Latent Semantic Indexing for Patent Documents

Andreea Moldovan * Radu Ioan Boţ † Gert Wanka ‡

Abstract. Since the huge database of patent documents is continuously increasing, the issue of classifying, updating and retrieving patent documents turned into an acute necessity. Therefore we investigate the efficiency of applying Latent Semantic Indexing, an automatic indexing method for information retrieval, to some classes of patent documents from the United States Patent Classification System. We present some experiments that provide the optimal number of dimensions for the Latent Semantic Space and we compare the performance of Latent Semantic Indexing (LSI) to the Vector Space Model (VSM) technique applied to real life text documents, namely patent documents. Though we do not strongly recommend the LSI as an improved alternative method to VSM, since the results are not significantly better.

Keywords. Latent Semantic Indexing (LSI), Singular Value Decomposition (SVD), Vector Space Model (VSM), patent classification

AMS (MOS) subject classification. 15A18, 62H30, 65A48, 68P20

*Faculty of Mathematics, Chemnitz University of Technology, D-09107 Chemnitz, Germany, e-mail: amol@mathematik.tu-chemnitz.de.

†Faculty of Mathematics, Chemnitz University of Technology, D-09107 Chemnitz, Germany, e-mail: radu.bot@mathematik.tu-chemnitz.de.

‡Faculty of Mathematics, Chemnitz University of Technology, D-09107 Chemnitz, Germany, e-mail: gert.wanka@mathematik.tu-chemnitz.de

1 Introduction

Latent Semantic Indexing (LSI) [3] is a mathematical approach to information retrieval that attempts to model the underlying structure of term associations by transforming the traditional representation of documents as vectors of weighted term frequencies into a new coordinate space where both documents and terms are represented as linear combinations of implicit semantic factors. Although it was originally applied in the context of information retrieval, since then it has been successfully applied to a wide variety of text-based tasks [15]. The use of LSI for text retrieval has been proposed in several works, being emphasized the improvement of both retrieval precision and recall ([2], [3], [5], [11], [15]). LSI turned out to improve also text categorization [6] and word sense discrimination [19].

These studies of LSI have mostly used standard text collections in information retrieval, some of them having simplified document models. For most of them, LSI proved to be a promising approach to automatic indexing. The purpose of this paper is to investigate the use of LSI on some real-life text documents, namely patent documents. Our motivation came from the continuous growth of the patent documents database in the recent years, an increase which requires the development of new and efficient methods for classification and retrieval of patent documents.

One such vast database of patents is the United States Patent Classification System (UPSTO) Patent Full-Text Database, covering US patents issued from 1790 to the present. It contains patents classified accordingly to their claims in 450 main classes and about 15000 subclasses. We applied LSI on ten of the main classes of patents belonging to UPSTO.

The motivation behind our work using databases of patent documents consists in the following facts. First, the huge amount of information which is suitable for the methods we used and, on the other hand, the existence of an *a priori* classification made by experts (patent attorneys). Thus we could use the information given there in order to verify the results provided by our program. In this way we relate our results to raw data unlike usually made in the past in the literature, where the standard information retrieval test collections (MED, CISI, TIME, etc.) were used.

Encouraged by the positive results reported so far in the literature (average improvements up to 30% brought by using LSI instead of VSM) the idea of applying LSI to patent classification came naturally. Unfortunately the results we obtained turned out to be not so good as expected. We wonder which is the cause of this fact: inefficiency of LSI or the special structure of the standard information retrieval test collections (MED, CISI, TIME, etc.), on which its success was highlighted.

This paper is organized as follows. In section 2 we start with a brief overview of LSI and we continue in the next section with the presentation of the patent documents collection used in our experiments, the preprocessing we performed to the text data and the way we implemented LSI. In section 4 we examine the results of our experiments and we conclude in section 5.

2 Latent Semantic Indexing

Latent Semantic Indexing [3] is a statistical technique which tries to surpass some limitations imposed by the traditional Vector Space Model (VSM) [17]. In VSM, which uses the so-called *bag-of-words* representation of documents, the collection

of text documents is represented by a *terms-documents* matrix $A = [a_{ij}] \in \mathbb{R}^{t \times d}$, where each entry a_{ij} corresponds to the number of times the term i appears in document j. Here t is the number of terms and d the number of documents in the collection. Therefore a document becomes a column vector and a query of a user can be represented as a vector of the same dimension. The similarity between the user's query vector and a document vector in the collection is measured as the cosine of the angle between the two vectors. A list of documents ranked in decreasing order of similarity is returned to the user for each query. The VSM considers the terms in documents as being independent from each other, an assumption which is never satisfied by the human language. An idea can be expressed in many ways (synonymy) and, moreover, many words may have multiple meanings (polysemy).

Latent Semantic Indexing is a variant of VSM that exploits the dependencies between words by assuming that there is some underlying or "latent" structure in word usage across documents and this structure can be revealed statistically. LSI is a method for *dimensionality reduction* because it transforms the original terms-documents vector space into a new coordinate system of *conceptual topics*, a lower dimensional space that captures the implicit higher-order structure in the association of terms with documents [3]. Both sets of documents and terms will be projected onto this new low-dimensional space spanned by the true *factors* or *concepts*, instead of representing documents as vectors of independent words. In order to obtain the space of concepts, i.e. the space of true representation of words and documents, LSI uses a truncated Singular Value Decomposition (SVD) applied to the terms-documents matrix A described above.

Given a $t \times d$ matrix A, where $m = \min(t, d)$, the singular value decomposition

of A is defined as

$$A = USV^T, \tag{1}$$

where U is a $t \times m$ orthonormal matrix ($U^TU = I_m$), whose columns define the left singular vectors, V is a $d \times m$ orthonormal matrix ($V^TV = I_m$), whose columns define the right singular vectors and S is a $m \times m$ diagonal matrix containing the singular values of A decreasingly ordered along its diagonal: $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > \sigma_{r+1} = ... = \sigma_m = 0$, where $r = rank(A)$. This decomposition is unique up to making the same permutations of columns of U, elements of S and columns of V (rows of V^T).

To reduce the noise and redundancy, LSI, taking as input the terms-documents matrix described above, uses a truncation of SVD which consists in retaining only the largest k singular values and deleting the remaining ones which are smaller and thus considered unimportant. We remove also from U and V the columns corresponding to the small singular values and we get

$$A_k = U_k S_k V_k^T,$$

where S_k is a $k \times k$ diagonal matrix containing the largest k singular values as entries, U_k is a $t \times k$ matrix of the corresponding left singular vectors as columns and V_k is a $d \times k$ matrix whose columns are the corresponding right singular vectors.

According to the following result, due to Eckart and Young ([11]), the matrix A_k may also be interpreted as an approximation of the original matrix A.

Theorem 1. *A_k is the best approximation of A by a matrix of rank k and*

the error is

$$\|A - A_k\|_F = \min_{rank(B) \leq k} \|A - B\|_F = \sqrt{\sigma_{k+1}^2 + ... + \sigma_{\min(t,d)}^2} \ .$$

The subscript F denotes the Frobenius norm, defined for a matrix $A = (a_{ij}) \in \mathbb{R}^{t \times d}$ as $\|A\|_F = \sqrt{\sum_{i=1}^{t} \sum_{j=1}^{d} |a_{ij}|^2}$.

The choice of k is a critical problem. While a too drastic reduction of the dimensionality may lead to loss of valuable information about the collection, problems like the difficulty of handling with dimensionality and, as observed empirically, poorer performance may also occur when a too high dimensionality is chosen. The appropriate value of k depends on the text collection and it can be detected only empirically. From what was reported in the literature, we can conclude that the optimal value of k lies between 50 and 350.

For processing a user's query Q, the query-document comparisons for every document in the collection are carried out, i.e. comparisons between the query vector and each column of the matrix A_k. If e_j denotes the jth canonical vector of dimension d, then $A_k e_j$ will be the jth column of the matrix A_k and one may compute the cosines of the angles θ_j, $j = 1, ..., d$, between the query vector and the columns of the matrix A_k

$$\cos \theta_j = \frac{(A_k e_j)^T Q}{\|A_k e_j\|_2 \|Q\|_2} = \frac{e_j^T V_k S_k (U_k^T Q)}{\|S_k V_k^T e_j\|_2 \|Q\|_2},$$

for $j = 1, ..., d$. Here we used that the multiplication of a vector by an orthonormal matrix does not alter the Euclidean norm of the vector. Let us denote $r_j := S_k V_k^T e_j$ and then we have $\|U_k r_j\|_2 = \sqrt{(U_k r_j)^T U_k r_j} = \sqrt{r_j^T U_k^T U_k r_j} = \sqrt{r_j^T I_k r_j} =$

$\sqrt{r_j^T r_j} = \|r_j\|_2$. Documents ranked by their similarity to the query are returned to the user.

It is important for the LSI method that the derived A_k matrix does not reconstruct exactly the original term document matrix A, since we assumed A to be unreliable. Deerwester et al. [3] claim that the truncated SVD, in one sense, captures most of the important underlying structure in the association of terms and documents, but in the same time removes the noise or variability in word-usage that diminish the performance of the Vector Space Model. Intuitively, since the number of dimensions k is much smaller than the number of terms t, it is expected that terms which occur in similar documents will be near each other in the k-dimensional factor space even if they never co-occur in the same document. This means that some documents, which do not share any words in common with a user's query, may however be near to it in the k-dimensional space. Although the logic of this claim seems clear, it has not been directly verified, though in [13] it has been shown that LSI can overcome the broader problem of term mismatch between query and their relevant documents. In regard to the other gain of LSI over literally matching terms in documents with those of a query, polysemy, this should arise from dimensionality reduction, by removing the rare and less important usages of certain terms. Since the LSI term vector is the weighted average of the different meanings of the term, when the real meaning differs from the average meaning, LSI might actually reduce the quality of retrieval. An SVD of the term similarity matrix is proposed in [18]in order to directly determine the sense of a particular word.

One might expect that using a low-rank dimensionality representation could reduce substantially the storage requirements, when in reality the opposite is true [11]. We can no longer take advantage of the sparsity of the original very

sparse vector representation since the matrices in (1) are not sparse anymore. The LSI values are real numbers while the original term frequencies are integers. Although the SVD is a one-time expense, Kolda and O'Leary suggested in [14] a LSI approach based on semi-discrete matrix decomposition. This approach, which demands much lesser storage space, produces results comparable with those reported for LSI based on SVD.

A specific source of LSI's discriminative power is not exactly clear. As an explanation for the performance of LSI was often cited Theorem 1. But this result can only provide an explanation for the fact that LSI does not decrease the performance obtained with the traditional VSM, but does not justify the improvement in precision and recall noticed when LSI was applied to some collections of text documents. Several researchers have been looking for theoretical foundations of the improved performance observed empirically. It has been furnished a theoretical interpretation of LSI in relation to the Bayesian regression model [20], Multidimensional Scaling [1], a similarity-based probabilistic model [4], a corpus model for which, under certain conditions, LSI succeeds in capturing the underlying semantics [16] and a method for selecting the number of dimensions founded on a subspace-based model and the minimal description length [21].

3 Methodology

The collection of text documents we used in our study was taken from United States Patent Classification System (USPTO). Our investigations focused on the patent documents which were classified in [22] in the following ten main US classes (left-US class number, right-class name)

307 Electrical transmission or interconnected systems

323 Power supply or regulation systems

329 Demodulators

330 Amplifiers

331 Oscillators

332 Modulators

338 Electrical resistors

343 Radio wave antenna

703 Structural design, modelling, simulation and emulation

706 Artificial intelligence.

For each of these classes, it has been extracted the list of terms and the list of patent documents, together with the frequency of the appearance of each word in each document. Following the suggestions given in the literature in order to improve retrieval but in the same time to improve computation time and storage requirements, we decided to perform some pre-processing of our text data. The first pre-processing step we proceed is elimination of one-letter words, digits and "_", "-" and "" characters ("t2o" became "to", "user's" became "users"). To avoid indexing uninteresting terms for our retrieval, the next step includes the removal of the so-called *stopwords* (e.g. "and", "but", "or") by using the SMART-stoplist of 571 stopwords [24]. The last step of pre-processing concerns the stemming. Some authors suggest that morphologically related words belonging to the same root (sharing a common "stem") may be treated alike, for example "decompose", "decomposed", "decomposing", "decomposes" and "decomposition". Although recent research [12] shows that "some of the stemming is almost always beneficial but the average absolute improvement due to stem-

ming is small, ranging from 1 to 3", we applied the classic stemming algorithm proposed by Porter [25] also from the storage interests.

After all of the above is done, and the number of terms to be indexed is reduced by about 40%, we were ready for constructing the terms-documents matrix corresponding to each class (i.e. corresponding to the words and patent documents belonging to each of the classes studied). No term weighting has been used for the entries of the terms-documents matrix.

Deciding whether a document is relevant or not is a very subjective matter, because it depends ultimately on the user to decide if the returned documents are pertinent or not to his goal. Usually, a standard information retrieval test collection (MED, CISI, TIME, etc.) contains a collection of articles, a set of queries and a list of documents for each query indicating its relevant documents. In the semantic space the cosines between the query vector and the documents vectors are computed and those documents for which the cosine with the query is greater than a given threshold are returned. Using this last list of relevant documents available, the efficiency of LSI to the respective test collection can be analyzed. In our case, the information we had for evaluating the performance of LSI was not a list of relevant documents (similar patents) corresponding to each of the patent documents, rather a set of document-class pairs, showing which patent in which class (main and/or subclass) is classified in USPTO. We decided to exploit this information in the form of a *classes-documents* matrix $B = [b_{ij}] \in \mathbb{R}^{c \times d}$, where each entry (b_{ij}) is 1 when the j-th document was classified in the i-th class and 0 otherwise. Here c is the number of classes and d the number of documents.

Before computing the classes-documents matrix, there were two things to notice about the data we had, which gave us the classification of patents into

classes. First remark is that one patent document could be classified in more than just one class or subclass. The second remark is connected with the structure of the US classification system of patents. We observed that among the classes and subclasses list associated to every patent, one could find the following situation: the list of classes in which the document was classified included both a class and some of its subclasses. Since we settled that in the classes-documents matrix the classes would be treated as independent to each other, arose the need of transforming the information data regarding the classification of patents (because we cannot consider, for instance, the main class 307 and its subclass 307/149 as two different classes). For this purpose, we stored the US classification system for patents in a tree structure where a node stands for a class and as its children in the tree are added its corresponding direct subclasses. Then, based on the structure of the tree, if one document was classified both in a "parent" class and in some of its "children" or "grand-children" subclasses, then we retain only the "parent" class, considering the information about "children" subclasses is already contained in the "parent" subclass (i.e. when a patent is classified in the class 307, in 307/154 and in 307/155, where 307/154 is a subclass of 307 and 307/155 is a subclass of 307/154, then we consider the patent as being classified in the class 307). After processing these transformations applied to the classes-documents information data, the number of subclasses where the documents are classified decreases consistently for each of the ten classes. For example, for the US class 307, from the originally 186 subclasses in which the patents of the class were classified, after the transformation mentioned above, we obtained in the end only 26 subclasses (all of them of level 2 in the tree, that means direct subclasses of the main class 307). The classes-documents matrix for each class were created from the transformed data information and in Table 1 can be seen the dimensions

of both terms-documents matrix A and classes-documents matrix B obtained for each of the ten investigated classes.

In order to analyze the effectiveness of applying LSI to patent documents and to compare the retrieval results with the US classification system for patents, considering consequently each document as being a query, we computed the matrix X of similarity scores between the documents derived from the LSI approach by $X = A_k^T A_k$, where the columns of A_k have been preliminarily normalized. The matrix X has as entries (x_{ij}) the cosine of the angle of vectors corresponding to the projection of documents d_i and d_j, respectively, into the latent semantic space.

To evaluate the retrieval results, we used two measures. The first one is a widely used measure of performance for retrieval systems, consisting in determining two values: precision and recall. Typically, for an information retrieval system, the precision relative to a query is defined as

$$p = \frac{\text{number of relevant documents retrieved}}{\text{number of all documents retrieved}}$$

and is a measure of accuracy, while the recall relative to the query is defined as

$$r = \frac{\text{number of relevant documents retrieved}}{\text{number of all relevant documents}},$$

being a measure of completeness. First, we explain the significance of "documents retrieved" and "relevant documents" in our experiments. For a certain document $d_j,\ j = 1, ..., d$ (seen as query), we extract from the matrix B the corresponding ranked list of relevant documents (referred in the definition of recall as "all relevant documents") as being the list of documents which co-occur with d_j in at least one class. The list of "all documents retrieved" by our retrieval system LSI

is the ranked list obtained for the document d_j after computing the cosines of the angles between the document vector d_j and all the other documents. In order to detect which of the retrieved documents are also relevant for our document d_j, the list of retrieved documents (obtained from LSI) is compared to the list of relevant documents (obtained from the matrix B). The next step is to compute the values of precision at levels of recall 0.1, 0.2, ..., 0.9, standard technique in literature for evaluating a retrieval system. As an example, at the given level of recall 0.5 we have already obtained say n_5 relevant documents which represent 50% of the entire list of relevant documents (actually we obtained the first half of the list of documents obtained from B). At the level of recall 0.5, the precision will be then the ratio between n_5 and the number of documents, say l_5, in the list returned by LSI, chosen so that all the n_5 relevant documents are contained in the first l_5 elements of the ranked list of retrieved documents. In this way we compute the precision at recall 0.5 for every document $d_j,\ j = 1, ..., d$, and then, to obtain a global characterization of our retrieval system, we compute the mean value of precisions at recall 0.5 for all the documents. As a last step, we calculate the average precision characterizing our LSI retrieval system as the mean value of precisions at levels of recall 0.1, 0.2, ..., 0.9.

The second measure used the co-occurences of documents based on the classes-documents matrix. The similarity matrix $Y = B^T B$ delivers us the co-occurences of the vector columns in B based on the given UPSTO classification. We wanted to compare the similarity matrix X derived from LSI with this similarity matrix Y in order to see if the LSI derived classification is a convenient approximation of the original UPSTO classification. For this aim, we computed the value of the norm

$$\left\| \frac{X}{\|X\|_F} - \frac{Y}{\|Y\|_F} \right\|_F, \tag{2}$$

where $\|\cdot\|_F$ denotes the Frobenius norm defined in Section 2. We searched for its minimum as a function of k, the number of singular values.

We summarize the methods described above in the following algorithms.

Algorithm 1. Computation of the average precision:

Input

$A = \big(A(i,j)\big)_{\substack{i=\overline{1,t}\\ j=\overline{1,d}}}$ - *the* $(t \times d)$ *matrix of terms and documents*

$B = \big(B(l,j)\big)_{\substack{l=\overline{1,c}\\ j=\overline{1,d}}}$ - *the* $(c \times d)$ *matrix of classes and documents*

Output

$AvP = \Big(AvP(j,k)\Big)$ - *the matrix of average precisions for* $j = 1, ..., d$ *and* $k = 5, ..., 500$

Loop

for $j = 1, ..., d$

- $B(j) \leftarrow \frac{1}{\|B(j)\|} B(j)$; *normalization of the column* $B(j)$ *of* B
- $rB(j) \leftarrow B(j)^T B$; $rB(j)$ *is the vector of relevance for document* d_j
- $nB(j) \leftarrow$ *the number of the strictly positive entries of the vector* $rB(j)$; $nB(j)$ *is the number of all relevant documents for document-query* d_j

end for j

for $k = 5, ..., 500$

- **perform SVD**: $A = USV^T$
- **compute** $A_k = U_k S_k V_k^T$ *by retaining the largest* k *singular values*
- **for** $j = 1, ..., d$

- $A_k(j) \leftarrow \frac{1}{\|A_k(j)\|} A_k(j), j = 1, ..., d$; *normalization of each column* $A_k(j)$ *of* A_k

- $rA_k(j) \leftarrow A_k(j)^T A_k$

- $nA_krel(j) \leftarrow$ *the number of the strictly positive entries of the vector* $rA_k(j)$; $nA_krel(j)$ *is the number of all documents retrieved for document-query* d_j

- $nA_krelret(j) \leftarrow$ *the number of documents* $d_i, i = 1, ..., d$, *for which the cosine with the vector* d_j *is positive both in* $rB(j)$ *and in* $rA_k(j)$; $nA_krelret(j)$ *is the number of relevant documents retrieved for document-query* d_j

- **for** $s = 1, ..., 9$

  compute *precision* $p(s, j)$ *at levels of recall* $0.s$

  end for s

  $AvP(j, k) \leftarrow \frac{1}{9} \sum_{s=1}^{9} p(s, j)$; $AvP(j, k)$ *is the mean average precision*

end for j

end for k

return *the matrix* AvP *with entries* $AvP(j, k)$

Algorithm 2. Computation of the norm in (2):

Input

$A = \left(A(i, j)\right)_{\substack{i=\overline{1,t} \\ j=\overline{1,d}}}$ - *the* $(t \times d)$ *matrix of terms and documents*

$B = \left(B(l, j)\right)_{\substack{l=\overline{1,c} \\ j=\overline{1,d}}}$ - *the* $(c \times d)$ *matrix of classes and documents*

Output

$f(k)$, *the value of the norm in* (2), *for* $k = 5, ..., 500$

Loop

$Y \leftarrow B^T B$; *Y is the similarity matrix*

for $k = 5, ..., 500$

- **perform** *SVD:* $A = USV^T$
- **compute** $A_k = U_k S_k V_k^T$ *by retaining the largest k singular values*
- **compute** *the matrix* $X = A_k^T A_k$
- **compute** *the Frobenius norm* $f(k) = \left\| \frac{X}{\|X\|_F} - \frac{Y}{\|Y\|_F} \right\|_F$

end for k

The truncated SVD was computed in our experiments with the ARPACK++ software package [23] and we developed C++ software for the computation of similarity matrices and calculation of the two measures described above. The computer we used was an 1.60 GHz Intel(R) Pentium 4 CPU machine with 1.048.048 KB RAM.

4 Results and Discussion

Our first interest was to compare the performance of the traditional VSM with the retrieval performance of LSI, separately, for each of the ten classes and for different values of k (all our experiments were made for a range of values of k going from 40 to 500). Various results were reported until now in the literature regarding the improvement in performance over the VSM, an average improvement of 30% being sometimes indicated. Although there is no evidence in the published literature that LSI can reliably be expected to deliver such performance on any

given collection ([3], [5], [6]), rather the LSI model can almost always match the VSM, sometimes slightly outperforming it.

Any two systems that provide documents retrieval can be compared by plotting their respective precision-recall curves on the same plot. In general, systems with higher precision across a wide range of recall values are superior, i.e. the higher the curve the better. Our results confirm the claim that LSI can almost always match VSM (see Figures 1 and 2), sometimes slightly improving it, since from the total number of ten classes we noticed that LSI has slightly improved the VSM over seven of the classes (an average improvement of 5%) and has slightly harmed the retrieval result for one class (US class 330) with an average damage of 3% until the choice for k for the respective class reached the value 500. For another two classes LSI has performed slightly better for values of k smaller than 100 for one class (US class 703) and $k = 200$ for the other (US class 329), respectively, and then damaged the retrieval for values greater than $k = 100$, respectively, $k = 200$. We add as a remark that for most of the seven classes the slightly improved performance of LSI over VSM was noticed for any value of k and two from these two classes (US classes 338 and 706) verified the assertion that LSI improves precision at higher levels of recall.

Another interest of ours went into the direction of determining the optimal value of singular values k which should be used in the truncation of SVD for obtaining best results. Therefore, we plotted the average precision as a function oh the number k of singular values for each one of the ten classes. We expected to obtain the optimal values of k to lie between 100 and 300, as reported in the literature, although in [13] it has been remarked that even for the standard test collections for information retrieval like MED (a collection of a set of Medline articles on various medical topics [26]) or TIME (a collection of articles from

TIME magazine from 1963 [27]), reasonable performance can be obtained only for $k = 50$ for MED and for TIME in the range of 50-300, although the best performance is seen for full or nearly full matrices. Analyzing the plots we obtained, and as can be seen in Figures 3, 5 and 7, we deduce that in our experiments and for our patent documents, a value of 80 singular values kept while truncating SVD suffices to assure us a good retrieval. For the classes we examined the average precision tends to decrease very slowly after reaching a maximum around 80-100 for some classes or, another tendency noticed in the plots was that for some other classes the values of average precision strongly increase till $k = 80 - 100$ and then continue to increase further but very slow. Hence, we conclude that setting $k = 80$ is a reasonable choice for our collection of patent documents.

This conclusion has been strengthened by the other measure we decided to use, that is computing the similarities of the vector documents in the classes-documents matrix, based on their co-occurence in the same classes. We were interested in seeing how good "approximation" of the similarity matrix obtained from classification is the LSI derived similarity matrix. We calculated the expression in (2) for k ranging across the same values as in the experiments evaluating average precision described above. Plotting the values obtained as a function of k, we notice that for each class the behavior of the values of norm in (2) mirrors the plot (with respect to the horizontal coordinate axis) of corresponding class in the average precision experiments as can be seen in Figures 3 and 4, 5 and 6, 7 and 8. This "mirroring" appears since for the average precision experiments we were interested in maximizing the average precision, while in these experiments we were searching for minimum of the norm values. Thus, using two different measures to evaluate the performance of LSI, we came to the same conclusion, namely that the best choice for the value of k, the number of singular values in

the truncation of SVD, is about $k = 80$.

5 Conclusions

We presented an empirical study of an unsupervised method for information retrieval, LSI based on SVD, a method which was mostly studied using some standard test collections in information retrieval. To the best of our knowledge, the performance of LSI has been studied on standard text collections which are often used in information retrieval, but we investigated the effectiveness of LSI in text retrieval by applying this method for automatic indexing to a special kind of documents of real life documents, namely patent documents. Since the documents have already been classified in the UPSTO, instead of the typical list with relevant documents to a given query, we used a classes-documents matrix specifying for each document in which class has been previously classified (a matrix which is generally needed in supervised methods for classification). Based on this information regarding the classification of documents in classes, we constructed the corresponding lists of relevant documents for the set of queries (which in our case is actually the set of documents) and we proceeded with analyzing the typical evaluation measures, precision and recall. On the other hand we have also evaluated the extent to which the similarities between the documents in the latent semantic space approximate the co-occurences of documents in the same class in the USPTO classification.

Thus, we provided results using two different measures which gave us the same behavior for each of the classes investigated: although LSI improves very slightly on VSM, the power of this tool for information retrieval is proved once again by applying it to patent documents and by managing with good performance to

rediscover the previous made classification of these documents. We found that for the used patent document collection it is enough to use a number of 80 factors which span the latent semantic space (i.e number of singular values retained in the truncation of SVD) and this number is sufficient for acquiring satisfying results in the retrieval process based on the LSI technique. As most of the researchers (including its inventors), we do not strongly recommend the LSI as an improved alternative and being far superior to VSM. Although we obtained slightly better results than VSM for some classes, the improvement in retrieval performance was nevertheless modest.

Another method in information retrieval which may be applied in patent classification is the probabilistic latent semantic indexing (cf. [7], [8]). This is subject to future research.

Acknowledgments. The first author's work has been supported by the Federal Ministry for Economy and Technology (Germany) through the PRO INNO Program. We would like to thank Dr. Ulf Bauerschäfer from the firm IP Century AG (www.ipcentury.com) for providing us the patent documents data and for fruitful discussions, comments and cooperation in the project. Thanks are also due to two anonymous referees for valuable comments that improved the quality of the paper.

References

[1] Bartell, B.T., Cotrell, G.W., Belew, R.K. (1992): *Latent Semantic Indexing is an optimal special case of Multidimensional Scaling.* In: Proceedings of ACM/SIGIR'92, Copenhagen, pp.161–167

[2] Berry, M.W., Dumais, S.T., O'brien, G.W., (1995): *Using linear algebra for intelligent information retrieval*. SIAM Review 37(4), pp. 573-595

[3] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R. (1990): *Indexing by latent semantic analysis*. Journal of the American Society for Information Science 41, pp. 391–407

[4] Ding, C.H.Q. (1999): *A similarity-based probability model for latent semantic indexing*. In Proceedings of the 22nd ACM/SIGIR Conference, Berkley, California, pp. 58–65

[5] Dumais, S.T. (1991): *Improving the retrieval of information from external sources*. Behavior Research Methods, Instruments and Computers 23(2), pp. 229–236

[6] Dumais, S.T. (1995): *Using LSI for information filtering: TREC-3 experiments*. Third Text REtrieval Conference (TREC3), D. Harman (eds), National Institute of Standards and Technology Special Publication

[7] Fuhr, N. (1989): *Models for retrieval with probabilistic indexing*. Information Processing and Management 25(1), pp. 55–72

[8] Fuhr, N. (1992): *Probabilistic models in information retrieval*. The Computer Journal 35(3), pp. 243–255

[9] Golub, G.H., Reinsch, C. (1971): *Handbook for matrix computations II, Linear Algebra*. Springer-Verlag, New York

[10] Golub, G.H., Van Loan, C.F. (1989): *Matrix Computations*. The Johns Hopkins University Press, London

[11] Hull, D. (1994): *Improving text retrieval for the routing problem using latent semantic indexing*. In: Proceedings of the 17th ACM/SIGIR Conference, pp. 282–290

[12] Hull, D. (1996): *Stemming Algorithms: A case study for detailed evaluation*. Journal of the American Society for Information Science 47(1), pp.70–84

[13] Jessup, E.R., Martin, J.H. (2001): *Taking a new look at the Latent Semantic Analysis approach to Information Retrieval*. In: Proceedings of the SIAM Workshop on Computational Information Retrieval, Raleigh, NC, pp. 121–144

[14] Kolda, T.G., O'Leary, D.P. (1998): *A semidiscrete matrix decomposition for latent semantic indexing information retrieval*. ACM Transactions on Information Systems (TOIS) 16(4), pp. 322–346

[15] Landauer, T.K., Foltz, P., Laham, D. (1998): *Introduction to latent semantic analysis*. Discourse Processes 25, pp. 259–284

[16] Papadimitriou, C.H., Raghavan, P., Tamaki, H., Vempala, S. (1998): *Latent Semantic Inmdexing: A probabilistic analysis*. In: Proceedings of Symposium on Principles of Database Systems (PODS), ACM Press, Seattle, Washington, pp. 150–168

[17] Salton, G. (1971): *The SMART retrieval system: Experiments in Automatic Document Processing*. Prentice Hall Inc., Englewood Cliffs NJ

[18] Schütze, H. (1992): *Dimensions of meaning*. In: Proceedings of Supercomputing '92, Minneapolis MN, pp. 787–796

[19] Schütze, H. (1998): *Automatic word Sense Discrimination*. Computational Linguistics 24, pp. 97–124

[20] Story, R.E. (1996): *An explanation of the effectiveness of Latent Semantic Indexing by means of a Bayesian regression model*. Information Processing and Management, 32(3), pp. 329–344

[21] Zha, H., Marques, O., Simon, H. (1998): *A Subspace-Based Model for Information Retrieval with Applications in Latent Semantic Indexing*. In: Proceedings of Irregular '98, Lecture Notes in Computer Science 1457, Springer-Verlag, pp.29–42

[22] http://www.uspto.gov

[23] http://www.ime.unicamp.br/∼chico/arpack++

[24] ftp://ftp.cs.cornell.edu/pub/smart/english.stop

[25] http://www.tartarus.org/∼martin/PorterStemmer

[26] ftp://ftp.cs.cornell.edu/pub/smart/med

[27] ftp://ftp.cs.cornell.edu/pub/smart/time

US class	**Nr. documents**	**Nr. words**	**Nr. subclasses**
307	4592	15077	26
323	3924	12148	22
329	730	5621	7
330	6097	14702	50
331	4255	13403	60
332	687	5808	8
338	2242	11384	46
343	6369	17135	3
703	2772	21552	6
706	2255	20934	10

Table 1. The dimensions of the matrices

List of figures

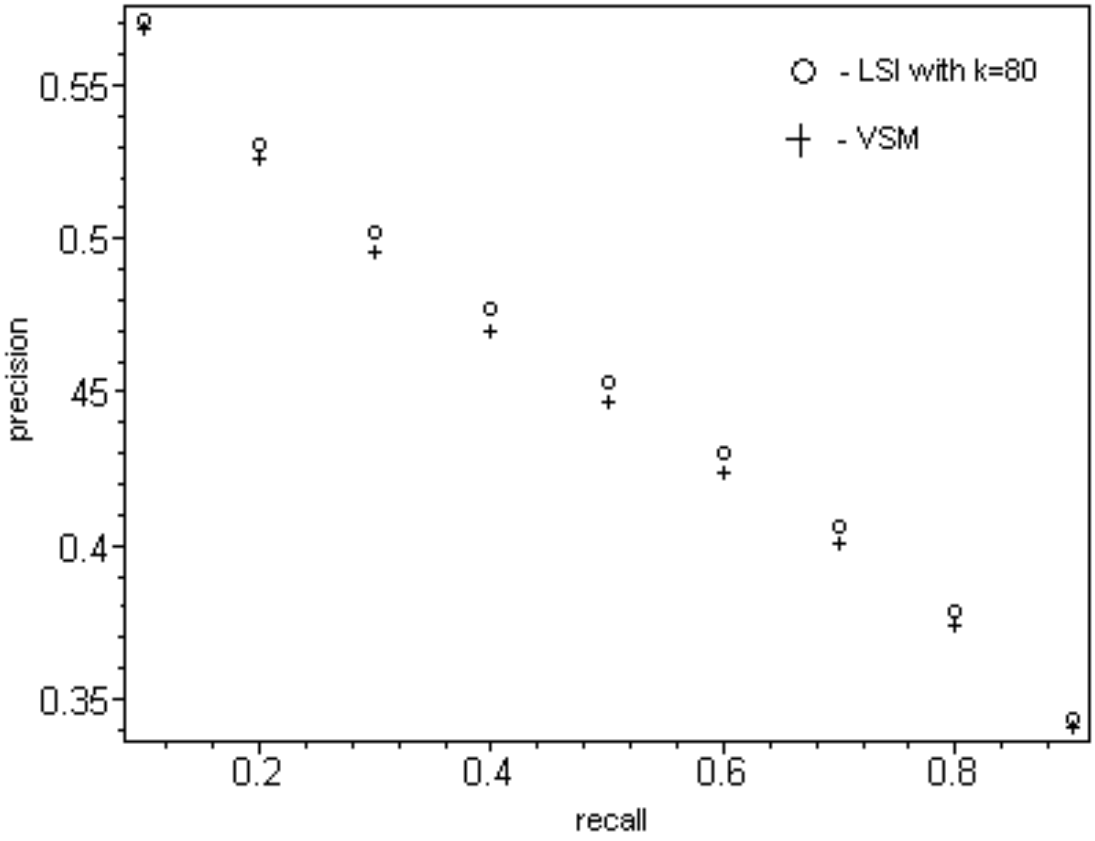
- LSI with k=80
- VSM
precision
0.55
0.5
45
0.4
0.35
0.2
0.4
0.6
0.8
recall

Figure 1.

A. Moldovan

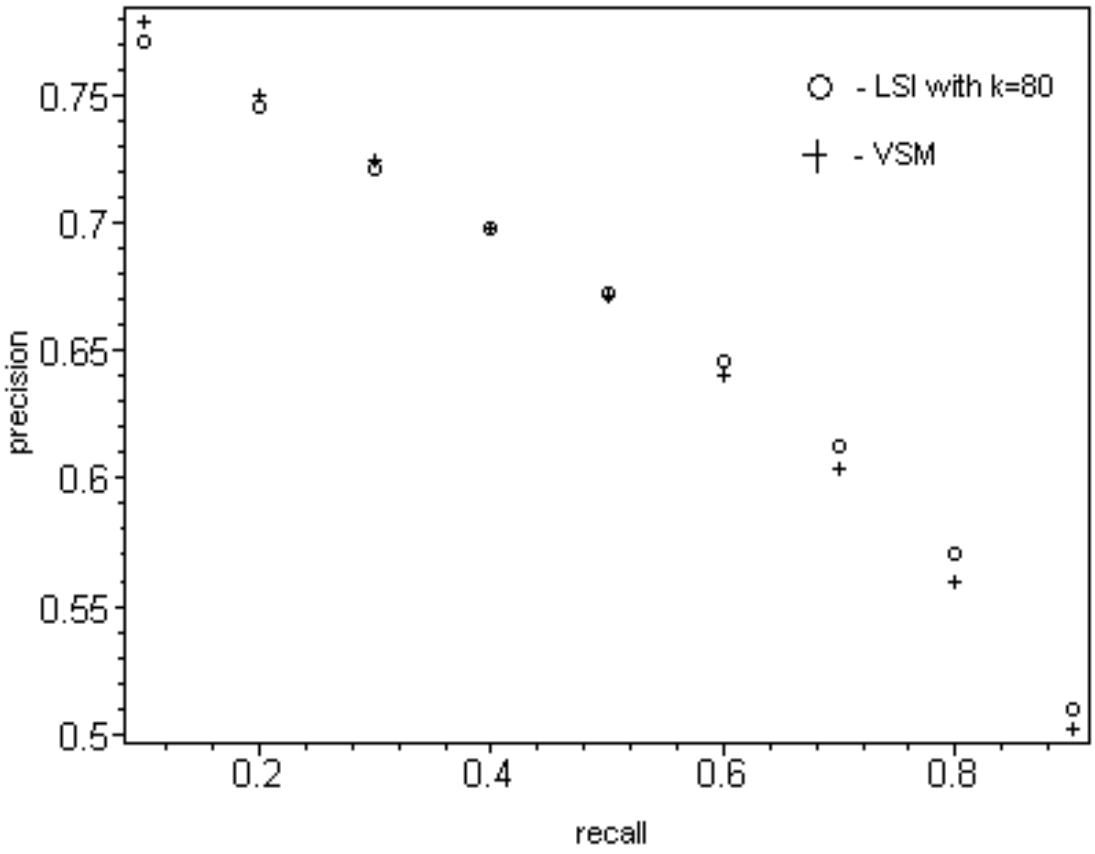
precision
0.75
0.7
0.65
0.6
0.55
0.5
0.2
0.4
0.6
0.8
recall
- LSI with k=80
- VSM

Figure 2.

A. Moldovan

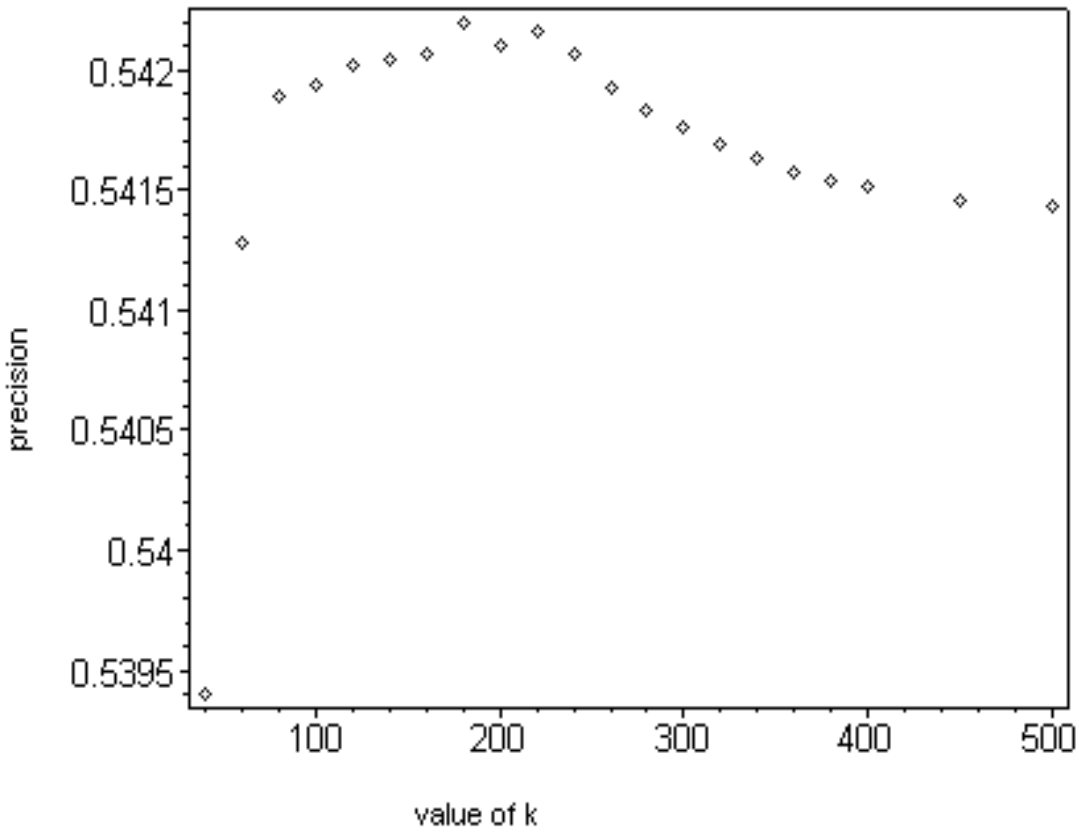
precision
0.542
0.5415
0.541
0.5405
0.54
0.5395
100
200
300
400
500
value of k

Figure 3.

A. Moldovan

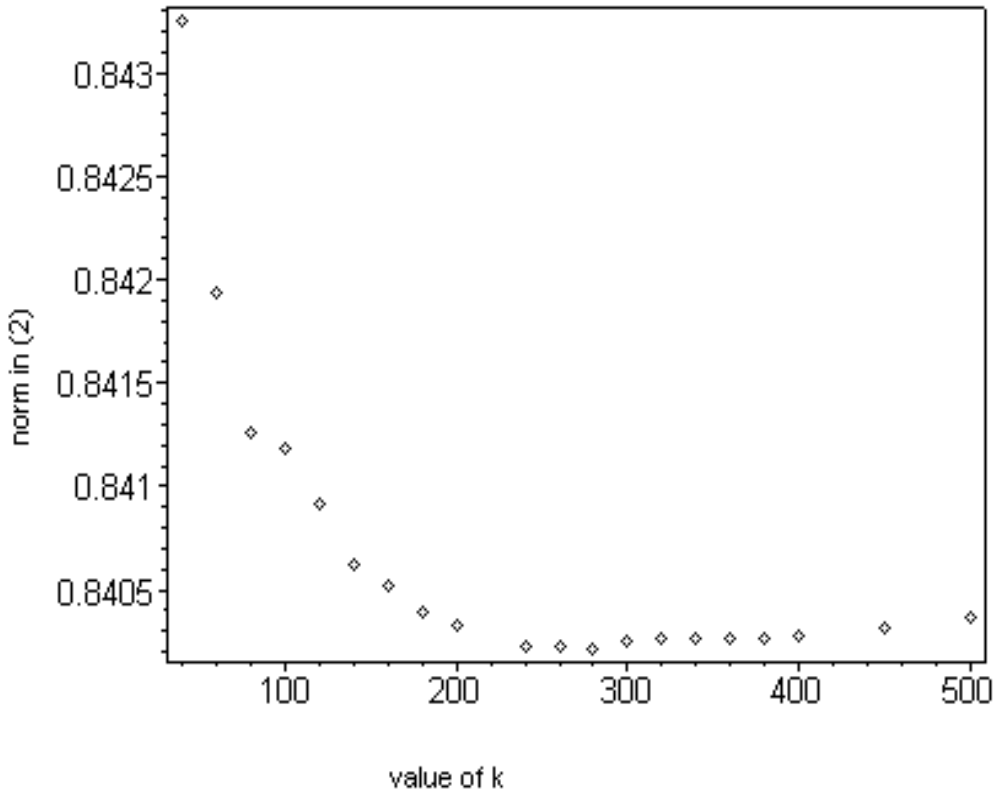
0.843
0.8425
0.842
0.8415
0.841
0.8405
norm in (2)
100
200
300
400
500
value of k

Figure 4.

A. Moldovan

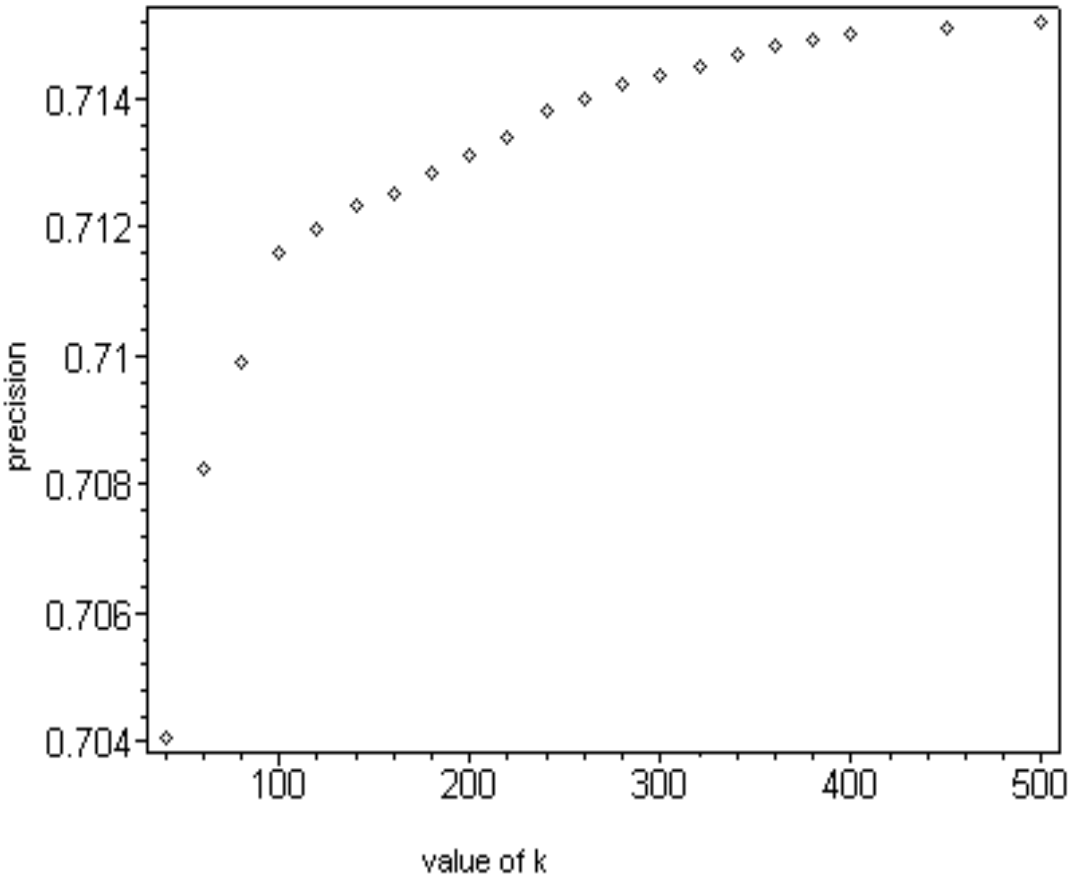
0.714
0.712
0.71
0.708
0.706
0.704
precision
100
200
300
400
500
value of k

Figure 5.

A. Moldovan

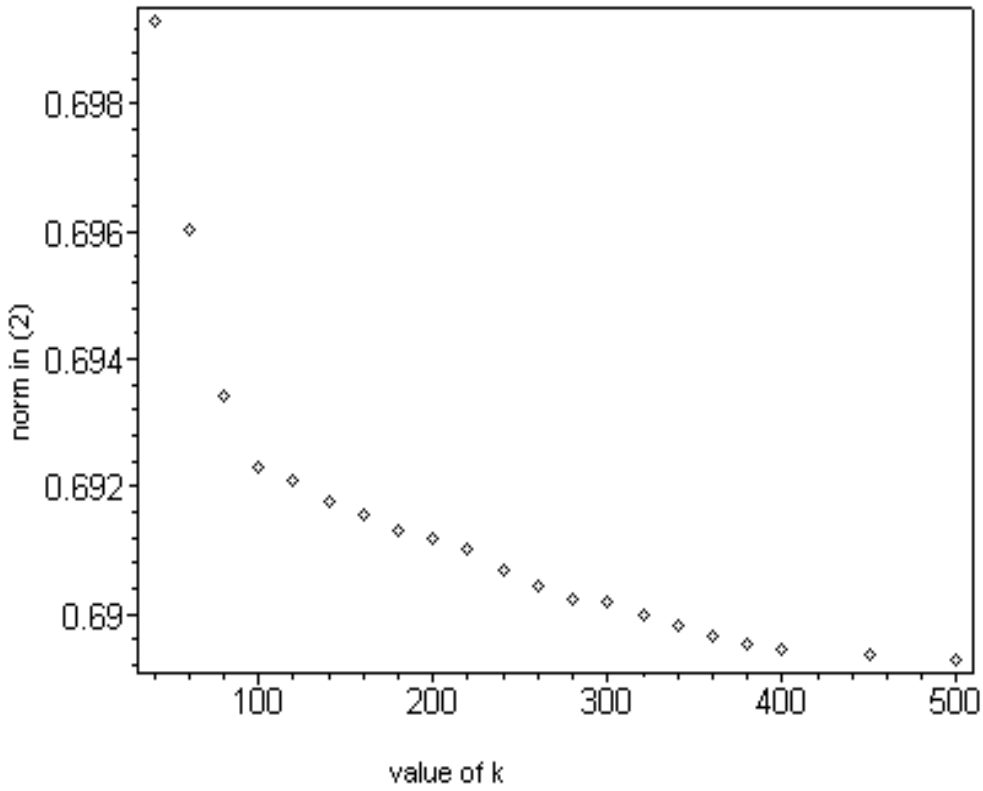
0.698
0.696
0.694
0.692
0.69
norm in (2)
100
200
300
400
500
value of k

Figure 6.

A. Moldovan

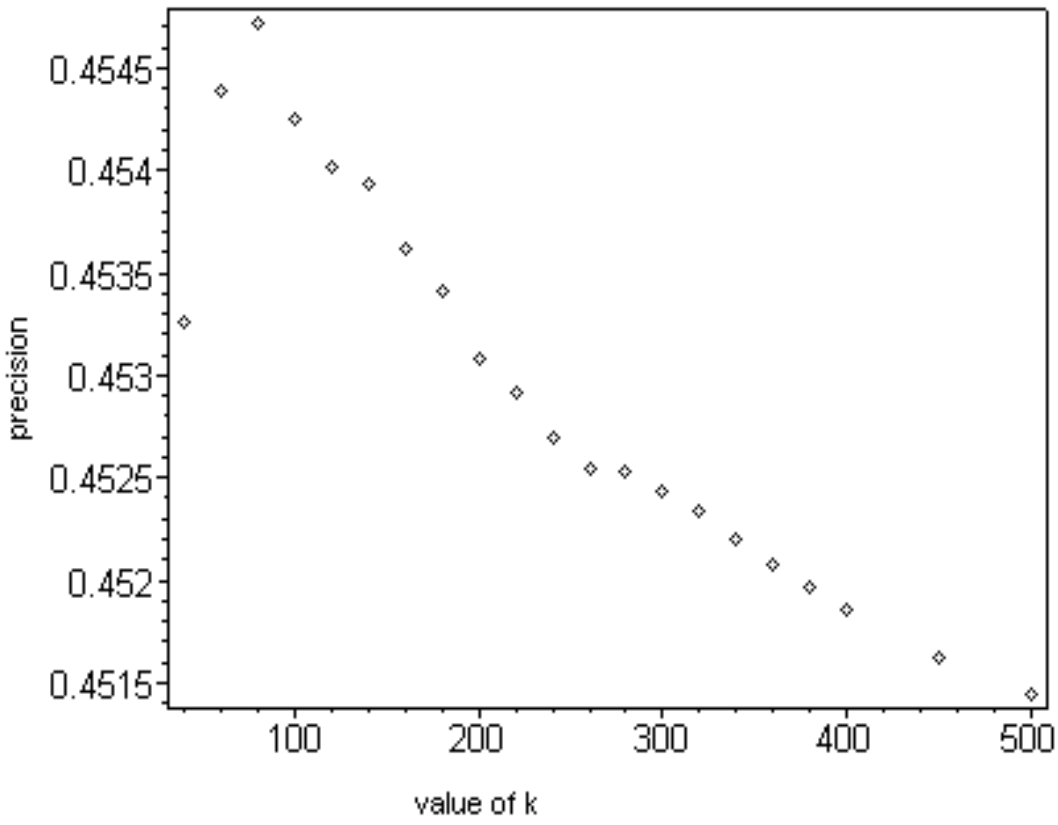
precision
0.4545
0.454
0.4535
0.453
0.4525
0.452
0.4515
100
200
300
400
500
value of k

Figure 7.

A. Moldovan

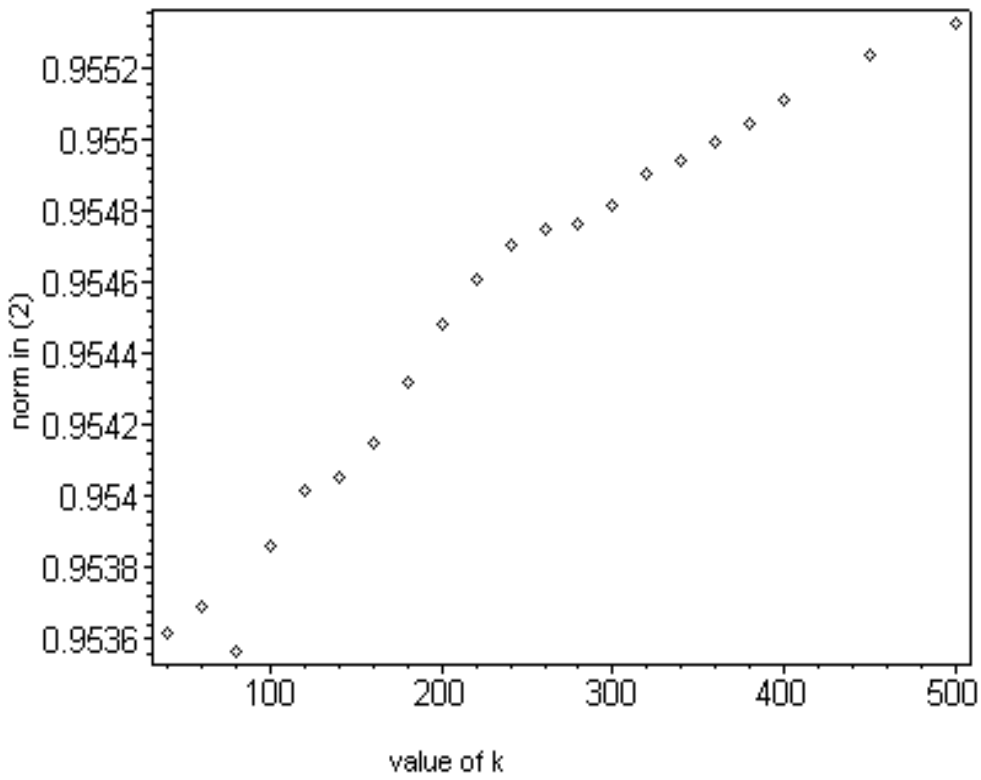
0.9552
0.955
0.9548
0.9546
0.9544
0.9542
0.954
0.9538
0.9536
norm in (2)
100
200
300
400
500
value of k

Figure 8.

A. Moldovan