topicmodels: An R package for fitting topic models

Grün, Bettina; Hornik, Kurt

Published in:
Journal of Statistical Software

Published: 01/06/2011

Document Version
Publisher's PDF, also known as Version of record

Journal of Statistical Software

May 2011, Volume 40, Issue 13. *http://www.jstatsoft.org/*

topicmodels: An R Package for Fitting Topic Models

Bettina Grün
Johannes Kepler Universität Linz

Kurt Hornik
WU Wirtschaftsuniversität Wien

Abstract

Topic models allow the probabilistic modeling of term frequency occurrences in documents. The fitted model can be used to estimate the similarity between documents as well as between a set of specified keywords using an additional layer of latent variables which are referred to as topics. The R package **topicmodels** provides basic infrastructure for fitting topic models based on data structures from the text mining package **tm**. The package includes interfaces to two algorithms for fitting topic models: the variational expectation-maximization algorithm provided by David M. Blei and co-authors and an algorithm using Gibbs sampling by Xuan-Hieu Phan and co-authors.

Keywords: Gibbs sampling, R, text analysis, topic model, variational EM.

1. Introduction

In machine learning and natural language processing topic models are generative models which provide a probabilistic framework for the term frequency occurrences in documents in a given corpus. Using only the term frequencies assumes that the information in which order the words occur in a document is negligible. This assumption is also referred to as the *exchangeability* assumption for the words in a document and this assumption leads to bag-of-words models.

Topic models extend and build on classical methods in natural language processing such as the unigram model and the mixture of unigram models (Nigam, McCallum, Thrun, and Mitchell 2000) as well as Latent Semantic Analysis (LSA; Deerwester, Dumais, Furnas, Landauer, and Harshman 1990). Topic models differ from the unigram or the mixture of unigram models because they are mixed-membership models (see for example Airoldi, Blei, Fienberg, and Xing 2008). In the unigram model each word is assumed to be drawn from the same term distribution, in the mixture of unigram models a topic is drawn for each document and all words in a document are drawn from the term distribution of the topic. In mixed-membership

models documents are not assumed to belong to single topics, but to simultaneously belong to several topics and the topic distributions vary over documents.

An early topic model was proposed by Hofmann (1999) who developed probabilistic LSA. He assumed that the interdependence between words in a document can be explained by the latent topics the document belongs to. Conditional on the topic assignments of the words the word occurrences in a document are independent. The latent Dirichlet allocation (LDA; Blei, Ng, and Jordan 2003b) model is a Bayesian mixture model for discrete data where topics are assumed to be uncorrelated. The correlated topics model (CTM; Blei and Lafferty 2007) is an extension of the LDA model where correlations between topics are allowed. An introduction to topic models is given in Steyvers and Griffiths (2007) and Blei and Lafferty (2009). Topic models have previously been used for a variety of applications, including ad-hoc information retrieval (Wei and Croft 2006), geographical information retrieval (Li, Wang, Xie, Wang, and Ma 2008) and the analysis of the development of ideas over time in the field of computational linguistics (Hall, Jurafsky, and Manning 2008).

C code for fitting the LDA model (`http://www.cs.princeton.edu/~blei/lda-c/`) and the CTM (`http://www.cs.princeton.edu/~blei/ctm-c/`) is available under the GPL from David M. Blei and co-authors, who introduced these models in their papers. The method used for fitting the models is the variational expectation-maximization (VEM) algorithm. Other implementations for fitting topic models—especially of the LDA model—are available. The standalone program lda (Mochihashi 2004a,b) provides standard VEM estimation. An implementation in Python of an online version of LDA using VEM estimation as described in Hoffman, Blei, and Bach (2010) is available under the GPL from the first author's web page (`http://www.cs.princeton.edu/~mdhoffma/`). For Bayesian estimation using Gibbs sampling several implementations are available. **GibbsLDA++** (Phan, Nguyen, and Horiguchi 2008) is available under the GPL from `http://gibbslda.sourceforge.net/`. The **MATLAB Topic Modeling** toolbox (Griffiths and Steyvers 2004; Steyvers and Griffiths 2011) is free for scientific use. A license must be obtained from the authors to use it for commercial purposes. **MALLET** (McCallum 2002) is released under the CPL and is a Java-based package which is more general in allowing for statistical natural language processing, document classification, clustering, topic modeling using LDA, information extraction, and other machine learning applications to text. A general toolkit for implementing hierarchical Bayesian models is provided by the Hierarchical Bayes compiler **HBC** (Daumé III 2008), which also allows to fit the LDA model. Another general framework for running Bayesian inference in graphical models which allows to fit the LDA model is available through **Infer.NET** (Microsoft Corporation 2010). The fast collapsed Gibbs sampling method is described in Porteous, Asuncion, Newman, Ihler, Smyth, and Welling (2008) and code is also available from the first author's web page (`http://www.ics.uci.edu/~iporteou/fastlda/`).

For R, an environment for statistical computing and graphics (R Development Core Team 2011), the Comprehensive R Archive Network (`http://CRAN.R-project.org/`) features two packages for fitting topic models: **topicmodels** and **lda**. The R package **lda** (Chang 2010) provides collapsed Gibbs sampling methods for LDA and related topic model variants, with the Gibbs sampler implemented in C. All models in package **lda** are fitted using Gibbs sampling for determining the posterior probability of the latent variables. Wrappers for the expectation-maximization (EM) algorithm are provided which build on this functionality for the E-step. Note that this implementation therefore differs in general from the estimation technique proposed in the original papers introducing these model variants, where the VEM algorithm is

usually applied.

The R package **topicmodels** currently provides an interface to the code for fitting an LDA model and a CTM with the VEM algorithm as implemented by Blei and co-authors and to the code for fitting an LDA topic model with Gibbs sampling written by Phan and co-authors. Package **topicmodels** builds on package **tm** (Feinerer, Hornik, and Meyer 2008; Feinerer 2011) which constitutes a framework for text mining applications within R. **tm** provides infrastructure for constructing a corpus, e.g., by reading in text data from PDF files, and transforming a corpus to a document-term matrix which is the input data for topic models. In package **topicmodels** the respective code is directly called through an interface at the C level avoiding file input and output, and hence substantially improving performance. The functionality for data input and output in the original code was substituted and R objects are directly used as input and S4 objects as output to R. The same main function allows fitting the LDA model with different estimation methods returning objects only slightly different in structure. In addition the strategies for model selection and inference are applicable in both cases. This allows for easy use and comparison of both current state-of-the-art estimation techniques for topic models. Packages **topicmodels** aims at extensibility by providing an interface for inclusion of other estimation methods of topic models.

This paper is structured as follows: Section 2 introduces the specification of topic models, outlines the estimation with the VEM as well as Gibbs sampling and gives an overview of pre-processing steps and methods for model selection and inference. The main fitter functions in the package and the helper functions for analyzing a fitted model are presented in Section 3. An illustrative example for using the package is given in Section 4 where topic models are fitted to the corpus of abstracts in the *Journal of Statistical Software*. A further example is presented in Section 5 using a subset of the Associated Press data set, a larger subset of which was also analyzed in Blei *et al.* (2003b). This data set consists of documents which focus on different content areas and while still being rather small has similar characteristics as other corpora used in the topic models literature. Finally, extending the package to new estimation methods is described in Section 6 using package **rjags** (Plummer 2011).

2. Topic model specification and estimation

2.1. Model specification

For both models—LDA and CTM—the number of topics k has to be fixed a-priori. The LDA model and the CTM assume the following generative process for a document $w = (w_1, \ldots, w_N)$ of a corpus D containing N words from a vocabulary consisting of V different terms, $w_i \in \{1, \ldots, V\}$ for all $i = 1, \ldots, N$.

For LDA the generative model consists of the following three steps.

Step 1: The term distribution β is determined for each topic by

$$\beta \sim \text{Dirichlet}(\delta).$$

Step 2: The proportions θ of the topic distribution for the document w are determined by

$$\theta \sim \text{Dirichlet}(\alpha).$$

Step 3: For each of the N words w_i

(a) Choose a topic $z_i \sim \text{Multinomial}(\theta)$.

(b) Choose a word w_i from a multinomial probability distribution conditioned on the topic z_i: $p(w_i|z_i, \beta)$.

β is the term distribution of topics and contains the probability of a word occurring in a given topic.

For CTM Step 2 is modified to

Step 2a: The proportions θ of the topic distribution for the document w are determined by drawing

$$\eta \sim N(\mu, \Sigma)$$

with $\eta \in \mathbb{R}^{(k-1)}$ and $\Sigma \in \mathbb{R}^{(k-1)\times(k-1)}$.

Set $\tilde{\eta}^\top = (\eta^\top, 0)$. θ is given by

$$\theta_K = \frac{\exp\{\tilde{\eta}_K\}}{\sum_{i=1}^{k} \exp\{\tilde{\eta}_i\}}$$

for $K = 1, \ldots, k$.

2.2. Estimation

For maximum likelihood (ML) estimation of the LDA model the log-likelihood of the data, i.e., the sum over the log-likelihoods of all documents, is maximized with respect to the model parameters α and β. In this setting β and not δ is in general the parameter of interest. For the CTM model the log-likelihood of the data is maximized with respect to the model parameters μ, Σ and β. For VEM estimation the log-likelihood for one document $w \in D$ is for LDA given by

$$\begin{aligned}\ell(\alpha, \beta) &= \log\left(p(w|\alpha, \beta)\right) \\ &= \log \int \left\{ \sum_z \left[\prod_{i=1}^{N} p(w_i|z_i, \beta) p(z_i|\theta) \right] \right\} p(\theta|\alpha) d\theta\end{aligned}$$

and for CTM by

$$\begin{aligned}\ell(\mu, \Sigma, \beta) &= \log\left(p(w|\mu, \Sigma, \beta)\right) \\ &= \log \int \left\{ \sum_z \left[\prod_{i=1}^{N} p(w_i|z_i, \beta) p(z_i|\theta) \right] \right\} p(\theta|\mu, \Sigma) d\theta.\end{aligned}$$

The sum over $z = (z_i)_{i=1,\ldots,N}$ includes all combinations of assigning the N words in the document to the k topics.

The quantities $p(w|\alpha, \beta)$ for the LDA model and $p(w|\mu, \Sigma, \beta)$ for the CTM cannot be tractably computed. Hence, a VEM procedure is used for estimation. The EM algorithm (Dempster,

Laird, and Rubin 1977) is an iterative method for determining an ML estimate in a missing data framework where the complete likelihood of the observed and missing data is easier to maximize than the likelihood of the observed data only. It iterates between an expectation (E)-step where the expected complete likelihood given the data and current parameter estimates is determined and a maximization (M)-step where the expected complete likelihood is maximized to find new parameter estimates. For topic models the missing data in the EM algorithm are the latent variables θ and z for LDA and η and z for CTM.

For topic models a VEM algorithm is used instead of an ordinary EM algorithm because the expected complete likelihood in the E-step is still computationally intractable. For an introduction into variational inference see for example Wainwright and Jordan (2008). To facilitate the E-step the posterior distribution $p(\theta, z|w, \alpha, \beta)$ is replaced by a variational distribution $q(\theta, z|\gamma, \phi)$. This implies that in the E-step instead of

$$\mathsf{E}_p[\log p(\theta, z|w, \alpha, \beta)]$$

the following is determined

$$\mathsf{E}_q[\log p(\theta, z|w, \alpha, \beta)].$$

The parameters for the variational distributions are document specific and hence are allowed to vary over documents which is not the case for α and β. For the LDA model the variational parameters γ and ϕ for a given document w are determined by

$$(\gamma^*, \phi^*) = \arg\min_{(\gamma,\phi)} \mathrm{D}_{\mathrm{KL}}(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \alpha, \beta)).$$

D_{KL} denotes the Kullback-Leibler (KL) divergence. The variational distribution is set equal to

$$q(\theta, z|\gamma, \phi) = q_1(\theta|\gamma) \prod_{i=1}^{N} q_2(z_i|\phi_i),$$

where $q_1()$ is a Dirichlet distribution with parameters γ and $q_2()$ is a multinomial distribution with parameters ϕ_i. Analogously for the CTM the variational parameters are determined by

$$(\lambda^*, \nu^*, \phi^*) = \arg\min_{(\lambda,\nu,\phi)} \mathrm{D}_{\mathrm{KL}}(q(\eta, z|\lambda, \nu^2, \phi)||p(\eta, z|w, \mu, \Sigma, \beta)).$$

Since the variational parameters are fitted separately for each document the variational covariance matrix can be assumed to be diagonal. The variational distribution is set to

$$q(\eta, z|\lambda, \nu^2, \phi) = \prod_{K=1}^{k-1} q_1(\eta_K|\lambda_K, \nu_K^2) \prod_{i=1}^{N} q_2(z_i|\phi_i),$$

where $q_1()$ is a univariate Gaussian distribution with mean λ_K and variance ν_K^2, and $q_2()$ again denotes a multinomial distribution with parameters ϕ_i. Using this simple model for η has the advantage that it is computationally less demanding while still providing enough flexibility. Over all documents this leads to a mixture of normal distributions with diagonal variance-covariance matrices. This mixture distribution allows to approximate the marginal distribution over all documents which has an arbitrary variance-covariance matrix.

For the LDA model it can be shown with the following equality that the variational parameters result in a lower bound for the log-likelihood

$$\log p(w|\alpha, \beta) = L(\gamma, \phi; \alpha, \beta) + \mathrm{D}_{\mathrm{KL}}(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \alpha, \beta))$$

where

$$L(\gamma, \phi; \alpha, \beta) = \mathsf{E}_q[\log p(\theta, z, w|\alpha, \beta)] - \mathsf{E}_q[\log q(\theta, z)]$$

(see Blei *et al.* 2003b, p. 1019). Maximizing the lower bound $L(\gamma, \phi; \alpha, \beta)$ with respect to γ and ϕ is equivalent to minimizing the KL divergence between the variational posterior probability and the true posterior probability. This holds analogously for the CTM.

For estimation the following steps are repeated until convergence of the lower bound of the log-likelihood.

E-step: For each document find the optimal values of the variational parameters $\{\gamma, \phi\}$ for the LDA model and $\{\lambda, \nu, \phi\}$ for the CTM.

M-step: Maximize the resulting lower bound on the log-likelihood with respect to the model parameters α and β for the LDA model and μ, Σ and β for the CTM.

For inference the latent variables θ and z are often of interest to determine which topics a document consists of and which topic a certain word in a document was drawn from. Under the assumption that the variational posterior probability is a good approximation of the true posterior probability it can be used to determine estimates for the latent variables. In the following inference is always based on the variational posterior probabilities if the VEM is used for estimation.

For Gibbs sampling in the LDA model draws from the posterior distribution $p(z|w)$ are obtained by sampling from

$$p(z_i = K|w, z_{-i}) \propto \frac{n^{(j)}_{-i,K} + \delta}{n^{(.)}_{-i,K} + V\delta} \frac{n^{(d_i)}_{-i,K} + \alpha}{n^{(d_i)}_{-i,.} + k\alpha}$$

(see Griffiths and Steyvers 2004; Phan *et al.* 2008). z_{-i} is the vector of current topic memberships of all words without the ith word w_i. The index j indicates that w_i is equal to the jth term in the vocabulary. $n^{(j)}_{-i,K}$ gives how often the jth term of the vocabulary is currently assigned to topic K without the ith word. The dot . implies that summation over this index is performed. d_i indicates the document in the corpus to which word w_i belongs. In the Bayesian model formulation δ and α are the parameters of the prior distributions for the term distribution of the topics β and the topic distribution of documents θ, respectively. The predictive distributions of the parameters θ and β given w and z are given by

$$\hat{\beta}^{(j)}_K = \frac{n^{(j)}_K + \delta}{n^{(.)}_K + V\delta}, \qquad \hat{\theta}^{(d)}_K = \frac{n^{(d)}_K + \alpha}{n^{(.)}_K + k\alpha},$$

for $j = 1, \ldots, V$ and $d = 1, \ldots, D$.

2.3. Pre-processing

The input data for topic models is a document-term matrix. The rows in this matrix correspond to the documents and the columns to the terms. The entry m_{ij} indicates how often the jth term occurred in the ith document. The number of rows is equal to the size of the corpus and the number of columns to the size of the vocabulary. The data pre-processing step involves selecting a suitable vocabulary, which corresponds to the columns of the document-term matrix. Typically, the vocabulary will not be given a-priori, but determined using the available data. The mapping from the document to the term frequency vector involves tokenizing the document and then processing the tokens for example by converting them to lower-case, removing punctuation characters, removing numbers, stemming, removing stop words and omitting terms with a length below a certain minimum. In addition the final document-term matrix can be reduced by selecting only the terms which occur in a minimum number of documents (see Griffiths and Steyvers 2004, who use a value of 5) or those terms with the highest term-frequency inverse document frequency (tf-idf) scores (Blei and Lafferty 2009). The tf-idf scores are only used for selecting the vocabulary, the input data consisting of the document-term matrix uses a term-frequency weighting.

2.4. Model selection

For fitting the LDA model or the CTM to a given document-term matrix the number of topics needs to be fixed a-priori. Additionally, estimation using Gibbs sampling requires specification of values for the parameters of the prior distributions. Griffiths and Steyvers (2004) suggest a value of $50/k$ for α and 0.1 for δ. Because the number of topics is in general not known, models with several different numbers of topics are fitted and the optimal number is determined in a data-driven way. Model selection with respect to the number of topics is possible by splitting the data into training and test data sets. The likelihood for the test data is then approximated using the lower bound for VEM estimation. For Gibbs sampling the log-likelihood is given by

$$\log(p(w|z)) = k \log\left(\frac{\Gamma(V\delta)}{\Gamma(\delta)^V}\right) + \sum_{K=1}^{k}\left\{\left[\sum_{j=1}^{V}\log(\Gamma(n_K^{(j)}+\delta))\right] - \log(\Gamma(n_K^{(\cdot)}+V\delta))\right\}.$$

The perplexity is often used to evaluate the models on held-out data and is equivalent to the geometric mean per-word likelihood.

$$\text{Perplexity}(w) = \exp\left\{-\frac{\log(p(w))}{\sum_{d=1}^{D}\sum_{j=1}^{V} n^{(jd)}}\right\}$$

$n^{(jd)}$ denotes how often the jth term occurred in the dth document. If the model is fitted using Gibbs sampling the likelihood is determined for the perplexity using

$$\log(p(w)) = \sum_{d=1}^{D}\sum_{j=1}^{V} n^{(jd)} \log\left[\sum_{K=1}^{k}\theta_K^{(d)}\beta_K^{(j)}\right]$$

(see Newman, Asuncion, Smyth, and Welling 2009). The topic weights $\theta_K^{(d)}$ can either be determined for the new data using Gibbs sampling where the term distributions for topics

are kept fixed or equal weights are used as implied by the prior distribution. If the perplexity is calculated by averaging over several draws the mean is taken over the samples inside the logarithm.

In addition the marginal likelihoods of the models with different numbers of topics can be compared for model selection if Gibbs sampling is used for model estimation. Griffiths and Steyvers (2004) determine the marginal likelihood using the harmonic mean estimator (Newton and Raftery 1994), which is attractive from a computational point of view because it only requires the evaluation of the log-likelihood for the different posterior draws of the parameters. The drawback however is that the estimator might have infinite variance.

Different methods for evaluating fitted topic models on held-out documents are discussed and compared in Wallach, Murray, Salakhutdinov, and Mimno (2009). Another possibility for model selection is to use hierarchical Dirichlet processes as suggested in Teh, Jordan, Beal, and Blei (2006).

3. Application: Main functions `LDA()` and `CTM()`

The main functions in package **topicmodels** for fitting the LDA and CTM models are `LDA()` and `CTM()`, respectively.

```
LDA(x, k, method = "VEM", control = NULL, model = NULL, ...)
CTM(x, k, method = "VEM", control = NULL, model = NULL, ...)
```

These two functions have the same arguments. `x` is a suitable document-term matrix with non-negative integer count entries, typically a `"DocumentTermMatrix"` as obtained from package **tm**. Internally, **topicmodels** uses the simple triplet matrix representation of package **slam** (Hornik, Meyer, and Buchta 2011)—which, similar to the "coordinate list" (COO) sparse matrix format, stores the information about non-zero entries x_{ij} in the form of (i, j, x_{ij}) triplets. `x` can be any object coercible to such simple triplet matrices (with count entries), in particular objects obtained from readers for commonly employed document-term matrix storage formats. For example the reader `read_dtm_Blei_et_al()` available in package **tm** allows to read in data provided in the format used for the code by Blei and co-authors. `k` is an integer (larger than 1) specifying the number of topics. `method` determines the estimation method used and currently can be either `"VEM"` or `"Gibbs"` for `LDA()` and only `"VEM"` for `CTM()`. Users can provide their own fit functions to use a different estimation technique or fit a slightly different model variant and specify them to be called within `LDA()` and `CTM()` via the `method` argument. Argument `model` allows to provide an already fitted topic model which is used to initialize the estimation.

Argument `control` can be either specified as a named list or as a suitable S4 object where the class depends on the chosen method. In general a user will provide named lists and coercion to an S4 object will internally be performed. The following arguments are possible for the control for fitting the LDA model with the VEM algorithm. They are set to their default values.

```
R> control_LDA_VEM <- list(
+    estimate.alpha = TRUE, alpha = 50/k, estimate.beta = TRUE,
+    verbose = 0, prefix = tempfile(), save = 0, keep = 0,
```

```
+     seed = as.integer(Sys.time()), nstart = 1, best = TRUE,
+     var = list(iter.max = 500, tol = 10^-6),
+     em = list(iter.max = 1000, tol = 10^-4),
+     initialize = "random")
```

The arguments are described in detail below.

`estimate.alpha`, `alpha`, `estimate.beta`: By default α is estimated (`estimate.alpha = TRUE`) and the starting value for α is $50/k$ as suggested by Griffiths and Steyvers (2004). If α is not estimated, it is held fixed at the initial value. If the term distributions for the topics are already given by a previously fitted model, only the topic distributions for documents can be estimated using `estimate.beta = FALSE`. This is useful for example if a fitted model is evaluated on hold-out data or for new data.

`verbose`, `prefix`, `save`, `keep`: By default no information is printed during the algorithm (`verbose = 0`). If `verbose` is a positive integer every `verbose` iteration information is printed. `save` equal to 0 indicates that no intermediate results are saved in files with prefix `prefix`. If equal to a positive integer, every `save` iterations intermediate results are saved. If `keep` is a positive integer, the log-likelihood values are stored every `keep` iteration.

`seed`, `nstart`, `best`: For reproducibility a random seed can be set which is used in the external code. `nstart` indicates the number of repeated runs with random initializations. `seed` needs to have the length `nstart`. If `best = TRUE` only the best model over all runs with respect to the log-likelihood is returned.

`var`, `em`: These arguments control how convergence is assessed for the variational inference step and for the EM algorithm steps by setting a maximum number of iterations (`iter.max`) and a tolerance for the relative change in the likelihood (`tol`). If during the EM algorithm the likelihood is not increased in one step, the maximum number of iterations in the variational inference step is doubled.

If the maximum number of iterations is set to `-1` in the variational inference step, there is no bound on the number of iterations and the algorithm continues until the tolerance criterion is met. If the maximum number of iterations is `-1` for the EM algorithm, no M-step is made and only the variational inference is optimized. This is useful if the variational parameters should be determined for new documents. The default values for the convergence checks are chosen similar to those suggested in the code available from Blei's web page as additional material to Blei *et al.* (2003b) and Blei and Lafferty (2007).

`initialize`: This parameter determines how the topics are initialized and can be either equal to `"random"`, `"seeded"` or `"model"`. Random initialization means that each topic is initialized randomly, seeded initialization signifies that each topic is initialized to a distribution smoothed from a randomly chosen document. If `initialize = "model"` a fitted model needs to be provided which is used for initialization, otherwise random initialization is used.

The possible arguments controlling how the LDA model is fitted using Gibbs sampling are given below together with their default values.

```
R> control_LDA_Gibbs <- list(
+    alpha = 50/k, estimate.beta = TRUE, verbose = 0, prefix = tempfile(),
+    save = 0, keep = 0, seed = as.integer(Sys.time()), nstart = 1,
+    best = TRUE, delta = 0.1, iter = 2000, burnin = 0, thin = 2000)
```

`alpha`, `estimate.beta`, `verbose`, `prefix`, `save`, `keep`, `seed` and `nstart` are the same as for estimation with the VEM algorithm. The other parameters are described below in detail.

`delta`: This parameter specifies the parameter of the prior distribution of the term distribution over topics. The default 0.1 is suggested in Griffiths and Steyvers (2004).

`iter`, `burnin`, `thin`: These parameters control how many Gibbs sampling draws are made. The first `burnin` iterations are discarded and then every `thin` iteration is returned for `iter` iterations.

`best`: All draws are returned if `best = FALSE`, otherwise only the draw with the highest posterior likelihood over all runs is returned.

For the CTM model using the VEM algorithm the following arguments can be used to control the estimation.

```
R> control_CTM_VEM <- list(
+    estimate.beta = TRUE, verbose = 0, prefix = tempfile(), save = 0,
+    keep = 0, seed = as.integer(Sys.time()), nstart = 1L, best = TRUE,
+    var = list(iter.max = 500, tol = 10^-6),
+    em = list(iter.max = 1000, tol = 10^-4), initialize = "random",
+    cg = list(iter.max = 500, tol = 10^-5))
```

`estimate.beta`, `verbose`, `prefix`, `save`, `keep`, `seed`, `nstart`, `best`, `var`, `em` and `initialize` are the same as for VEM estimation of the LDA model. If the log-likelihood is decreased in an E-step, the maximum number of iterations in the variational inference step is increased by 10 or, if no maximum number is set, the tolerance for convergence is divided by 10 and the same E-step is continued. The only additional argument is `cg`.

`cg`: This controls how many iterations at most are used (`iter.max`) and how convergence is assessed (`tol`) in the conjugate gradient step in fitting the variational mean and variance per document.

`LDA()` and `CTM()` return S4 objects of a class which inherits from `"TopicModel"` (or a list of objects inheriting from class `"TopicModel"` if `best = FALSE`). Because of certain differences in the fitted objects there are sub-classes with respect to the model fitted (LDA or CTM) and the estimation method used (VEM or Gibbs sampling). The class `"TopicModel"` contains the call, the dimension of the document-term matrix, the number of words in the document-term matrix, the control object, the number of topics and the terms and document names and the number of iterations made. The estimates for the topic distributions for the documents are included which are the estimates of the corresponding variational parameters for the VEM algorithm and the parameters of the predictive distributions for Gibbs sampling. The term distribution of the topics are also contained which are the ML estimates for the VEM algorithm

and the parameters of the predictive distributions for Gibbs sampling. In additional slots the objects contain the assignment of terms to the most likely topic and the log-likelihood which is $\log p(w|\alpha, \beta)$ for LDA with VEM estimation, $\log p(w|z)$ for LDA using Gibbs sampling and $\log p(w|\mu, \Sigma, \beta)$ for CTM with VEM estimation. For VEM estimation the log-likelihood is returned separately for each document. If a positive `keep` control argument was given, the log-likelihood values of every `keep` iteration is contained. The extending class `"LDA"` has an additional slot for α, `"CTM"` additional slots for μ and Σ. `"LDA_Gibbs"` which extends class `"LDA"` has a slot for δ and `"CTM_VEM"` which extends `"CTM"` has an additional slot for ν^2.

Helper functions to analyze the fitted models are available. `logLik()` obtains the log-likelihood of the fitted model and `perplexity()` can be used to determine the perplexity of a fitted model also for new data. `posterior()` allows one to obtain the topic distributions for documents and the term distributions for topics. There is a `newdata` argument which needs to be given a document-term matrix and where the topic distributions for these new documents are determined without fitting the term distributions of topics. Finally, functions `terms()` and `topics()` allow to obtain from a fitted topic model either the `k` most likely terms for topics or topics for documents respectively, or all terms for topics or topics for documents where the probability is above the specified `threshold`.

4. Illustrative example: Abstracts of JSS papers

The application of the package **topicmodels** is demonstrated on the collection of abstracts of the *Journal of Statistical Software* (JSS) up to 2010-08-05. This illustrative application has the advantage that the analysis can be reproduced in an interactive way and each of the commands can easily be tried out. By contrast re-estimating the models of the application in Section 5 using the Associated Press data would take too long for such a purpose. But Section 5 provides a rather genuine medium-to-large scale application.

The JSS data is available as a list matrix in the package **corpus.JSS.papers** which can be installed and loaded by

```
R> install.packages("corpus.JSS.papers",
+    repos = "http://datacube.wu.ac.at/", type = "source")
R> data("JSS_papers", package = "corpus.JSS.papers")
```

Alternatively, one can harvest JSS publication Dublin Core (http://dublincore.org/) metadata (including information on authors, publication date and the abstract) from the JSS web site using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), for which package **OAIHarvester** (Hornik 2011) provides an R client.

```
R> library("OAIHarvester")
R> x <- oaih_list_records("http://www.jstatsoft.org/oai")
R> JSS_papers <- oaih_transform(x[, "metadata"])
R> JSS_papers <- JSS_papers[order(as.Date(unlist(JSS_papers[, "date"]))), ]
R> JSS_papers <- JSS_papers[grep("Abstract:", JSS_papers[, "description"]), ]
R> JSS_papers[, "description"] <- sub(".*\nAbstract:\n", "",
+    unlist(JSS_papers[, "description"]))
```

For reproducibility of results we use only abstracts published up to 2010-08-05 and omit those containing non-ASCII characters in the abstracts.

```
R> JSS_papers <- JSS_papers[JSS_papers[,"date"] < "2010-08-05",]
R> JSS_papers <- JSS_papers[sapply(JSS_papers[, "description"],
+    Encoding) == "unknown",]
```

The final data set contains 348 documents. Before analysis we transform it to a `"Corpus"` using package **tm**. HTML markup in the abstracts for greek letters, subscripting, etc., is removed using package **XML** (Temple Lang 2010).

```
R> library("topicmodels")
R> library("XML")
R> remove_HTML_markup <- function(s) {
+    doc <- htmlTreeParse(s, asText = TRUE, trim = FALSE)
+    xmlValue(xmlRoot(doc))
+  }
R> corpus <- Corpus(VectorSource(sapply(JSS_papers[, "description"],
+    remove_HTML_markup)))
```

The corpus is exported to a document-term matrix using function `DocumentTermMatrix()` from package **tm**. The terms are stemmed and the stop words, punctuation, numbers and terms of length less than 3 are removed using the `control` argument. (We use a `C` locale for reproducibility.)

```
R> Sys.setlocale("LC_COLLATE", "C")

[1] "C"

R> JSS_dtm <- DocumentTermMatrix(corpus,
+    control = list(stemming = TRUE, stopwords = TRUE, minWordLength = 3,
+      removeNumbers = TRUE, removePunctuation = TRUE))
R> dim(JSS_dtm)

[1]  348 3273
```

The mean term frequency-inverse document frequency (tf-idf) over documents containing this term is used to select the vocabulary. This measure allows to omit terms which have low frequency as well as those occurring in many documents. We only include terms which have a tf-idf value of at least 0.1 which is a bit less than the median and ensures that the very frequent terms are omitted.

```
R> summary(col_sums(JSS_dtm))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   2.000   6.931   5.000 550.000

R> term_tfidf <-
+    tapply(JSS_dtm$v/row_sums(JSS_dtm)[JSS_dtm$i], JSS_dtm$j, mean) *
+      log2(nDocs(JSS_dtm)/col_sums(JSS_dtm > 0))
R> summary(term_tfidf)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02276 0.08615 0.11400 0.14570 0.16240 1.20600

R> JSS_dtm <- JSS_dtm[, term_tfidf >= 0.1]
R> JSS_dtm <- JSS_dtm[row_sums(JSS_dtm) > 0,]
R> summary(col_sums(JSS_dtm))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   2.000   3.406   4.000  64.000
```

After this pre-processing we have the following document-term matrix with a reduced vocabulary which we can use to fit topic models.

```
R> dim(JSS_dtm)

[1]  348 2020
```

In the following we fit an LDA model with 30 topics using (1) VEM with α estimated, (2) VEM with α fixed and (3) Gibbs sampling with a burn-in of 1000 iterations and recording every 100th iterations for 1000 iterations. The initial α is set to the default value. By default only the best model with respect to the log-likelihood $\log(p(w|z))$ observed during Gibbs sampling is returned. In addition a CTM is fitted using VEM estimation.

We set the number of topics rather arbitrarily to 30 after investigating the performance with the number of topics varied from 2 to 200 using 10-fold cross-validation. The results indicated that the number of topics has only a small impact on the model fit on the hold-out data. There is only slight indication that the solution with two topics performs best and that the performance deteriorates again if the number of topics is more than 100. For applications a model with only two topics is of little interest because it enables only to group the documents very coarsely. This lack of preference of a model with a reasonable number of topics might be due to the facts that (1) the corpus is rather small containing less than 500 documents and (2) the corpus consists only of text documents on statistical software.

```
R> k <- 30
R> SEED <- 2010
R> jss_TM <- list(
+    VEM = LDA(JSS_dtm, k = k, control = list(seed = SEED)),
+    VEM_fixed = LDA(JSS_dtm, k = k, control = list(estimate.alpha = FALSE,
+      seed = SEED)),
+    Gibbs = LDA(JSS_dtm, k = k, method = "Gibbs", control = list(
+      seed = SEED, burnin = 1000, thin = 100, iter = 1000)),
+    CTM = CTM(JSS_dtm, k = k, control = list(seed = SEED,
+      var = list(tol = 10^-4), em = list(tol = 10^-3))))
```

To compare the fitted models we first investigate the α values of the models fitted with VEM and α estimated and with VEM and α fixed.

```
R> sapply(jss_TM[1:2], slot, "alpha")
```

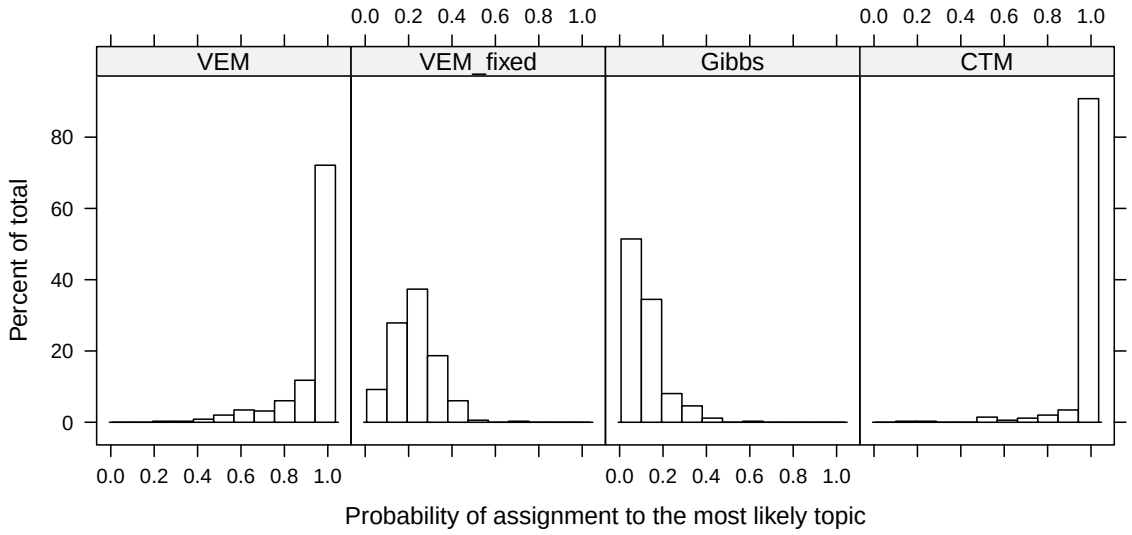

Figure 1: Histogram of the probabilities of assignment to the most likely topic for all documents for the different estimation methods.

```
       VEM   VEM_fixed
0.009669373 1.666666667
```

We see that if α is estimated it is set to a value much smaller than the default. This indicates that in this case the Dirichlet distribution has more mass at the corners and hence, documents consist only of few topics. The influence of α on the estimated topic distributions for documents is illustrated in Figure 1 where the probabilities of the assignment to the most likely topic for all documents are given. The lower α the higher is the percentage of documents which are assigned to one single topic with a high probability. Furthermore, it indicates that the association of documents with only one topic is strongest for the CTM solution.

The entropy measure can also be used to indicate how the topic distributions differ for the four fitting methods. We determine the mean entropy for each fitted model over the documents. The term distribution for each topic as well as the predictive distribution of topics for a document can be obtained with `posterior()`. A list with components `"terms"` for the term distribution over topics and `"topics"` for the topic distributions over documents is returned.

```
R> sapply(jss_TM, function(x) mean(apply(posterior(x)$topics,
+    1, function(z) - sum(z * log(z)))))

      VEM VEM_fixed     Gibbs       CTM
0.2863427 3.0925014 3.2519352 0.1839297
```

Higher values indicate that the topic distributions are more evenly spread over the topics.

The estimated topics for a document and estimated terms for a topic can be obtained using the convenience functions `topics()` and `terms()`. The most likely topic for each document is obtained by

```
R> Topic <- topics(jss_TM[["VEM"]], 1)
```

The five most frequent terms for each topic are obtained by

```
R> Terms <- terms(jss_TM[["VEM"]], 5)
R> Terms[, 1:5]
```

```
     Topic 1       Topic 2  Topic 3      Topic 4     Topic 5
[1,] "constraint"  "robust" "multivari"  "densiti"   "correl"
[2,] "fechnerian"  "genet"  "gene"       "exponenti" "gee"
[3,] "metaanalysi" "intern" "aspect"     "mixtur"    "qls"
[4,] "pattern"     "pilot"  "robust"     "zeroinfl"  "critic"
[5,] "ptak"        "plan"   "microarray" "random"    "hypothes"
```

If any category labelings of the documents were available, these could be used to validate the fitted model. Some JSS papers should have similar content because they appeared in the same special volume. The most likely topic of the papers which appeared in Volume 24 called "Statistical Modeling of Social Networks with **statnet**" (see Handcock, Hunter, Butts, Goodreau, and Morris 2008) is given by

```
R> (topics_v24 <-
+   topics(jss_TM[["VEM"]])[grep("/v24/", JSS_papers[, "identifier"])])
```

```
243 244 245 246 247 248 249 250 251
  7   4   7   7  26   7   7  27   7
```

```
R> most_frequent_v24 <- which.max(tabulate(topics_v24))
```

The similarity between these papers is indicated by the fact that the majority of the papers have the same topic as their most likely topic. The ten most likely terms for topic 7 are given by

```
R> terms(jss_TM[["VEM"]], 10)[, most_frequent_v24]
```

```
 [1] "network"   "ergm"      "popul"     "captur"    "multivari"
 [6] "rcaptur"   "social"    "criterion" "growth"    "ssa"
```

Clearly this topic is related to the general theme of the special issue. This indicates that the fitted topic model was successful at detecting the similarity between papers in the same special issue without using this information.

5. Associated Press data

In the following a subset from the Associated Press data from the First Text Retrieval Conference (TREC-1) 1992 (Harman 1992) is analyzed. The data is available together with the code for fitting LDA models and CTMs on Blei's web page (http://www.cs.princeton.edu/~blei/lda-c) and is also contained in package **topicmodels**.

```
R> data("AssociatedPress", package = "topicmodels")
R> dim(AssociatedPress)
```

```
[1]  2246 10473
```

It consists of 2246 documents and the vocabulary was already selected to contain only words which occur in more than 5 documents.

```
R> range(col_sums(AssociatedPress))
```

```
[1]    6 2073
```

The analysis uses 10-fold cross-validation to evaluate the performance of the models. First, the data set is split into 10 test data sets with the remaining data as training data. Only LDA models are fitted in the following using the different estimation methods. CTM is not fitted because of the high memory demand and longer times required for estimating this model. The complete R code for the simulation as well as the summarization of results is given in the Appendix. Below only the main code parts are presented. First, a random seed is set and indices are randomly drawn to split the data into the 10 folds of training and test data.

```
R> set.seed(0908)
R> folding <- sample(rep(seq_len(10),
+    ceiling(nrow(AssociatedPress)))[seq_len(nrow(AssociatedPress))])
```

With `fold` having values from $1, \ldots, 10$ we estimate the model with the three different variants.

```
R> testing <- which(folding == fold)
R> training <- which(folding != fold)
```

The number of topics are varied from

```
R> topics <- 10 * c(1:5, 10, 20)
```

For VEM with α free, we have:

```
R> train <- LDA(AssociatedPress[training,], k = k,
+    control = list(verbose = 100))
R> test <- LDA(AssociatedPress[testing,], model = train,
+    control = list(estimate.beta = FALSE))
```

For VEM with α fixed:

```
R> train <- LDA(AssociatedPress[training,], k = k,
+    control = list(verbose = 100, estimate.alpha = FALSE))
R> test <- LDA(AssociatedPress[testing,], model = train,
+    control = list(estimate.beta = FALSE))
```

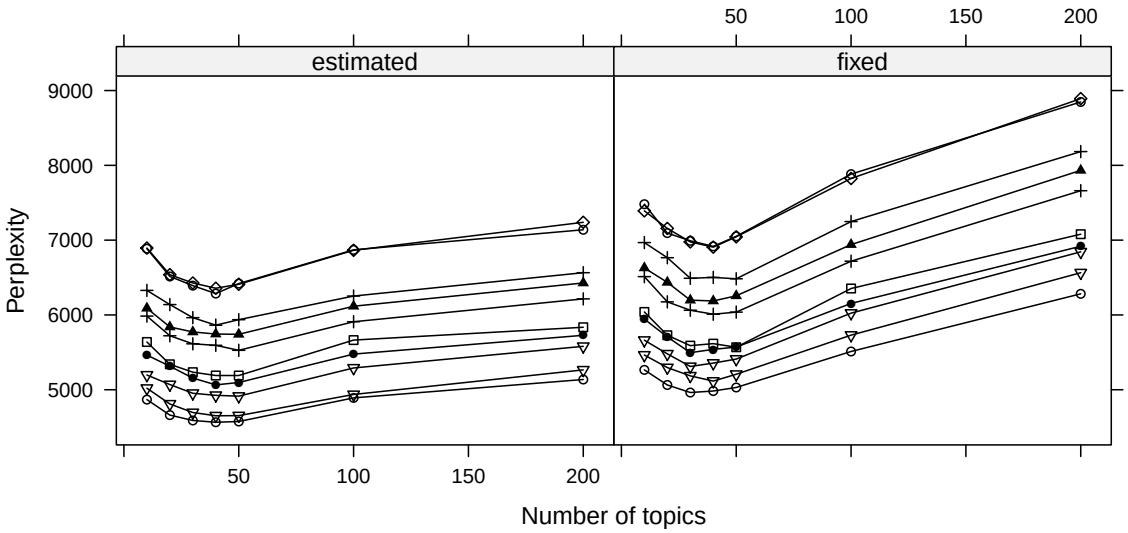

Figure 2: Perplexities of the test data for the models fitted with VEM. Each line corresponds to one of the folds in the 10-fold cross-validation.

For Gibbs sampling we use a burn-in of 1000 followed by 1000 draws with a thinning of 100 and have all draws returned.

```
R> train <- LDA(AssociatedPress[training,], k = k, method = "Gibbs",
+    control = list(burnin = 1000, thin = 100, iter = 1000, best = FALSE))
R> test <- LDA(AssociatedPress[testing,],
+    model = train[[which.max(sapply, train, logLik)]],
+    control = list(estimate.beta = FALSE, burnin = 1000, thin = 100,
+    iter = 1000, best = FALSE))
```

The perplexities of the test data for the models fitted using VEM are given in Figure 2. For both estimation methods about 40 topics are suggested as optimal. The α values estimated by VEM are given in Figure 3 on the left. Obviously these values are much smaller than those used as default with $50/k$. Again, note that small values of α indicate that the topic distribution over documents has most of its weight in the corners. This implies that the documents consist only of a small number of topics. For the model fitted using Gibbs sampling model selection is also performed by determining the perplexities for the test data. Figure 3 on the right suggests that about 20–40 topics are optimal.

We estimate the LDA model with the three different estimation methods with 40 topics to the complete data set. α is initialized as the mean value of the optimal α values in the cross-validation for the models fitted with VEM and α estimated. The R code is given in the Appendix. We compare the topics detected by the VEM algorithm with α estimated and the Gibbs sampling solution after matching the topics. The topics are matched based on the Hellinger distance between the term distributions of the topics using package **clue** (Hornik 2005).

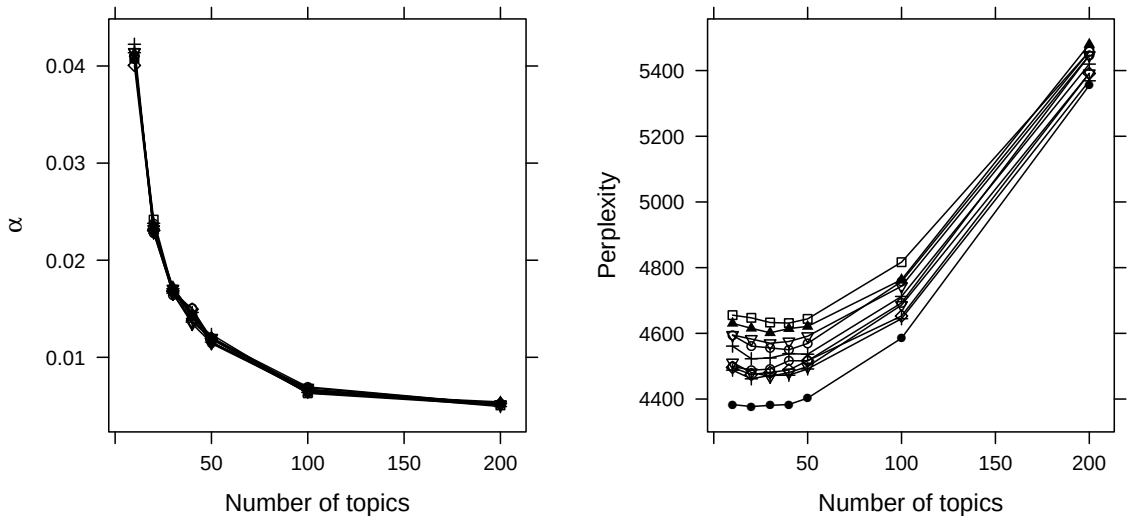

Figure 3: Left: estimated α values for the models fitted using VEM. Right: perplexities of the test data for the models fitted using Gibbs sampling. Each line corresponds to one of the folds in the 10-fold cross-validation.

```
R> dist_models <- distHellinger(posterior(AP[["VEM"]])$terms,
+    posterior(AP[["Gibbs"]])$terms)
R> library("clue")
R> matching <- solve_LSAP(dist_models)
R> dist_models <- dist_models[, matching]
R> d <- mean(diag(dist_models))
```

The best matches with the smallest distance are determined and compared with respect to their eight most likely words.

```
R> best_match <- order(diag(dist_models))
R> terms(AP$VEM, 8)[,best_match[1:4]]
```

```
     Topic 29     Topic 37   Topic 22   Topic 21
[1,] "dukakis"    "company"  "court"    "bill"
[2,] "bush"       "million"  "charges"  "senate"
[3,] "campaign"   "new"      "trial"    "house"
[4,] "democratic" "inc"      "case"     "committee"
[5,] "jackson"    "corp"     "attorney" "rep"
[6,] "i"          "business" "prison"   "legislation"
[7,] "republican" "billion"  "judge"    "sen"
[8,] "president"  "stock"    "drug"     "congress"
```

```
R> terms(AP$Gibbs, 8)[,matching[best_match[1:4]]]
```

```
     Topic 27       Topic 13  Topic 34    Topic 36
[1,] "bush"         "company" "prison"    "bill"
[2,] "dukakis"      "million" "court"     "committee"
[3,] "president"    "inc"     "trial"     "senate"
[4,] "campaign"     "new"     "charges"   "house"
[5,] "jackson"      "stock"   "judge"     "sen"
[6,] "democratic"   "corp"    "attorney"  "rep"
[7,] "presidential" "billion" "convicted" "congress"
[8,] "convention"   "share"   "guilty"    "members"
```

These four topics clearly are on the same subjects and consist of very similar words. However, this clear correspondence between topics is only present for a small subset of the topics. This is also indicated by the image plot of the distances with the matched topics in Figure 4. According to the image we would not expect the worst four matching topics to have much in common. This is also seen by inspecting the eight most important words for each of these topics.

```
R> worst_match <- order(diag(dist_models), decreasing = TRUE)
R> terms(AP$VEM, 8)[, worst_match[1:4]]
```

```
     Topic 14     Topic 13    Topic 26     Topic 27
[1,] "iraq"       "people"    "panama"     "china"
[2,] "government" "estate"    "noriega"    "west"
[3,] "kuwait"     "ireland"   "government" "chinese"
[4,] "iraqi"      "officials" "i"          "east"
[5,] "people"     "three"     "women"      "government"
[6,] "plan"       "ira"       "president"  "years"
[7,] "president"  "army"      "military"   "city"
[8,] "soviet"     "years"     "new"        "german"
```

```
R> terms(AP$Gibbs, 8)[, matching[worst_match[1:4]]]
```

```
     Topic 26  Topic 29    Topic 6      Topic 39
[1,] "cents"   "computer"  "election"   "church"
[2,] "oil"     "company"   "campaign"   "catholic"
[3,] "prices"  "business"  "state"      "pope"
[4,] "futures" "corp"      "republican" "john"
[5,] "cent"    "co"        "vote"       "roman"
[6,] "lower"   "defense"   "percent"    "religious"
[7,] "higher"  "companies" "democratic" "vatican"
[8,] "farmers" "industry"  "candidates" "paul"
```

6. Extending to new fit methods

Package **topicmodels** already provides two different estimation methods for the LDA model and one for the CTM. Users can extend the methods and supply their own fit functions via

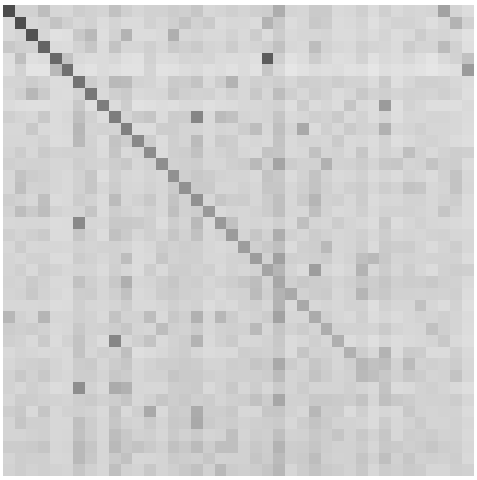

Figure 4: Matched topics of the solution with 40 topics for VEM with free α and with 40 topics for Gibbs sampling.

the `method` argument. In the following we outline how package **rjags** can be used to fit the LDA model using Gibbs sampling with a different implementation. **rjags** is a general purpose Gibbs sampler built on the library **JAGS** (Just another Gibbs sampler; Plummer 2003) and model specification is by a model language not unlike the one for **WinBUGS** (Lunn, Thomas, Best, and Spiegelhalter 2000). The advantage of a general purpose Gibbs sampler is that a wide range of models can be fitted without writing new code for each of the models. This allows to easily try out different models and compare them. However, a drawback is that in general the Gibbs sampler used is less efficient. This is especially a problem in applications where complicated models are fitted and/or large data sets are used. For application of the LDA model the size of the document-term matrix will in general be too large to use a general purpose Gibbs sampler and achieve a reasonable performance. The following code therefore only serves as an illustrative example for extending the package to new methods.

If **rjags** is used for fitting the LDA model, the model needs to be specified in the **JAGS** language. Assuming that the data is given by vectors `D` and `W`. Both vectors have a length equal to the number of words in the document-term matrix. `D` indicates the document and `W` the word from the vocabulary. `n` is the number of documents in the corpus.

```
R> BUGS_MODEL <-
+  "model {
+    for (i in 1:length(W)) {
+      z[i] ~ dcat(theta[D[i],]);
+      W[i] ~ dcat(beta[z[i],]);
+    }
+    for (j in 1:n) {
+      theta[j,1:k] ~ ddirch(alpha);
+    }
+    for (K in 1:k) {
+      beta[K,1:V] ~ ddirch(delta);
+    }
+  }"
```

The following code implements a new method function to fit the LDA model using Gibbs sampling with package **rjags**. In this function for fitting the model the log-likelihood as well as the most probable topic membership of each word in the corpus are not determined and therefore not part of the returned object.

```
R> LDA_rjags.fit <- function(x, k, control = NULL, model = NULL, call, ....)
+  {
+    if (!require("rjags"))
+      stop("\nThis method requires package 'rjags'")
+
+    control <- as(control, "LDA_Gibbscontrol")
+    if (length(control@alpha) == 0)
+      control@alpha <- if (!is.null(model)) model@alpha else 50/k
+
+    DATA <- list(D = rep(x$i, x$v), W = rep(x$j, x$v), n = nrow(x), k = k,
+      V = ncol(x), alpha = rep(control@alpha, k),
+      delta = rep(control@delta, ncol(x)))
+
+    FILE <- file()
+    cat(BUGS_MODEL, file = FILE)
+    model <- jags.model(FILE, data = DATA,
+      inits = list(.RNG.name = "base::Wichmann-Hill",
+      .RNG.seed = control@seed))
+    close(FILE)
+
+    if (control@burnin > 0) update(model, iter = control@burnin)
+    SAMPLES <- coda.samples(model, c("theta", "beta", "z"),
+      thin = control@thin, n.iter = control@iter)[[1]]
+    index_beta <- seq_len(k * ncol(x))
+    index_gamma <- k * ncol(x) + seq_len(nrow(x) * k)
+    obj <- lapply(seq_len(nrow(SAMPLES)), function(i)
+      new("LDA_Gibbs",
+        call = call, Dim = dim(x), k = as.integer(k), control = control,
+        alpha = control@alpha, delta = control@delta,
+        terms = colnames(x), documents = rownames(x),
+        beta = matrix(SAMPLES[i, index_beta], nrow = k),
+        gamma = matrix(SAMPLES[i, index_gamma], ncol = k)))
+    if (nrow(SAMPLES) == 1) obj <- obj[[1]]
+    obj
+  }
```

We apply the new fit function only to a small subset of the Associated Press data and perform only 20 iterations of the Gibbs sampler in order to limit the time needed. The time needed is also compared to the LDA specific implementation of the Gibbs sampler.

```
R> AP_small <- AssociatedPress
R> AP_small <- AP_small[row_sums(AP_small) > 500,]
```

```
R> AP_small <- AP_small[,col_sums(AP_small) > 10,]
R> dim(AP_small)

[1]  18 162

R> system.time({
+    lda <- LDA(AP_small, k = 5, method = "Gibbs",
+      control = list(burnin = 0, iter = 20, seed = 2010))
+  })

   user  system elapsed
  0.036   0.000   0.036

R> system.time({
+    lda_rjags <- LDA(AP_small, k = 5, method = LDA_rjags.fit,
+      control = list(burnin = 0, iter = 20, seed = 2010))
+  })

module basemod loaded
module bugs loaded
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 11839

   user  system elapsed
  5.664   0.048   5.981

R> terms(lda_rjags, 4)

     Topic 1     Topic 2   Topic 3      Topic 4      Topic 5
[1,] "gorbachev" "defense" "department" "department" "united"
[2,] "people"    "soviet"  "leaders"    "people"     "defense"
[3,] "i"         "i"       "new"        "bush"       "new"
[4,] "held"      "weapons" "president"  "military"   "bush"
```

As can be seen from this example, adding a new estimation method in package **topicmodels** requires writing a suitable fit function. The fit function takes the document-term matrix, fits the model and returns an object of class `"TopicModel"` or an extended class thereof. The accessor functions `terms()` and `topics()` can then be used to extract the fitted term distributions for the topics as well as the fitted topic distributions for the documents.

7. Summary

Package **topicmodels** provides functionality for fitting the topic models LDA and CTM in R. It builds on and complements functionality for text mining already provided by package

tm. Functionality for constructing a corpus, transforming a corpus into a document-term matrix and selecting the vocabulary is available in **tm**. The basic text mining infrastructure provided by package **tm** is hence extended to allow also fitting of topic models which are seen nowadays as state-of-the-art techniques for analyzing document-term matrices. The advantages of package **topicmodels** are that (1) it gives access within R to the code written by David M. Blei and co-authors, who introduced the LDA model as well as the CTM in their papers, and (2) allows different estimation methods by providing VEM estimation as well Gibbs sampling. Extensibility to other estimation techniques or slightly different model variants is easily possible via the `method` argument.

Packages **Snowball** (Hornik 2009) and **tm** provide stemmers and stop word lists not only for English, but also for other languages. To the authors' knowledge topic models have so far only been used for corpora in English. The availability of all these tools in R hopefully does not only lead to an increased use of these models, but also facilitates to try them out for corpora in other languages as well as in different settings. In addition different modeling strategies for model selection, such as cross-validation, can be easily implemented with a few lines of R code and the results can be analyzed and visualized using already available tools in R.

Due to memory requirements package **topicmodels** will for standard hardware only work for reasonably large corpora with numbers of topics in the hundreds. Gibbs sampling needs less memory than using the VEM algorithm and might therefore be able to fit models when the VEM algorithm fails due to high memory demands. In order to be able to fit topic models to very large data sets distributed algorithms to fit the LDA model were proposed for Gibbs sampling in Newman *et al.* (2009). The proposed Approximate Distributed LDA (AD-LDA) algorithm requires the Gibbs sampling methods available in **topicmodels** to be performed on each of the processors. In addition functionality is needed to repeatedly distribute the data and parameters to the single processors and synchronize the results from the different processors until a termination criterion is met. Algorithms to parallelize the VEM algorithm for fitting LDA models are outlined in Nallapati, Cohen, and Lafferty (2007). In this case the processors are used in the E-step such that each processor calculates only the sufficient statistics for a subset of the data. We intend to look into the potential of leveraging the existing infrastructure for large data sets along the lines proposed in Nallapati *et al.* (2007) and Newman *et al.* (2009).

The package allows us to fit topic models to different corpora which are already available in R using package **tm** or can easily be constructed using tools such as the package **OAIHarvester**. We are also interested in comparing the performance of topic models for clustering documents to other approaches such as using mixtures of von Mises-Fisher distributions to model the term distributions of the documents (Banerjee, Dhillon, Ghosh, and Sra 2005) where the R package **movMF** (Hornik and Grün 2011) is available on CRAN.

Different variants of topic models have been recently proposed. Some models aim at relaxing the assumption of independence of topics which is imposed by LDA such as the CTM, hierarchical topic models (Blei, Griffiths, Jordan, and Tenenbaum 2003a) or Pachinko allocation (Li and McCallum 2006) and hierarchical Pachinko allocation (Mimno, Li, and McCallum 2007). Another possible extension of the LDA model is to include additional information. Using the time information leads to dynamic topic models (Blei and Lafferty 2006) while using the author information of the documents gives the author-topic model (Rosen-Zvi, Chemudugunta, Griffiths, Smyth, and Steyvers 2010). We are interested in extending the package to cover at least a considerable subset of the different proposed topic models. As a starting point we will

use Heinrich (2009) and Heinrich and Goesele (2009) who provide a common framework for topic models which only consist of Dirichlet-multinomial mixture "levels". Examples for such topic models are LDA, the author-topic model, Pachinko allocation and hierarchical Pachinko allocation.

Acknowledgments

We would like to thank two anonymous reviewers for their valuable comments which led to several improvements. This research was supported by the the Austrian Science Fund (FWF) under Hertha-Firnberg grant T351-N18 and under Elise-Richter grant V170-N18.

References

Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008). "Mixed Membership Stochastic Blockmodels." *Journal of Machine Learning Research*, **9**, 1981–2014.

Banerjee A, Dhillon IS, Ghosh J, Sra S (2005). "Clustering on the Unit Hypersphere Using von Mises-Fisher Distributions." *Journal of Machine Learning Research*, **6**, 1345–1382.

Blei DM, Griffiths TL, Jordan MI, Tenenbaum JB (2003a). "Hierarchical Topic Models and the Nested Chinese Restaurant Process." In S Thrun, LK Saul, B Schölkopf (eds.), *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA.

Blei DM, Lafferty JD (2006). "Dynamic Topic Models." In *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*, pp. 113–120. ACM Press.

Blei DM, Lafferty JD (2007). "A Correlated Topic Model of Science." *The Annals of Applied Statistics*, **1**(1), 17–35.

Blei DM, Lafferty JD (2009). "Topic Models." In A Srivastava, M Sahami (eds.), *Text Mining: Classification, Clustering, and Applications*. Chapman & Hall/CRC Press.

Blei DM, Ng AY, Jordan MI (2003b). "Latent Dirichlet Allocation." *Journal of Machine Learning Research*, **3**, 993–1022.

Chang J (2010). ***lda**: Collapsed Gibbs Sampling Methods for Topic Models*. R package version 1.2.3, URL http://CRAN.R-project.org/package=lda.

Daumé III H (2008). ***HBC**: Hierarchical Bayes Compiler*. Pre-release version 0.7, URL http://www.cs.utah.edu/~hal/HBC/.

Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R (1990). "Indexing by Latent Semantic Analysis." *Journal of the American Society for Information Science*, **41**(6), 391–407.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood from Incomplete Data Via the EM-Algorithm." *Journal of the Royal Statistical Society B*, **39**, 1–38.

Feinerer I (2011). ***tm**: Text Mining Package*. R package version 0.5-5., URL http://CRAN.R-project.org/package=tm.

Feinerer I, Hornik K, Meyer D (2008). "Text Mining Infrastructure in R." *Journal of Statistical Software*, **25**(5), 1–54. URL http://www.jstatsoft.org/v25/i05/.

Griffiths TL, Steyvers M (2004). "Finding Scientific Topics." *Proceedings of the National Academy of Sciences of the United States of America*, **101**, 5228–5235.

Hall D, Jurafsky D, Manning CD (2008). "Studying the History of Ideas Using Topic Models." In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A Meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 363–371. ACL.

Handcock MS, Hunter DR, Butts CT, Goodreau SM, Morris M (2008). "**statnet**: Software Tools for the Representation, Visualization, Analysis and Simulation of Network Data." *Journal of Statistical Software*, **24**(1), 1–11. URL http://www.jstatsoft.org/v24/i01/.

Harman D (1992). "Overview of the First Text Retrieval Conference (TREC-1)." In D Harman (ed.), *Proceedings of the First Text Retrieval Conference (TREC-1)*, NIST Special Publication 500-207, pp. 1–20. National Institute of Standards and Technology.

Heinrich G (2009). "A Generic Approach to Topic Models." In WL Buntine, M Grobelnik, D Mladenic, J Shawe-Taylor (eds.), *Machine Learning and Knowledge Discovery in Databases*, volume 5781 of *Lecture Notes in Computer Science*, pp. 517–532. Springer-Verlag, Berlin.

Heinrich G, Goesele M (2009). "Variational Bayes for Generic Topic Models." In B Mertsching, M Hund, Z Aziz (eds.), *KI 2009: Advances in Artificial Intelligence*, volume 5803 of *Lecture Notes in Computer Science*, pp. 161–168. Springer-Verlag, Berlin.

Hoffman MD, Blei DM, Bach F (2010). "Online Learning for Latent Dirichlet Allocation." In J Lafferty, CKI Williams, J Shawe-Taylor, R Zemel, A Culotta (eds.), *Advances in Neural Information Processing Systems 23*, pp. 856–864. MIT Press, Cambridge, MA.

Hofmann T (1999). "Probabilistic Latent Semantic Indexing." In *SIGIR'99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 50–57. ACM Press.

Hornik K (2005). "A CLUE for CLUster Ensembles." *Journal of Statistical Software*, **14**(12), 1–25. URL http://www.jstatsoft.org/v14/i12/.

Hornik K (2009). ***Snowball**: Snowball Stemmers*. R package version 0.0-7, URL http://CRAN.R-project.org/package=Snowball.

Hornik K (2011). ***OAIHarvester**: Harvest Metadata Using OAI-PMH v2.0*. R package version 0.1-3, URL http://CRAN.R-project.org/package=OAIHarvester.

Hornik K, Grün B (2011). ***movMF**: Mixtures of von Mises Fisher Distributions*. R package version 0.0-0, URL http://CRAN.R-project.org/package=movMF.

Hornik K, Meyer D, Buchta C (2011). ***slam**: Sparse Lightweight Arrays and Matrices*. R package version 0.1-21, URL http://CRAN.R-project.org/package=slam.

Li W, McCallum A (2006). "Pachinko Allocation: DAG-Structured Mixture Models of Topic Correlations." In *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*, pp. 577–584. ACM Press, New York.

Li Z, Wang C, Xie X, Wang X, Ma WY (2008). "Exploring LDA-Based Document Model for Geographic Information Retrieval." In C Peters, V Jijkoun, T Mandl, H Müller, D Oard, AP nas, V Petras, D Santos (eds.), *Advances in Multilingual and Multimodal Information Retrieval*, volume 5152 of *Lecture Notes in Computer Science*, pp. 842–849. Springer-Verlag, Berlin.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "WinBUGS—A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**(4), 325–337.

McCallum AK (2002). ***MALLET**: Machine Learning for Language Toolkit*. URL http://mallet.cs.umass.edu/.

Microsoft Corporation (2010). ***Infer.NET** User Guide*. Version 2.4 beta 2, URL http://research.microsoft.com/en-us/um/cambridge/projects/infernet/.

Mimno D, Li W, McCallum A (2007). "Mixtures of Hierarchical Topics with Pachinko Allocation." In *ICML'07: Proceedings of the 21st International Conference on Machine Learning*, pp. 633–640. ACM Press.

Mochihashi D (2004a). "A Note on a Variational Bayes Derivation of Full Bayesian Latent Dirichlet Allocation." Unpublished manuscript, URL http://chasen.org/~daiti-m/paper/lda-fullvb.pdf.

Mochihashi D (2004b). ***lda**, a Latent Dirichlet Allocation Package*. MATLAB and C package version 0.1, URL http://chasen.org/~daiti-m/dist/lda/.

Nallapati R, Cohen W, Lafferty J (2007). "Parallelized Variational EM for Latent Dirichlet Allocation: An Experimental Evaluation of Speed and Scalability." In *ICDMW'07: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, pp. 349–354. IEEE Computer Society, Washington, DC.

Newman D, Asuncion A, Smyth P, Welling M (2009). "Distributed Algorithms for Topic Models." *Journal of Machine Learning Research*, **10**, 1801–1828.

Newton MA, Raftery AE (1994). "Approximate Bayesian Inference with the Weighted Likelihood Bootstrap." *Journal of the Royal Statistical Society B*, **56**(1), 3–48.

Nigam K, McCallum AK, Thrun S, Mitchell T (2000). "Text Classification from Labeled and Unlabeled Documents Using EM." *Machine Learning*, **39**(2–3), 103–134.

Phan XH, Nguyen LM, Horiguchi S (2008). "Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-Scale Data Collections." In *Proceedings of the 17th International World Wide Web Conference (WWW 2008)*, pp. 91–100. Beijing, China.

Plummer M (2003). "**JAGS**: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In K Hornik, F Leisch, A Zeileis (eds.), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Vienna, Austria*. ISSN 1609-395X, URL http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/.

Plummer M (2011). ***rjags**: Bayesian Graphical Models Using MCMC*. R package version 2.2.0-3, URL http://CRAN.R-project.org/package=rjags.

Porteous I, Asuncion A, Newman D, Ihler A, Smyth P, Welling M (2008). "Fast Collapsed Gibbs Sampling for Latent Dirichlet Allocation." In *KDD'08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 569–577. ACM Press.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Rosen-Zvi M, Chemudugunta C, Griffiths T, Smyth P, Steyvers M (2010). "Learning Author-Topic Models from Text Corpora." *ACM Transactions on Information Systems*, **28**(1).

Steyvers M, Griffiths T (2007). "Probabilistic Topic Models." In TK Landauer, DS McNamara, S Dennis, W Kintsch (eds.), *Handbook of Latent Semantic Analysis*. Lawrence Erlbaum Associates.

Steyvers M, Griffiths T (2011). *MATLAB **Topic Modeling** Toolbox 1.4*. URL http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm.

Teh YW, Jordan MI, Beal MJ, Blei DM (2006). "Hierarchical Dirichlet Processes." *Journal of the American Statistical Association*, **101**(476), 1566–1581.

Temple Lang D (2010). ***XML**: Tools for Parsing and Generating XML Within R and S-PLUS*. R package version 3.2-0, URL http://CRAN.R-project.org/package=XML.

Wainwright MJ, Jordan MI (2008). "Graphical Models, Exponential Families, and Variational Inference." *Foundations and Trends in Machine Learning*, **1**(1–2), 1–305.

Wallach HM, Murray I, Salakhutdinov R, Mimno D (2009). "Evaluation Methods for Topic Models." In *ICML'09: Proceedings of the 26th International Conference on Machine Learning*, pp. 1105–1112. ACM Press.

Wei X, Croft WB (2006). "LDA-Based Document Models for Ad-Hoc Retrieval." In *SIGIR'06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 178–185. ACM Press, New York.

A. R code for the Associated Press analysis

In the following the R code is provided for the analysis of the Associated Press data set. Due to the long duration of the simulation as well as the fitting of the LDA model to the complete data set the results are saved and loaded for the analysis in the paper.

A.1. Simulation using 10-fold cross-validation

```
set.seed(0908)
topics <- 10 * c(1:5, 10, 20)
SEED <- 20080809
library("topicmodels")
data("AssociatedPress", package = "topicmodels")
D <- nrow(AssociatedPress)
folding <-
  sample(rep(seq_len(10), ceiling(D))[seq_len(D)])
for (k in topics) {
  for (chain in seq_len(10)) {
    FILE <- paste("VEM_", k, "_", chain, ".rda", sep = "")
    training <- LDA(AssociatedPress[folding != chain,], k = k,
      control = list(seed = SEED))
    testing <- LDA(AssociatedPress[folding == chain,], model = training,
      control = list(estimate.beta = FALSE, seed = SEED))
    save(training, testing, file = file.path("results", FILE))
    FILE <- paste("VEM_fixed_", k, "_", chain, ".rda", sep = "")
    training <- LDA(AssociatedPress[folding != chain,], k = k,
      control = list(seed = SEED, estimate.alpha = FALSE))
    testing <- LDA(AssociatedPress[folding == chain,], model = training,
      control = list(estimate.beta = FALSE, seed = SEED))
    save(training, testing, file = file.path("results", FILE))
    FILE <- paste("Gibbs_", k, "_", chain, ".rda", sep = "")
    training <- LDA(AssociatedPress[folding != chain,], k = k,
      control = list(seed = SEED, burnin = 1000, thin = 100,
      iter = 1000, best = FALSE), method = "Gibbs")
    best_training <- training@fitted[[which.max(logLik(training))]]
    testing <- LDA(AssociatedPress[folding == chain,],
      model = best_training, control = list(estimate.beta = FALSE,
        seed = SEED, burnin = 1000, thin = 100, iter = 1000, best = FALSE))
    save(training, testing, file = file.path("results", FILE))
  }
}
```

A.2. Summarizing the cross-validation simulation results

```
set.seed(0908)
```

```
topics <- 10 * c(1:5, 10, 20)
library("topicmodels")
data("AssociatedPress", package = "topicmodels")
D <- nrow(AssociatedPress)
folding <-
  sample(rep(seq_len(10), ceiling(D))[seq_len(D)])
AP_test <- AP_alpha <- list()
for (method in c("VEM", "VEM_fixed", "Gibbs")) {
  AP_alpha[[method]] <- AP_test[[method]] <- matrix(NA,
    nrow = length(topics), ncol = 10, dimnames = list(topics, seq_len(10)))
  for (fold in seq_len(10)) {
    for (i in seq_along(topics)) {
      T <- topics[i]
      FILE <- paste(method, "_", T, "_", fold, ".rda", sep = "")
      load(file.path("results", FILE))
      AP_alpha[[method]][paste(T),fold] <-
        if (is(training, "Gibbs_list")) training@fitted[[1]]@alpha
        else training@alpha
      AP_test[[method]][paste(T),fold] <- perplexity(testing,
        AssociatedPress[folding == fold,], use_theta = FALSE)
    }
  }
}
save(AP_alpha, AP_test, file = "AP.rda")
```

A.3. Fitting the LDA model to the complete data set using 40 topics

```
library("topicmodels")
data("AssociatedPress", package = "topicmodels")
topics <- 10 * c(1:5, 10, 20)
k <- 40
load("AP.rda")
alpha <- mean(AP_alpha[["VEM"]][which(topics == k),])
rm(AP_alpha, AP_test)
SEED <- 20080806
AP <- list(
  VEM = LDA(AssociatedPress, k = k,
    control = list(alpha = alpha, seed = SEED)),
  VEM_fixed = LDA(AssociatedPress, k = k,
    control = list(alpha = alpha, estimate.alpha = FALSE, seed = SEED)),
  Gibbs = LDA(AssociatedPress, k = k, method = "Gibbs",
    control = list(alpha = alpha, seed = SEED, burnin = 1000,
    thin = 100, iter = 1000)))
save(AP, file = "AP-40.rda")
```

Affiliation:

Bettina Grün
Institut für Angewandte Statistik / IFAS
Johannes Kepler Universität Linz
Altenbergerstraße 69
4040 Linz, Austria
E-mail: Bettina.Gruen@jku.at
URL: http://ifas.jku.at/gruen/

Kurt Hornik
Institute for Statistics and Mathematics
WU Wirtschaftsuniversität Wien
Augasse 2–6
1090 Wien, Austria
E-mail: Kurt.Hornik@R-project.org
URL: http://statmath.wu.ac.at/~hornik/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 40, Issue 13 *Submitted:* 2010-09-23
May 2011 *Accepted:* 2011-04-26