Scalable Moment-based Inference for Latent Dirichlet Allocation

Chi Wang, Xueqing Liu, Yanglei Song, and Jiawei Han

Computer Science Department, University of Illinois at Urbana-Champaign,
Urbana, IL 61801, USA
{chiwang1, xliu93, ysong44, hanj}@illinois.edu

Abstract. Topic models such as Latent Dirichlet Allocation have been useful text analysis methods of wide interest. Recently, moment-based inference with provable performance has been proposed for topic models. Compared with inference algorithms that approximate the maximum likelihood objective, moment-based inference has theoretical guarantee in recovering model parameters. One such inference method is tensor orthogonal decomposition, which requires only mild assumptions for exact recovery of topics. However, it suffers from scalability issue due to creation of dense, high-dimensional tensors. In this work, we propose a speedup technique by leveraging the special structure of the tensors. It is efficient in both time and space, and only requires scanning the corpus twice. It improves over the state-of-the-art inference algorithm by one to three orders of magnitude, while preserving equal inference ability.

1 Introduction

Statistical topic modeling techniques are powerful tools for exploring large data sets such as text and social networks. They are frequently used for text summarization, dimensionality reduction and community detection. One important model is latent Dirichlet allocation (LDA) [6], which has widespread use and variations in data mining and machine learning. It models documents as mixtures of multiple topics, while every topic is modeled as a multinomial distribution over a vocabulary. We consider the unsupervised inference problem for LDA: estimating the unknown word distribution of every topic so as to fit the observed word occurrences in the documents.

The inference can be performed under different principles. Maximum likelihood is the most commonly employed principle, but exact inference based on this objective is proved to be intractable [5]. Recently, researchers have found that a new inference principle, *method of moments*, enables tractable computations to recover the topics with theoretical bound [2, 3]. The intuition is to relate model parameters to *population moments*, which are expected frequencies of co-occurred word pairs, triples *etc.*, and infer parameters from empirical estimation of the population moments. Under mild assumptions, a *tensor orthogonal decomposition* algorithm in [3] can perform error-bounded topic recovery, with best known sample complexity and numerical stability. However, it has a severe scalability issue, which is the the problem we tackle in this paper.

The tensor orthogonal decomposition algorithm has two steps. In the first step, it collects empirical moments (*i.e.*, expectation of word co-occurrence frequencies) from the data, and constructs two large and dense tensors (*i.e.*, hypermatrices). In the second step, it uses the two large tensors to compute a small tensor, and performs orthogonal decomposition for the small tensor based on an iterative tensor power method. With their scale being V^2 and V^3 respectively, where V is the vocabulary size, the large and dense tensors are prohibitive to construct. [3] suggests an alternative by computing the tensor power iterations on the fly scanning through the original data, without creating any tensor in memory. However, it requires one scan of the data *per iteration*. The efficiency is not satisfactory for large scale corpora.

In this work, we propose a novel strategy to scale up the tensor orthogonal decomposition algorithm. By careful analysis of the problem, we advocate to avoid explicit creation of large and dense tensors, but still construct the small tensor and store it in memory. With this strategy, we bypass the scale bottleneck, yet are still able to perform efficient tensor power iterations. To directly construct the small tensor without creating the large and dense tensors, we leverage the special structures of the moments: *sparse*, *low rank* and *decomposable*. We design an efficient algorithm that only requires two scans of data in total while consuming much smaller space. With experiments on both synthetic and real data, we demonstrate that our method can be 20-3000 times faster than the state-of-the-art inference method, while preserving robust topic recovery capability.

2 Related Work

In the last decade, statistical topic modeling techniques have gained popularity. Two important methods are probabilistic latent semantic analysis (PLSA) [12] and its Bayesian extension latent Dirichlet allocation (LDA) [6]. They model the generative process of each word from each document in a corpus. The model parameters can be partitioned into corpus-level (the unknown word distribution of every topic) and document-level (the unknown topic distribution of every document). The goal of inference is to find parameters that best explain the observed data, *i.e.*, word occurrences in the documents. Yet there are different principles to quantify what it means by 'best explain the observed data'.

Most of existing topic model inference methods are based on the *maximum likelihood* (ML) principle (including its Bayesian version *maximum a posterior*). For example, PLSA [12] uses an Expectation-Maximization algorithm to approximately optimize the data likelihood. For LDA, two most popular approximate inference methods have been variational Bayesian inference [6] and Markov Chain Monte Carlo (especially Gibbs sampling) [9]. In spite of the vast body of followup work, the computational complexity of ML inference is not studied until 2011. Sontag and Roy [19] show that the document-level inference is not always well defined, and Arora, Ge and Moitra [5] prove the NP-hardness of exact corpus-level ML inference.

In accordance with their theoretical hardness, the above inference methods tend to suffer from slow convergence and long runtime. As a result, there has been a substantial amount of work targeting on accelerating the above methods. *e.g.*, by leveraging sparsity [18, 20, 11] and parallelization [16, 21], or online learning mechanism [1, 10, 8]. However, none of them have theoretical guarantee of convergence within a bounded number of iterations, and are nondeterministic either due to sampling or the random initialization.

Recently, an alternative inference of topic models has been proposed based on the *method of moments* [2], and improved in [3]. Compared with ML inference, it has the following two advantages: i) the distance between inferred corpus-level parameters and the true parameters has a theoretical upper bound that is inversely related to sample size; ii) the convergence is guaranteed with a bounded number of iterations. Another related study [5] assumes the existence of *anchor word* that only exists in one topic, and uses that assumption to bound the recovery error. Its efficiency is improved in [4]. This method requires stronger assumptions than [2] and the error bound is weaker.

3 Preliminaries

We first introduce the notations:

- The input to the inference problem is a corpus of D documents with vocabulary size V. The i-th document d_i has l_i word tokens, and the whole corpus has L tokens in total. We use a k-dimensional vector θ_i $(i \in [D])$ to denote the document-level topic distribution for d_i, and $\alpha = (\alpha_1, \ldots, \alpha_k)$ to denote the Dirichlet prior where θ_i's are drawn from. Larger α_t posits stronger prior at $\theta_{i,t}$. Define $\alpha_0 = \sum_{t=1}^{k} \alpha_t$. We use a V-dimensional vector ϕ_t $(t \in [k])$ to denote the corpus-level word distribution for topic t. To generate a token $w_{i,j} \in [V]$ in position j of document d_i, one first samples a topic $z_{i,j}$ according to θ_i, and then samples a word from $\phi_{z_{i,j}}$;
- A tensor is a hypermatrix that can contain more than two degrees. The outer product $\otimes$ of any p-degree tensor $A \in \mathbb{R}^{s_1 \times \cdots \times s_p}$ and any q-degree tensor $B \in \mathbb{R}^{s_{p+1} \times \cdots \times s_{p+q}}$ is a $(p+q)$-degree tensor $A \otimes B \in \mathbb{R}^{s_1 \times \cdots \times s_{p+q}}$: $A \otimes B[i_1, \ldots, i_{p+q}] = A[i_1, \ldots, i_p] B[i_{p+1}, \ldots, i_{p+q}]$;
- For any tensor $A \in \mathbb{R}^{s \times s \times s}$, matrix $B \in \mathbb{R}^{s \times s_1}, C \in \mathbb{R}^{s \times s_2}, D \in \mathbb{R}^{s \times s_3}$, $A(B, C, D)$ is a tensor in $\mathbb{R}^{s_1 \times s_2 \times s_3}$, $A(B, C, D)[i_1, i_2, i_3] = \sum_{j_1, j_2, j_3 \in [s]} A[j_1, j_2, j_3] B[j_1, i_1] C[j_2, i_2] D[j_3, i_3]$;
- W^+ denotes the Moore-Penrose pseudoinverse of W;
- $\Omega(A, a, b, c)$ permutes the modes of tensor A, s.t. $\Omega(A, a, b, c)[i_1, i_2, i_3] = A[i_a, i_b, i_c]$.

We focus on the corpus-level inference problem in this paper. The document-level inference problem can be solved using the method in [19] after we infer the corpus-level parameters. Our goal is to recover the unknown ϕ_t's based on the observed corpus.

Anandkumar *et al.* [3] propose a tractable exact inference method based on the *method of moments*. In statistics, the ξ-th order *population moment* of a random variable is the expectation of its ξ-th power. The method of moments derives equations that relate the population moments to the model parameters. Then, it collects empirical moments from observed samples, and solves the equations using the empirical moments in place of the population moments.

In our case, the random variable is a token $w_{i,j}$ in a document d_i. The value of $w_{i,j}$ is a word in $[V]$. The ξ-th population moment is the expected co-occurrence of words in ξ token positions. We can collect empirical moments from the corpus, and estimate ϕ_t's by fitting the empirical moments with population moments. For example, the 2nd order moment is a matrix $E_2 \in \mathbb{R}^{V \times V}$. The element $E_2[x_1, x_2]$ in x_1-th row and x_2-th column of E_2 is equal to the probability $w_{i,1}$ being x_1 and $w_{i,2}$ being x_2 given α.

$$\begin{aligned} &E_2[x_1, x_2] = p(w_{i,1} = x_1, w_{i,2} = x_2 | \alpha) \\ &= \int_{\theta_i} p(\theta_i|\alpha) \sum_{t_1=1}^{k} \sum_{t_2=1}^{k} p(z_{i,1} = t_1|\theta_i) p(z_{i,2} = t_2|\theta_i) p(w_{i,1} = x_1 | z_{i,1} = t_1) \\ &\cdot p(w_{i,2} = x_2 | z_{i,2} = t_2) d\theta_i = \sum_{t_1 \neq t_2} \frac{\alpha_{t_1}\alpha_{t_2}}{\alpha_0(\alpha_0+1)} \phi_{t_1,x_1}\phi_{t_2,x_2} + \sum_{t=1}^{k} \frac{\alpha_t(\alpha_t+1)}{\alpha_0(\alpha_0+1)} \phi_{t,x_1}\phi_{t,x_2} \end{aligned} \tag{1}$$

Likewise, we can derive the 3rd order moment as a tensor $E_3 \in \mathbb{R}^{V \times V \times V}$. The element $E_3[x_1, x_2, x_3]$ is equal to the probability $w_{i,1}$ being x_1, $w_{i,2}$ being x_2 and $w_{i,3}$ being x_3 given α. The equation below follows a similar derivation as Eq. (1), written in a more concise form:

$$\begin{aligned} E_3 = &\sum_{t_1 \neq t_2 \neq t_3 \neq t_1} \frac{\alpha_{t_1}\alpha_{t_2}\alpha_{t_3}}{\alpha_0(\alpha_0+1)(\alpha_0+2)} \phi_{t_1} \otimes \phi_{t_2} \otimes \phi_{t_3} \\ + &\sum_{t_1 \neq t_2} \frac{\alpha_{t_1}\alpha_{t_2}(\alpha_{t_1}+1)}{\alpha_0(\alpha_0+1)(\alpha_0+2)} (\phi_{t_1} \otimes \phi_{t_1} \otimes \phi_{t_2} + \phi_{t_1} \otimes \phi_{t_2} \otimes \phi_{t_1} + \\ + &\phi_{t_2} \otimes \phi_{t_1} \otimes \phi_{t_1}) + \sum_{t=1}^{k} \frac{\alpha_t(\alpha_t+1)(\alpha_t+2)}{\alpha_0(\alpha_0+1)(\alpha_0+2)} \phi_t \otimes \phi_t \otimes \phi_t \end{aligned} \tag{2}$$

Moment-based inference methods set the left hand sides to empirical estimation of moments, and solve these equations to estimate the parameters ϕ_t's.

In general, one can compute moments of an arbitrary order, but the cost can be high for computing high-order moments. Fortunately, Anandkumar *et al.* [3] find that we only need up to 3rd order moments to infer an LDA model, under some mild non-degeneracy conditions. We restate their algorithm in Section 4, and propose a more scalable algorithm in Section 5.

4 Tensor Orthogonal Decomposition

The tensor orthogonal decomposition algorithm for LDA relies on the following theorem (revised statement of Theorem 4.3 in [3]).

Theorem 1. *Assume that*

$$M_2 = \sum_{t=1}^{k} \lambda_t v_t \otimes v_t, M_3 = \sum_{t=1}^{k} \lambda_t v_t \otimes v_t \otimes v_t \tag{3}$$

where $\lambda_t > 0, v_t \in \mathbb{R}^V, t \in [k]$ *are linearly independent and* $\|v_t\| = 1$*. Given* $M_2 \in \mathbb{R}^{V\times V}$ *and* $M_3 \in \mathbb{R}^{V\times V\times V}$ *as input, the equations can be uniquely solved for unknown variables* λ_t *and* v_t *in polynomial time.*

Proof sketch. Let $M_2 = M\Sigma M^T$ be the spectral decomposition of M_2, define $W = M\Sigma^{-\frac{1}{2}}$ as the *whitening matrix* of M_2, then $\widetilde{v_t} = \sqrt{\lambda_t}W^T v_t$ are orthonormal since $\sum_{t=1}^{k} \widetilde{v_t} \otimes \widetilde{v_t} = I$. It follows that $\sum_{t=1}^{k} \frac{1}{\sqrt{\lambda_t}} \widetilde{v_t} \otimes \widetilde{v_t} \otimes \widetilde{v_t} = \sum_{t=1}^{k} \lambda_t (W^T v_t) \otimes (W^T v_t) \otimes (W^T v_t) = M_3(W, W, W) \equiv \widetilde{T}$. Due to the uniqueness of tensor's orthogonal decomposition [3], $\widetilde{v_t}$ and $\frac{1}{\sqrt{\lambda_t}}$ are uniquely determined from $\widetilde{T}$ in polynomial time. So each $v_t = \frac{1}{\sqrt{\lambda_t}}(W^T)^+ \widetilde{v_t}$ and λ_t are uniquely determined. ■

Note that the sole equation $M_2 = \sum_{t=1}^{k} \lambda_t v_t \otimes v_t$ is not sufficient to determine v_t uniquely, because v_t's are not constrained to be orthogonal. By defining M_1 and M_2 in the following way, M_2 fits into Eq. (3):

$$M_1 = \sum_{t=1}^{k} \frac{\alpha_t}{\alpha_0} \phi_t, M_2 = (\alpha_0 + 1)E_2 - \alpha_0 M_1 \otimes M_1 = \sum_{t=1}^{k} \frac{\alpha_t}{\alpha_0} \phi_t \otimes \phi_t \tag{4}$$

And the following definition of M_3 fits into Eq. (3):

$$U_1 = E_2 \otimes M_1, U_2 = \Omega(U_1, 1, 3, 2), U_3 = \Omega(U_1, 2, 3, 1) \tag{5}$$

$$\begin{aligned} M_3 = {} & \frac{(\alpha_0 + 1)(\alpha_0 + 2)}{2} E_3 + \alpha_0^2 M_1 \otimes M_1 \otimes M_1 \\ & - \frac{\alpha_0(\alpha_0 + 1)}{2}[U_1 + U_2 + U_3] = \sum_{t=1}^{k} \frac{\alpha_t}{\alpha_0} \phi_t \otimes \phi_t \otimes \phi_t \end{aligned} \tag{6}$$

It is clear now that the corpus-level parameters ϕ_t's can be uniquely determined by up to 3rd order moments. Algorithm 1 outlines the tensor orthogonal decomposition method for recovering the components, given the summation α_0 of Dirichlet prior α as input. It includes two main parts:

1. Lines 1.1 to 1.5 to compute the $k \times k \times k$ tensor $\tilde{T}$;
2. Lines 1.6 to 1.16 to perform orthogonal decomposition of $\widetilde{T}$ via a robust *power method*, and recover the unique λ_t's and v_t's (Line 1.13).

Lemma 5.1 in [3] ensures that the power iteration loop Line 1.10 with iteration # n converges in a quadratic rate when the tensor $\widetilde{T}$ is accurate. The outer loop with iteration # N ensures the convergence when $\widetilde{T}$ is perturbed.

Algorithm 1: Tensor Orthogonal Decomposition (TOD)

```
     Input: Corpus with $L$ tokens and vocabulary size $V$, number of components $k$,
            number of outer and inner iterations $N$ and $n$, $\alpha_0$
     Output: The model parameters $(\alpha_t, \phi_t), t = 1, \ldots, k$
1.1  Compute $M_2 \in \mathbb{R}^{V \times V}$ and $M_3 \in \mathbb{R}^{V \times V \times V}$;
1.2  Compute $k$ orthonormal eigenpairs $(\sigma_t, \mu_t)$ of $M_2$;
1.3  Compute the whitening matrix $W = M\Sigma^{-\frac{1}{2}}$;
1.4  Compute $(W^T)^+ = M\Sigma^{\frac{1}{2}}$ ;
1.5  Compute a $k \times k \times k$ tensor $\widetilde{T} = M_3(W, W, W)$;
1.6  for $t = 1..k$ do
1.7      $\lambda^* \leftarrow 0$ ;                    // the largest eigenvalue so far
1.8      for $outIter = 1..N$ do
1.9          $v \leftarrow$ a random unit-form vector in $\mathbb{R}^k$;
1.10         for $innerIter = 1..n$ do $v \leftarrow \frac{\widetilde{T}(I,v,v)}{||\widetilde{T}(I,v,v)||}$ ;
1.11         if $\widetilde{T}(v,v,v) > \lambda^*$ then $(\lambda^*, v^*) \leftarrow (\widetilde{T}(v,v,v)$,v);
1.12     end
1.13     $\lambda_t = \frac{1}{(\lambda^*)^2}$, $v_t = \lambda_t (W^T)^+ v^*$;
1.14     $\widetilde{T} \leftarrow \widetilde{T} - \lambda^* v^* \otimes v^* \otimes v^*$ ;                    // deflation
1.15     $\alpha_t = \alpha_0 \lambda_t, \phi_t = v_t$;
1.16 end
1.17 return $(\alpha_t, \phi_t), t = 1, \ldots, k$
```

5 Scalable Tensor Orthogonal Decomposition

5.1 Scalability analysis of TOD

Although Algorithm 1 theoretically guarantees robust convergence, it is not scalable. In general, when we directly deal with large and dense 2nd or 3rd order tensors, computation cost is huge in both time and space, and this hinders the scalability of part 1 described in Section 4, where explicit computations for a matrix of size V^2 and a tensor of size V^3 are involved. In contrast, part 2 ($k \times k \times k$ tensor orthogonal decomposition) can be efficient in practice, because in most cases of LDA inference, only a small number of topics are desired. In total, the space complexity of Algorithm 1 is $\mathcal{O}(V^3)$ and the time complexity is $\mathcal{O}(V^3 k + L\hat{l}^2 + Nnk^4)$, where $\hat{l}$ is the maximum document length.

Anandkumar *et al.* [3] discusses a plausible way to reduce the memory cost. It suggests no explicit creation of the tensors M_3 and $\widetilde{T}$, but going through the document-word occurrence data for computing the power iteration update Line 1.10. This mitigates the space challenge of part 1, but gives away the efficiency of part 2 of Algorithm 1. One obvious disadvantage is that it needs to scan the whole corpus for Nnk times to execute Line 1.10. The space complexity is $\mathcal{O}(V^2)$ and the time complexity is $\mathcal{O}(V^2 k + LNnk)$.

We make key contributions to solving the challenge in a different approach. We avoid explicit creation of both tensor M_3 and M_2, but we do explicitly create $\widetilde{T}$ since it is memory efficient. It reduces the cost of part 1, and retains efficient power iteration updates as in part 2 of Algorithm 1. Utilizing the special structure of the tensors in our problem, we show that $\widetilde{T}$ can be created by scanning the corpus only twice, without incurring creations of any dense V^2 or V^3 tensors. One scan is needed for computing the whitening matrix W, and the other for computing the tensor product $M_3(W, W, W)$, as discussed in the following two sections.

5.2 Scalable computation of whitening matrix

To compute the whitening matrix, the straightforward approach by spectral decomposition of the dense matrix M_2 requires $\mathcal{O}(V^2 k)$ time and $\mathcal{O}(V^2)$ space. However, by observation of Eq. (4), we can decouple M_2 into matrix E_2 and $M_1 \otimes M_1$. Taking advantage of the *low rank* and *sparsity* of E_2, we can compute the spectral decomposition of M_2 in an efficient alternative procedure.

Low rank. We notice that M_1, E_2 and M_2 are in the same *column space* $\mathcal{S}$ spanned by k linearly independent vectors $\phi_t, t \in [k]$. Thus E_2 has a low rank.

Sparsity. Let vector $c_i \in \mathbb{R}^V$ be the counts of word 1 to V in document d_i. An empirical estimation of M_1 and E_2 is:

$$M_1 = \sum_{i=1}^{D} \frac{1}{l_i} c_i, E_2 = \sum_{i=1}^{D} \frac{1}{l_i(l_i-1)} [c_i \otimes c_i - diag(c_i)] \tag{7}$$

where $l_i = \sum_{x=1}^{V} c_{i,x}$ is the length of document d_i. The estimated M_1 and E_2 can be computed by one scan of the data. E_2 is sparse because many word pairs do not co-occur in the real documents.

Our alternative procedure performs two spectral decompositions, one on the sparse and low rank matrix E_2 and the other on a small size matrix.

1. Let $E_2 = U \Sigma_1 U^T$ be its spectral decomposition, where $U \in \mathbb{R}^{V \times k}$ is the matrix of k eigenvectors, and $\Sigma_1 \in \mathbb{R}^{k \times k}$ is the diagonal eigenvalue matrix. The k column vectors of U form an orthonormal basis of $\mathcal{S}$. M_1's representation in this basis is $M_1' = U^T M_1$. Now, M_2 can be written as:

$$M_2 = U[(\alpha_0 + 1)\Sigma_1 - \alpha_0 M_1' \otimes M_1']U^T = U M_2' U^T$$

2. A second spectral decomposition can be performed on $M_2' \in \mathbb{R}^{k \times k}$. Let the decomposition be $M_2' = U' \Sigma U'^T$. It follows that:

$$M_2 = U M_2' U^T = (UU')\Sigma(UU')^T$$

Let $M = UU'$. Now we effectively obtain the spectral decomposition of $M_2 = M \Sigma M^T$ without explicitly creating M_2. With this new procedure, we only need to store a sparse matrix E_2 with $m \ll V^2$ nonzero elements, and the time complexity is reduced to $\mathcal{O}(km + k^3) = \mathcal{O}(km)$.

5.3 Scalable product of M_3 and W

The straightforward computation of $\widetilde{T} = M_3(W, W, W)$ using explicit M_3 and W requires $\mathcal{O}(V^3)$ space and $\mathcal{O}(V^3 k + L\hat{l}^2)$ time, where $\hat{l}$ is the maximal document length. To solve this challenge, we utilize two *decomposing laws*:

i) $(v \otimes v \otimes v)(W, W, W) = (W^T v) \otimes (W^T v) \otimes (W^T v) = (W^T v)^{\otimes 3}$; and

ii) $(v \otimes B)(W, W, W) = (W^T v) \otimes B(W, W) = (W^T v) \otimes (W^T B W)$

where v is a vector and B is a matrix.

We break down M_3 as a summation of multiple tensors, such that the product between each tensor and W has a **decomposable** form as in the left hand side of one decomposing law. According to Eq. (2), M_3 is a linear combination of tensors E_3, U_1, U_2, U_3 and $M_1^{\otimes 3}$. We discuss how to efficiently compute the product between each part and W.

Compute $\mathbf{E_3(W, W, W)}$. E_3 can be estimated by averaging the frequency of all the 3-word triples in each document. Using the word count vector c_i we defined before, we have:

$$\begin{aligned} E_3 &= \frac{1}{D}\left[A_1 - A_2 - \Omega(A_2, 2, 1, 3) - \Omega(A_2, 2, 3, 1) + 2A_3\right] \\ A_1 &= \sum_{i=1}^{D} \rho_i c_i \otimes c_i \otimes c_i, A_2 = \sum_{i=1}^{D} \rho_i c_i \otimes diag(c_i), A_3 = \sum_{i=1}^{D} \rho_i tridiag(c_i) \end{aligned} \tag{8}$$

where $\rho_i = \frac{1}{l_i(l_i-1)(l_i-2)}$, $tridiag(v)$ is a tensor with vector v on its diagonal: $tridiag(v)_{i,i,i} = v_i$.

According to decomposing law i) and ii):

$$A_1(W, W, W) = \sum_{i=1}^{D} \rho_i (W^T c_i)^{\otimes 3} \tag{9}$$

$$A_2(W, W, W) = \sum_{i=1}^{D} \rho_i (W^T c_i) \otimes W^T diag(c_i) W \tag{10}$$

Let W_x^T be the x-th column of W^T. We have:

$$A_3(W, W, W) = \sum_{x=1}^{V} \sum_{i=1}^{D} \rho_i c_{i,x} (W_x^T)^{\otimes 3} \tag{11}$$

Using Eq. (9)-(11), we can compute $E_3(W, W, W)$ without explicit creation of E_3. The time complexity is $\mathcal{O}(Lk^2)$.

Compute $\mathbf{M_1^{\otimes 3}(W, W, W), U_i(W, W, W)}, i = 1, 2, 3$. Using the two decomposing laws, we can obtain:

$$(M_1 \otimes M_1 \otimes M_1)(W, W, W) = (W^T M_1)^{\otimes 3} \tag{12}$$

$$U_1(W, W, W) = W^T E_2 W \otimes W^T M_1 \tag{13}$$

Eq. (13) requires $O(k^2 m)$ time to compute, where m is the number of nonzero elements in E_2. We can further speed it up. By definition we have $W^T M_2 W = I$. Substituting M_2 with Eq. (4), we have:

$$W^T[(\alpha_0+1)E_2 - \alpha_0 M_1 \otimes M_1]W = I \tag{14}$$

$$\Rightarrow W^T E_2 W = \frac{1}{(\alpha_0+1)}[I + \alpha_0 (W^T M_1)^{\otimes 2}] \tag{15}$$

Plugging Eq. (15) into (13) further reduces the complexity of computing $U_1(W,W,W)$ to $\mathcal{O}(Vk + k^3)$. $U_2(W,W,W)$ and $U_3(W,W,W)$ can be obtained by permuting $U_1(W,W,W)$'s modes, in $\mathcal{O}(k^3)$ time.

Putting these together based on the distributive law, we can compute $\widetilde{T} = M_3(W,W,W)$ by one scan of the data:

$$\begin{aligned}\widetilde{T} = M_3(W,W,W) = \frac{(\alpha_0+1)(\alpha_0+2)}{2} E_3(W,W,W) \\ -\frac{\alpha_0(\alpha_0+1)}{2}[(U_1+U_2+U_3)(W,W,W)] + \alpha_0^2 (W^T M_1)^{\otimes 3}\end{aligned} \tag{16}$$

which requires $O(Lk^2 + Vk + k^3) = O(Lk^2)$ time.

```
Algorithm 2: Scalable Tensor Orthogonal Decomposition (STOD)
    Input: Corpus with $L$ tokens and vocabulary size $V$, number of topics $k$,
           number of outer and inner iterations $N, n$, $\alpha_0$
    Output: The model parameters $(\alpha_t, \phi_t), t = 1, \ldots, k$
2.1 First scan of data: Compute $M_1$ and $E_2$ according to Eq. (7);
2.2 Find $k$ largest orthonormal eigenpairs $(\sigma_t, \mu_t)$ of $E_2$;
2.3 $M_1' = UM_1$ ;                        // $U = [\mu_1, \ldots, \mu_k], \Sigma_1 = diag(\sigma_1, \ldots, \sigma_k)$
2.4 Compute spectral decomposition for $M_2' = (\alpha_0+1)\Sigma_1 - \alpha_0 M_1' \otimes M_1' = U' \Sigma U'^T$;
2.5 $M = UU', W = M\Sigma^{-\frac{1}{2}}, (W^T)^+ = M\Sigma^{\frac{1}{2}}$;
2.6 Second scan of data: Compute $\widetilde{T} = M_3(W,W,W)$ according to Eq. (16);
2.7 Perform power method Line 1.6 to 1.16 in Algorithm 1;
2.8 return $(\alpha_t, \phi_t), t = 1, \ldots, k$
```

5.4 Our final algorithm

Algorithm 2 outlines our scalable tensor orthogonal decomposition algorithm. Line 2.1 scans the data once to collect E_2, and Line 2.2–2.5 are asymptotically equivalent to Line 1.2–1.4. Line 2.6 uses a second scan of the data to compute $\widetilde{T}$, which is equivalent to Line 1.5 but much more efficient. The power method on $\widetilde{T}$ remains the same as in Algorithm 1.

In most applications, $V^3 \gg L, V^2 \gg m, V \gg \hat{l} > k$. We reduce the time complexity for constructing the small tensor $\widetilde{T} \in \mathbb{R}^{k \times k \times k}$ to $\mathcal{O}(Lk^2 + km)$, and the space complexity to $\mathcal{O}(m)$. The total time complexity for STOD is $\mathcal{O}(Lk^2 + km + Nnk^4)$. Comparing with TOD, STOD is superior in both space and time. The practical speedup is significant with orders of magnitude, as we will demonstrate in experiments.

6 Experiments

In this section we first introduce the methods used for comparison, then present evaluation on synthetic and real datasets respectively.

Methods for comparison. Our main contribution is the STOD algorithm. It accelerates the TOD algorithm, which has bounded error for LDA inference. We also compare STOD with one of the most popular LDA inference methods collapsed Gibbs sampling, although its error is not theoretically bounded. We do not include other maximum-likelihood based inference methods for LDA, e.g., collapsed variational Bayesian inference, because they mostly have a similar performance with collapsed Gibbs sampling, and no theoretical error bound either. We list the implementation details below:

- STOD – our Algorithm 2. Outer iteration # N and inner iteration # n are both set to 30. They are sufficient for our experiments. In fact, in most cases, the algorithm converges with $N = n = 10$. α_0 is fixed to be 1.
- TOD – A faster implementation of Algorithm 1 as we discussed in Section 5.1. It skips the tensor construction and computes the power iteration on the fly. Outer iteration # N and inner iteration # n are both set to 30. $\alpha_0 = 1$.
- Collapsed Gibbs sampling. We use a fast implementation by Griffiths and Steyvers [9]. The iteration # is set to 1500, following the common practice. From now on, we use ‘Gibbs’ or ‘Gibbs sampling’ for short.

For fair comparison, we do not use distributed computation for any of the methods. We conduct all the experiments on a single Linux server running MATLAB 2013a with Inter Xeon CPU E5-2680 2.80GHz and 256GB RAM.

6.1 Synthetic data

We use synthetic data to conduct controlled experiments with known topics and other parameters. With synthetic data we are able to evaluate the error of each method in recovering the known topics. We compare each method’s: i) topic recovery error; and ii) runtime.

The generative process of synthetic data simply follows LDA [6]. The length of each document is generated from a Poisson distribution, where the Poisson parameter λ, or the expected document length, is set to 100. The Dirichlet prior α of each document-level topic distribution θ_i is uniform: $\alpha_t = \frac{1}{k}, t \in [k]$; the Dirichlet prior β of each corpus-level topic-word distribution ϕ_t is also uniform: $\beta_x = \frac{200}{V}, x \in [V]$. The same Dirichlet prior is used for Gibbs sampling inference. We creat three controlled sets of pseudo corpora by varying the following parameters:

1. D, ranging from 5,000 to 500,000, with fixed $V = 10000, k = 50$.
2. V, ranging from 3,000 to 100,000, with fixed $D = 100,000, k = 50$.
3. k, ranging from 10 to 100, with fixed $D = 100,000, V = 10,000$.

Topic recovery error. We measure the topic recovery error in the following way. For each run of each algorithm, let $\tilde{\phi}_t, t \in [k]$ denote the corpus-level multinomial distributions inferred by the algorithm; and $\phi_t^*, t \in [k]$ the ground-truth generated from Dirichlet allocation. We compute all the k^2 L1 distances: $\|\tilde{\phi}_{t_1} - \phi_{t_2}^*\|_1, t_1, t_2 \in [k]$, and build a bipartite graph with negative L1 distances as edge weights. Then we use the Hungarian algorithm to compute a maximum matching between the inferred topics and the ground truth topics. Finally, we average the k L1 distances between matched pairs as the error for this run.

Figure 1a, 1c and 1e show the recovery errors of different methods. For each fixed triple of (D, V, k), we run each algorithm for 5 times, then plot the mean and standard deviation of the 5 recovery errors using an error bar. To put the numbers in context, we include a baseline 'uniform', which reflects the average distance of a uniform distribution over the vocabulary to every topic.

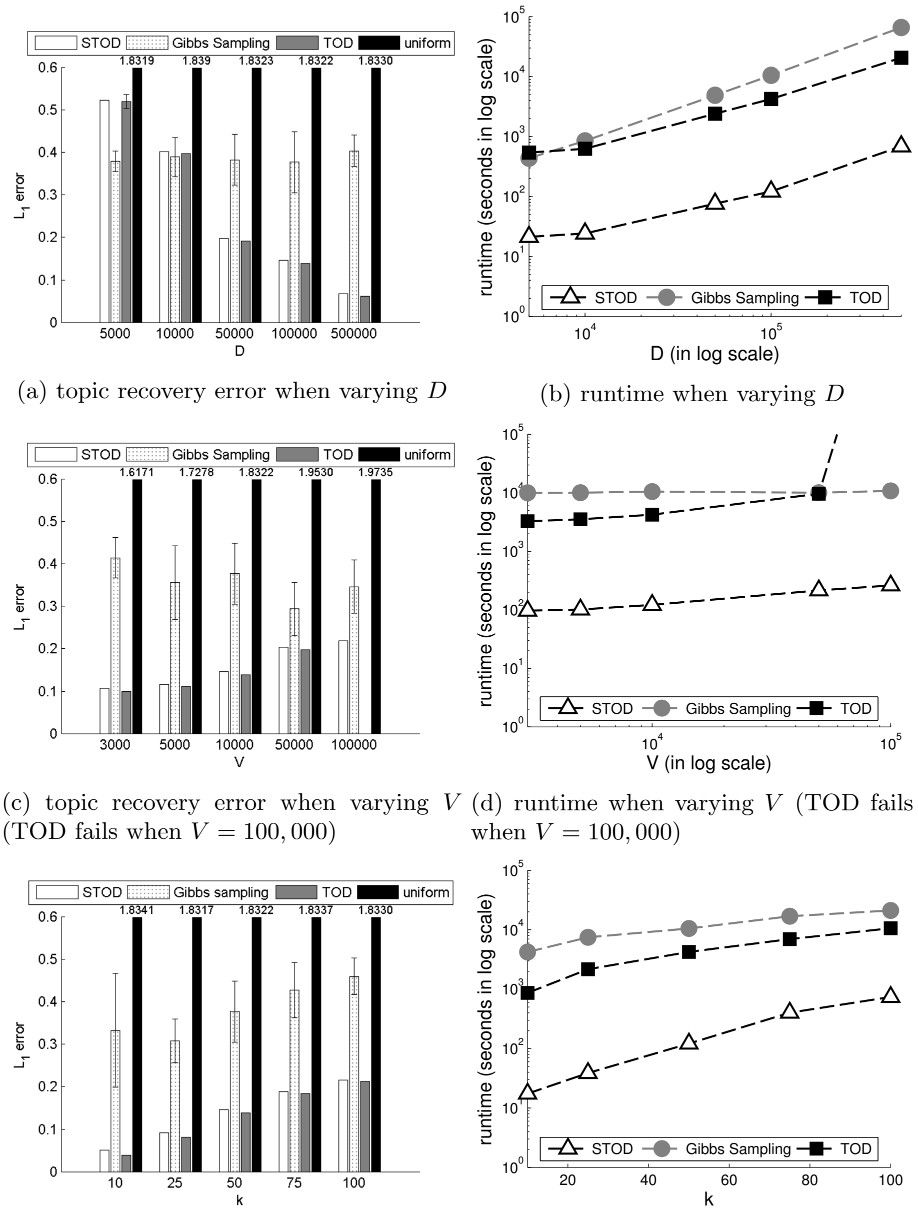
We observe that TOD and STOD have almost zero variance across multiple runs, due to the robustness of tensor decomposition. Gibbs sampling produces large variance comparing with the other two, which is a drawback most existing maximum likelihood based LDA inference algorithms suffer.

In general, the errors of STOD and TOD decrease when D increases or V, k decrease, *i.e.*, the sample size increases or the model complexity decreases. This is because the error of tensor orthogonal decomposition is bounded by the distance of empirical moments from theoretical moments. For Gibbs sampling, this trend is not as clear as the moment-based methods. It has no error bound of topic recovery.

In all these datasets, the moment-based methods TOD and STOD have almost equal errors. When the corpus size is sufficiently large ($D \geq 50,000$ in these datasets), the error is 37–85% lower than Gibbs sampling. This verifies that TOD has the state-of-the-art capability of topic recovery accuracy, and that our STOD algorithm preserves that capability.

Runtime. From Figure 1b,1d and 1f, we see a clear superiority in efficiency of our STOD algorithm in all the datasets. STOD is faster than TOD and Gibbs sampling by 1 to 3 orders of magnitude. While TOD is generally faster than Gibbs sampling, it consumes much larger memory, and fails to terminate when $V = 100,000$.

The runtime of STOD grows more slowly with respect to D than Gibbs sampling, because STOD only scans the corpus twice while Gibbs sampling iteratively passes through the corpus for thousands of times. The runtime of STOD grows more tenderly with respect to V than TOD, because the former does not even need to construct the dense tensor of size V^2. The runtime of STOD grows more rapidly with respect to k than Gibbs sampling and TOD, because it constructs a tensor of size k^3 explicitly. Therefore, the advantage of STOD is most prominent when the corpus size and vocabulary size are large, while the number of topics is small.



(a) topic recovery error when varying D

(b) runtime when varying D

(c) topic recovery error when varying V (TOD fails when $V = 100,000$)

(d) runtime when varying V (TOD fails when $V = 100,000$)

(e) topic recovery error when varying k

(f) runtime when varying k

Fig. 1: Performance study on synthetic data (lower values are better)

6.2 Real-world data

We use two real-world datasets to evaluate the performance of our algorithm in practice (we perform stemming to the favor of baseline methods, and remove stopwords in the corpus):

- TREC AP news: A TREC news dataset (1998). It contains 106K full articles, 170K unique words, and 19M tokens. After preprocessing, the size of vocabulary is 45,105.
- CS abstract: A dataset of computer science paper abstracts from Arnetminer[1]. The set has 529K papers, 186K unique words, and 39M tokens. After preprocessing, the size of vocabulary is 51,069.

Runtime. Table 1 shows the overall runtime in these datasets, in two scenarios. One scenario is that the data can be all loaded into memory, and the other scenario is that the data are too large to be loaded into memory. STOD is one to two orders of magnitude faster than the other methods in the first scenario, and two to three orders of magnitude faster in the second scenario. On the largest dataset it reduces the runtime of TOD/Gibbs sampling from 3 weeks/1.2 days to 9.6 minutes.

Table 2 shows the decomposed runtime for STOD and TOD. In both datasets, the most time consuming part for STOD is the spectral decomposition (Line 2.1–2.5) and tensor construction (Line 2.6). The news dataset has longer documents but fewer tokens than the CS dataset. As a result, the spectral decomposition in news dataset bears a larger fraction though the total runtime is shorter than in CS. The practical implementation of TOD does not create tensors but goes through the corpus many times to compute the power iteration on the fly, and that part accounts for the slow execution.

Table 1: Total runtime (in seconds) on real-world datasets (K=50)

method	*loaded into memory*			*not loaded into memory*		
dataset	STOD	TOD	Gibbs sampling	STOD	TOD	Gibbs sampling
news	**293**	6877	21641	**310**	768110	48999
CS	**541**	14439	47293	**577**	1661101	102136

Table 2: Decomposed runtime (%) on real-world datasets for STOD and TOD

method	STOD			TOD	
dataset	*spectral decomp*	*construct tensor*	*power iter*	*spectral decomp*	*power iter*
news	38.0	47.8	14.2	1.2	98.8
CS	11.1	80.7	8.3	1.2	98.8

Quality of inferred topics. Lack of gold standard is a well known challenge for unsupervised topic modeling methods. As such, people have proposed evaluation metrics without relying on labels. The conventional evaluation using the held-out

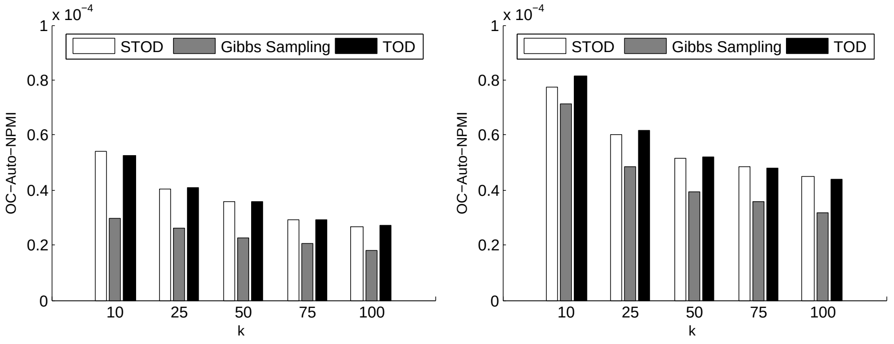
[1] `http://www.arnetminer.org`

perplexity of test data has been challenged [7, 17], and found to have negative correlation with human interpretability. According to the most recent work in topic model quality evaluation [14], there are two major approaches to measure the human interpretability: indirect approach with *word intrusion*, and direct approach with *observed coherence*. In this study we take the direct approach, and use the automated evaluation measure OC-Auto-NPMI in [14], which was reported to have above 0.9 Pearson correlation with human judgment.

The OC-Auto-NPMI measure for one topic is defined to be the average of normalized pointwise mutual information between every pair of the top-X words:

$$\text{OC-Auto-PMI}(t) = \frac{2}{X(X-1)} \sum_{j=2}^{X} \sum_{i=1}^{j-1} \log \frac{p(w_j, w_i)}{p(w_i)p(w_j)} \tag{17}$$

where $w_1, \ldots, w_X$ are the top-X words in topic t. Then the mean of the OC-Auto-NPMI measure for all the topics can be used to measure the quality of the inferred topics by an inference algorithm (the higher the better).



(a) Coherence on news

(b) Coherence on CS

Fig. 2: Quality of inferred topics on real-world data (the higher the better)

As shown in Figure 2, STOD and TOD again have close performance, and both outperform Gibbs sampling[2], by as much as 80% in the news dataset, and as much as 40% in the CS dataset. The moment-based methods not only have theoretical low error, but also good practical performance with real-world data.

Table 3 visualizes several example topics with top-ranked words from the TREC AP news dataset. Since STOD and TOD have identical results in this experiment, we only keep STOD in the table. We can see when k is set to 25, these four topics are interpretable in both methods, although a few top ranked words are less intuitive to interpret in Gibbs. For example, in topic 4 of Gibbs sampling, word 'oil' cannot be recognized as a part of *weather* topic, while most of the other words have a strong correlation with *weather*.

[2] We experimented with hyperparameter optimization for Gibbs sampling as well, and it does not affect the conclusion

Table 3: Example topics from a 25-topic run of STOD & Gibbs on news

topic 1: finance		**topic 2: politics**		**topic 3: law**		**topic 4: weather**	
STOD	*Gibbs*	*STOD*	*Gibbs*	*STOD*	*Gibbs*	*STOD*	*Gibbs*
dollar	cents	vote	city	court	court	fair	oil
yen	market	house	black	case	state	cloudy	fair
prices	stock	democratic	state	state	law	city	coast
late	trade	senate	democratic	judge	ruling	northern	state
trade	prices	republican	mayor	abortion	union	part	rain
close	dollar	bill	white	ruling	abortion	rain	texas
london	exchange	committee	campaign	law	strike	central	national
gold	higher	election	election	appeals	judge	north	northen
rate	futurers	members	republican	federal	case	coast	part
bid	lower	party	year	supreme	federal	south	north

7 Discussion

In this work, we propose a scalable moment-based inference algorithm STOD for latent Dirichlet allocation topic model. The algorithm is based on recent advancement of moment-based inference methods which have robust theoretical properties. STOD inherits the advantage of low error and high stability, while solving critical challenge in time and space efficiency. By leveraging the special structures of the 2nd order and 3rd order moments, we dramatically overhaul the standard computing procedure to scale up the algorithm. It renders the tensor orthogonal decomposition for LDA inference practical for the first time, with orders of magnitude faster speed.

As we observe in the experiments, both STOD and TOD require a certain amount of documents to estimate the precise empirical moments and recover the topics with low error. This is easy to satisfy in the setting of large-scale text corpora, such as the real-world datasets in our experiments. STOD is most promising when the corpus size is large, and when the number of topics is small. This makes it a desirable method to summarize a large corpus' topics in a hierarchical structure where every topic has a few number of subtopics, which is one of our ongoing study.

Although we do not compare with distributed or online inference mechanism for MCMC or variational Bayesian inference, we would like to point out that: i) STOD can be easily parallelized by employing distributed spectral decomposition method such as [13, 15], with theoretically guaranteed performance; and ii) STOD scans the data only twice, which is similar to online inference methods requiring only one pass of data, but STOD does not trade in inference accuracy. Besides parallelization, the advantage of STOD can be further fulfilled by adaptation to dynamic text collections, or more advanced spectral decomposition methods.

Acknowledgments. This work was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF–09–2–0053 (NS-CTA). Chi Wang was supported by a Microsoft Research PhD Fellowship.

Bibliography

[1] A. Ahmed, Q. Ho, C. H. Teo, J. Eisenstein, E. P. Xing, and A. J. Smola. Online inference for the infinite topic-cluster model: Storylines from streaming text. In *AISTATS*, 2011.

[2] A. Anandkumar, D. P. Foster, D. Hsu, S. Kakade, and Y.-K. Liu. A spectral algorithm for latent dirichlet allocation. In *NIPS*, 2012.

[3] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012.

[4] S. Arora, R. Ge, Y. Halpern, D. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu. A practical algorithm for topic modeling with provable guarantees. In *ICML*, 2013.

[5] S. Arora, R. Ge, and A. Moitra. Learning topic models–going beyond svd. In *FOCS*, 2012.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[7] J. Chang, J. Boyd-Graber, C. Wang, S. Gerrish, and D. M. Blei. Reading tea leaves: How humans interpret topic models. In *NIPS*, 2009.

[8] J. Foulds, L. Boyles, C. DuBois, P. Smyth, and M. Welling. Stochastic collapsed variational bayesian inference for latent dirichlet allocation. In *KDD*, 2013.

[9] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc. of the National Academy of Sciences of USA*, 101(Suppl 1):5228–5235, 2004.

[10] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

[11] M. Hoffman, D. M. Blei, and D. M. Mimno. Sparse stochastic inference for latent dirichlet allocation. In *ICML*, 2012.

[12] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1-2):177–196, Jan. 2001.

[13] D. Kempe and F. McSherry. A decentralized algorithm for spectral analysis. In *STOC*, 2004.

[14] J. H. Lau, D. Newman, and T. Baldwin. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *EACL*, 2014.

[15] K. J. Maschhoff and D. Sorensen. P_arpack: An efficient portable large scale eigenvalue package for distributed memory parallel architectures. In *Applied Parallel Computing Industrial Computation and Optimization*, pages 478–486. 1996.

[16] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10:1801–1828, 2009.

[17] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin. Automatic evaluation of topic coherence. In *NAACL-HLT*, 2010.

[18] I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *KDD*, 2008.

[19] D. Sontag and D. Roy. Complexity of inference in latent dirichlet allocation. In *NIPS*, 2011.

[20] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *KDD*, 2009.

[21] K. Zhai, J. Boyd-Graber, N. Asadi, and M. L. Alkhouja. Mr. lda: A flexible large scale topic modeling package using variational inference in mapreduce. In *WWW*, 2012.