**OVERVIEW**

This document details methods for studying time series of high-dimensional data. Our specific motivating context is the study of ecological communities comprised of species, where samples of organisms are collected over time (Christensen *et al.* 2018). We are interested in determining if the composition of communities changes over the course of the study, and if it does, we seek to quantify those dynamics. However, ecological communities are typically composed of many species relative to the number of samples collected (*i.e.,* the data are high-dimensional), which presents a challenge to time series modeling. To address this problem, we reduce the dimensionality of the community data prior to time series analysis (Christensen *et al.* 2018).

We accomplish this through a two-stage analysis referred to here as LDATS. The first stage (LDA) uses Latent Dirichlet Allocation (Blei *et al*. 2003)—a so-called “topic model”—to find a latent, simplified representation of the data, which is then analyzed in the second stage (TS) using multinomial Time Series models built upon the changepoint time series model of Western and Kleykamp (2004).

The LDA model derived and developed by Blei *et al*. (2003) has been successfully applied to ecological data by Valle *et al*. (2014) and Christensen *et al*. (2018). In relation to the linguistic models that motivated the original LDA description and notation (Blei *et al.* 2003), species are like terms (word options) in a vocabulary, communities are like linguistic topics, samples are like documents using the terms, and organisms within a sample are like words within documents (Valle *et al*. 2014). For the sake of maintaining the relationship between our LDATS model and the topic model derivation of LDA, we retain the original naming scheme (*i.e.*, observations of words within documents, latent grouping of terms into topics).

Here, the TS models include discrete (changepoint) and continuous temporal changes as well as covariate impacts and are fit using parallel tempering Markov Chain Monte Carlo (ptMCMC) methods (Earl and Deem 2005). To align our approach with existing topic models

This document describes the two-stage LDATS model in a unified mathematic setting and accompanies the codebase (Simonis *et al*. *In Development*). Because of the overlap in notation between classical LDA (Blei *et al.* 2013) and the time series models used here (Western and Kleykamp 2004) (*e.g.*, both use but with different meanings), we create a new notational set for use here and within the LDATS code and curate a translational list of terms (Table 1). Given that LDATS specifically uses LDA as the first stage and that our methods build upon topic models, in instances of notational overlap between the LDA and TS components, we defer to the original LDA usage.

We do make one important deviation from the original LDA notation (Blei *et al.* 2003) to clarify the dimensionality of parameters. Specifically, we use the lowercase, regular type letter (*e.g.,* ) to indicate a singular value; the lowercase, boldface letter (*e.g.,* )to indicate a vector of values; and the capital, boldface letter (*e.g.,* ) to indicate a matrix of values.

**STAGE ONE: DIMENSION REDUCTION**

The first stage of the analysis is focused on reducing the raw, high-dimensional data to a lower dimensional representation of the information contained in the data using Latent Dirichlet Allocation (Blei *et al*. 2003). This section describes the application of the Variational Expectation Maximization version of the LDA model derived and developed by Blei *et al*. (2003), with no novel developments.

A corpus (set of documents) consists of total documents comprising total words from total terms. Each document (in ) consists of words ( in ) assigned to one of in terms. The total number of words in the corpus is the sum of the words within each document:

LDA involves grouping the terms into in total latent (unobserved) component topics, such that “component topic” means a group of terms that tend to be found together in specific proportions. The allocation process (Blei *et al.* 2003)allows that individual terms can be assigned to multiple component topics. The total number of latent topics is also unknown, and for the present approach is fixed *a priori* within a given Stage 1 (LDA) model in (note the difference between , the total number of Stage 1 models and , the total number of documents).

Each word within a document has an observed term identity, notated as , and a latent topic membership, notated as . Because there are varying numbers of words in each document, we use a vector structure to hold word-level data across the study. The term identities of all words within document are (or just ) and the term identities of all words across all documents are (or simply ), an -length vector. Similarly, the topic identities of all words in document are represented as (or simply ) and the topic identities of all words across all documents are (or just ), an -length vector. Across all words from all documents, is the latent state (topic identity) and is what is observed (term identity).

For a specific model with a total number of topics , the word-level topic distribution within a document (*i.e.*, the allocation of an words among the possible topics) is a -dimension categorical random variable (equivalent to a multinomial variable with sample size 1) described by probabilities held in vector () and collated across samples into -dimension matrix . Thus, the topic identity of a word within a document is

The vector of topic probabilities within a document () is described by a -dimensional Dirichlet distribution with concentration parameters ,which we assume do not change among documents (*i.e.*, ). Further, we assume that the concentration parameters are symmetric (*i.e.,* ), thereby reducing the distribution to reliance on a single parameter :

which is an unknown parameter to be estimated.

The word-level topic distribution within a document (*i.e.*, the allocation of an words among the possible terms) is a -dimension categorical random variable contingent upon the topic identity of the word and defined by probabilities , where the sum across terms within a topic for a given document is 1 (*i.e.*, ). The probabilities across all topics within a document are held in a -dimension matrix (), which we assume is constant across documents, (), thus the term state of an word within a document is

which is equivalent to a categorical variable defined by the row of interest in given topic identity

The components of are unknown parameters to be estimated.

The inferential problem of interest lies in determining the posterior distribution of the latent quantities (the topic probabilities and states ), given the observations () and fitted parameters ( and )

an equation which obviously necessitates an estimation of the parameters and . From a parameter estimation standpoint, we are concerned with the probability of the suite of observations () given the parameters and , or the likelihood () of the parameters given the data:

However, for fitting purposes, we are interested in the log-likelihood () of the parameters, given the data

The log-likelihood of (the data across all samples) is the sum of the logged probabilities of each document’s data given the parameters (under the assumption that all documents are derived from the same parameters):

The probability of a document’s data given the parameters () can be decomposed into the product of [1] the word-by-word term-identity distributions () and [2] the sample-level community distribution (), integrated over the uncertainty in the latent community distribution, :

The word-level topic-identity distribution can be further decomposed into the product of [1] the term identity distribution given the topic identity and the unknown parameter matrix () and [2] the topic identity distribution given the latent topic distribution (), integrated (summed due to discreteness) over the uncertainty in topic type

The probability of a single document’s observations can therefore be decomposed into the product of [A] the product of the word-by-word topic distributions, each integrated over the uncertainty in topic type (which are themselves the product of [1] the term identity given the topic identity and the unknown parameters and [2] the community identity given the latent parameter ) and [B] the probability of the latent parameter given the unknown parameter , integrated over the uncertainty in

This is then scaled up to the probability of the entire corpus under the unknown parameters and , which is the product of the sample-level probabilities given those parameters:

The probabilities for in cannot be tractably estimated due to the coupling of (and thus ) and in the summation over latent topics (Blei *et al*. 2003). To address this, we use a variational approximation to the equations that decouples the parameters, and which we fit using the expectation-maximization routine (aka VEM for Variational Expectation Maximization; Blei *et al*. 2003).

To accomplish this, we endow the model with free variational parameters ( and ) that characterize a family of distributions (notated by to distinguish from ) which provide a lower bound on the log likelihood. is an -dimension matrix similar to . However, the rows of (row corresponding to document : or simply ) correspond to the concentration parameters of a -dimension Dirichlet distribution and therefore are not constrained to sum to 1. is an -dimensional matrix, where the rows correspond to the words across documents (indexed akin to and ) and the columns correspond to the topics. describes the probability that word within document is from topic , and (or simply ) is a -length vector of probabilities that define the categorical distribution controlling that word’s topic identity, where . is composed of document-specific matrices, where the matrix for a specific document is -dimensional and notated .

In comparison to and , the variational parameters are document-specific (able to vary among documents), and so are found separately for each document. For a specific document , the variational distribution is

As the Expectation Step (“E-Step”) in the VEM algorithm, the distribution can be used to find a tight lower bound on by optimizing the variational parameters and (*i.e.*,find and , where the asterisks notate optimal values) with respect to minimizing the Kullback-Leibler Divergence () between and the true posterior :

Minimization of the distance is achieved through an iterative fixed-point method, where the derivative of the is set to zero, producing a pair of update equations. First, the parameters describing the topic allocation of each word () are updated based on the topic distribution for the document ():

where the expected value of the (log-scale) topic probability is calculated using the digamma function (), which is the logarithmic derivative of the gamma function (), a quantity that is calculated through Taylor approximation:

And then, the parameters describing the topic distribution for the document () are updated based on the word-level topic distributions for the sample ():

The update equations are alternated until the bound converges (*i.e.*, the updates do not yield changes to the parameters), at which point the document-specific variation parameters have been optimized ( and have been found) for the set of main parameters ().

Similarly to the overall log likelihood for the set of documents being equal to the sum of the log likelihoods for the individual documents, the variational lower bound () for all of the documents is equal to the sum of the variational lower bounds for the individual documents:

The complete E-Step maximizes this overall lower bound with respect to the full variational parameters and (*i.e.*, given the main model parameters). The Maximization Step (“M-Step”) maximizes the overall lower bound with respect to the main model parameters and (*i.e.*, given the optimal variational parameters). This corresponds to obtaining maximum likelihood values of the model parameters using expected sufficient statistics for each sample under the approximate posterior calculated in the E-Step.

The update for the topic-level term distribution () can be written analytically:

where is an indicator variable based on the term identity () of the observed word ():

The update for the concentration parameter underlying the document-level topic distribution () requires an iterative approach to find a stationary point estimate. The optimization is conducted using the Newton-Raphson method (Ronning 1989), which repeats

(where is the hessian matrix and is the gradient) until convergence. Having updated the main model parameters (the M-Step), a new iteration of the E-Step followed by the M-Step is conducted, and the E-Step and M-Step are alternated until the lower bound of the log-likelihood converges. Thus, the VEM approach can be considered coordinate ascent in (the space defined by the lower bound).

Once the log-likelihood has converged, the VEM algorithm has arrived at approximate maximum likelihood estimates for the model parameters ( and ) given the full set of observations () for the specific Stage 1 model (notated by the component in the superscripts). This estimation procedure is executed using the LDA function in the topicmodels package (v0.2-7; Grun and Hornik 2011) in R (v 3.5.1; R Core Team 2018), which leverages C code written by Blei *et al*. (2003).

Given the fit of a specific Stage 1 model (), we can then consider multiple Stage 1 models to determine the most parsimonious number of topics . Specifically, we use as our Stage 1 model selection criterion, defined for a specific LDA model :

where is the number of parameters in the model: corresponding to and corresponding to each entry in , a -dimensional matrix (Grün and Hornik 2011). If small sample size is a concern with respect to the degrees of freedom being used, we can use the correction based on the number of observations, here corpus size (, Grün and Hornik 2011):

Because of the use of multiple iterative optimization routines (which require starting values to be drawn at random) to solve otherwise intractable likelihood functions, it is critical to account for the potential influence of starting values on analytical results. Here, we accomplish this by running multiple models with the same number of topics () using different starting values, assigned through the random number generator seed ().

Specifically, we use replicates ( in ) at each number of topics from 2 to , the total number of topics to be explored. The minimal number of topics is set to 2 by the current coding implementation of the LDA algorithm (Blei *et al.* 2003, Grün and Hornik 2011), although it is possible to generalize the code to include the limiting case of a single topic. Thus, the total number of models in the Stage 1 () is

The optimal (according to ) LDA model () is determined by

and has a corresponding set of parameters:

Having found the optimal model, we can obtain the posterior point estimates for the document-level topic probabilities (held in a -length vector) by normalizing the vector of optimal values of the variational Dirichlet concentration parameters (taken from the final step of the VEM algorithm in model ) for the document () so it sums to one and is thereby a proper proportion (notated by the bar accent ):

The posterior point estimates of the topic proportions across the documents are then held in an -dimension matrix , corresponding to the optimal (from an “ based on VEM inference”-perspective) decomposition of the word-level data to topic-level data. This matrix forms the multivariate response variable analyzed in the time series model, as outlined in the next section.

**MULTINOMIAL TIME SERIES MODEL**

The approach we take in analyzing the temporal topic dynamics leverages change point time series analyses (Western and Kleykamp 2004) that allow for continuous and discrete changes in dynamics, combined with multinomial generalized regression (Ripley 1996, Venables and Ripley 2002) to manage the proportional nature of the response variable (document-level topic compositions).

The output from the decomposition detailed in the previous section () is a matrix of multinomial variables corresponding to proportions of each topic in present in each document in . Further, each document has a weight , which is a function of the number of words relative to the maximum number of words across all documents:

thereby allowing us to account for variable efforts (numbers of words) among documents (with vector ).

We are interested in quantifying changes in topic composition over time, and so we define the time of document to be and the vector of all document times to be or simply . defines the temporal relationship among documents, and must be a discrete variable. In addition, we collate total covariates, indexed as in and measured for each document. The value of a particular covariate for a specific document is and the set of covariates for the document is a vector or simply . All of the covariates (including the intercept) across all of the documents are held in , an -dimensional matrix.

Although the analysis of the topic proportions is via “time series” models, the time of the document () *per se* does not necessarily enter every Stage 2 (time series) model ( in ) directly. Rather, depending upon the model, the time of the documents may control the application of the predictor variables in the model (in the case of discrete change points) or may directly influence quantitative values of predictors (in the case of continuous time impacts). Future developments of the model will allow time to also impact the non-independence of the data via autocorrelation structures, but presently autocorrelation is not included.

A given Stage 2 model has a non-negative integer number of discrete change points () that divide the time series into distinct segments or “chunks” ( in ), where the number of chunks () is always one more than the number of change points (). If there are change points (*i.e.,* ), then their locations (for the in change points) are unknown parameters to be estimated. A specific change point ’s location is represented by and the set of change points are represented by the –length vector or simply .

To define the full deconstruction of the time series into chunks, we augment the vector of change point locations with the time before the minimum () and maximum times () in the series, generating the –length vector , where the overbrace accent references the addition of the fixed time range to the unknown change points. In the instance that there are no change points (*i.e.,* ), is still defined, but now is simply a length-2 vector including the minimum and maximum times, and therefore includes no unknown change point locations to be determined.

We use the locations to assign the documents to specific chunks via a document ↔ segment indicator function (), which returns an indication if document belongs to chunk (0 for no or 1 for yes) that we notate as . operates on the timestamp of the document () and the start and end time of a given chunk , defined by its start time and end time :

where the start time for a segment is the first time step after the previous change point and the end time is the timestep of the change point (*i.e.*, a chance point location of would break the time series of documents with times into chunks corresponding to and). This indicator function allows us to identify to which chunk each document belongs and we use this identity to deconstruct the full output matrix from the first stage model () into matrices ( in ), each of which is analyzed separately. For each chunk in , the function produces an -length vector of 0s and 1s ( or simply ). is a submatrix of :

where we use the parenthetical notation to mean “of chunk ” and the square bracket indicates the subset of the matrix as , such that only the rows corresponding to 1s in are included, and all columns are included.

We note an important distinction between the original change point model (Western and Kleykamp 2004) and our model here with respect to the application of the change points to the regression. In Western and Kleykamp’s (2004) model, the indicator function is only applied within the “fixed-effects” component of a singular regression fit, such that all chunks have the same error terms. In our model, however, the data are fully separated into multiple independent regression fits, as a result applying the indicator to both the “fixed-effects” and error components.

Further, the original change point model (Western and Kleykamp 2004) assumed a univariate normal response variable, whereas our response data are multivariate and non-normal. Specifically, our response variable is a set of multinomial probabilities, each of which must be non-negative and which must sum to 1. To accommodate this structure, we take a generalized linear model approach and, within each given chunk of time, model the data using a log-linear multinomial (aka multinomial logit, softmax) regression (Ripley 1996, Venables and Ripley 2002).

Within chunk of time , we are interested in describing the proportion of each topic in in a document in . We notate this proportion as , which is the probability that a word in document comes from topic . Following general linear modeling, we wish to define the proportion in terms of predictor covariates (, a vector of covariates—including the intercept—for document ) and coefficients (, a vector of the coefficients associated with the specific topic during this chunk of time).

However, we must acknowledge the two constraints on the response variable (proportions): that each be non-negative and that they sum to 1. We accommodate the non-negative constraint by using a log-linear model that relates the log of the proportion to the linear predictors. We handle the sum-to-1 constraint by subtracting a “constant” value for the document (, where is known as the partition function) from all topics’ proportions:

or, with respect to the raw proportions,

where is a column matrix and is a row matrix.

Given the sum-to-1 (), we can define :

Thus, the specific equation for the proportion of topic in document can be written as

replacing the with in the denominator summation to avoid confusion with the focal topic .

However, because of the sum-to-1 constraint, only of the proportions ( in ) are uniquely identifiable (and, by extension only of the parameter vectors in are uniquely identifiable). Thus, we define the augmented parameter vectors in in relation to the parameters of the first topic ():

This sets the parameters associated with the first topic to be 0 (, and thus ), thereby reducing the number of free parameter vectors (and by extension the number of proportions that are estimated), by 1 to the that we are able to fit. This results in the modified equation

where . Thus, the parameters in the vectors to are the unknowns to be fit for this specific chunk of time . This equation is known as the softmax function, and so we also can write

For notational condensation, we combine all of the parameter vectors to (including the vector of 0s in ) into a matrix  ( is capital ), which has columns and a number of rows equal to the number of coefficients in the model ()including the intercept (*i.e.*, the length of ). This allows us to further condense the probability equation to

thereby facilitating use of the generalized linear modeling framework. We expand the model to predict the proportions across all of the topics within the document, which means we can drop the input and produce the full set of values from the softmax function, which is a length- row vector corresponding to the topic distribution of a single document:

We then expand the model across all documents within the chunk of time

where the covariates are held in a matrix () with the number of columns equal to the number of coefficients () and the number of rows equal to the number of documents in the chunk (). That is, is a series of row vectors.

As stated earlier, it is at this level (time chunks of documents) of our model where the regression parameters are estimated, differing from the original model formulation of Western and Kleykamp (2004), which fits the parameters across all observations within a single model. Thus, before we expand the model upwards across time chunks, we detail the estimation of the regression coefficients for the chunk .

This equation relates directly to the generalized linear modeling equation that is typically written as or , where is the so-called link function and is the inverse link function. The component from our model missing thus far is uncertainty, which we acknowledge now through the inclusion of the expected value notation ( is the expected value of ):

thereby producing a generalized linear model equation, wherein our link function is the multinomial logit (similar to the binomial logit being the link function of a logistic regression) and our inverse link is the softmax function (akin to the logistic function being the inverse link of a logistic regression). Recognizing the uncertainty in the relationship between the observations () and the predictions () allows us to formulate an estimation problem whereby we are interested in finding an optimal set of parameters given the data. To do this, we leverage the conditional (in the sense of being conditional on the full input complex via the partition function) probability for a given document :

Note that the index on the expected probability is outside of the bracket () to indicate that the indexing was done *after* the application of the softmax function. The conditional probability of all of the documents within the chunk is the weighted joint probability of each of the documents, and is the conditional likelihood () of the parameters, given the data:

Where is the weight of the document and is the vector of weights for all of the documents in . We use to create a loss function (for minimization during optimization), defined as the negative log-likelihood function ():

which is equivalent to the within-document cross-entropy between the observed () and predicted () distributions. Given that the negative log likelihood function is convex, there exists a set of parameters (the asterisk again indicates the global optimal value) where the derivative of the function is 0. This optimal set of parameters is jointly estimated as maximum *a posterior* (MAP; *i.e.*, posterior mode) values, where the MAP estimation is an extension of the maximum likelihood (ML) estimation with a regularization (*i.e.*, penalty) to smooth the likelihood surface and avoid pathological solutions. This is equivalent to taking a Bayesian approach and adding a Gaussian prior on the model parameters.

Specifically here, a classical “weight decay” (so called because the regressors in the neural networks used to fit the model are confusingly named “weights”) is used, where a decay parameter () scales the regularization penalty, which is the sum of squared model coefficients:

Adding this term to the negative log-likelihood equation, we have an updated regularized loss function:

which is now strictly convex. The original formulation of the LDATS model (Christensen *et al*. 2018) did not explicitly include the decay penalization (that is, was assumed to be 0), but including a small penalty (; Ripley 1993, Ripley 1996) can aid in finding the optimal solution in multinomial regression, and so the potential to explicitly set is now included in the software package. Note, however, that the regularization requires rescaling all coefficients to be on scales of about .

The Stage 2 model for a given chunk of time is fit using the multinom and nnet functions within the nnet package (v7.3-12; Venables and Ripley 2002) in R (R Core Team 2018), which formulate the regression as a single-hidden-layer neural network with skip-layer connections. The solution is found with the gradient-based optimization algorithm known as the Broyden–Fletcher–Goldfarb–Shanno or BFGS Algorithm (Brayden 1970, Fletcher 1970, Goldfarb 1970, Shanno 1970), a quasi-Newtonian iterative searching method for non-linear optimization. The BFGS Algorithm works efficiently by not calculating the Hessian (matrix of second derivatives) at every step in the optimization, but rather approximating the Hessian by comparing successive gradient vectors.

Semantically, however, it is imprecise, to call the application of BFGS here a “gradient” method, as gradients are technically only defined for scalar functions (which map vectors of values to scalar output), whereas the softmax is a vector function (it maps a vector of values to a vector of values). Rather, we define a general derivative of the loss function, contained in the Jacobian matrix, whose components cover all combinations of input element (each coefficient for each topic) and output element (loss). Thus, for each entry in the Jacobian, we must specify with respect which coefficient-topic we want the partial derivative of the loss with respect to.

The full Jacobian of the loss equation has an extensive derivation based on the nuances of the data set being analyzed but which collapses neatly. In Appendix 1, we fully derive the Jacobian for the loss function based on probability responses (rather than one-hot vector responses) with linear predictors (covariates) acting on the predicted probability through matrix multiplication with parameters (thereby allowing linear modeling) and with a flexible weight decay penalty. Here we highlight the derivation of the matrix.

To reduce clutter, we name the total loss value across all documents , the total loss value for documents within a chunk , and the loss value of a single document ; the cross entropy function ; the softmax function ; the matrix multiplication ; and the penalty function , allowing us to write

for a single document and

for all documents within the chunk of time together, where is the document weight (relative number of words to maximum). To reduce clutter, we condense into , recognizing that the intermediate functions are still being executed. Thus, we have

The partial derivative of with respect to a parameter-topic combination is notated as (where capital refers to the matrix structure being derived and the subscript indicates which parameter the partial derivative is with respect to):

Allowing us to populate our Jacobian for the complete loss calculation with penalty

We fully detail the loss Jacobian derivation and component details in Appendix 1.

With a complete Jacobian for our loss function in addition to our loss function, we can now execute the BFGS Algorithm (Brayden 1970, Fletcher 1970, Goldfarb 1970, Shanno 1970) to estimate the parameters in and calculate the loss value for the documents in the chunk of time,

As to be expected, the next step is to combine the within-chunk models across the chunks of time. While we do expand the model up across time chunks, we do so with a formulation that differs slightly from Western and Kleykamp (2004) with respect in particular to parameter estimation. Most specifically, in our model, there is no covariance among regression parameters fit in different chunks, whereas the Western and Kleykamp (2004) model allows for covariance among all of the regression parameters. Estimating the covariance among many parameters is computationally very expensive, and Western and Kleykamp’s (2004) model included a single change point, simple predictors, and a normal response variable—all three assumptions we have relaxed. Given our hypothesis of the change points being discrete and abrupt, it is reasonable to codify that in our model formulation via the assumption of no covariance among parameters in different time chunks.

*Not sure if needed exactly as is*

To do this, we first define an expanded version of the parameter matrix , which has columns and a number of rows equal to the number of coefficients across all of the time chunks:

We next expand the covariates to the block-diagonal matrix , which has a number of columns equal to the number of coefficients across all of the time chunks and rows:

Finally, we compute the full matrix of topic-by-document proportions:

which relates directly to the generalized linear modeling equation that is typically written as or , where is the so-called link function and is the inverse link function. The component from our model missing thus far is uncertainty, which we acknowledge now through the inclusion of the expected value notation ( is the expected value of ):

thereby producing a generalized linear model equation, wherein our link function is the multinomial logit (similar to the binomial logit being the link function of a logistic regression) and our inverse link is the softmax function (akin to the logistic function being the inverse link of a logistic regression).

Now can move to the rest of what W and K use to describe the model

Add the var-cov matrix and the distributions of the priors and the change points

Changepoint model (Western and Kleykamp 2004)

Parallel tempering MCMC (Earl and Deem 2005)

**LITERATURE CITED**

Blei, D. M., A. Y. Ng, and M. I Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* **3**: 993-1022.

Broyden, C. G. 1970. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and Its Applications* **6**:76–90.

Christensen, E. M., D. J. Harris, and S. K. M. Ernest. 2018. Long-term community change through multiple rapid transitions in a desert rodent community. *Ecology* **99**:1523–1529.

Earl, D. J. and M. W. Deem. 2005. Parallel tempering: theory, applications, and new perspectives. *Physical Chemistry Chemical Physics* **7**: 3910-3916.

Fletcher, R. 1970. A new approach to variable metric algorithms. *Computer Journal* **13**:317-322.

Goldfarb, D. 1970. A family of variable metric updates derived by variational means. *Mathematics of Computation* **24**:23-26.

Grün, B. and K. Hornik. 2011. topicmodels: an R package for fitting topic models. *Journal of Statistical Software* **40**:13. http://dx.doi.org/10.18637/jss.v040.i13

R Core Team. 2018. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.

Ripley, B. D. 1993. Statistical aspects of neural networks. In *Networks and Chaos: Statistical and Probabilistic Aspects*, eds: O. E. Barndorff-Nielsen, J. L. Jensen, and W. S. Kendall. pp

40-123. Chapman Hall, London.

Ripley, B. D. 1996. Pattern Recognition and Neural Networks. Cambridge.

Ronning, G. 1989. Maximum likelihood estimation of Dirichlet distributions. *Journal of Statistical Computation and Simulation* **34**:215-221.

Shanno, David F. July 1970. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation* **24**:647-656.

Simonis, J. L., E. M. Christensen, D. J. Harris, H. Ye, R. Diaz, and S. K. Morgan Ernest. *In Development*. LDATS: Latent Dirichlet Allocation coupled with Time Series analyses. R package v 0.0.6. https://github.com/weecology/LDATS

Valle, D., B. Balser, C. W. Woodall, and R. Chazdon. 2014. Decomposing biodiversity data using the Latent Dirichlet Allocation model, a probabilistic multivariate statistical model. *Ecology Letters* **17**: 1591-1601.

Venables, W. N. and B. D. Ripley. 2002. Modern and Applies Statistics with S. Fourth Edition. Springer.

Western, B. and M. Kleykamp. 2004. A Bayesian change point model for historical time series analysis. *Political Analysis* **12**:354-374.

Appendix 1: Derivation of the Jacobian for the loss function.

To construct the Jacobian of the loss function, we must first recognize the order of operations of the component multivariate functions within the loss function:

which can be further generalized to

This highlights the chained aspect of the non-penalty functions (the cross entropy is calculated using the output of the softmax, which uses the output of the matrix multiplication). To reduce clutter in our derivation of the Jacobian, we name the cross entropy function , the softmax function , the matrix multiplication , and the penalty function , allowing us to write

The derivative of the non-penalty functions can be expanded using the multivariate chain rule. For two general functions and chained as (where contains the multivariate input values), we can write the function composite using the ring operator as . We then take the multivariate derivative (denoted as function ) of the composite:

where is the dot product operator. Thus, the derivative of of of is the dot product of the derivative of evaluated at of and the derivative of evaluated at . Using the chain rule, we now expand the derivative of the loss function applied to a specific document within a specific chunk (). We start by expanding the outer layers ( and ):

Note that to reduce clutter we drop because we are taking the derivative with respect to ( is still a relevant part of , as will be apparent later, but it is not the focus of the derivation and so is functionally a constant). We next expand the inner layers ( and ) by working only with the right-hand-side of the dot product:

Combining these chained results gives the full Jacobian for the non-penalized component of the loss function applied to a single document:

which is the dot product between [1] the dot product between [a] derivative of the cross-entropy function (evaluated at the softmax of the matrix multiplication of the coefficients and covariates) and [b] the derivative of the softmax (evaluated at the matrix multiplication of the coefficients and covariates), and [2] the derivative of the matrix multiplication evaluated at the coefficients and covariates.

We now define the derivative matrices (Jacobians) of each of the functions , , and . To aid in this, we consider that the Jacobian of a function contains the partial derivatives of each output (, a general component of ) with respect to each input ( a general component of , which can be written generally as or . The Jacobian for a given function then maps the input to the output, and so has dimensions equal to the number of output classes the number of input classes.

The function maps the matrix () to the dimensions of () by left-multiplying by the covariate row matrix (). Thus, its Jacobian has rows and columns:

For notekeeping purposes, the entry in the coefficient matrix corresponds to the column in the Jacobian. In effect, the coefficient matrix is linearized in column-major order (iterating through all covariates within a given topic before progressing to the next topic). Recall that the matrix multiplication used to generate an output element (row) (for in ) is just a linear combination of components

and therefore, the partial derivative of the output element with respect to an input element is simply the relevant covariate or 0 (when beyond the relevant part of the Jacobian):

Moving next to the softmax function, maps to , both of which are of dimension , because we are working within a single document. Thus, its Jacobian has rows and columns:

We can write a generalized equation for the entries by describing the partial derivative of output with respect to input , :

We decompose the generalized entry using the quotient rule, where for a function that is equal to the ratio of two other functions: , the derivative of the function is

Here, and and we differentiate each with respect to :

Regardless of the specific input that we are computing the partial derivative for () with respect to, the value will always be :

For , however, the value of the partial derivative is 0 unless , in which case it is :

Thus, when ,

refactoring the numerator

splitting the numerator and denominator

distributing the denominator in the right-hand product

and dividing through each fraction (replacing the long-form notation with the shorthand for softmax)

Similarly, when ,

Combining these conditions, we have

We can use the Kronecker delta function to condense the conditional equation to be:

where

The function set , which we simplify to for notational clarity, maps the matrix () to the dimensions of () and so, like , its Jacobian has rows and columns. We combine the Jacobians of and to define the Jacobian of :

For a general entry, the partial derivative is

Recalling that is 0 except for when (in which case it is ), we can simplify this equation to

And recalling that (or here ), we can write this equation as

Finally, moving to the cross-entropy function, , which maps (dimension ) to the cross-entropy (loss) for the document, which is a scalar value. Thus, the Jacobian of is of dimension .

A general entry in the Jacobian (*i.e.*, the partial derivative of the cross entropy loss with respect to topic ’s probability) is

Notably, the only instance where appears in the function being derived is when , in which case, the derivative is

Otherwise (*i.e.*, when ), the function being derived is a constant and therefore has a derivative = 0.

Combining these conditions, we have

For notation, we identify the element as .

We can verify the dimensionalities of the Jacobians are proper for combination via dot products:

is of dimensions , is of dimensions , and is of dimensions . Thus, each of the two dot products has proper matrices being multiplied. In addition, the resulting matrix is of dimensions , which heuristically matches the fact that the composite of the three functions maps the set of parameters ( in total) to a single scalar value of the cross-entropy loss.

Having verified the dimensionsNext, we combine the elements across the three Jacobians to determine the derivative of (of of) with respect to : . To reduce clutter, we simplify the notation of to and the corresponding Jacobian of interest is then . At this condensed level, maps the parameters (each entry in ) to a scalar output (cross-entropy loss), so the resulting Jacobian is of dimensions :

Similar to the rows in , the single row of partial derivatives in corresponds to the column-major order linearized . Following the indexing of , we will index with and (as ), where refers to column (element) in . For a general entry , then

Since only the element of is non-0, in which case it is , we can simplify this equation to

Substituting in the derivative of the softmax (of the matrix multiplication) of the parameters,

Noting that, by our definition, , we can further simplify this equation. First, we substitute in :

then we cancel out the ’s:

and then rearrange the equation to

Remembering that , we can expand the partial derivative as a conditional statement for explanatory purposes:

This row matrix holds the partial derivatives associated with the parameters and the topics. With these functions, we can calculate the full set of partial derivatives required to evaluate the “gradient” for the loss equation without the penalty.

Following the sum rule, given that we need to calculate the partial derivative of the loss with respect to each of the model parameters , we start with

and take the derivative with respect to a particular parameter in the set,

Therefore, we need to calculate the partial derivative of the penalty function with respect to a particular parameter . Remembering that the penalty function is,

the partial derivative with respect to is

which is

Given that, and that

is now fully defined

This equation determines the gradient of the loss across with respect to each input (parameter-topic combination ) within a single document. Acknowledging multiple documents fall under the same parameter combination, we simply sum across all :