RPLSA: A Novel Updating Scheme for Probabilistic Latent Semantic Analysis

N. Bassiou C. Kotropoulos

Department of Informatics, Aristotle University of Thessaloniki
Box 451 Thessaloniki 541 24, GREECE
phone: + 30 2310 998225, fax: + 30 2310 998225

Abstract

A novel updating method for Probabilistic Latent Semantic Analysis (PLSA), called Recursive PLSA (RPLSA), is proposed. The updating of conditional probabilities is derived from first principles for both the asymmetric and the symmetric PLSA formulations. The performance of RPLSA for both formulations is compared to that of the PLSA folding-in, the PLSA rerun from the breakpoint, and well-known LSA updating methods, such as the singular value decomposition (SVD) folding-in and the SVD-updating. The experimental results demonstrate that the RPLSA outperforms the other updating methods under study with respect to the maximization of the average log-likelihood and the minimization of the average absolute error between the probabilities estimated by the updating methods and those derived by applying the non-adaptive PLSA from scratch. A comparison in terms of CPU run time is conducted as well. Finally, in document clustering using the Adjusted Rand index, it is demonstrated that the clusters generated by

Email address: {nbassiou, costas}@aiia.csd.auth.gr (N. Bassiou C. Kotropoulos)

the RPLSA are: a) similar to those generated by the PLSA applied from scratch; b) closer to the ground truth than those created by the other PLSA or LSA updating methods.
Key words: PLSA - PLSA updating - document clustering - information retrieval - Adjusted Rand - Expectation Maximization

---

1. Introduction

The ease to access, process, and retrieve multimedia data is mainly attributed to various machine learning algorithms employing computationally efficient statistical methods to extract information from the data. Such methods include the *latent variable models* that aim at revealing a hidden structure within the high dimensional data in order not only to retain, but also to abstract the information content.

Among the main latent variable models, Probabilistic Latent Semantic Analysis (PLSA) [12, 13, 14], which stems from Latent Semantic Analysis (LSA) [8], manipulates huge amounts of data under a solid probabilistic framework, thus finding applications in modeling, classification, and retrieval of text, audio, images, and videos. However, the dynamic nature of data together with memory limitations impose the need for devising updating methods for LSA or PLSA, which are also frequently met as *on-line*, *incremental*, or *folding-in* methods in the literature.

Several methods have been presented for updating the LSA model, that is estimated by a truncated singular value decomposition (SVD), such as *recomputing* the SVD, *SVD folding-in*, *SVD-updating* [21, 2, 1], and *SVD folding-up* which alternates repeatedly between the SVD folding-in and the

SVD-updating in order to avoid the loss of orthogonality [23].

Similarly, methods for updating the PLSA model, which performs a probabilistic mixture decomposition by means of the Expectation-Maximization (EM) algorithm, have been proposed. In more detail, an incremental variant of the EM algorithm is adopted in the *PLSA folding-in* [5, 10, 4], while a modified EM scheme based on the Generalized Expectation Maximization is proposed in [25]. In *Incremental PLSA* [7], PLSA folding-in is used to fold-in new terms and documents in a four step procedure where a batch of new incoming documents are added and a batch of old documents are discarded. Instead of using a maximum likelihood estimator for folding-in, a maximum a posteriori estimator is employed in *Bayesian folding-in*, which uses a Dirichlet density kernel as prior [11]. An adaptive Bayesian PLSA framework that incorporates two new adaptation paradigms for PLSA, namely the *MAP Estimation for Corrective Training* and the *Quasi-Bayes Estimation for Incremental Learning*, is proposed in [6]. The aforementioned paradigms model the priors of the PLSA parameters by using Dirichlet densities as well.

In this paper, a novel method referred to as *Recursive PLSA (RPLSA)* is proposed for updating the PLSA model probabilities. The PLSA is studied within the widely spread document modeling framework, where the observed term and document frequencies are modeled by latent topics. The proposed RPLSA derives the updating equations for the PLSA model probabilities from first principles, when new documents are appended to an initial document collection by adding incrementally the words of any new document in the term-document matrix. Two different initialization schemes of the model probabilities for the newly added documents are also tested. The per-

formance of the proposed RPLSA is compared to that of PLSA folding-in [12] and the PLSA rerun from the breakpoint in terms of accuracy and speed. It is demonstrated that the proposed updating method outperforms the established updating methods under study with respect to the *minimum average absolute error* between the probabilities derived by the updating methods and those estimated by the original non-adaptive PLSA algorithm applied to the augmented document collection starting from scratch. Moreover, the proposed RPLSA method achieves a higher *average log-likelihood* value upon EM convergence than the PLSA folding-in. In addition, by measuring the average CPU run time for RPLSA, the PLSA folding-in, the PLSA rerun from the breakpoint, and the PLSA executed from scratch, it is shown that the PLSA rerun from the breakpoint is the most time consuming updating method, whereas the PLSA folding-in is less time consuming than the RPLSA in most cases. However, the excessive computational time of the RPLSA is compensated with its higher accuracy compared to the accuracy of PLSA folding-in. In a document clustering framework, it is demonstrated by using the *Adjusted Rand* index, that the RPLSA produces more pure and correct clusters than the PLSA folding-in and the PLSA rerun from the breakpoint do. The superiority of the RPLSA over the LSA and its associated updating methods (i.e., SVD folding-in and SVD updating) is also verified in the document clustering framework.

The outline of the paper is as follows. The PLSA is briefly discussed in Sect. 2. In Sect. 3, the traditional LSA/PLSA updating schemes which are employed for comparison purposes are summarized. The proposed updating algorithms are derived from first principles in Sect. 4. Experimental results

are demonstrated in Sect. 5, and conclusions are drawn in Sect. 6.

2. Probabilistic Latent Semantic Analysis (PLSA)

The PLSA stems from LSA. It defines a proper generative latent data model, the so called aspect model [15] and performs probabilistic mixture decomposition. The aspect model is a latent variable model for co-occurrence data, which associates an unobserved class variable z_k, $k = 1, 2, \ldots, K$ with each observation. For text processing, the observation is the occurrence of a word/term w_j, $j = 1, 2, \ldots, M$ in a document d_i, $i = 1, 2, \ldots, N$, while the unobserved class variable z_k usually represents the topic a document has generated from. The basic assumption underlying the aspect model is that all the observation pairs (d_i, w_j) are independent and identically distributed and furthermore they are conditionally independent given the respective latent class z_k. The data generation process can be described by the following scheme [14]: 1) select a document d_i with probability $P(d_i)$, 2) pick a latent topic z_k for the document with probability $P(z_k|d_i)$, and 3) generate a term w_j with probability $P(w_j|z_k)$. For the joint distribution of the word w_j in the document d_i generated by the latent topic z_k, the following identity holds:

$$P(d_i, w_j, z_k) = P(d_i)P(z_k|d_i)P(w_j|z_k). \tag{1}$$

2.1. Asymmetric Formulation

The joint distribution of the observed data is obtained by summing (1) over all possible realizations of z_k, i.e.,

$$P(d_i, w_j) = \sum_{k=1}^{K} P(d_i, w_j, z_k) = P(d_i) \underbrace{\sum_{k=1}^{K} P(z_k|d_i)P(w_j|z_k)}_{P(w_j|d_i)}. \tag{2}$$

As it can be seen from Eq. (2), the document-specific term distributions $P(w_j|d_i)$ are obtained by the convex combination of the K aspects/factors $P(w_j|z_k)$. This implies that the documents are not assigned to clusters, but they are characterized by a specific mixture of factors with weights $P(z_k|d_i)$. In order to determine $P(d_i)$, $P(z_k|d_i)$ and $P(w_j|z_k)$, the PLSA algorithm maximizes the log-likelihood function, i.e.,

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{j=1}^{M} n(d_i, w_j) \log P(d_i, w_j) \tag{3}$$

with respect to all the aforementioned probabilities by applying the EM algorithm [9]. In Eq. (3), $n(d_i, w_j)$ denotes the term frequency, (i.e., how many times w_j occurs in d_i) . The estimation of $P(d_i)$ can be carried out independently resulting in $P(d_i) = \frac{n(d_i)}{\sum_{i=1}^{N} n(d_i)}$, where $n(d_i) = \sum_{j=1}^{M} n(d_i, w_j)$. $P(z_k|d_i)$ and $P(w_j|z_k)$ are estimated by alternating between the two steps of the EM algorithm [14]:

Expectation step (E-step):

$$\hat{P}(z_k|d_i, w_j) = \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_{k'=1}^{K} P(w_j|z_{k'})P(z_{k'}|d_i)}. \tag{4}$$

Maximization step (M-step):

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j)\hat{P}(z_k|d_i, w_j)}{\sum_{i=1}^{N}\sum_{j'=1}^{M} n(d_i, w_{j'})\hat{P}(z_k|d_i, w_{j'})} \tag{5}$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j)\hat{P}(z_k|d_i, w_j)}{n(d_i)}. \tag{6}$$

By alternating Eq. (4) with Eqs. (5)-(6), a procedure that converges toward a local maximum of the log-likelihood results.

2.2. Symmetric Formulation

It is worth noting that by applying the Bayes' chain rule in order to invert the conditional probability $P(z_k|d_i)$, an *equivalent symmetric version* of the aspect model can be obtained [14]. As a result, Eq. (2) takes the form:

$$P(d_i, w_j) = \sum_{k=1}^{K} P(z_k)P(d_i|z_k)P(w_j|z_k). \tag{7}$$

By applying the Bayes formula taking into account Eq. (7), we arrive at the following E-step [12]:

$$\hat{P}(z_k|d_i, w_j) = \frac{P(z_k)P(d_i|z_k)P(w_j|z_k)}{\sum_{k'=1}^{K} P(z_{k'})P(d_i|z_{k'})P(w_j|z_{k'})} \tag{8}$$

while the maximization of the expected data log-likelihood given the posterior probabilities in Eq. (8) yields the M-step equations:

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j)\hat{P}(z_k|d_i, w_j)}{\sum_{i=1}^{N}\sum_{j'=1}^{M} n(d_i, w_{j'})\hat{P}(z_k|d_i, w_{j'})} \tag{9}$$

$$P(d_i|z_k) = \frac{\sum_{j=1}^{M} n(d_i, w_j)\hat{P}(z_k|d_i, w_j)}{\sum_{i'=1}^{N}\sum_{j=1}^{M} n(d_{i'}, w_j)\hat{P}(z_k|d_{i'}, w_j)} \tag{10}$$

$$P(z_k) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j)\hat{P}(z_k|d_i, w_j)}{R} \tag{11}$$

where $R = \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j)$.

A better insight into the two equivalent aspect models can be obtained by means of the graphical models [3]. The latent topic variable z_k acts as a bottleneck variable, since its cardinality is smaller than the number of documents N and the number of terms M in the collection, i.e., $K \ll \min\{N, M\}$.

The symmetric PLSA formulation can be rewritten in matrix notation as $P = \mathbf{U}_K\mathbf{S}_K\mathbf{V}_K^T$, where $\mathbf{U}_K$ is an $M \times K$ matrix with jk element $P(w_j|z_k)$,

$\mathbf{V}_K$ is an $N \times K$ matrix with ik element $P(d_i|z_k)$, $\mathbf{S}_K$ is a $K \times K$ diagonal matrix having as elements on its main diagonal $P(z_k)$, $k = 1, 2, \ldots, k$, and $\mathbf{P}$ is an $M \times N$ matrix with ji element $P(w_j, d_i)$. Such a decomposition looks like a truncated SVD employed within the LSA. Despite the just described resemblance, it should become clear that the LSA and the PLSA solve different optimization problems. Indeed, the LSA minimizes the ℓ_2 norm (i.e. the Frobenius norm) between the original-term document matrix and its best K-rank approximation, while the PLSA maximizes the likelihood function of multinomial sampling or equivalently minimizes the cross entropy or Kullback-Leibler divergence between the model and the empirical distribution. The superior modeling power of PLSA over LSA is attributed to this fundamental difference.

However, the just described resemblance of the PLSA to the LSA, has motivated research in updating the PLSA by resorting to SVD updating methods. The methods proposed in the literature for updating both the LSA and the PLSA model, which were tested in this paper, are described in Sect. 3 that follows.

3. LSA/PLSA Updating

Updating refers to the general process of adding new terms and/or documents to an existing LSA/PLSA model. Several methods were proposed for LSA updating, *recomputing* the SVD, *SVD folding-in*, *SVD-updating* [21, 26, 1] and *SVD folding-up* [23]. The method of *recomputing* the SVD is not actually an updating method, since it performs the SVD to the augmented term-document matrix [2, 21]. *SVD folding-in* of new terms/documents is

the simplest approach, in which the new term/document vectors are projected to the existing latent space. *SVD-updating* is performed by exploiting the reduced QR decomposition of the augmented term/document matrix [26] or by means of suitable Cholesky factorizations [1]. In contrast to the *SVD-updating*, *SVD folding-in* looses the orthonormality of the right or/and left singular vectors, when new documents or/and terms are added [2].

For PLSA updating, the *PLSA folding-in* [5, 20] has been tested, which is based on an incremental variant of the EM algorithm discussed in [20]. The online EM algorithm in [20] keeps $P(w|z)$ fixed and iterates between $P(z|d,w)$ and $P(z|d)$. This scheme was further simplified for topic detection in [10]. Usually a small number of iterations are needed for the EM to converge.

In this paper, PLSA folding-in, SVD folding-in, and SVD updating were compared to the proposed *Recursive PLSA* described in the next section.

4. Recursive Probabilistic Latent Semantic Analysis (RPLSA)

A novel method of updating the PLSA model parameters (i.e., $P(w_j|z_k)$ and $P(z_k|d_i)$ for the asymmetric model and $P(w_j|z_k)$, $P(d_i|z_k)$, and $P(z_k)$ for the symmetric model) is derived from first principles next. The proposed method is referred to as *Recursive Probabilistic Latent Semantic Analysis (RPLSA)*.

In the analysis, the simplest case of adding a new document with just one word is treated in detail first and generalizations follow. Assume that a new document d_{N+1} is added to an existing $M \times N$ term-document matrix at the end of the lth iteration of the EM algorithm. Suppose that the document

is pivotal. That is, it contains just one word appearing α times, which is the first word of the vocabulary without any loss of generality. Therefore, it holds that

$$n(w_j, d_{N+1}) = \begin{cases} \alpha & \text{if } j = 1 \\ 0 & \text{if } j = 2, 3, \ldots, M. \end{cases} \tag{12}$$

Let us refer to the $M \times 1$ vector $d_{N+1} = [\alpha_1, 0, \ldots, 0]^T$ as the new incoming pivotal document to be appended to the existing $M \times N$ term-document matrix. Next, concepts or model parameters for the symmetric formulation will appear inside parentheses. A new column (row) has to be appended to the $K \times N$ ($N \times K$) matrix holding the probabilities $P(z_k|d_i)$ ($P(d_i|z_k)$). To achieve this, two initialization schemes have been proposed. In *stdUniform* initialization, the elements $P(z_k|d_{N+1})_l$ ($P(d_{N+1}|z_k)_l$) of the appended column (row) are assumed to be uniformly distributed numbers in $[0, 1]$, while in *wordtopics* initialization they are assumed to be equal to the conditional probability of the word w_1 appearing in the new document d_{N+1} given each topic. That is, $P(z_k|d_{N+1})_l = P(w_1|z_k)_l$ or $P(d_{N+1}|z_k)_l = P(w_1|z_k)_l$, $k = 1, 2, \ldots, K$ for the asymmetric formulation and the symmetric one, respectively. In either case, a normalization is necessary so that $\sum_{k=1}^{K} P(z_k|d_{N+1})_l = 1$, $k = 1, 2, \ldots, K$ for the asymmetric formulation and $\sum_{i=1}^{N+1} P(d_i|z_k)_l = 1$, $k = 1, 2, \ldots, K$ for the symmetric one. For simplicity reasons, we also make the assumption that the addition of the new document alters neither the number of topics K nor the vocabulary. The objective is to derive the equations for the $(l+1)$th iteration of the EM algorithm.

4.1. Asymmetric formulation

Initially, let us focus on the computations that take place by proceeding from the $l-$th iteration to the iteration $l+1$ of the EM algorithm, when no document is added.

The E-step for iteration $l+1$ is given by

$$\hat{P}(z_k|d_i,w_j)_{l+1} = \frac{P(w_j|z_k)_l P(z_k|d_i)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l}. \tag{13}$$

After the substitution of Eq. (13), the M-step equations (5) and (6) take the form:

$$\begin{aligned} P(w_j|z_k)_{l+1} &= \frac{\sum_{i=1}^{N} n(d_i,w_j)\hat{P}(z_k|d_i,w_j)_{l+1}}{\sum_{i=1}^{N}\sum_{j'=1}^{M} n(d_i,w_{j'})\hat{P}(z_k|d_i,w_{j'})_{l+1}} \\ &= \frac{P(w_j|z_k)_l \sum_{i=1}^{N} \frac{n(d_i,w_j)P(z_k|d_i)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l}}{\sum_{j'=1}^{M} P(w_{j'}|z_k)_l \left[\sum_{i=1}^{N} \frac{n(d_i,w_{j'})P(z_k|d_i)_l}{\sum_{k'=1}^{K} P(w_{j'}|z_{k'})_l P(z_{k'}|d_i)_l}\right]} \end{aligned} \tag{14}$$

$$\begin{aligned} P(z_k|d_i)_{l+1} &= \frac{\sum_{j=1}^{M} n(d_i,w_j)\hat{P}(z_k|d_i,w_j)_{l+1}}{n(d_i)} \\ &= \frac{P(z_k|d_i)_l \sum_{j=1}^{M} \frac{n(d_i,w_j)P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l}}{n(d_i)}. \end{aligned} \tag{15}$$

To simplify notation, Eqs. (14) and (15) are rewritten as follows:

$$P(w_j|z_k)_{l+1} = \frac{P_1(w_j|z_k)_{l+1}}{\sum_{j'=1}^{M} P_1(w_{j'}|z_k)_{l+1}}, \text{ where} \tag{16}$$

$$P_1(w_j|z_k)_{l+1} = P(w_j|z_k)_l \sum_{i=1}^{N} \frac{n(d_i,w_j)P(z_k|d_i)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l} \tag{17}$$

and

$$P(z_k|d_i)_{l+1} = \frac{P_2(z_k|d_i)_{l+1}}{n(d_i)}, \text{ where} \tag{18}$$

$$P_2(z_k|d_i)_{l+1} = P(z_k|d_i)_l \sum_{j=1}^{M} \frac{n(d_i,w_j)P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l}. \tag{19}$$

The pivotal document $d_{N+1} = [\alpha, 0, \ldots, 0]^T$ is added in the collection at the end of the lth iteration. Therefore, Eqs. (16)-(19) have to be re-estimated. Eq. (17) takes the form

$$\begin{aligned} P_1^+(w_j|z_k)_{l+1} &= P_1(w_j|z_k)_{l+1} + \frac{n(d_{N+1}, w_j)P(w_j|z_k)_l P(z_k|d_{N+1})_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_{N+1})_l} \\ &= \begin{cases} P_1(w_j|z_k)_{l+1}, & \text{if } j \neq 1 \\ P_1(w_1|z_k)_{l+1} + \frac{\alpha P(w_1|z_k)_l P(z_k|d_{N+1})_l}{\sum_{k'=1}^{K} P(w_1|z_{k'})_l P(z_{k'}|d_{N+1})_l}, & \text{if } j = 1 \end{cases} \end{aligned} \tag{20}$$

where the notation $P^+(\,)$ denotes the probability estimate after the insertion of the new document d_{N+1}. Eq. (16) with the help of Eq. (20) is re-written as

$$\begin{aligned} P^+(w_j|z_k)_{l+1} &= \frac{P_1^+(w_j|z_k)_{l+1}}{\sum_{j'=1}^{M} P_1^+(w_{j'}|z_k)_{l+1}} = \frac{P_1^+(w_j|z_k)_{l+1}}{P_1^+(w_1|z_k)_{l+1} + \sum_{j'=2}^{M} P_1^+(w_{j'}|z_k)_{l+1}} \\ &= \begin{cases} \frac{P_1(w_j|z_k)_{l+1}}{A_{l+1}^+}, & \text{if } j \neq 1 \\ \frac{P_1^+(w_1|z_k)_{l+1}}{A_{l+1}^+}, & \text{if } j = 1 \end{cases} \end{aligned} \tag{21}$$

where

$$A_{l+1}^+ = \sum_{j'=1}^{M} P_1(w_{j'}|z_k)_{l+1} + P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k)_{l+1}. \tag{22}$$

Similarly, Eq. (19) is written as

$$\begin{aligned} P_2^+(z_k|d_i)_{l+1} &= P(z_k|d_i)_l \sum_{j=1}^{M} \frac{n(d_i, w_j)P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l} \\ &= \begin{cases} P_2(z_k|d_i)_{l+1}, & \text{if } i = 1, 2, \ldots, N \\ \frac{\alpha P(w_1|z_k)_l P(z_k|d_{N+1})_l}{\sum_{k'=1}^{K} P(w_1|z_{k'})_l P(z_{k'}|d_{N+1})_l}, & \text{if } i = N+1 \end{cases} \end{aligned} \tag{23}$$

By rewriting the second branch of Eq. (23) with the help of Eq. (20), we obtain

$$P_2^+(z_k|d_i)_{l+1} = P(z_k|d_i)_l \sum_{j=1}^{M} \frac{n(d_i, w_j)P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(w_j|z_{k'})_l P(z_{k'}|d_i)_l}$$

$$= \begin{cases} P_2(z_k|d_i)_{l+1}, & \text{if } i = 1, 2, \ldots, N \\ P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k)_{l+1}, & \text{if } i = N+1 \end{cases} \quad (24)$$

Finally, Eq. (18) with the help of Eq. (24) takes the form:

$$P^+(z_k|d_i)_{l+1} = \frac{P_2^+(z_k|d_i)_{l+1}}{n(d_i)} \quad (25)$$

$$= \begin{cases} \frac{P_2(z_k|d_i)_{l+1}}{n(d_i)} = P(z_k|d_i)_{l+1}, & \text{if } i = 1, 2, \ldots, N \\ \frac{P_2^+(z_k|d_i)_{l+1}}{n(d_{N+1})} = \frac{P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k)_{l+1}}{\alpha}, & \text{if } i = N+1. \end{cases}$$

4.2. Symmetric formulation

Following similar lines to Sect. 4.1, let us begin with the E-step and M-step of the EM algorithm, when we proceed from iteration l to $l+1$ prior to the addition of a new document in the collection. The E-step for iteration $l+1$ is given by

$$\hat{P}(z_k|d_i, w_j)_{l+1} = \frac{P(z_k)_l P(d_i|z_k)_l P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(z_{k'})_l P(d_i|z_{k'})_l P(w_j|z_{k'})_l} \quad (26)$$

while the M-step equations (9)-(11) after the substitution of Eq. (26) take the following simplified form:

$$P(w_j|z_k)_{l+1} = \frac{P_1(w_j|z_k)_{l+1}}{\sum_{j'=1}^{M} P_1(w_{j'}|z_k)_{l+1}} \quad (27)$$

$$P(d_i|z_k)_{l+1} = \frac{P_2(d_i|z_k)_{l+1}}{\sum_{i'=1}^{N} P_2(d_{i'}|z_k)_{l+1}} \quad (28)$$

$$P(z_k)_{l+1} = \frac{1}{R_l} \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{n(d_i, w_j) P(z_k)_l P(d_i|z_k)_l P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(z_{k'})_l P(d_i|z_{k'})_l P(w_j|z_{k'})_l}, \quad (29)$$

where

$$P_1(w_j|z_k)_{l+1} = P(w_j|z_k)_l \left[\sum_{i=1}^{N} \frac{n(d_i, w_j) P(d_i|z_k)_l}{\sum_{k'=1}^{K} P(z_{k'})_l P(w_j|z_{k'})_l P(d_i|z_{k'})_l} \right] P(z_k)_l$$

(30)

$$P_2(d_i|z_k)_{l+1} = P(d_i|z_k)_l \left[\sum_{j=1}^{M} \frac{n(d_i, w_j) P(w_j|z_k)_l}{\sum_{k'=1}^{K} P(z_{k'})_l P(w_j|z_{k'})_l P(d_i|z_{k'})_l} \right] P(z_k)_l. \quad (31)$$

When the new pivotal document $d_{N+1} = [\alpha, 0, \ldots, 0]^T$ is added at the end of the lth iteration, Eqs. (27)-(31), are reformulated as follows:

$$P_1^+(w_j|z_k)_{l+1} = \begin{cases} P_1(w_j|z_k)_{l+1}, & \text{if } j \neq 1 \\ P_1(w_1|z_k)_{l+1} + \frac{\alpha P(z_k)_l P(w_1|z_k)_l P(d_{N+1}|z_k)_l}{\sum_{k'=1}^{K} P(z_k)_{l'} P(w_1|z_{k'})_l P(d_{N+1}|z_{k'})_l}, & \text{if } j = 1 \end{cases} \quad (32)$$

$$P^+(w_j|z_k)_{l+1} = \begin{cases} \frac{P_1(w_j|z_k)_{l+1}}{A_{l+1}^+}, & \text{if } j \neq 1 \\ \frac{P_1^+(w_1|z_k)_{l+1}}{A_{l+1}^+}, & \text{if } j = 1, \end{cases} \quad (33)$$

where A_{l+1}^+ is given by Eq. (22). Let

$$P_2^+(d_i|z_k)_{l+1} = \begin{cases} P_2(d_i|z_k)_{l+1}, & \text{if } i = 1, 2, \ldots, N \\ P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k), & \text{if } i = N+1 \end{cases} \quad (34)$$

and

$$\begin{aligned} B_{l+1}^+ &= \sum_{i'=1}^{N} P_2(d_{i'}|z_k)_{l+1} + P_2^+(d_{N+1}|z_k)_{l+1} \\ &= \sum_{i'=1}^{N} P_2(d_{i'}|z_k)_{l+1} + P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k)_{l+1}. \end{aligned} \quad (35)$$

Then, we have:

$$P^+(d_i|z_k)_{l+1} = \begin{cases} \frac{P_2(d_i|z_k)_{l+1}}{B_{l+1}^+}, & \text{if } i = 1, 2, \ldots, N \\ \frac{P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k)}{B_{l+1}^+}, & \text{if } i = N+1. \end{cases} \quad (36)$$

Finally, we proceed to updating $P(z_k)_{l+1}$ as follows:

$$P^+(z_k)_{l+1} = \frac{1}{R_{l+1}} \left[R_l P(z_k)_{l+1} + \frac{n(d_{N+1}, w_1) P(z_k)_l P(d_{N+1}|z_k)_l P(w_1|z_k)_l}{\sum_{k'=1}^{K} P(z_{k'})_l P(d_{N+1}|z_{k'})_l P(w_1|z_{k'})_l} \right]$$

$$= \frac{1}{R_{l+1}} \left[R_l P(z_k)_{l+1} + P_1^+(w_1|z_k)_{l+1} - P_1(w_1|z_k) \right] \quad (37)$$

where

$$R_{l+1} = \sum_{i=1}^{N+1} \sum_{j=1}^{M} n(d_i, w_j) = \sum_{i=1}^{N} \sum_{j=1}^{M} n(d_i, w_j) + n(d_{N+1}, w_j) = R_l + a. \quad (38)$$

Obviously, the just derived updating equations for the asymmetric and the symmetric PLSA model parameters can be extended for a document with more than one terms, if we assume that every time we deal with a pivotal document having just one word and we incrementally append as many pivotal documents as the terms of the new incoming document are. Moreover, additional recursions can be executed in order to process more than one documents. Needless to say that for the asymmetric formulation one may alternate between the E-step in Eq. (13) and the M-step in Eqs. (16) - (19) whenever no additions are made, and switch to the M-step in Eqs. (20) - (22) and in Eq. (25) whenever additions occur. Similar remarks can be made for the symmetric formulation as well.

5. Experimental results

5.1. Objectives

The comparison of the PLSA updating methods under study (that is the proposed RPLSA, PLSA folding-in, and PLSA rerun from the breakpoint) in correctly estimating the PLSA probabilities for every incoming document when the PLSA is executed from scratch is the first objective of the experimental evaluation. To achieve this, the average absolute error between the model probabilities estimated by the PLSA updating methods and those estimated by the PLSA executed from scratch was measured. Additionally,

the average log-likelihood at the limit points of the EM algorithm was also examined to test whether the convergence of the PLSA updating methods under study was close to the convergence of the original PLSA algorithm, when executed from scratch.

The *accuracy* alone, reflected in the average absolute error and the average log-likelihood, however, is insufficient to assess the performance of the updating methods, since the *time complexity* consists a second performance factor of great importance. Therefore, the average CPU run time per added document was also reported.

Third, the PLSA updating methods are assessed in *document clustering*. That is, RPLSA, PLSA folding-in, and PLSA rerun from the breakpoint were tested with respect to the Adjusted Rand cluster validity index, that measures the similarity between the resulted document clusters and the ground truth document clustering [17]. In this framework, a comparison was also conducted between the RPLSA and the LSA updating methods, namely the SVD folding-in, and the SVD updating, outlined in Sect. 3.

It is finally worth noting that the aforementioned evaluations were conducted for both initialization methods (i.e., stdUniform and wordtopics) and for the asymmetric and symmetric PLSA model formulation.

5.2. Datasets - Parameters

The experiments were conducted on the *20-Newsgroups* corpus [18]. Four datasets related to the main corpus topics (comp, talk, sci, rec) were created consisting of either 5 or 4 subtopics. All the documents were preprocessed using the Bow toolkit [19] in order to have their tags removed and their words stemmed. For the stemming, the Porter stemmer was used [22]. The

term-document matrix for each dataset was built by measuring the word frequencies of the 500 words with the highest information gain, which belong to 500 documents randomly selected from each topic. In Table 1, the main features of each dataset are summarized.

Table 1: Extracted datasets.

Dataset	*Topics* (K)	*Dataset Size* (*words* $\times$ *documents*)
comp_500	5	500×2448
rec_500	4	500×1973
sci_500	4	500×1957
talk_500	4	500×1962

The algorithms implemented depend on the following parameters, which have to be set:

1. The number of latent topics K that was set according to Table 1 for each dataset.
2. The criterion value ϵ used to determine the convergence of the EM algorithm. The convergence criterion that was used expresses the relative log-likelihood change between two successive $(l-1,\, l)$ EM-steps, i.e.,
$$\left| \frac{[\mathcal{L}]^l - [\mathcal{L}]^{l-1}}{[\mathcal{L}]^{l-1}} \right| \geq \epsilon \tag{39}$$
Experiments were run for different values of ϵ. A typical value is 10^{-8}.
3. The probability distribution for the column and row appended respectively to matrices $P(z_k|d_i)$ (asymmetric formulation) and $P(d_i|z_k)$ (symmetric formulation) each time a new document is appended to the initial collection. Two different initializations were tested:

- *stdUniform*: all the entries of row (column) were set to the value $1/K$,
- *wordtopics*: all the entries of row (column) were set equal to the conditional probability of each word appearing in the newly added document given each of the K topics estimated in the previous step, as described at the beginning of Sect. 4.

5.3. Methods and procedures

A four-fold cross validation was performed over each dataset in order to define different subsets for initial batch PLSA training and incremental PLSA training subsets of the dataset. That is, from each dataset, approximately 75% of documents were selected to build the subset used for the initial training of the PLSA model. The remaining documents were retained to be incrementally added to the initial subset dataset, one by one.

The experimental procedure for every subset of the dataset consisted of the following steps: Initially, the PLSA model parameters were initialized with numbers uniformly distributed in $(0, 1)$ and the PLSA algorithm was executed for all the documents in the training subset. Next, every document from the second subset was appended to the training document subset, one at a time and the PLSA algorithm was executed from scratch for the augmented set, while RPLSA, PLSA folding-in, and PLSA rerun from the breakpoint update the model parameters, which had been estimated up to the point, before the addition of the new document.

The aforementioned experimental procedure was also set up for the LSA updating methods, SVD folding-in and SVD updating, where the best K rank approximation of the term-document matrix for K equal to the number

of topics in each dataset is sought, as is explained in Sect. 5.2. The values of K used for each dataset are given in Table 1.

5.4. Results

5.4.1. Accuracy of the model parameter updating

The RPLSA, the PLSA folding-in, and the PLSA rerun from the breakpoint for both the asymmetric and the symmetric formulations were compared to the PLSA algorithm computed from scratch for each document from the augmented document dataset. This was done, by averaging the absolute difference between the probabilities $P(w_j|z_k)$ and $P(z_k|d_i)$ derived by the PLSA and the same probabilities estimated by the RPLSA, the PLSA folding-in (PLSA fold.), and the PLSA rerun from the breakpoint (PLSA brk.) over the K latent variables in each dataset, after the addition of a new document from the second dataset. The probability $P(z_k|d_i)$ for the symmetric formulation was estimated by using the Bayes chain rule. The results obtained were further averaged across all appended documents. The above procedure was repeated for the two initialization methods. The mean and the standard deviation across the four folds of the average absolute error for the asymmetric and the symmetric formulations are summarized in Tables 2 and 3, respectively.

As can be easily be seen from Tables 2 and 3, the proposed RPLSA updating method yields on average model parameters closer to those estimated by the PLSA applied from scratch to the augmented term-document matrix than the PLSA folding-in or the PLSA rerun from the breakpoint does. More precisely, the RPLSA is shown to yield on average the lowest error for the conditional probability of the latent topics given the documents $P(z_k|d_i)$

Table 2: Mean and standard deviation across the 4 folds of the average absolute error between the probability $P(z_k|d_i)$ estimated by the asymmetric updating methods under study and that estimated by the PLSA executed from scratch.

Dataset	*stdUniform*					
	RPLSA		*PLSA brk.*		*PLSA fold.*	
	mean	*std*	*mean*	*std*	*mean*	*std*
comp_500	0.1583	0.0522	0.1632	0.0542	0.2114	0.0617
rec_500	0.1572	0.0701	0.1947	0.0830	0.2448	0.1167
sci_500	0.1157	0.0307	0.1280	0.0336	0.2392	0.1040
talk_500	0.0618	0.0193	0.0828	0.0178	0.1048	0.0331

Dataset	*wordtopics*					
	RPLSA		*PLSA brk.*		*PLSA fold.*	
	mean	*std*	*mean*	*std*	*mean*	*std*
comp_500	0.1381	0.0476	0.1879	0.0562	0.1705	0.0587
rec_500	0.1232	0.0436	0.1600	0.0574	0.1864	0.0676
sci_500	0.1112	0.0564	0.1196	0.0642	0.1568	0.0654
talk_500	0.0787	0.0500	0.0802	0.0485	0.0904	0.0431

Table 3: Mean and standard deviation across the 4 folds of the average absolute error between the probability $P(z_k|d_i)$ estimated by the symmetric updating methods under study and that estimated by the PLSA executed from scratch.

Dataset	*stdUniform*					
	RPLSA		*PLSA brk.*		*PLSA fold.*	
	mean	*std*	*mean*	*std*	*mean*	*std*
comp_500	0.2040	0.0387	0.2066	0.0335	0.2326	0.0458
rec_500	0.1850	0.0281	0.2064	0.0360	0.2081	0.0311
sci_500	0.1723	0.0626	0.1958	0.0657	0.1932	0.0555
talk_500	0.0999	0.0580	0.1006	0.0582	0.1019	0.0561

Dataset	*wordtopics*					
	RPLSA		*PLSA brk.*		*PLSA fold.*	
	mean	*std*	*mean*	*std*	*mean*	*std*
comp_500	0.1884	0.0665	0.2053	0.0785	0.2289	0.0766
rec_500	0.1336	0.0337	0.1842	0.0625	0.1928	0.0773
sci_500	0.1395	0.0376	0.1591	0.0530	0.1866	0.0639
talk_500	0.0858	0.0281	0.1134	0.0786	0.1174	0.0531

for both the asymmetric and the symmetric formulations than all the other methods under study. In particular,

- For the asymmetric formulation, there is a relative decrease ranging between 2% and 58% in the average absolute error of $P(z_k|d_i)$ estimated by the RPLSA and the $P(z_k|d_i)$ estimated by the PLSA executed from scratch compared to the same error, when either the PLSA rerun from the breakpoint or the PLSA folding-in replaces the RPLSA.

- For the symmetric formulation, there is a relative decrease ranging between 1.3% and 39% in the average absolute error of $P(z_k|d_i)$ estimated by the RPLSA and $P(z_k|d_i)$ estimated by the PLSA executed from scratch compared to the same error when either the PLSA rerun from the breakpoint or the PLSA folding-in replaces the RPLSA.

In all cases, the standard deviation of the average absolute error across the folds is much smaller than the mean value (at least twice smaller for most of the datasets, initializations, and updating methods) in Tables 2 and 3, supporting the statistical significance of the improvements. Further tests by performing a paired Wilcoxon test [24] between the updating methods under study at the significance level $p = 0.05$ have validated the just mentioned claim. In all cases, RPLSA is superior than the updating methods under study, since all the values of the paired Wilcoxon test are found to be less than $p = 0.05$. More precisely, all Wilcoxon test statistic values are found equal to 0, except those values estimated between the RPLSA and the PLSA rerun from the breakpoint. In particular, in the asymmetric formulation when using the stdUniform initialization to the comp_500

and talk_500 datasets the corresponding Wilcoxon test statistic values are 0.0107 and 0.0455, respectively. When the wordtopics initialization is employed to the talk_500 dataset the Wilcoxon test statistic value is found to be 0.0167. Similarly, for the symmetric formulation, the Wilcoxon test between the RPLSA and the PLSA rerun from the breakpoint yields a test statistic value equal to 0.0002 when the stdUniform initialization was used in the comp_500 dataset. By repeating the aforementioned test in talk_500 dataset, the test statistic was measured to be 0.0116. When wordtopics initialization was employed in sci_500 dataset, the Wilcoxon test between the RPLSA and the PLSA rerun from the breakpoint yields a test statistic value equal to 0.0309. The just reported values of the paired Wilcoxon test provide complementary evidence, when the mean value and the standard deviation in Tables 2 and 3 do not differ significantly (e.g., the entry for talk_500 and wordtopics initialization in Table 2).

The errors for the conditional probability of the words given the latent topics, $P(w_j|z_k)$, are not included, because they are extremely low for both models and the differences between the updating methods are negligible. This can be attributed to 1) the assumption that all the documents added do not have any out-of-vocabulary words and 2) the fact that PLSA folding-in leaves the conditional probability of words given the latent topics unchanged, while the RPLSA and the PLSA rerun from the breakpoint make only slight corrections to the aforementioned probability.

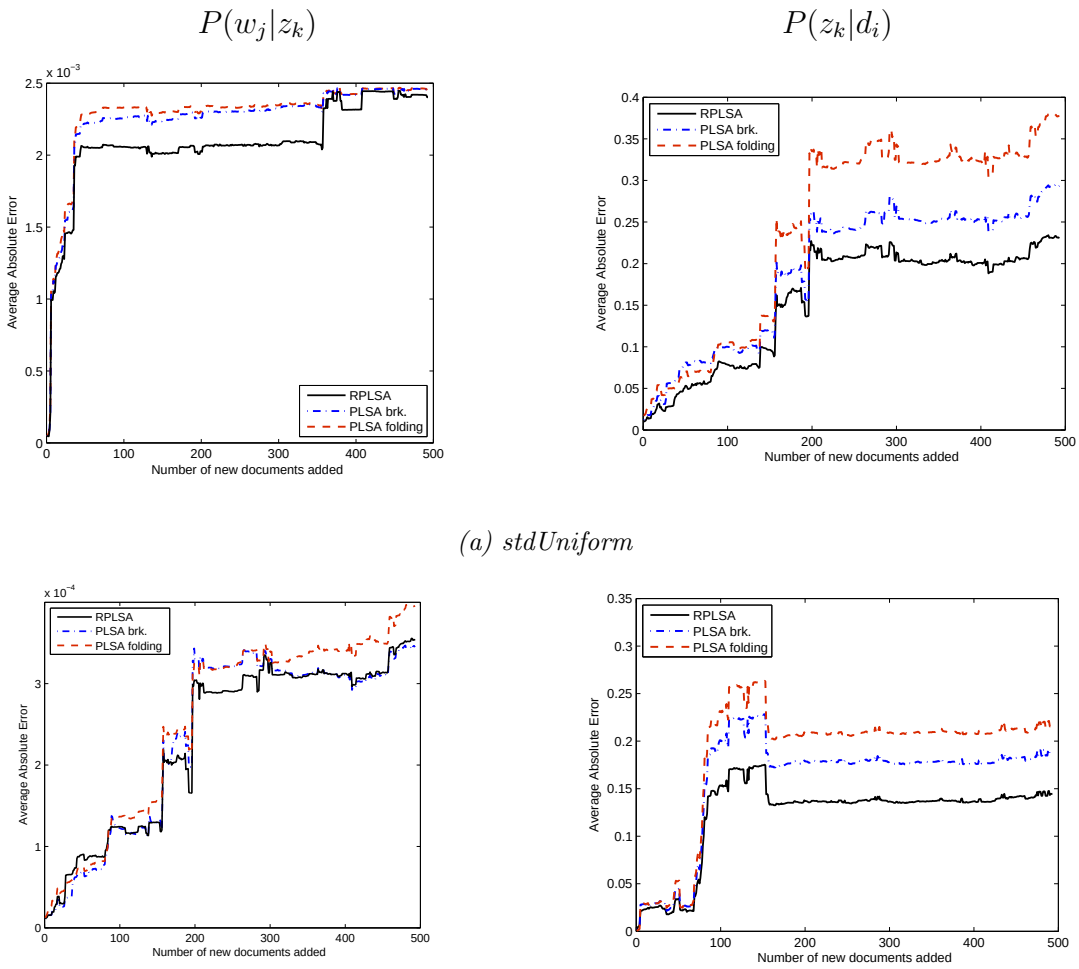
Moreover, the comparison between Tables 2 and 3 reveals that the RPLSA is more accurate for the asymmetric formulation than the symmetric one, since the error in $P(z_k|d_i)$ admits lower values for the asymmetric formulation

than the symmetric formulation.

Studying also Tables 2 and 3 with respect to the initialization method used in the estimation of the RPLSA model parameter for the document added, it can be verified that the RPLSA yields better results for the *wordtopics* initialization in 7 out of the 8 cases and in 13 out of 16 cases for the other two updating methods. Thus, it is claimed that the *wordtopics* initialization in RPLSA is more suitable than the *stdUniform* initialization.

The aforementioned conclusions are also verified by the inspection of Fig. 1 and 2, where the average absolute error between the probabilities $P(w_j|z_k)$ and $P(z_k|d_i)$ estimated by the PLSA beginning from scratch and those estimated by the RPLSA, the PLSA folding-in, and the PLSA rerun from the breakpoint over the K latent variables are plotted for the asymmetric and the symmetric formulations applied to the *rec_500* dataset. The superiority of the proposed RPLSA over the established updating methods under study is evident for all the initialization methods. The superiority of *wordtopics* initialization over *stdUniform* is also depicted.

Furthermore, in Fig. 3, the average-log likelihood at the limit point of the EM algorithm (i.e., when convergence is achieved) in the PLSA, the RPLSA, the PLSA folding-in, and the PLSA rerun from the breakpoint for each new incoming document incrementally added to the first subset of the *rec_500* dataset, under *stdUniform* and *wordtopics* initialization, is depicted. As it can be easily seen the proposed updating method for both the asymmetric and the symmetric formulations achieves an average log-likelihood close to that of the PLSA and higher than that of the PLSA folding-in under any initialization. The PLSA rerun from the breakpoint also achieves an average



(a) stdUniform

(b) wordtopics

Figure 1: Average absolute error over the K latent topics between the asymmetric PLSA model probabilities, $P(w_j|z_k)$ (first column) and $P(z_k|d_i)$ (second column), applied from scratch and the same probabilities of the PLSA updating methods under study for (a) stdUniform and (b) wordtopics initializations.

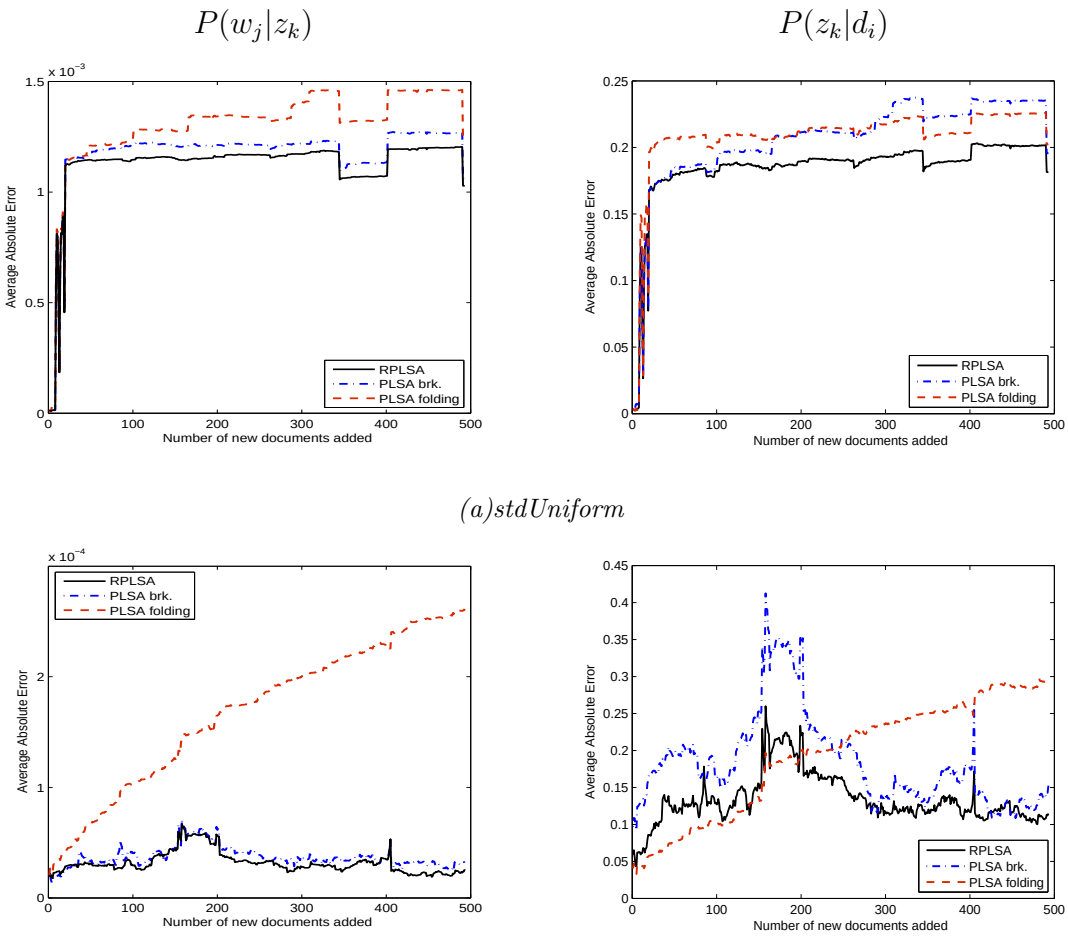
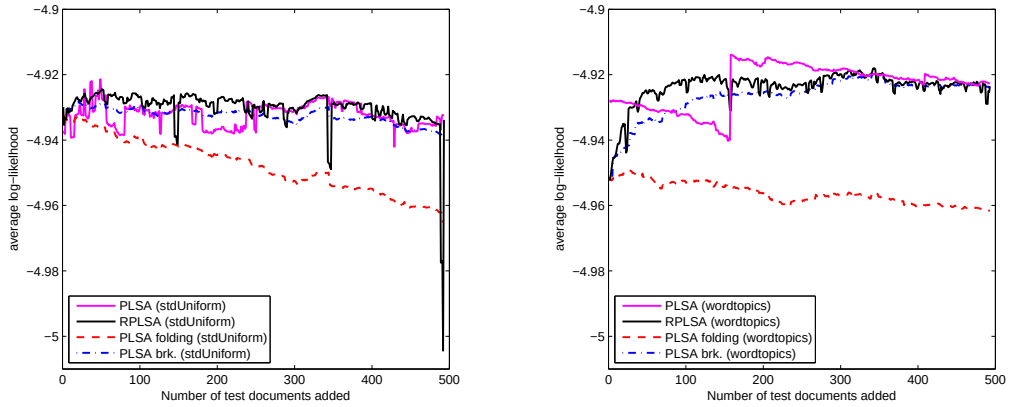
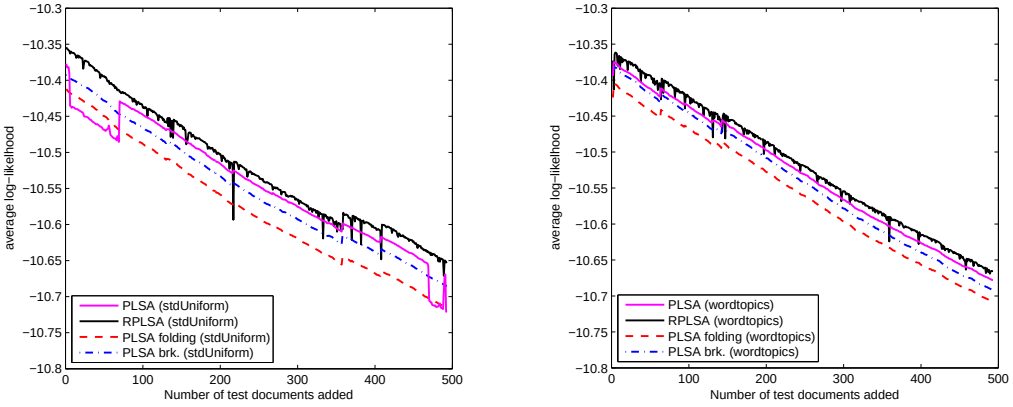


Figure 2: Average absolute error over the K latent topics between the symmetric PLSA model probabilities, $P(w_j|z_k)$ (first column) and $P(z_k|d_i)$ (second column), applied from scratch and the same probabilities of the PLSA updating methods under study for (a) stdUniform and (b) wordtopics initializations.

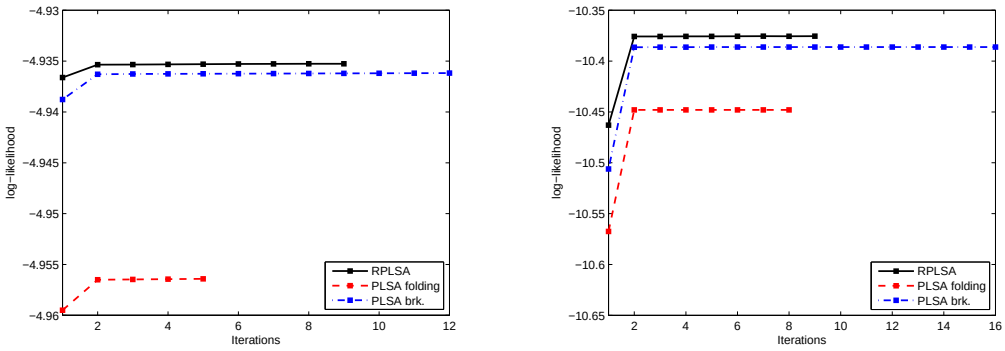


(a) Asymmetric formulation



(b) Symmetric formulation

Figure 3: Average log-likelihood at the limit point of the EM algorithm in the PLSA and the updating methods under study for each test document, when *stdUniform* and *wordtopics* initializations are applied: (a) asymmetric and (b) symmetric formulation.



(a) Asymmetric formulation *(b) Symmetric formulation*

Figure 4: Log-likelihood of the updating methods under study for a new incoming document: (a) asymmetric and (b) symmetric formulation.

log-likelihood quite close to that of PLSA. It is worth noting, that the average log-likelihood at the limit point is higher for the asymmetric formulation than the symmetric formulation, thus explaining the better performance of the former formulation.

Finally, Fig. 4 focuses on the convergence of the RPLSA, the PLSA folding-in and the PLSA rerun from the breakpoint for just one new incoming document added to the first subset of the *rec_500* dataset when the *wordtopics* initialization is employed for both the asymmetric and the symmetric formulations. As it can be seen, the RPLSA, the PLSA folding-in and the PLSA rerun from breakpoint maximize log-likelihood in both formulations. The convergence of the PLSA executed from scratch is not shown in Fig. 4, since the great number of iterations required for it to converge would deteriorate the level of detail in the plots.

5.4.2. CPU run time

In Table 4, the average over all the datasets CPU run time (in seconds) per newly added document is listed, when the RPLSA, the PLSA rerun from the breakpoint, the PLSA folding-in, and the PLSA executed from scratch are employed for the two initialization methods and the asymmetric and symmetric formulations. All the experiments were performed on a AMD Athlon 64bit 3200+ processor running at 2.01 GHz with 2GB RAM. Matlab R2007b for the Windows XP Professional x64 edition was used. By examining Table 4, it can be seen that among the three updating methods under study, PLSA rerun from the breakpoint is the most time consuming one, since it calculates all the model parameters in every EM step in contrast to PLSA folding-in that calculates only the probabilities of the latent topics given the documents. RPLSA is less time consuming than the PLSA folding-in in 1 out of the 4 cases. On average, the PLSA folding-in is by 9% faster in time than the RPLSA in 3 out of the 4 cases. However, the better accuracy offered by the RPLSA over the PLSA folding-in, as shown in Tables 2 and 3, compensates for the excessive computation time. In addition, it is verified that the updating methods, when the asymmetric formulation is used, are less time consuming than when the symmetric formulation is employed in the majority of the cases. Finally, the large amount of time needed by the PLSA executed from scratch compared to the time for the three PLSA updating methods demonstrates the need for adopting updating methods instead of running the PLSA from scratch, when new documents are added in the document collection.

Table 4: Average CPU time (in sec) per document for the updating methods under study and the PLSA executed from scratch when either *stdUniform* or *wordtopics* initialization method is employed in the asymmetric and symmetric formulation.

Formulation	*stdUniform*				*wordtopics*			
	RPLSA	*PLSA brk.*	*PLSA fold.*	*PLSA scratch*	*RPLSA*	*PLSA brk.*	*PLSA fold.*	*PLSA scratch*
asymmetric	2.70	12.38	2.93	107.95	2.68	11.83	2.00	105.80
symmetric	3.11	11.62	3.08	116.60	3.26	11.77	3.21	116.50

5.4.3. Evaluation Through Document Clustering

The performance of the PLSA updating methods under study was also tested on document clustering applications. After estimating the model parameters, the assignment of a document to a class was determined by means of the cosine similarity between the feature vector containing the conditional probabilities of the latent topics given the document and the prototype vector for each class obtained when the conditional probabilities of the latent topics given the document are averaged for all the documents previously assigned to the class.

Let D be the set of N documents $D = [d_1, \ldots, d_N]$. Given two clusterings of D, namely $\boldsymbol{C} = [C_1, \ldots, C_i, \ldots, C_K]$ with K clusters and $\boldsymbol{P} = [P_1, \ldots, P_j, \ldots, P_K]$ with K clusters ($\cap_{i=1}^{K} C_i = \cap_{j=1}^{K} P_j = \oslash$, $\cup_{i=1}^{K} C_i = \cup_{j=1}^{K} P_j = D$) in our case [1], the information on cluster overlap between $\boldsymbol{C}$ and $\boldsymbol{P}$ can be summarized in the form of a $K \times K$ contingency table $M = [n_{ij}]_{i=1\ldots K}^{j=1\ldots K}$ as illustrated in Table 5, where n_{ij} denotes the number of documents that are

[1] In general, the numbers of clusters in $\boldsymbol{C}$ and $\boldsymbol{P}$ can differ.

common to clusters C_i and P_j.

Table 5: Contingency table, $n_{ij} = |C_i \cap P_j|$

$\boldsymbol{C/P}$	P_1	P_2	$\dots$	P_j	$\dots$	P_K	*Sums*
C_1	n_{11}	n_{12}	$\dots$	n_{1j}	$\dots$	n_{1K}	a_1
C_2	n_{21}	n_{22}	$\dots$	n_{2j}	$\dots$	n_{2K}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	
C_i	n_{i1}	n_{i2}	$\dots$	n_{ij}	$\dots$	n_{iK}	a_i
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	
C_K	n_{K1}	n_{K2}	$\dots$	n_{Kj}	$\dots$	n_{KK}	a_K
Sums	b_1	b_2	$\dots$	b_j	$\dots$	b_K	$\sum_{ij} n_{ij} = N$

The Adjusted Rand Index (ARI) can be calculated as follows [16]:

$$ARI = \frac{\sum_{i=1}^{K}\sum_{j=1}^{K}\binom{n_{ij}}{2} - \left[\sum_{i=1}^{K}\binom{a_i}{2}\sum_{j=1}^{K}\binom{b_j}{2}\right]/\binom{N}{2}}{\frac{1}{2}\left[\sum_{i=1}^{K}\binom{a_i}{2} + \sum_{j=1}^{K}\binom{b_j}{2}\right] - \left[\sum_{i=1}^{K}\binom{a_i}{2}\sum_{j=1}^{K}\binom{b_j}{2}\right]/\binom{N}{2}} \tag{40}$$

where a_j and b_j are the marginal sums in the contigency table. The index admits a value between 0 and 1. The higher value is admitted by the ARI index, the stronger the similarity between the two clusterings.

The Adjusted Rand values for the clustering partitions derived by the PLSA executed from scratch, the RPLSA, the PLSA folding-in, and the PLSA rerun from the breakpoint are shown in Table 6.

The index values reveal the superiority of RPLSA over the established updating methods in producing more correct clusters, since the Adjusted Rand values for the RPLSA are higher than those for the PLSA folding-in and the PLSA rerun from the breakpoint. In addition, the index values for

Table 6: Adjusted Rand index values for clusterings derived by the updating methods under study and the PLSA executed from scratch, when either *stdUniform* or *wordtopics* initialization is applied to both the asymmetric and symmetric formulations.

Method	*comp_500*				*rec_500*			
	stdUniform		*wordtopics*		*stdUniform*		*wordtopics*	
	asym.	*sym.*	*asym*	*sym*	*asym*	*sym*	*asym*	*sym*
RPLSA	0.871	0.852	0.888	0.877	0.939	0.908	0.943	0.936
PLSA brk.	0.840	0.822	0.858	0.847	0.891	0.880	0.923	0.918
PLSA fold.	0.853	0.833	0.871	0.854	0.929	0.896	0.932	0.924
PLSA scratch	0.880	0.864	0.899	0.893	0.958	0.934	0.973	0.957

Method	*sci_500*				*talk_500*			
	stdUniform		*wordtopics*		*stdUniform*		*wordtopics*	
	asym.	*sym.*	*asym*	*sym*	*asym*	*sym*	*asym*	*sym*
RPLSA	0.901	0.893	0.939	0.927	0.846	0.834	0.869	0.859
PLSA brk.	0.878	0.875	0.906	0.897	0.806	0.793	0.849	0.843
PLSA fold.	0.888	0.885	0.920	0.912	0.814	0.798	0.856	0.852
PLSA scratch	0.938	0.929	0.964	0.948	0.870	0.858	0.884	0.879

RPLSA are closer to those for the PLSA executed from scratch than the values for the other updating methods under study are, thus revealing the ability of the proposed updating method in producing similar clusters to the ones generated by PLSA executed from scratch.

A last comparison of the PLSA and the RPLSA with the LSA and its updating methods, namely the SVD folding-in, and the SVD updating, within the document clustering framework was also made. The Adjusted Rand index values for the document clustering results produced by applying the LSA, the SVD folding-in, and the SVD updating, are presented in Table 7. As was expected, the SVD updating produces more correct clusters than the SVD folding-in. By comparing Table 7 with Table 6, the superiority of the PLSA and its updating methods over the LSA, the SVD folding-in, and the SVD updating is verified.

Table 7: Adjusted Rand index values for clustering results derived from the LSA, the SVD folding-in, and the SVD updating

Method	*comp_500*	*rec_500*	*sci_500*	*talk_500*
LSA	0.784	0.876	0.851	0.795
SVD folding-in	0.726	0.832	0.829	0.768
SVD updating	0.746	0.856	0.847	0.772

5.5. Discussion

The just presented results confirm that the proposed RPLSA updating method is more accurate than the other PLSA updating methods, such as the PLSA folding-in and the PLSA rerun from breakpoint. In addition, as expected the PLSA folding-in method is in most cases the fastest updating

method. However, there exist cases where the most accurate RPLSA updating method is the less time consuming one. Thus, in applications where accuracy is of critical importance, RPLSA can be selected as an updating method, when new documents are inserted in the initial document collections. Taking into account the ever increasing processing power of computers, the time requirements of the RPLSA will decline to almost negligible in applications, such as search engines. Clearly when accuracy is the key figure of merit in selecting an updating method and clearly RPLSA outperforms all the other updating schemes.

Moreover, as shown in the experiments all the updating methods under study yield in the majority of cases better results when the *wordtopics* rather than when the *stdUniform* initialization is employed. This can be attributed to the fact that, the probability of each added document given a topic is not set to a fixed value in *wordtopics* initialization, but is set to the probability of the words appearing in the document given a topic. This turns to be plausible, since the assignment of the document in one particular topic is related to the assignment of the words, it contains, to the topic. More accurate results are also obtained for the asymmetric formulation. This can be attributed to the fact that the symmetric formulation entails the estimation of one additional model parameter (i.e., $P(z_k)$) under the same convergence criterion used in asymmetric formulation. This additional parameter makes the symmetric formulation more time consuming than the asymmetric one.

In terms of document clustering, the proposed RPLSA produces more correct document clusters than the other updating methods under study, since the Adjusted Rand Index admits greater values for the RPLSA gener-

ated clusterings rather than for the clusterings generated by the other PLSA updating methods under study. RPLSA outperforms also the LSA updating methods, namely SVD folding-in and SVD updating. This fact demonstrates that the PLSA updating methods have superior modelling power than the LSA updating methods, as RPLSA has superior modelling power than LSA.

Finally, it is worth noting that in order to derive the RPLSA updating equations, it was assumed when the new documents, added one by one in the initial dataset, alter neither the number of topics nor the vocabulary. However, the proposed updating framework can be further extended to handle new documents that contain out of vocabulary words. This extension is currently investigated.

6. Conclusions

A novel method for updating the parameters of the PLSA for both the asymmetric and the symmetric formulations has been proposed, when new documents are added incrementally to an initial document collection. The proposed method has been compared to the PLSA folding-in and the PLSA rerun from the breakpoint. A first attempt was also made to investigate an efficient initialization of the probability distribution of the added documents. The experimental results have demonstrated that the probabilities estimated by the proposed RPLSA differ less from those estimated by the PLSA applied from scratch, when new documents are added incrementally than the probabilities estimated by either the PLSA folding-in or the PLSA rerun from the breakpoint. Moreover, the RPLSA achieves a higher average log-likelihood value upon EM convergence compared to that of the PLSA

folding-in and the PLSA rerun from the breakpoint. In terms of speed, the RPLSA is more time consuming than the PLSA folding-in in the majority of cases, but it is considerably faster than the PLSA rerun from the breakpoint. However, the RPLSA achieves a higher accuracy, thus compensating for the excessive computation time. Finally, the RPLSA updating method has been proven more effective in document clustering than the PLSA folding-in and the PLSA rerun from the breakpoint as well as the LSA, the SVD folding-in and the SVD updating which were also implemented. The implementation of other PLSA updating methods, such as the two variants of incremental PLSA [7, 25], the Bayesian folding in [11], the MAP estimation for corrective training and Quasi-Bayes estimation for incremental learning [6] and their performance comparison could be a topic of further research.

Acknowledgments

The authors would like to thank Mr. Athanasios Papaioannou for the first implementation of the asymmetric RPLSA.

References

[1] J. R. Bellegarda. Fast update of latent semantic spaces using a linear transform framework. In *Proc. 2002 IEEE Int. Conf. Acoust., Speech, and Signal Process.*, volume 1, pages 769–772, 2002.

[2] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. Technical Report UT-CS-94-270, 1994.

[3] J. A. Bilmes. Graphical models and automatic speech recognition. In M. Johnson, S. P. Khudanpur, M. Ostendorf, and R. Rosenfeld, editors,

Mathematical Foundations of Speech and Language Processing, pages 191–246. Springer-Verlag, New York, N.Y., 2004.

[4] T. Brants. Test data likelihood for PLSA models. *Information Retrieval*, 8(2):181–196, 2005.

[5] T. H. Brants, I. Tsochantaridis, T. Hofmann, and F. R. Chen. Methods, apparatus, and program products for performing incremental probabilitstic latent semantic analysis. Patent No. 20060112128, May 2006.

[6] J.-T. Chien and M.-S. Wu. Adaptive Bayesian latent semantic analysis. *IEEE Trans. Audio, Speech, and Language Processing*, 16(1):198–207, January 2008.

[7] T.-C. Chou and M. C. Chen. Using incremental PLSI for threshold-resilient online event analysis. *IEEE Trans. Knowledge and Data Engineering*, 20(3):289–299, March 2008.

[8] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal American Society of Information Science*, 41(6):391–407, 1990.

[9] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal Royal Statistical Society, Series B*, 39:1–38, 1977.

[10] D. Gildea and T. Hofmann. Topic-based language models using EM. In *Proc. 6th European Conf. Speech Communication and Technology*, pages 2167–2170, Budapest, 1999.

[11] A. Hinneburg, H.-H. Gabriel, and A. Gohr. Bayesian folding-in with dirichlet kernels for PLSI. In *Proc. 7th IEEE Int. Conf. Data Mining*, pages 499–504, Omaha, Nebraska, USA, October 28-31 2007.

[12] T. Hofmann. Probabilistic latent semantic indexing. In *Proc. Research and Development in Information Retrieval*, pages 50–57, 1999.

[13] T. Hofmann. Learning the similarity of documents: An information-geometric approach to document retrieval and categorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 914–920. The MIT Press, 2000.

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

[15] T. Hofmann and J. Puzicha. Unsupervised learning from dyadic data. Technical Report TR-98-042, International Computer Science Institute, Berkeley, CA, 1998.

[16] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[17] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, N.J., 1988.

[18] K. Lang. Newsweeder: Learning to filter netnews. In *Proc. 12th Int. Conf. Machine Learning*, pages 331–339, 1995.

[19] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification, and clustering. 1996.

[20] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[21] G. W. O'Brien. Information management tools for updating an SVD-encoded indexing scheme. Technical Report UT-CS-94-258, 1994.

[22] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.

[23] J. E. B. Tougas. Folding-up: A hybrid method for updating the partial singular value decomposition in latent semantic indexing. Master of computer science, Dalhousie University, 2005, Halifax, Nova Scotia, December 2005.

[24] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, 1(6):80–83, 1945.

[25] H. Wu, D. Zhang, Y. Wang, and X. Cheng. Incremental probabilistic latent semantic analysis for automatic question recommendation. In *Proc. ACM Conf. Recommender Systems*, pages 99–106, Lausanne, Switzerland, October 23-25, 2008.

[26] H. Zha and H. D. Simon. On updating problems in latent semantic indexing. *SIAM Journal on Scientific Computing*, 21(2):782–791, 1999.