Sparse Topic Modeling with Concave-Convex Procedure: EMish Algorithm for Latent Dirichlet Allocation

Jussi Kujala

Department of Information and Computer Science
Aalto University School of Science and Technology*
Finland
jussi.kujala@iki.fi

Abstract. This paper studies an efficient approach to infer a sparse topic model for a text corpus using Latent Dirichlet Allocation (LDA). To this end, we use the Concave-Convex Procedure (CCCP) to optimize over the LDA objective. The resulting update equations are intuitive and simple. The improvement over previous methods is that either the previous methods are computationally more expensive, they do not parallelize well in theory or are not applicable for parallel computation models such as MapReduce, or they have approximation issues with small parameter values of the Dirichlet prior in LDA. The algorithm suggested in this paper, LDA-CCCP, parallelizes embarrassingly well, because the most time-consuming part of the algorithm is identical to the EM algorithm for Probabilistic Latent Semantic Allocation. Our preliminary experiments show that LDA-CCCP performs computationally well and it has accuracy similar to competing methods.

1 Introduction

Algorithms for topic models seek structure in documents that corresponds to the intuitive notion of topics. The inferred topics are useful for several purposes, e.g., in finding similarities between documents or documents that have a particular topic.

The topics intuitively have certain properties. For example, a document typically belongs to a small set of topics, and a topic has a constrained vocabulary (at least if common words are modeled separately [15]). Dirichlet prior is one approach to incorporate sparseness. The Dirichlet prior is a probability distribution over monomial distributions. In it the density p of a particular word is proportional to $p^{\alpha-1}$ with a parameter $\alpha > 0$. Hence, low values of the parameter α indicate smaller probabilities for an average word. For example, values of the parameter α very close to zero cause a single word to have probability one.

The data sets from which we want to find topics might be large, because even small subsets of text written by humanity are large. Furthermore, topic models

* Previously known as Helsinki University of Technology (HUT).

are not limited to text, and they have also been applied to other sources of data, for example behavior of humans [6], biological data, and software traces [1]. All of these sources potentially produce large quantities of data. Hence, efficient and distributed algorithms for topic models are required.

Several methods have been devised for topic models, for example Probabilistic Latent Semantic Allocation (PLSA) [9] and Latent Dirichlet Allocation (LDA) [4]. LDA takes advantage of the properties of the Dirichlet prior: in LDA the term distributions of topics and topic distributions of documents have Dirichlet priors.

This paper studies an approach to efficiently compute an approximate maximum of a posterior of a distribution with Dirichlet prior. The focus is on LDA, but the method is more general than that. More precisely, we use Concave-Convex Procedure (CCCP) to optimize over the LDA objective. CCCP is an iterative heuristic that finds a minimum of a function: in each iteration it finds the location of the minimum of the convex part of the function plus linearized concave part approximated at the current location. In LDA the concave part is the Dirichlet prior. Intuitively, CCCP on LDA first solves a simpler problem (PLSA), where the priors over topics in documents and words in topics are uniform, and proceeds to refine that solution to a more sparse one.

With CCCP we derive an EM -like algorithm for LDA, which we call LDA-CCCP. We can not directly use EM algorithm for LDA, because it is difficult to take into account constrains that probabilities are positive.

In short, we believe that the main contributions of LDA-CCCP are in computational properties. More precisely, it has the following advantages compared to the previous work:

- The most time consuming part of LDA-CCCP is identical to computations in the EM algorithm for the PLSA. Hence, the updates are embarrassingly parallel and we can implement them on parallel computation platforms such as MapReduce that seek to minimize the communication between parallel work threads. [1]
- Competing methods have issues under certain conditions. We will discuss this in more detail in Section 6, but we give a short summary here. First, the exact variational EM algorithm needs to compute digamma functions which is expensive. Also, more efficient approximations of the variational EM do not work well if the parameter α of the Dirichlet prior is small. Another approach, collapsed Gibbs sampler, is a random sampler which might explain why it converges slowly to the optimum after initial iterations [2]. Also, in theory it does not parallelize well (although in practise it does to some extent [10]) and is not applicable on computing platforms that minimize communication costs such as MapReduce [7]. Collapsed variational Bayes algorithm shares these parallelization issues.

The structure of the remaining paper is as follows. Section 2 gives the necessary background on the Dirichlet prior, topic models, and CCCP. The the-

[1] An implementation is available at `http://www.cis.hut.fi/jukujala/lda-cccp/lda-cccp.tar.bz2`. Be warned that this software is still in early alpha.

ory behind LDA-CCCP is introduced in Section 3 and the following Section 4 then presents the concrete algorithm and gives some practical advice. Section 5 presents the empirical findings. Section 6 discusses related work. Finally, Section 7 concludes.

2 Background

This section introduces the necessary background: Dirichlet prior, topic modelling methods PLSA and LDA, and the optimization method CCCP.

2.1 Dirichlet prior

The Dirichlet prior is a distribution over multinomial distributions. The density of the Dirichlet prior at a distribution $x = (x_1, \ldots, x_k)$ with K possible outcomes is

$$\texttt{Dir}(x; \alpha) = \frac{\prod_{i=1}^{K} x_i^{(\alpha-1)}}{Z(\alpha, K)},$$

in which $Z(\alpha, K)$ is the normalizing constant $\int \prod_{i=1}^{K} x_i^{\alpha-1}\, dx$ and α is the parameter of the distribution. Observe that if α is one then the Dirichlet prior is the uniform distribution.

2.2 Topic model methods: PLSA and LDA

Hofmann [9] introduced Probabilistic Latent Semantic Allocation (PLSA) which finds representative topics from a corpus of text documents, i.e., topics are distributions over words, into which the documents decompose. More precisely, PLSA assigns to each document d a probability distribution $p(w|d)$ over terms:

$$p(w|d) = \sum_{z=1}^{K} p(w|z)\, p(z|d),$$

where the *term distribution* $p(w|z)$ gives the probability of the term w appearing in the topic z and the *topic distribution* $p(z|d)$ gives the probability of document d producing a word from the topic z. The number of topics is K.

Latent Dirichlet Analysis (LDA) is an extension of PLSA where the topic and term distributions have Dirichlet priors [4]. PLSA implicitly assumes that they have uniform prior. In LDA the term distributions $p(w|z)$ have Dirichlet prior with parameter α and the topic distributions $p(z|d)$ have Dirichlet prior with parameter β. In empirical experiments LDA appears to outperform PLSA in cases where the number of parameters is large compared to the size of the data [5]. The log-density of the LDA model is:

$$\sum_{(w,d)\in D} \log p(w|d) + \sum_{z} \sum_{w} (\alpha - 1) \log p(w|z) - \sum_{d} \sum_{z} (\beta - 1) \log p(z|d), \quad (1)$$

where D is the set of word-document (w, d)-pairs in the corpus and the sums are over topics (z), terms (w), and documents (d). Note that the left part of the log-density (1) depending on the data is concave as a function of probabilities and the right part depending on the model is convex.

More detailed introduction to the literature of topic models is found, e.g., in [3].

2.3 CCCP

Constrained Concave-Convex Procedure (CCCP) is a general optimization method for functions [16]. In it the minimized function $f(x)$ is decomposed into a convex part v and a concave part c:

$$f(x) = v(x) + c(x),$$

and optimized with an iterative procedure in which $c(x)$ is linearized at previous solution x_t:

$$x_{t+1} = \arg\min_x v(x) + (x - x_t)\, c'(x_t). \tag{2}$$

The first solution x_0 is initialized to the best guess. Note that we can use any method to perform the above minimization. Sriperumbudur and Lanckriet [13] state that CCCP is widely used in machine learning and provide convergence results for it. They note that each iteration of CCCP decreases the objective function value, but it is unknown whether CCCP always converges to a local minima.

3 Theory of LDA with EM and CCCP

This section first gives the EM algorithm update equations for PLSA, then notes why applying EM is not straightforward for LDA, and finally proceeds to show the update equations of LDA-CCCP. Theoretical derivations are postponed to Appendix A.

Let q_{wdz} denote the current estimate of the conditional probability distribution $p(z|w, d)$ of the topic of the word w in the document d. Let q_{wz} denote the sum over all documents $\sum_d q_{wdz}$ and similarly q_{dz} sum over words in the document $\sum_w q_{wdz}$. The EM algorithm updates for PLSA are:

$$p(w|z) = \frac{q_{wz}}{N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd}}{N(d)},$$

where $N(z)$ and $N(d)$ are the Lagrange multipliers that normalize the sums of probability distributions to one. Note we can compute the values q_{wdz} in parallel, and also to some extent the distributions $p(w|z)$ and $p(z|d)$. More precise details on parallelization and an example of an implementation on MapReduce framework is found in [6].

Similarly derived EM algorithm update equations for LDA are:

$$p(w|z) = \frac{q_{wz} + \beta - 1}{N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd} + \alpha - 1}{N(d)},$$

which are problematic because $\alpha - 1$ or $\beta - 1$ in the numerator could result in a negative probability.

In CCCP we proceed in rounds. During a round we will need the values of the parameters during the previous round. We will denote these by the superscript 0, for example $p^0(w|z)$ and $p^0(z|d)$. Given these, the update equations for LDA-CCCP are:

$$p(w|z) = \frac{q_{wz}}{(1-\beta)/p^0(w|z) + N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd}}{(1-\alpha)/p^0(z|d) + N(d)},$$

which are derived with the EM algorithm and the Lagrange's method. We can solve the Lagrange multipliers $N(z)$ and $N(d)$ with, e.g., a line search. Note that these updates are intuitive; the denominator is larger if the probability was small in the previous iteration.

In practise, we first set the initial parameters $p^0(w|z)$ and $p^0(z|d)$ to uniform distributions. This means that the first iteration of CCCP corresponds to PLSA. Then CCCP will refine that objective with the updates above.

4 Algorithm and Implementation Details

This section describes the LDA-CCCP algorithm in more detail and gives some practical advice, especially on avoiding local extreme values.

Table 1 gives the pseudocode for LDA-CCCP. It performs `N_CCCP` rounds of the CCCP algorithm. During each CCCP round it does `N_EM` iterations corresponding to the EM-algorithm, but in practise, we set `N_EM` to one. The CCCP algorithm assumes that during one CCCP iteration the updates are applied until convergence, and only after that the next iteration of the CCCP can start. In Appendix B we show that we do not have to perform the full minimization in Equation (2). Any parameters that improve the CCCP optimization criteria decrease the objective value. Hence, we can do only one EM iteration per CCCP round.

Our implementation of LDA-CCCP finds the Lagrange multiplier `L` in the function `mstep_single` with a simple line search where the minimal possible value for `L` is the maximum of values `-x[i]` (if the corresponding numerator `q[i]` is positive), and the maximal value is the sum of all numerators `q[i]`.

We use two heuristics to avoid local extreme values. The first heuristic modifies the Dirichlet prior depending on the current iteration number so that at the

Input:

- The corpus `corpus` containing triples `(doc_id,word_id,count)`. The ids are assumed to form a continuous range from one.
- Number of CCCP rounds `N_CCCP`, number of EM iterations during each CCCP round `N_EM`, Dirichlet parameter `alfa` for topic distributions of documents, Dirichlet parameter `beta` for term distributions of topics, number of topics `NK`, number of documents `ND`, and number of terms `NT`.

Notation: `pwz` is an array of distributions `pwz[*|z]` which are arrays of values `pwz[w|z]`. `(pwz,pzd)` is a tuple containing `pwz` and `pzd`.
Output: distributions `pwz` and `pzd`.

```
function lda-cccp(corpus)
  Initialize (pwz,pzd) by e.g. running few iterations of the PLSA algorithm
  for N_CCCP times do
    (pwz0,pzd0) := (pwz,pzd)
    for N_EM times do
      (qwz,qzd) := estep(corpus,pwz,pzd)
      (pwz,pzd) := mstep(qwz,qzd,pwz0,pzd0)
    end for
  end for
  return tuple (pwz,pzd)

function estep(corpus,pwz,pzd)
  qwz := NK arrays of NT zero elements
  pzd := ND arrays of NK zero elements
  for each triple (d,w,c) in corpus do
    zwd := vector such that (zwd[z] = pwz[w|z]*pzd[z|d])
    zwd := zwd/sum(zwd)
    qwz[w|*] += zwd
    qzd[*|d] += zwd
  end for
  return tuple (qwz,qzd)
function mstep(qwz,qzd,pwz0,pzd0)
  for all distributions qwz[*|z] do
    pwz[*|z] := mstep_single(qwz[*|z],pwz0[*|z],beta)
  end for
  for all distributions qzd[*|d] do
    pzd[*|d] := mstep_single(qzd[*|z],pzd0[*|d],alfa)
  end for
  return tuple (pwz,pzd)
function mstep_single(q,p0,alfa)
  x:= vector such that (x[i]=(1-alfa)/p0[i])
  L:= scalar such that (sum q[i]/(x[i]+L) = 1.0)
  p:= vector such that (p[i] = q[i]/(x[i]+L))
  return p
```

Table 1. Pseudocode for LDA-CCCP algorithm.

beginning of LDA-CCCP the prior does not behave as badly near the edges, and during the last iteration the prior is the Dirichlet prior. More precisely, the prior is linearized at positions $(1 - i/N)$ times the uniform distribution, where i is the current iteration number and N is the total number of iterations. The second heuristic adds smoothing to probabilities `zwd` in the function `estep`: we assume there is ca. 1% chance that a word token is generated uniformly from the known words of the document and is assigned to a random topic for the purposes of computing the conditional topic distribution $p(z|w, d)$ of the word token. These two heuristics together guarantee, e.g., that LDA-CCCP will not converge too quickly to zeros.

We implemented LDA-CCCP in Python, and bottlenecks in computation were written with C. Sorting the corpus based on the word id and secondarily on the document id resulted in runtime decreases (for example, from 62 seconds to 40 seconds). This was due to more coherent caches, because for each word token in a document LDA-CCCP has to reference both the topic distribution of that document and the word in all term distributions of topics. Finally, we noticed that underflows in floating point operations caused significant runtime increase. We addressed this by rounding each number less than 10^{-40} to zero.

The implementation is publicly available at `http://www.cis.hut.fi/jukujala/lda-cccp/lda-cccp.tar.bz2`.

5 Empirical results

In this section we provide experimental results on the performance of the LDA-CCCP algorithm. The aim is of this section is to give information on the quality of the inferred topics and computational performance of the method.

5.1 Experimental setup

Data set	Documents	Terms	Words	Unique term-document pairs
Kos	3,430	6,906	467,714	353,163
NY Times	300,000	102,660	99,542,125	69,679,430

Table 2. Data set statistics.

Data sets. We performed experiments on two data sets: Kos corpus and NY Times corpus. Both of these are available at `http://archive.ics.uci.edu/ml/machine-learning-databases/bag-of-words/`. Kos is a relatively small document collection, whereas NY Times corpus is on the same order of magnitude what we were able to store to the main memory of the computer on which we performed the experiments. Table 2 gives the summary statistics of the data sets.

Algorithms/software. We compare five different software:

- `C-LDA` is an implementation of the variational EM available at `http://www.cs.princeton.edu/~blei/lda-c/`. Unfortunately, in `C-LDA` it is not possible to set the Dirichlet prior for the term distributions of topics. Instead, `C-LDA` freely optimizes term distributions during each iteration. Also, NY Times data set was too large for `C-LDA`.
- `GibbsLDA++` [11] is an implementation of the collapsed Gibbs sampling available from `http://gibbslda.sourceforge.net/`.
- `FastLDA` [12] is an implementation of collapsed Gibbs sampling which is faster than `GibbsLDA++` but does not output topic distributions. We use it to measure the speed of collapsed Gibbs sampling. The implementation is available at `http://www.ics.uci.edu/~iporteou/fastlda/fastldacode.tar.gz`.
- Our implementation of LDA-CCCP.
- EM algorithm implementation of `PLSA` that is integrated in LDA-CCCP.

Setup. We randomly split the *documents* in the corpus to train and test set. The train set contains 80% of the documents and the test set contains 20% of the documents. The models are trained with 200 iterations over all words in the train set. Topic distributions of documents in the test set are inferred with an EM algorithm. The parameter α of the topic distributions of documents is set to 0.5 and the parameter β of the term distributions of topics is set to 0.01 (we also perform additional grid searches over the parameters to verify these values). The number of topics is set to 100.

5.2 Quality of the inferred topics

We want to know the quality of the inferred topics. Unfortunately, even defining the quality of topics is in itself a difficult problem. Chang et al. [5] observed that test set likelihood does not always correlate with the semantic coherence of the topics interpreted by people. However, test set likelihood is easy to measure so we use it. Instead of data likelihood we measure how well the topics are able to encode test data. Hence, we encode each word in a document using term distributions. For a word w in a document d the length $l(w, d)$ of the encoding is the maximum of $-\log p(w|d)$ and $\log W$, where W is the number of terms in the vocabulary. The idea is that topic distributions are used if they are useful and otherwise a uniform distribution is used. We also get around the problem that if there is a previously unknown word in the test set then we can not assign a positive probability to it (note that even fully Bayesian methods can not deal with unknown events). The length $l(w, d)$ is finally divided by the number of words in the test set times $\log W$ to give a comparison to uniform distribution. This normalizes the quality metric to interval $[0, 1]$.

Figures 1 and 2 present the results. On the smaller Kos data set there is a large difference between PLSA and methods based on LDA. The results also suggest that the Gibbs sampler converges slowly, if ever, which has been observed before [2]. We did an additional grid search over the parameter ranges $\alpha \in \{0.01, 0.1, 0.5\}$ and $\beta \in \{0.001, 0.005, 0.009, 0.01, 0, 02, 0.05, 0.1\}$, but the original

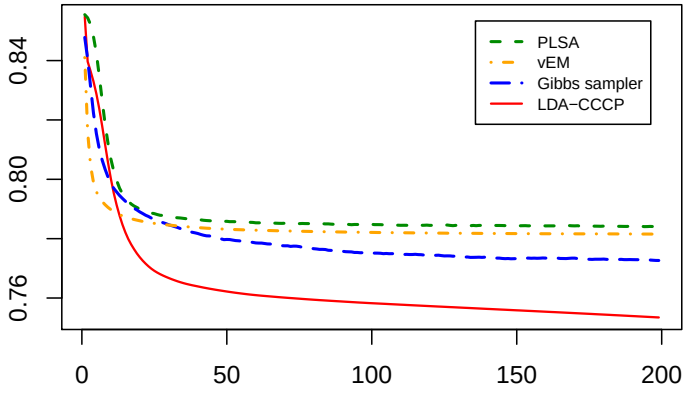

Fig. 1. Performance of algorithms during different iterations on the Kos dataset. Smaller is better. Note that the variational EM (vEM) algorithm is not comparable to others, because the implementation does not allow to set the Dirichlet parameter of term distributions of topics.

Data Set / Algorithm	PLSA	C-LDA	GibbsLDA++	LDA-CCCP
Kos 2 iterations	0.854	0.815	0.841	0.837
Kos 10 iterations	0.808	0.790	0.799	0.784
NY Times 2 iterations	0.780	NA	0.781	0.778
NY Times 10 iterations	0.750	NA	0.729	0.752

Table 3. Performance of algorithms when run for a small number of iterations. Smaller is better.

parameters were the best for the Gibbs sampler. However, the results also suggest that Gibbs sampling might converge faster during the first iterations, perhaps because its updates are run sequentially. On the larger NY Times data set the differences in accuracy were small. Smaller number of parameters per data might provide a plausible explanation. On the Kos data set the number of parameters per word is 2.21, whereas on the NY Times data set it is 0.40.

Note that the performance of LDA-CCCP depends on the total number of iterations it is run, so although in these figures it converges slower during the first iterations, nothing can be said on its performance if it is run, e.g., only ten iterations. Hence, Table 3 shows the performance for a small number of iterations which might be a more realistic setting for large-scale data sets.

5.3 Computational performance

We now turn our attention to the computational performance. Table 4 presents the runtimes of the algorithms per one iteration. The runtimes exclude parsing

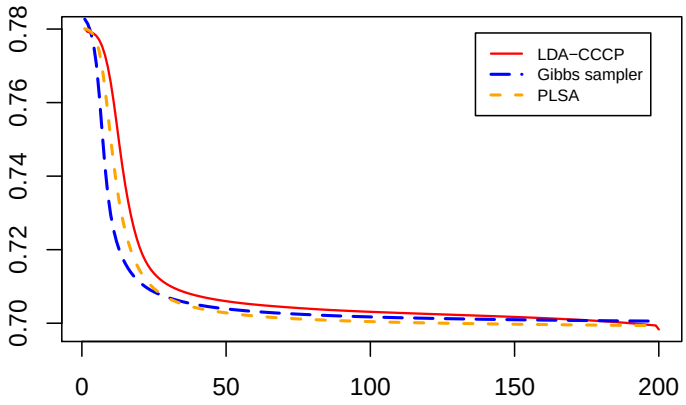

Fig. 2. Performance of algorithms during different iterations on the NY Times dataset. Smaller is better.

Algorithm / Data Set	Kos	NY Times
`C-LDA`	183.0	NA
`GibbsLDA++`	0.682	176.0
`FastLDA`	0.572	264.6
`LDA-CCCP`	0.192	40.1

Table 4. The runtime in seconds of different software packages for one iteration over the train sets.

and preprocessing of input data and are averaged over 200 iterations over the train set.

The exact variational EM algorithm is the slowest. We did not run it on the larger NY Times data set. Other approaches were comparable in speed. Possible differences in runtime could be contributed to differences in implementation. We do not know why the Gibbs sampler methods are slower than LDA-CCCP, because the algorithms are similar, generating sufficiently random numbers is not expensive, and there are not that many more word tokens than unique word-document -pairs.

6 Comparison to related work

We are aware of three approaches on optimizing LDA: in their original paper on LDA Blei et al. [4] used variational EM, Griffiths et al. [8] suggested a collapsed Gibbs sampler algorithm, and Asuncion et al. [2] proposed an approach they called collapsed variational Bayes. We now go through each of these approaches and discuss how they compare to LDA-CCCP.

The variational EM algorithm uses the EM algorithm to find a good approximation to the complete posterior distribution P with a simpler distribution Q in which typically some variables are assumed to be independent of other. In LDA, the approximating distribution Q is composed of Dirichlet distributions for topic distributions $p(z|d)$ and term distributions $p(w|z)$, and monomial distributions for topic distributions of words in the corpus. To find a good approximation Q, the variational EM minimizes the Kullback-Leibler divergence $D(Q|P)$ between the exact P and approximate Q distributions:

$$D(Q|P) = \int Q(x) \log P(x)\, dx - \int Q(x) \log Q(x).$$

Intuitively, $D(Q|P)$ measures how many bits we lose by encoding items sampled from the distribution Q with an optimal code for distribution P. Also, it has a connection to the information retrieval concept of precision [14]. More concretely, one possible way to phrase the parameter updates for the variational EM updates is as follows [2]:

$$q_{wdz}^{\text{new}} \propto \frac{\exp \psi(q_{wz} + \beta)\ \exp \psi(q_{dz} + \alpha)}{\exp \psi(q_z + W\,\beta)}, \tag{3}$$

where q_{wdz} was earlier defined to be the current estimate of $p(z|w,d)$, $q_{wz} = \sum_d q_{wdz}$, $q_{dz} = \sum_w q_{wdz}$, $q_z = \sum_{w,d} q_{wdz}$, and $\psi(x)$ is the digamma function. A problem in the updates (3) is the computational cost of computing the digamma function. Hence, one approach is to approximate this update equation with [2]:

$$q_{wdz}^{\text{new}} \propto \frac{(q_{wz} + \beta - 0.5)\,(q_{dz} + \alpha - 0.5)}{q_z + W\,\beta - 0.5},$$

which is accurate for $x > 1$ in $\psi(x)$. However, because the numerator has terms such as $\alpha - 0.5$ in it the approximation performs poorly if the parameter α is small [2].

Another approach is to use **collapsed Gibbs sampling**. In Gibbs sampling all random variables in the model are sequentially re-sampled conditioned on the current values of all other variables. In *collapsed* Gibbs sampling for LDA the term distributions of topics and topic distributions of documents are integrated away, i.e., collapsed away. The conditional topic distribution of a single word token given all other word tokens is:

$$p(z|w,d) \propto \frac{(N_{wz}^{\neg} + \beta)\ (N_{dz}^{\neg} + \alpha)}{N_z^{\neg} + W\,\beta}, \tag{4}$$

where the superscript $\neg$ indicates that the current word token is not taken into account when computing the value, and the variables N are analogous to the variables q except that they are concrete counts of tokens. Asuncion et al. [2] noted that convergence of the test accuracy of the collapsed Gibbs sampler can be slow even when run for a large number of iterations. Also, in theory the Gibbs sampler can not be implemented in parallel, because each topic distribution

depends on the values of the other random variables. If the dependence between counts is completely ignored during one iteration over data, then the updates (4) appear wrong: they resemble the EM algorithm updates of the PLSA algorithm if $\alpha \to 0$ and $\beta \to 0$ and are different if $\alpha = 1$ and $\beta = 1$ which should correspond to PLSA updates. However, in practise it is possible to parallelize collapsed Gibbs sampler to some extent [10].

One advantage of the collapsed Gibbs sampler is the memory consumption when the number of topics is high compared to the average number of words in documents: collapsed Gibbs sampler requires tracking the current discrete topic counts of documents in the corpus, whereas other approaches need to keep track of whole topic distributions of documents.

The last approach is **collapsed variational Bayes** [2], which is a deterministic analogue to collapsed Gibbs sampler. It is derived by collapsing away the term distributions of topics and topic distributions of documents. After that, the remaining topic distributions of words in the corpus are treated variationally and approximated with monomial distributions. The resulting variational EM optimization contains intractable summations, so it must be approximated. Even then, the update equations are complicated, so an approximation to these, called CVB0, was proposed by Asuncion et al. [2]. In it the update equations are:

$$q_{wdz}^{\text{new}} \propto \frac{(q_{wz}^{\neg} + \beta)\ (q_{dz}^{\neg} + \alpha)}{q_{z}^{\neg} + W\,\beta}.$$

The difference to the collapsed Gibbs sampler updates is that CVB0 works over the topic distribution, whereas the collapsed Gibbs sampler randomly assigns each word to a topic. A general difference between the collapsed methods and other methods is that the collapsed algorithms work sequentially over the tokens and update their variables ($q_{wz}^{\neg}$, $q_{dz}^{\neg}$, and $q_{k}^{\neg}$ in CVB0) after processing each token. This causes issues with parallelization. We see that if in CVB0 the updates are heuristically performed in parallel then the same observations apply as for the collapsed Gibbs sampler. We implemented CVB0 updates fully in parallel (by modifying LDA-CCCP) and their performance was not comparable to other approaches based on LDA. On the Kos data set the performance of these parallel updates was 0.784, the same as for PLSA.

The parallelization issues make collapsed methods difficult to implement on computation platforms, such as MapReduce, which seek to minimize the communication between parallel jobs (Map-step of MapReduce). On them computing the topic distribution of the word tokens must be parallel, like it is in the EM algorithm for PLSA.

7 Conclusions

We studied the suitability of CCCP for optimizing the LDA criteria. The resulting theoretical updates were intuitive in the sense that they produce sparser models than PLSA. The most time consuming part of the LDA-CCCP is identical to computations in PLSA, thus making the approach viable for as large data

sets as PLSA. We expect LDA-CCCP be superior to PLSA if the number of parameters is high compared to the size of data. Also, it is possible to implement LDA-CCCP on parallel computation platforms such as MapReduce.

References

1. Andrzejewski, D., Mulhern, A., Liblit, B., Zhu, X.: Statistical debugging using latent topic models. In: ECML '07: Proceedings of the 18th European conference on Machine Learning. pp. 6–17. Springer-Verlag, Berlin, Heidelberg (2007)
2. Asuncion, A., Welling, M., Smyth, P., Teh, Y.W.: On smoothing and inference for topic models. In: Proceedings of 29th Conference on Uncertainty in Artificial Intelligence (UAI) (2009)
3. Blei, D., Lafferty, J.: Topic models. In: Srivastava, A., Sahami, M. (eds.) Text Mining: Theory and Applications. Taylor and Francis (2009)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
5. Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., Blei, D.: Reading tea leaves: How humans interpret topic models. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems 22, pp. 288–296 (2009)
6. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: WWW '07: Proceedings of the 16th international conference on World Wide Web. pp. 271–280. ACM, New York, NY, USA (2007)
7. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Communications of the ACM 51(1), 107–113 (2008)
8. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences of the United States of America 101(Suppl 1), 5228–5235 (2004)
9. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. Machine Learning 42(1-2), 177–196 (2001)
10. Newman, D., Asuncion, A., Smyth, P., Welling, M.: Distributed algorithms for topic models. Journal of Machine Learning Research 10, 1801–1828 (2009)
11. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: WWW '08: Proceeding of the 17th international conference on World Wide Web. pp. 91–100. ACM, New York, NY, USA (2008)
12. Porteous, I., Newman, D., Ihler, A., Asuncion, A., Smyth, P., Welling, M.: Fast collapsed Gibbs sampling for latent dirichlet allocation. In: KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 569–577. ACM, New York, NY, USA (2008)
13. Sriperumbudur, B., Lanckriet, G.: On the convergence of the concave-convex procedure. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems 22, pp. 1759–1767 (2009)
14. Venna, J., Peltonen, J., Nybo, K., Aidos, H., Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. Journal of Machine Learning Research 11, 451–490 (2010)
15. Wallach, H., Mimno, D., McCallum, A.: Rethinking lda: Why priors matter. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems 22, pp. 1973–1981 (2009)

16. Yuille, A.L., Rangarajan, A.: The concave-convex procedure. Neural Computation 15(4), 915–936 (2003)

A Theory on LDA-CCCP update equations

In this Appendix we derive update equations for LDA-CCCP. Let us first define our notation. We denote by data D the word tokens in the corpus, i.e., term-document -pairs (w, d). Topics are denoted by variable z. The parameters of the model, e.g., distributions $p(z|d)$ and $p(w|z)$, are denoted by t. Now we can abbreviate the log-likelihood in Expression (1) to $\log L(D, t)$. Sums with subindex w are over terms of the vocabulary, with d over documents, and with z over topics.

Recall that EM-algorithm consists of two steps, where in the E-step we assume some hidden things in the model to be fixed, which in the case of the PLSA are the topic distributions $p(z|w, d)$ of words in the corpus. Then we compute the expected log-likelihood over the parameters t of the model assuming that these topic distributions are fixed, which equals to assuming that they are generated by the previous values for the parameters t, denoted by t':

$$Q(t|t') = E_{Z\sim Dist(z|t')} \log L(t|D, Z),$$

where Z indicates topic assignments to words and $Dist(z|t')$ is the distribution from which they are sampled. Then in the M-step the EM-algorithm finds the optimal values for the parameters t given the above expression:

$$t_{\text{new}} = \arg\max_t Q(t|t').$$

We will need an expression for $L(t|D, Z)$ which is easy to obtain through Bayes rule:

$$L(t|D, Z) = \frac{L(D|t, Z)L(t)}{L(D|Z)}.$$

The numerator $L(D|Z)$ does not depend on the parameters t so we ignore it. The log-likelihood $\log L(D|t, Z)$, where we assume topics of words to be known, is

$$\sum_{(w,d)\in D} \log p(w|Z_{w,d}) + \sum_{(w,d)\in D} \log p(Z_{w,d}|d),$$

where $Z_{w,d}$ refers to the known topic of the word w in the document d. The expectation of $\log L(D|t, Z)$ over topics is given the by previous parameters t' (containing distributions $p'(w|z)$ and $p'(z|d)$) from which we can compute the topic distribution $q(z|w, d)$ of word tokens:

$$q(z|w, d) = \frac{p'(w|z)\, p'(z|d)}{p'(w|d)}$$

Hence,

$$Q(t|t') = \sum_{(w,d)\in D} \sum_z q(z|w, d) \log p(w|z) + \sum_{(w,d)\in D} \sum_z q(z|w, d) \log p(z|d). \quad (5)$$

Maximizing the above expression equals doing an optimization over the distributions $p(w|z)$ and $p(z|d)$. The multiplier of $\log p(w|z)$ is the sum of all term specific probabilities $q(z|w,d)$ in the corpus, which we will denote with the constant

$$q_{wz} = \sum_{(w',d)\in D} I(w'=w)\, q(z|w',d).$$

Similarly we will denote by q_{dz} the multiplier of $\log p(z|d)$, summed over all words appearing in the document d. Equation (5) now simplifies to

$$Q(t|t') = \sum_z \sum_w q_{wz} \log p(w|z) + \sum_d \sum_z q_{dz} \log p(z|d). \tag{6}$$

Hence, given all constants q_{xy} we have several independent optimization tasks, because the parameters are coupled only if they are part of the same conditional probability distribution. For example, given a particular topic z the parameters $p(w|z)$ over all terms w are coupled, because $\sum_w p(w|z) = 1$.

EM updates in PLSA. In PLSA the prior $L(t)$ over parameters is flat, so we can use directly Lagrange's method to maximize the right side of Equation (6), which gives the updates:

$$p(w|z) = \frac{q_{wz}}{N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd}}{N(d)},$$

where $N(z)$ and $N(d)$ are the Lagrange multipliers that normalize the sums of probability distributions to one.

EM updates in LDA. However, if the prior $L(t)$ is the Dirichlet distribution:

$$\log L(t) = (\alpha - 1) \sum_{d,z} \log p(z|d) + (\beta - 1) \sum_{z,w} \log p(w|z),$$

then the maximization no longer has single maxima, because $Q(t|t')$ is concave and $\log L(t)$ is convex. For example, applying Lagrange's method to $Q(t|t') + \log L(t)$ and disregarding positivity constrains for probabilities gives updates of the form

$$p(w|z) = \frac{q_{wz} + \beta - 1}{N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd} + \alpha - 1}{N(d)},$$

which are problematic because $\alpha - 1$ or $\beta - 1$ in the numerator could result in negative probability.

EM updates in LDA-CCCP. Now we finally derive the update equations for LDA-CCCP. We wanted to minimize $-Q(t|t') - \log L(t)$, where $-Q(t|t')$ is convex and $-\log L(t)$ is concave. In CCCP the concave part is linearized during

each iteration of the CCCP. Let parameters t^0 denote the parameters of the previous CCCP iteration: distributions $p^0(z|d)$ and $p^0(w|z)$. Linearized Dirichlet prior without the constant part is then:

$$t \cdot L'(t^0) = (\alpha - 1) \sum_{d,z} \frac{p(z|d)}{p^0(z|d)} + (\beta - 1) \sum_{z,w} \frac{p(w|z)}{p^0(w|z)}.$$

Optimizing this together with $Q(t|t')$ results in the following updates:

$$p(w|z) = \frac{q_{wz}}{(1-\beta)/p^0(w|z) + N(z)} \text{ and}$$
$$p(z|d) = \frac{q_{zd}}{(1-\alpha)/p^0(z|d) + N(d)}.$$

B Incomplete CCCP

Recall that in CCCP a function $f(x) = v(x) + c(x)$ is decomposed to a convex part $v(x)$ and a concave part $c(x)$. Then one CCCP iteration consists of computing:

$$x_{t+1} = \arg\min_x v(x) + (x - x_t)\, c'(x_t). \tag{7}$$

However, if we use, e.g., EM algorithm to perform the minimization, then it seems wasteful to run the minimization until convergence. Unfortunately, we did not find convergence results on how CCCP performs with partial optimization. Nevertheless, the argument below is a direct implication of techniques used in [16, 13], although is not for some reason stated explicitly in them.

Any solution x_{t+1} that improves Expression (7) over x_t decreases the objective value $f(x)$. To see why, note that x_{t+1} improves the expression on the right side of Equation (7):

$$v(x_t) > v(x_{t+1}) + (x_{t+1} - x_t)\, c'(x_t),$$

Also, Jensen's inequality on the concave $c(x)$ implies that

$$c(x_{t+1}) < c(x_t) + (x_{t+1} - x_t)\, c'(x_t),$$

which is equal to:

$$c(x_t) > c(x_{t+1}) - (x_{t+1} - x_t)\, c'(x_t).$$

When summed together:

$$\begin{aligned} f(x_t) &= v(x_t) + c(x_t) \\ &> v(x_{t+1}) + c(x_{t+1}) \\ &= f(x_{t+1}). \end{aligned}$$

The claim follows.