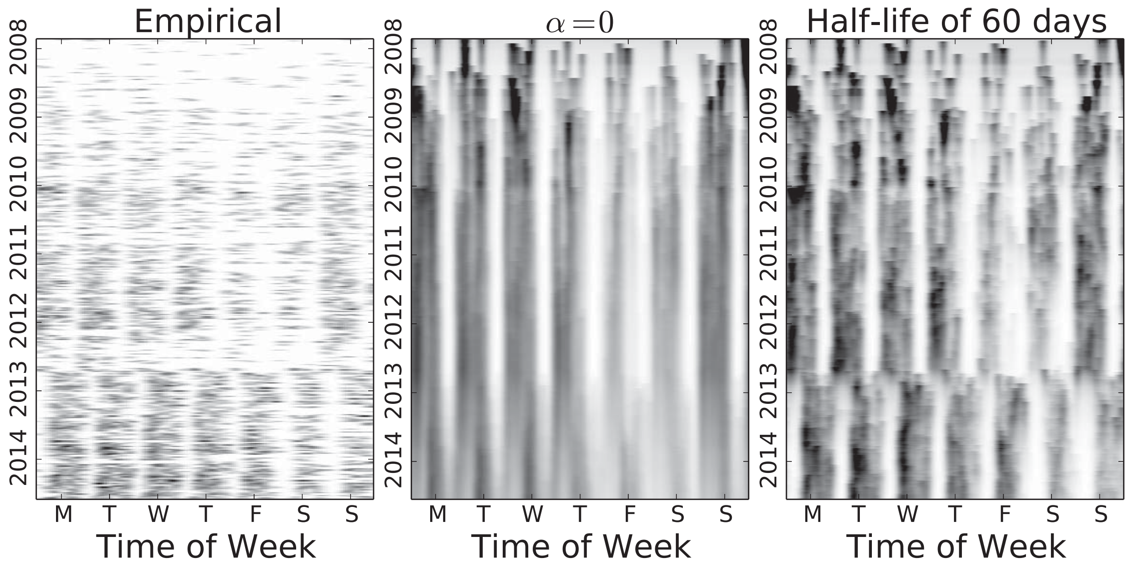


Figure 6.16: Example of the advantages in setting $\alpha > 0$. Left: smoothed histogram counts over number of emails sent throughout each week. Middle: estimated activity function over time with $\alpha = 0$. Right: estimated activity function over time with $\alpha = \frac{\log(2)}{60}$.

tion procedure is able to quickly adapt to the change in behavior, resulting in an accurate estimation of the activity function throughout the entire history of the individual.

While the introduction of the α parameter is beneficial for adapting to behavioral changes, a degree of inefficiency is added in cases where the behavioral patterns remain stationary. In such cases, noisier estimates of $a(t)$ are produced as only the most recent historical datapoints hold significant weight. This problem is amplified in the presence of little historical data. A heuristic approach to setting α using a validation dataset is discussed in Section 6.6.4 that helps to mitigate this problem.

Online Updates to the Estimation of the Activity Function

As with the estimation of $a(t)$ in Section 6.6.1, updates in the estimation of $a(t)$ via Equation 6.13 in an online fashion can be efficiently made using a discretized array and making use of the following recursive properties:

$$\begin{aligned}
w_k^{(K+1)} &= \exp\left(-\alpha(t_{K+1} - t_k)\right) = \exp\left(-\alpha(t_{K+1} - t_K)\right) \exp\left(-\alpha(t_K - t_k)\right) \\
&= \exp\left(-\alpha(t_{K+1} - t_K)\right) w_i^{(K)} \\
Z^{(K+1)} &= \exp\left(-\alpha(t_{K+1} - t_K)\right) Z^{(K)} + \frac{\sqrt{2\pi h^2}}{7} \\
a^{(K+1)}(t) &= \frac{1}{Z^{(K+1)}} \left(\exp\left(-\alpha(t_{K+1} - t_K)\right) Z^{(K)} a^{(K)}(t) + \exp\left(-\frac{\Delta_w(t, t_{K+1})^2}{2h^2} \right) \right)
\end{aligned}$$

where the superscript K indicates quantities calculated with respect to $\{t_k : 1 \leq k \leq K\}$.

6.6.3 Regularization of the Activity Function

Another issue in estimating the activity function occurs when the value of W (the effective number of datapoints represented by kernel estimate) is small. Similar to the effect of overfitting, the estimated function $a(t)$ can be extremely peaked around the historical datapoints when there are only a few to estimate the function from. For this section, we will assume that $\alpha = 0$ (where $W = K$), although similar patterns emerge for $\alpha > 0$.

An example of where this is problematic is shown in the blue solid curve of Figure 6.17, where $a(t)$ is estimated from only three datapoints. Suppose we are interested in calculating the effective response time $\tilde{\Delta t}$ to an event received at $\tau = 12$ PM on Friday, which was responded to at $t = 12$ PM on Saturday. Taking the integral of $a(t)$ between these two points in time (represented by the vertical lines), $\tilde{\Delta t} \approx 0$ seconds — an instantaneous response from the individual. Not only is this non-representative of the individual's behavior, but it will introduce biases in the estimated values of model parameters over effective response times.

One way to mitigate this problem is to have the estimated activity function $a(t)$ be regularized by taking a convex combination between the estimation via Equation 6.13 and a

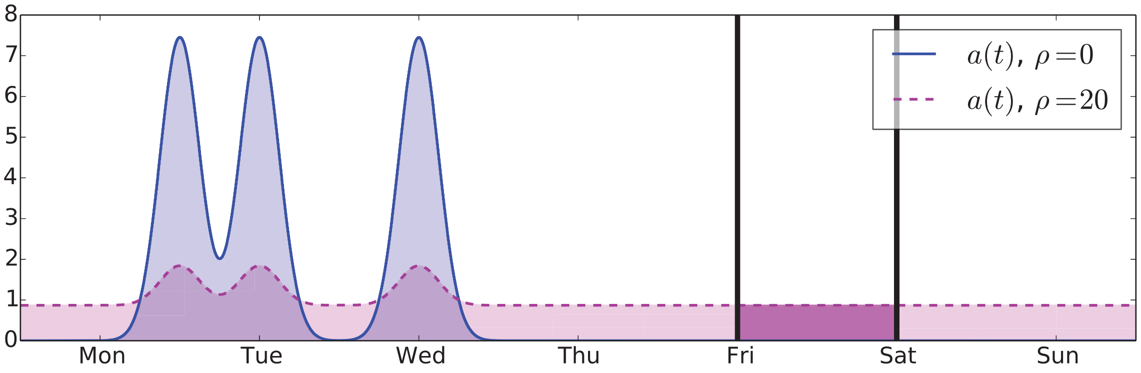


Figure 6.17: Example of when regularizing the estimate of $a(t)$ is necessary, as per Equation 6.15. The 24-hour interval between the vertical black lines has an effective time of 0 seconds when $\rho = 0$. When regularizing with $\rho = 20$ pseudo-datapoints, the effective time over the same interval becomes ≈ 0.85 days.

"uniform" function where $a(t) = 1$:

$$a(t) \equiv \left(\frac{W}{W+\rho}\right)\left[\frac{1}{Z}\sum_{k=1}^{K} w_k \exp\left(-\frac{\Delta_w(t,t_k)^2}{2h^2}\right)\right] + \left(\frac{\rho}{W+\rho}\right) \tag{6.15}$$

where $\rho \geq 0$ is a *smoothing parameter*. The introduction of the smoothing parameter does not affect the calculation of the normalization constant Z, and the properties of $a(t)$ (e.g. the periodicity and integration to 7 over one-week intervals) are preserved.

The strength of this regularization towards the uniform function is data-driven — when W is large, the effect of the regularization is negligible. When $W \approx 0$, the estimated $a(t)$ becomes very close to the uniform function. The relative strength of the regularization depends on the smoothing parameter ρ, which can be interpreted as the number of "pseudo-datapoints" associated with the uniform function.

Considering the problem of calculating effective response time in Figure 6.17, the magenta dashed curve shows the estimation of $a(t)$ when $W = 3$ and $\rho = 20$. Because $\rho > W$, the estimated form of $a(t)$ via Equation 6.15 becomes very close to the uniform function. Thus, the effective time passed between 12 PM Friday and 12 PM Saturday becomes approximately 0.85 days, instead of the instantaneous response time as calculated when $\rho = 0$.

6.6.4 Selection of Estimation Hyperparameters

In this section, we describe how to set the values of the hyperparameters h, α, and ρ given a set of validation data (described in the experimental section). We assume that the validation data is in the form $\boldsymbol{t} = \{t_k : 1 \leq k \leq K\}$, where K is the number of validation events given.

First, we fix the value of the smoothing parameter ρ to be equal to 10 pseudo-datapoints. Preliminary experimentation showed that the results were broadly similar for other values of ρ ranging from 5 to 30. Setting $\rho = 0$ causes numerical issues as described by Figure 6.17 and setting ρ too large forces the regularization to dominate the observed data.

Next, the bandwidth parameter h is determined by performing 10-fold cross-validation over the validation data, selecting the value of h that maximizes

$$\sum_{k=1}^{K} \log \hat{a}_k(t_k) + \log p(h)$$

where $\hat{a}_k$ is the estimated activity function via Equation 6.12 using datapoints from the 9 folds that datapoint t_k is not in, and p a prior density over the values of h; a Gamma with a mean of 4 hours is used as a prior (shown in the left plot of Figure 6.18). Optimization is done using gradient ascent, much like the optimization of the bandwidth parameter for binary kernels discussed in Chapter 3. For the estimation of h, we set $\alpha = 0$ and $\rho = 10$.

Lastly, we heuristically set α based on properties of the validation data. Estimating α by cross-validation would lead to unstable estimates, as a substantial amount of data is needed in order to set the optimal value of α with respect to the individual's long-term behavioral changes. Let $\tilde{n}$ represent the average number of events the individual creates a day, estimated from the validation data. The value of α is set heuristically such that, in the presence of observed data over an infinitely long window of time,

$$\tilde{W} = \sum_{k=1}^{K} w_K \approx \tilde{n} \sum_{d=0}^{\infty} \exp\left(-\alpha d\right) = \frac{\tilde{n}}{1 - \exp\left(-\alpha\right)}$$

Thus, on average the effective number of datapoints in the estimation of $a(t)$ is $\tilde{W}$. For the upcoming experiments, we set $\tilde{W} = 150$; similar results were found for $\tilde{W} = 200$ and 250.

For an individual that creates many events a day, the value of α can easily be overestimated, resulting in an extremely short half-life of influence weights amongst historical datapoints. We mitigate this problem by setting the maximum value of α to be $\frac{\log(2)}{35}$, where the weight of a datapoint is halved after 35 days.

Figure 6.18 shows the estimated values of h and α using the process described above, along with the prior distribution over h, across all datasets used in the experiments (described in Section 6.3.1).

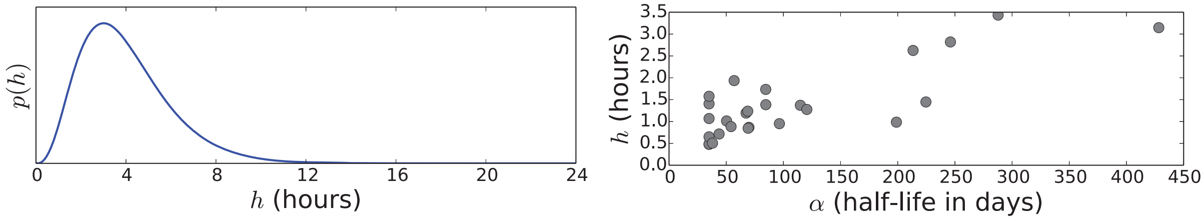


Figure 6.18: Estimated hyperparameters across all email datasets (discussed in Section 6.3.1). Left: Gamma prior over values of h. Right: estimated values of h and α per email dataset.

6.7 Experiments over Response Times

For these experiments, the advantages of applying the time transformation method discussed in Section 6.4.1 is explored. In particular, we measure the impact on the model's performance when transforming time according to the individual's activity function $a(t)$. The main models of interest are

- Direct models of response times (see Section 6.5.1) using exponential, Gamma, and lognormal distributions.
- Hawkes process models of response times (see Section 6.5.2) using exponential, Gamma, and lognormal triggering functions.

For each model, three separate versions are trained, each using different estimators of $a(t)$. The first approach sets $a(t) = 1$, i.e. no transformation of time is made, thus the effective and actual response times are equivalent. The second approach estimates $a(t)$ using the method described in Section 6.6, where the individual's circadian and weekly patterns are used to measure effective response times. The third approach uses a smoothed histogram (piecewise-constant) estimation of $a(t)$. The periodicity of $a(t)$ for this baseline estimation is set to one day, with the size of piecewise-constant intervals set to 1, 4, and 8 hours.

Appropriate priors are placed over each of the exponential, Gamma, and lognormal parameters; for example, a Gaussian prior is placed over the lognormal mean, and a Gamma prior is placed over the exponential mean. The values for each of the prior hyperparameters are set to be relatively weak, so that only the first few days worth of minibatches are significantly affected by the prior. For some parameters (such as those in the Gamma distribution), non-conjugate priors are used; for these parameters, MAP estimates are obtained numerically. Non-conjugate Gamma priors are placed over parameters specific to the Hawkes process, e.g. the ρ and ν scaling parameters discussed in Section 6.5.2.

The email datasets and experimental setup is identical to the previous experiments over the number of responses, described in Sections 6.3.1 and 6.3.2 respectively. The model parameters that are updated after processing each daily minibatch are 1) the distributional parameters, and 2) the activity function. When the model parameters are updated, the activity function is updated first, and then the distributional parameters.

6.7.1 Log-Likelihood of Response Times

The evaluation metric used in this experiment is the mean log-likelihood across all responses found in the minibatches of online data. For the direct model of response time, this is calculated as the second term in Equation 6.7, divided by the number of responses. For the Hawkes process model, the log-likelihood is calculated according to Equation 6.10. The log-likelihood of each response event is calculated with respect to the parameters estimated from data up to the day before the *corresponding incoming event* occurred. For example, if on day y the individual responded to an event that occurred on day $x \leq y$, then the log-likelihood of that response time is calculated using parameters estimated from data up to day $x - 1$.

Results

Figure 6.19 compares the differences in mean log-likelihood when transforming time via activity function $a(t)$, estimated as discussed in Section 6.6, and not transforming time by setting $a(t) = 1$ for both direct and Hawkes process models of response times. Each box-plot is over the 26 different email datasets, with larger positive values indicating a better fit to data when accounting for the transformation of time.

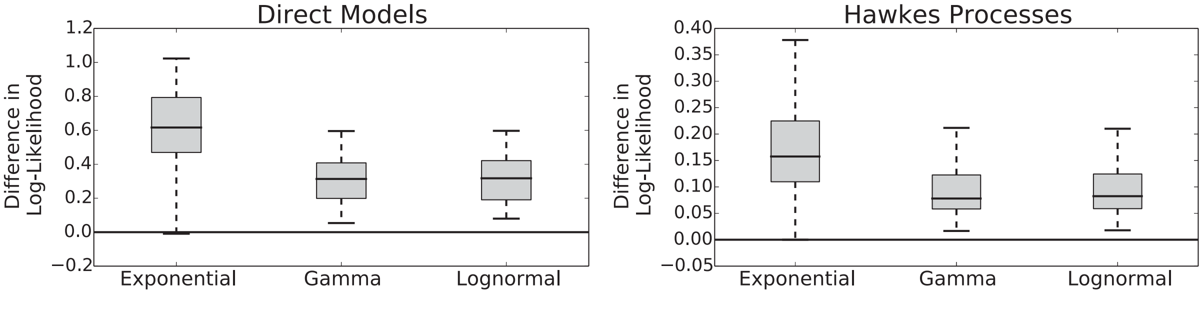


Figure 6.19: Difference in mean log-likelihood after transforming time according to $a(t)$.

A significant increase in log-likelihood is seen across all models when modeling a transformed distribution over response times ($q(\Delta t|\tau, \Omega)$, see Equation 6.6) induced by the activity func-

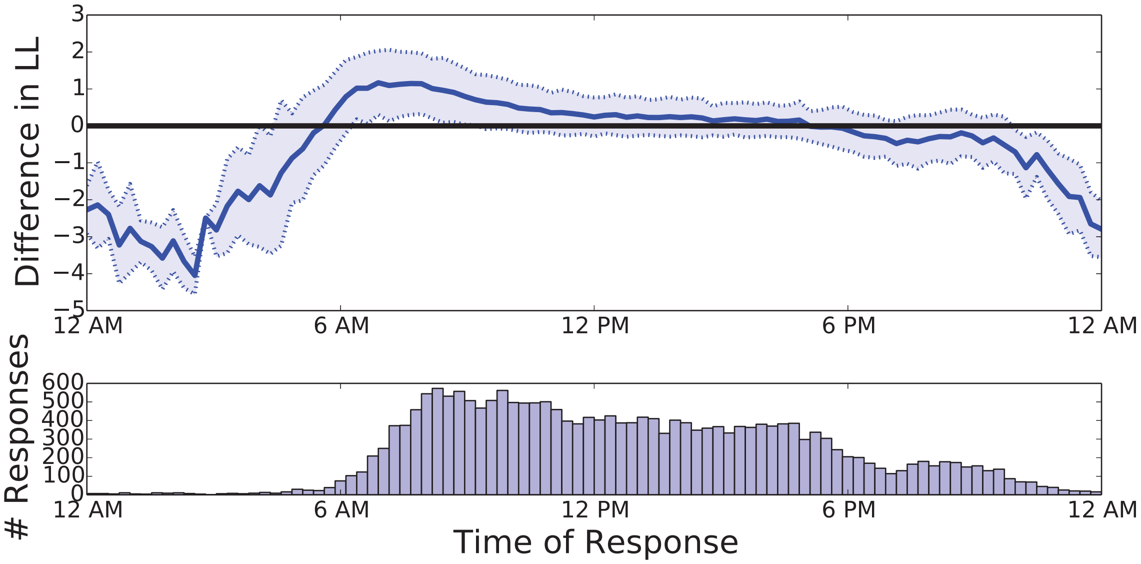


Figure 6.20: Difference in log-likelihood per event between estimating the activity function $a(t)$ and setting $a(t) = 1$. Top: differences as a function of the time-of-day the response was sent, for the direct response model via lognormal distribution, for a single individual's dataset. Bottom: number of responses sent as a function of time-of-day.

tion $a(t)$, compared to when $a(t) = 1$, i.e. no transformation of time takes place. This improvement in log-likelihood results from the ability of the model to rescale time to take into account the daily and weekly activity pattern of each individual. For example, the largest gain in response time log-likelihood occurs during the morning hours of the day, where the individual is presumably responding to emails from the previous night; this is precisely the scenario portrayed in Figure 6.8.

Figure 6.20 illustrates this point using a single individual's data, for the direct lognormal model over response times. Note that the difference in log-likelihood systematically *decreases* in the middle of the night; this is because the individual is typically asleep during this time, thus the model will always assign low probability to a response being sent during that time. However, the general effect in transforming time is positive, as these occurrences are rare (see bottom plot in Figure 6.20). Although not shown here, similar trends are seen for other distributional forms of response times, and for the Hawkes process model.

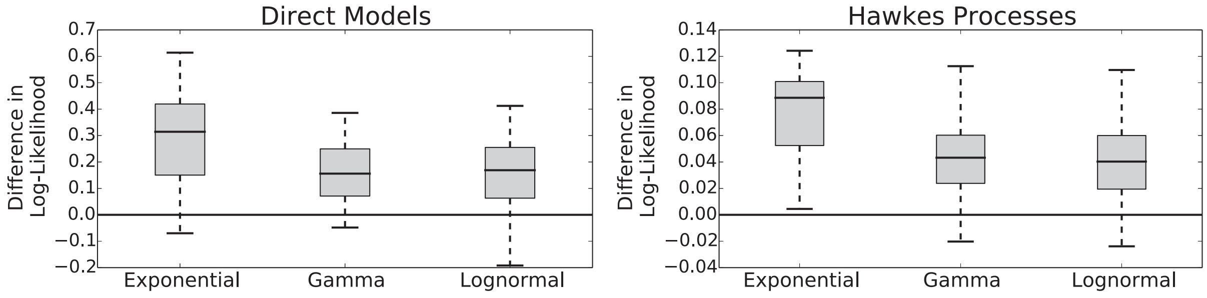


Figure 6.21: Difference in mean log-likelihood when estimating the activity function $a(t)$ between kernel and histogram methods.

Figure 6.21 compares the difference in log-likelihood between models where $a(t)$ is estimated as in Section 6.6 (referred to as the "kernel" estimate or method, due to its similarity to kernel density estimation), and where a smoothed histogram estimate is used. For these results, only the histogram estimate (using 1, 4, or 8 hour piecewise-constant intervals) that resulted in the largest log-likelihood is considered. Larger positive values indicate a better fit to data when the kernel estimate of $a(t)$ is used.

An increase in log-likelihood is generally seen across all models and datasets when estimates of $a(t)$ are given using the kernel method, as opposed to the histogram method. However, the magnitude of the increase is less than that compared to when $a(t) = 1$. This indicates that a simple histogram estimate of one's activity function will yield increased performance over a model that does not transform time, as general daily patterns (specifically, daytime and nighttime hours) are accounted for. The kernel estimation method will yield even greater increases in performance as histogram estimates must average over all events in a predefined time interval, thus increasing the level of approximation to the individual's actual activity.

6.7.2 Predicting Which Events will be Quickly Responded To

In this experiment, the model's power in predicting whether or not an incoming event will be responded to within a certain interval of time (e.g. within two hours) is explored. Here it is assumed that the individual will reply to the event and, conditioned on that, the task is to predict whether it will be responded to sooner or later. Being able to successfully make such predictions is useful for the task of creating a "priority inbox", where an individual's incoming emails are sorted by estimated importance (Aberdeen et al., 2010; Dredze et al., 2005).

The setup of this experiment is identical to that in the log-likelihood experiment, however the probability of a prompt response is calculated instead of the log-likelihood of the time of the response. Both direct and Hawkes process models over response times are considered, each using exponential, Gamma, and lognormal distributions. For each model, we compare against versions that 1) transform time according to the kernel estimation of $a(t)$, 2) transform time according to the histogram estimation of $a(t)$, and 3) do not transform time by setting $a(t) = 1$.

Evaluation Metric

Let $Q(\Delta t|\tau, \Omega)$ represent the cumulative density function of the distribution $q(\Delta t|\tau, \Omega)$ over response time (see Equation 6.6). For a given time window ϵ, the probability that an incoming event received at time τ will be responded to within ϵ time by the individual is

$$p_\tau = \begin{cases} Q(\Delta t = \epsilon|\tau, \Omega) & \text{for direct response-time models} \\ 1 - \exp\left(-Q(\Delta t = \epsilon|\tau, \Omega)\right) & \text{for Hawkes processes (Daley and Vere-Jones, 2003)} \end{cases}$$

After generating the probability of response p_{τ_i} for each incoming event i across minibatches (defined in Section 6.3.2), the set of resulting probabilities $\{p_{\tau_i}\}$ are sorted and the area under the ROC curve (AUC) is computed relative to binary ground truth (i.e. whether or not the incoming event was replied to within ϵ time). In this experiment, we report results

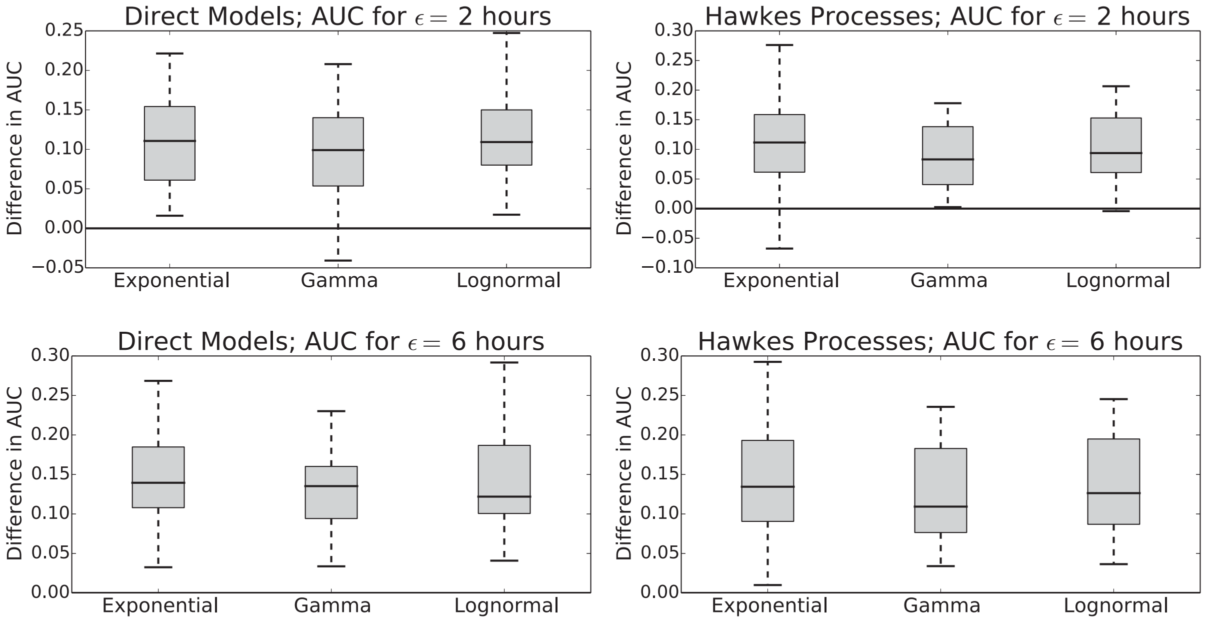


Figure 6.22: Difference in AUC between estimating the activity function with the kernel method and setting $a(t) = 1$ (i.e. no transformation of time).

for $\epsilon = 2$ hours and $\epsilon = 6$ hours — we obtained broadly similar results for other windows of time ranging from 1 to 8 hours.

Results

Figure 6.22 compares the differences in AUC between models where $a(t)$ is estimated with the kernel method and where $a(t) = 1$. As with the log-likelihood results in Section 6.7.1, modeling the transformation of time to account for the circadian and weekly patterns described by $a(t)$ results in a significant increase in predictive power across all models and datasets. This is largely due to the ability of the models to use transformed time to predict that the probability of a response within ϵ time is high (low) when the individual's activity during the time of the received event is typically high (low).

Figure 6.23 illustrates this phenomenon for two individual's email activity, where the fraction of emails responded to within $\epsilon = 2$ hours (blue dotted line) is shown as a function of time-

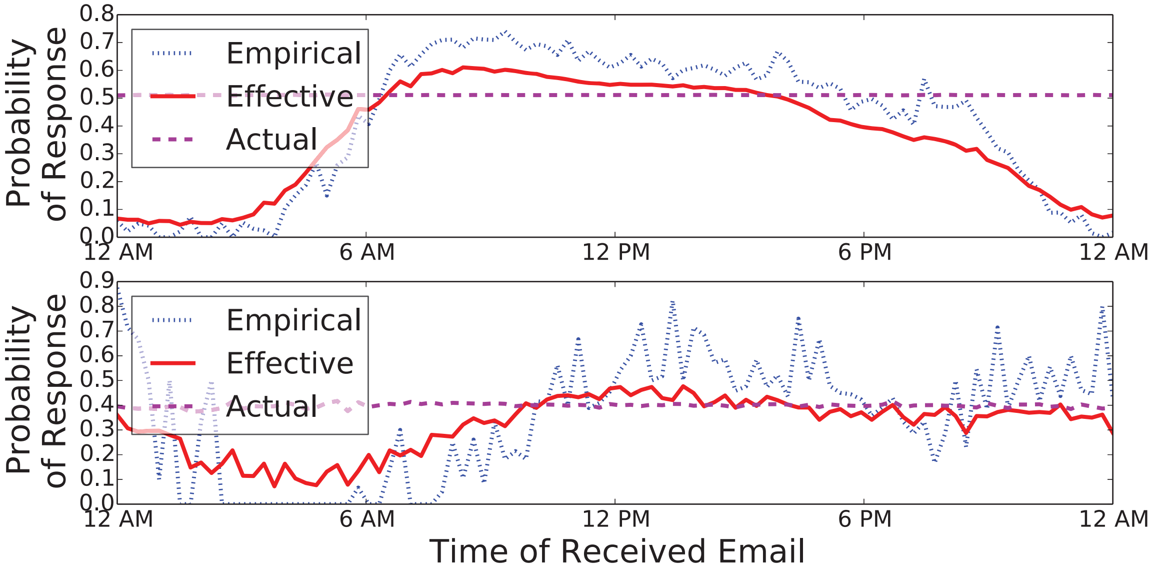


Figure 6.23: Probability of responding to an email as a function of the time-of-day the email was received, for two different individuals.

of-day the email was received. As expected, the probability of responding within two hours is lowest in the middle of the night, where the individuals are typically sleeping. When no transformation of time is given, the distribution over response times is invariant to the time of day, thus the probability of responding within two hours is constant[9] (the magenta dashed line). Transforming time according to $a(t)$ allows the distribution over response time to change shape according to the time of day, thus allowing the model to change the estimated probability of the individual responding within ϵ time based on their circadian and weekly patterns (the red solid line).

Similar to the log-likelihood results, the AUC for models where $a(t)$ is estimated as a piecewise-constant function generally falls between the estimation of $a(t) = 1$ and the kernel method of estimating the activity function (not shown). The ability to distinguish between daytime and nighttime hours in the histogram estimate allows the probability of responding within ϵ time to change as a function of time-of-day, providing an advantage over models where $a(t) = 1$. However, as the histogram estimate of $a(t)$ is a rough approximation of the

[9] Minor fluctuations in the estimated probability occur due to the online nature of parameter estimates

individual's activity, more accurate predictions can be made when estimating $a(t)$ using the kernel method.

6.8 Discussion and Future Work

In this chapter, we addressed the problem of modeling 1) the number of responses an individual sends to incoming events, and 2) the time it takes an individual to respond, in the context of egocentric event data. We expanded upon the work of Jo et al. (2012) by introducing a kernel method for estimating an individual's circadian and weekly patterns, used in a time transformation process for considering effective response times relative to those patterns. We demonstrated the flexibility of the time transformation technique by applying it to two differences approaches for modeling response times. While we focused on the application of an individual sending responses back to received emails, the techniques proposed in this chapter are modular and can be easily applied to any dataset described within the event response framework. Additionally, the time transformation mechanism is general in that it can be used within any model over response times.

Experimental results across 26 different email histories showed a significant improvement in both model fitting and predictive performance when introducing the activity function $a(t)$ into both direct and stochastic process models of response times. Thus suggests that the models were able to adapt to the strong circadian and weekly patterns experienced by the individual.

While the estimation procedure of $a(t)$ proposed in this chapter is straightforward, there are additional improvements that could be made. For instance, the value of the decay parameter α (determining the rate at which historical datapoints lose influence) is heuristically set based on properties of the 20% data set aside for validation. An alternative approach would be to set a prior over α and allow its value to change over time as additional minibatches of data are processed. A similar online approach could also be explored for the bandwidth parameter h (currently set and fixed to a value via cross-validation). Alternatively, an adaptive bandwidth parameter could be used, allowing h to vary across datapoints. Such a technique has been successfully explored for kernel density estimation (Breiman et al., 1977), and would reduce the need to regularize the estimation of $a(t)$ as described in Section 6.6.3.

Another direction of future work would be to apply the activity function to additional models of response behavior. For example, under the right normalization the activity function can be interpreted as the rate at which the individual sends an email over time. This rate could be used as a continuous alternative to the fixed-time-interval estimates of Poisson rates in non-homogeneous Poisson processes (Malmgren et al., 2008; Scott, 2000), providing a non-parametric estimation of the event rate (Møller and Waagepetersen, 2003, Section 4.3.1). Under this model, the time between consecutive emails the individual sends would follow the transformed exponential distribution in Section 6.5.1. However, as with the Hawkes process, this model would not describe the response time to an email, but rather the time it takes for the individual to send *any* email.

The last avenue of future work lies in the modeling choice of response times. While the shape of the distribution $q(\Delta t|\tau, \Omega)$ changes according to the individual's activity function, its shape is invariant to the event's metadata. A straightforward extension is to include the event's metadata within $q(\Delta t|\tau, \Omega)$, similar to the logistic regression model discussed in Section 6.2. We explored one such model — an "actor-based" model where separate parameters are given across all actors (e.g. the senders of emails the individual receives). While this variant experienced the same benefits in log-likelihood and predictive performance when including the transformation of time, their performance relative to the "global" model discussed in this chapter was generally lower, i.e. poorer. This is largely due to the actor-based model effectively partitioning the data, estimating actor-specific parameters from a sparse set of datapoints. Being able to properly handle the sparse data, or using a different approach for including metadata into the distribution over response times, could potentially improve the fitting of the models over response behavior.

6.9 Summary of Contributions

The primary contributions of this chapter include the following:

- We explored the modeling over the number of responses to an event as a combination between a logistic regression model indicating whether or not an event will receive a reply, and a Geometric distribution over the number of replies (conditioned on a reply being sent).
- We applied the above model to 26 email datasets, showing significant improvement in both log-likelihood and predictions over which emails will be responded to when event metadata (e.g. sender, number of recipients, etc) are accounted for.
- We introduced the concept of effective response time $\tilde{\Delta} t$, defined as the integration between the time of the received event and the time of the response with respect to an activity function $a(t)$. This activity function parameterizes the individual's circadian and weekly patterns.
- We defined the actual response time Δt to be a transformation of $\tilde{\Delta} t$, allowing complex distributions to be specified over Δt that vary in shape according to one's circadian and weekly patterns by defining standard distributions over $\tilde{\Delta} t$.
- We expanded upon the work of Jo et al. (2012) by developing an efficient inference procedure based on kernel methods for estimating the activity function $a(t)$, allowing

the estimation to adapt to behavioral changes over time. The inference procedure can also be applied recursively in an online fashion.

- We applied the time transformation technique to two different models of response times: direct response-time distributions and Hawkes processes (with different distributional forms of each). Results showed, across 26 email datasets, a significant improvement in both log-likelihood and predictive performance when transforming time according to one's circadian and weekly patterns.

Chapter 7

Conclusion

In this dissertation, we investigated generative probabilistic models for statistically describing properties of communication behavior. We focused primarily on the application of email communication, and explored models over 1) the recipient patterns in emails the individual sends, and 2) the time it takes for the individual to respond to a received email.

For modeling the recipients which appear in an email (or, more generally, the collection of items appearing in an *itemset*), we introduced a kernel density estimator for learning a probability density over possible itemsets given historical data. This estimator can be viewed as a non-symmetrical generalization of Aitchison and Aitken (1976). We show, both theoretically and empirically over approximately 100 individual email histories over four different email datasets, that the non-symmetrical nature of the estimator is able to better capture the sparsity traits commonly found in itemset data. Thus, the proposed kernel model is a better fit to unobserved data in terms of log-likelihood. We explored a stochastic gradient approach for estimating the kernel parameters as new data is observed over time, showing that similar estimates can be obtained compared to traditional cross-validation or gradient approaches in a much shorter amount of time.

Additionally, we showed that the kernel model can be viewed as a special form of *Bernoulli mixture model.* We detailed different properties of Bernoulli mixture models, including the form of their marginal and conditional distributions, in addition to how the dependencies between different recipients of an email are captured. We used the Bernoulli mixture model (and kernel model) for two predictive tasks related to the general problem of *itemset completion*: given a partially observed list of recipients, 1) find the most-likely configuration of the remaining set of recipients, and 2) rank potential recipients by their likelihood to be added to the email. Experimental results across many email datasets showed that the kernel and Bernoulli mixture models are competitive against a baseline model specific to the task of itemset completion.

Next, we investigated the clustering of email recipients when time is accounted for. We modeled the individual's rate of sending emails towards each latent cluster of recipients as a piecewise-constant Poisson process, where changes in the Poisson rate indicate significant changes in behavior between the individual and corresponding recipients. We modeled email recipients using a simplified form of a Bernoulli mixture model, where the latent cluster variable is assigned based on the group-specific Poisson rates. We derived a MCMC inference method for learning the model parameters given data, and gave a detailed interpretation of inferred parameters given both real and simulated data. Experiments over several email datasets spanning multiple years indicated that the model is able to simultaneously infer recipient groups and the time-varying activity between the individual and these groups, compared to simpler baseline methods.

Finally, we considered probabilistic models over the time it takes for an individual to respond to an incoming event, such as receiving an email. We introduced a "time-warping" mechanism where the absolute response time is defined as a transformation of the *effective* response time relative to an individual's *activity function.* We parameterized the activity function such that the individual's circadian and weekly patterns were properly captured, and provided an iterative non-parametric approach for learning such patterns. We showed the modularity of this approach by applying the time-warping mechanism to two different probabilistic models of event response times: distributions directly over response time and Hawkes processes. In both cases, experimental results showed a significant increase in both log-likelihood and accuracy in predicting which emails will be quickly responded to when the time-warping mechanism was applied.

As accessing and interfacing with the Internet becomes more convenient (e.g. tablets, smartphones, wifi hotspots, etc), digital communication such as email is becoming increasingly prevalent in our daily lives. As such, there is likely to be increasing interest and need in using statistical models in order to better understand and predict such behavior — particularly from an egocentric viewpoint. This dissertation can be viewed as a starting point in the statistical analysis of such data — in the Discussion sections of Chapters 3 through 6, additional future directions for research on this topic is provided. In the near future, we expect that the increasing computation abilities of devices such as cellphones will allow for the development of advanced exploratory and communication management systems based on advanced statistical theory.

Bibliography

O. Aalen, O. Borgan, and H. Gjessing. *Survival and Event History Analysis: A Process Point of View*. Springer, 2008.

D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind Gmail priority inbox. In *NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*, 2010.

C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 575–586. ACM, 2003.

E. Airoldi, D. Blei, S. Fienberg, and E. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

J. Aitchison and C. Aitken. Multivariate binary discrimination by the kernel method. *Biometrika*, 63(3):413–420, 1976.

A. Al-Alwani. A novel email response algorithm for email management systems. *Journal of Computer Science*, 10(4):689–696, 2014.

D. Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII – 1983*, volume 1117 of *Lecture Notes in Mathematics*, pages 1–198. Springer Berlin Heidelberg, 1985.

A. Aliabadi, F. Razzaghi, S. Kochak, and A. Ghorbani. Classifying organizational roles using email social networks. In *Advances in Artificial Intelligence*, volume 7884, pages 301–307. Springer Berlin Heidelberg, 2013.

C. Andrieu, N. de Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1–2):5–43, 2003.

I. Auger and C. Lawrence. Algorithms for the optimal identification of segment neighborhoods. *Bulletin of Mathematical Biology*, 51(1):39–54, 1989.

Z. Bar-Yossef, I. Guy, R. Lempel, Y. Maarek, and V. Soroka. Cluster ranking with an application to mining mailbox networks. *Knowledge and Information Systems*, 14(1):101–139, 2008.

A. Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039): 207–211, 2005.

M. Beal, Z. Ghahramani, and C. Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*, pages 577–584. MIT Press, 2002.

T. Berger-Wolf and J. Saia. A framework for analysis of dynamic social networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 523–528. ACM, 2006.

C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

D. Blei and J. Lafferty. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. ACM, 2006.

D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

C. Blundell, J. Beck, and K. Heller. Modelling reciprocating relationships with Hawkes processes. In *Advances in Neural Information Processing Systems 25*, pages 2600–2608. Curran Associates, Inc., 2012.

L. Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, pages 9–42, 1998.

L. Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer, 2012.

A. Bowman, P. Hall, and D. Titterington. Cross-validation in nonparametric estimation of probabilities and probability densities. *Biometrika*, 71(2):341–351, 1984.

J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 43–52. Morgan Kaufmann Publishers, 1998.

L. Breiman, W. Meisel, and E. Purcell. Variable kernel estimates of multivariate densities. *Technometrics*, 19(2):135–144, 1977.

P. Brown and P. Rundell. Kernel estimates for categorical data. *Technometrics*, 27(3): 293–299, 1985.

J. Bryant, A. Sanders-Jackson, and A. Smallwood. IMing, text messaging, and adolescent social networks. *Journal of Computer Mediated Communication*, 11(2):577–592, 2006.

C. Butts. A relational event framework for social action. *Sociological Methodology*, 38(1): 155–200, 2008.

R. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

V. Carvalho and W. Cohen. Ranking users for intelligent message addressing. In *Advances in Information Retrieval*, volume 4956 of *Lecture Notes in Computer Science*, pages 321–333. Springer Berlin Heidelberg, 2008.

A. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comptes Rendu des Séances de L'Académie des Sciences*, 25:536–538, 1847.

S. Chib. Estimation and comparison of multiple change-point models. *Journal of Econometrics*, 86(2):221–241, 1998.

D. Choi, P. Wolfe, and E. Airoldi. Stochastic blockmodels with a growing number of classes. *Biometrika*, 99(2):273–284, 2012.

T. Cover and J. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications and Signal Processing. Wiley-Interscience, 2 edition, 2006.

D. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes. Vol. I.* Springer-Verlag, 2003.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

A. Dawid. Prequential data analysis. In *Current Issues in Statistical Inference: Essays in Honor of D. Basu*, volume 17 of *Lecture Notes-Monograph Series*, pages 113–126. 1992.

W. de Nooy. Networks of action and events over time. A multilevel discrete-time event history model for longitudinal network data. *Social Networks*, 33(1):31–40, 2011.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

J. Desmet, M. Maeyer, B. Hazes, and I. Lasters. The dead-end elimination theorem and its use in protein side-chain positioning. *Nature*, 356(6369):539–542, 1992.

A. Devkar and P. McReynolds. Conspire, 2013. `https://www.conspire.com/`.

J. Diesner, T. Frantz, and K. Carley. Communication networks from the Enron email corpus: it's always about the people. Enron is no different. *Computational & Mathematical Organization Theory*, 11(3):201–228, 2005.

M. Dredze, J. Blitzer, and F. Pereira. Reply expectation prediction for email management. In *2nd Conference on Email and Anti-Spam*, Stanford, CA, 2005.

M. Dredze, B. Schilit, and P. Norvig. Suggesting email view filters for triage and search. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1414–1419. Morgan Kaufmann Publishers Inc., 2009a.

M. Dredze, H. Wallach, D. Puller, T. Brooks, J. Carroll, J. Magarick, J. Blitzer, and F. Pereira. Intelligent email: aiding users with AI. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 1524–1527. AAAI Press, 2009b.

R. Duda and P. Hart. *Pattern Recognition and Scene Analysis*. Wiley, 1973.

J. Eckmann, E. Moses, and D. Sergi. Entropy of dialogues creates coherent structures in e-mail traffic. *Proceedings of the National Academy of Sciences of the United States of America*, 101(40):14333–14337, 2004.

P. Fearnhead. Exact and efficient Bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213, 2006.

D. Fisher. Using egocentric networks to understand communication. *IEEE Internet Computing*, 9(5):20–28, 2005.

D. Fisher, A. Brush, E. Gleave, and M. Smith. Revisiting Whittaker Sidner's "email overload" ten years later. In *Proceedings of the 20th Conference on Computer Supported Cooperative Work*, pages 309–312. ACM, 2006.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. A sticky HDP-HMM with application to speaker diarization. *The Annals of Applied Statistics*, 5(2A):1020–1056, 2011.

E. Fox, M. Short, F. Schoenberg, K. Coronges, and A. Bertozzi. Modeling e-mail networks and inferring leadership using self-exciting point processes. 2013. Preprint.

L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer-Mediated Communication*, 3(1), 1997.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

D. Graus, D. van Dijk, M. Tsagkias, W. Weerkamp, and M. de Rijke. Recipient recommendation in enterprises using communication graphs and email content. In *37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1079–1082. ACM, 2014.

C. Grillenzoni. Sequential kernel estimation of the conditional intensity of nonstationary point processes. *Statistical Inference for Stochastic Processes*, 9(2):135–160, 2006.

R. Grob, M. Kuhn, R. Wattenhofer, and M. Wirz. Cluestr: mobile social networking for enhanced group communication. In *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, pages 81–90. ACM, 2009.

B. Grund. Kernel estimators for cell probabilities. *Journal of Multivariate Analysis*, 46(2): 283–308, 1993.

V. Guralnik and J. Srivastava. Event detection from time series data. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 33–42. ACM, 1999.

P. Hall. On nonparametric multivariate binary discrimination. *Biometrika*, 68(1):287–294, 1981.

P. Halpin and P. De Boeck. Modelling dyadic interaction with Hawkes processes. *Psychometrika*, 78(4):793–814, 2013.

W. Hastings. Monte Carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

A. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

A. Hawkes and D. Oakes. A cluster process representation of a self-exciting process. *Journal of Applied Probability*, 11(3):493–503, 1974.

C. Heinz and B. Seeger. Towards kernel density estimation over streaming data. In *Proceedings of the 13th International Conference on Management of Data*, pages 80–91. Tata McGraw-Hill, 2006.

A. Ihler, J. Hutchins, and P. Smyth. Learning to detect events with Markov-modulated Poisson processes. *ACM Transactions on Knowledge Discovery from Data*, 1(3), 2007.

H. Jo, M. Karsai, J. Kertész, and K. Kaski. Circadian pattern and burstiness in mobile phone communication. *New Journal of Physics*, 14(1), 2012.

M. Jones, J. Marron, and S. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433):401–407, 1996.

A. Juan and E. Vidal. On the use of Bernoulli mixture models for text classification. *Pattern Recognition*, 35(12):2705–2710, 2002.

A. Juan and E. Vidal. Bernoulli mixture models for binary images. In *Proceedings of the 17th International Conference on Pattern Recognition*, pages 367–370. IEEE, 2004.

A. Juan, J. García-Hernández, and E. Vidal. EM initialisation for Bernoulli mixture learning. In *Structural, Syntactic, and Statistical Pattern Recognition*, volume 3138 of *Lecture Notes in Computer Science*, pages 635–643. Springer Berlin Heidelberg, 2004.

A. Kaltenbrunner, V. Gómez, A. Moghnieh, R. Meza, J. Blat, and V. López. Homogeneous temporal activity patterns in a large online communication space. *IADIS International Journal on WWW/INTERNET*, 6(1):61–76, 2008.

M. Karsai, N. Perra, and A. Vespignani. Time varying networks and the weakness of strong ties. *Scientific reports*, 4, 2014.

J. Kiefer. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4(3):502–506, 1953.

J. Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

B. Klimt and Y. Yang. The Enron corpus: a new dataset for email classification research. In *Proceedings of the European Conference on Machine Learning*, pages 217–226. Springer, 2004.

Y. Koren, E. Liberty, Y. Maarek, and R. Sandler. Automatically tagging email by leveraging other users' folders. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 913–921. ACM, 2011.

B. Letham, C. Rudin, and D. Madigan. Sequential event prediction. *Machine learning*, 93 (2-3):357–380, 2013.

P. Liang and D. Klein. Online EM for unsupervised models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 611–619. Association for Computational Linguistics, 2009.

Q. Liu and A. Ihler. Variational algorithms for marginal map. *Journal of Machine Learning Research*, 14:3165–3200, 2013.

D. Loftsgaarden and C. Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, 1965.

D. MacLean, S. Hangal, S. Teh, M. Lam, and J. Heer. Groups without tears: mining social topologies from email. In *Proceedings of the 16th International Conference on Intelligent User Interfaces*, pages 83–92. ACM, 2011.

R. Malmgren, D. Stouffer, A. Motter, and L. Amaral. A Poissonian explanation for heavy tails in e-mail communication. *Proceedings of the National Academy of Sciences of the United States of America*, 105(47):18153–18158, 2008.

R. Malmgren, J. Hofman, L. Amaral, and D. Watts. Characterizing individual communication patterns. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 607–616. ACM, 2009.

G. Mark, S. Voida, and A. Cardello. A pace not dictated by electrons: an empirical study of work without email. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 555–564. ACM, 2012.

N. Masuda, T. Takaguchi, N. Sato, and K. Yano. Self-exciting point process modeling of conversation event sequences. In *Temporal Networks*, Understanding Complex Systems, pages 245–264. Springer Berlin Heidelberg, 2013.

J. McAuley and J. Leskovec. Learning to discover social circles in ego networks. In *Advances in Neural Information Processing Systems*, pages 548–556. Curran Associates, Inc., 2012.

A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on Enron and academic email. *Journal of Artificial Intelligence Research*, 30(1):249–272, 2007.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

R. Michalski, S. Palus, and P. Kazienko. Matching organizational structure and social network extracted from email communication. In *Lecture Notes in Business Information Processing*, volume 87, pages 197–206. Springer Berlin Heidelberg, 2011.

G. Miritello, E. Moro, R. Lara, R. Martínez-López, J. Belchamber, S. Roberts, and R. Dunbar. Time as a limited resource: communication strategy in mobile phone networks. *Social Networks*, 35(1):89–95, 2013.

A. Mokkadem, M. Pelletier, and Y. Slaoui. The stochastic approximation method for the estimation of a multivariate probability density. *Journal of Statistical Planning and Inference*, 139(7):2459–2478, 2009.

J. Møller and R. Waagepetersen. *Statistical Inference and Simulation for Spatial Point Processes.* Chapman & Hall/CRC, 2003.

N. Navaroli, C. DuBois, and P. Smyth. Statistical models for exploring individual email communication behavior. In *Proceedings of the 4th Asian Conference on Machine Learning*, volume 25, pages 317–332. JMLR, 2012.

N. Navaroli, C. DuBois, and P. Smyth. Modeling individual email patterns over time with latent variable models. *Machine Learning*, 92(2-3):431–455, 2013.

R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

R. Neal. Slice sampling. *Annals of statistics*, 31(3):705–767, 2003.

J. Nocedal and S. Wright. *Numerical Optimization.* Springer, 1999.

J. Olson and K. Carley. Exact and approximate EM estimation of mutually exciting Hawkes processes. *Statistical Inference for Stochastic Processes*, 16(1):63–80, 2013.

C. Pal and A. McCallum. CC prediction with graphical models. In *Proceedings of the Third Conference on Email and Anti-Spam*, 2006.

J. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.

A. Pozdnoukhov and F. Walsh. Exploratory novelty identification in human activity data streams. In *Proceedings of the ACM SIGSPATIAL International Workshop on GeoStreaming*, pages 59–62. ACM, 2010.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

S. Ross. *Introduction to Probability Models, Ninth Edition.* Academic Press, Inc., 2006.

M. Roth, A. Ben-David, D. Deutscher, G. Flysher, I. Horn, A. Leichtberg, N. Leiser, Y. Matias, and R. Merom. Suggesting friends using the implicit social graph. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 233–242. ACM, 2010.

C. Rudin, B. Letham, A. Salleb-Aouissi, E. Kogan, and D. Madigan. Sequential event prediction with association rules. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 615–634. JMLR, 2011.

S. Sain and D. Scott. On locally adaptive density estimation. *Journal of the American Statistical Association*, 91(436):1525–1534, 1996.

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295. ACM, 2001.

S. Scott. Detecting network intrusion using a Markov modulated nonhomogeneous Poisson process. Technical report, 2000.

S. Scott. A Bayesian paradigm for designing intrusion detection systems. *Computational Statistics and Data Analysis*, 45(1):69–83, 2004.

S. Scott and P. Smyth. The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic modeling. *Bayesian Statistics*, 7:671–680, 2003.

S. Sheather and M. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B*, 53(3):683–690, 1991.

J. Shetty and J. Adibi. The Enron email dataset database schema and brief statistical report. Technical report, 2004.

ShuttleCloud Corp. Gmail meter, 2014. `http://gmailmeter.com/`.

B. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

A. Simma and M. Jordan. Modeling events with cascades of Poisson processes. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 546–555. AUAI Press, 2010.

D. Smilkov, D. Jagdish, and C. Hidalgo. Immersion: a people-centric view of your email life, 2013. `https://immersion.media.mit.edu/`.

D. Stouffer, R. Malmgren, and L. Amaral. Log-normal statistics in e-mail communication patterns. *arXiv preprint physics/0605027*, 2006.

C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 717–726. ACM, 2007.

J. Tikka, J. Hollmn, and S. Myllykangas. Mixture modeling of DNA copy number amplification patterns in cancer. In *Computational and Ambient Intelligence*, volume 4507 of *Lecture Notes in Computer Science*, pages 972–979. Springer Berlin Heidelberg, 2007.

D. Titterington. A comparative study of kernel-based density estimates for categorical data. *Technometrics*, 22(2):259–268, 1980.

G. Tutz. An alternative choice of smoothing for kernel-based density estimates in discrete discriminant analysis. *Biometrika*, 73(2):405–411, 1986.

G. Tutz and H. Grob. Discrete kernels, loss functions and parametric models in discrete discrimination: a comparative study. *Zeitschrift für Operations Research*, 42(2):217–230, 1995.

J. Wainer, L. Dabbish, and R. Kraut. Should I open this email?: inbox-level cues, curiosity and attention to email. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*, pages 3439–3448. ACM, 2011.

C. Wang, D. Blei, and D. Heckerman. Continuous time dynamic topic models. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, pages 579–586. AUAI Press, 2008.

S. Whittaker, T. Matthews, J. Cerruti, H. Badenes, and J. Tang. Am I wasting my time organizing email?: a study of email refinding. In *Proceedings of the 2011 Annual Conference on Human factors in Computing Systems*, pages 3449–3458. ACM, 2011.

M. Woodroofe. On choosing a delta-sequence. *The Annals of Mathematical Statistics*, 41(5): 1665–1671, 1970.

K. Xu, M. Kliger, and A. Hero. Tracking communities in dynamic social networks. *Social Computing, Behavioral-Cultural Modeling and Prediction*, 6589:219–226, 2011.

T. Zaman, E. Fox, and E. Bradlow. A Bayesian approach for predicting the popularity of tweets. *Annals of Applied Statistics*, 8(3):1583–1611, 2014.

A. Zambom and R. Dias. A review of kernel density estimation with applications to econometrics. *arXiv preprint arXiv:1212.2812*, 2012.

M. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

L. Zenk, C. Stadtfeld, and F. Windhager. How to analyze dynamic network patterns of high performing teams. *Procedia-Social and Behavioral Sciences*, 2(4):6418–6422, 2010.

A. Zhou, Z. Cai, L. Wei, and W. Qian. M-kernel merging: towards density estimation over data streams. In *Proceedings of the Eighth International Conference on Database Systems for Advanced Applications*, pages 285–292. IEEE Computer Society, 2003.

M. Zhou, W. Zhang, B. Smith, E. Varga, M. Farias, and H. Badenes. Finding someone in my social directory whom I do not fully remember or barely know. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, pages 203–206, 2012.

J. Zipkin, F. Schoenberg, K. Coronges, and A. Bertozzi. Point-process models of social network interactions: parameter estimation and missing data recovery. 2014. Under Review.

Appendix A

Processing of Email Datasets

This appendix gives an overview of the preprocessing methods used to parse the email datasets described in Section 1.1. All email processing code is publicly available[1].

A.1 Parsing Email Corpora

Personal Gmail Data

A Python script was prepared that downloads all email headers associated with a personal Gmail account, and stores anonymized versions of those headers to file. To minimize privacy concerns, only the following information was downloaded from Gmail's servers:

- The message ID corresponding to the email.
- The date of the message.
- The email address of the sender.
- The email addresses in the to / cc / bcc fields.
- If the email was a response to another email, then the message ID of the original email.

[1] Available at `http://www.datalab.uci.edu/resources/` and tested using a Python 2.7 installation with numpy 1.8.1, scipy 0.14.0, and matplotlib 1.3.1 installed.

No subject text, body text, or attachments were downloaded from the Gmail servers. An example of an email header that would be downloaded is shown in Figure A.1.

```
In-Reply-To: <CAB=JVjKPMm_ac3oqO3T1FbzkQ@mail.gmail.com>
From: Person A <first.last@gmail.com>
Date: Sun, 25 Nov 2012 11:47:27 -0800
Message-ID: <CAB=JVj+szdgBDFALwwBNEA@mail.gmail.com>
To: Person B <person@ics.uci.edu>
Cc: Person C <address@gmail.com>, Nicholas Navaroli <nnavarol@uci.edu>
```

Figure A.1: Example email header downloaded from Gmail server.

While the message ID and email addresses are not anonymized from the server, they are anonymized to integers prior to storing on file. Each message ID was assigned to a unique integer, and email addresses were normalized before anonymization. Normalization of email addresses is needed due to inconsistencies in formatting; all email addresses were lower-cased with any punctuation removed. For example, the following email addresses would all be normalized to the same anonymous ID:

$$\left.\begin{array}{l}\text{nick.navaroli@gmail.com}\\ \text{NickNavaroli@gmail.com}\\ \text{nick.navaroli@GMAIL.COM}\\ \text{NICK.NAVAROLI@G.MAIL.COM}\end{array}\right\}\text{nicknavaroli@gmailcom} \rightarrow \text{anonymous ID 123}$$

Each email header is stored on file under the following format:

```
[anonymous email id],[anonymous email id of original email, if applicable],
   [date string],[anonymous id of sender],
   [anonymous ids of addresses in "to" field, separated by "|"],
   [anonymous ids of addresses in "cc" field, separated by "|"],
   [anonymous ids of addresses in "bcc" field, separated by "|"]
```

For example, the email header from Figure A.1 might look like

```
123,37,2012/11/25 11:47:27 -0800,111,222,345|517,
```

As part of the Python script, the user is asked to provide any aliases they may have (i.e. school or work accounts linked to their Gmail account). The anonymous ids associated with the main email address and all aliases are then merged into a single “super account”.

In addition to the saved header file, a second file is created that reverses the mapping of anonymous ID back to original email address. This is used for the sole purpose of qualitatively evaluating the models in this thesis. Only the owner of the email account (that runs the Python script) keeps this file; for the purpose of experimentation and modeling only a copy of the anonymous header file is used.

Enron Data

The Enron email corpus is publicly available[2], with each email stored as a text file containing *all* header information (including text and attachments) and separated into folders based on employee and their self-made inbox directories. We parsed the email header in each file, ignoring the folder it was stored in, using the same process and anonymization as with the Gmail data (ignoring the text and other fields). There were originally 517,424 emails, with 244,436 remaining after removing duplicates found across multiple employee’s inboxes.

Unfortunately, the response structure of emails (the “In-Reply-To” field in Figure A.1) was omitted. We used the following heuristic to determine if email X was a response to email Y:

1. Is the sender of email X a recipient of email Y (in any of the to / cc / bcc fields)?
2. Are the normalized subjects (i.e., removing “Re:” and “Fwd:” tags) equivalent?
3. Was email X sent within 140 days after email Y was sent?

If the answer to all three questions is “yes”, then we marked X as a response to Y. Inferring response structure is the only usage of the email subject lines; they were discarded afterwards.

[2] See `http://www.cs.cmu.edu/~./enron/`. For this thesis, we parsed the version dated August 21, 2009.

Next, the email inboxes for all employees were reconstructed so that each inbox contained all emails the employee was involved in. A separate file for each employee is created containing the relevant header information, using the same format as the Gmail data.

Eckmann Data

All email interactions of the Eckmann email corpus are stored in a single file, with email interactions taking the following format:

```
[id of sender] [id of recipient] [size of email] [date of email]
```

where the user ids are anonymous integers. If an email was sent to R recipients, the interaction was recorded as R different interactions, one to each recipient. We assume the size of the email is in units of bytes, and measures the length of the header and text. No other information regarding the emails was provided.

We use an approach similar to Malmgren et al. (2009) for preprocessing the dataset. First, interactions that differed by at most 5 seconds and 5 units in size were merged together by concatenating the recipient lists, preserving the original ordering of recipients within the raw data. Email inboxes were then reconstructed for each user by associating with the inbox all emails the user was involved in, and saved to file.

Manufacturing Data

The Manufacturing email corpus is publicly available[3], with email interactions across all employees stored in a single file with the following form:

[3] Visit `http://konect.uni-koblenz.de/networks/radoslaw_email` to download the data.

```
[anonymous id of sender] [anonymous id of recipient] [date of the email]
```

where multi-recipient emails were recorded as multiple interactions, one to each recipient (similar to the Eckmann data). Interactions with identical {sender, date} combinations had their recipient lists merged, preserving the recipient order within the raw data. No other information regarding the emails was provided. Email inboxes were then created for each employee using all emails the employee was involved in, and saved to file.

A.2 Preprocessing of Emails and Inboxes

Once each email corpus had their respective inboxes stored in separate files, both the individual emails and inboxes were preprocessed prior to any experiments within this thesis.

First, each email went through the following preprocessing steps:

1. If the date was missing or invalid, or the email did not have any recipients specified across the to / cc / bcc fields, the email was removed from the inbox.
2. Duplicate recipients (across the to / cc / bcc fields) were removed, keeping only the recipient's first appearance.

Second, each email inbox went through the following preprocessing steps:

1. The first and last 1% of the emails were removed. This helped to filter out emails with unusual dates (i.e. a year before 1980 or after 2020).
2. Response emails that appear to be forwarded messages (i.e. the recipient(s) of the response do not appear in the original email) had their response field removed[4].

[4] The removal of the response field treats these messages as new email threads.

3. Emails that are "self-responses" (the sender of both the response and original email are equivalent) had their response field removed.

4. The response structure of emails with multiple responses (i.e. follow-up/correction emails sent after the initial response) were corrected[5].

Lastly, email inboxes within the same corpus (except for the Gmail corpus) were filtered according to the following:

1. Individuals that received less than 50 emails or sent less than 100 emails had their inboxes removed from the corpus.

2. Individuals with unusual activity, according to the statistic $\log\left(\frac{\#\text{ received}}{\#\text{ sent}}\right)$ (Malmgren et al., 2009), had their inboxes removed from the corpus. An "unusual" value was considered to be one outside of two standard deviations from the mean statistic across individuals in the corpus. Individuals removed according to this statistic typically correspond to specialized email accounts, such as mailing lists.

Table A.1 shows the number of inboxes remaining after filtering, for each email corpus. The experiments throughout this dissertation apply additional inbox filtering methods, further reducing the number of email inboxes considered. Such filtering methods are discussed in their respective sections.

Email Corpus	# Original inboxes	# Inboxes after filtering
Eckmann	2997	213
Enron	249	109
Gmail	4	4
Manufacturing	167	93

Table A.1: Number of inboxes before and after filtering for each dataset.

[5] As an example, suppose emails B and C were sent by the same individual and were responses to email A. Often, the email client will mark email C as a response to email B (assume C came after B), when in fact it was a response to email A. This is easy to detect and correct via recipient matching.

Appendix B

Derivations of Equations 4.3a and 4.3b

When modeling a D-dimensional binary vector x with a Bernoulli mixture model (Section 2.1.2), we are interested in finding dimensions that satisfy one of the following conditions:

a) $P(x_d = 1, x_{-d}|\pi, \theta) \geq P(x_d = 0, x_{-d}|\pi, \theta)$ for all x_{-d}

b) $P(x_d = 1, x_{-d}|\pi, \theta) \leq P(x_d = 0, x_{-d}|\pi, \theta)$ for all x_{-d}

If condition a (or b) is satisfied for a dimension d, then it must be that the MAP instantiation $\hat{x} \equiv \arg\max_x P(x|\pi, \theta)$ has $\hat{x}_d = 1$ (or 0).

To derive Equations 4.3a and 4.3b, we define the following:

$$\Delta^d_{1\to 0} \equiv P(x_d = 0, x_{-d}|\pi, \theta) - P(x_d = 1, x_{-d}|\pi, \theta)$$

$$\Delta^d_{0\to 1} \equiv P(x_d = 1, x_{-d}|\pi, \theta) - P(x_d = 0, x_{-d}|\pi, \theta) = -\Delta^d_{1\to 0}$$

which are the differences in probability when changing the value of x_d from 1 to 0 or from 0 to 1, respectively (conditioned on the value of x_{-d}).

Using the density of the Bernoulli mixture model (Equation 2.4), the value of $\Delta^d_{1\to 0}$ is

$$\begin{aligned}\Delta^d_{1\to 0} &\equiv P(x_d = 0, x_{-d}|\pi,\theta) - P(x_d = 1, x_{-d}|\pi,\theta)\\ &= \sum_{k=1}^{K} \pi_k(1-\theta_{kd}) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} - \sum_{k=1}^{K} \pi_k\theta_{kd} \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i}\\ &= \sum_{k=1}^{K} \pi_k(1-2\theta_{kd}) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i}\\ &= \sum_{k:\theta_{kd}\leq\frac{1}{2}} \pi_k(1-2\theta_{kd}) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(1-2\theta_{kd}) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i}\end{aligned} \tag{B.1}$$

where the summation is split into two parts in the last equality: the first over mixture components where $\theta_{kd} \leq \frac{1}{2}$, and the other where $\theta_{kd} > \frac{1}{2}$. All terms in the first summation are nonnegative, while the terms in the second summation are negative.

A similar derivation is made for the value of $\Delta^d_{0\to 1}$:

$$\begin{aligned}\Delta^d_{0\to 1} &= -\Delta^d_{1\to 0}\\ &= \sum_{k:\theta_{kd}\leq\frac{1}{2}} \pi_k(2\theta_{kd}-1) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(2\theta_{kd}-1) \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i}\end{aligned} \tag{B.2}$$

To simplify notation, we define two additional terms:

$$\alpha_{kd} \equiv \min_{x_{-d}} \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} \qquad\qquad \beta_{kd} \equiv \max_{x_{-d}} \prod_{i\neq d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i}$$

which are the minimum and largest achievable probabilities for any value of x_{-d}, after marginalizing over x_d, with respect to component k.

Cases where $\hat{x}_d = 0$

Continuing our derivation of $\Delta^d_{1\to 0}$ from Equation B.1, we have

$$\begin{aligned}
\Delta^d_{1\to 0} &= \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(1-2\theta_{kd}) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(1-2\theta_{kd}) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} \\
&\ge \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(1-2\theta_{kd})\alpha_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(1-2\theta_{kd}) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} \\
&\ge \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(1-2\theta_{kd})\alpha_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(1-2\theta_{kd})\beta_{kd} \qquad \text{(B.3)}
\end{aligned}$$

Thus, the quantity

$$\sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(1-2\theta_{kd})\alpha_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(1-2\theta_{kd})\beta_{kd}$$

is a lower bound on $P(x_d = 0, x_{-d}|\pi,\theta) - P(x_d = 1, x_{-d}|\pi,\theta)$. If the above quantity is greater than 0 (corresponding to the inequality of Equation 4.3a), then the probability of x is always larger when x_d is set to 0. Thus, the value of x_d under the MAP instantiation must be 0.

Cases where $\hat{x}_d = 1$

Continuing our derivation of $\Delta^d_{0\to 1}$ from Equation B.2, we have

$$\begin{aligned}
\Delta^d_{0\to 1} &= \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(2\theta_{kd}-1) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(2\theta_{kd}-1) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} \\
&\ge \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(2\theta_{kd}-1)\beta_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(2\theta_{kd}-1) \prod_{i\ne d} {\theta_{ki}}^{x_i}(1-\theta_{ki})^{1-x_i} \\
&\ge \sum_{k:\theta_{kd}\le\frac{1}{2}} \pi_k(2\theta_{kd}-1)\beta_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(2\theta_{kd}-1)\alpha_{kd} \qquad \text{(B.4)}
\end{aligned}$$

Thus, the quantity:

$$\sum_{k:\theta_{kd}\leq\frac{1}{2}} \pi_k(2\theta_{kd}-1)\beta_{kd} + \sum_{k:\theta_{kd}>\frac{1}{2}} \pi_k(2\theta_{kd}-1)\alpha_{kd}$$

is a lower bound on $P(x_d = 1, x_{-d}|\pi, \theta) - P(x_d = 0, x_{-d}|\pi, \theta)$. If the above quantity is greater than 0 (corresponding to the inequality of Equation 4.3b), then the probability of x is always larger when x_d is set to 1. Thus, the value of x_d under the MAP instantiation must be 1.

Appendix C

Properties of the Dirichlet Process Prior in Section 5.2.2

In this appendix we explore the significance of the Dirichlet process hyperparameters γ and κ in Section 5.2.2 (specifically, in Equation 5.2), with respect to the expected segment length in the HMM under this prior. The transition probabilities between latent states in the HMM, after integrating out the transition matrix, become functions of γ and κ. When a new state is transitioned to in the HMM, the probability of staying in the new state is the ratio $\frac{\gamma}{\gamma+\kappa}$. The magnitude $\gamma + \kappa$ represents the prior certainty of this ratio.

Setting values for γ and κ places a prior expectation on the length of a segment. Let ν_x be the amount of time the HMM stays in the latent state x, before transitioning to a different state. Making use of Equation 5.2, the expected value for ν_x is

$$E[\nu_x|\gamma,\kappa] = \sum_{n=1}^{\infty} nP(\nu_k = n|\gamma,\kappa) = \sum_{n=1}^{\infty} n \left(\prod_{i=0}^{n-1} \frac{i+\gamma}{i+\gamma+\kappa} \right) \frac{\kappa}{n+\gamma+\kappa} \tag{C.1}$$

assuming that the time series is infinite.

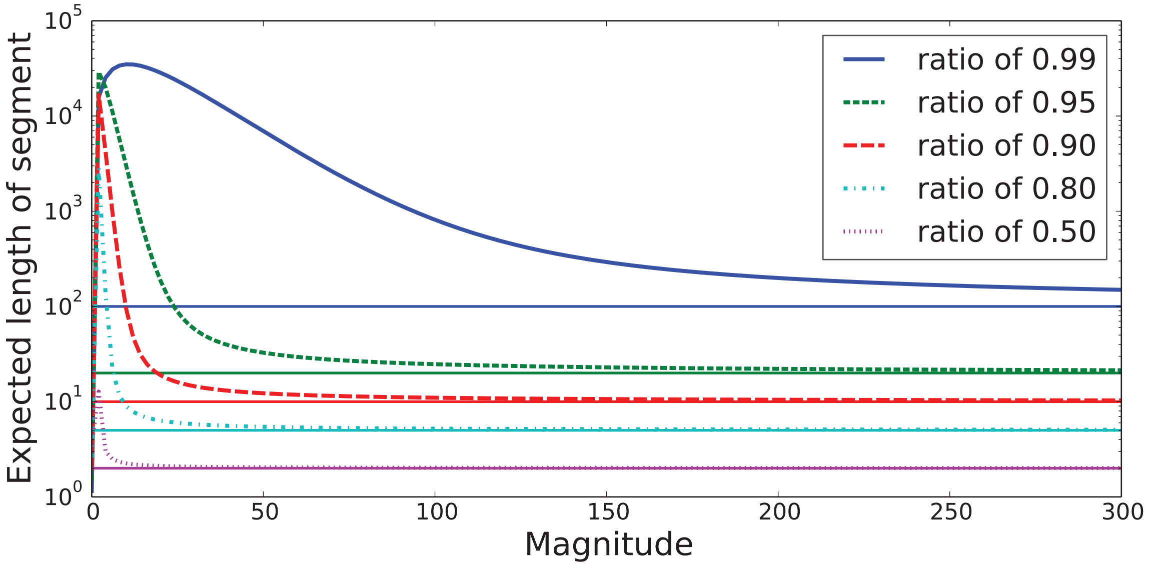


Figure C.1: Expected segment length under the prior for various values of ratio and magnitude. Solid horizontal lines indicate expected values under a Geometric($\frac{\kappa}{\gamma+\kappa}$) distribution.

Figure C.1 shows the expected segment length as a function of the ratio $\frac{\gamma}{\gamma+\kappa}$, for different values of magnitude $\gamma+\kappa$. The expectation in Equation C.1 is approximated by truncating the summation to $n = 1000000$.

When the magnitude $\gamma+\kappa$ is large, the value of V_t in Equation 5.2 becomes negligible and the expected time in a segment approaches the expected value for a Geometric($\frac{\kappa}{\gamma+\kappa}$) distribution (as in a traditional HMM). When the magnitude is small, the expected value becomes much larger, as a single self-transition will make the probability of leaving that state according to Equation 5.2 approximately zero.

To illustrate the behavior of the prior under various hyperparameter settings, in Figure C.2 we show the posterior probability of a changepoint under different values of γ and κ for the simulated email user in Section 5.5. Different configurations of magnitude and ratio were considered. As the ratio becomes closer to 1, the prior is strong and favors large segments, regardless of the magnitude. This can be seen by the few peaks in changepoint probability when $r = 0.999$.

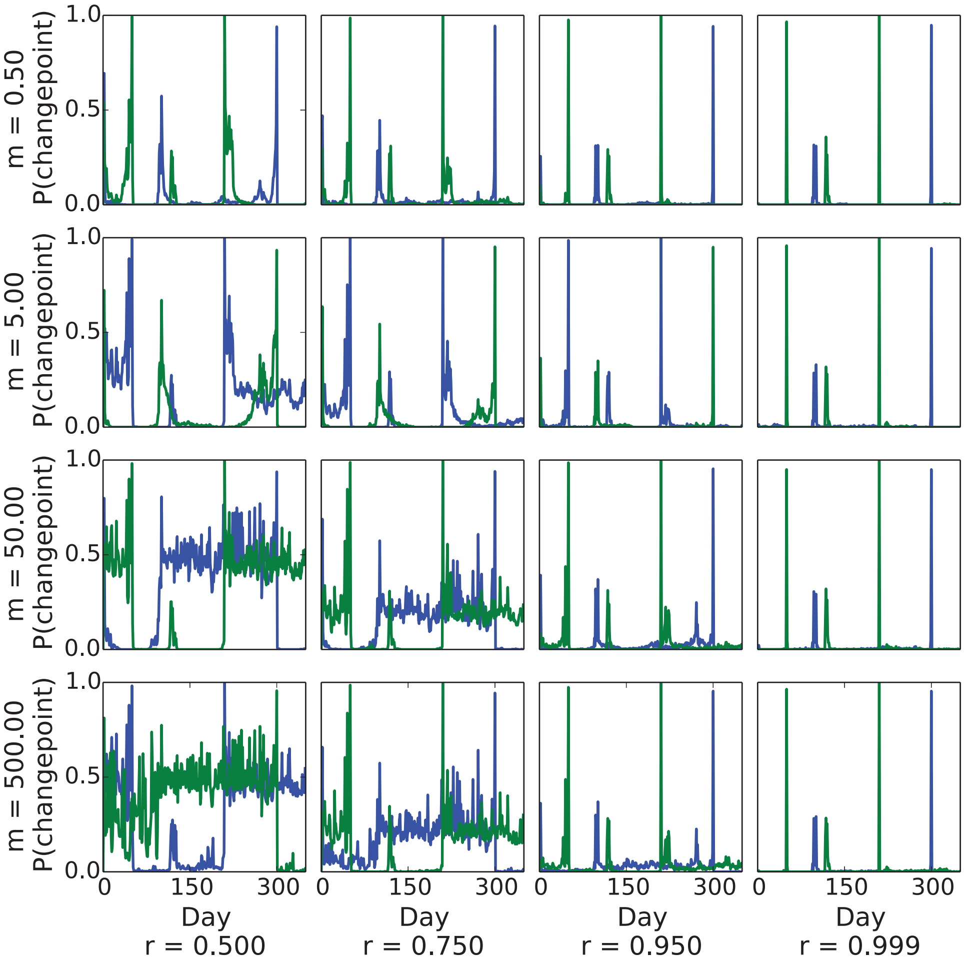


Figure C.2: Posterior probability of a changepoint for the simulated email user in Section 5.5, using various values of hyperparameter ratio and magnitude.

Figure C.2 also illustrates that as the ratio becomes smaller, the magnitude becomes increasingly important. When the magnitude is large, the a priori expected length of the segment becomes $\frac{r}{1-r}$. As a result, smaller segments are found. Evidence for this is seen in the significant probability of a changepoint across most days in the bottom-left plots in Figure C.2. When the magnitude is small, the prior favors large segments regardless of the ratio, resulting in fewer peaks in the posterior distributions of changepoints.

Appendix D

Derivations of Sampling Equations in Section 5.4

In this appendix we briefly derive the sampling equations used for Gibbs sampling the model parameters, as described in Section 5.4.

Sampling Latent Group Indicators z

$z_{t,n}$, the group assignment for email n on day t, is sampled from the discrete distribution created when normalizing $P(z_{t,n} = k|\boldsymbol{x}, \Omega \setminus z_{t,n})$ for all k. According to the graphical model, this distribution can be written as

$$\begin{aligned}
P(z_{t,n}|\boldsymbol{x}, \Omega \setminus z_{t,n}) &\propto P(\boldsymbol{z}, \boldsymbol{x}|\Omega \setminus \{\boldsymbol{z}, \boldsymbol{x}\}) \\
&= P(\boldsymbol{z}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \int P(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{\theta}|\alpha^{(z)}, \beta^{(z)}) d\boldsymbol{\theta}.
\end{aligned}$$

The term $P(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\theta})$ is equal to

$$\begin{aligned} P(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\theta}) &= \prod_{t=1}^{T}\prod_{n=1}^{N_t}\prod_{r=1}^{R}\theta_{z_{t,n},r}^{x_{t,n,r}}(1-\theta_{z_{t,n},r})^{1-x_{t,n,r}} \\ &= \prod_{k=1}^{K}\prod_{r=1}^{R}\theta_{k,r}^{c_{1,k,r}}(1-\theta_{k,r})^{c_{0,k,r}}, \end{aligned}$$

where $c_{1,k,r}$ and $c_{0,k,r}$ are the number of emails sent to group k that recipient r does and does not appear in, respectively. Combining this with the prior term $P(\boldsymbol{\theta}|\alpha^{(z)},\beta^{(z)})$ and moving the products outside the integral, we get

$$\begin{aligned} P(z_{t,n}|\boldsymbol{x},\Omega\setminus z_{t,n}) &\propto P(z_{t,n}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s})\prod_{k=1}^{K}\prod_{r=1}^{R}\int\theta_{k,r}^{c_{1,k,r}+\alpha^{(z)}-1}(1-\theta_{k,r})^{c_{0,k,r}+\beta^{(z)}-1}d\theta \\ &= P(z_{t,n}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s})\prod_{k=1}^{K}\prod_{r=1}^{R}\frac{\Gamma(c_{1,k,r}+\alpha^{(z)})\Gamma(c_{0,k,r}+\beta^{(z)})}{\Gamma(c_{1,k,r}+\alpha^{(z)}+c_{0,k,r}+\beta^{(z)})} \qquad \text{(D.1)} \end{aligned}$$

where the equality arises as the integral takes the form of an unnormalized Beta distribution with parameters $c_{1,k,r}+\alpha^{(z)}$ and $c_{0,k,r}+\beta^{(z)}$.

The terms contained in the second product, for $k=z_{t,n}$, can be rewritten as

$$\prod_{r=1}^{R}\frac{\Gamma(c_{1,z_{t,n},r}^{-(t,n)}+x_{t,n,r}+\alpha^{(z)})\Gamma(c_{0,z_{t,n},r}^{-(t,n)}+(1-x_{t,n,r})+\beta^{(z)})}{\Gamma(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)}+c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)}+1)}.$$

where $c_{1,k,r}^{-(t,n)}$ is equivalent to $c_{1,k,r}$, but ignoring email n on day t. Similarly for $c_{0,k,r}^{-(t,n)}$. Using the property $\Gamma(x+1)=x\Gamma(x)$, this quantity is equal to

$$\prod_{r=1}^{R}\left(\frac{\Gamma(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)})\Gamma(c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)})}{\Gamma(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)}+c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)})}\right)\left(\frac{(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)})^{x_{t,n,r}}(c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)})^{(1-x_{t,n,r})}}{c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)}+c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)}}\right).$$

Plugging this value back into Equation D.1, we get

$$\begin{aligned}
P(z_{t,n}|\boldsymbol{x},\Omega\setminus z_{t,n}) &\propto P(z_{t,n}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s})\left(\prod_{k=1}^{K}\prod_{r=1}^{R}\frac{\Gamma(c_{1,k,r}^{-(t,n)}+\alpha^{(z)})\Gamma(c_{0,k,r}^{-(t,n)}+\beta^{(z)})}{\Gamma(c_{1,k,r}^{-(t,n)}+\alpha^{(z)}+c_{0,k,r}^{-(t,n)}+\beta^{(z)})}\right)\\
&\quad\times\left(\prod_{k=1}^{K}\prod_{r=1}^{R}\frac{(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)})^{x_{t,n,r}}(c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)})^{(1-x_{t,n,r})}}{c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)}+c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)}}\right)\\
&\propto \lambda_{z_{t,n},t}\left(\prod_{r=1}^{R}\frac{(c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)})^{x_{t,n,r}}(c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)})^{(1-x_{t,n,r})}}{c_{1,z_{t,n},r}^{-(t,n)}+\alpha^{(z)}+c_{0,z_{t,n},r}^{-(t,n)}+\beta^{(z)}}\right)
\end{aligned}$$

where $P(z_{t,n}|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s})$ is calculated using Equation 5.5.

Sampling Day-of-Week Communication Rate Parameters α

In order to sample α_w, the rate parameter with respect to day-of-week w, the log of the unnormalized conditional distribution is required. The distribution can be written as

$$\begin{aligned}
P(\alpha_w|\boldsymbol{x},\Omega\setminus\alpha_w) &\propto P(\alpha_w,\boldsymbol{N}|\Omega\setminus\{\alpha_w,\boldsymbol{N}\})\\
&\propto P(\alpha_w|\mu,\sigma^2)\prod_{t:d(t)=w}P(N_t|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s}),
\end{aligned}$$

where $d(t)$ is the day-of-week for day t. From Section 5.2, we have

$$N_t|\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{s}\sim\text{Poisson}\left(\sum_{k=1}^{K}\lambda_{k,t}\right)=\text{Poisson}\left(e^{\alpha_w}\sum_{k=1}^{K}e^{\beta_{k,s_{k,t}}}\right).$$

Taking the log of this unnormalized conditional distribution, we get

$$\begin{aligned}
&\log P(\alpha_w|\boldsymbol{x}, \Omega \setminus \alpha_w) \\
&\quad \propto -\frac{(\alpha_w - \mu)^2}{2\sigma^2} + \sum_{t:d(t)=w} \left[N_t \left(\alpha_w + \log \left(\sum_{k=1}^{K} e^{\beta_{k,s_{k,t}}} \right) \right) - e^{\alpha_w} \left(\sum_{k=1}^{K} e^{\beta_{k,s_{k,t}}} \right) \right] \\
&\quad \propto -\frac{(\alpha_w - \mu)^2}{2\sigma^2} + \alpha_w \sum_{t:d(t)=w} N_t - e^{\alpha_w} \left(\sum_{t:d(t)=w} \sum_{k=1}^{K} e^{\beta_{k,s_{k,t}}} \right)
\end{aligned}$$

Sampling Group Communication Rate Parameters β

As with α, the log of the unnormalized conditional distribution is needed for sampling. This distribution for $\beta_{k,m}$, the rate parameter for group k while in segment m, is

$$\begin{aligned}
P(\beta_{k,m}|\boldsymbol{x}, \Omega \setminus \beta_{k,m}) &\propto P(\beta_{k,m}, \boldsymbol{N}, \boldsymbol{z}|\Omega \setminus \{\beta_{k,m}, \boldsymbol{N}, \boldsymbol{z}\}) \\
&\propto P(\beta_{k,m}|\mu, \sigma^2) \prod_{t:s_{k,t}=m} P(N_t|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \prod_{t:s_{k,t}=m} \prod_{n=1}^{N_t} P(z_{t,n}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}).
\end{aligned}$$

Using Equation 5.5 for $P(z_{t,n}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s})$, substituting $\boldsymbol{\lambda}$ for $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}\}$, and taking the log of this unnormalized conditional distribution, we get

$$\begin{aligned}
\log P(\beta_{k,m}|\boldsymbol{x}, \Omega \setminus \beta_{k,m}) &\propto -\frac{(\beta_{k,m} - \mu)^2}{2\sigma^2} + \sum_{t:s_{k,t}=m} \left[N_t \log \left(\sum_{k'=1}^{K} \lambda_{k',t} \right) - \sum_{k'=1}^{K} \lambda_{k',t} \right] \\
&\qquad + \sum_{t:s_{k,t}=m} \left(\sum_{n=1}^{N_t} \log(\lambda_{z_{t,n},t}) - N_t \log \left(\sum_{k'=1}^{K} \lambda_{k',t} \right) \right) \\
&\propto -\frac{(\beta_{k,m} - \mu)^2}{2\sigma^2} - \sum_{t:s_{k,t}=m} \lambda_{k,t} + \sum_{t:s_{k,t}=m} \sum_{n=1}^{N_t} \log(\lambda_{z_{t,n},t}).
\end{aligned}$$

Making use of the form of $\boldsymbol{\lambda}$ using Equation 5.3, we get

$$\begin{aligned}&\log P(\beta_{k,m}|\boldsymbol{x}, \Omega \setminus \beta_{k,m}) \\ &\quad\propto -\frac{(\beta_{k,m}-\mu)^2}{2\sigma^2} - \sum_{t:s_{k,t}=m} e^{\alpha_{d(t)}} e^{\beta_{k,m}} + \Big|\{t, n : s_{k,t} = m, z_{t,n} = k\}\Big|\beta_{k,m}\end{aligned}$$

Sampling HMM Latent States s

Because of the HMM restrictions described in Section 5.2, the Gibbs sampler only samples the HMM latent state $s_{k,t}$ if $s_{k,t-1} \neq s_{k,t+1}$. When a new value for $s_{k,t}$ is sampled, its possible values are $s_{k,t-1}$, $s_{k,t+1}$, or a brand new state. Thus, a sample of $s_{k,t}$ is produced from the discrete distribution defined by the normalized probabilities of these outcomes. The unnormalized conditional probability for $s_{k,t}$ is

$$\begin{aligned}P(s_{k,t}|\boldsymbol{x}, \Omega \setminus s_{k,t}) &\propto P(\boldsymbol{s}, \boldsymbol{z}, \boldsymbol{N}|\Omega \setminus \{\boldsymbol{s}, \boldsymbol{z}, \boldsymbol{N}\}) \\ &\propto P(N_t|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \prod_{n=1}^{N_t} P(z_{t,n}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{s}) \prod_{t'=t}^{g(t)} P(s_{k,t'}|\{s_{k,u} : u < t'\}, \boldsymbol{\gamma}, \boldsymbol{\kappa}),\end{aligned}$$

where $g(t) \equiv \min(\{t' : t' \geq t+1, s_{k,t'} \neq s_{k,t+1}\})$, the starting time of the segment following the one $s_{k,t+1}$ is in. Using Equation 5.2, the last product can be written as

$$\prod_{t'=t}^{g(t)} \frac{(V_{t'}+\gamma)^{\delta\left(s_{k,t'}=s_{k,t'-1}\right)} \kappa^{\delta\left(s_{k,t'}\neq s_{k,t'-1}\right)}}{V_{t'}+\gamma+\kappa}.$$

The new value of $s_{k,t}$ is sampled by first calculating the unnormalized probabilities above for each value of $s_{k,t}$ (note that V_t will change based on the value of $s_{k,t}$ under consideration), then normalizing and sampling from the corresponding discrete distribution.