LATENT DIRICHLET ALLOCATION FOR NATURAL LANGUAGE STATISTICS

by

Thomas W. Jones
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computational Sciences and Informatics

Committee:

_____	Dr. Jason Kinser, Dissertation Director
_____	Dr. William G. Kennedy, Committee Member
_____	Dr. Olga Gkountouna, Committee Member
_____	Dr. Carlotta Domeniconi, Committee Member
_____	Dr. Mark Meyers, Committee Member
_____	Dr. Jason Kinser, Department Chairperson
_____	Dr. Donna M. Fox, Associate Dean Office of Student Affairs & Special Programs, College of Science
_____	Dr. Fernando R. Miralles-Wilhelm Dean, College of Science
Date: _____	Spring Semester 2023 George Mason University Fairfax, VA

Latent Dirichlet Allocation for Natural Language Statistics

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Thomas W. Jones
Master of Science
Georgetown University, 2012
Bachelor of Arts
The College of William and Mary, 2009

Director: Jason Kinser, Chair
Department of Computational and Data Sciences

Spring Semester 2023
George Mason University
Fairfax, Virginia

Dedications

To my loving mother, who sadly passed away before she could see this moment. Your love and dedication shaped me into the man I am today. For that I am eternally grateful.

To my loving father. You are the smartest man that I know. Your hard work and dedication to your career have served me as an example and a curse. May I some day be fit to fill your shoes.

Acknowledgments

To Bill Kennedy, your guidance and advice have been invaluable. I wouldn't be here without you. Thank you.

To Mark Meyer, your thoughtful comments have kept my research rigorous and grounded in statistical best practices. Thank you.

To Jason Kinser, thank you for being my dissertation director and keeping this committee (and our department) running.

To my remaining committee members, Carlotta Domeniconi and Olga Gkountouna, thank you for your time and thoughtful comments.

To Alyson Wilson and Will Doane, from STPI to here! Thank you for all of your coaching and guidance over the years.

To In-Q-Tel, who has supported my resarch through tuition assistance and work flexibility, thank you.

To the members of ASA's Section on Text Analysis, thank you for forging a path to bring analyses of language to the statistical mainstream. I am eager to see what a collection of motivated statisticians can contribute to the field.

Table of Contents

List of Tables

List of Figures

Abstract

LATENT DIRICHLET ALLOCATION FOR NATURAL LANGUAGE STATISTICS

Thomas W. Jones, PhD.

George Mason University, 2023

Dissertation Director: Dr. Jason Kinser

Written language is one of the most information rich and abundant data sources available. Rigorous statistical study of linguistic phenomena can elevate the digital humanities, linguistic applications to computational social science, and statistics itself. As yet, statistical applications to language, whether in linguistics, computation, or otherwise, are largely *ad hoc*. Performance gains in modeling have largely been due to two factors: fine tuning transfer learning and increasing the size and complexity of models. Due to the statistical nature of human language—governed by several power law phenomena—fine tuning transfer learning may be advantageous for corpus analyses, not just artificial intelligence applications. Yet state of the art "transformer" models are expensive and opaque. I propose we revisit Latent Dirichlet Allocation (LDA). As a parametric statistical model of a data generating process, it has the potential to be used in statistically rigorous ways to study written language. In this dissertation, I state my case for what I refer to as "natural language statistics", a more statistically rigorous take on analyzing text data. I link the data generating process for LDA to Zipf's law and use that relationship to engage in simulation studies for LDA. I also develop a coefficient of determination for topic models, extend LDA to enable pre-train/fine tuning transfer learning, and implement this research and more in an R package, *tidylda*.

Chapter 1: Introduction and Motivation

Written language is one of the most information rich sources of data that exists. Language is literally the medium humans use to communicate information to each other. And in an increasingly digitally connected world, the amount of text available for analysis has exploded. Improvements in computing power and algorithmic advances have driven staggering progress in machine learning tasks for natural language, including machine translation, question answering, automatic summarization, information extraction and more.

Such advances have lead to an increase in textual analysis in two relatively new, interdisciplinary fields—the "*digital humanities*" and "*computational social science*." The digital humanities represent a quantification of traditionally qualitative fields such as history, literature, communications, and the arts. Computational social science emphasizes computationally-intensive methods for disciplines including economics, political science, and psychology. Examples of textual analyses from these two new fields include statistical modeling of language in the *Pennsylvania Gazette* from 1728 to 1800 [1], tracking the evolution of the Greek word *kosmos* from 700 BC onward [2], using language from the Federal Reserve Board's public press releases to predict the Fed's non-public economic forecasts [3] and more. Applications to economics and social science were presented in recent workshops for the Association for Computational Linguistics [4] [5].

1.1 Natural Language Statistics

The tools used for text analyses in these new fields derive from linguistics, computing, and statistics. Linguistics includes the sub-field of *corpus linguistics*, the study of language as it appears in samples of "real world" text (corpora). Computing has the sub-field of *natural language processing* (NLP) that concerns itself with the interactions of people and computers. A particularly relevant component of NLP is *distributional semantics*, which quantifies semantic similarities in language in terms of relative frequencies of linguistic items (such as words). Yet in statistics no named sub-field exists.

Statistics as a field is waking up to its role regarding linguistic data, however. The American Statistical Association (ASA) recently formed an interest group for text analysis in 2019 and elevated it a full section

in 2022[1] [6]. A year later, the Joint Statistics Meetings—the annual conference of the ASA and 11 other statistical associations—had over 20 sessions featuring text analysis research [7]. This isn't to say the interest group caused the volume of text analysis research. Note that many of the references in this dissertation publish in statistics journals—such research is happening anyway. But only recently has there been effort to organize a community of practice for text analyses under a statistics umbrella.

Rigorous statistical study of linguistic phenomena can elevate the digital humanities, linguistic applications to computational social science, and statistics itself. As yet, statistical applications to language, whether in linguistics, computation, or otherwise, are largely ad hoc. Save a handful of empirical laws, there is little statistical theory guiding the modeling of textual data. What theory does exist generally does not inform specification or use of statistical or machine learning models of text. Instead, the field has relied on increasingly complex models, requiring tremendous computational power, to drive these advances.

To envision the art of the possible, consider the state of statistical theory in linear regression. Linear regression—and its related statistical theory—dates back to the early 1800's from the works of Legendre [8], Gauss [9], and Galton [10]. We have formal statements of the assumptions required for valid statistical inference using linear regression. We have multitudes of diagnostic statistics to assess the degree these assumptions are violated in the data or with model specifications. There are a plethora of remediations to apply when key assumptions are violated so that valid statistical inference may still be made. And there are an abundance of statistical inferential methods built on top of regression models used to detect structural breaks [11], calculate individual variable's contribution to the coefficient of determination [12], and on and on. We have been studying linear regression for a long time. As a result, we have an extremely powerful kit of statistical tools that are so easy to use that in 2020, we mostly take them for granted. Imagine what we could do if we brought this level of rigor to models and applications using language data in all of its abundance.

Again, statistics does not have a named sub-field dedicated to such study. Since naming a concept can give it power, I humbly submit "*Natural Language Statistics*", so named because it is distinct from task-based Natural Language Processing. Natural Language Statistics may focus on "traditional" statistical concerns such as constructing appropriate random samples, quantifying uncertainty about claims learned from data,

[1] I have been involved since the beginning and am the group's web master.

developing rigorous evaluation metrics for models, and so on where language is the topic of study. Natural Language Statistics can build off of work already begun in linguistics, machine learning, and complexity theory which concerns itself with the study of complex phenomena and power laws.

Language is saturated with the statistics of power laws. Two empirical laws of language, Zipf's law and Heaps's law—covered in more detail later in this dissertation—are two manifestations of power laws in language data. Power laws are extreme valued distributions, with significant mass in their tails and therefore have extremely large variance. In fact, for many parameterizations, the second moment is undefined, meaning the variance tends towards infinity [13, Ch. 3]. Because of power laws in language, the law of large numbers operates very slowly and very large samples are required to reliably estimate population parameters [13, Ch. 8]. As a consequence, most corpora (samples of language) are missing key information.

Power laws in language might imply, then, that one needs external information for a thorough analysis of any one corpus. In fact, this is actually how humans learn from language. Consider the following thought experiment where one wants to learn about chemistry from an English-language textbook. Presumably, this person has good grasp on the English language already, having a vocabulary and covering subject matter greater than the book itself contains. This person then reads the textbook, learns about chemistry, and in the process updates and expands their knowledge of the English language.[2] In addition to traditional statistical concerns like representative samples and reasonable model specification, Natural Language Statistics needs pre-trained models of language to be updated for analyzing finite corpora.

1.2 Fine Tuning Transfer Learning

In machine learning, starting with a base model and then updating its parameters with a new sample is called the *fine tuning paradigm of transfer learning*. Transfer learning is when a model is developed for one task—on one data set—and then re-used for another task and data set [14]. In the fine tuning paradigm, the base model is modified for the new task, allowed to update based on the new data, or both modified to the task and updated with new data [15].

Neural networks lend themselves naturally to fine tuning transfer learning. They are trained (or fit)

[2]This sounds philosophically Bayesian to me. Yet I see no reason why one needs to limit their study to the use of Bayesian statistical models.

using the iterative back propagation algorithm. Instead of initializing the model parameters—often called "weights"—at random, they are initialized at the same values they had in the base model. Then back propagation is resumed using the new data and all or some of the weights are allowed to update. This lets the analyst leverage information from a very large data set encoded in the base model and adjust the parameters to fold in information in their new data set. Fine tuning transfer learning has been used for many years in deep learning models for computer vision, but it has recently gained widespread adoption in deep learning models for language.

Current state of the art natural language processing models belong to a class of deep neural networks called "transformers."[3] Famous examples of transformers include BERT [15], XLM-R [17], GPT-2 [18], GPT-3 [19], and more. In terms of raw accuracy for benchmark task-specific objectives, transformer models outperform other models in Natural Language Processing [20]. For example, transformers have made improvements in text classification [21], machine translation [22], coreference resolution [23], and summarization [24] among others.

Transformers are extremely accurate on pre-defined natural language processing tasks, but they are not without problems. First, and most famously, these models are huge and expensive. BERT has 110 million parameters, XLM-R has 550 million parameters, and GPT-3 has a whopping 175 billion parameters.[4] These base models can cost from hundreds to tens-of-thousands of dollars in compute costs for a single run [25]. GPT-3 is rumored to have cost $4.6 million in cloud computing to train it [26]. These figures exclude trial and error runs inherent in any model development process.

Second, the data sets used to build these base models affect results, but we don't know what the "right" data set looks like. It is not controversial to assert that models inherit biases from the data on which they are developed. Yet some of the issues in these large language models are particularly jarring. They can encode—then reproduce—racist or otherwise extreme language [27]. And they may exclude language we wish was included, perhaps those of underrepresented groups. Yet the data sets used to construct transformers are so large, researchers cannot truly audit what they do or do not include. And the investment required to construct large language models comes at an opportunity cost of learning how to strategically construct data

[3] Transformers get their name from the "transformer" sub-architecture for deep neural networks [16]. Technically, this is distinct from fine tuning transfer learning. But as of this writing, the two approaches are used hand-in-hand for natural language processing models.

[4] If each parameter is stored as a 4 byte float, then GPT-3 is 700 Gb on disk, larger than most data sets.

sets without these downsides [27].[5]

Third, transformers are built for supervised tasks for artificial intelligence, not corpus analysis. Mostly these are sequence to sequence models used for question answering, machine translation, text generation, document summarization, and so on. While some of these tasks may be useful for corpus analysis, they really are distinct from a statistical analysis of a corpus.

Fourth, and most obviously, transformers are deep neural networks. Deep neural networks are spectacular in their flexibility and accuracy for many supervised learning tasks. Yet this flexibility and accuracy has come at the cost of complexity, often antithetical to understanding.

In summary, the fine-tuning approach of modeling would be beneficial to develop for Natural Language Statistics. It reflects how we intuitively understand that humans use language to learn themselves. Yet we would need simpler, more transparent models upon which to apply and build on statistical theory. Transformers are too big, complex, and opaque for this task. Boyd-Graber and Mimno make this point explicitly on p. 116 of *Applications of Topic Models*, "Deep learning has a reputation for inscrutable parameters but state-of-the-art performance. One of the strengths of probabilistic models is their interpretability and grounded generative processes" [28].

1.3 Summary of Research

I reexamine a model that has become less popular in machine learning circles, Latent Dirichlet Allocation (LDA) [29]. Despite this and with the above comments in mind, LDA has some desirable properties. It models a data generating process which may be linked to the empirical laws of language. This property makes LDA, and related models, candidates for helping to develop a more robust statistical theory for modeling language. And while LDA may be less popular at the cutting edge of machine learning, it and its variants are still popular in fields such as computational social science [30] and the digital humanities [31].

This dissertation makes four intellectual contributions: three studies and one software library. The first study investigates the data generating process modeled by LDA, linking it to empirical language laws and using it for simulation studies concerned with model specification. The second introduces a new (yet old)

[5] To me, this sounds like a task that the statistics community is well suited for, adapting sampling theory and statistical design for use in constructing data sets of language.

evaluation metric for topic models: a generalized coefficient of determination. The third study introduces a method for fine tuning transfer learning applicable to MCMC algorithms for LDA. Finally, this sort of research is not practical if people cannot use it. Therefore, I have developed a software package for the R language [32] that draws on this research and a framework known as the "tidyverse" [33] to make a principled, flexible, performant, and user-friendly interface for training and using LDA models. This package is called *tidylda*.

Chapter 2: Latent Dirichlet Allocation and Related Models

2.1 Vector Space Model

Most modern approaches to language modeling can trace their origin to the vector space model [34]. The vector space model represents contexts (usually documents) in multidimensional space whose coordinates are given by the frequencies of words within that context. These frequencies may be explicit counts, or they may be re-weighted. The vector space model makes the *bag of words* assumption. Within a context, word order and proximity have no meaning. When one says a document is a "bag of words", they mean that word order is discarded and the document is only the relative frequencies of words within it. More formally, let $\boldsymbol{X}$ be a $D \times V$ matrix of contexts. Each row represents a context, and each column a word.[1] The key to the vector space model is that if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close in the vector space given by $\boldsymbol{X}$, then the contexts represented by $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ must be semantically similar.

The key limitation of the vector space model is that $\boldsymbol{X}$ is large and sparse. This leads to the "curse of dimensionality" where meaningful comparisons can be different because most dimensions have little signal [35]. The bag of words assumption also disregards the use of synonyms or the fact that the same words can have multiple meanings—known as "polysemy".

2.2 Latent Semantic Indexing

Latent Semantic Indexing (LSI) reduces the dimensions of $\boldsymbol{X}$ through a singular value decomposition (SVD) [36]. Since $\boldsymbol{X}$ is large and sparse, one can approximate it by $\boldsymbol{X}_{(K)}$ which is of rank K and corresponds to the K largest eigenvalues of $\boldsymbol{X}$. This projects the data matrix $\boldsymbol{X}$ into a K-dimensional Euclidean "semantic" space. Formally,

$$\boldsymbol{X} \approx \boldsymbol{X}_{(K)} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{2.1}$$

[1] While there are distinct differences in the definitions of "word" and "token", for the purposes of this work I will use the two terms interchangeably for simplicity.

where $\boldsymbol{U}$ and $\boldsymbol{V}^T$ are orthonormal matrices and $\boldsymbol{\Sigma}$ is a diagonal matrix of singular values. New documents, $\boldsymbol{X}'$ may be embedded by post multiplying them by the projection matrix, $\boldsymbol{\Lambda} = [\boldsymbol{\Sigma}\boldsymbol{V}^T]^{-1}$.

A key limitation of LSI is that there is no obvious way to choose K, the embedding dimension. This problem plagues nearly all models that follow it.

2.3 Probabilistic Latent Semantic Indexing

Probabilistic Latent Semantic Indexing (pLSI) brings a probabilistic approach to LSI [37]. pLSI models a generative process for $\boldsymbol{X}$ where $\boldsymbol{X}$ is explicitly a matrix of integer counts of word occurrences. Assuming there are D contexts, K topics, V unique words, N total words and N_d words in the d-th document, the process is as follows. For each word, n, in context d:

1. Draw topic $z_{d,n}$ from Multinomial($\boldsymbol{\theta}_d$)
2. Draw word $w_{d,n}$ from Multinomial($\boldsymbol{\beta}_{z_{d,n}}$)
3. Repeat 1. and 2. N_d times.

From the above, it's clear that $P(z_k|d) = \theta_{d,k}$ and $P(w_v|z_k) = \beta_{k,v}$.

pLSI is fit using the EM algorithm to find the values of parameters in $\boldsymbol{\Theta}$ and $\boldsymbol{B}$ that maximize the joint likelihood of $\boldsymbol{X}$. The likelihood of a single word in a single document is $P(w_v, d) = P(d)P(w_v|d)$. Taking $P(d) = \frac{N_d}{N}$ and noting that $P(w_v|d) = \sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,v}$ we can derive the joint likelihood

$$\mathscr{L}(\boldsymbol{\Theta}, \boldsymbol{B}|\boldsymbol{X}) = \prod_{d=1}^{D}\prod_{v=1}^{V}\left(\frac{N_d}{N}\sum_{k=1}^{K}\theta_{d,k}\cdot\beta_{k,v}\right)^{x_{d,v}} \tag{2.2}$$

It is stated that pLSI cannot be extended to unseen contexts. "It is not clear how to assign probability to a document outside of the training set" [29]. However, one can use Bayes's rule to derive a projection matrix, $\boldsymbol{\Lambda}$, to embed new contexts into the probability space fit by pLSI. A derivation is in Appendix A. pLSI is alleged to habitually over fit its training data [38]. And, as with LSI, there's no clear guidance for selecting the number of topics, K.

2.4 Latent Dirichlet Allocation

LDA is a Bayesian version of pLSI and was developed by David Blei, Andrew Ng, and Michael Jordan to address perceived shortcomings of pLSI [29]. LDA adds Dirichlet priors to the parameters $\boldsymbol{\Theta}$ and $\boldsymbol{B}$. This modifies the data generating process (DGP) as

1. Generate $\boldsymbol{B}$ by sampling K topics $\beta_k \sim \text{Dirichlet}(\boldsymbol{\eta}), \forall k \in \{1,2,...,K\}$
2. Generate $\boldsymbol{\Theta}$ by sampling D documents $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha}), \forall d \in \{1,2,...,D\}$
3. Then for each document, d
    a. Draw topic $z_{d,n}$ from Multinomial$(\boldsymbol{\theta}_d)$
    b. Draw word $w_{d,n}$ from Multinomial$(\beta_{z_{d,n}})$
    c. Repeat a. and b. N_d times.

For ease of notation, I refer to the above process as the LDA-DGP throughout this dissertation.[2]

The above process has a joint posterior of

$$P(\mathbf{W},\mathbf{Z},\boldsymbol{\Theta},\boldsymbol{B}|\boldsymbol{\alpha},\boldsymbol{\eta}) \propto \left[\prod_{d=1}^{D}\prod_{n=1}^{n_d} P(w_{d,n}|\beta_{z_{d,n}})P(z_{d,n}|\boldsymbol{\theta}_d)P(\boldsymbol{\theta}_d|\boldsymbol{\alpha})\right]\left[\prod_{k=1}^{K} P(\beta_k|\boldsymbol{\eta})\right] \tag{2.3}$$

The above posterior does not have an analytical closed form. Consequently, a variety of Bayesian estimation methods have been employed to estimate the model. Blei et. al used variational expectation maximization (VEM). Shortly thereafter, Griffiths and Steyvers developed a collapsed Gibbs sampler for LDA [39]. This sampler is "collapsed" in that the parameters of interest—$\boldsymbol{\Theta}$ and $\boldsymbol{B}$—are integrated out for faster computation. After iteration is complete, $\boldsymbol{\Theta}$ and $\boldsymbol{B}$ can be easily calculated. (See Appendix B for the collapsed Gibbs algorithm and posterior calculation.) Others have since developed many other MCMC algorithms, discussed more in the next section.

The priors $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ may be either asymmetric or symmetric. Asymmetric priors are those where element of the vector hyper parameter (e.g., $\boldsymbol{\alpha}$) is a different value. Symmetric priors are those where each element of the vector hyper parameter is same value. In the latter case, researchers will often represent the hyper

[2]The original specification of the LDA-DGP [29] specified that each document's length is drawn from a Poisson random variable. This specification has been dropped from most subsequent work on LDA. Even so, I use the Poisson distribution in Section 3 to create synthetic data sets. However, there is no reason to believe that context lengths have a Poisson distribution in real world corpora.

parameter as a scalar rather than a vector. The magnitudes of $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ affect the average concentration of topics within documents and words within topics, respectively. This is a property of the Dirichlet Distribution.

2.4.1 Persistent Issues with LDA (and Most Other Topic Models)

In spite of its being almost 20 years old, LDA has several persistent issues. Many of these are shared by the models that came before and after. Given its continued popularity it is worth further discussing its limitations. The critical issues relate to model specification, evaluation, algorithmic scalability, and LDA's strong assumptions of independence.

First, there is no generally accepted method for choosing hyper parameters—i.e., model specification—for an LDA model [40]. Anecdotally, the most concern is in choosing K, the number of topics.[3] There has been less exploration in choosing the prior parameters, $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. Yet, I have some preliminary evidence that all three parameters need to work together in concert, at least to get a semantically coherent model [41]. A rigorous examination of how all three hyper parameters interact in the LDA-DGP is warranted.

There are two philosophies for selecting K, the number of topics or dimensions of embedding for a topic model. The first is to select a value—perhaps including different values for $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ as well—to optimize a metric. This metric may be the log likelihood [39], perplexity [40], a coherence metric[4] [46], or something else [47]. Chen et al. use a fully-Bayesian approach and put a prior on the number of topics [48]. The second philosophy holds that optimizing for such metrics leads to topics that are too specific for human interpretability. Chang and Boyd-Graber constructed a topic model on a general news corpus, then had people using Amazon's Mechanical Turk evaluate topic quality [49]. The subjects covered by the model were broad and the audience were not subject matter experts.

There has been less attention paid to selecting $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. A notable—and influential—exception is Wallach, Mimno, and McCallum's *Rethinking LDA: Why Priors Matter*. They argue that $\boldsymbol{\alpha}$ should be asymmetric and that $\boldsymbol{\eta}$ should be symmetric and small [50]. This is counter to a proof included in Appendix C, where I show that the expected term frequency of a collection of documents generated by the LDA-DGP is proportional to $\boldsymbol{\eta}$. Given Zipf's law [51]—described in more detail in Section 3—a symmetric $\boldsymbol{\eta}$ constitutes a prior that is

[3] A quick search of Stack Overflow will quickly reveal a large number of questions related to choosing the number of topics.

[4] Not all coherence [42] metrics are created equal. Evaluation of several coherence metrics has found considerable variation in correlation with human judgement [43], [44]. In particular, the "UMASS" metric [45] is widely used, but does not correlate highly with human judgement [43], [44].

impossible in any real corpus. More recently, George and Doss explore an MCMC approach to help select $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ [52]. However, they too assume a symmetric $\boldsymbol{\eta}$.

Second, there is still no consensus on how to evaluate topic models. Shi et al. divide approaches to evaluation into three categories: manual inspection, intrinsic evaluation, and extrinsic evaluation [53]. Manual inspection—such as the "intruder test" used in [49]—is subjective and labor intensive. Intrinsic evaluation evaluates the model directly against the data on which it was developed. This is measured in goodness of fit metrics such as log likelihood and perplexity, with a coherence metric[5], or other metrics—such as comparing to baseline distributions [54]. Intrinsic evaluation may be measured in-sample or out-of-sample. Extrinsic methods evaluate the performance of topic models against some external task, such as identifying document class. In both his dissertation [38] and related publication [53], Shi argues that comparing topic models to "gold standard" synthetic data is a more principled approach. This is correct, however the link between the data generating process and statistical laws of language is missing.

Third, algorithms to fit LDA models have scaleability issues. Gibbs sampling is a naturally sequential algorithm. When the data set is large, it can be prohibitively slow. Gibbs sampling scales quadratically with the number of documents, topics, vocabulary size, and total number of tokens. Newman et al. developed a distributed "approximate" Gibbs sampler for LDA [55]. It is implemented in MALLET and I breifly implemented it in a version of *tidylda* before subsequently removing it. Quality of the model—by any intrinsic evaluation metric—decreases when this approximated Gibbs sampling is used. This is due to the approximation voiding theoretical guarantees of convergence inherent to MCMC samplers, a point that Newman et al. concede [55]. VEM has more-easily distributed computations, yet anecdotally VEM gives less coherent topics than MCMC algorithms, especially on smaller data sets [56]. A host of other MCMC algorithms have been developed to try to scale LDA to very large corpora [57] [58] [59] [60]. One approach uses a variational autoencoder to approximate VEM in a neural network [61]. Even so, the most approachable implementations of LDA still use VEM or collapsed Gibbs.

Finally, the LDA-DGP makes strong independence assumptions. Human language is said to have a property called "burstiness" [62]. Burstiness is the idea that words cannot be independent draws from a dis-

[5]Coherence metrics are intrinsic measures that are designed to approximate rigorous manual inspection.

tribution since they occur in clusters. i.e., Seeing a word once, greatly increases the probability you will see it again. Mimno and Blei developed posterior predictive checks for LDA to detect the degree to which burstiness affects LDA [63].

2.5 Related Topic Models

Researchers have developed many topic models intended to improve upon LDA. This section covers some notable examples.

Teh et al. used a hierarchical Dirichlet process (HDP) to determine the number of topics automatically [64]. They show that HDP has performance advantages over LDA in terms of perplexity. Yet Chen et al. point out that HDP is computationally intensive, that it is actually a fundamentally different model from LDA, and that using it to select the number of topics is not well defined [48].

Dynamic topic models (DTM) add a time series component to topic modeling [65]. By incorporating date of publication, DTMs allow the linguistic mixture of topics to change dynamically over time. Consider an intuitive example: the amount of writing about a given topic changes over time, but also *how* writers write about that topic changes as well. DTMs require explicit metadata on when a context appeared.

Topic models are often used to embed contexts into a vector space to aid a supervised task downstream. Supervised LDA (sLDA) incorporates this outcome in the modeling process [66]. McAuliffe and Blei find that sLDA improves accuracy of the predictive task and gives improved topic quality over a two-step of LDA then using a separate supervised algorithm.

Two closely-related algorithms are the correlated topic model (CTM) [67] and structural topic model (STM) [68]. CTMs modify LDA by placing a log-normal prior on $\mathbf{\Theta}$ to allow modeling the correlations between documents. STMs extend CTMs by allowing the user to declare context-level co-variates, rather than simply trying to estimate them. If no such co-variates are provided, then STM collapses to CTM [68].

More recently, researchers have been using variational autoencoders—a form of artificial neural network—to estimate topic models [69] [70] [71]. Approaching topic modeling in this way has the advantage of using GPUs to scale computations. MCMC methods may theoretically scale similarly, but implementations are not as readily available as deep learning frameworks.

The above models—and many more—encompass a large landscape, but they are not so different. Each

of these models effectively embeds their contexts into a probability space. (See Section 2.7.) As a result, I hypothesize that much of the research in this dissertation should extend—or can be modified to extend—to these models.

2.6 Text Embeddings and Distributional Semantics

Word embeddings were developed in parallel with topic models, beginning with the neural language model of Bengio et al. [72]. The idea is to represent words as distributions embedded into a vector space such that proximity corresponds to semantic similarity. The correspondence of distance in vector space to semantic distance is known as *the distributional hypothesis* and forms the basis of distributional semantics [73, Ch. 14]. The distributional hypothesis states that "you shall know a word by the company it keeps" [74]. In other words, a word's meaning may be inferred from the context in which it appears.[6] Since 2003, word embeddings have been extended to cover larger linguistic units such as sentences and documents [75] and have taken on the more general moniker "text embeddings". Notable models include word2vec [76], GloVe [77], ELMo [78], and more.

One popular method for constructing this "context" is a term co-occurrence matrix of *skip grams*. Skip grams count the number of times each word in the vocabulary occurs in a window around a target word [79]. For example, a skip gram window of five counts the number of times each word appears within five words to the left or five words to the right of the target word. The result is a symmetric $V \times V$ matrix of integers, with each row and column representing a word. Levy and Goldberg show that neural word embeddings may be viewed as a factorization of this matrix [80].

Text embedding research brings exciting possibilities to distributional semantics. Researchers have found that comparisons and compositions of embeddings may be linked to semantic meaning. For example, Mikolov et al. found similarities between the vectors representing countries and their capitals [76]. Compositions between word vectors may also capture semantic meaning as demonstrated by the oft-used "king $-$ man $+$ woman $\approx$ queen" example [81]. And alignment between vector spaces across languages promises the ability to compare meaning of words and phrases across languages [82].

[6] I have heretofore been using the general word "context" instead of "document" to discuss topic models. This is intentionally related to text embeddings and discussed more in Section 2.7, *LDA for Natural Language Statistics*.

As with topic models, there is no principled way to choose K, the number of embedding dimensions. Anecdotally, this is far less studied in the distributional semantics community than the topic modeling community. Yet, as I describe below, I believe the two communities are actually working on the same class of models, in spite of their disjoint lineage.

2.7 LDA for Natural Language Statistics

Latent Dirichlet Allocation has lost favor in machine learning circles. A colleague once asked me without any irony, "who still uses LDA?!?!" This sentiment is unfortunately widespread in the machine learning community. Jacob Eisenstein—a research scientist at Google—recently published *Introduction to Natural Language Processing*—one of the first comprehensive textbooks on the subject since deep learning models came to dominate in NLP [73]. He does not mention LDA at all in the chapter on distributional semantics (pages 309–332) and only mentions it in a footnote in the chapter on discourse (on page 358).

Yet LDA is still in widespread use in the digital humanities and computational social science. Machine learning may have moved on, but applications of LDA are still common in these less explicitly technical disciplines. This points to a need to make LDA more accessible [28, Ch. 10.2]. Accessibility means computational tools that are easy to use as well as a deeper understanding of *how* these models may be deployed to answer questions these disciplines have.

Moreover, I argue that the distinction between "topic models"—of which LDA is a member—and "text embeddings" made in the machine learning community is meaningless. Viewed through the lens of topology, LDA and newer text embeddings belong in the same class of models. In topology, an embedding, f, is a mapping between topological spaces. $f : \boldsymbol{X} \rightarrow \boldsymbol{Y}$ means that f maps a point in topological space $\boldsymbol{X}$ to a point in topological space $\boldsymbol{Y}$. From this perspective, LDA maps $\boldsymbol{X}$—points in a V-dimensional integer space—to $\boldsymbol{\Theta}$—points in a K-dimensional probability space. Newer text embeddings typically map from $\boldsymbol{X}$ to Euclidean space $\boldsymbol{Y}$.[7]

LDA is also criticized for its over reliance of the bag of words assumption. This conflates data preprocessing with the model itself. For this reason, I prefer to refer to the rows of $\boldsymbol{X}$—the data being modeled—

[7]We can extend this logic to encompass Transformers as well. A multi layer neural network may be viewed through this lens as a collection of embeddings $\{f_0, f_1, ..., f_O\}$ such that $f_0 : \boldsymbol{X} \rightarrow \boldsymbol{H_1}$, $f_1 : \boldsymbol{H_1} \rightarrow \boldsymbol{H_2}$, and so on until $f_{o-1} : \boldsymbol{H}_{o-1} \rightarrow \boldsymbol{O}$. $\boldsymbol{X}$ is the input layer; $\boldsymbol{H}_i$ are hidden layers; and $\boldsymbol{O}$ is the output layer. One might then consider relationship between contexts at any layer $\boldsymbol{H}_i$ or $\boldsymbol{O}$.

as "contexts" rather than "documents". A context may be a whole document, a chapter, a paragraph, a sentence, etc. Or a context may incorporate proximity or word order. For example a context could be the count of times each word in a corpus appears around a target word, as with skip-grams. A context could also be a count of each word appearing after or before a target word.[8] It's true that within a context, the bag-of-words assumption still holds. So, LDA cannot be a full sequence to sequence model. However, the way a context is constructed may encode proximity and order. And that construction affects the interpretations of probabilities resulting from the model. I have done this explicitly with LDA when writing the vignettes for a previously-released R package, *textmineR* [83]. In 2020, Adji Deng co-authored a paper with David Blei constructing the "embedding topic model" in a similar fashion [47].[9] Panigrahi et al. use LDA for word embeddings and find evidence that the "topic" dimensions encode different word senses, potentially addressing polysemy [84].

LDA and other probabilistic topic models also have an advantage *because* they embed into a probability space. As we know from traditional statistics, probability spaces are well-suited to quantify uncertainty around claims and inferences made from data—with or without linguistic origins. It is possible that embedding to probability spaces may aid tasks in distributional semantics such as uncovering analogies and cross-lingual embedding alignment. Probability spaces have well-defined relationships, transformations, and methods for composition. Euclidean spaces have fewer constraints on operations within them, leading to greater researcher degrees of freedom. However, analogies and cross-lingual embedding alignment are beyond the scope of the work proposed at present.

With this in mind, I believe that LDA is a good candidate of study for Natural Language Statistics for three reasons. First, LDA is a parametric Bayesian statistical model, which allows for uncertainty quantification, model diagnostics, etc. in line with established statistical practices. Second, as stated above, LDA embeds into probability spaces with the benefits they bring. Finally, LDA is a generative model of language. This allows us to use simulation studies of the LDA-DGP to help develop methods to help build models and diagnose model misspecification, as described in section 4 below.

LDA is a good candidate for Natural Language Statistics, but Natural Language Statistics can also be

[8] I'd call these lead-grams and lag-grams, respectively.

[9] I published the vignette doing this with LDA in 2017 and had made the connection years earlier, yet I never formalized it into proper research. I am glad that Deng and Blei did..

good for LDA. The tension over intrinsic evaluation versus human interpretability—as highlighted by Chang et al. [49]—is premature as LDA has an identification problem. If one cannot tell if a model is pathologically misspecified, then one should not rely on any interpretation of it. Analogously, if a linear regression model's residuals are clearly not Gaussian distributed with mean zero, one does not attempt to interpret its coefficients. We should expect the same from LDA and related models.

For LDA, Natural Language Statistics should focus on three tasks: detect pathological model misspecification[10] (e.g., choice of K, $\boldsymbol{\alpha}$, and $\boldsymbol{\eta}$), develop metrics to aid researchers in justifying a model's specification, and quantifying uncertainty around claims a researcher might want to make using a model. Relating empirical laws of language to the LDA-DGP is a first step for the former two tasks. The latter task relates closely to interpretation and depends on a model being statistically valid.

[10] One might argue that humans do not write using the LDA-DGP and thus no "correct" specification exists. They would be right! Yet, "this model is not right" is a much narrower statement than "this model is right". To use linear regression as an example again, having Gaussian distributed residuals does not mean that one specified the "right" model. But having non-Gaussian residuals means the model is wrong.

Chapter 3: Linking Zipf's law to the LDA-DGP for Simulation Studies

LDA is a latent variable model and, as a result, it can be challenging to study. No ground truth exists against which to compare models and methods. Selecting hyper parameters when building models is notoriously difficult as little formal guidance exists to guide a researcher's choices. Lack of ground truth complicates development of new metrics and methods. How is one to know if a new method is an improvement or a new metric insightful?

Fortunately, LDA models a generative process. This allows researchers to generate synthetic data sets by sampling from the LDA data generating process (LDA-DGP) where they have chosen K, $\boldsymbol{\alpha}$, and $\boldsymbol{\eta}$ themselves. They then may use these simulated data sets to compare a model against a synthetic ground truth. Such simulation studies have a lengthy history in the statistical literature, going back to 1975 or further [85]. Yet to be valid, synthetic data from the LDA-DGP must conform to empirical statistical laws of language such as Zipf's and Heaps's laws.

In this chapter I examine the LDA-DGP analytically and empirically to do the following:

1. Link Zipf's law—and by proxy Heaps's law—to the LDA-DGP,
2. Describe the effects hyper parameters of the LDA-DGP have in generating 4,096 synthetic corpora consistent with Zipf's and Heaps's laws, and
3. Describe the effects of misspecification of hyper parameters on the posterior of LDA models fit on a sample of the above synthetic data sets.

I show that setting $\boldsymbol{\eta}$ proportional to a power law produces synthetic data sets generated by the LDA-DGP that are consistent with Zipf's and Heaps's laws. I find that $\boldsymbol{\eta}$ is more important than previously appreciated, that $\boldsymbol{\alpha}$ may be less important than previously thought, and that K has the biggest effects on the posterior when misspecified.

3.1 Related Work

Work related to studying the LDA-DGP pulls from two seemingly disparate fields: topic modeling and complex systems theory. Complex systems theory deals in part with the emergence of power law distributions—as exemplified by some empirical laws of language—from complex systems [86].

3.1.1 Simulation Studies for LDA

Synthetic corpora appear commonly in topic modeling research. Shi et al. [53] list 14 works using synthetic corpora to study topic models. (See Table S3 in [53].) Most use some flavor of the LDA-DGP and focus on only one or two aspects of comparison between model and ground truth in the synthetic data set. Boyd-Graber, Hu, and Mimno [28] suggest the use of semi-synthetic data—i.e., drawing simulations from the posterior of LDA models fit on real-world data—to capture properties of natural language. I am unaware of any work linking empirical laws of language to the LDA-DGP. Shi et al. [38], [53] link their simulation method to Zipf's law but it does not use the LDA-DGP to produce its simulations.

Shi et al. [38], [53] go further than others and argue that use of synthetic corpora is "a principled approach" to evaluating topic models. Their approach does reproduce Zipf's law of language. Yet it does not use the LDA-DGP. It may represent a principled approach to studying probabilistic topic models with simulated data, but it is a parallel path to the one in this chapter. Using the LDA-DGP specifically allows for stronger statements related to pathological misspecification of LDA models. e.g., "Under the true model, then I would expect to see outcome A. Instead I see outcome B. Therefore, my model must be misspecified."[1]

3.1.2 Empirical Language Laws

Altmann and Gerlach [87] describe 9 universal laws purported to describe statistical regularities in human language. These laws are Zipf's [51], Heaps's [88], Taylor's [89], Menzerath-Altmann [90], Recurrence [51], Long-range correlation [91], Entropy scaling [87], Information content [51], and various network topology laws [87].

Of these laws, Zipf's and Heaps's laws are the most well known and relevant to LDA. The laws of Menzerath-Altmann, Recurrence, Long-range correlation, and Information content refer to properties of sub-

[1] An analogue from linear regression might be, "under the true model, residuals are independent and identically distributed (*i.i.d.*) Gaussian with mean zero. The residuals of my model are not *i.i.d.* Gaussian with mean zero. Therefore, my model must be misspecified."

words (e.g. length to information content), order of words, or proximity between words. LDA-DGP does not purport to model any of these. The law of Entropy scaling links entropy—in the information theoretic sense [92]—to the number of words in a block of text. The LDA-DGP may or may not be able to reproduce this law. Similarly, some network views of a corpus may or may not apply to the LDA-DGP. Both may warrant future exploration but are less directly macroscopic properties of a corpus.

Taylor's law may be relevant to the LDA-DGP but I leave its examination to future work as Zipf's and Heaps's laws are most commonly known as linguistic laws. Taylor's law was originally posed in the context of ecology [93]. Its linguistic interpretation is that the standard deviation of the total number of words is proportional to the power of the mean of the total number of words in a corpus. Gerlach and Altmann [89] explore the relationships between Zipf's, Heaps's, and Taylor's laws in the context of topic modeling. Instead of using the LDA-DGP directly, they examine its asymptotic form, which is a Poisson process.

Zipf's law

Zipf's law states that the frequency of a word is inversely proportional to the power of its frequency-rank. It is not unique to any language as the power law pattern appears to apply to all of them [94]. Zipf's law has also been applied to the sizes of cities [95], casualties in armed conflict [96], and more. While Zipf's law is a statement about a word's frequency related to its rank. Yet if word frequencies in a corpus are plotted as a histogram, the power law relationship holds [97].

Empirical distributions of Zipf's law for large corpora demonstrate a relationship somewhat inconsistent with that predicted by the law. Some have proposed that the frequency-to-rank relationship is actually a set of broken power laws with one parametarization for the head of the distribution, another for the body, and a third parametarization for the tail. Yet, Ha et al. find that this is a trick of tokenization. "Language is not made of individual words but also consists of phrases of 2, 3 and more words, usually called n-grams for n=2, 3, etc." When including n-grams in both English and Chinese corpora, Ha et al. [98] find that Zipf's law holds through the tail. Mandelbrot [99] developed a generalization of Zipf's law that better accounts for behavior at the head of the distribution.

Formally, Zipf's law is

$$F(r) \propto r^{-\gamma} \text{ for } \gamma \geq 1, r > 1 \tag{3.1}$$

where r is a word's rank, $F(r)$ is the frequency of a word's rank, and γ is a parameter to be estimated. For language $\gamma \approx 1$ [94] and can be found through maximum likelihood estimation [96]. Altmann and Gerlach [87] find $1.03 \leq \gamma \leq 1.58$ depending on the corpus and estimation method.

Goldwater et al. [100] explore the relationship between Zipf's law and statistical models of language. They develop a framework for producing power law word frequencies in two stages. While they link this framework to several models closely related to LDA, they do not extend it to the LDA-DGP itself.

Heaps's law

Heaps's law states that the number of unique words in a corpus, V, scales sub-linearly with the total number of words in a corpus, N [101]. Under mild assumptions, Heaps's law is asymptotically equivalent to Zipf's law [102]. Unlike Zipf's law, it is an increasing power law.

$$V \propto N^{\delta} \text{ for } N > 1,\ 0 < \delta < 1 \tag{3.2}$$

There are several ways to compute Heaps's law for a single corpus. Altmann and Gerlach use two methods: the first computes V and N for each document in the corpus and the second progresses over each word in a corpus calculating new values of V and N as each word is added. The latter approach works for single documents of sufficient length as well, such as a book [87].

3.2 The LDA Data Generating Process (LDA-DGP)

Assuming there are D contexts, K topics, V unique words, N total words and N_d words in the d-th context, the LDA-DGP is as follows:

1. Generate $\boldsymbol{B}$ by sampling K topics $\beta_k \sim \text{Dirichlet}(\boldsymbol{\eta}), \forall k \in \{1,2,...,K\}$
2. Generate $\boldsymbol{\Theta}$ by sampling D documents $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha}), \forall d \in \{1,2,...,D\}$
3. Then for each word, n in each context, d

a. Draw topic $z_{d,n}$ from Multinomial($\boldsymbol{\theta}_d$)

b. Draw word $w_{d,n}$ from Multinomial($\boldsymbol{\beta}_{z_{d,n}}$)

4. Repeat a. and b. N_d times.

Note that context d has N_d words $\forall\, d \in \{1,2,...,D\}$ and that the total number of words in the corpus is $N = \sum_{d=1}^{D} N_d$.

3.3 Linking the LDA-DGP to Zipf's law

Zipf's law describes the relative frequencies between words in a corpus. Therefore, the expected term frequency distribution of a corpus drawn from the LDA-DGP should describe the relationship between Zipf's law and the LDA-DGP. The following relationship is derived in Appendix C:

$$\mathbb{E}(w_v) = N\frac{\eta_v}{\sum_{v=1}^{V}\eta_v} \tag{3.3}$$

where w_v is the total number of times word v was sampled across all D contexts.

Equation (3.3) implies that the relative frequencies of words i and j are given by their ranks in a power law relationship. In other words, the shape of $\boldsymbol{\eta}$ is a power law, given by Zipf's law. No elements of $\boldsymbol{\alpha}$ appear in (3.3). As shown in Appendix C, the α_i's sum and factor out of the derivation. This makes some intuitive sense. An author can theoretically write an entire corpus of documents about only one topic or an author may write a corpus of documents with an even distribution of topics. In either case the aggregate word frequencies must follow Zipf's law.

This result is consistent with the framework laid out by Goldwater et al. [100]. The Dirichlet multinomial model - of which LDA is an application - is a special case of their framework. In this framework, the Dirichlet prior has a parameter following a power law. They did not put this in the specific form of LDA, however, as is done in this paper. Using this specification of the Dirichlet multinomial it is now possible to generate corpora with the statistical properties of natural language within an LDA framework. This facilitates exploring the properties of LDA under different conditions and the development of metrics aiding in model specification and post-estimation diagnostics.

The relationship in (3.3) also implies that one can estimate the shape of $\boldsymbol{\eta}$ from a document term matrix. Formally linking Zipf's law with the LDA-DGP we have

$$\mathbb{E}(w_v) = N\frac{\eta_v}{\sum_{v=1}^{V}\eta_v} = C \cdot r_v^{-\gamma} \tag{3.4}$$

where C and γ may be estimated from data and r_v, the rank of word v, is known. Some algebra then yields

$$\frac{\eta_v}{\sum_{v=1}^{V}\eta_v} = \frac{C}{N} \cdot r_v^{-\gamma}, \forall\ v \tag{3.5}$$

What still remains is for a researcher to specify the magnitude of $\boldsymbol{\eta}$, in other words $\sum_{v=1}^{V}\eta_v$, $\boldsymbol{\alpha}$, and the number of topics, K. The empirical evidence for specifying $\boldsymbol{\eta}$ as a power law rather than a flat prior is explored, though not resolved, in Section 3.4.3.

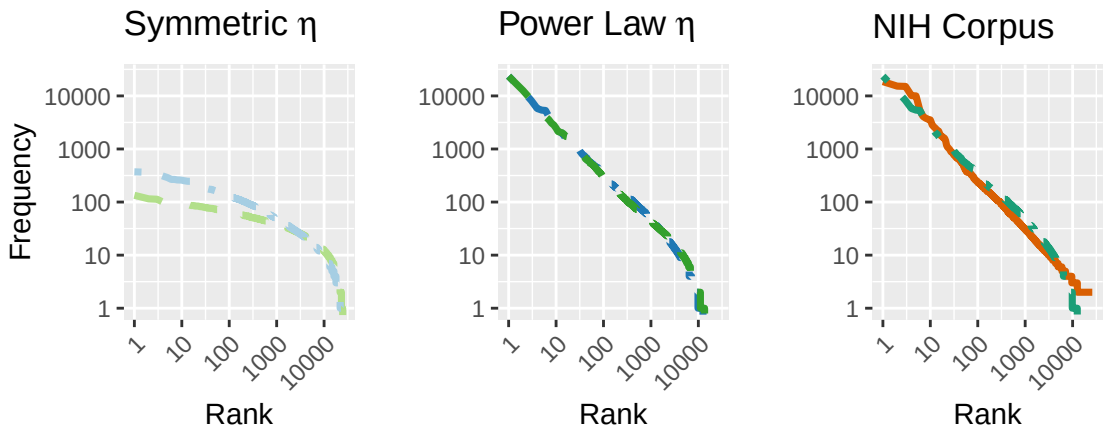


Figure 3.1: Frequency vs. rank curves plotted in log-log space for several synthetic corpora and one real corpus. The leftmost chart plots both a power law and symmetric $\boldsymbol{\alpha}$ with a symmetric $\boldsymbol{\eta}$. The center chart plots both a power law and symmetric $\boldsymbol{\alpha}$ with a power law $\boldsymbol{\eta}$. The rightmost chart plots real data from the NIH corpus against a power law $\boldsymbol{\alpha}$ and power law $\boldsymbol{\eta}$. Only when $\boldsymbol{\eta}$ is proportional to a power law do we see Zipf's law in synthetic corpora.

Figure 3.1 plots word frequencies of simulated and real data on a log-log scale. The leftmost image plots

word frequencies of simulated data sampled from the LDA-DGP with a symmetric η. The light blue dashed and dotted line has a power law α and the light green dashed line has a symmetric α. The center image plots word frequencies of simulated data sampled from the LDA-DGP with a power law η. The green dashed and dotted line has a power law α and the blue dashed line has a symmetric α. The rightmost image contains word frequencies from a random sample of 1,000 abstracts from grants awarded by the US National Institutes of Health (NIH) in 2014 [103], which is the solid orange line. Consistent with [98], unigrams and bigrams are included. Crucially, stop words[2] have *not* been removed from the NIH data, allowing the head of the distribution to reach its full magnitude. (Implications of this choice are considered in Section 3.5 below.) The dashed and dotted green line is the same as in the center image, provided as a comparison. All six lines in Figure 3.1 represent corpora that have the same number of contexts and words. The left and center images confirm the implications of the derrivation in Appendix C. Specifically, that to generate word frequencies consistent with Zipf's law, η must have a power law distribution regardless of the shape of α. The rightmost image demonstrates that simulated data can have a power law word frequency distribution similar to real world data.

Finally, the asymptotic equivalence between Zipf's and Heap's law indicates that η drives both Zipf's and Heap's laws, the two most commonly-understood emprical language laws.

3.4 Experiments

Simulation studies allow researchers to know the true parameters that generated a data set and evaluate how an estimated model performs against "the truth". But if simulated data cannot replicate the key statistical properties of its real world counterpart, it calls into question any conclusion made with such synthetic data. However, linking the LDA-DGP Zipf's law (and Heaps's law) adds rigor to simulation studies of LDA by giving synthetic data the gross statistical properties of human language.

In this section, I generate 4,096 synthetic data sets using different parametarizations of the LDA-DGP and baseline corpus parameters, such as number of contexts, vocabulary size, etc. Each of these corpora are consistent with Zipf's/Heaps's laws. After briefly describing the properties of these synthetic corpora, I use

[2]Stop words are words filtered out prior to analysis. Often these are very common words and are assumed to add little analytical value.

them to explore the effects of misspecifying hyper parameters when fitting LDA models.

3.4.1 Synthetic Data Generation

Each data set is drawn from a model with a unique combination of the following hyper parameters, resulting in 4,096 unique combinations:

1. $\sum_{v=1}^{V} \eta_v \in \{50, 250, 500, 1000\}$
2. $\sum_{k=1}^{K} \alpha_k \in \{0.5, 1, 3, 5\}$
3. $N_d \sim \text{Pois}(\lambda)$ where $\lambda \in \{50, 100, 200, 400\}$
4. $V \in \{1000, 5000, 10000, 20000\}$
5. $D \in \{500, 1000, 2000, 4000\}$
6. $K \in \{25, 50, 100, 200\}$

I choose $\boldsymbol{\eta}$ to be proportional to a power law, consistent with Appendix C and Zipf's law [51]. I choose $\boldsymbol{\alpha}$ to be symmetric for all data sets. The magnitudes of the Dirichlet hyper parameters, $\boldsymbol{\eta}$ and $\boldsymbol{\alpha}$ vary while keeping their shapes fixed. The magnitude of the parameter of a Dirichlet distribution plays a strong role in tuning the covariance between draws from that distribution. A smaller magnitude means larger variability between draws. In other words, if $\left(\sum_{v=1}^{V} \eta_v\right)$ is smaller, topics are less linguistically similar. If $\left(\sum_{v=1}^{V} \eta_v\right)$ is larger, topics are more linguistically similar. Similarly, $\left(\sum_{k=1}^{K} \alpha_k\right)$ tunes the topical similarity of documents.

3.4.2 Describing Properties of Synthetic Corpora

For each synthetic data set, I calculate parameter values for Zipf's law using linear regression in log-log space. Specifically, I fit $\log_{10} w_v = \log_{10} C - \gamma \log_{10} r_v + \varepsilon_v$, where ε_v is the linear regression error term for the v-th word. This method is often biased by low rank words. To compensate, I use the `poweRLaw` package [104] to select a threshold for excluding low rank words.

I also calculate several statistics from the synthetic data that a researcher would have available to them. These are N, the total number of word instances and $\hat{V}$, the total *observed* vocabulary size.

Results

Each of the synthetic corpora was generated with $\boldsymbol{\eta}$ such that its Zipf coefficient was the same, $\gamma = 1.07$. Even so, there was considerable variation in the observed Zipf coefficient, Zipf constant (C), and minimum word

frequency used to estimate the two Zipf parameters, *gamma* and C. I find that the estimate for γ is biased downwards in most of the synthetic data sets. The estimated value of γ is highly correlated with the number of topics and, to a lesser degree, the magnitude of η from the generating distribution. The Zipf constant, C, is largely correlated with the total volume of words in the data set. The minimum word frequency used to estimate the Zipf parameters is highly correlated with the number of topics in the generating distribution. The magnitude of α does not appear to be a significant driver of any Zipf-related variable.

Heaps's law describes the relationship between the total number of words in a corpus and the size of the vocabulary. It is an increasing power law with a slope that gets flatter as the total number of words increases. In the context of a simulated corpus, it is related to the difference between the total size of the lexicon available to the DGP, V, and the total number of unique words observed. If every word is sampled at least once, then the number of observed words equals V. However, under Zipf's law it is more likely that the total number of observed words will be less than V. I find that having more topics is highly correlated with observing more words.

Figure 3.2 below plots histograms of the Zipf coefficient, γ, Zipf constant, C, minimum word frequency used to estimate the Zipf parameters, and the difference between the total vocabulary size and observed vocabulary size. The leftmost histogram for γ has a dashed line at $\gamma = 1.07$, the true value of γ in the simulations. The Zipf constant and minimum used word frequency are presented on a log scale. All four demonstrate variability and the downward bias of the estimated Zipf coefficient over the true value is apparent. I hypothesize that this is due to the method used to estimate the Zipf coefficient than it is a bias in the simulations. Manual inspection of a sample of simulations do not show evidence of bias.

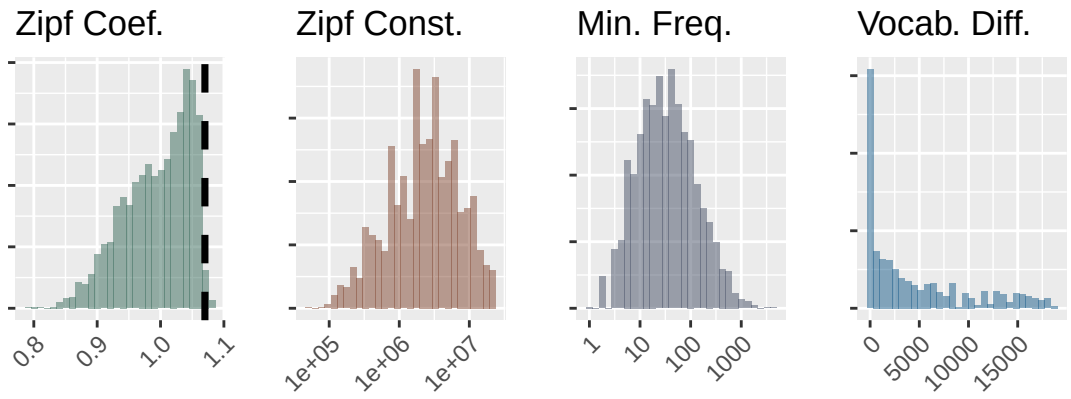


Figure 3.2: Histograms of the Zipf coefficient, γ, Zipf constant, C, minimum word frequency used to estimate the Zipf parameters, and the difference between the total vocabulary size and observed vocabulary size. The Zipf constant and minimum used word frequency are presented on a log scale.

I use linear regression as a multiple correlation to examine which features of the LDA-DGP are associated with variability in the four parameters in Figure 3.2. This use of linear regression is not to model drivers of each parameter. Instead I use regression to describe linear correlations holding other variables constant. For each of the three parameters of interest, I regressed them on the seed parameters of the LDA-DGP from Section 3.4.1.1 and their quadratics. The result, excluding regression constants, along with 95% confidence intervals are plotted in Figure 3.3. The top row includes both $\sum_{k=1}^{K} \alpha_k$ and $\left(\sum_{k=1}^{K} \alpha_k\right)^2$. The bottom row excludes both $\sum_{k=1}^{K} \alpha_k$ and $\left(\sum_{k=1}^{K} \alpha_k\right)^2$.

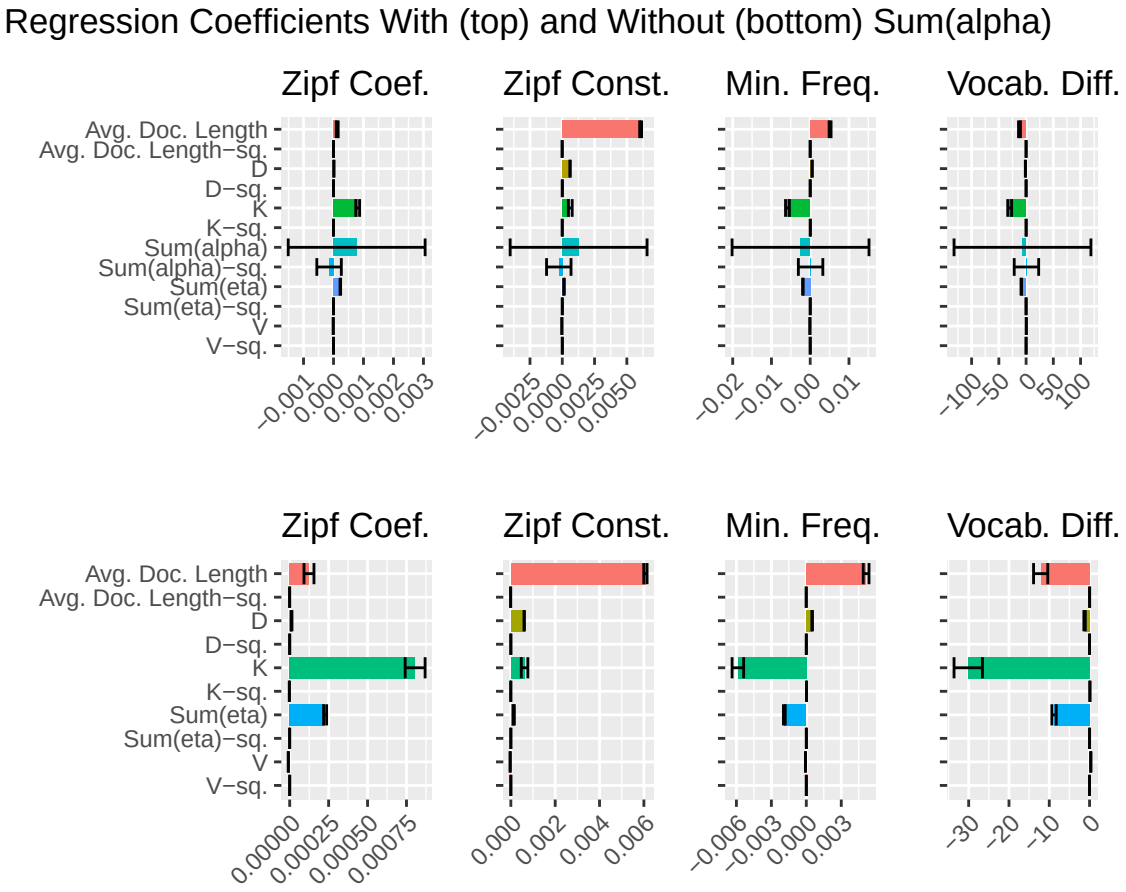


Figure 3.3: Coefficients from regressions of four Zipf's law-related variables on parameters from the LDA-DGP and their quadratics, excluding intercept terms. The top row includes terms related to α and the bottom row excludes them. Removing the α-related terms does not affect goodness of fit. The adjusted R^2 of the models in the top row are 0.733, 0.984, 0.866, and 0.920 respectively. The adjusted R^2 of the models in the bottom row are 0.736, 0.984, 0.866, and 0.920 respectively. The regressions for the Zipf constant and minimum word frequency are scaled by log 10.

The effect of α's magnitude has on the four Zipf variables appears to be negligible. In each case, the coefficient on $\sum_{k=1}^{K} \alpha_k$ show a moderately-sized mean response on the outcome. However, in all three cases, the 95% confidence intervals are extremely large. Removing both $\sum_{k=1}^{K} \alpha_k$ and $\left(\sum_{k=1}^{K} \alpha_k\right)^2$ does not lower the R^2 of any of the regressions, nor appreciably change the coefficients associated with other variables.

The number of topics, K, is positively associated with the Zipf coefficient, γ, and negatively associated with the minimum frequency of words used to estimate the Zipf parameters. In other words, an increase in

topics is associated with a steeper slope of the Zipf curve in log-log space. And an increase in topics is also associated with Zipf's law "falling" off for low frequency words deeper into the tail. In plain speech: a corpus with more topics seems to express Zipf's law more strongly.

Having topics that are more highly correlated with each other is associated with a corpus expressing Zipf's law more strongly. In other words, when the magnitude of $\boldsymbol{\eta}$ is larger, the estimate of γ is larger. This is similar to the relationship between K and γ.

The Zipf constant, C, is most strongly associated with the total volume of words in a corpus. The total volum of words in a corpus is the average context length multipled by the total number of contexts (i.e., Avg. Doc Length times D). In a separate regression not reported here, the average context length and number of contexts alone account for 95% of the variance in $\log_{10} C$. This makes intuitive sense as C acts only to scale Zipf's law to the corpus on which it is applied.

Interestingly, the number of unique words in a corpus, V, does not appear to have a strong correlation to Zipf's law.

Finally, having more topics, a larger magnitude of $\boldsymbol{\eta}$, and longer contexts are all associated with observing more vocabulary words. It makes intuitive sense that longer contexts would be associated with observing more vocabulary words; it's simply a larger sample. A larger magnitude of $\boldsymbol{\eta}$ means that there is less variability in vocabulary between topics, which would make the tail end of the Zipf distribution less sparse. This, again, would make it more likely that one would observe a larger vocabulary.

3.4.3 Describing LDA Model Misspecification

Since we know the process that created a synthetic data set, we can measure the difference between a perfectly-specified model and misspecified models.

Modeling Choices and Evaluation Methods

For each of the 4,096 data sets, I fit five LDA models, one perfectly specified according to the parameters that generated the data and four misspecified models. The misspecified models misspecify K, $\sum_{k=1}^{K} \alpha_k$, $\sum_{v=1}^{V} \eta_v$, and the shape of $\boldsymbol{\eta}$, making it flat. For each model, the Gibbs sampler was run for 200 iterations, 150 of which

were burn in iterations. The final posterior is the average over the last 50 iterations[3]. For the hyper parameters K, $\sum_{k=1}^{K} \alpha_k$, and $\sum_{v=1}^{V} \eta_v$, misspecifications are chosen by random sample of the available options in Section 3.4.1, excluding the correct value. For example, to misspecify the number of topics in a model where the true value is $K = 100$, the estimated K is selected at random from $\{25, 50, 200\}$. As a result, approximately half of the misspecified models will have the key hyper parameter too small and half will be too big.

I compare the following between correctly-specified and misspecified models:

- Log Likelihood, averaged over the last 50 Gibbs iterations, of perfectly specified to misspecified models. The log likelihood calculation follows that used in [60].
- Deviance information criterion, or DIC. This is a somewhat Bayesian version of the Akaike information criterion, an estimate of predictive error, trading off goodness of fit and model simplicity.
- Perplexity, a transformation of the log likelihood—above—weighted by the number of word occurrences in a corpus and multiplied by negative one.
- Coefficient of determination, or R^2. This statistic calculates the proportion of variability in the data that has been accounted for in the fit model. This statistic and a study of its properties is in Chapter 4.
- Partial coefficient of determination—see Chapter 4—averaged over all topics in a model.
- Prevalence of each topic, averaged across all topics in the model. The prevalence of topic k is calculated with $\sum_{d=1}^{D} N_d \cdot \theta_{d,k} \cdot 100$.
- Probabilistic coherence (see Appendix E)—or just "coherence"—calculated on β_k and averaged over all topics in a model.
- Lambda coherence (L. coherence), probabilistic coherence calculated on $\boldsymbol{\lambda}_k$ (see Appendix A) and averaged over all topics in a model.

In the comparing correctly specified and misspecified models, I use a paired t-statistic to calculate 95% confidence intervals in the difference between the correct (i.e., "untreated") and misspecified (i.e., "treated") groups. This paired t-statistic confidence interval is applied to all metrics above.

[3] It is common to burn in half of the total number of iterations. I chose to burn in over the last quarter for computational reasons. Specifically, I wanted at least 150 burn in iterations but for time complexity reasons, limited the total number of iterations to 200.

Results

Figure 3.4 plots the mean differences and 95% confidence intervals comparing the perfectly-specified model and each of the models where a single hyper parameter was misspecified. Note that because only one hyper parameter was misspecified at a time, that these mean differences represent a causal effect of single hyper parameter misspecification. This analysis does not evaluate the effects of multiple misspecifications at once.

Misspecifying the number of topics has the largest effects on model goodness of fit, whether measured with log likelihood, perplexity, DIC, or R^2. This may reflect the nature of the goodness of fit metrics. All else constant, a model with more topics will have a lower likelihood, reflecting a larger sample space. And, as demonstrated in Chapter 4, R^2 increases with the number of estimated topics, seemingly independently of the true number of topics. However, it is worth noting that having too many topics does not increase R^2 nearly as much as having too few topics decreases it.

Misspecifying the number of topics is seemingly the only misspecification that affects the average prevalence of topics. This result is intuitive and similarly a reflection of the properties of prevalence. Having more estimated topics will spread prevalence over more topics, decreasing the average.

Misspecifying η has mixed effects. A flat η seems to improve all evaluation metrics except for R^2. Specifying the magnitude of η to be too large has a negative effect on log likelihood, R^2, and coherence. Yet specifying the magnitude of η to be too small seems to have little effect save a moderate boost in R^2.

Misspecifying α does not seem to have much of an effect at all.

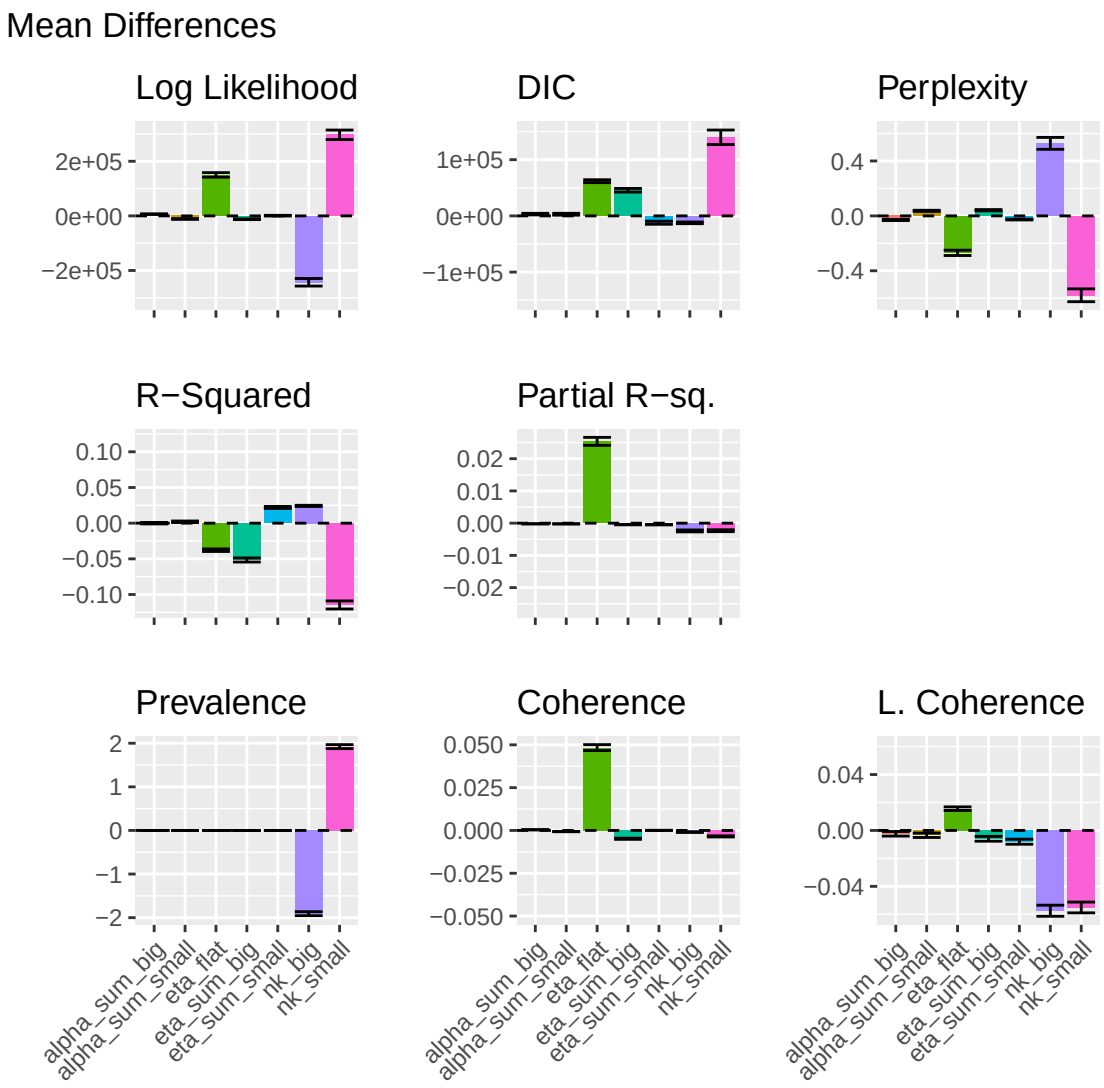


Figure 3.4: Mean differences (with 95% confidence intervals) in the evaluation metrics between perfectly-specified models and misspecified models. The misspecified hyper parameter is on the x-axis.

3.5 Discussion

This chapter demonstrates how to generate synthetic corpora using the LDA-DGP that are consistent with the key empirical statistical laws of language. This enables researchers to engage in more principled simulation studies for LDA as a model for human language. Such simulations are used throughout the remainder of this dissertation.

Two analyses are performed here. The first is describing which aspects of the LDA-DGP are associated with parameter values for Zipf's law and (by proxy) Heaps's law. The second systematically misspecifies LDA models and measures the causal effects of hyper parameter misspecification for the generated data sets.

To align the synthetic corpus's word frequencies to the real corpus's word frequencies in Figure 3.1, I did not remove stop words from the NIH corpus. All synthetic corpora in this chapter (and dissertation) produced with the LDA-DGP include stop words. However in practice, researchers tend to remove stop words prior to fitting an LDA model. This presents several options for future consideration. If we use synthetic data studies to examine the properties of topic models leaving stop words in (as is done here), researchers would need to include stop words in their models to be consistent. This is in line with [105], which shows that stop words may be removed after modeling to aid human interpretation while modeling the *whole* corpus. An alternative of the method I use here, using the more general Zipf-Mandlebrodt law may allow the production of synthetic corpora where stop words have been removed [99].

The studies here demonstrate that $\boldsymbol{\eta}$ is more important than previously appreciated. This chapter also brings some evidence that $\boldsymbol{\alpha}$ is not as important as is conventionally understood. Specifically, the results about $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ presented here run counter to the results seen in [50]. I view this as evidence that more research is needed, rather than evidence overturning convntional views on LDA's priors. The studies in this chapter are preliminary. I did not generate synthetic corpora using asymmetric specifications of $\boldsymbol{\alpha}$, nor did I seek to validate patterns of misspecification on LDA models fit on real data. I leave such efforts for future work and only note that there is much more to be understood about prior specification for LDA models.

Misspecifying K has the most consistent overall impact on posterior statistics. The instinct of many to worry about it is correct. We still need future research to find principled ways of setting the number of topics. However by linking Zipf's (and thus Heaps's) law to the LDA-DGP, such studies are now significantly closer to being available. We can now create synthetic data where the true data generation process is known for LDA, enabling a more rigorous examination of modeling effects and best practices.

Chapter 4: Coefficient of Determination for Topic Models

According to an often-quoted but never cited definition, "the goodness of fit of a statistical model describes how well it fits a set of observations. Measures of goodness of fit typically summarize the discrepancy between observed values and the values expected under the model in question."[1] Goodness of fit measures vary with the goals of those constructing the statistical model. Inferential goals may emphasize in-sample fit while predictive goals may emphasize out-of-sample fit. Prior information may be included in the goodness of fit measure for Bayesian models, or it may not. Goodness of fit measures may include methods to correct for model over fitting. In short, goodness of fit measures the performance of a statistical model against the ground truth of observed data. Fitting the data well is generally a necessary—though not sufficient—condition for trust in a statistical model, whatever its goals.

Many researchers have eschewed goodness of fit measures as metrics of evaluating topic models. This is unfortunate. Goodness of fit is often the first line of defense against a pathologically misspecified model. If the model does not fit the data well, can it be relied on for inference or interpretation? Of course goodness of fit is not the only, perhaps not even the most important, measure of a good model. In fact a model that fits its training data too well is itself problematic. Nevertheless, a consistent and easily-interpreted measure of goodness of fit for topic models can serve to demystify the dark art of topic modeling to otherwise statistically literate audiences.

Goodness of fit manifests itself in topic modeling through word frequencies. Topic models are not fully unsupervised methods. If they were unsupervised, this would mean that no observations exist upon which to compare a model's fitted values. However, probabilistic topic models are ultimately generative models of word frequencies [29]. The expected value of word frequencies in a document under a topic model is given by the expected value of a multinomial random variable. The outcomes that can be compared to the predictions, then, are the word frequencies themselves. Most goodness of fit measures in topic modeling are restricted to in-sample fit. Yet some out-of-sample measures have been developed [106].

[1] This quote appears verbatim on Wikipedia and countless books, papers, and websites. I cannot find its original source.

For any probabilistic topic model, the following relationship holds

$$\mathbb{E}(\boldsymbol{X}) = \boldsymbol{n} \odot \boldsymbol{\Theta} \cdot \boldsymbol{B} \tag{4.1}$$

where $\boldsymbol{X}$ is a matrix of observed word frequencies, $\boldsymbol{n}$ is a D-length vector whose d-th entry is the number of terms in the d-th document, $\boldsymbol{\Theta}$ is a matrix of estimated topic distributions over documents, and $\boldsymbol{B}$ is a matrix of estimated word distributions over topics. Also$\odot$ denotes elementwise multiplication whereas $\cdot$ is the typical dot product.

4.1 Related Work

Evaluation methods for topic models can be broken down into three categories: manual inspection, intrinsic evaluation, and extrinsic evaluation [53]. Manual inspection involves human judgement upon examining the outputs of a topic model. Intrinsic evaluation measures performance based on model internals and its relation to training data. R-squared is an intrinsic evaluation method. Extrinsic methods compare model outputs to external information not explicitly modeled, such as the document class.

Manual inspection is subjective but closely tracks how many topic models are used in real-world applications. Most research on topic model evaluation has focused on presenting ordered lists of words that meet human judgement about words that belong together. For each topic, words are ordered from the highest value of β_k—the multinomial parameter for the k-th topic—to the lowest (i.e. the most to least frequent in each topic). In [49] the authors introduce the "intruder test" to evaluate the reasonableness of a topic's words appearing together. Judges are shown a few high-probability words in a topic, with one low-probability word mixed in. Judges must find the low-probability word, the intruder. They then repeat the procedure with documents instead of words. A good topic model should allow judges to easily detect the intruders.

One class of intrinsic evaluation methods attempts to approximate human judgment. These metrics are called "coherence" metrics. Coherence metrics attempt to approximate the results of intruder tests in an automated fashion. Researchers have put forward several coherence measures. These typically compare pairs of highly-ranked words within topics. Röder et al. evaluate several of these [43]. They have human evaluators rank topics by quality and then compare rankings based on various coherence measures to the ranking of the

evaluators. They express skepticism that existing coherence measures are sufficient to assess topic quality. In a paper presented at the conference of the Association for Computational Linguistics (ACL), Lau, Newman, and Baldwin [107] find that normalized pointwise mutual information (NPMI) is a coherence metric that closely resembles human judgement.

Other popular intrinsic methods are types of goodness of fit. The primary goodness of fit measures in topic modeling are likelihood metrics. Likelihoods, generally the log likelihood, are naturally obtained from probabilistic topic models. Researchers have used likelihoods to select the number of topics [39], compare priors [50], or otherwise evaluate the efficacy of different modeling procedures [108] [109]. A popular likelihood method for evaluating out-of-sample fit is called perplexity. Perplexity measures a transformation of the likelihood of the held-out words conditioned on the trained model.

The most common extrinsic evaluation method is to compare topic distributions to known document classes. The most prevalent topic in each document is taken as a document's topical classification. Then, researchers will calculate precision, recall, and/or other diagnostic statistics for classification as measures of a topic model's success.

Though useful, prevalent evaluation metrics in topic modeling are difficult to interpret, are inappropriate for use in topic modeling, or cannot be produced easily. Intruder tests are time-consuming and costly, making intruder tests infeasible to conduct regularly. Coherence is not primarily a goodness of fit measure. AUC, precision, and recall metrics mis-represent topic models as binary classifiers. This misrepresentation ignores one fundamental motivation for using topic models: allowing contexts to contain multiple topics. This approach also requires substantial subjective judgement. Researchers must examine the high-probability words in a topic and decide whether it corresponds to the corpus topic tags or not.

Likelihoods have an intuitive definition: they represent the probability of observing the training data if the model is true. Yet properties of the underlying corpus influence the scale of the likelihood function. Adding more contexts, having a larger vocabulary, and even having longer contexts all reduce the likelihood. Likelihoods of multiple models on the same corpus can be compared. (Researchers often do this to help select the number of topics for a final model [39].) Topic models on different corpora cannot be compared, however. One corpus may have 1,000 contexts and 5,000 words, while another may have 10,000 contexts and 25,000

words. The likelihood of a model on the latter corpus will be much smaller than a model on the former simply because the latter has more degrees of variability. Yet this does not indicate the model on the latter corpus is a worse fit; the likelihood function is simply on a different scale. Perplexity is a transformation of the likelihood often used for out-of-sample contexts. The transformation makes the interpretation of perplexity less intuitive than a raw likelihood. Further, perplexity's scale is influenced by the same factors as the likelihood.

4.2 The Coefficient of Determination: R^2

The coefficient of determination is a popular, intuitive, and easily-interpretable goodness of fit measure. The coefficient of determination, denoted R^2, is most common in ordinary least squares (OLS) regression. However, researchers have developed R^2 and several pseudo R^2 measures for many classes of statistical models. The largest value of R^2 is 1, indicating a model fits the data perfectly. The formal definition of R^2 (below) is interpreted—without loss of generality—as the proportion of *variability* in the data that is explained by the model. For linear models with outcomes in $\mathbb{R}_1$, R^2 is bound between 0 and 1 and is the proportion of *variance* in the data explained by the model [110]. Even outside of the context of a linear model, R^2 retains its maximum of 1 and its interpretation as the proportion of explained variability. Negative values of R^2 are possible for non-linear models or models in $\mathbb{R}_M$ where $M > 1$. These negative values indicate that simply guessing the mean outcome is a better fit than the model.

4.2.1 The Standard Definition of R^2

For a model, f, of outcome variable, y, where there are N observations, R^2 is derived from the following:

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{4.2}$$

$$SST = \sum_{i=1}^{N} (y_i - \bar{y})^2 \tag{4.3}$$

$$SSR = \sum_{i=1}^{N} (f_i - y_i)^2 \tag{4.4}$$

The standard definition of R^2 is a ratio of summed squared errors.

$$R^2 \equiv 1 - \frac{SSR}{SST} \tag{4.5}$$

SST and *SSR* are known as the total sum of squares and residual sum of squares, respectively.

4.2.2 A Geometric Interpretation of R^2

R^2 may be interpreted geometrically as well. *SST* is the total squared-Euclidean distance from each y_i to the mean outcome, $\bar{y}$. Then *SSR* is the total squared-Euclidean distance from each y_i to its predicted value under the model, f_i. Recall that for any two points $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{R}_M$

$$d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_{i=1}^{M} (p_i - q_i)^2} \tag{4.6}$$

where $d(\boldsymbol{p}, \boldsymbol{q})$ denotes the Euclidean distance between $\boldsymbol{p}$ and $\boldsymbol{q}$. R^2 is often taught in the context of OLS where $y_i, f_i \in \mathbb{R}_1$. In that case, $d(y_i, f_i) = \sqrt{(y_i - f_i)^2}$; by extension $d(y_i, \bar{y}) = \sqrt{(y_i - \bar{y})^2}$. In the multidimensional case where $\boldsymbol{y}_i, \boldsymbol{f}_i \in \mathbb{R}_M; M > 1$, then $\bar{\boldsymbol{y}} \in \mathbb{R}_M$ represents the point at the center of the data in $\mathbb{R}_M$.[2]

We can rewrite R^2 using the relationships above. Note than now $\bar{\boldsymbol{y}}$ is now a vector with M entries. The j-th entry of $\bar{\boldsymbol{y}}$ is averaged across all N observations. i.e $\bar{y}_j = \frac{1}{N} \sum_{i=1}^{N} y_{i,v}$. From there we have:

$$\bar{\boldsymbol{y}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{y}_i \tag{4.7}$$

$$SST = \sum_{i=1}^{N} d(\boldsymbol{y}_i, \bar{\boldsymbol{y}})^2 \tag{4.8}$$

$$SSR = \sum_{i=1}^{N} d(\boldsymbol{y}_i, \boldsymbol{f}_i)^2 \tag{4.9}$$

$$\Rightarrow R^2 \equiv 1 - \frac{SSR}{SST} \tag{4.10}$$

Figure 4.1 visualizes the geometric interpretation of R^2 for outcomes in $\mathbb{R}_2$. The left image represents *SST*: the red dots are data points ($\boldsymbol{y}_i$); the black dot is the mean ($\bar{\boldsymbol{y}}$); the line segments represent the Euclidean distance from each $\boldsymbol{y}_i$ to $\bar{\boldsymbol{y}}$. *SST* is obtained by squaring the length of each line segment and then adding the squared lengths together. The right image represents *SSR*: the blue dots are the fitted values under the model ($\boldsymbol{f}_i$); the line segments represent the Euclidean distance from each $\boldsymbol{f}_i$ to its corresponding $\boldsymbol{y}_i$. *SSR* is obtained by squaring the length of each line segment and then adding the squared segments together.

[2] In the one-dimensional case, $y_i, f_i \in \mathbb{R}_1$, *SSR* can be considered the squared-Euclidean distance between the n-dimensional vectors y and f. However, this relationship does not hold when $\boldsymbol{y}_i, \boldsymbol{f}_i \in \mathbb{R}_M; M > 1$.

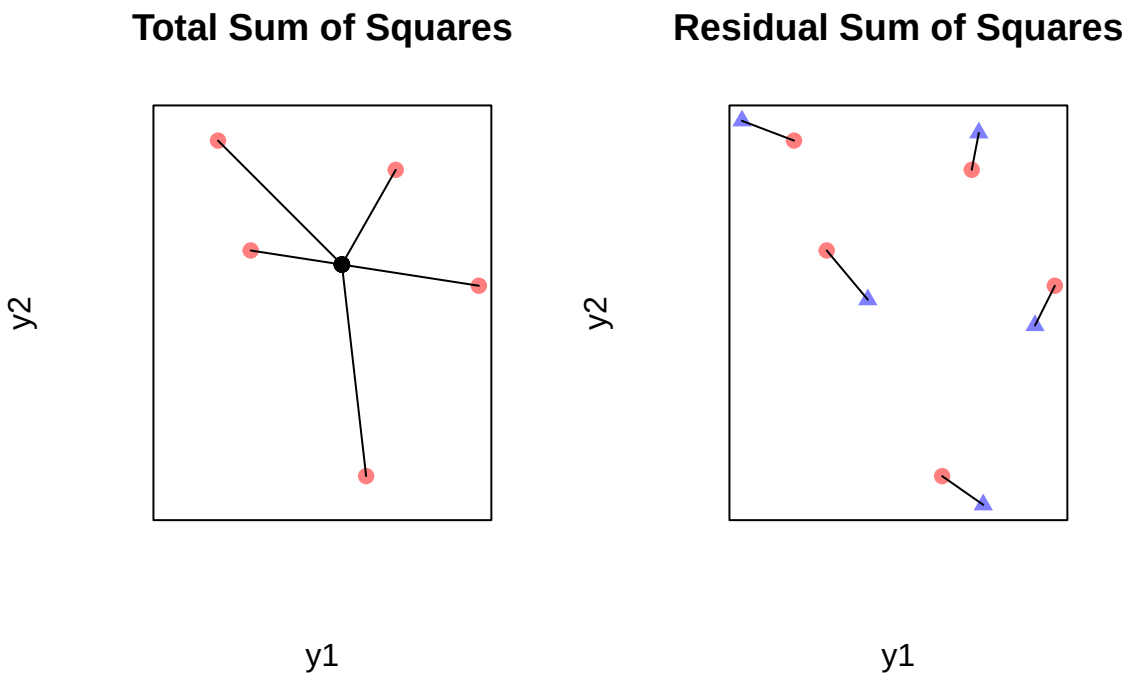


Figure 4.1: Visualizing the geometric interpretation of R-squared: corresponds to an R-squared of 0.87

The geometric interpretation of R^2 is similar to the "explained variance" interpretation. When $SSR = 0$, then the model is a perfect fit for the data and $R^2 = 1$. If $SSR = SST$, then $R^2 = 0$ and the model is no better than just guessing $\bar{y}$. When $0 < SS_{resid} < SS_{tot}$, then the model is a better fit for the data than a naive guess of $\bar{\boldsymbol{y}}$. In a non-linear or multi-dimensional model, it is possible for $SSR > SST$. In this case, R^2 is negative, and guessing $\bar{\boldsymbol{y}}$ is better than using the model.

4.2.3 Extending R^2 to Topic Models

An R^2 for topic models follows from the geometric interpretation of R^2. For a document, d, the observed value, $\boldsymbol{x}_d$, is a vector of integers counting the number of times each token appears in n_d draws. The document's fitted value under the model follows that $\boldsymbol{x}_d$ is the outcome of a multinomial random variable. The

fitted value is

$$\boldsymbol{f}_d = \mathbb{E}(\boldsymbol{x}_d) = n_d \odot \boldsymbol{\theta}_d \cdot \boldsymbol{B} \tag{4.11}$$

The center of the contexts in the corpus, $\bar{\boldsymbol{x}}$, is obtained by averaging the token counts across all contexts. From this we obtain R^2.

$$\bar{\boldsymbol{x}} = \frac{1}{D}\sum_{d=1}^{D} \boldsymbol{x}_d \tag{4.12}$$

$$SST = \sum_{d=1}^{D} d(\boldsymbol{x}_d, \bar{\boldsymbol{x}})^2 \tag{4.13}$$

$$SSR = \sum_{d=1}^{D} d(\boldsymbol{x}_d, \boldsymbol{f}_d)^2 \tag{4.14}$$

$$R^2 \equiv 1 - \frac{SSR}{SST} \tag{4.15}$$

4.2.4 Pseudo Coefficients of Variation

Several pseudo coefficients of variation have been developed for models where the traditional R^2 is inappropriate. Some of these, such as Cox and Snell's R^2 [111] or McFadden's R^2 [112] may apply to topic models. As pointed out in *UCLA's Institute for Digital Research and Education*, [113]

> These are 'pseudo' R-squareds because they look like R-squared in the sense that they are on a similar scale, ranging from 0 to 1 (though some pseudo R-squareds never achieve 0 or 1) with higher values indicating better model fit, but they cannot be interpreted as one would interpret an OLS R-squared and different pseudo R-squareds can arrive at very different values.

The experiments in this chapter calculate an uncorrected McFadden's R^2 for topic models to compare to the standard (non-pseudo) R^2. McFadden's R^2 is defined as

$$R^2_{Mc} \equiv 1 - \frac{ln(L_{full})}{ln(L_{restricted})} \tag{4.16}$$

where L_{full} is the estimated likelihood of the data under the model and $L_{restricted}$ is the estimated likelihood of the data free of the model. In the context of OLS, the restricted model is a regression with only an intercept term. For other types of models (such as topic models), care should be taken in selecting what "free of the model" means.

For topic models, "free of the model" may mean that the words were drawn from a simple multinomial distribution, whose parameter is proportional to the relative frequencies of words in the corpus overall. This is the specification used for the empirical analysis in this paper.

4.3 Experiments

To evaluate R^2 for topic models, I perform several simulation experiments and two experiments with a real-world data set. The simulation experiments are based on 4,096 synthetic data sets drawn from the LDA data generating process. (See the previous chapter.) The real-world data set is a sample of 1,000 abstracts from grants awarded by the US National Institutes of Health (NIH) in 2014 [103].

4.3.1 Simulation Analysis

Effective use of R^2 for topic models requires an understanding of its properties under varying conditions. I generate 4,096 (or 4^6) synthetic corpora using LDA as a data generating process and compare how both R^2 and McFadden's R^2 change as properties of the simulated data change.

Synthetic Data Generation and Evaluation Metrics

Each data set is drawn from a model with a unique combination of the following hyper parameters:

1. $\sum_{v=1}^{V} \eta_v \in \{50, 250, 500, 1000\}$
2. $\sum_{k=1}^{K} \alpha_k \in \{0.5, 1, 3, 5\}$
3. $N_d \sim \text{Pois}(\lambda)$ where $\lambda \in \{50, 100, 200, 400\}$
4. $V \in \{1000, 5000, 10000, 20000\}$
5. $D \in \{500, 1000, 2000, 4000\}$
6. $K \in \{25, 50, 100, 200\}$

I choose $\boldsymbol{\eta}$ to be proportional to a power law, consistent with Appendix C and and Zipf's law [51]. I chose $\boldsymbol{\alpha}$

to be symmetric. The magnitudes of the Dirichlet hyperparameters, $\boldsymbol{\eta}$ and $\boldsymbol{\alpha}$ vary while keeping their shapes fixed. The magnitude of the parameter of a Dirichlet distribution plays a strong role in tuning the covariance between draws from that distribution. A smaller magnitude means larger variability between draws. In other words, if $\left(\sum_{v=1}^{V} \eta_v\right)$ is smaller, topics are less linguistically similar. If $\left(\sum_{v=1}^{V} \eta_v\right)$ is larger, topics are more linguistically similar. Similarly, $\left(\sum_{k=1}^{K} \alpha_k\right)$ tunes the topical similarity of documents.

To remove the effects of estimating a pathologically-misspecified model, I use the data generating multinomial parameters $\boldsymbol{\Theta}$ and $\boldsymbol{B}$ to calculate R^2, and McFadden's R^2. Thus, these calculations represent the best possible model for a given simulated data set.

McFadden's R^2 is a ratio of likelihoods. Various methods exist for calculating likelihoods of topic models [106]. Most of these methods have a Bayesian perspective and incorporate prior information. Not all topic models are Bayesian, however [37]. And in the case of simulated corpora, the exact data-generating parameters are known a priori. As a result, likelihoods calculated for here follow the simplest definition: they represent the probability of observing the generated data, given the (known or estimated) multinomial parameters of the model, $\boldsymbol{\Theta}$ and $\boldsymbol{B}$. The "model-free" likelihood assumes that each document is generated by drawing from a single multinomial distribution. The parameters of this model-free distribution are proportional to the frequency of each token in the data.

Results

Each plot in Figure 4.2 shows how the distribution of R^2 and McFadden's R^2 change as each of the number of estimated topics (K), average context length (λ), vocabulary size (V), and number of contexts (D) changes using boxplots. The standard R^2 does not change with properties of the data except for the average context length. This indicates that one should expect, for example, a higher R^2 if analyzing a corpus of full-length articles compared to a corpus of tweets. However, McFadden's R^2 is invariant only to the number of contexts. It is positively correlated to the number of topics, average context length, and vocabulary size. This indicates that McFadden's R^2 suffers from the same problems as the likelihood function. Many properties of the underlying data affect the resulting metric.

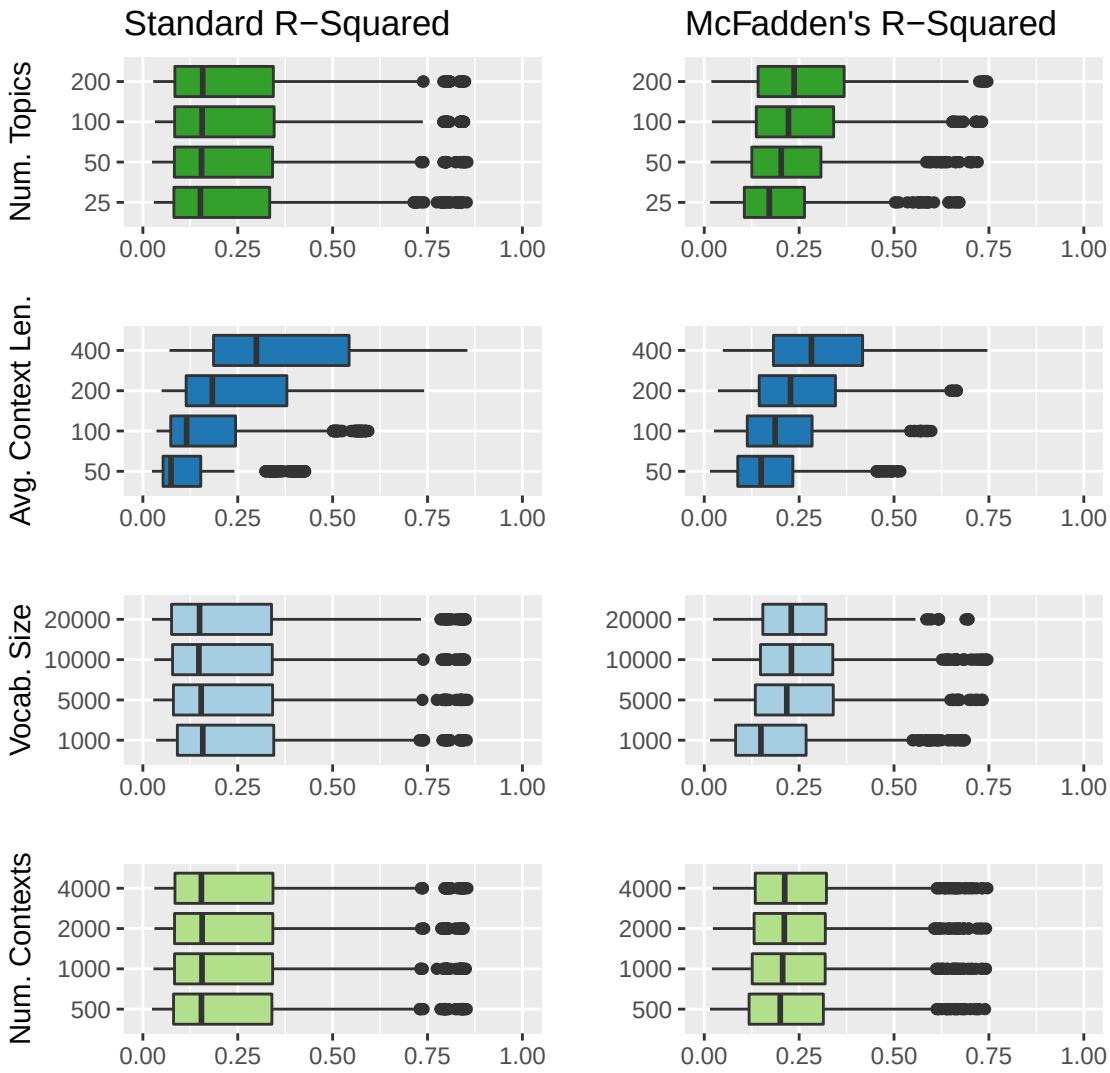


Figure 4.2: Comparing the standard geometric R^2 to McFadden's using population parameters and sampled data. Standard R^2 is positively correlated with the average document length but invariant to the number of topics, number of unique words, and number of documents. McFadden's is correlated with all but the number of contexts.

These properties of R^2 suggest that context length may be important factor in model fit whereas the number of contexts is not. When choosing between fitting a topic model on short abstracts, full-page contexts, or multi-page papers, it appears that longer contexts are better. Second, since model fit is invariant for corpora over 500 contexts (our lower bound for simulation), taking a sample from a large corpus may yield estimates

of parameters that accurately reflect the variability in word frequencies from a larger corpus. The topic modeling community has heretofore focused on scalability to large corpora of hundreds-of-thousands to millions of contexts. There has been less focus on context length.

4.3.2 Emperical Analysis

The analysis below demonstrates the use of R^2 for topic models on a real world data set. The analyses above demonstrated the properties of R^2 with perfectly-specified models. The real world is far messier with the number of topics and adequate prior specifications unknown. The analysis below compares R^2 and the log likelihood at picking the number of topics and calculates the partial R^2 of each topic in a 50-topic model.

Data Description

The data set consists of a random sample of 1,000 abstracts of research grants awarded by the National Institutes of Health (NIH) in 2014. The NIH invests roughly $32 billion annually in biomedical research [114]. After applying common preprocessing steps (described below) the data set contains 8,903 unique words with a mean of 226.7 words and a median of 223 words per abstract (i.e., context).

Modeling Choices

All text is converted to lower case, common stop words are removed, and punctuation is stripped. The vocabulary is pruned on a per-document level, rather than for the whole corpus. Words that appear only once in a document are removed from that document's vocabulary.

I construct ten LDA models from $K = 50$ to $K = 500$ topics, increasing by 50 topics each time. I use symmetric priors $\alpha_k = 0.1$ and $\eta_v = 0.05$. This corresponds to $\sum_k \alpha_k \in \{5, 10, 15, ..., 45, 50\}$ and $\sum_v \eta_v = 445.15$. The Gibbs sampler is run for 200 iterations and posteriors are averaged over the last 50 iterations[3].

Results

R^2 increases with the *estimated* number of topics using LDA. This indicates a risk of model overfit with respect to the number of estimated topics. To evaluate the effect of R^2 on the number of estimated topics, LDA models were fit to two corpora. The first corpus is simulated, using the parameter defaults: $K = 50$, $D = 2{,}000$, $V = 5{,}000$, and $\lambda = 500$. The second corpus is on the abstracts of 1,000 randomly-sampled

[3]It is common to burn in half of the total number of iterations. I chose to burn in over the last quarter for computational reasons. Specifically, I wanted at least 150 burn in iterations but for time complexity reasons, limited the total number of iterations to 200.

research grants for fiscal year 2014 in the National Institutes of Health's ExPORTER database. In both cases, LDA models are fit to the data estimating a range of K. For each model, R^2 and the log likelihood are calculated. The known parameters for this NIH corpus are $D = 1{,}000$, $V = 8,751$, and the median document length is 222 words. This vocabulary has standard stop words removed,[4] is tokenized to consider only unigrams, and excludes tokens that appear in fewer than 2 documents.

The same likelihood calculation is used here, as described above. This likelihood calculation excludes prior information typically used when calculating the likelihood of an LDA model. (See, for example, the likelihood used in [60].) However, this prior information is excluded here for two reasons. First, it is consistent with the method used for calculating McFadden's R^2 earlier in this chapter. Second, this log likelihood can be calculated exactly. The likelihood of an LDA model including the prior is contained within an intractable integral. Various approximations have been developed [106]. However, there is some risk that a comparison of an approximated likelihood to R^2 may be biased by the approximation method.

[4] 175 English stop words from the snowball stemmer project. `http://snowball.tartarus.org/algorithms/english/stop.txt`

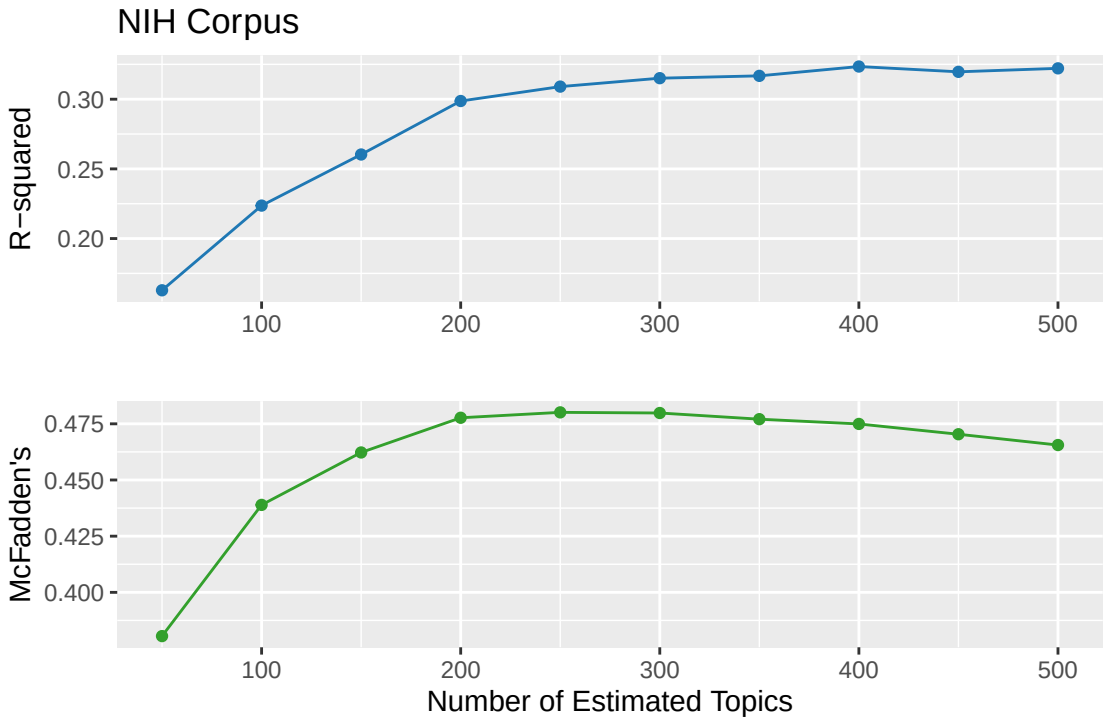


Figure 4.3: Comparison of R-squared and McFadden's R-squared for LDA models fit on the NIH corpus over a range of topics. R-squared and McFadden's R-squared increase with the number of estimated topics, peaking for 350 topics and 300 topics, respectively. In both cases, the metric tapers slightly after its peak. McFadden's is higher than the standard R-squared for all values.

One can calculate a partial R^2 for topic models, similar to a partial coefficient of determination for standard linear models. The partial coefficient of determination is calculated as

$$R^2_{partial} = 1 - \frac{SSR_{(full)}}{SSR_{(reduced)}} \tag{4.17}$$

where $SSR_{(full)}$ is the standard SSR above and $SSR_{(reduced)}$ is the residual sum of squares with topic k removed.

A key difference between a partial R^2 for topic models and the one for standard linear models is how one obtains $SSR_{(reduced)}$. For a standard linear model, the variables are explicit. A variable is removed, a new model fitted, and $SSR_{(reduced)}$ calculated. However topics are latent variables and topic models like LDA are initialized at random during fitting. So instead of fitting a new model, the k-th column is removed from $\mathbf{\Theta}$

and k-th row removed from $\boldsymbol{B}$. The rows of $\boldsymbol{\Theta}$ are normalized to sum to 1. Then $SSR_{(reduced)}$ is calculated from the reduced $\boldsymbol{\Theta}$ and $\boldsymbol{B}$.

The left of Figure 4.4 plots the partial R^2 for the 10 topics contributing the most to the goodness of fit for a 50-topic model on the sample of NIH grant abstracts. The right of Figure 4.4 plots the correlation between probabilistic coherence and the partial R^2 of topics in the same 50-topic model. There is a slight positive correlation between a topic's partial R^2 and its probabilistic coherence in this model. I leave a deeper exploration between the two variables' relationship to future research.

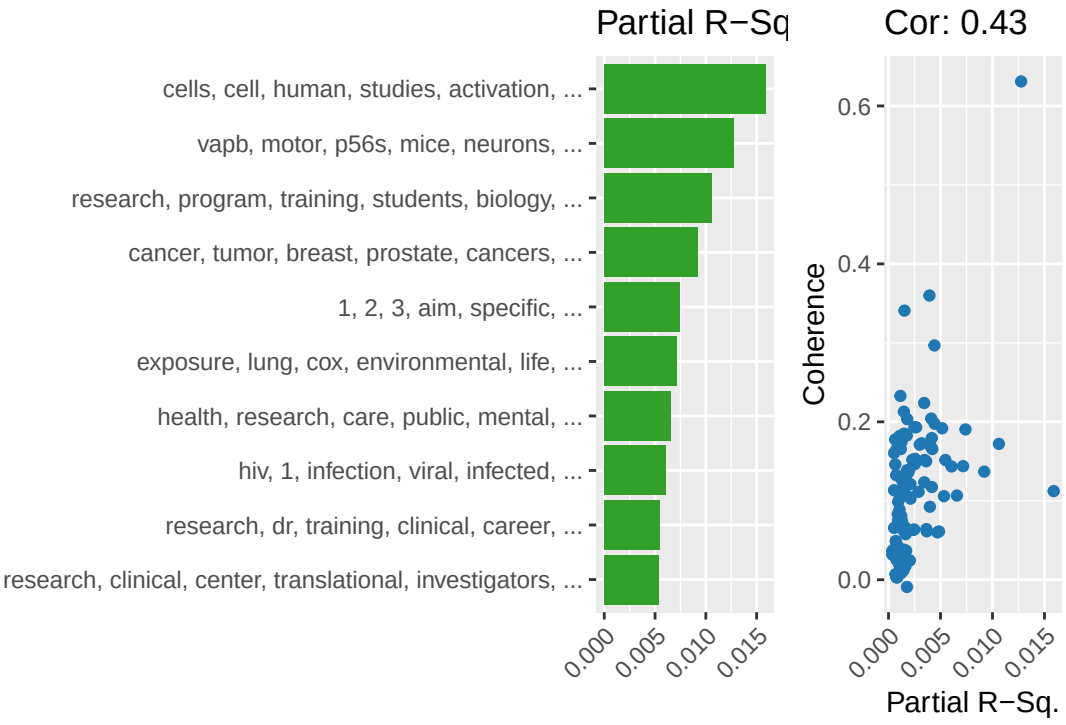


Figure 4.4: 10 topics with the highest partial R^2 from a 50-topic model of the NIH 2014 corpus.

4.4 Discussion

R^2 has many advantages over standard goodness of fit measures commonly used in topic modeling. Current goodness of fit measures are difficult to interpret, to compare across corpora, and to explain to lay audiences.

R^2 does not have any of these issues. Its scale is effectively bounded between 0 and 1, as negative values (though possible) indicate extreme model misspecification. R^2 may be used to compare models of different corpora, if necessary. Scientifically-literate lay audiences are almost uniformly familiar with R^2 in the context of linear regression; the topic model R^2 has a similar interpretation, making it an intuitive measure.

The standard (geometric) interpretation of R^2 is preferred to McFadden's pseudo R^2. The effective upper bound for McFadden's R^2 is considerably smaller than 1. A scale correction measure is needed. Also, it is debatable which likelihood calculation(s) are most appropriate. These issues make McFadden's R^2 complicated and subjective. However, a primary motivation for deriving a topic model R^2 is to remove the complications that currently hinder evaluating and communicating the fidelity with which topic models represent observed text. Most problematically, McFadden's R^2 varies with the number of *true* topics in the data. It is therefore unreliable in practice where the true number of topics in unknown.

Lack of consistent evaluation metrics has limited the use of topic models as a mature statistical method. The development of an R^2 for topic modeling is no silver bullet. However, it represents a step towards establishing consistency and rigor in topic modeling. Going forward, I propose reporting R^2 as a standard metric alongside topic models, as is typically done with ordinary least squares regression.

Chapter 5: Fine Tuning Latent Dirichlet Allocation for Transfer Learning

As stated in Section 1, statistical properties of language make the fine tuning paradigm of transfer learning attractive for analyses of corpora. The pervasiveness of power laws in language—the most famous example of which is Zipf's law [51]—mean that we can expect just about any corpus of language to not contain information relevant to the analysis. (i.e., Linguistic information necessary for understanding the corpus of study would be contained in a super set of language around—but not contained in—said corpus.)

Intuitively, when people learn a new subject by reading, they do it in a way that on its surface appears consistent with fine tuning transfer learning. Humans have general competence in a language before consulting a corpus to learn a new subject. This person then reads the corpus, learns about the subject, and in the process updates and expands their knowledge of the language. Also as stated in Section 1, LDA has attractive properties as a model for statistical analyses of corpora. Its very nature allows us to use probability theory to guide model specification and quantify uncertainty around claims made with the model.

This chapter introduces *tLDA*, short for transfer-LDA. tLDA enables use cases for fine-tuning from a base model with a single incremental update (i.e., "fine tuning") or with many incremental updates—e.g., on-line learning, possibly in a time-series context—using Latent Dirichlet Allocation. This work uses collapsed Gibbs sampling to implement tLDA [39] but tLDA should be able to be implemented in other MCMC methods [57] [58] [59] [60]. tLDA is available for the R language for statistical computing [32] in the *tidylda* package [115].

5.1 Related Work

Work on transfer learning with LDA and other probabilistic topic models falls into three categories. The first category contains topic models that explicitly model a topic's evolution over time [65] [116] [117]. These models differ from true transfer learning in that time is explicitly part of the model, rather than being updated post-hoc. The second category contains models that allow external information to guide the development of topics. External information may be in the form of supervised outcomes [66] [118] [119], introduced by

model structure [120], introduced in the prior [121], or constructed interactively with subject matter experts [122]. The third category contains models designed for incremental and on-line learning [123] [124] [125].

AlSumait, Barbará, and Domeniconi's work, *On-Line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking* [123], is especially influential in developing tLDA. Their approach encodes the topics from the previous time slices into the prior, η, for the next time slice. More formally

$$\eta_k^{(t)} = \omega^{(\delta)} \cdot \beta_k^{(t-1)} \tag{5.1}$$

where $\eta_k^{(t)}$ is the prior for the k-th topic for the current model, $\omega^{(\delta)}$ is a vector of weights with one entry for the previous δ models, and $\beta_k^{(t-1)}$ is a *matrix* whose rows correspond to the expected value of the posterior distribution of the k-th topic for the previous δ models. In essence, $\eta_k^{(t)}$ is a weighted linear combination of the last δ posterior distributions for the k-th topic. The weights sum to 1 such that $\sum_{j=t-\delta-1}^{t-1} \omega_j = 1$. There are two mechanical differences between On-Line LDA and tLDA. First, when $\delta > 1$, On-Line LDA loses the Markov property and can only be applied to time series use cases. Second, if $\delta = 1$, then $\omega = 1$. tLDA does not have a restriction on the size of the weight.

5.2 Contribution

tLDA is a model for updating topics in an existing model with new data, enabling incremental updates, and time-series use cases. It has three characteristics differentiating it from previous work:

1. Flexibility - Most prior work can only address use cases from one of the above categories of related work. In theory, tLDA can address all three. However, exploring use of tLDA to encode expert input into the η prior is left for the future.
2. Tunability - tLDA introduces only a single new tuning parameter, a. Its use is intuitive, balancing the ratio of tokens in $\boldsymbol{X}^{(t)}$ to the base model's data, $\boldsymbol{X}^{(t-1)}$.
3. Analytical - tLDA allows data sets and model updates to be chained together preserving the Markov property, enabling analytical study through incremental updates.

5.3 tLDA

5.3.1 The Model

Formally, tLDA is:

$$z_{d_n}|\boldsymbol{\theta}_d \sim \text{Categorical}(\boldsymbol{\theta}_d) \tag{5.2}$$

$$w_{d_n}|z_k, \boldsymbol{\beta}_k^{(t)} \sim \text{Categorical}(\boldsymbol{\beta}_k^{(t)}) \tag{5.3}$$

$$\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha}_d) \tag{5.4}$$

$$\boldsymbol{\beta}_k^{(t)} \sim \text{Dirichlet}(\omega_k^{(t)} \cdot \mathbb{E}\left[\boldsymbol{\beta}_k^{(t-1)}\right]) \tag{5.5}$$

The above indicates that tLDA places a matrix prior for words over topics where $\eta_{k,v}^{(t)} = \omega_k^{(t)} \cdot \mathbb{E}\left[\beta_{k,v}^{(t-1)}\right] = \omega_k^{(t)} \cdot \frac{Cv_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)}}{\sum_{v=1}^{V} Cv_{k,v}^{(t-1)}}$. Because the posterior at time t depends only on data at time t and the state of the model at time $t-1$, tLDA models retain the Markov property.

Selecting the prior weight

Each $\omega_k^{(t)}$ tunes the relative weight between the base model (as prior) and new data in the posterior for each topic. This specification introduces K new tuning parameters and setting $\omega_k^{(t)}$ directly is possible but not intuitive. However, after introducing a new parameter, we can algebraically show that the K tuning parameters collapse into a single parameter with several intuitive critical values. This tuning parameter, $a^{(t)}$, is related to each $\omega_k^{(t)}$ as follows:

$$\omega_k^{(t)} = a^{(t)} \cdot \sum_{v=1}^{V} Cv_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)} \tag{5.6}$$

Appendix D shows the full derivation of the relationship between $a^{(t)}$ and $\omega_k^{(t)}$.

When $a^{(t)} = 1$, fine tuning is equivalent to adding the data in $\boldsymbol{X}^{(t)}$ to $\boldsymbol{X}^{(t-1)}$. In other words, each word occurrence in $\boldsymbol{X}^{(t)}$ carries the same weight in the posterior as each word occurrence in $\boldsymbol{X}^{(t-1)}$. If $\boldsymbol{X}^{(t)}$ has more data than $\boldsymbol{X}^{(t-1)}$, then it will carry more weight overall. If it has less, it will carry less.

When $a^{(t)} < 1$, then the posterior has recency bias. Each word occurrence in $\boldsymbol{X}^{(t)}$ carries more weight than each word occurrence in $\boldsymbol{X}^{(t-1)}$. When When $a^{(t)} > 1$, then the posterior has precedent bias. Each word

occurrence in $\boldsymbol{X}^{(t)}$ carries less weight than each word occurrence in $\boldsymbol{X}^{(t-1)}$.

Another pair of critical values are $a^{(t)} = \frac{N^{(t)}}{N^{(t-1)}}$ and $a^{(t)} = \frac{N^{(t)}}{N^{(t-1)}+\sum_{d,v}\eta_{d,v}}$, where $N^{(\cdot)} = \sum_{d,v} X^{(\cdot)}_{d,v}$. These put the total number of word occurrences in $\boldsymbol{X}^{(t)}$ and $\boldsymbol{X}^{(t-1)}$ on equal footing excluding and including $\boldsymbol{\eta}^{(t-1)}$, respectively. These values may be useful when comparing topical differences between a baseline group in $\boldsymbol{X}^{(t-1)}$ and "treatment" group in $\boldsymbol{X}^{(t)}$, though this use case is left to future work.

5.3.2 The *tidylda* Implementation of tLDA

The *tidylda* package—described in Chapter 6—implements an algorithm for tLDA in 6 steps.

1. Construct $\boldsymbol{\eta}^{(t)}$
2. Predict $\hat{\boldsymbol{\Theta}}^{(t)}$ using topics from $\hat{\boldsymbol{B}}^{(t-1)}$
3. Align vocabulary
4. Add new topics
5. Initialize $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$
6. Begin Gibbs sampling with $P(z = k) = \frac{Cv_{k,n}+\eta_{k,n}}{\sum_{v=1}^{V} Cv_{k,v}+\eta_{k,v}} \cdot \frac{Cd_{d,k}+\alpha_k}{(\sum_{k=1}^{K} Cd_{d,k}+\alpha_k)-1}$

Step 1 uses the relationship in (5.6), above. Step 2 uses a standard prediction method for LDA models. The *tidylda* implementation uses a dot-product prediction described in Appendix A for speed. MCMC prediction would work as well.

Any real-world application of tLDA presents several practical issues which are addressed in steps 3 - 5, described in more detail below. These issues include: the vocabularies in $\boldsymbol{X}^{(t-1)}$ and $\boldsymbol{X}^{(t)}$ will not be identical; users may wish to add topics, expecting $\boldsymbol{X}^{(t)}$ to contain topics not in $\boldsymbol{X}^{(t-1)}$; and $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$ should be initialized proportional to $\boldsymbol{Cd}^{(t-1)}$ and $\boldsymbol{Cv}^{(t-1)}$, respectively.

Aligning Vocabulary

The *tidylda* package implements an algorithm to fold in new words. This method slightly modifies the posterior probabilities in $\boldsymbol{B}^{(t-1)}$ and adds a non-zero prior by modifying $\boldsymbol{\eta}^{(t)}$. It involves three steps. First, append columns to $\boldsymbol{B}^{(t-1)}$ and $\boldsymbol{\eta}^{(t)}$ that correspond to out-of-vocabulary words. Next, set the new entries for these new words to some small value, $\varepsilon > 0$ in both $\boldsymbol{B}^{(t-1)}$ and $\boldsymbol{\eta}^{(t)}$. Finally, re-normalize the rows of $\boldsymbol{B}^{(t-1)}$ so that they sum to one. For computational reasons, ε must be greater than zero. Specifically, the *tidylda*

implementation chooses ε to the lowest decile of all values in $\boldsymbol{B}^{(t-1)}$ or $\boldsymbol{\eta}^{(t)}$, respectively. This choice is somewhat arbitrary. I have not performed a sensitivity analysis to the choice of ε. Yet I hypothesize that if ε is sufficiently small, there should not be a significant effect on the posterior. The other entries of $\boldsymbol{B}^{(t-1)}$ and $\boldsymbol{\eta}^{(t)}$ should swamp the magnitude of either vector. The lowest decile of a power law seems like a sufficiently small choice.

Adding New Topics

tLDA employs a similar method to add new, randomly initialized, topics if desired. This is achieved by appending rows to both $\boldsymbol{\eta}^{(t)}$ and $\boldsymbol{B}^{(t)}$, adding entries to $\boldsymbol{\alpha}$, and adding columns to $\boldsymbol{\Theta}^{(t)}$, obtained in step two above. The tLDA implementation in *tidylda* sets the rows of $\boldsymbol{\eta}^{(t)}$ equal to the column means across previous topics. Then new rows of $\boldsymbol{B}^{(t)}$ are the new rows of $\boldsymbol{\eta}^{(t)}$ but normalized to sum to one. This effectively sets the prior for new topics equal to the average of the weighted posteriors of pre-existing topics.

The choice of setting the prior to new topics as the average of pre-existing topics is admittedly subjective. A uniform prior over words is unrealistic, being inconsistent with Zipf's law [51]. (See Chapter 3.) The average over existing topics is only one viable choice. Another choice might be to choose the shape of new $\boldsymbol{\eta}_k$ from an estimated Zipf's coefficient of $\boldsymbol{X}^{(t)}$ and choose the magnitude by another means. I leave this exploration for future research.

New entries to $\boldsymbol{\alpha}$ are initialized to be the median value of the pre-existing topics in $\boldsymbol{\alpha}$. Similarly, columns are appended to $\boldsymbol{\Theta}^{(t)}$. Entries for new topics are taken to be the median value for pre-existing topics on a per-document basis. This effectively places a uniform prior for new topics. This choice is also subjective. Other heuristic choices may be made, but it is not obvious that they would be any better or worse choices. I also leave this to be explored in future research.

Initializing $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$

Like most other LDA implementations, *tidylda*'s tLDA initializes tokens for $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$ with a single Gibbs iteration. However, instead of sampling from a uniform random for this initial step, a topic for the *n*-th

word of the d-th document is drawn from the following:

$$P(z_{d_n} = k) = \hat{\beta}^{(t)}_{k,n} \cdot \hat{\theta}^{(t)}_{d,k} \tag{5.7}$$

where $\hat{\beta}^{(t)}_{k,n}$ is the estimated probability of choosing the n-th word from topic k in the t-th period and $\hat{\boldsymbol{\theta}}^{(t)}_{d,k}$ is the estimated probability of choosing the k-th topic from the d-th document in the t-th period.

After a single iteration, the number of times each topic was sampled at each document and word occurrence is counted to produce $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$. After initialization where topic-word distributions are fixed, MCMC sampling then continues in a standard fashion, recalculating $\boldsymbol{Cd}^{(t)}$ and $\boldsymbol{Cv}^{(t)}$ (and therefore $P(z_{d_n} = k)$) at each step.

5.4 Experiments

To evaluate tLDA, I conduct two simulation experiments and an analysis on a real-world data set. The simulation experiments evaluate the degree to which tLDA converges towards a ground truth data generating distribution. The real-world data analysis constructs time series of topics from a data set of Small Business Innovation Research (SBIR) grant abstracts from 1983 to 2021.

5.4.1 Simulation Analysis

Under two assumptions, tLDA should converge to stable topic estimates as it is fine tuned on more data. Specifically, first, if $\boldsymbol{X}^{(i)}$ and $\boldsymbol{X}^{(j)}$ are generated from the same distribution, $\forall\ i, j \geq 0, i \neq j$, and, second, if $a \geq 1$, then tLDA should converge to stable topic estimates. However, unless a user correctly chooses $\boldsymbol{\alpha}$, $\boldsymbol{\eta}^{(0)}$, and K there is no guarantee that tLDA will converge to *correct* topic estimates. This is not a limitation of tLDA as selecting hyper-parameters for any LDA model remains an open problem. Yet, I hypothesize that there are heuristics one can use to get reasonably good estimates, so long as the two assumptions above hold.

To test the above, I conduct two simulation experiments. I generate topics by drawing from 128 data generating distributions. Then, for each data generating distribution, I sample 100 corpora of 100 contexts each. I fit a model on the first 100 contexts of the data and iteratively add the remaining 100 contexts at a time, fine tuning at each stage, with values of a ranging from 0.2 to 2. The experiments are described in more

detail in the below sections.

On average, models converge towards the true topics of the data generating process when values of a are greater than 0.8. When values of a are less than 0.8, the models tend not to converge as there is high variance between estimates. There is very high variance as to whether or not models converge when a is around 0.8. Properties of the data generating process, particularly the magnitude of $\boldsymbol{\alpha}$, also influence whether or not the models converge. This points towards a need for more study of LDA as a data generating process of language, rather than an inherent limitation of tLDA.

Synthetic Data Generation

I assume that the total vocabulary size, V, and number of topics, K, are fixed with value of $5,000$ and 25 respectively. I further assume that $\boldsymbol{\alpha}$ is flat (symmetric) and that $\boldsymbol{\eta}$ is proportional to a power law (asymmetric) to account for Zipf's law [51]. I vary the magnitudes of $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ and vary the number of words sampled in each context.

Specifically, I chose the following:

1. $\left(\sum_{v=1}^{V} \eta_v\right) \in \{50, 250, 500, 1000\}$
2. $\left(\sum_{k=1}^{K} \alpha_k\right) \in \{0.5, 1, 3, 5\}$
3. $N_d \sim \text{Pois}(\lambda)$ where $\lambda \in \{50, 100, 200, 400\}$
4. $V = 5000$ and $K = 25$

The above creates 64 unique sets of hyper-parameters. From each, I generate two sets of population statistics, $\{\boldsymbol{\Theta}, \boldsymbol{B}\}$, leaving 128 unique sets of population parameters to parametarize the generation process described in Section 1.1. From each of these sets of population parameters, i, 100 word count matrices, $\boldsymbol{X}_i^{(t)}$, with 100 contexts each are sampled.

The magnitudes of the Dirichlet hyper-parameters, $\boldsymbol{\eta}$ and $\boldsymbol{\alpha}$ vary while keeping their shapes fixed. The magnitude of the parameter of a Dirichlet distribution plays a strong role in tuning the covariance between draws from that distribution. A smaller magnitude means larger variability between draws. In other words, if $\left(\sum_{v=1}^{V} \eta_v\right)$ is smaller, topics are less linguistically similar. If $\left(\sum_{v=1}^{V} \eta_v\right)$ is larger, topics are more linguistically similar. Similarly, $\left(\sum_{k=1}^{K} \alpha_k\right)$ tunes the topical similarity of contexts.

Modeling Choices

I fit a total of 128,000 LDA models to the simulated data. For each of the 128 unique sets of population parameters, I sequentially fit models on 100 batches of data, using tLDA to update the parameters. I do this for 10 different settings for a from $a = 0.2$ to $a = 2$ with a step size of 0.2. This leads to 1,280 separate model chains.

In each case, the base model is fit with $K = 30$ topics, and commonly-used flat priors with $\alpha_k = 0.05$ and $\eta_v = 0.01$. The Gibbs sampler runs for 200 iterations. Posterior estimates are taken averaging over the last 50 iterations. Subsequent models are fine tuned with an additional 200 iterations with posterior estimates averaged over the last 50 and a set as described above[1].

I fit more topics than in the population data for two reasons. First, it's unrealistic that a researcher will know the right number of topics a-priori, and in practice a true number likely does not exist. Second, fitting a fewer number of topics than is in the population would lead to a pathologically-mispecified model regardless of other choices.

Evaluation Method

I evaluate the models using pairwise Hellinger distance [126] calculated from the estimated topics to each of the ground truth simulated topics. For estimated topic $\hat{\beta}_k^{(t)}$ and ground-truth topic $\beta_{\mathfrak{k}}$, the Hellinger distance is calculated as

$$H(\hat{\beta}_k^{(t)}, \beta_{\mathfrak{k}}) = \sqrt{\frac{1}{2} \sum_{v=1}^{V} \left(\sqrt{\hat{\beta}_{k,v}^{(t)}} - \sqrt{\beta_{\mathfrak{k},v}} \right)^2} \tag{5.8}$$

Note that it is not necessarily true that $k = \mathfrak{k}$ for any pair of topics. For simplicity, I denote the above $H_{k,\mathfrak{k}}^{(t)}$ and its derivative with respect to t as $H_{k,\mathfrak{k}}^{'(t)}$.

The definition of convergence in probability for random variables on a separable metric space means that if

$$\lim_{t \to \infty} P(H_{k,\mathfrak{k}}^{'(t)} \geq \varepsilon) = 0 \tag{5.9}$$

[1] It is common to burn in half of the total number of iterations. I chose to burn in over the last quarter for computational reasons. Specifically, I wanted at least 150 burn in iterations but due to computation time, limited the total number of iterations to 200.

then topic $\beta_k^{(t)}$ converges in probability to $\beta_{\mathfrak{k}}$. (5.9) is equivalent to

$$\lim_{t\to\infty} H_{k,\mathfrak{k}}^{(t)} = \varepsilon \tag{5.10}$$

And since ε is a constant, that would imply

$$\text{If } \hat{\beta}_k^{(t)} \text{ converges in probability towards a stable posterior, then } \lim_{t\to\infty} H_{k,\mathfrak{k}}^{'(t)} = 0 \tag{5.11}$$

Note that if (5.11) holds, that does not imply that $\hat{\beta}_k^{(t)}$ is a consistent estimator of $\beta_{\mathfrak{k}}$. For consistency, ε would have to be zero. This is unlikely to be true as an imperfect specification of α and η mean that $H_{k,\mathfrak{k}}^{(t)}$ will always be positive. However, if $\hat{\beta}_k^{(t)}$ is heading *towards* $\beta_{\mathfrak{k}}$, then $H_{k,\mathfrak{k}}^{'(t)} < 0$ as it approaches its limit. I approximate $H_{k,\mathfrak{k}}^{'(t)}$ by $H_{k,\mathfrak{k}}^{(t)} - H_{k,\mathfrak{k}}^{(t-1)}$, normalized by $H_{k,\mathfrak{k}}^{(0)}$.

Assessing (5.11) is a two step process, first assessing the convergence of each estimated topic and then assessing the convergence of the model as a whole. To assess the convergence of a topic, I perform a t-test on the last 20 periods of $H_{k,\mathfrak{k}}^{'(t)}$, with a null hypothesis of the mean being unequal to zero at the 0.05 significance level. If I reject the null, I consider a topic's estimate to have converged. If the null is true for all topics, we'd still expect a 5% rejection rate by random chance.

The second step corrects for multiple tests and assesses the convergence of the model as a whole. I perform a binomial test on the proportion of topics whose null hypothesis is rejected in the first step. Specifically, I test the null hypothesis that the true probability of rejection in step one is less than or equal to 5%. The binomial test is also done at the 0.05 significance level.

I use logistic regression to analyze which factors lead to model convergence or divergence. I use the hyper-parameters that define the data generating process (described in Chapter 2) and the value of a used in fine tuning that model as predictors. The outcome variable is binary indicating whether the model has converged or not.

Assessing whether or not the model converges *towards* the data generating distribution is a similar two step process. A topic converges in probability towards the true generating distribution if the average slope of $H_{k,\mathfrak{k}}^{(t)}$ is negative, $\mathfrak{k}$ indexes the topic closest to k, averaging over the last. After discarding the first two periods,

a t-test with a null hypothesis that the average of $H^{\prime(t)}_{k,\mathfrak{k}}$ from periods 3 to 100 is greater than zero. I consider the whole model to have converged towards the true data generating distribution using a binomial test, as above. Only models considered to have converged, using the procedure defined above, are considered.

I use linear regression to analyze which factors lead to fitted topics being closer to their matches in the true data generating distributions. I use the hyper-parameters that define the data generating process (described in Chapter 2) and the value of a used in fine tuning that model as predictors. The outcome variable is the hellinger distance between matched topics averaged over the last 10 periods.

Results

Models where $a < 0.8$ almost never converge; 384 out of 385 such models diverged. Models where $a > 0.8$ tend to converge; 94% of such models converged. There is high variability in convergence where $a = 0.8$. Roughly 1/3 of the models converged when $a = 0.8$, the other 2/3 diverged. Figure 5.1, below, shows a box plot of p-values in the binomial test for each value of a. There is a clear structural break at $a = 0.8$, reflecting the clear patterns of convergence/divergence on either side of 0.8 and a high variance of convergence at 0.8.

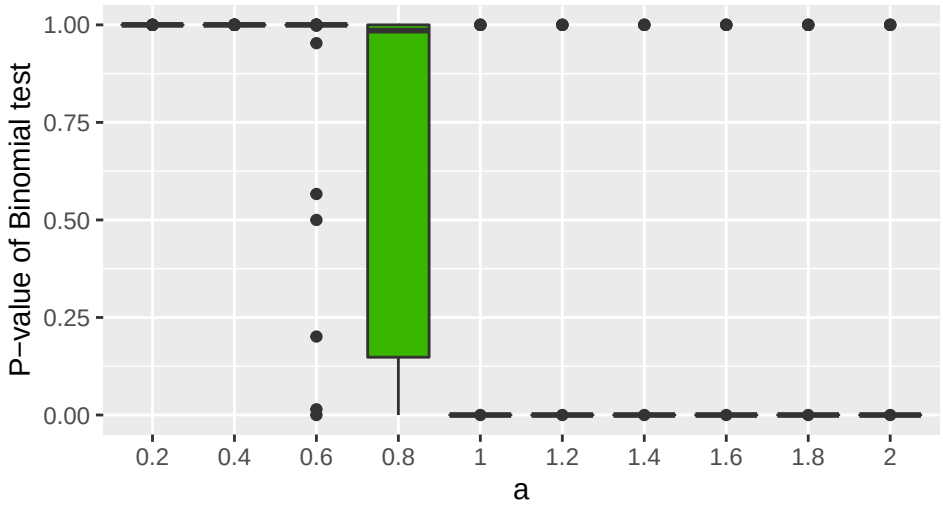


Figure 5.1: Boxplots of the p-values assessing convergence for each value of a. Below $a = 0.8$ almost all p-values are near one. Above $a = 0.8$ almost all p-values are near zero, with a few exceptions. But at $a = 0.8$ there is high variance in the p-values assessing model convergence.

I introduced additional models for $a = 0.65$ to $a = 0.95$—with a step size of 0.05—to further examine the region around $a = 0.8$. The transition between divergence to convergence happens quickly as shown at the top of Figure 5.2. And the bottom of Figure 5.2 reveals that $a = 0.8$ is the point of maximum variance in p-values of tests measuring whether or not a model converges.

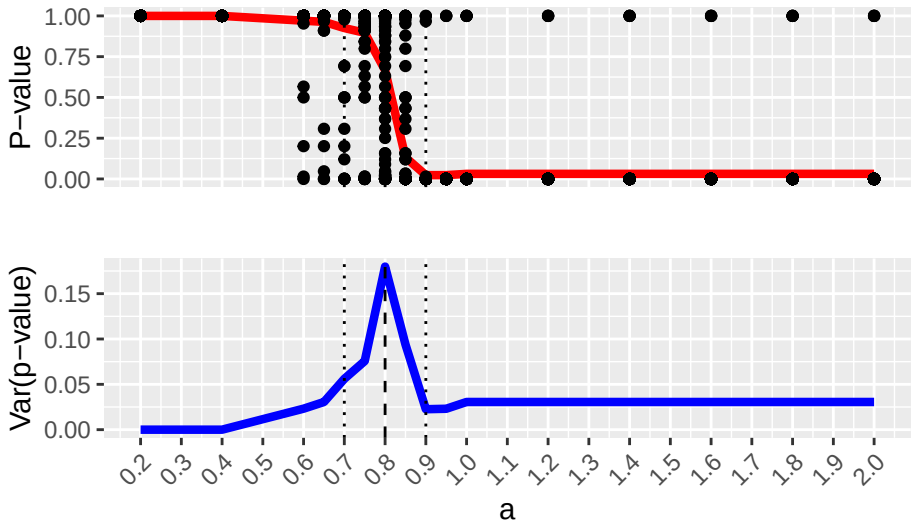


Figure 5.2: P-values assessing convergence for each value of a with the mean in read (top) and the variance in blue (bottom). Individual observations are black points (top).

Table 5.1 contains the results of four regression models. The models labeled "Convergence" represent logistic regression models predicting whether each of the model chains converged ($N = 1,280$). The models labeled "Direction" represent linear regression models predicting the Hellinger distance of estimated topics from converged models matched to data-generating topics, averaged over the last 20 periods and multiplied by 100 for scale ($N = 22,920$). Each of the models regresses their outcomes on a and the hyper-parameters of the data generating distributions with and without quadratic effects. Both logistic models have an in-sample accuracy of 93% and the linear models have an adjusted R^2 of 0.90 and 0.72 with and without quadratics, respectively. Asterisks represent the p-value associated with each coefficient: $* \equiv p \leq 0.01$ and $. \equiv p \leq 0.05$.

In all cases, a has the most substantial impact on the outcome. If a is too small, a chain of models is

Table 5.1: Effects of a on convergence

Variable	Convergence		Convergence		Direction		Direction	
Intercept	-11.18	*	-30.46	*	62.05	*	62.92	*
a	12.75	*	50.53	*	3.12	*	10.40	*
Avg. Doc. Length	0.00		0.00		0.01	*	0.02	*
Sum(eta)	0.00	*	0.00		-0.02	*	-0.06	*
Sum(alpha)	0.11	.	0.13		-0.09	*	-0.15	*
a Squared			-17.13	*			-2.69	*
Avg. Doc. Length Squared			0.00				0.00	*
Sum(eta) Squared			0.00				0.00	*
Sum(alpha) Squared			0.01				0.00	

unlikely to converge and if a is too large, the chain will converge far from the true data generating topics. As a increases, the probability of a model converging also increases, albeit with decreasing returns as indicated by the decreasing quadratic. Intuitively, larger values of a decrease the variance between models in a chain leading to faster convergence. Yet, of the models that converge, a increases the distance of the fitted topics to their matches in the data generating distribution, again with decreasing returns.

5.4.2 Emperical Analysis

The simulations above examines the properties of tLDA when fine tuning is performed by adding data from the same generating distribution. Simulations are helpful for isolating effects, but the real world data is far messier. This section demonstrates tLDA for topic discovery on real data in a time series context. The analysis consists of two parts: exploring the aggregate effect of varying values of a on the model chains and a more detailed exploration of the model chain when $a = 0.8$.

Data Description

The data set consists of 188,554 titles and abstracts of Small Business Innovation Research (SBIR) and Small Business Technology Transfer (STTR) research grants awarded from 1983 to 2021. The SBIR and STTR programs provide grants to small businesses to encourage them to participate in US federal government research programs. The data set grows from 785 grants in 1983 to an average of 6,377 per year from 2001 to 2020. Only 815 grants had been published for 2021 at the time the data set was downloaded. After applying common preprocessing steps (described below), the data set contained 92,196 unique words across all years. From the charts in Figure 5.3, we can see there is a structural break in the volume of data at the year 2000.

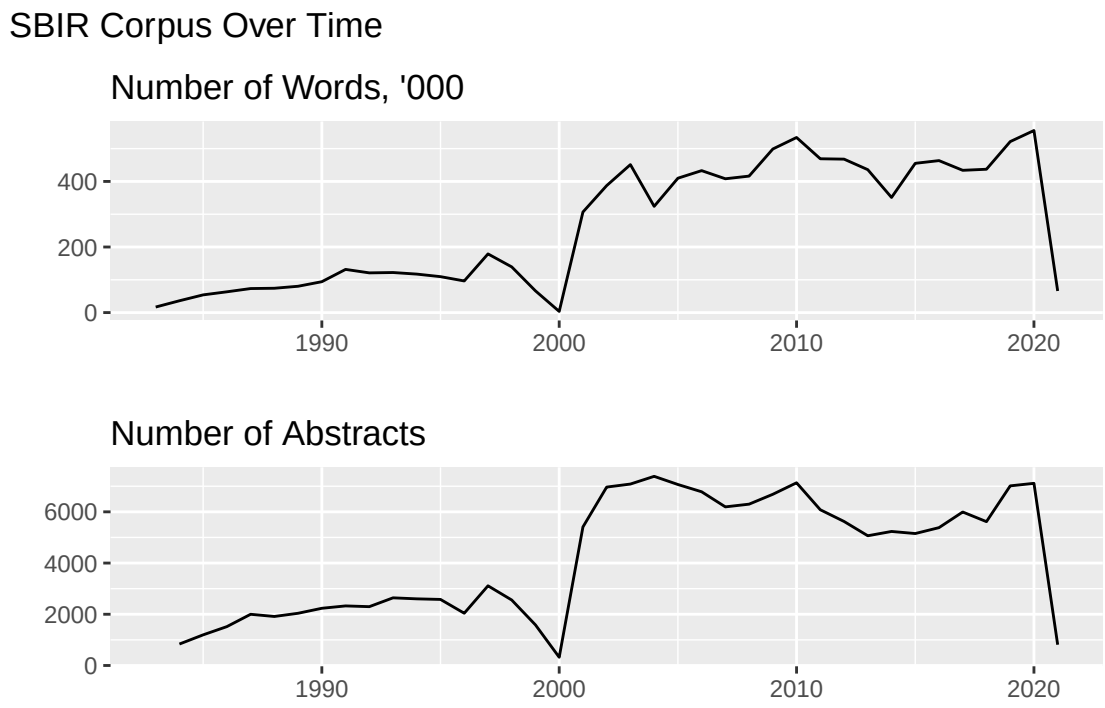


Figure 5.3: Number of abstracts and number of words for the SBIR corpus over time. In 2000, there appears to be a structural break. Only 815 grants had been published for 2021 at the time the data set was downloaded.

Modeling Choices

The data are modeled successively by award year using tLDA. For each year, a document term matrix is constructed using *tidytext* [127]. All text is converted to lowercase, common stop words are removed, and punctuation is stripped. The vocabulary is pruned on a per-document level, rather than for the whole corpus. This prevents unintentional information leakage from period to period. Words that appear only once in a document are removed from that document's vocabulary.

An initial LDA model is constructed for 1983 with $K = 100$ topics and symmetric priors such that $\alpha_k = 0.1$ and $\eta_v = 0.05$. This corresponds to $\sum_k \alpha_k = 10$ and $\sum_v \eta_v = 148.9$, respectively, as 1983 has 2,978 unique words. The Gibbs sampler is run for 200 iterations and posteriors are averaged over the last 50 iterations[2].

For each year from 1984 to 2021, tLDA models are constructed with $a \in \{0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6\}$.

[2]It is common to burn in half of the total number of iterations. I chose to burn in over the last quarter for computational reasons. Specifically, I wanted at least 150 burn in iterations but due to computation time, limited the total number of iterations to 200.

As above, the Gibbs sampler is run for 200 iterations and posteriors averaged over the last 50[3]. For each year, one randomly-initialized topic is added to the model in an attempt to detect emergent topics.

Results

Analysis for varying values of a For each value of a, a chain of 39 models is produced. The first model in the chain has 100 topics and the last has 138. The analysis focuses on four key statistics: R^2, log likelihood, probabilistic coherence (see Appendix E)—or just "coherence"—averaged over all topics, and the Hellinger distance of each topic at time t to itself at time $t-1$ averaged across all topics. R^2 and the log likelihood are model-wide goodness-of-fit metrics. Coherence is a proxy for human interpretability. And mean Hellinger distance measures how much the estimated topics change from year to year.

Figure 5.4 plots these four statistics for each value of a over time. A clear pattern appears for mean Hellinger distance but not for other statistics. Mean Hellinger distance is highest when a is smallest and decrease as a grows, consistent with expectation. When a is small, the prior has less weight in the posterior than the data. This results in a posterior that changes significantly period to period as it is more-strongly shaped by that period's data.

[3]It is common to burn in half of the total number of iterations. I chose to burn in over the last quarter for computational reasons. Specifically, I wanted at least 150 burn in iterations but due to computation time, limited the total number of iterations to 200.

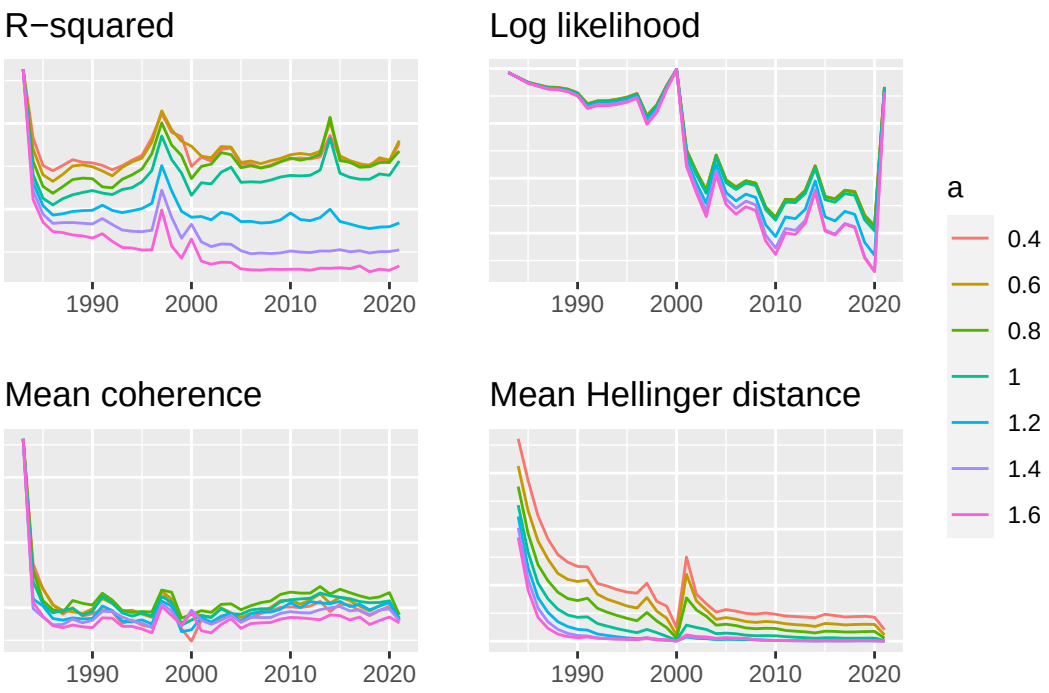


Figure 5.4: Plotted over time, mean Hellinger distance has a clear pattern consistent with expectation. Higher values of a put more mass on the prior, leading to less change year-to-year. The pattern is harder to see for R^2, coherence, and log likelihood.

Figure 5.5 plots the average of each statistic over all years. The pattern for mean Hellinger distance is the same and the patterns for other statistics are now clearer. Log likelihood and R^2 are also highest for low values of a and decrease as a increases. Yet, over $a = 0.4$ to $a = 0.8$, the difference is extremely subtle. The pattern for coherence is somewhat surprising. That coherence is maximized when a = 0.8 may suggest the importance of this value. Further study is warranted on the optimal choice of a.

Based on the peak of coherence and the fact that R^2 and log likelihood are still near their peaks when $a = 0.8$, I choose the model where $a = 0.8$ for deeper analysis.

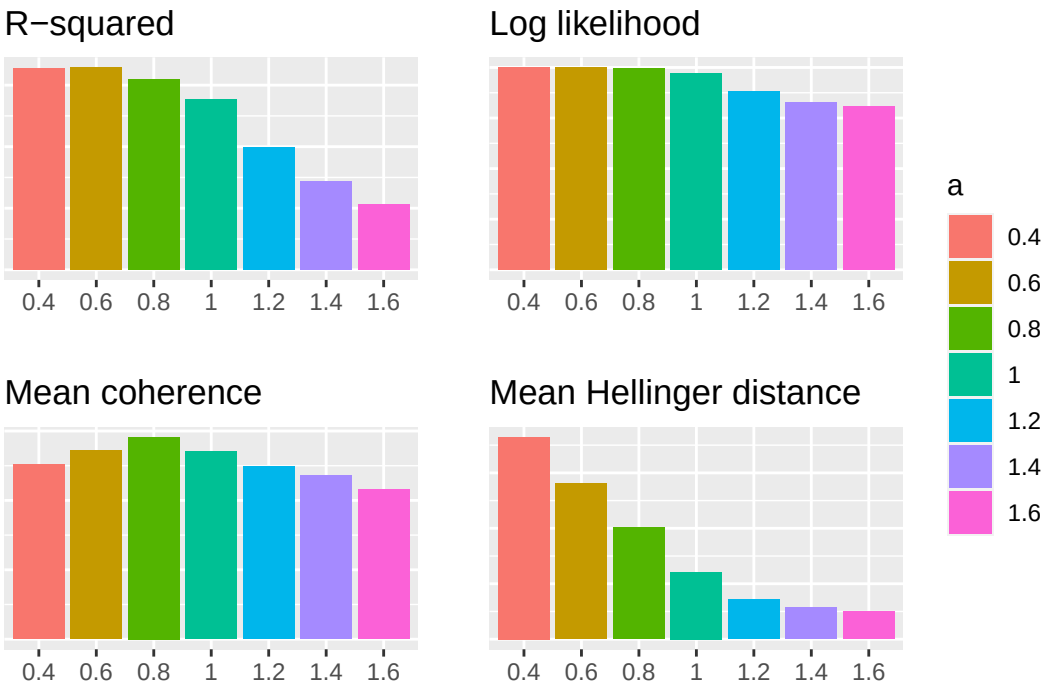


Figure 5.5: Averaging over all years, the patterns for each statistic are clearer. R^2 and log likelihood are decreasing functions of a, but for values less than $a = 1$, differences are very small. Coherence reaches a small peak at $a = 0.8$. Mean Hellinger distance is also a decreasing function of a as expected.

Analysis for $a = 0.8$ Figure 5.6 contains two charts derived from model chains where $a = 0.8$. The top plots the R^2 statistic. The bottom plots probabilistic coherence (Appendix E) averaged across topics. The black line is the actual value of each statistic. The blue line and associated shaded interval is the result of a loess model using default parameters when calling `geom_smooth` from the R package ggplot2 [128].

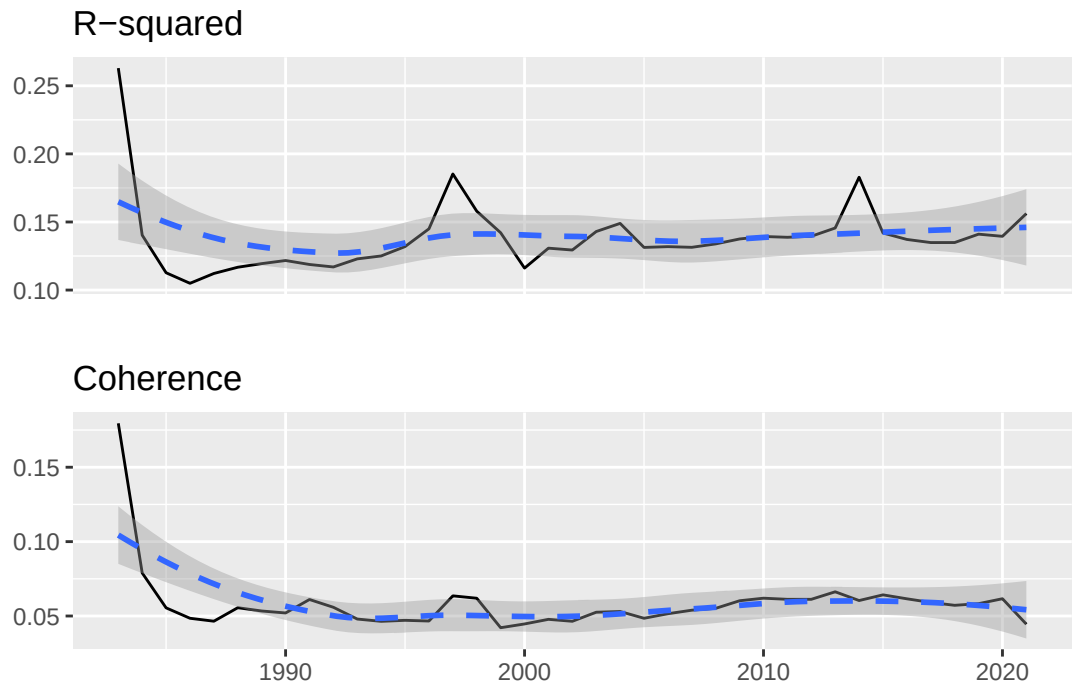


Figure 5.6: Both R^2 and probabilistic coherence are highest in 1983, drop off quickly, and remain flat for remaining time periods. This is likely due to R^2 and coherence being calculated using only data in the current period.

Both show a similar patter. The initial value of the statistic in 1983 is the largest. The statistic then quickly drops off and remains more or less flat over the remainder of the series. I hypothesize that this is due to two factors: the relative effects of the prior and data on the posterior and the fact that R^2 and probabilistic coherence are comparisons of the posterior to the data, disregarding the prior. The initial model has a flat and small prior, its relative weight in the posterior is small compared to the words.[4] But because $a = 0.8$, there is more weight on the prior, relative to the data, as time goes by. The end result is that posterior contains a significant amount of information not in the data at any time period.

I also examine changes in individual topics over time. To showcase one example, Figure 5.7 contains two charts for topic 24. The top chart plots changes in the top 5 words. As the mix of words in the top 5 changes, some lines drop off and new lines emerge. The top 5 words are labeled at the first period where a

[4]Note that this does not imply the prior is non-informative as the non-informative Dirichlet prior would have $\eta_k = 1$ or $\eta_k = \frac{1}{2}, \forall k$.

topic emerges (1983 for topic 24) and in the final period, 2021. The bottom chart plots a topic's prevalence over time using the actual statistic in black and blue loess[5] line with a corresponding shaded interval. The calculation for prevalence is in (5.12), below.

$$\text{Prevalence} \equiv \sum_{d=1}^{D} n_d \cdot \theta_{d,k} \cdot \beta_k \cdot 100 \tag{5.12}$$

Most topics show signs of convergence in their top words, as depicted in the top of Figure 5.7. The top word grows to be much more probable than the second and subsequent words, consistent with Zipf's law.

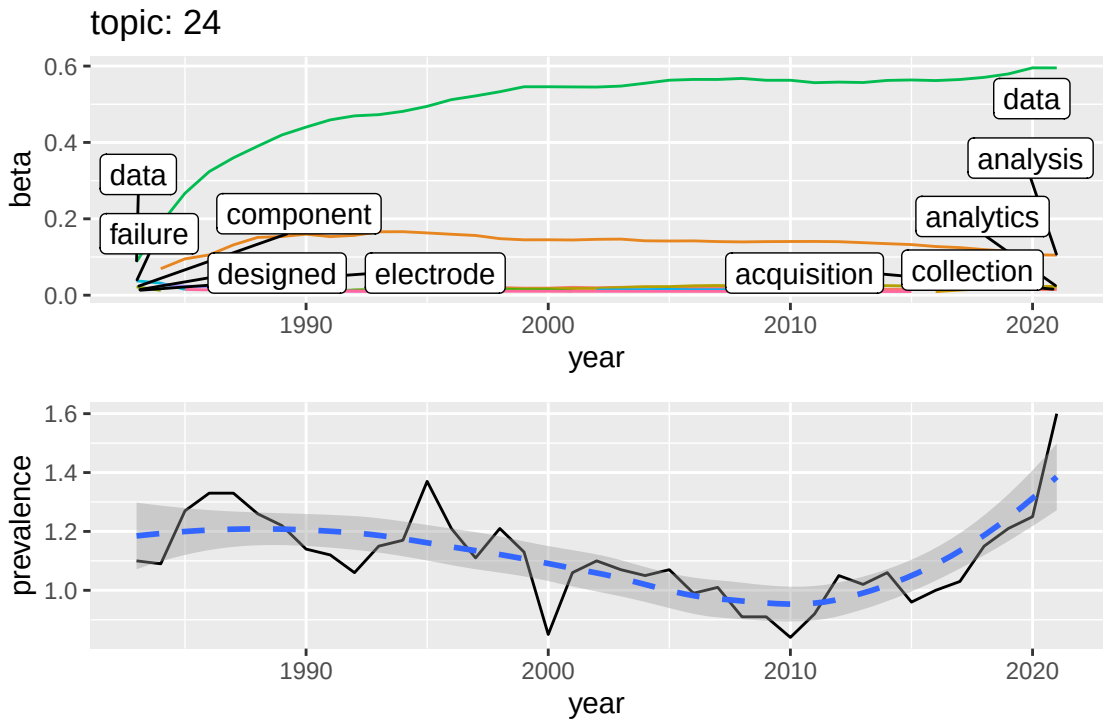


Figure 5.7: Each topic is a time series of its prevalence and distribution of its words. The top depicts changes in the top 5 words for topic 24. The bottom shows changes in prevalence for topic 24 with actual data in black and a blue loess curve with corresponding shaded confidence region.

[5]Locally weighted scatterplot smoothing

Table 5.2: Top 5 Words of Topics that Increased Over Time

Topic	All Years Combined	Only 2021
8	power, efficiency, microwave, 10, output	power, efficiency, systems, voltage, electric
24	data, analysis, failure, plant, collection	data, analysis, analytics, collection, acquisition
50	drug, agents, delivery, drugs, agent	drug, treatment, efficacy, studies, delivery
90	training, technologies, team, simulation, virtual	training, team, virtual, environment, simulation
96	cost, low, density, vehicles, vehicle	cost, low, effective, lower, costs

In most cases, topic prevalence decreases over time. In fact, only 15 of 138 topics show sign of increasing over time on average. These results are obtained by averaging over the slope of the loess curves for prevalence (the dashed blue line in Figure 5.7) for each topic. Of those, only one topic (topic 126) was not in the initial 100 topics covering all years. The top five words for the five topics showing most increase are in Table 5.2. I hypothesize that most topics decreased over time because of the addition of one randomly initialized topic in each year. This is due to two related effects. Having more topics, all else constant, diffuses the probability mass of the posterior over more categories. In the years where a new topic is introduced, more posterior probability mass comes from the data than the prior for these topics, again shifting mass away from established topics.

Yet there is some evidence that adding randomized topics is successful for topic discovery. For example, a topic related to Covid-19 emerges (topic 137, top terms in 2021 {19, covid, opiod, oud, sars}) from a randomly-initialized topic in 2020. Topic 126, which emerged in 2019 and whose words relate to the US Air Force (air, force, uas, solution, benefit), correspond to a major increase in the number of SBIR grants from the US Air Force. Presumably the granting institution is mentioned in the title or abstract. (See Figure 5.8)

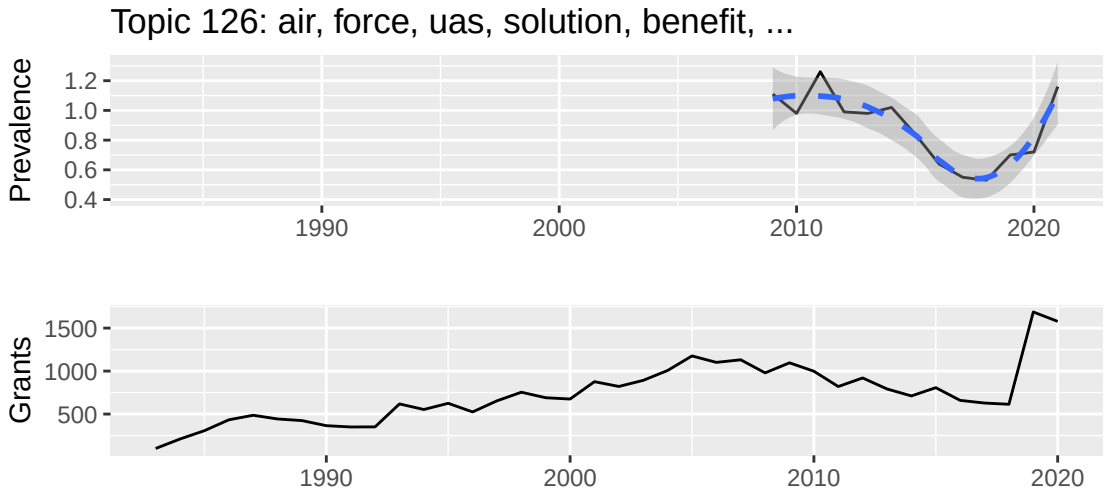


Figure 5.8: SBIR/STTR grants issued by the US Air Force, 1983 to 2020. There is a significant increase in 2019, corresponding to the emergence of topic 171, whose most probable words relate to the Air Force.

5.5 Discussion

tLDA supports several transfer learning use cases. Using simulated and real-world data, I show some properties of the tuning parameter a on models using tLDA. I leave exploration of using tLDA to encode expert input for future work. In principle this would require encoding expert judgement into a matrix prior $\boldsymbol{\eta}$.

One other area I leave for future study is using tLDA to build large pre-trained language models, as is common with deep learning transformers like BERT [15] and the GPT [18] [19] family. This is similarly possible in principle. However, the Gibbs algorithm used here paper makes building such large models computationally infeasible. Algorithms exist for scaling LDA estimates to "web-scale" data, such as [60], which can be used to implement tLDA in the future.

Empirical studies here show that tLDA should converge if $a \geq 1$. The simulations showed this when fine tuning on more data drawn from the same distribution. The real-world data analysis still showed signs of convergence for large a even though the generating distributions are presumed to be different over time.

Values of a around 0.8 emerge as a critical zone in all analyses. However, $a = 1$ has the most intuitive interpretation—that each word in the previous data set(s) has the same weight as each word in the current

data set. Yet it appears there is more fruitful work to be done in determining when it's best to have slight recency bias versus equal footing between base model and training set.

tLDA is implemented in *tidylda* for the R language, described in more detail in Chapter 6, next.

Chapter 6: The *tidylda* R Package

Off the shelf implementations of LDA are plentiful, yet existing LDA implementations do not integrate well with the research in this dissertation. Moreover, putting the methods proposed in this dissertation in a package and hosting it on a well used package repository like CRAN [129] makes those methods accessible to a wider audience. And creating a topic modeling implementation supporting the philosophies of the *tidyverse* [33] movement in the R ecosystem, which focus on making data analysis and computing tools work for humans, increases the accessiblity of these topic modeling methods. Having topic modeling software compatible with the *tidyverse* ecosystem is in line with Boyd-Graber et al.'s cry for "automatic text analysis for the people" [28, Ch. 10.3].

The *tidylda* package for topic modeling is compatible with the *tidyverse* and includes methods contained in the previous chapters of this dissertation. The Gibbs sampler implemented in *tidylda* is written in C++ for performance and offers several novel features. It also has methods for sampling from the posterior of a trained model, for more traditional Bayesian analyses.

6.1 Related Work

6.1.1 The "tidyverse" and "tidy" text mining

tidylda takes its syntactic cues from an ecosystem of R packages known as *the tidyverse*. The tidyverse's goal is to "facilitate a conversation between a human and computer about data" [33]. Packages in—and adjacent to—the tidyverse share a common design philosophy and syntax based on "tidy data" principles [130]. Tidy data has each variable in a column, each observation in a row, and each observational unit in a table. Extensions include the *broom* package [131] for "tidying" up outputs from statistical models and the in-development *tidymodels* ecosystem [132] which extends the tidyverse philosophy to statistical modeling and machine learning workflows.

Recently, Silge et al. [133] articulated a "tidy data" framework for text analyses—the *tidytext* package. Their approach has "one row per document per token". The *tidytext* package provides functionality to tok-

enize a corpus, transform it into this "tidy" format, and manipulate it in various ways, including preparing data for input into some of R's many topic modeling packages. The *tidytext* package also provides tidying functions in the style of *broom* to harmonize outputs from some of R's topic modeling packages. The *tidylda* package manages inputs and outputs in the flavor of *tidytext* but in one self contained package.

6.1.2 Topic modeling software in R

R has many packages for topic modeling. As far as I am aware, none are natively "tidy" though some have wrapper functions available in *tidytext* for interoperability. In almost all cases—at least for R packages—these models support only symmetric $\boldsymbol{\eta}$ priors, though some support asymmetric $\boldsymbol{\alpha}$.

The *textmineR* package [134] supports fitting models for LDA, correlated topic models [67], and LSA. It also contains a suite of analysis functions that are (mostly) interoperable with other topic modeling packages in the R ecosystem. In many ways, *textmeineR* is the predecessor of *tidylda*, supporting asymmetric priors for both $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. But *textmineR* does not support transfer learning nor is it fully consistent with the *tidyverse* principles.

The *topicmodels* package [135] supports fitting models for LDA and correlated topic models [67] with both a collapsed Gibbs sampler and VEM. When using VEM, $\boldsymbol{\alpha}$ may be treated as a free parameter and estimated during fitting. It is designed to be interoperable with the *tm* package [136], the oldest framework for text analysis in R.[1] The *tidytext* package provides "tidier" functions to make the *topicmodels* package interoperable with other frameworks, such as *quanteda* [137], *text2vec* [138], and more.

The *lda* package [139] provides a collapsed Gibbs sampler for LDA, supervised LDA [140], and other less well-known models. It allows users to specify only symmetric $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$. Its syntax is esoteric and it requires text documents as input, but does not offer much flexibility in the way of pre-processing. It is generally not interoperable with other packages without significant programming on the part of its users. However, its sequential Gibbs sampler is one of the fastest.

The *text2vec* package [138] is a framework for very fast text pre-processing and modeling. The *textmineR* package wraps *text2vec* for its pre-processing functions and *text2vec* implements LDA using the WarpLDA algorithm [60], but it only allows symmetric priors. The *text2vec* package also offers other models related to

[1] From private conversations, there is an immense frustration with working with *tm* within the R user community.

distributional semantics. Its syntax is also esoteric using R's *R6* objects that reach back to actively running C++ code for performance reasons. One of *text2vec*'s novel features is that it implements many different coherence calculations; most packages implement one or none.

The *STM* package [68] implements VEM algorithms for structural topic models [141] and correlated topic models [67]. *STM* is well-supported with interfaces in *tidytext*. It offers unique capabilities for model initialization somewhat analogous to transfer learning. Models may be initialized at random or from an LDA model that has run for a few iterations. *STM* does not offer this as a fully-fledged "transfer learning" paradigm. Instead it is a flag the user sets at run time. *STM* then produces the LDA model to hand off to the STM model internally. STM has several unique methods for setting priors but inspecting the documentation makes me believe that they are still symmetric, where applicable.

6.1.3 Topic modeling software in other languages

There are many (many) programs to implement topic models in many languages. Anecdotally, the two most common are also two of the oldest: *MALLET* and *Gensim*.

MALLET [142] stands for "MAchine Learning for LanguagE Toolkit." It is a Java program that implements LDA and many other models for working with text. Its LDA capabilities have a wrapper available in R—the *mallet* package—which launches a Java Virtual Machine in the background while users interact with it from R. MALLET allows users to set symmetric priors, but has an option to estimate an asymmetric $\boldsymbol{\alpha}$ based on [143]. Input must be text files, as it does all of its own pre-processing. MALLET's LDA implementation offers both a collapsed Gibbs sampler and distributed approximate Gibbs in the flavor of Newman et al. [55]. The *tidytext* package has *broom*-like functions to format the outputs of the *mallet* package.

Gensim [144] is a Python package that implements many models for working with text data, including LDA, LSI, and HDP [64]. Its LDA implementation is based on VEM and can be distributed across many compute nodes. Users can set flags for both $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ to be estimated and $\boldsymbol{\eta}$ can be a scalar, vector, or matrix. To my knowledge, Gensim is the only other program that offers this option—other than *tidylda*. Gensim uses other Python packages for pre-processing.

6.2 The *tidylda* Package's Novel Features

6.2.1 Model Initialization and Gibbs Sampling

The *tidylda* pakcage's Gibbs sampler has several unique features, described below.

Non-uniform initialization: Most LDA Gibbs samplers initialize by assigning words to topics and topics to documents (i.e., construct the $\boldsymbol{Cd}$ and $\boldsymbol{Cv}$ matrices) by sampling from a uniform distribution. This ensures initialization without incorporating any prior information. The *tidylda* package incorporates the priors in its initialization. It begins by drawing $P(\text{topic}|\text{document})$ and $P(\text{word}|\text{topic})$ from Dirichlet distributions with parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$, respectively. Then *tidylda* uses the above probabilities to construct $P(\text{topic}|\text{word}, \text{document})$ and makes a single run of the Gibbs sampler to initialize $\boldsymbol{Cd}$ and $\boldsymbol{Cv}$. This non-uniform initialization powers tLDA, described in Chapter 5, by starting a Gibbs run near where the previous run left off. For initial models, it uses the user's prior information to tune where sampling starts.

Flexible priors: *tidylda* has multiple options for setting LDA priors. Users may set scalar values for $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ to construct symmetric priors. Users may also choose to construct vector priors for both $\boldsymbol{\alpha}$ and $\boldsymbol{\eta}$ for a full specification of LDA. Additionally, *tidylda* allows users to set a matrix prior for $\boldsymbol{\eta}$, enabled by its implementation of tLDA. This enables users to set priors over word-topic relationships informed by expert input. The best practices for encoding expert input in this manner are not yet well studied. Nevertheless, this capability makes *tidylda* unique among LDA implementations.

Burn in iterations and posterior averaging: Most LDA Gibbs samplers construct posterior estimates of $\boldsymbol{\Theta}$ and $\boldsymbol{B}$ from $\boldsymbol{Cd}$ and $\boldsymbol{Cv}$'s values of the final iteration of sampling, effectively using a single sample. This is inconsistent with best practices from Bayesian statistics, which is to average over many samples from a stable posterior. *tidylda* enables averaging across multiple samples of the posterior with the `burnin` argument. When `burnin` is set to a positive integer, *tidylda* averages the posterior across all iterations larger than `burnin`. For example, if `iterations` is 200 and `burnin` is 150, *tidylda* will return a posterior estimate that is an average of the last 50 sampling iterations. This ensures that posterior estimates are more likely to be representative than any single sample.

Transfer learning with tLDA: Finally, and as discussed previously, *tidylda*'s Gibbs sampler enables transfer learning with tLDA. The full specification of tLDA and details on its implementation in *tidylda* are

in Chapter 5.

6.2.2 Tidy Methods

The *tidylda* package's construction follows *Conventions of R Modeling Packages* [145]. In particular, it contains methods for `print`, `summary`, `glance`, `tidy`, and `augment`, consistent with other "tidy" packages. These methods are briefly described below and are demonstrated in Appendix F.

- `print`, `summary`, and `glance` return various summaries of the contents of a `tidylda` object, into which an LDA model trained with *tidylda* is stored.
- `tidy` returns the contents of $\boldsymbol{\Theta}$, $\boldsymbol{B}$, or $\boldsymbol{\Lambda}$ (stored as `theta`, `beta`, and `lambda` respectively), as specified by the user, formatted as a tidy `tibble`, instead of a numeric matrix.
- `augment` appends model outputs to observational-level data. Taking the cue from *tidytext* [127], "observational-level" data is one row per word per document. Therefore, the key statistic used by `augment` is $P(\text{topic}|\text{word}, \text{document})$. *tidylda* calculates this as $\boldsymbol{\Lambda} \times P(\text{word}|\text{document})$, where $P(\text{word}|\text{document}_d) = \frac{\boldsymbol{x}_d}{\sum_{v=1}^{V} x_{d,v}}$.

6.2.3 Other Noteable Features

The *tidylda* package has three evaluation metrics for topic models, two goodness-of-fit measures—R^2 as described in Chapter 4 and the log likelihood of the model given the data—and one coherence measure—probabilistic coherence. R-squared is calculated via the *mvrsquared* package [146] and is a flag set during model fitting with `calc_r2 = TRUE`[2]. Similarly, the log likelihood of the model, given the data, is calculated at each Gibbs iteration if the user selects `calc_likelihood = TRUE` during model fitting.

The coherence measure is called probabilistic coherence. (See Appendix E.) Probabilistic coherence is bound between -1 and 1. Values near zero indicate that the top words in a topic are statistically independent from each other. Positive values indicate that the top words in a topic are positively correlated in a statistically-dependent manner. Negative values indicate that the words in a topic are negatively correlated with each other in a statistically-dependent manner. (In practice, negative values tend to be very near zero.)

The *tidylda* package enables traditional Bayesian uncertainty quantification by sampling from the pos-

[2]Users can calculate R^2 after a model is fit by using the *mvrsquared* package or calling `tidylda:::calc_lda_rsquared`. `calc_lda_rsquared` is an internal function to *tidylda*, requiring the package name followed by three colons, as is R's standard.

terior. The posterior distribution for $\boldsymbol{\theta}_d$ is Dirichlet($\boldsymbol{C}\boldsymbol{d}_d + \boldsymbol{\alpha}$) and the posterior distribution for β_k is Dirichlet($\boldsymbol{C}\boldsymbol{v}_k + \boldsymbol{\eta}$) (or Dirichlet($\boldsymbol{C}\boldsymbol{v}_k + \boldsymbol{\eta}_k$) for tLDA). *tidylda* enables a `posterior` method for `tidylda` objects, allowing users to sample from the posterior to quantify uncertainty for estimates of estimated parameters.

The *tidylda* package uses one of two calculations for predicting topic distributions (i.e., $\hat{\boldsymbol{\theta}}_d$) for new documents. The first, and default, is to run the Gibbs sampler, constructing a new $\boldsymbol{C}\boldsymbol{d}$ for the new documents but without updating topic-word distributions in $\boldsymbol{B}$. The second uses a dot product as described in Appendix A. The *tidylda* package actually uses the dot product prediction combined with the *non-uniform initialization*—described above—to initialize $\boldsymbol{C}\boldsymbol{d}$ when predicting using the Gibbs sampler.

6.3 Discussion

While many other topic modeling packages exist, *tidylda* is very user friendly and brings novel features. Its user friendliness comes from compatibility with the tidyverse. And *tidylda* includes tLDA and other methods contained in the previous chapters of this dissertation. It also has methods for sampling from the posterior of a trained model, for more traditional Bayesian analyses. The *tidylda* package's Gibbs sampler is written in C++ for performance.

Chapter 7: Conclusion

I dream of a world where we can measure the ideas and narratives that drive human behavior with the same rigor we use to measure the economy, human health, and other societal markers based on structured numeric data. To achieve this dream, we need better statistical theory for working with language as data—which I refer to as Natural Language Statistics—and intuitive tooling for language analysis. In this dissertation, I have taken small but substantive steps to address both needs. I have argued that LDA is an appealing model for Natural Language Statistics. It models a data generating process which may be linked to the empirical laws of language. This property makes LDA, and related models, candidates for helping to develop a more robust statistical theory for modeling language. Moreover, LDA embeds collections of word occurrences into a probability space, ideal for quantifying uncertainty and leaning on probability theory to analyze results.

In this dissertation I have linked the LDA-DGP to Zipf's and Heaps's laws, enabling simulation studies of LDA to incorporate fundamental properties of human language, and making them more principled. I have employed such simulations to produce a pilot study of the effects of misspecifying hyper parameters when fitting LDA models. I have derived a generalization of the coefficient of determination, allowing it to be extended to topic models. The coefficient of determination—R^2—is intuitive and in wide use for linear statistical models, thus making interpreting LDA more accessible. I have extended LDA to tLDA, a model enabling the use of LDA to develop foundation models and fine tune them in a way that preserves the Markov property. And I have written and released *tidylda* a package for the R programming language. The *tidylda* package implements much of the research in this dissertation and follows conventions of R's "tidyverse" system of packages, known for their user friendliness, thus making topic modeling more accessible to a wider audience of practitioners. And yet, while the work in this dissertation has taken meaningful steps towards making use of LDA more rigorous, much more remains to be done.

Simulations in this dissertation could be more realistic, especially in incorporating stop words, accounting for correlations in a corpus, or changing topics over time. I used the standard definition of Zipf's law for this dissertation, but the more general Zipf-Mandlebrodt law may better lead to better simulations of corpora with

stop words removed. Adopting a probabilistic definition of the Zipf-Mandlebrodt law may offer ultimate flexibility and tie the LDA-DGP further into an end-to-end statistical theory of language data.

The simulations in this dissertation were also of static corpora with no intra-context correlations. Might we extend simulations to account for such correlations as the CTM and STM models do? Could we extend simulations to account for linguistic changes over time? I would hypothesize that simulations incorporating these would either better account for patterns seen in real-world data or they would further justify using the simpler approach as done here.

Researchers still need better guidance for selecting hyper parameters when fitting LDA models. This research shows that $\boldsymbol{\eta}$ is more important than conventionally understood and that K is as important as conventional wisdom suggests. Even so, it was not sufficient to find a protocol or concrete diagnostics about model specification. As future work, I would explore various protocols to find support for their use or evidence that they do not help (or that they hurt). For example, it is common in the literature to build models for many different values of K and choose one that maximizes a statistic of interest. This is computationally infeasible for a large corpus. Yet anecdotally, fitting a model with too many topics and then discarding "junk" topics (those with low prevalence or coherence etc.) post-hoc seems effective. Might there be validity to this approach? Could partial R^2—as shown in Chapter 4—inform selecting junk topics? Another example: MALLET uses a method for optimizing $\boldsymbol{\alpha}$ during fitting. Does this method often find the correct $\boldsymbol{\alpha}$ on synthetic data? Might setting a non-informative $\boldsymbol{\alpha}$ be beneficial? We are yet a long way from having guidance for LDA that is of the quality for fitting linear models, but this dissertation constitutes a meaningful start.

tLDA is a flexible and analyzable method for pre-train/fine tune transfer learning. Yet to scale to the corpora used to train deep learning transformers, the Gibbs algorithm is insufficient. Extending tLDA to algorithms like WarpLDA [60] and perhaps implementing it on a GPU would help compare the tradeoffs of using probabilistic topic models versus black box transformers. But until we can fit LDA models of comparable scale, any benefits are hypothetical.

In this dissertation, I demonstrated tLDA on a time series use case. But other use cases are possible and should be explored. Most obviously, one should build foundation models and compare results to transformers where applicable, perhaps using LDA to vectorize text before performing a downstream task. This requires

an algorithm that scales, as discussed above. Also, tLDA can enable a "differences" analysis comparing groups, analogous to estimating treatment effects. Such analyses might be applied to researching the efficacy of marketing or the effects of exposure to propaganda.

The *tidylda* package implements the methods demonstrated in this dissertation. It is available for anyone who wishes to use it. Yet *tidylda* could still benefit from user feedback. An obvious place to start would be to implment a more scaleable algorithm, as above. Another obvious improvement would be to develop a method for expert-seeded topics using tLDA. Finally, as we get better guidance on how to set LDA's hyper-parameters, *tidylda* would modify its defaults so that a user has a good shot of a reasonable model with minimum effort.

The real-world data used in this research was all in English and came from abstracts of research funded by the US federal government. That compels the follwing questions: Would the findings in this dissertation apply to non-English languages? Would the findings in this dissertation apply to a wider variety of length and formality of languages? (Tweets vs. newspaper articles vs. conversational language, etc.) I believe the answer to both questions is "yes" but with an important caveat.

Zipf's and Heaps's laws are universal. They apply to any language in any context, even if their numeric parametarizations change. These empirical laws hold for English and Mandarin, speeches to the European Parliament and the tweets of Elon Musk. As a result, the results using simulated data presented in this dissertation should hold. Yet, while the simulations used here accounted for a large variety in corpus properties, it is reasonable to question whether or not the ranges of parameters represented here are representative of the ranges of real-world corpora researchers would encounter. I leave addressing this important caveat to future research.

A work is never finished, it's just taken away. In spite of the need for more research, the results contained in this dissertation take a tiny step towards making LDA better for Natural Language Statistics and improving our understanding of how to measure and describe ideas and narratives with statistical rigor.

Appendix A: Projection Matrix for Probabilistic Embedding Models

We can use Bayes's rule to derive a projection matrix, $\mathbf{\Lambda}$, for any model that embeds text into a probability space such as pLSI, LDA, and similar.

Given a model developed on a matrix $\boldsymbol{X}$ with D contexts, indexed by $d \in \{1,2,...,D\}$; V unique words in the vocabulary, indexed by $v \in \{1,2,...,V\}$; N total word instances in the vocabulary and N_d word instances in the d-th document such that $N = \sum_{d=1}^{D} N_d$ and $N_d = \sum_{v=1}^{V} x_{d,v}$. The model itelf has; K topics, indexed by $k \in \{1,2,...,K\}$.

In the resulting model, $P(z_k|d) = \theta_{d,k}$ is the probability that a token in document d was sampled from topic k and $P(w_v|z_k) = \beta_{k,v}$ is the probability of sampling token v from topic k.

We need to derive $P(z_k|w_v) = \lambda_{v,k}$ using Bayes's rule.

$$\lambda_{v,k} = P(z_k|w_v) \tag{A.1}$$

$$= \frac{P(w_v|z_k) \cdot P(z_k)}{P(w_v)} \tag{A.2}$$

$$= \frac{\beta_{k,v} \cdot P(z_k)}{P(w_v)} \tag{A.3}$$

We can use the law of total probability to find $P(z_k)$.

$$P(z_k) = \sum_{d=1}^{D} P(z_k|d) \cdot P(d) \tag{A.4}$$

$$= \sum_{d=1}^{D} \theta_{d,k} \cdot P(d) \tag{A.5}$$

We can take $P(w_v)$ and $P(d)$ from $\boldsymbol{X}$ with

$$P(w_v) = \frac{1}{N} \sum_{d=1}^{D} x_{d,v} \tag{A.6}$$

$$P(d) = \frac{1}{N} \sum_{v=1}^{V} x_{d,v} \tag{A.7}$$

This gives us our final form

$$\lambda_{v,k} = \frac{\beta_{k,v} \cdot (\sum_{d=1}^{D} \theta_{d,k}) \cdot (\frac{1}{N} \sum_{v=1}^{V} x_{d,v})}{\frac{1}{N} \sum_{d=1}^{D} x_{d,v}} \tag{A.8}$$

Each of the above values represents the v,k entry of $\mathbf{\Lambda}$. We can then project a new data set, $\boldsymbol{X}'$ into the embedding space in two steps:

1. Normalize the rows of $\boldsymbol{X}'$ so that each row sums to 1; call this $\boldsymbol{X}'_n$
2. Right multiply $\boldsymbol{X}'_n$ by $\mathbf{\Lambda}^T$ such that $\mathbf{\Theta}' = \boldsymbol{X}'_n \cdot \mathbf{\Lambda}^T$

The above is valid for any probabilistic topic model, though it is a purely frequentist approach. My experience has been that this leads to noisier projections than, for example, "folding in" new data with a Gibbs sampler when using LDA.

Appendix B: Collapsed Gibbs Sampling for LDA

Algorithm 1: Allocate $\boldsymbol{Cd}$ and $\boldsymbol{Cv}$ by sampling

//Initialize $\boldsymbol{Cd}$, $\boldsymbol{Cv}$ as zero-valued matrices;

for *each document, d* **do**

  for *each word in the d-th context, n* **do**

    sample z such that $P(z=k) \sim \text{Uniform}(1,K)$;

    $Cd_{d,z}$ += 1;

    $Cv_{z,n}$ += 1;

  end

end

//Begin Gibbs sampling ;

for *each iteration, i* **do**

  for *each document, d* **do**

    for *each word in the d-th context, n* **do**

      $Cd_{d,z^{(i-1)}}$ -= 1;

      $Cv_{z^{(i-1)},n}$ -= 1;

      sample z such that $P(z^{(i)}=k) = \frac{Cv_{k,n}+\eta_n}{\sum_{v=1}^{V} Cv_{k,v}+\eta_v} \cdot \frac{Cd_{d,k}+\alpha_k}{\left(\sum_{k=1}^{K} Cd_{d,k}+\alpha_k\right)-1}$;

      $Cd_{d,z^{(i)}}$ += 1;

      $Cv_{z^{(i)},n}$ += 1;

    end

  end

end

Once sampling is complete, one can derive posterior estimates with

$$\hat{\theta}_{d,k} = \frac{Cd_{d,k} + \alpha_k}{\sum_{k=1}^{K} Cd_{d,k} + \alpha_k} \tag{B.1}$$

$$\hat{\beta}_{k,v} = \frac{Cv_{k,v} + \eta_v}{\sum_{v=1}^{V} Cv_{k,v} + \eta_v} \tag{B.2}$$

Appendix C: Expected Term Frequency of the LDA-DGP

Below derives the expected term frequency of a corpus whose terms are generated by the stochastic process modeled by Latent Dirichlet Allocation, the LDA-DGP. The expected term frequencies of a corpus generated with the above process are proportional to $\boldsymbol{\eta}$—the parameter for the Dirichlet prior for terms over topics. This implies that for a simulated corpus to follow Zipf's law, then $\boldsymbol{\eta}$ must be proportional to a power law.

Assuming there are D contexts, K topics, V unique words, N total words and N_d words in the d-th context, the LDA-DGP is as follows. For each word, n, in context d:

1. Generate $\boldsymbol{B}$ by sampling K topics $\beta_k \sim \text{Dirichlet}(\boldsymbol{\eta}), \forall k \in \{1,2,...,K\}$
2. Generate $\boldsymbol{\Theta}$ by sampling D documents $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\boldsymbol{\alpha}), \forall d \in \{1,2,...,D\}$
3. Then for each context, d
    1. Draw topic $z_{d,n}$ from Multinomial$(\boldsymbol{\theta}_d)$
    2. Draw word $w_{d,n}$ from Multinomial$(\boldsymbol{\beta}_{z_{d,n}})$
    3. Repeat 1. and 2. N_d times.

Note that context d has N_d words $\forall d \in \{1,2,...,D\}$ and that the total number of words in the corpus is $N = \sum_{d=1}^{D} N_d$.

Under the model, the expected term frequency of a single context is

$$\mathbb{E}(\mathbf{w}_d|\boldsymbol{\theta}_d, \boldsymbol{B}) = n_d \odot \boldsymbol{\theta}_d \cdot \boldsymbol{B} \tag{C.1}$$

Using the law of total expectation, we have

$$
\begin{aligned}
\mathbb{E}(\mathbf{w}_d) &= \mathbb{E}(\mathbb{E}(\mathbf{w}_d|\boldsymbol{\theta}_d, \boldsymbol{B})) \\
&= \mathbb{E}(n_d \odot \boldsymbol{\theta}_d \cdot \boldsymbol{B}) \\
&= \mathbb{E}\left(n_d \odot \boldsymbol{\theta}_d \cdot \boldsymbol{\beta}\right) \\
&= \mathbb{E}\left(\begin{array}{c} n_d \begin{pmatrix} \sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,1} \\ \sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,2} \\ \cdots \\ \sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,V} \end{pmatrix} \end{array}\right) \\
&= \mathbb{E}(n_d) \begin{pmatrix} \mathbb{E}(\sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,1}) \\ \mathbb{E}(\sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,2}) \\ \cdots \\ \mathbb{E}(\sum_{k=1}^{K} \theta_{d,k} \cdot \beta_{k,V}) \end{pmatrix} \\
&= \mathbb{E}(n_d) \begin{pmatrix} \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k})\mathbb{E}(\beta_{k,1}) \\ \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k})\mathbb{E}(\beta_{k,2}) \\ \cdots \\ \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k})\mathbb{E}(\beta_{k,V}) \end{pmatrix}
\end{aligned}
$$

The last step, above, is due to indpendence of $\boldsymbol{\theta}_d$ and β_k $\forall d, k$.

Before carrying on, note two more relationships:

1. $\beta_k \sim$ i.i.d. Dirichlet($\boldsymbol{\eta}$) means that $\mathbb{E}(\beta_i) = \mathbb{E}(\beta_j)$ $\forall i, j \in \{1, 2, .., K\}$
2. The expected value of a Dirchlet random variable—$\mathbf{X}$—with parameter $\boldsymbol{\delta}$ is $\mathbb{E}(\mathbf{X}) = \frac{1}{\sum_{m=1}^{M} \delta_m} \cdot \boldsymbol{\delta}$

From number 1., above, we can pull $\mathbb{E}(\beta_{k,.})$ outside of the summation. And we can carry through the expected

values using number 2., above.

$$
\begin{aligned}
\mathbb{E}(\mathbf{w}_d) &= \mathbb{E}(n_d) \begin{pmatrix} \mathbb{E}(\beta_{k,1}) \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k}) \\ \mathbb{E}(\beta_{k,2}) \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k}) \\ \cdots \\ \mathbb{E}(\beta_{k,V}) \sum_{k=1}^{K} \mathbb{E}(\theta_{d,k}) \end{pmatrix} \\
&= n_d \begin{pmatrix} \frac{\eta_1}{\sum_{v=1}^{V} \eta_v} \sum_{k=1}^{K} \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} \\ \frac{\eta_2}{\sum_{v=1}^{V} \eta_v} \sum_{k=1}^{K} \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} \\ \cdots \\ \frac{\eta_V}{\sum_{v=1}^{V} \eta_v} \sum_{k=1}^{K} \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} \end{pmatrix} \\
&= n_d \begin{pmatrix} \frac{\eta_1}{\sum_{v=1}^{V} \eta_v} \frac{\sum_{k=1}^{K} \alpha_k}{\sum_{k=1}^{K} \alpha_k} \\ \frac{\eta_2}{\sum_{v=1}^{V} \eta_v} \frac{\sum_{k=1}^{K} \alpha_k}{\sum_{k=1}^{K} \alpha_k} \\ \cdots \\ \frac{\eta_V}{\sum_{v=1}^{V} \eta_v} \frac{\sum_{k=1}^{K} \alpha_k}{\sum_{k=1}^{K} \alpha_k} \end{pmatrix} \\
&= n_d \begin{pmatrix} \frac{\eta_1}{\sum_{v=1}^{V} \eta_v} \cdot 1 \\ \frac{\eta_2}{\sum_{v=1}^{V} \eta_v} \cdot 1 \\ \cdots \\ \frac{\eta_V}{\sum_{v=1}^{V} \eta_v} \cdot 1 \end{pmatrix} \\
&= \frac{n_d}{\sum_{v=1}^{V} \eta_v} \boldsymbol{\eta} \\
&\propto \boldsymbol{\eta}
\end{aligned}
$$

The end result is that the expected term frequency of a single document is proportional to $\boldsymbol{\eta}$—the Dirichlet parameter for terms over topics.

The term frequency for the whole corpus is the sum of the term frequencies for each document. Specifi-

cally

$$\mathbf{w} = \sum_{d=1}^{D} \mathbf{w}_d$$

The expected value under the model, then, can be carried through.

$$\begin{aligned}
\mathbb{E}(\mathbf{w}) &= \mathbb{E}\left(\sum_{d=1}^{D} \mathbf{w}_d\right) \\
&= \sum_{d=1}^{D} \mathbb{E}(\mathbf{w}_d) \\
&= \sum_{d=1}^{D} \frac{n_d}{\sum_{v=1}^{V} \eta_v} \boldsymbol{\eta} \\
&= \frac{\sum_{d=1}^{D} n_d}{\sum_{v=1}^{V} \eta_v} \boldsymbol{\eta} \\
&\propto \boldsymbol{\eta}
\end{aligned}$$

Appendix D: Partitioning the Posterior

The posterior distribution of topic k is

$$\boldsymbol{\beta}_k \sim \text{Dirichlet}(\boldsymbol{C}\boldsymbol{v}_k + \boldsymbol{\eta}_k) \tag{D.1}$$

For two arbitrary sets of documents, we can break up the posterior parameter.

$$\boldsymbol{\beta}_k \sim \text{Dirichlet}(\boldsymbol{C}\boldsymbol{v}_k^{(1)} + \boldsymbol{C}\boldsymbol{v}_k^{(2)} + \boldsymbol{\eta}_k) \tag{D.2}$$

This has two implications:

1. We can quantify how much a set of documents contributes to the posterior topics. This may allow us to quantify biases in our topic models. (This is left for future research.)
2. We can interpret the weight parameter, $\boldsymbol{\omega}_k$ for transfer learning as the weight that documents from the base model affect the posterior of the fine-tuned model.

Changing notation, for (2) we have

$$\boldsymbol{\beta}_k \sim \text{Dirichlet}(\boldsymbol{C}\boldsymbol{v}_k^{(t)} + \boldsymbol{\eta}_k^{(t)}) \tag{D.3}$$

$$\sim \text{Dirichlet}(\boldsymbol{C}\boldsymbol{v}_k^{(t)} + \boldsymbol{C}\boldsymbol{v}_k^{(t-1)} + \boldsymbol{\eta}_k^{(t-1)}) \tag{D.4}$$

Substituting in the definition from tLDA, we have

$$\boldsymbol{\eta}_k^{(t)} = \boldsymbol{C}\boldsymbol{v}_k^{(t-1)} + \boldsymbol{\eta}_k^{(t-1)} \tag{D.5}$$

$$= \boldsymbol{\omega}_k^{(t)} \cdot \mathbb{E}\left[\boldsymbol{\beta}_k^{(t-1)}\right] \tag{D.6}$$

Solving for $\omega_k^{(t)}$ in $\boldsymbol{C}\boldsymbol{v}_k^{(t-1)} + \boldsymbol{\eta}_k^{(t-1)} = \omega_k^{(t)} \cdot \mathbb{E}\left[\boldsymbol{\beta}_k^{(t-1)}\right]$ gives us

$$\omega_k^{*(t)} = \sum_{v=1}^{V} \left(C v_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)} \right) \tag{D.7}$$

Where $\omega_k^{*(t)}$ is a critical value such that fine tuning is just like adding data to the base model. In other words each token from the base model, $\boldsymbol{X}^{(t-1)}$, has the same weight as each token from $\boldsymbol{X}^{(t)}$. This gives us an intuitive means to tune the weight of the base model when fine tuning and collapses K tuning parameters into one. Specifically:

$$\omega_k^{(t)} = a^{(t)} \cdot \omega_k^{*(t)} \tag{D.8}$$

$$= a^{(t)} \cdot \sum_{v=1}^{V} \left(C v_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)} \right) \tag{D.9}$$

and

$$\boldsymbol{\eta}_k^{(t)} = a^{(t)} \cdot \sum_{v=1}^{V} \left(C v_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)} \right) \cdot \mathbb{E}\left[\boldsymbol{\beta}_k^{(t-1)}\right] \tag{D.10}$$

Appendix E: Probabilistic Coherence

Coherence measures seek to emulate human judgement quantifiably. They tend to rely on the degree to which words co-occur together. Yet, words may co-occur in ways that are unhelpful. For example, consider the words "the" and "this". They co-occur very frequently in English-language documents. Yet these words are statistically independent. Knowing the relative frequency of the word "the" in a document does not give you any additional information on the relative frequency of the word "this" in a document.

Probabilistic coherence uses the concepts of co-occurence and statistical independence to rank topics. Note that probabilistic coherence has not yet been rigorously evaluated to assess its correlation to human judgement. Anecdotally, those that have used it tend to find it helpful. Probabilistic coherence is included in both the *textmineR* and *tidylda* packages in the R language.

Probabilistic coherence averages differences in conditional probabilities across the top M most-probable words in a topic. Let $x_{i,k}$ correspond the i-th most probable word in topic k, such that $x_{1,k}$ is the most probable word, $x_{2,k}$ is the second most probable and so on. Further, let

$$x_{i,k} = \begin{cases} 1 & \text{if the } i\text{-th most probable word appears in a randomly selected document} \\ 0 & \text{otherwise} \end{cases} \tag{E.1}$$

Then probablistic coherence for the top M terms in topic k is calculated as follows:

$$C(k,M) = \frac{1}{\sum_{j=1}^{M-1} M - j} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} P(x_{j,k} = 1 | x_{i,k} = 1) - P(x_{j,k} = 1) \tag{E.2}$$

Where $P(x_{j,k} = 1 | x_{i,k} = 1)$ is the fraction of contexts containing word i that contain word j and $P(x_{j,k} = 1)$ is the fraction of all contexts containing word j.

This brings us to interpretation:

1. If $P(x_{j,k} = 1 | x_{i,k} = 1) - P(x_{j,k} = 1)$ is zero, then $P(x_{j,k} = 1 | x_{i,k} = 1) = P(x_{j,k} = 1)$, the definition of

statistical independence.

2. If $P(x_{j,k} = 1 | x_{i,k} = 1) > P(x_{j,k} = 1)$, then word j is more present than average in contexts also containing word i.

3. If $P(x_{j,k} = 1 | x_{i,k} = 1) < P(x_{j,k} = 1)$, then word j is less present than average in conexts that contain word i. LDA is unlikely to find strong negative co-occurrences. In practice, negative values tend to be near zero.

Appendix F: The *tidylda* Package Code Examples

This section performs an example analysis on a toy data set to demonstrate *tidylda*'s functionality. The data is a randomly selected subset of 100 grant abstracts funded by NIH in 2014 [103].

First, we load *tidylda* and several helpful libraries. We then create a document term matrix and split the documents into two groups to demonstrate tLDA.

```
library(tidytext)

library(tidyverse)

library(tidylda)

library(Matrix)


### Initial set up ---

# load some documents

docs <- nih_sample


# tokenize using tidytext's unnest_tokens

tidy_docs <- docs %>%

  select(APPLICATION_ID, ABSTRACT_TEXT) %>%

  unnest_tokens(output = word,

                input = ABSTRACT_TEXT,

                stopwords = stop_words$word,

                token = "ngrams",

                n_min = 1, n = 2) %>%

  count(APPLICATION_ID, word) %>%

  filter(n>1) #Filtering for words/bigrams per document, rather than per corpus
```

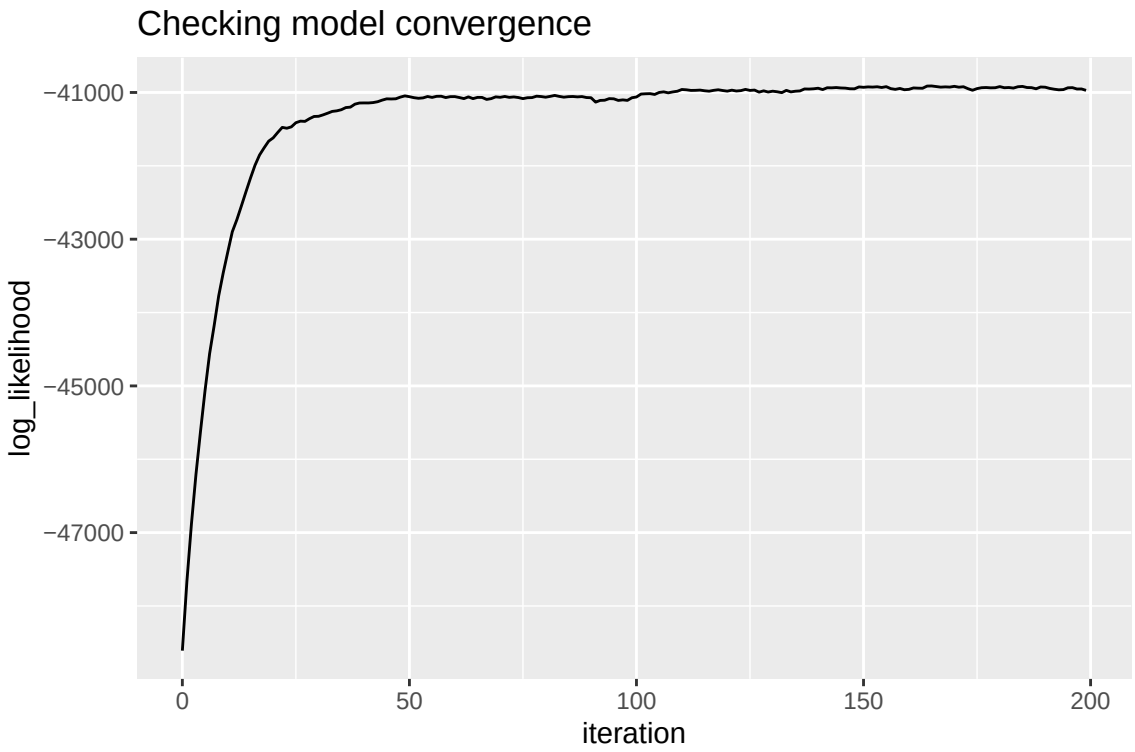
```
tidy_docs <- tidy_docs %>% # filter words that are just numbers
  filter(! stringr::str_detect(tidy_docs$word, "^[0-9]+$"))


# append observation level data
colnames(tidy_docs)[1:2] <- c("document", "term")



# turn a tidy tbl into a sparse dgCMatrix
# note tidylda has support for several document term matrix formats
d <- tidy_docs %>%
  cast_sparse(document, term, n)


# let's split the documents into two groups to demonstrate predictions and updates
d1 <- d[1:50, ]


d2 <- d[51:nrow(d), ]


# make sure we have different vocabulary for each data set to simulate the "real world"
# where you get new tokens coming in over time
d1 <- d1[, colSums(d1) > 0]


d2 <- d2[, colSums(d2) > 0]
```

Next, we fit an initial LDA model of 10 topics. Choices for `alpha`, `eta`, and number of Gibbs iterations are common choices. We set `calc_likelihood` and `calc_r2` to `TRUE` so that we get some goodness of fit metrics. Probabilistic coherence is calculated by default.

```
### fit an initial model and inspect it ----

set.seed(123)


lda <- tidylda(

  data = d1,

  k = 10,

  iterations = 200,

  burnin = 175,

  alpha = 0.1, # also accepts vector inputs

  eta = 0.05, # also accepts vector or matrix inputs

  calc_likelihood = TRUE,

  calc_r2 = TRUE

)
```

We can plot the log likelihood to visually inspect for convergence. It's also possible to perform tests of convergence over the post-burn in iterations. That is an exercise left to the reader.

```
# did the model converge?

# there are actual test stats for this, but should look like "yes"

lda$log_likelihood %>%

  ggplot(aes(x = iteration, y = log_likelihood)) +

  geom_line(color = "black") +

  ggtitle("Checking model convergence")
```



The methods `summary`, `glance`, and `print` produce various model summaries. `tidylda` objects contain a summary Tibble with the top 5 words in each topic, the prevalence of each topic in the training corpus, and probabilistic coherence of the top 5 words in each topic.

```
# look at the model overall

summary(lda)
```

```
               Length Class  Mode

beta           15240  -none- numeric

theta            500  -none- numeric

lambda         15240  -none- numeric

alpha              1  -none- numeric

eta                1  -none- numeric

summary            4  tbl_df list

call               9  -none- call

log_likelihood     2  tbl_df list
```

```
counts             2  -none- list
r2                 1  -none- numeric
```

```
glance(lda)
```

```
# A tibble: 1 x 5
  num_topics num_documents num_tokens iterations burnin
       <int>         <int>      <int>      <dbl>  <dbl>
1         10            50       1524        200    175
```

```
print(lda)
```

```
A Latent Dirichlet Allocation Model of  10 topics,  50  documents, and  1524  tokens:
tidylda(data = d1, k = 10, iterations = 200, burnin = 175, alpha = 0.1,
    eta = 0.05, calc_likelihood = TRUE, calc_r2 = TRUE)

The model's R-squared is  0.2677
The  5  most prevalent topics are:
# A tibble: 10 x 4
  topic prevalence coherence top_terms
  <dbl>      <dbl>     <dbl> <chr>
1     7       11.9     0.140 cancer, dcis, cells, breast, rb, ...
2     1       11.0     0.22  cdk5, effects, v4, stiffening, wall, ...
3     6       10.9     0.295 sud, risk, factors, drug, imaging, ...
4    10       10.8     0.303 mitochondrial, plasticity, studies, redox, functio~
5     3       10.5     0.248 sleep, cell, lung, cells, specific, ...
# ... with 5 more rows

The  5  most coherent topics are:
```

```
# A tibble: 10 x 4
  topic prevalence coherence top_terms
  <dbl>      <dbl>     <dbl> <chr>
1     9      10.2      0.509 diabetes, numeracy, data, intervention, management~
2     8       8.83     0.494 function, cmybp, injury, fragment, 29 kda, ...
3    10      10.8      0.303 mitochondrial, plasticity, studies, redox, functio~
4     6      10.9      0.295 sud, risk, factors, drug, imaging, ...
5     5       8.06     0.258 research, disparities, program, cancer, mhirt, ...
# ... with 5 more rows
```

```
# it comes with its own summary matrix that's printed out with print(), above
lda$summary
```

```
# A tibble: 10 x 4
   topic prevalence coherence top_terms
   <dbl>      <dbl>     <dbl> <chr>
 1     1      11.0      0.22  cdk5, effects, v4, stiffening, wall, ...
 2     2       8.07     0.18  research, responses, antibodies, natural, respons~
 3     3      10.5      0.248 sleep, cell, lung, cells, specific, ...
 4     4       9.76     0.145 cns, cells, brain, infection, based, ...
 5     5       8.06     0.258 research, disparities, program, cancer, mhirt, ...
 6     6      10.9      0.295 sud, risk, factors, drug, imaging, ...
 7     7      11.9      0.140 cancer, dcis, cells, breast, rb, ...
 8     8       8.83     0.494 function, cmybp, injury, fragment, 29 kda, ...
 9     9      10.2      0.509 diabetes, numeracy, data, intervention, managemen~
10    10      10.8      0.303 mitochondrial, plasticity, studies, redox, functi~
```

The `tidy` function reformats the `theta`, `beta`, and `lambda` matrices into tidy Tibbles. This can make plotting and summarization easier when using tools from the "tidyverse".

```
# inspect the individual matrices

tidy_theta <- tidy(lda, matrix = "theta")


tidy_theta
```

```
# A tibble: 500 x 3

   document topic   theta

   <chr>    <dbl>   <dbl>

 1 8574224      1 0.00238

 2 8574224      2 0.00238

 3 8574224      3 0.00238

 4 8574224      4 0.00238

 5 8574224      5 0.00238

 6 8574224      6 0.976

 7 8574224      7 0.00333

 8 8574224      8 0.00238

 9 8574224      9 0.00429

10 8574224     10 0.00238

# ... with 490 more rows
```

```
tidy_beta <- tidy(lda, matrix = "beta")


tidy_beta
```

```
# A tibble: 15,240 x 3

   topic token             beta

   <dbl> <chr>            <dbl>

 1     1 adolescence  0.0000673

 2     1 age          0.0000673
```

```
 3     1 application  0.0000673
 4     1 depressive   0.0000673
 5     1 disorder     0.0000673
 6     1 emotionality 0.0000673
 7     1 information  0.00276
 8     1 mdd          0.0000673
 9     1 onset        0.0000673
10     1 onset mdd    0.0000673
# ... with 15,230 more rows
```

```
tidy_lambda <- tidy(lda, matrix = "lambda")


tidy_lambda
```

```
# A tibble: 15,240 x 3
   topic token         lambda
   <dbl> <chr>          <dbl>
 1     1 adolescence  0.00780
 2     1 age          0.00912
 3     1 application  0.00780
 4     1 depressive   0.0201
 5     1 disorder     0.0201
 6     1 emotionality 0.0201
 7     1 information  0.273
 8     1 mdd          0.0111
 9     1 onset        0.00773
10     1 onset mdd    0.0201
# ... with 15,230 more rows
```

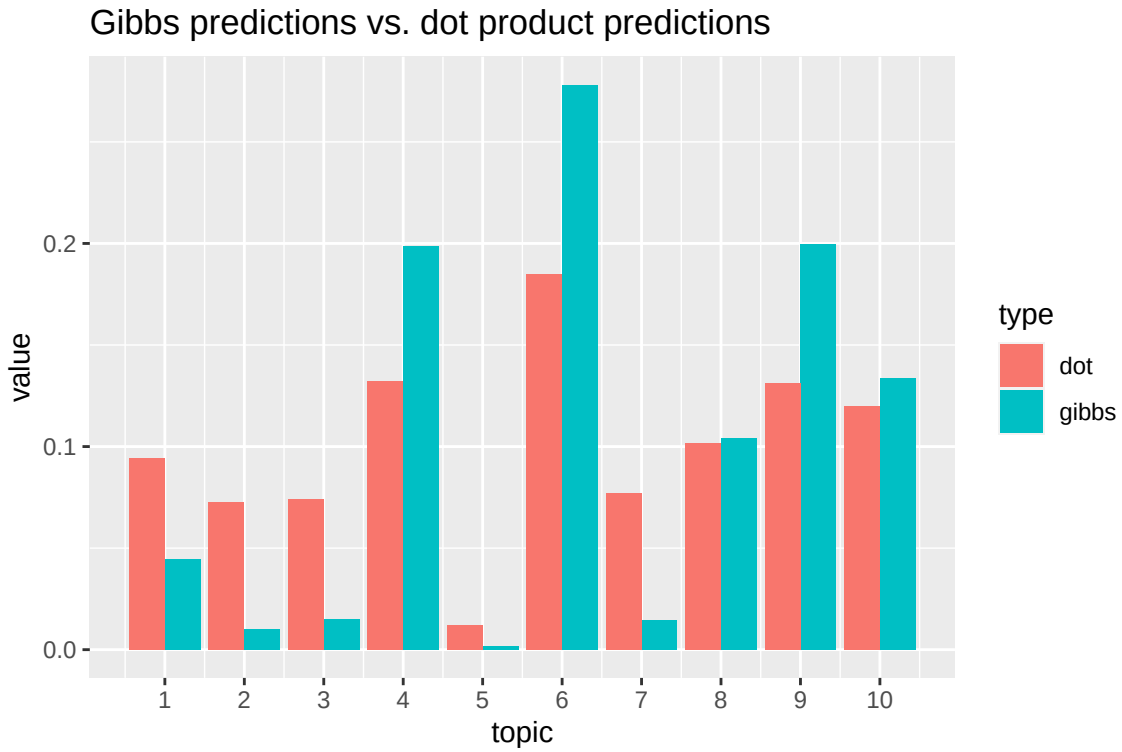
`augment` appends observation-level data, either $P(\text{topic}|\text{word}, \text{document})$ for each topic or the most probable topic for each word/document combination.

```
# append observation-level data
augmented_docs <- augment(lda, data = tidy_docs)


augmented_docs
```

```
# A tibble: 4,566 x 4
   document term            n topic
   <chr>    <chr>       <int> <int>
 1 8574224  adolescence     1     6
 2 8646901  adolescence     1     6
 3 8689019  adolescence     1     6
 4 8705323  adolescence     1     6
 5 8574224  age             1     6
 6 8705323  age             1     6
 7 8757072  age             1     6
 8 8823186  age             1     6
 9 8574224  application     1     6
10 8605875  application     1     6
# ... with 4,556 more rows
```

`tidylda` allows users to get predictions for new documents, using either Gibbs sampling (the default) or using a dot product with the $\mathbf{\Lambda}$ matrix. Below gets predictions using both techniques for a single document and plots them for comparison. However, users can get predictions for batches of documents by passing a sparse matrix, dense matrix, or numeric vector to `predict.tidylda` through the `new_data` argument.

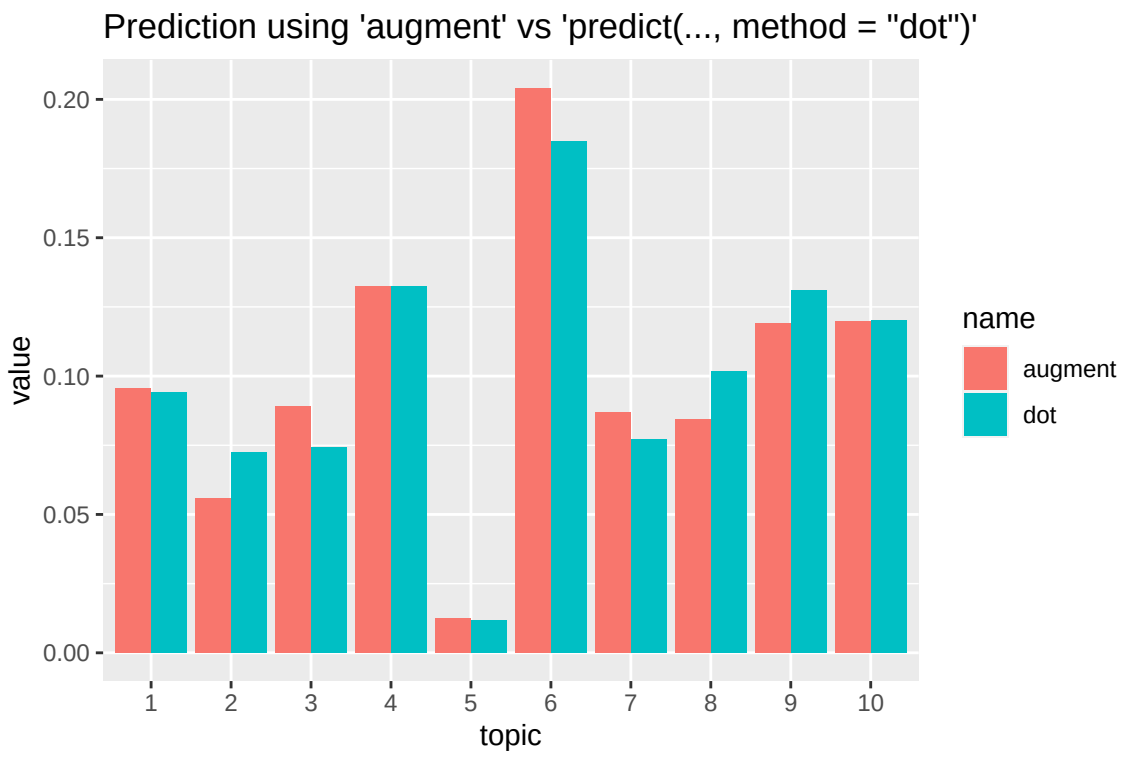
```
### predictions on held out data ---

# two methods: gibbs is cleaner and more technically correct in the bayesian sense
p_gibbs <- predict(lda, new_data = d2[1, ], iterations = 100, burnin = 75)


# dot is faster, less prone to error (e.g. underflow), noisier, and frequentist
p_dot <- predict(lda, new_data = d2[1, ], method = "dot")


# pull both together into a plot to compare
tibble(topic = 1:ncol(p_gibbs), gibbs = p_gibbs[1,], dot = p_dot[1, ]) %>%
  pivot_longer(cols = gibbs:dot, names_to = "type") %>%
  ggplot() +
  geom_bar(mapping = aes(x = topic, y = value, group = type, fill = type),
           stat = "identity", position="dodge") +
  scale_x_continuous(breaks = 1:10, labels = 1:10) +
  ggtitle("Gibbs predictions vs. dot product predictions")
```



Aggregating over terms on augmented data (as appended with `augment`) results in implicit prediction using the dot product for data used in training.

```
### Augment as an implicit prediction using the 'dot' method ----

# Aggregating over terms results in a distribution of topics over documents

# roughly equivalent to using the "dot" method of predictions.

augment_predict <-

  augment(lda, tidy_docs, "prob") %>%

  group_by(document) %>%

  select(-c(document, term)) %>%

  summarise_all(function(x) sum(x, na.rm = T))


# reformat for easy plotting

augment_predict <-

  as_tibble(t(augment_predict[, -c(1,2)]), .name_repair = "minimal")


colnames(augment_predict) <- unique(tidy_docs$document)


augment_predict$topic <- 1:nrow(augment_predict) %>% as.factor()


compare_mat <-

  augment_predict %>%

  select(

    topic,

    augment = matches(rownames(d2)[1])

  ) %>%

  mutate(

    augment = augment / sum(augment), # normalize to sum to 1
```

```
    dot = p_dot[1, ]
  ) %>%
  pivot_longer(cols = c(augment, dot))


ggplot(compare_mat) +
  geom_bar(aes(y = value, x = topic, group = name, fill = name),
           stat = "identity", position = "dodge") +
  labs(title = "Prediction using 'augment' vs 'predict(..., method = \"dot\")'")
```

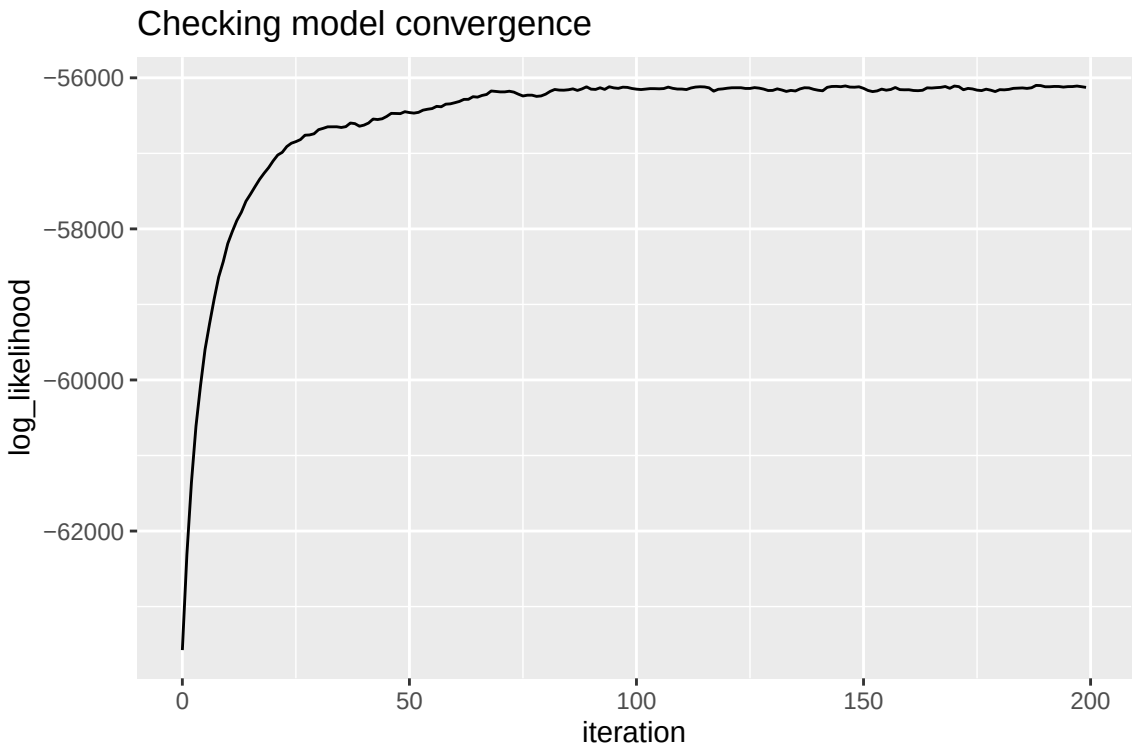


`tidylda` enables fine tuning pre-trained LDA models by adding new data using tLDA. (See Chapter 5.)

Below, we add the second half of our corpus and update the topic distributions.

```
### updating the model ----

# now that you have new documents, maybe you want to fold them into the model?

lda2 <- refit(
  object = lda,
```

```
  new_data = d2,
  iterations = 200,
  burnin = 175,
  calc_likelihood = TRUE,
  calc_r2 = TRUE
)


# we can do similar analyses
# did the model converge?
lda2$log_likelihood %>%
  ggplot(aes(x = iteration, y = log_likelihood)) +
  geom_line(color = "black") +
  ggtitle("Checking model convergence")
```



```
# look at the model overall
glance(lda2)
```

```
# A tibble: 1 x 5
  num_topics num_documents num_tokens iterations burnin
       <int>         <int>      <int>      <dbl>  <dbl>
1         10            49       2962        200    175
```

```
print(lda2)
```

```
A Latent Dirichlet Allocation Model of  10 topics,  49  documents, and  2962  tokens:
refit.tidylda(object = lda, new_data = d2, iterations = 200,
    burnin = 175, calc_likelihood = TRUE, calc_r2 = TRUE)

The model's R-squared is  0.174
The  5  most prevalent topics are:
# A tibble: 10 x 4
  topic prevalence coherence top_terms
  <dbl>      <dbl>     <dbl> <chr>
1     5       15.6    0.0677 research, program, cancer, health, disparities, ...
2     9       13.9    0.105  health, hiv, research, project, diabetes, ...
3     8       12.9    0.0336 injury, function, determine, cmybp, project, ...
4     2       11.2    0.0531 ptc, influenza, core, microbiome, brafv600e, ...
5     7       11.2    0.370  cancer, brain, tumor, treatment, aim, ...
# ... with 5 more rows

The  5  most coherent topics are:
# A tibble: 10 x 4
  topic prevalence coherence top_terms
```

```
  <dbl>      <dbl>     <dbl> <chr>
1     7      11.2      0.370 cancer, brain, tumor, treatment, aim, ...
2     3       7.34     0.319 lung, cells, ipf, cell, data, ...
3     4       9.09     0.278 cell, mast, cells, cns, mast cell, ...
4     1       8.24     0.157 muscle, inflammation, study, arterial, cdk5, ...
5     9      13.9      0.105 health, hiv, research, project, diabetes, ...
# ... with 5 more rows
```

```
# how does that compare to the old model?
print(lda)
```

```
A Latent Dirichlet Allocation Model of  10 topics,  50  documents, and  1524  tokens:
tidylda(data = d1, k = 10, iterations = 200, burnin = 175, alpha = 0.1,
    eta = 0.05, calc_likelihood = TRUE, calc_r2 = TRUE)

The model's R-squared is  0.2677
The  5  most prevalent topics are:
# A tibble: 10 x 4
  topic prevalence coherence top_terms
  <dbl>      <dbl>     <dbl> <chr>
1     7       11.9     0.140 cancer, dcis, cells, breast, rb, ...
2     1       11.0     0.22  cdk5, effects, v4, stiffening, wall, ...
3     6       10.9     0.295 sud, risk, factors, drug, imaging, ...
4    10       10.8     0.303 mitochondrial, plasticity, studies, redox, functio~
5     3       10.5     0.248 sleep, cell, lung, cells, specific, ...
# ... with 5 more rows

The  5  most coherent topics are:
```

```
# A tibble: 10 x 4
  topic prevalence coherence top_terms
  <dbl>      <dbl>     <dbl> <chr>
1     9      10.2      0.509 diabetes, numeracy, data, intervention, management~
2     8       8.83     0.494 function, cmybp, injury, fragment, 29 kda, ...
3    10      10.8      0.303 mitochondrial, plasticity, studies, redox, functio~
4     6      10.9      0.295 sud, risk, factors, drug, imaging, ...
5     5       8.06     0.258 research, disparities, program, cancer, mhirt, ...
# ... with 5 more rows
```

If we believe that there are additional topics in the new corpus, we can add them while using tLDA.

```
### updating the model and adding topics ----

# now that you have new documents, maybe you want to fold them into the model?
lda3 <- refit(
  object = lda,
  new_data = d2,
  additional_k = 2,
  iterations = 200,
  burnin = 175,
  calc_likelihood = TRUE,
  calc_r2 = TRUE
)


# look at the model summary object
print(lda3$summary, n = 12)
```

```
# A tibble: 12 x 4
   topic prevalence coherence top_terms
```

```
   <dbl>      <dbl>     <dbl> <chr>
 1     1       5.21    0.0111 inflammation, arterial, cdk5, fertility, effects,~
 2     2      10.8     0.137  influenza, core, response, vaccine, infection, ...
 3     3       8.09    0.173  lung, ipf, cells, cell, sleep, ...
 4     4       2.48    0.0315 cns, battery, cells, brain, capacity, ...
 5     5      15.4     0.110  research, program, cancer, dr, disparities, ...
 6     6       5.63    0.0353 risk, sud, clinical, factors, drug, ...
 7     7      10.3     0.364  cancer, brain, treatment, tumor, aim, ...
 8     8       4.77   -0.0554 injury, function, cmybp, fragment, determine, ...
 9     9      13.2     0.0809 health, hiv, research, diabetes, project, ...
10    10       6.23    0.0830 studies, mitochondrial, function, data, muscle, .~
11    11       9.41    0.411  mast, cell, mast cell, ma, fc, ...
12    12       8.44    0.223  microbiome, ptc, brafv600e, gut, crc, ...
```

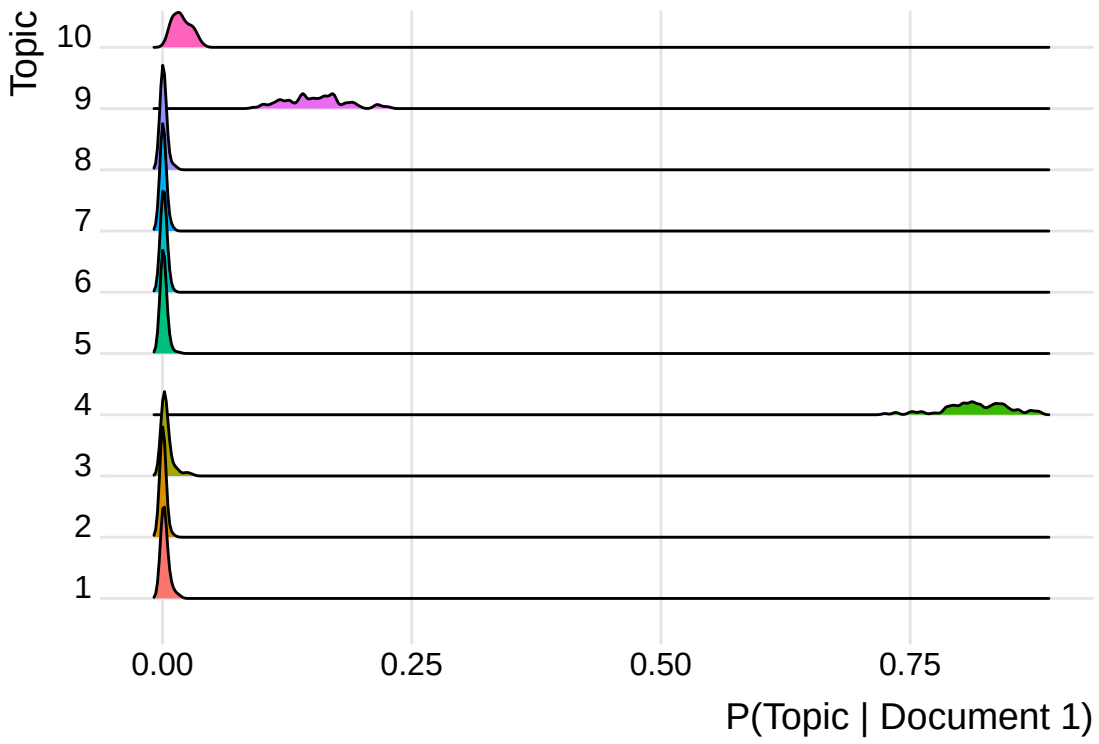
```
# how does that compare to the old model?
print(lda$summary, n = 10)
```

```
# A tibble: 10 x 4
   topic prevalence coherence top_terms
   <dbl>      <dbl>     <dbl> <chr>
 1     1      11.0      0.22  cdk5, effects, v4, stiffening, wall, ...
 2     2       8.07     0.18  research, responses, antibodies, natural, respons~
 3     3      10.5      0.248 sleep, cell, lung, cells, specific, ...
 4     4       9.76     0.145 cns, cells, brain, infection, based, ...
 5     5       8.06     0.258 research, disparities, program, cancer, mhirt, ...
 6     6      10.9      0.295 sud, risk, factors, drug, imaging, ...
 7     7      11.9      0.140 cancer, dcis, cells, breast, rb, ...
 8     8       8.83     0.494 function, cmybp, injury, fragment, 29 kda, ...
```

```
 9     9      10.2      0.509 diabetes, numeracy, data, intervention, managemen~
10    10      10.8      0.303 mitochondrial, plasticity, studies, redox, functi~
```

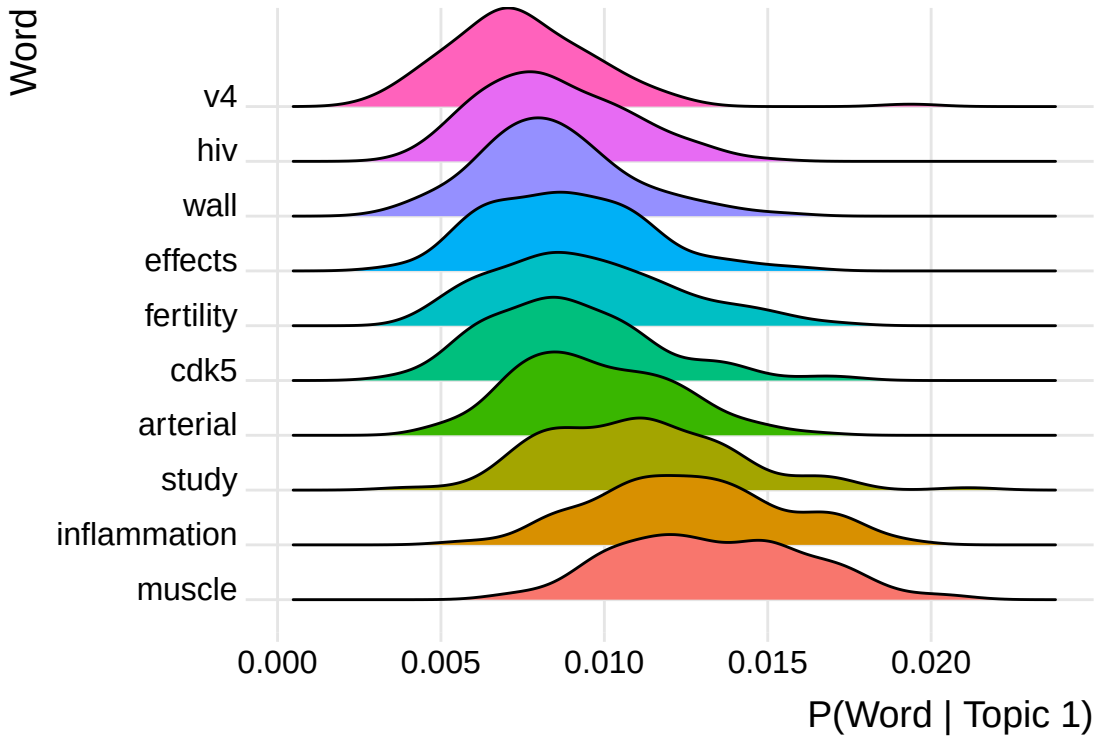
Finally, *tidylda* enables sampling from the posterior of a model to enable uncertainty quantification and other emperical posterior analyses. The below ridgeline plots show distributions for the topics in a single document and for the top 10 words in a single topic.

```
library(ggridges)


### Construct posterior samples of theta for document 1 ----

posterior_theta_1 <- posterior(lda2)


# make ridgeline plot

posterior_theta_1 %>%
  generate(matrix = "theta", which = 1, times = 100) %>%
  ggplot(aes(x = theta, y = factor(topic), fill = factor(topic))) +
  geom_density_ridges() +
  theme_ridges() +
  theme(legend.position = "none") +
  ylab("Topic") +
  xlab("P(Topic | Document 1)")
```




```
### Construct posterior samples of beta for topic 1 ----

posterior_beta_1 <- posterior(lda2)


# get top 10 words

token_list <- colnames(lda2$beta)[order(lda2$beta[1,], decreasing = TRUE)[1:10]]


# make ridgeline plot

posterior_beta_1 %>%
  generate(matrix = "beta", which = 1, times = 100) %>%
  filter(token %in% token_list) %>%
  mutate(token = factor(token, levels = token_list)) %>%
  ggplot(aes(x = beta, y = token, fill = token)) +
  geom_density_ridges() +
  theme_ridges() +
```

```
theme(legend.position = "none") +

ylab("Word") +

xlab("P(Word | Topic 1)")
```



References

[1] D. J. Newman and S. Block, "Probabilistic topic decomposition of an eighteenth-century american newspaper," *Journal of the American Society for Information Science and Technology*, vol. 57, no. 6, pp. 753–767, 2006.

[2] V. Perrone, M. Palma, S. Hengchen, A. Vatri, J. Q. Smith, and B. McGillivray, "GASC: Genre-aware semantic change for ancient greek," *arXiv preprint arXiv:1903.05587*, 2019.

[3] N. R. Ericsson, "Predicting fed forecasts," *Journal of Reviews on Global Economics*, vol. 6, pp. 175–180, 2017.

[4] U. Hahn, V. Hoste, and Z. Zhang, Eds., *Proceedings of the second workshop on economics and natural language processing*. Hong Kong: Association for Computational Linguistics, 2019.

[5] S. Volkova, D. Jurgens, D. Hovy, D. Bamman, and O. Tsur, Eds., *Proceedings of the third workshop on natural language processing and computational social science*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019.

[6] AMSTAT News, "New ASA interest group: Text analysis," 2020. `https://magazine.amstat.org/blog/2020/07/01/new-asa-interest-group-text-analysis/` (accessed Dec. 25, 2020).

[7] *Joint statistics meetings*. 2020.

[8] A. M. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[9] J. D. Angrist and J. S. Pischke, *Mostly harmless econometrics: An empiricist's companion*. Princeton University Press, 2008.

[10] J. M. Stanton, "Galton, pearson, and the peas: A brief history of linear regression for statistics instructors," *Journal of Statistics Education*, vol. 9, no. 3, 2001.

[11] G. C. Chow, "Tests of equality between sets of coefficients in two linear regressions," *Econometrica: Journal of the Econometric Society*, pp. 591–605, 1960.

[12] R. Anderson-Sprecher, "Model comparisons and r^2," *The American Statistician*, vol. 48, no. 2, pp. 113–117, 1994.

[13] N. N. Taleb, *Statistical consequences of fat tails*. New York, Ny: STEM Academic Press, 2020.

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv*, 2018.

[16] A. Vaswani *et al.*, "Attention Is All You Need," *arXiv*, 2017.

[17] A. Conneau *et al.*, "Unsupervised Cross-lingual Representation Learning at Scale," *arXiv*, 2019.

[18] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[19] T. B. Brown *et al.*, "Language Models are Few-Shot Learners," *arXiv*, 2020.

[20] T. Wolf *et al.*, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations*, 2020, p. 38—45.

[21] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[22] A. Conneau and G. Lample, "Cross-lingual language model pretraining," *Advances in neural information processing systems*, vol. 32, 2019.

[23] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "Spanbert: Improving pre-training by representing and predicting spans," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64–77, 2020.

[24] M. Lewis *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[25] K. Hao, "We read the paper that forced timnit gebru out of google. Here's what it says." 2020. `https://www.technologyreview.com/2020/12/04/1013294/google-ai-ethics-research-paper-forced-out-timnit-gebru/` (accessed Dec. 24, 2020).

[26] C. Li, "Demystifying GPT-3," 2020. `https://lambdalabs.com/blog/demystifying-gpt-3/` (accessed Dec. 24, 2020).

[27] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" 2021.

[28] J. Boyd-Graber, Y. Hu, and D. Mimno, *Applications of Topic Models*, vol. 11. now Publishers Incorporated, 2017.

[29] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, 2003.

[30] M. E. Roberts, B. M. Stewart, and E. M. Airoldi, "A Model of Text for Experimentation in the Social Sciences," *Journal of the American Statistical Association*, vol. 111, no. 515, pp. 988–1003, 2016, doi: 10.1080/01621459.2016.1141684.

[31] M. Erlin, "Topic modeling, epistemology, and the english and german novel," 2018.

[32] R. C. Team, *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2013.

[33] H. Wickham *et al.*, "Welcome to the tidyverse," *Journal of Open Source Software*, vol. 4, no. 43, p. 1686, 2019, doi: 10.21105/joss.01686.

[34] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[35] D. L. Lee, H. Chuang, and K. Seamons, "Document ranking and the vector-space model," *IEEE software*, vol. 14, no. 2, pp. 67–75, 1997.

[36] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990, doi: 10.1002/(sici)1097-4571(199009)41:6<391::aid-asi1>3.0.co;2-9.

[37] T. Hofmann, "Probabilistic latent semantic analysis," in *Proceedings of the fifteenth conference on uncertainty in artificial intelligence*, 1999, pp. 289–296.

[38] H. Shi, "A Principled Approach to the Evaluation of Topic Modeling Algorithms," PhD thesis, 2019.

[39] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, vol. 101, no. Supplement 1, pp. 5228–5235, 2004, doi: 10.1073/pnas.0307752101.

[40] W. Zhao *et al.*, "A heuristic approach to determine an appropriate number of topics in topic modeling," *BMC Bioinformatics*, vol. 16, no. Suppl 13, p. S8, 2015, doi: 10.1186/1471-2105-16-s13-s8.

[41] T. Jones, "Optimizing topic models for classification tasks," 2019. `https://www.jonesingfordata.com/talk/2019_11_09_dcr/` (accessed Dec. 27, 2020).

[42] I. Douven and W. Meijs, "Measuring coherence," *Synthese*, vol. 156, no. 3, pp. 405–425, 2007, doi: 10.1007/s11229-006-9131-z.

[43] M. Röder, A. Both, and A. Hinneburg, "Exploring the Space of Topic Coherence Measures," in *Proceedings of the eighth ACM international conference on web search and data mining*, 2015, pp. 399–408, doi: 10.1145/2684822.2685324.

[44] P. Pietilainen, "Properties of Semantic Coherence Measures - Case of Topic Models," 2020.

[45] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing Semantic Coherence in Topic Models," 2011.

[46] K. Stevens, P. Kegelmeyer, D. Andrzejewski, and D. Buttler, "Exploring Topic Coherence over many models and many topics," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, p. 952—961.

[47] A. B. Dieng, F. J. R. Ruiz, and D. M. Blei, "Topic Modeling in Embedding Spaces," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 439–453, 2020, doi: 10.1162/tacl_a_00325.

[48] Z. Chen and H. Doss, "Inference for the Number of Topics in the Latent Dirichlet Allocation Model via Bayesian Mixture Modeling," *Journal of Computational and Graphical Statistics*, 2019.

[49] J. Chang and J. Boyd-Graber, "Reading Tea Leaves: How Humans Interpret Topic Models," 2009.

[50] H. M. Wallach, D. Mimno, and A. McCallum, "Rethinking LDA: Why Priors Matter," 2009.

[51] G. K. Zipf, *Human behavior and the principle of least effort.* Oxford, England: Addison-Wesley Press, 1949.

[52] C. P. George and H. Doss, "Principled Selection of Hyperparameters in the Latent Dirichlet Allocation Model," *Journal of Machine Learnign Research*, vol. 18, no. 1, pp. 5937–5974, 2018.

[53] H. Shi, M. Gerlach, I. Diersen, D. Downey, and L. A. N. Amaral, "A new evaluation framework for topic modeling algorithms based on synthetic corpora," in *Proceedings of machine learning research*, 2019, vol. 89, p. 816—826, [Online]. Available: `http://proceedings.mlr.press/v89/shi19a.html`.

[54] L. AlSumait, D. Barbará, J. Gentle, and C. Domeniconi, "Topic Significance Ranking of LDA Generative Models," in *Joint european conference on machine learning and knowledge discovery in databases*, 2009, pp. 67–82.

[55] D. Newman, A. Asuncion, P. Smyth, and M. Welling, "Distributed Algorithms for Topic Models," *Journal of Machine Learning Research*, vol. 10, no. 8, 2009.

[56] M. Antoniak, "Tweet," 2020. `https://twitter.com/maria_antoniak/status/1338155254756462592` (accessed Dec. 25, 2020).

[57] L. Yao, D. Mimno, and A. McCallum, "Efficient methods for topic model inference on streaming document collections," pp. 937–946, 2009, doi: 10.1145/1557019.1557121.

[58] J. Yuan *et al.*, "LightLDA: Big Topic Models on Modest Computer Clusters," in *Proceedings of the 24th international conference on world wide web*, 2015, p. 1351—1361.

[59] A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. J. Smola, "Scalable inference in latent variable models," pp. 123–132, 2012, doi: 10.1145/2124295.2124312.

[60] J. Chen, K. Li, J. Zhu, and W. Chen, "WarpLDA: a Cache Efficient O(1) Algorithm for Latent Dirichlet Allocation," *arXiv*, 2015.

[61] A. Srivastava and C. Sutton, "Autoencoding variational inference for topic models," *arXiv preprint arXiv:1703.01488*, 2017.

[62] P. Rychlỳ, "Words' burstiness in language models." in *RASLAN*, 2011, pp. 131–137.

[63] D. Mimno and D. Blei, "Bayesian Checking for Topic Models," 2011.

[64] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Hierarchical Dirichlet Processes," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1566–1581, 2012, doi: 10.1198/016214506000000302.

[65] D. M. Blei and J. D. Lafferty, "Dynamic topic models," pp. 113–120, 2006, doi: 10.1145/1143844.1143859.

[66] J. McAuliffe and D. Blei, "Supervised topic models," *Advances in neural information processing systems*, vol. 20, p. 121—128, 2007.

[67] D. M. Blei and J. D. Lafferty, "A correlated topic model of Science," *The Annals of Applied Statistics*, vol. 1, no. 1, pp. 17–35, 2007, doi: 10.1214/07-aoas114.

[68] M. E. Roberts, B. M. Stewart, and D. Tingley, "stm : An R Package for Structural Topic Models," *Journal of Statistical Software*, vol. 91, no. 2, 2019, doi: 10.18637/jss.v091.i02.

[69] A. Srivastava and C. Sutton, "Autoencoding variational inference for topic models," *arXiv preprint arXiv:1703.01488*, 2017.

[70] M. R. Bhat, M. A. Kundroo, T. A. Tarray, and B. Agarwal, "Deep LDA: A new way to topic model," *Journal of Information and Optimization Sciences*, vol. 41, no. 3, pp. 823–834, 2020.

[71] L. Calvo-Bartolomé and J. Arenas-García, "Federated neural topic models," *arXiv preprint arXiv:2212.02269*, 2022.

[72] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[73] J. Eisenstein, *Introduction to natural language processing*. MIT Press, 2019.

[74] J. R. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in linguistic analysis*, 1957.

[75] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[76] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, pp. 3111–3119, 2013.

[77] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[78] M. E. Peters *et al.*, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[79] C. McCormick, "Word2vec tutorial-the skip-gram model." Retrieved, 2016.

[80] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," *Advances in neural information processing systems*, vol. 27, pp. 2177–2185, 2014.

[81] T. Mikolov, W. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 2013, pp. 746–751.

[82] A. Søgaard, I. Vulić, S. Ruder, and M. Faruqui, "Cross-lingual word embeddings," *Synthesis Lectures on Human Language Technologies*, vol. 12, no. 2, pp. 1–132, 2019.

[83] T. Jones, "Text embeddings," 2017. `https://cran.r-project.org/web/packages/textmineR/vignettes/d_text_embeddings.html` (accessed Dec. 25, 2020).

[84] A. Panigrahi, H. V. Simhadri, and C. Bhattacharyya, "Word2Sense : Sparse Interpretable Word Embeddings," in *Proceedings of the 57th annual meeting of the association for computational linguistics*, 2019, p. 5692—5705.

[85] D. C. Hoaglin and D. F. Andrews, "The Reporting of Computation-Based Results in Statistics," *The American Statistician*, vol. 29, no. 3, pp. 122–126, 1975, doi: 10.1080/00031305.1975.10477393.

[86] C. Cioffi-Revilla, "Power laws and non-equilibrium distributions of complexity in the social sciences," 2008.

[87] E. G. Altmann and M. Gerlach, "Statistical laws in linguistics," *arXiv*, 2015, doi: 10.1007/978-3-319-24403-7_2.

[88] L. Egghe, "Untangling herdan's law and heaps' law: Mathematical and informetric arguments," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 5, pp. 702–709, 2007.

[89] M. Gerlach and E. G. Altmann, "Scaling laws and fluctuations in the statistics of word frequencies," *New Journal of Physics*, vol. 16, no. 11, p. 113010, 2014, doi: 10.1088/1367-2630/16/11/113010.

[90] G. Altmann, "Prolegomena to menzerath's law," *Glottometrika*, vol. 2, no. 2, pp. 1–10, 1980.

[91] F. J. Damerau and B. B. Mandelbrot, "Tests of the degree of word clustering in samples of written english," *Linguistics*, vol. 11, no. 102, pp. 58–75, 1973.

[92] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE mobile computing and communications review*, vol. 5, no. 1, pp. 3–55, 2001.

[93] L. R. Taylor, "Aggregation, variance and the mean," *Nature*, vol. 189, no. 4766, pp. 732–735, 1961.

[94] R. F. i. Cancho and R. V. Sole, "Least effort and the origins of scaling in human language," *Proceedings of the National Academy of Sciences*, vol. 100, no. 3, p. 788—791, 2003.

[95] S. Arshad, S. Hu, and B. N. Ashraf, "Zipf's law and city size distribution: A survey of the literature and future research agenda," *Physica A: Statistical Mechanics and its Applications*, vol. 492, pp. 75–92, 2018.

[96] C. S. Gillespie, "Fitting Heavy Tailed Distributions: The {poweRlaw} Package," *Journal of Statistical Software*, vol. 64, no. 2, p. 1—16, 2015, [Online]. Available: `http://www.jstatsoft.org/v64/i02/`.

[97] R. V. Sole, "Scaling laws in language evolution," in *Power laws and non-equilibrium distributions of complexity in the social sciences*, C. Cioffi-Revilla, Ed. Unpublished, 2008.

[98] L. Q. Ha, E. I. Sicilia-Garcia, J. Ming, and F. J. Smith, "Extension of Zipf's law to words and phrases," 2002, doi: 10.3115/1072228.1072345.

[99] B. Mandelbrot, "Information theory and psycholinguistics," *BB Wolman and E*, 1965.

[100] S. Goldwater, T. L. Griffiths, and M. Johnson, "Producing Power-Law Distributions and Damping Word Frequencies with Two-Stage Language Models," *Journal of Machine Learning Research*, vol. 12, 2011.

[101] H. S. Heaps, *Information retrieval, computational and theoretical aspects*. Academic Press, 1978.

[102] A. Kornai, "Zipf's law outside the middle range," 1999.

[103] "Welcome to ExPORTER." `https://exporter.nih.gov/`.

[104] C. S. Gillespie, "Fitting heavy tailed distributions: The poweRlaw package," *Journal of Statistical Software*, vol. 64, no. 2, pp. 1–16, 2015, doi: 10.18637/jss.v064.i02.

[105] A. Schofield, M. Magnusson, and D. Mimno, "Pulling out the stops: Rethinking stopword removal for topic models," in *Proceedings of the 15th conference of the european chapter of the association for computational linguistics: Volume 2, short papers*, 2017, pp. 432–436.

[106] W. Buntine, "Lecture Notes in Computer Science," pp. 51–64, 2009, doi: 10.1007/978-3-642-05224-8_6.

[107] J. H. Lau, D. Newman, and T. Baldwin, "Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality," in *Proceedings of the 14th conference of the european chapter of the association for computational linguistics*, 2014, pp. 530–539, doi: 10.3115/v1/e14-1056.

[108] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh, "On Smoothing and Inference for Topic Models," *arXiv preprint arXiv:1205.2662*, 2012.

[109] V.-A. Nguyen, J. Boyd-Graber, and P. Resnik, "Sometimes Average is Best: The Importance of Averaging for Prediction using MCMC Inference in Topic Modeling," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1752–1757, doi: 10.3115/v1/d14-1182.

[110] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied linear statistical models*, vol. 4. Irwin Chicago, 1996.

[111] D. Cox and E. Snell, *Analysis of binary data*, vol. 32. CRC Press, 1989.

[112] D. McFadden, W. B. Tye, and K. Train, *An application of diagnostic tests for the independence from irrelevant alternatives property of the multinomial logit model*. Institute of Transportation Studies, University of California, 1977.

[113] J. Bruin, "FAQ. What are pseudo-r-squareds." UCLA: Academic Technology Services, Statistical Consulting Group. Retrieved . . . , 2006.

[114] "Grants & funding." `https://www.nih.gov/grants-funding`.

[115] T. Jones, "tidylda." 2020, [Online]. Available: `https://github.com/TommyJones/tidylda`.

[116] C. Wang, D. Blei, and D. Heckerman, "Continuous Time Dynamic Topic Models," *arXiv*, 2012.

[117] Y. Wang, E. Agichtein, and M. Benzi, "TM-LDA: Efficient online modeling of latent topic transitions in social media," in *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, 2012, pp. 123–131.

[118] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, "Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora," in *Proceedings of the 2009 conference on empirical methods in natural language processing*, 2009, p. 248—256.

[119] D. Andrzejewski and X. Zhu, "Latent Dirichlet Allocation with Topic-in-Set Knowledge," in *Proceedings of the NAACL HLT 2009 workshop on semi-supervised learning for natural language processing*, 2009, p. 43—48.

[120] J. Jagarlamudi, R. Udupa, and H. D. III, "Incorporating Lexical Priors into Topic Models," 2012.

[121] D. Andrzejewski, X. Zhu, and M. Craven, "Incorporating domain knowledge into topic modeling via Dirichlet Forest priors," pp. 25–32, 2009, doi: 10.1145/1553374.1553378.

[122] Y. Hu, J. Boyd-Graber, B. Satinoff, and A. Smith, "Interactive topic modeling," *Machine Learning*, vol. 95, no. 3, pp. 423–469, 2014, doi: 10.1007/s10994-013-5413-0.

[123] L. AlSumait, D. Barbará, and C. Domeniconi, "On-Line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking," *2008 Eighth IEEE International Conference on Data Mining*, pp. 3–12, 2008, doi: 10.1109/icdm.2008.140.

[124] M. D. Hoffman, D. M. Blei, and F. Bach, "Online Learning for Latent Dirichlet Allocation," *advances in neural information processing systems*, vol. 23, p. 856—864, 2010.

[125] J. Rieger, C. Jentsch, and J. Rahnenfuhrer, "RollingLDA: An Update Algorithm of Latent Dirichlet Allocation to Construct Consistent Time Series from Textual Data," 2021, [Online]. Available: http://www.statistik.tu-dortmund.de/fileadmin/user/_upload/Lehrstuehle/IWuS/Forschung/rollinglda.pdf.

[126] E. Hellinger, "Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen." *Journal für die reine und angewandte Mathematik*, vol. 1909, no. 136, pp. 210–271, 1909, doi: doi:10.1515/crll.1909.136.210.

[127] C. Fay, "Text Mining with R: A Tidy Approach," *Journal of Statistical Software*, vol. 83, no. Book Review 1, 2018, doi: doi: 10.18637/jss.v083.b01.

[128] H. Wickham, "ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics." 2020, [Online]. Available: https://CRAN.R-project.org/package=ggplot2.

[129] T. C. Team, "The Comprehensive R Archive Network." Accessed: Jan. 01, 2020. [Online]. Available: https://cran.r-project.org/.

[130] H. Wickham *et al.*, "Tidy data," *Journal of Statistical Software*, vol. 59, no. 10, pp. 1–23, 2014.

[131] D. Robinson, "broom: An R Package for Converting Statistical Analysis Objects Into Tidy Data Frames," *arXiv*, 2014.

[132] M. Khun and H. Wickham, "Tidymodels," 2018. https://www.tidymodels.org/ (accessed Jan. 01, 2021).

[133] J. Silge and D. Robinson, "tidytext: Text Mining and Analysis Using Tidy Data Principles in R," *The Journal of Open Source Software*, vol. 1, no. 3, p. 37, 2016, doi: 10.21105/joss.00037.

[134] T. Jones, "textmineR: Functions for Text Mining and Topic Modeling." 2015, [Online]. Available: https://CRAN.R-project.org/package=textmineR.

[135] B. Grün and K. Hornik, "topicmodels: An R Package for Fitting Topic Models," *Journal of Statistical Software*, vol. 40, no. 13, 2011.

[136] I. Feinerer, K. Hornik, and D. Meyer, "Text Mining Infrastructure in R," *Journal of Statistical Software*, vol. 25, no. 5, 2008.

[137] K. Benoit *et al.*, "quanteda: An R package for the quantitative analysis of textual data," *Journal of Open Source Software*, vol. 3, no. 30, p. 774, 2018, doi: 10.21105/joss.00774.

[138] D. Selivanov, M. Bickel, and Q. Wang, "text2vec." CRAN, 2020, [Online]. Available: `https://CRAN.R-project.org/package=text2vec`.

[139] J. Chang, "lda." 2015, [Online]. Available: `https://CRAN.R-project.org/package=lda`.

[140] T. Nguyen, J. Boyd-Graber, J. Lund, K. Seppi, and E. Ringger, "Is Your Anchor Going Up or Down? Fast and Accurate Supervised Topic Models," in *Human language technologies: The 2015 annual conference of the north american chapter of the ACL*, 2015, pp. 746–755, doi: 10.3115/v1/n15-1076.

[141] M. E. Roberts, B. M. Stewart, D. Tingley, and E. M. Airoldi, "The Structural Topic Model and Applied Social Science," 2013.

[142] A. K. McCallum, "MALLET: A Machine Learning for Language Toolkit," 2002.

[143] T. Minka, "Estimating a dirichlet distribution." Technical report, MIT, 2000.

[144] R. R. u. rek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*, 2010, p. 45—50.

[145] M. Khun, "Conventions for r modeling packages," 2019. https://tidymodels.github.io/model-implementation-principles/index.html (accessed Apr. 12, 2022).

[146] T. Jones, "mvrsquared: Compute the Coefficient of Determination for Vector or Matrix Outcomes." 2020, [Online]. Available: https://CRAN.R-project.org/package=mvrsquared.

Biography

Thomas Jones received a Bachelor of Arts in Economics from the College of William and Mary in 2009 and went on to receive a Master of Science in Mathematics and Statistics from Georgetown University in 2012. He is a veteran of the US Marine Corps, having served as an enlisted rifleman from 2000 to 2004. After receiving his Doctor of Philosophy in Computational Sciences and Informatics from George Mason University, he will continue his career as a technology entrepreneur.