A Framework for Understanding Latent Semantic Indexing (LSI) Performance

April Kontostathis [a] and William M. Pottenger [b]

[a] *Ursinus College, PO Box 1000, 601 Main Street, Collegeville, PA 19426*

[b] *Lehigh University, 19 Memorial Drive West, Bethlehem, PA 18015*

Abstract

In this paper we present a theoretical model for understanding the performance of Latent Semantic Indexing (LSI) search and retrieval applications. Many models for understanding LSI have been proposed. Ours is the first to study the values produced by LSI in the term by dimension vectors. The framework presented here is based on term co-occurrence data. We show a strong correlation between second-order term co-occurrence and the values produced by the Singular Value Decomposition (SVD) algorithm that forms the foundation for LSI. We also present a mathematical proof that the SVD algorithm encapsulates term co-occurrence information.

Key words: Latent Semantic Indexing, Term Co-occurrence, Singular Value Decomposition, Information Retrieval Theory

1 Introduction

Latent Semantic Indexing (LSI) (Deerwester et al., 1990) is a well-known information retrieval algorithm. LSI has been applied to a wide variety of learning tasks, such as search and retrieval (Deerwester et al., 1990), classification (Zelikovitz and Hirsh, 2001) and filtering (Dumais, 1994, 1995). LSI is a vector space approach for modeling documents, and many have claimed that the technique brings out the 'latent' semantics in a collection of documents (Deerwester et al., 1990; Dumais, 1992).

LSI is based on a mathematical technique termed Singular Value Decomposition (SVD). The algebraic foundation for LSI was first described in (Deer-

Email addresses: `akontostathis@ursinus.edu` (April Kontostathis), `billp@cse.lehigh.edu` (William M. Pottenger).

wester et al., 1990) and has been further discussed in (Berry et al., 1995, 1999). These papers describe the SVD process and interpret the resulting matrices in a geometric context. The SVD, truncated to k dimensions, gives the best rank-k approximation to the original matrix. (Wiemer-Hastings, 1999) shows that the power of LSI comes primarily from the SVD algorithm.

Other researchers have proposed theoretical approaches to understanding LSI. (Zha and Simon, 1998) describes LSI in terms of a subspace model and proposes a statistical test for choosing the optimal number of dimensions for a given collection. (Story, 1996) discusses LSI's relationship to statistical regression and Bayesian methods. (Ding, 1999) constructs a statistical model for LSI using the cosine similarity measure.

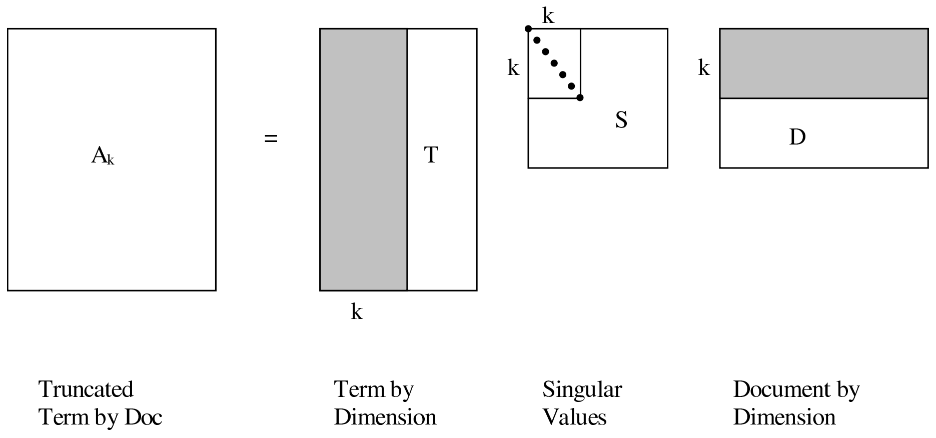
Although other researchers have explored the SVD algorithm to provide an understanding of SVD-based information retrieval systems, to our knowledge, only Schütze has studied the values produced by SVD (Schütze, 1992). We expand upon this work, showing here that LSI exploits higher-order term co-occurrence in a collection. We provide a mathematical proof of this fact herein, thereby providing an intuitive theoretical understanding of the mechanism whereby LSI emphasizes latent semantics.

This work is also the first to study the values produced in the SVD term by dimension matrix and we have discovered a correlation between the performance of LSI and the values in this matrix. Thus, in conjunction with the aforementioned proof of LSI's theoretical foundation on higher-order co-occurrences, we have discovered the basis for the claim that is frequently made for LSI: LSI emphasizes underlying semantic distinctions (latent semantics) while reducing noise in the data. This is an important component in the theoretical basis for LSI.

Additional related work can be found in a recent article by Dominich. In (Dominich, 2003), the author shows that term co-occurrence is exploited in the connectionist interaction retrieval model, and this can account for or contribute to its effectiveness.

In Section 2 we present an overview of LSI along with a simple example of higher-order term co-occurrence in LSI. Section 3 explores the relationship between the values produced by LSI and term co-occurrence. In Sections 3 and 4 we correlate LSI performance to the values produced by the SVD, indexed by the order of co-occurrence. Section 5 presents a mathematical proof of LSI's basis on higher-order co-occurrence. We draw conclusions and touch on future work in Section 6.

Fig. 1. Truncation of SVD for LSI



2 Overview of Latent Semantic Indexing

In this section we provide a brief overview of the LSI algorithm. We also discuss higher-order term co-occurrence in LSI, and present an example of LSI assignment of term co-occurrence values in a small collection.

2.1 Latent Semantic Indexing Algorithm

In traditional vector space retrieval, documents and queries are represented as vectors in t-dimensional space, where t is the number of indexed terms in the collection. When Latent Semantic Indexing (LSI) is used for document retrieval, Singular Value Decomposition (SVD) is used to decompose the term by document matrix into three matrices: T, a term by dimension matrix, S a singular value matrix (dimension by dimension), and D, a document by dimension matrix. The number of dimensions is r, the rank of the term by document matrix. The original matrix can be obtained, through matrix multiplication of TSD^T.

In an LSI system, the T, S and D matrices are truncated to k dimensions. This process is presented graphically in Figure 1 (taken from (Berry et al., 1995)). The shaded areas in the T and D matrices are kept, as are the k largest singular values, the non-shaded areas are removed. The purpose of dimensionality reduction is to reduce 'noise' in the latent space, resulting in a richer word relationship structure that reveals latent semantics present in the collection. (Berry et al., 1995) discusses LSI processing in detail and provides a numerical example. Queries are represented in the reduced space by $T_k^T q$, where T_k^T is the transpose of the term by dimension matrix, after truncation to k dimensions. Queries are compared to the reduced document vectors, scaled

Table 1
Deerwester Term by Document Matrix

	c1	**c2**	**c3**	**c4**	**c5**	**m1**	**m2**	**m3**	**m4**
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

by the singular values ($S_k D_k$) by computing the cosine similarity.

LSI relies on a parameter k, for dimensionality reduction. The optimal k is determined empirically for each collection. In general, smaller k values are preferred when using LSI, due to the computational cost associated with the SVD algorithm, as well as the cost of storing and comparing large dimension vectors.

In the next section we apply LSI to a small collection which consists of only nine documents, and show how LSI transforms values in the term by document matrix.

2.2 Co-occurrence in LSI - An Example

The data for the following example is taken from (Deerwester et al., 1990). In that paper, the authors describe an example with 12 terms and 9 documents. The term-document matrix is shown in Table 1 and the corresponding term-to-term matrix is shown in Table 2.

As mentioned above, the SVD process used by LSI decomposes the matrix into three matrices: T, a term by dimension matrix, S a singular value matrix, and D, a document by dimension matrix. The reader is referred to (Deerwester et al., 1990) for the T, S, and D matrices corresponding to the term by document matrix in Table 1. After dimensionality reduction the term-to-term matrix can be re-computed using the formula $T_k S_k (T_k S_k)^T$. The term-to-term matrix, after reduction to 2 dimensions, is shown in Table 3.

We will assume that the value in position (i,j) of the matrix represents the similarity between term i and term j in the collection. As can be seen in Table 3, *user* and *human* now have a value of .94, representing a strong similarity,

Table 2
Deerwester Term-to-Term Matrix

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
human (t1)	x	1	1	0	2	0	0	1	0	0	0	0
interface (t2)	1	x	1	1	1	0	0	1	0	0	0	0
computer (t3)	1	1	x	1	1	1	1	0	1	0	0	0
user (t4)	0	1	1	x	2	2	2	1	1	0	0	0
system (t5)	2	1	1	2	x	1	1	3	1	0	0	0
response (t6)	0	0	1	2	1	x	2	0	1	0	0	0
time (t7)	0	0	1	2	1	2	x	0	1	0	0	0
EPS (t8)	1	1	0	1	3	0	0	x	0	0	0	0
survey (t9)	0	0	1	1	1	1	1	0	x	0	1	1
trees (t10)	0	0	0	0	0	0	0	0	0	x	2	1
graph (t11)	0	0	0	0	0	0	0	0	1	2	x	2
minors (t12)	0	0	0	0	0	0	0	0	1	1	2	x

Table 3
Deerwester Term-to-Term Matrix, Truncated to two dimensions

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
human (t1)	x	0.54	0.56	0.94	1.69	0.58	0.58	0.84	0.32	-0.32	-0.34	-0.25
interface (t2)	0.54	x	0.52	0.87	1.50	0.55	0.55	0.73	0.35	-0.20	-0.19	-0.14
computer (t3)	0.56	0.52	x	1.09	1.67	0.75	0.75	0.77	0.63	0.15	0.27	0.20
user (t4)	0.94	0.87	1.09	x	2.79	1.25	1.25	1.28	1.04	0.23	0.42	0.31
system (t5)	1.69	1.50	1.67	2.79	x	1.81	1.81	2.30	1.20	-0.47	-0.39	-0.28
response (t6)	0.58	0.55	0.75	1.25	1.81	x	0.89	0.80	0.82	0.38	0.56	0.41
time (t7)	0.58	0.55	0.75	1.25	1.81	0.89	x	0.80	0.82	0.38	0.56	0.41
EPS (t8)	0.84	0.73	0.77	1.28	2.30	0.80	0.80	x	0.46	-0.41	-0.43	-0.31
survey (t9)	0.32	0.35	0.63	1.04	1.20	0.82	0.82	0.46	x	0.88	1.17	0.85
trees (t10)	-0.32	-0.20	0.15	0.23	-0.47	0.38	0.38	-0.41	0.88	x	1.96	1.43
graph (t11)	-0.34	-0.19	0.27	0.42	-0.39	0.56	0.56	-0.43	1.17	1.96	x	1.81
minors (t12)	-0.25	-0.14	0.20	0.31	-0.28	0.41	0.41	-0.31	0.85	1.43	1.81	x

where before the value was zero. In fact, *user* and *human* is an example of second-order co-occurrence. The relationship between *user* and *human* comes from the transitive relation: *user* co-occurs with *interface* and *interface* co-occurs with *human*.

A closer look reveals a value of 0.15 in the relationship between *trees* and *computer*. Looking at the co-occurrence path gives us an explanation as to why these terms received a positive (albeit weak) similarity value. From Table 2, we see that *trees* co-occurs with *graph*, *graph* co-occurs with *survey*, and *survey* co-occurs with *computer*. Hence the *trees/computer* relationship is an example of third-order co-occurrence. In Section 3 we explore the relationship between higher-order term co-occurrence and the values which appear in the term-to-term matrix, and in Section 4 we present correlation data that confirms the relationship between the term-to-term matrix values and the performance of LSI.

To completely understand the dynamics of the SVD process, a closer look at Table 1 is warranted. The nine documents in the collection can be split into

Table 4
Modified Deerwester Term-to-Term Matrix, Truncated to two dimensions

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
human (t1)	x	0.50	0.60	1.01	1.62	0.66	0.66	0.76	0.45	–	–	–
interface (t2)	0.50	x	0.53	0.90	1.45	0.59	0.59	0.68	0.40	–	–	–
computer (t3)	0.60	0.53	x	1.08	1.74	0.71	0.71	0.81	0.48	–	–	–
user (t4)	1.01	0.90	1.08	x	2.92	1.19	1.19	1.37	0.81	–	–	–
system (t5)	1.62	1.45	1.74	2.92	x	1.91	1.91	2.20	1.30	–	–	–
response (t6)	0.66	0.59	0.71	1.19	1.91	x	0.78	0.90	0.53	–	–	–
time (t7)	0.66	0.59	0.71	1.19	1.91	0.78	x	0.90	0.53	–	–	–
EPS (t8)	0.76	0.68	0.81	1.37	2.20	0.90	0.90	x	0.61	–	–	–
survey (t9)	0.45	0.40	0.48	0.81	1.30	0.53	0.53	0.61	x	–	–	–
trees (t10)	–	–	–	–	–	–	–	–	–	x	2.37	1.65
graph (t11)	–	–	–	–	–	–	–	–	–	2.37	x	1.91
minors (t12)	–	–	–	–	–	–	–	–	–	1.65	1.91	x

two subsets C1-C5 and M1-M4. If the term *survey* did not appear in the M1-M4 subset, the subsets would be disjoint. The data in Table 4 was developed by changing the $survey/m4$ entry to 0 in Table 1, computing the SVD of this new matrix, truncating to two dimensions, and deriving the associated term-to-term matrix.

The terms are now segregated; all values between the $trees, graph, minors$ subset and the rest of the terms have been reduced to zero. In Section 5 we prove a theorem that explains this phenomenon, showing, in all cases, that if there is no connectivity path between two terms, the resultant value in the term-to-term matrix must be zero.

3 Higher-order Co-occurrence in LSI

In this section we study the relationship between the values produced by LSI and term co-occurrence. We show a relationship between the term co-occurrence patterns and resultant LSI similarity values. This data shows how LSI emphasizes important semantic distinctions, while de-emphasizing terms that co-occur frequently with many other terms (reduces 'noise'). A full understanding of the relationship between higher-order term co-occurrence and the values produced by SVD is a necessary step toward the development of an approximation algorithm for LSI.

3.1 Data Sets

We chose the MED, CRAN, CISI and LISA collections for our study of the higher-order co-occurrence in LSI. Table 5 shows the attributes of each collection.

Table 5
Data Sets Used for Analysis

Identifier	**Description**	**Docs**	**Terms**
MED	Medical Abstracts	1033	5831
CISI	Information Science Abstracts	1450	5143
CRAN	Cranfield Collection	1398	3932
LISA Words	Library and Information Science Abstracts	6004	18429
LISA Noun Phrase	Library and Information Science Abstracts	5998	81,879

The LISA collection was processed in two ways. The first was an extraction of words only, resulting in a collection with 6004 documents and 18429 terms. We will refer to this collection as LISA Words. Next the HDDI Collection Builder (Pottenger and Yang, 2001; Pottenger et al., 2001) was used to extract maximal length noun phrases. This collection contains 5998 documents (no noun phrases were extracted from several short documents) and 81,879 terms. The experimental results for the LISA Noun Phrase collection were restricted to 1000 randomly chosen terms (due to processing time considerations). However, for each of the 1000 terms, all co-occurring terms (up to 81,879) were processed, giving us confidence that this data set accurately reflects the scalability of our result. In the future, we plan to expand this analysis to larger data sets, such as those used for the Text Retrieval Conference (TREC) experiments.

3.2 Methodology

Our experiments captured four main features of these data sets: the order of co-occurrence for each pair of terms in the truncated term-to-term matrix (shortest path length), the order of co-occurrence for each pair of terms in the original (not truncated) matrix, the distribution of the similarity values produced in the term-to-term matrix, categorized by order of co-occurrence, and the number of second-order co-occurrence paths between each set of terms.

In order to complete these experiments, we needed a program to perform the SVD decomposition. The SVDPACK suite (Berry et al., 1993) that provides eight algorithms for decomposition was selected because it was readily available, as well as thoroughly tested. The singular values and vectors were input into our algorithm.

Table 6
Order of Co-occurrence Summary Data (k=100 for all Collections)

Collection	First	Second	Third	Fourth	Fifth	Sixth
MED Truncated	1,110,485	15,867,200	17,819	-	-	
MED Original	1,110,491	15,869,045	17,829	-	-	
CRAN Truncated	2,428,520	18,817,356	508	-	-	
CRAN Original	2,428,588	18,836,832	512	-	-	
CISI Truncated	2,327,918	29,083,372	17,682	-	-	
CISI Original	2,328,026	29,109,528	17,718	-	-	
LISA Words Truncated	5,380,788	308,556,728	23,504,606	-	-	
LISA Words Original	5,399,343	310,196,402	24,032,296	-	-	
LISA Noun Phrase Truncated	51,350	10,976,417	65,098,694	1,089,673	3	-
LISA Noun Phrase Original	51,474	11,026,553	68,070,600	2,139,117	15,755	34

3.3 Results

The order of co-occurrence summary for all of the collections is shown in Table 6. Fifth order co-occurrence was the highest order observed in the truncated matrices. In is interesting that the noun phrase collection is the only collection that resulted in a co-occurrence order higher than three. The order-two and order-three co-occurrences significantly reduce the sparsity of the original data. The lines labeled 'Collection' Original indicate the number of pairs with co-occurrence order n determined from a trace of the original term-to-term matrix. LSI incorporates over 99% of the (higher-order) term co-occurrences present in the data for the first four collections, and 95% for the LISA Noun Phrase collection.

Table 7 shows the weight distribution for the LISA Words data, for k=100. MED and CISI showed similar trends. This data provides insight into understanding the values produced by SVD in the truncated term-to-term matrix. The degree-two pair weights range from -0.3 to values larger than 8. These co-occurrences will result in significant differences in document matching when the LSI algorithm is applied in a search and retrieval application. However, the weights for the degree-three pairs are between -0.2 and 0.1, adding a relatively small degree of variation to the final results. Many of the original term-to-term co-occurrences (degree-one pairs) are reduced to nearly zero, while others are significantly larger.

Table 7 also shows the average number of paths by term-to-term value range for LISA Words. Clearly the degree-two pairs that have a similarity close to zero have a much smaller average number of paths than the pairs with either higher negative or positive similarities. The degree-one pairs with higher average number of paths tend to have negative similarity values, pairs with fewer co-occurrences paths tend to receive low similarity values, and pairs with a moderate number of co-occurrence paths tend to receive high similarity values. This explains how LSI emphasizes important semantic distinctions,

Table 7
Average Number of paths by term-term value for LISA Words, k=100

Term Term Matrix Value	Degree 1 Pairs	Degree 2 Pairs	Degree 3 Pairs	Degree 2 Paths	Average No. Paths for Deg 1 pairs	Average No. Paths for Deg 2 pairs	Average No. Paths for Deg 3 pairs
less than -0.2	21,946	186,066	-	66,323,200	3,022	356	-
-0.2 to -0.1	10,012	422,734	2	59,418,198	5,935	141	29,709,099
-0.1 to 0.0	76,968	127,782,170	18,398,756	1,587,147,584	20,621	12	86
0.0 to 0.1	1,670,542	175,021,904	5,105,848	4,026,560,130	2,410	23	789
0.1 to 0.2	662,800	3,075,956	-	721,472,948	1,089	235	-
0.2 to 0.3	418,530	974,770	-	389,909,456	932	400	-
0.3 to 0.4	320,736	439,280	-	259,334,214	809	590	-
0.4 to 0.5	309,766	232,584	-	195,515,980	631	841	-
0.5 to 0.6	241,466	136,742	-	151,687,510	628	1,109	-
0.6 to 0.7	158,210	85,472	-	117,150,688	740	1,371	-
0.7 to 0.8	128,762	56,042	-	96,294,828	748	1,718	-
0.8 to 0.9	113,826	38,156	-	81,799,460	719	2,144	-
0.9 to 1.0	119,440	25,958	-	72,273,400	605	2,784	-
1.0 to 2.0	547,354	70,616	-	393,001,792	718	5,565	-
2.0 to 3.0	208,238	6,678	-	172,335,854	828	25,807	-
3.0 to 4.0	105,332	1,112	-	98,575,368	936	88,647	-
4.0 to 5.0	62,654	334	-	64,329,366	1,027	192,603	-
5.0 to 6.0	40,650	78	-	45,174,210	1,111	579,157	-
6.0 to 7.0	28,264	36	-	33,514,804	1,186	930,967	-
7.0 to 8.0	21,316	24	-	26,533,666	1,245	1,105,569	-
over 8.0	113,976	16	-	188,614,174	1,655	11,788,386	-

while de-emphasizing terms that co-occur frequently with many other terms (reduces 'noise'). On the other hand, degree-two pairs with many paths of connectivity tend to receive high similarity values, while those with a moderate number tend to receive negative values. These degree-two pairs with high values can be considered the 'latent semantics' that are emphasized by LSI.

4 Analysis of the LSI Values

In this section we expand upon the work described in Section 3. The results of our analysis show a strong correlation between the values produced by LSI and higher-order term co-occurrences.

4.1 Data Sets

We chose six collections for our study of the values produced by LSI, the four collections used in Section 3, and two additional collections, CACM and NPL. These collections are described in Table 8. These collections were chosen because they have query and relevance judgment sets that are readily available.

Table 8
Data Sets Used for Analysis

Identifier	**Description**	**Docs**	**Terms**	**Queries**	**Optimal k**	**Values for k used in the study**
MED	Medical Abstracts	1033	5831	30	40	10, 25, 40, 75, 100, 125, 150, 200
CISI	Information Science Abstracts	1450	5143	76	40	10, 25, 40, 75, 100, 125, 150, 200
CACM	Communications of the ACM Abstracts	3204	4863	52	70	10, 25, 50, 70, 100, 125, 150, 200
CRAN	Cranfield Collection	1398	3931	225	50	10, 25, 50, 75, 100, 125, 150, 200
LISA	Library and Information Science Abstracts	6004	18429	35	165	10, 50, 100, 150, 165, 200, 300, 500
NPL	Larger collection of very short documents	11429	6988	93	200	10, 50, 100, 150, 200, 300, 400, 500

The Parallel General Text Parser (PGTP) (Martin and Berry, 2004) was used to preprocess the text data, including creation and decomposition of the term document matrix. For our experiments, we applied the log entropy weighting option and normalized the document vectors. We chose this weighting scheme because it is commonly used for information retrieval applications. We plan to expand our analysis to include other weighting schemes in the future.

We were interested in the distribution of values for both optimal and sub-optimal parameters for each collection. In order to identify the most effective k (dimension truncation parameter) for LSI, we used the F_{beta}, a combination of precision and recall (van Rijsbergen, 1979), as a determining factor. The equation for F_{beta} is shown in (1). The *beta* parameter allows us to place greater or lesser emphasis on precision, depending on our needs. In our experiments we used *beta*=1 for the F_{beta} parameter, which balances precision and recall. We explored possible values from k=10, incrementing by five, up to k=200 for the smaller collections and values up to k=500 for the LISA and NPL collections. For each value of k, precision and recall averages were identified for each rank from 10 to 100 (incrementing by 10), and the resulting F_{beta} was calculated.

$$F_{beta} = \frac{(beta^2 + 1)(Precision)(Recall)}{beta^2(Precision) + (Recall)} \tag{1}$$

The results of these runs for selected values of k are summarized in Figures 2 and 3. To choose the optimal k, we selected the smallest value that provided substantially similar performance as larger k values. For example, in the LISA collection k=165 was chosen as the optimum because the k values higher than 165 provide only slighter better performance. A smaller k is preferable to reduce the computational overhead of both the decomposition and the search and retrieval processing. The optimal k was identified for each collection and is shown in Table 8.

Fig. 2. LISA Performance for LISA and NPL

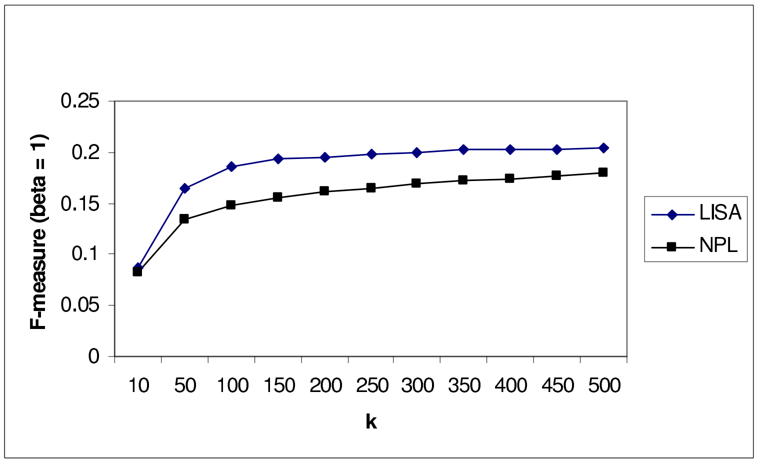
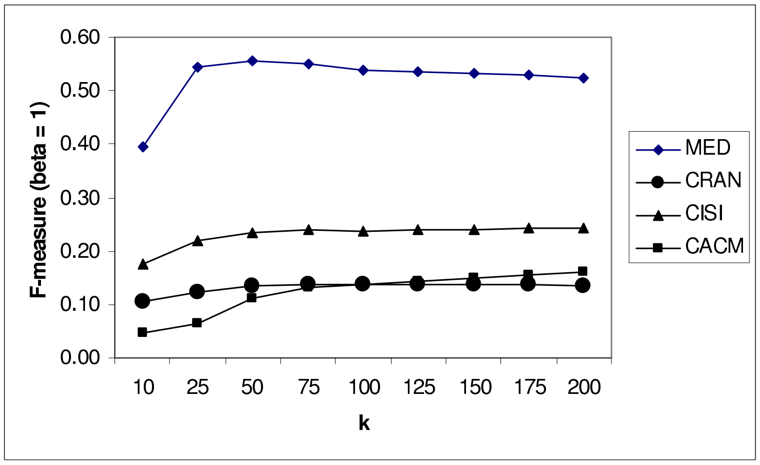


Fig. 3. LISA Performance for Smaller Collections



4.2 Methodology

The algorithm used to collect the co-occurrence data appears in Figure 4. After we compute the SVD using the original term by document matrix, we calculate term-to-term similarities. LSI provides two natural methods for describing term-to-term similarity. First, the term-to-term matrix can be created using $T_kS_k(T_kS_k)^T$. This approach results in values such as those shown in Table 3. Second, the term by dimension (T_kS_k) matrix can be used to compare terms using a vector distance measure, such as cosine similarity. In this case, the cosine similarity is computed for each pair of rows in the T_kS_k matrix. This computation results in a value in the range [-1, 1] for each pair of terms (i, j).

Fig. 4. Algorithm for Data Collection

```
Create the term by document matrix
Compute the SVD for the matrix
For each pair of terms (i,j) in the collection
    Compute the term-to-term matrix value for the (i,j) element after truncation
        to k dimensions
    Compute the cosine similarly value for the (i,j) element after truncation to k
        dimensions
    Determine the 'order of co-occurrence'
Summarize the data by range of values and order of co-occurrence
```

After the term similarities are created, we need to determine the order of co-occurrence for each pair of terms. The order of co-occurrence is computed by tracing the co-occurrence paths. In Figure 5 we present an example of this process. In this small collection, terms A and B appear in document D1, terms B and C appear in document D2, and terms C and D occur in document D3. If each term is considered a node in a graph, arcs can be drawn between the terms that appear in the same document. Now we can assign order of co-occurrence as follows: nodes that are connected are considered first-order pairs, nodes that can be reached with one intermediate hop are second-order co-occurrences, nodes that can be reached with two intermediate hops are third-order pairs, etc. In general the order of co-occurrence is $n + 1$, where n is the number of hops needed to connect the nodes in the graph. Some term pairs may not have a connecting path; the LSI term-to-term matrix for this situation is exemplified in Table 4 in which entries are zero for terms that do not have a connectivity path. The number of paths of co-occurrence can also be derived by successively multiplying the original term-to-term matrix (e.g. using Table 2), with itself. If A is the original term-to-term matrix, the values in the (i, j) entry of A^n will represent the number of paths of length n between term i and term j.

4.3 Results

The order of co-occurrence summary for the NPL collection is shown in Table 9. The values are expressed as a percentage of the total number of pairs of first-, second- and third-order co-occurrences for each collection. The values in Table 9 represent the distribution using the cosine similarity. LSI performance is also shown.

Table 10 shows the correlation coefficient for all collections. There is a strong correlation between the percentage of second-order negative values and LSI

Fig. 5. Tracing order of co-occurrence

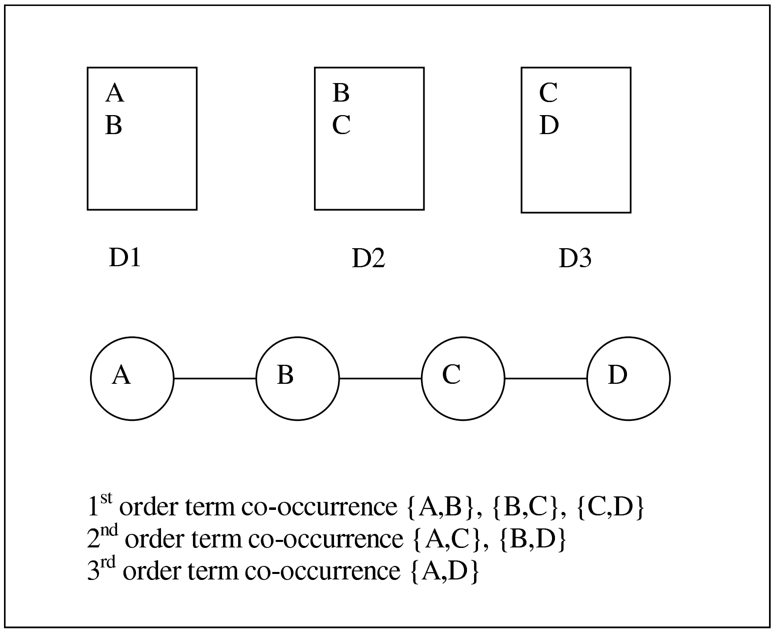

A
B
B
C
C
D
D1
D2
D3
A
B
C
D
1st order term co-occurrence {A,B}, {B,C}, {C,D}
2nd order term co-occurrence {A,C}, {B,D}
3rd order term co-occurrence {A,D}

Table 9
Distribution summary by sign and order of co-occurrence for NPL

1st Order

	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
[-1.0,-.01]	1.4%	2.5%	3.0%	3.1%	3.1%	2.8%	2.2%	1.6%
(-.01,.01)	0.5%	1.7%	2.9%	4.1%	5.0%	6.7%	8.1%	9.4%
[.01, 1.0]	98.1%	95.7%	94.1%	92.8%	91.8%	90.5%	89.7%	89.0%

2nd Order

	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
[-1.0,-.01]	14.0%	24.6%	28.2%	30.1%	31.2%	32.1%	32.2%	31.8%
(-.01,.01)	2.4%	6.7%	9.9%	12.4%	14.7%	18.9%	22.7%	26.4%
[.01, 1.0]	83.6%	68.6%	61.9%	57.5%	54.2%	49.0%	45.1%	41.7%

3rd Order

	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
[-1.0,-.01]	44.6%	62.7%	66.9%	69.3%	70.0%	69.7%	68.3%	66.6%
(-.01,.01)	3.9%	8.8%	11.6%	13.5%	15.1%	18.1%	21.0%	23.6%
[.01, 1.0]	51.5%	28.5%	21.6%	17.2%	14.9%	12.2%	10.7%	9.8%

LSI Performance $Beta$=1

	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
F_{beta}	0.08	0.13	0.15	0.15	0.16	0.17	0.17	0.18

performance for all collections, with the correlations for MED appearing slightly weaker than the other collections. There also appears to be a strong inverse correlation between the positive third-order values and the performance of LSI. In general the values for each order of co-occurrence/value pair appear to be consistent across all collections, with the exception of the third-order negative values for CACM.

Table 10
Correlation Coefficients for Cosine Similarity, all Collections

NPL				CISI			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.38	0.99	0.92	[-1.0,-.01]	0.77	0.95	0.99
(-.01,.01)	0.88	0.89	0.93	(-.01,.01)	0.86	0.82	0.81
[.01, 1.0]	-0.95	-0.98	-1.00	[.01, 1.0]	-0.95	-0.92	-0.98
LISA				**MED**			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.55	0.99	0.99	[-1.0,-.01]	0.71	0.76	0.80
(-.01,.01)	0.79	0.77	0.84	(-.01,.01)	0.52	0.48	0.52
[.01, 1.0]	-0.93	-0.92	-0.97	[.01, 1.0]	-0.68	-0.66	-0.74
CRAN				**CACM**			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.91	0.94	0.97	[-1.0,-.01]	0.99	0.98	-0.21
(-.01,.01)	0.77	0.76	0.78	(-.01,.01)	0.92	0.93	0.93
[.01, 1.0]	-0.88	-0.88	-0.96	[.01, 1.0]	-0.96	-0.96	-0.94

Table 11
Correlation Coefficients for Term-to-Term Similarity, all Collections

NPL				CISI			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.90	0.99	0.99	[-1.0,-.01]	0.89	0.76	0.72
(-.01,.01)	-0.87	-0.99	-0.99	(-.01,.01)	-0.87	-0.77	-0.72
[.01, 1.0]	-0.90	-0.99	-0.99	[.01, 1.0]	0.84	0.82	0.68
LISA				**MED**			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.90	0.95	0.99	[-1.0,-.01]	0.86	0.88	0.82
(-.01,.01)	-0.89	-0.95	-1.00	(-.01,.01)	-0.96	-0.88	-0.82
[.01, 1.0]	-0.89	-0.95	-0.97	[.01, 1.0]	0.96	0.86	0.87
CRAN				**CACM**			
	1st Order	2nd Order	3rd Order		1st Order	2nd Order	3rd Order
[-1.0,-.01]	0.90	0.75	0.59	[-1.0,-.01]	0.98	0.91	0.93
(-.01,.01)	-0.86	-0.73	-0.59	(-.01,.01)	-0.95	-0.88	-0.92
[.01, 1.0]	0.81	-0.87	0.46	[.01, 1.0]	0.94	0.62	0.90

The corresponding data using the term-to-term similarity as opposed to the cosine similarity is shown in Table 11. In this data we observe consistent correlations for negative and zero values across all collections, but there are major variances in the correlations for the positive values.

Table 12 shows the values when the correlation coefficient is computed for selected ranges of the cosine similarity, without taking order of co-occurrence into account. Again we note strong correlations for all collections for value ranges (-.2,-1], (-.1,-.01] and (-.01,.01).

Table 12
Correlation Coefficients by Value Only for Cosine Similarity

Similarity	NPL	LISA	CISI	CRAN	MED	CACM
(-.3,-.2]	-0.74	-0.54	-0.16	-0.28	0.12	0.48
(-.2,-.1]	0.97	0.96	0.90	0.89	0.60	0.99
(-.1,-.01]	0.78	0.77	0.82	0.76	0.40	0.97
(-.01,.01)	0.98	0.98	0.92	0.90	0.62	0.99
[.01,.1]	-0.36	-0.14	0.18	0.30	0.30	0.82
(.1,.2]	-0.85	-0.81	-0.66	-0.64	-0.34	-0.07
(.2,.3]	-0.98	-0.99	-0.89	-0.90	-0.62	-0.62
(.3,.4]	-0.99	-0.99	-0.98	-0.99	-0.78	-0.84
(.4,.5]	-0.98	-0.97	-0.99	-1.00	-0.87	-0.93
(.5,.6]	-0.96	-0.96	-0.96	-0.98	-0.91	-0.94

Table 13
Correlation Coefficients by Value Only for Term-to-Term Similarity

Similarity	NPL	LISA	CISI	CRAN	MED	CACM
(-.02,-.01]	0.87	0.71	0.79	0.74	0.41	0.97
(-.01,-.001]	1.00	0.96	0.80	0.75	0.33	0.95
(-.001,.001)	-0.99	-0.95	-0.81	-0.73	-0.34	-0.94
[.001,.01]	-0.99	-0.95	0.89	-0.93	0.34	0.34
(.01,.02]	0.35	0.93	0.84	0.82	0.43	0.96
(.02,.03]	0.52	0.79	0.81	0.79	0.37	0.97
(.03,.04]	0.95	0.72	0.80	0.77	0.37	0.97
(.04,.05]	0.87	0.69	0.80	0.73	0.38	0.98
(.05,.06]	0.86	0.69	0.81	0.74	0.38	0.98
(.06,.07]	0.84	0.68	0.82	0.78	0.42	0.98
(.07,.08]	0.82	0.68	0.83	0.75	0.44	0.99
(.08,.09]	0.83	0.70	0.81	0.80	0.46	0.98
(.09,.1)	0.84	0.71	0.84	0.77	0.49	0.98
[.1, 9999]	0.87	0.81	0.89	0.83	0.58	0.99

Table 13 shows the corresponding values for selected ranges of the term-to-term similarity. These results are more difficult to interpret. We see some similarity in the (-.02,-.01], (-.01,-.001] and (-.001,.001) ranges for all collections except MED. The positive values do not lend weight to any conclusion. NPL and CACM show strong correlations for some ranges, while the other collections report weaker correlations.

Our next step was to determine if these correlations existed when the distributions and LSI performance were compared across collections. Two studies were done, one holding k constant at k=100 and the second using the optimal k (identified in Table 8) for each collection. Once again we looked at both the cosine and the term-to-term similarities. Table 14 shows the value distribution for the cosine similarity for k=100. The correlation coefficients for the cross collection studies are shown in Table 15. There is little evidence that the distribution of values has an impact on determining the optimal value of k; however, there is a strong correlation between the second-order negative and zero values and LSI performance, when k=100 is used. These correlations are not as strong as the correlations obtained when comparing different values

Table 14
Cross Collection Distribution by Sign and Order of Co-occurrence, Cosine, k=100

1st Order

	CACM	MED	CISI	CRAN	LISA	NPL
[-1.0,-.01]	1.8%	1.9%	2.6%	2.5%	2.3%	3.0%
(-.01,.01)	1.9%	1.9%	2.3%	2.6%	2.1%	2.9%
[.01, 1.0]	96.3%	96.2%	95.0%	95.0%	95.6%	94.1%

2nd Order

	CACM	MED	CISI	CRAN	LISA	NPL
[-1.0,-.01]	21.3%	35.0%	31.7%	31.2%	28.7%	28.2%
(-.01,.01)	7.8%	11.4%	9.2%	10.6%	8.5%	9.9%
[.01, 1.0]	71.0%	53.6%	59.1%	58.2%	62.8%	61.9%

3rd Order

	CACM	MED	CISI	CRAN	LISA	NPL
[-1.0,-.01]	55.6%	75.0%	77.3%	72.8%	69.9%	66.9%
(-.01,.01)	17.3%	9.9%	8.7%	12.1%	10.3%	11.6%
[.01, 1.0]	27.1%	15.1%	14.0%	15.2%	19.9%	21.6%

LSI Performance *Beta*=1

	CACM	MED	CISI	CRAN	LISA	NPL
F_{beta}	0.13	0.56	0.23	0.14	0.20	0.16

Table 15
Cross collection correlation coefficients

Cosine Similarity

***k*=100**

	1st Order	2nd Order	3rd Order
[-1.0,-.01]	-0.40	0.68	0.49
(-.01,.01)	-0.47	0.63	-0.45
[.01, 1.0]	0.44	-0.69	-0.48

Optimal *k*

	1st Order	2nd Order	3rd Order
[-1.0,-.01]	-0.36	0.32	0.23
(-.01,.01)	-0.35	-0.17	-0.34
[.01, 1.0]	0.36	-0.12	0.02

Term-to-Term Similarity

***k*=100**

	1st Order	2nd Order	3rd Order
[-1.0,-.01]	-0.53	-0.24	-0.29
(-.01,.01)	0.21	0.36	0.32
[.01, 1.0]	-0.04	-0.43	-0.38

Optimal *k*

	1st Order	2nd Order	3rd Order
[-1.0,-.01]	-0.43	-0.29	-0.31
(-.01,.01)	0.48	0.36	0.31
[.01, 1.0]	-0.49	-0.44	-0.32

of k within a single collection, but finding any similarity across these widely disparate collections is noteworthy.

4.4 Discussion

Our results show strong correlations between higher orders of co-occurrence in the SVD algorithm and the performance of LSI, particularly when the cosine similarity metric is used. In fact higher-order co-occurrences play a key role in the effectiveness of many systems used for information retrieval and text mining. We detour briefly to describe recent applications that are implicitly or explicitly using higher orders of co-occurrence to improve performance in applications such as Search and Retrieval, Word Sense Disambiguation, Stemming, Keyword Classification and Word Selection.

Philip Edmonds shows the benefits of using second- and third-order co-occurrence in (Edmonds, 1997). The application described selects the most appropriate term when a context (such as a sentence) is provided. Experimental results show that the use of second-order co-occurrence significantly improved the precision of the system. Use of third-order co-occurrence resulted in incremental improvements beyond second-order co-occurrence.

(Zhang et al., 2000) explicitly used second-order term co-occurrence to improve an LSI based search and retrieval application. Their approach narrows the term and document space, reducing the size of the matrix that is input into the LSI system. The system selects terms and documents for the reduced space by first selecting all the documents that contain the terms in the query, then selecting all terms in those documents, and finally selecting all documents that contain the expanded list of terms. This approach reduces the nonzero entries in the term document matrix by an average of 27%. Unfortunately average precision also was degraded. However, when terms associated with only one document were removed from the reduced space, the number of nonzero entries was reduced by 65%, when compared to the baseline, and precision degradation was only 5%.

(Schütze, 1998) explicitly uses second-order co-occurrence in his work on Automatic Word Sense Disambiguation. In this article, Schütze presents an algorithm for discriminating the senses of a given term. For example, the word senses in the previous sentence can mean the physical senses (sight, hearing, etc.) or it can mean 'a meaning conveyed by speech or writing.' Clearly the latter is a better definition of this use of senses, but automated systems based solely on keyword analysis would return this sentence to a query that asked about the sense of smell. The paper presents an algorithm based on use of second-order co-occurrence of the terms in the training set to create context vectors that represent a specific sense of a word to be discriminated.

Xu and Croft introduce the use of co-occurrence data to improve stemming algorithms (Xu and Croft, 1998). The premise of the system described in this paper is to use contextual (e.g., co-occurrence) information to improve the equivalence classes produced by an aggressive stemmer, such as the Porter stemmer. The algorithm breaks down one large class for a family of terms into small contextually based equivalence classes. Smaller, more tightly connected equivalence classes result in more effective retrieval (in terms of precision and recall), as well an improved run-time performance (since fewer terms are added to the query). Xu and Croft's algorithm implicitly uses higher orders of co-occurrence. A strong correlation between terms A and B, and also between terms B and C will result in the placement of terms A, B, and C into the same equivalence class. The result will be a transitive semantic relationship between A and C. Orders of co-occurrence higher than two are also possible in this application.

In this section we have empirically demonstrated the relationship between higher orders of co-occurrence in the SVD algorithm and the performance of LSI. Thus we have provided a model for understanding the performance of LSI by showing that second-order co-occurrence plays a critical role. In the following section we provide an intuitive mathematical basis for understanding how LSI employs higher-order co-occurrence in emphasizing latent semantics. In the conclusion we touch briefly on the applicability of this result to applications in information retrieval.

5 Transitivity and the SVD

In this section we present mathematical proof that the LSI algorithm encapsulates term co-occurrence information. Specifically we show that a connectivity path exists for every nonzero element in the truncated matrix.

We begin by setting up some notation. Let A be a term by document matrix. The SVD process decomposes A into three matrices: a term by dimension matrix, T, a diagonal matrix of singular values, S, and a document by dimension matrix D. The original matrix is re-formed by multiplying the components, $A = TSD^T$. When the components are truncated to k dimensions, a reduced representation matrix, A_k is formed as $A_k = T_k S_k D_k^T$ (Deerwester et al., 1990).

The term-to-term co-occurrence matrices for the full matrix and the truncated matrix are shown in (2) and (3), respectively.

$$B = TSST^T \tag{2}$$

$$Y = T_k S_k S_k T_k^T \tag{3}$$

We note that elements of B represent term co-occurrences in the collection, and $b_{ij} \geq 0$ for all i and j. If term i and term j co-occur in any document in the collection, $b_{ij} > 0$. Matrix multiplication results in equations (4) and (5) for the ij^{th} element of the co-occurrence matrix and the truncated matrix, respectively. Here u_{ip} is the element in row i and column p of the matrix T, and s_p is the p^{th} largest singular value.

$$b_{ij} = \sum_{p=1}^{m} s_p^2 u_{ip} u_{jp} \tag{4}$$

$$y_{ij} = \sum_{p=1}^{k} s_p^2 u_{ip} u_{jp} \tag{5}$$

B^2 can be represented in terms of T and S as shown in (6).

$$B^2 = (TSST^T)(TSST^T) = TSS(T^TT)SST^T = TSSSST^T = TS^4T^T \tag{6}$$

An inductive proof can be used to show (7).

$$B^n = TS^{2n}T^T \tag{7}$$

And the element b_{ij}^n can be written using (8).

$$b_{ij}^n = \sum_{p=1}^{m} s_p^{2n} u_{ip} u_{jp} \tag{8}$$

To complete our argument, we need two lemmas related to the powers of the matrix B.

Lemma 1 *Let i and j be terms in a collection, there is a transitivity path of order ≤ 0 between the terms, iff the ij^{th} element of B_n is nonzero.*

Lemma 2 *If there is no transitivity path between terms i and j, then the ij^{th} element of B_n (b_{ij}^n) is zero for all n.*

The proof of these lemmas can be found in (Kontostathis and Pottenger, 2002). We are now ready to present our theorem.

Theorem 1 *If the ij^{th} element of the truncated term-to-term matrix, Y, is nonzero, then there is a transitivity path between term i and term j.*

We need to show that if $y_{ij} \neq 0$, then there exists terms $q_1, \ldots, q_n$, $n \geq 0$ such that $b_{iq_1} \neq 0$, $b_{q_1q_2} \neq 0$, $\ldots$, $b_{q_nj} \neq 0$. Alternately, we can show that if there is no path between terms i and j, then $y_{ij} = 0$ for all k.

Assume the T and S matrices have been truncated to k dimensions and the resulting Y matrix has been formed. Furthermore, assume there is no path between term i and term j. Equation (5) represents the y_{ij} element. Assume that $s_1 > s_2 > s_3 > \ldots > s_k > 0$. By Lemma 2, $b^n_{ij} = 0$ for all n. Dividing (8) by s_1^{2n}, we have equation (9).

$$b^n_{ij} = u_{i1}u_{j1} + \sum_{p=2}^{m} \frac{s_p^{2n}}{s_1^{2n}} u_{ip}u_{jp} = 0 \tag{9}$$

We take the limit of this equation as $n \to \infty$, and note that $(\frac{s_p}{s_1}) < 1$ when $2 \leq p \leq m$. Then as $n \to \infty$, $(\frac{s_p}{s_1})^{2n} \to 0$ and the summation term reduces to zero. We conclude that $u_{i1}u_{j1} = 0$. Substituting back into (8) we have (10).

$$\sum_{p=2}^{m} s_p^{2n} u_{ip}u_{jp} = 0 \tag{10}$$

Dividing by s_2^{2n} yields (11).

$$u_{i2}u_{j2} + \sum_{p=3}^{m} \frac{s_p^{2n}}{s_2^{2n}} u_{ip}u_{jp} = 0 \tag{11}$$

Taking the limit as $n \to \infty$, we have that $u_{i2}u_{j2} = 0$. If we apply the same argument k times we will obtain $u_{ip}u_{jp} = 0$ for all p such that $1 \leq p \leq k$. Substituting back into (5) shows that $y_{ij} = 0$ for all k.

The argument thus far depends on our assumption that $s_1 > s_2 > s_3 > \ldots > s_k > 0$. When using SVD it is customary to truncate the matrices by removing all dimensions whose singular value is below a given threshold (Dumais, 1993); however, for our discussion, we will merely assume that, if $s_1 > s_2 > \ldots > s_{z-1} > s_z = s_{z+1} = s_{z+2} = \ldots = s_{z+w} > s_{z+w+1} > \ldots > s_k$ for some z and some $w \geq 1$, the truncation will either remove all of the dimensions with the duplicate singular value, or keep all of the dimensions with this value.

We need to examine two cases. In the first case, $z > k$ and the $z \ldots z+w$ dimensions have been truncated. In this case, the above argument shows that either $u_{iq} = 0$ or $u_{jq} = 0$ for all $q \leq k$ and, therefore, $y_{ij} = 0$.

To handle the second case, we assume that $z < k$ and the $z \ldots z+w$ dimensions have not been truncated and rewrite equation (8) as (12).

$$b^n_{ij} = \sum_{p=1}^{z-1} s_p^{2n} u_{ip}u_{jp} + \sum_{p=z}^{z+w} s_z^{2n} u_{ip}u_{jp} + \sum_{p=z+w+1}^{m} s_p^{2n} u_{ip}u_{jp} = 0 \tag{12}$$

The argument above can be used to show that $u_{ip}u_{jp} = 0$ for $p \leq z-1$, and the first summation can be removed. After we divide the remainder of the equation by s_z^{2n}, we have (13).

$$b_{ij}^n = \sum_{p=z}^{z+w} u_{ip}u_{jp} + \sum_{p=z+w+1}^{m} \frac{s_p^{2n}}{s_z^{2n}} u_{ip}u_{jp} = 0 \tag{13}$$

Taking the limit as $n \to \infty$, we conclude that $\sum_{p=z}^{z+w} u_{ip}u_{jp} = 0$, and b_{ij}^n is reduced to (14).

$$b_{ij}^n = \sum_{p=z+w+1}^{m} s_p^{2n} u_{ip}u_{jp} = 0 \tag{14}$$

Again using the argument above, we can show that $u_{ip}u_{jp} = 0$ for $z + w + 1 \leq p \leq k$. Furthermore (15) holds, and our proof is complete. ■

$$y_{ij} = \sum_{p=1}^{z-1} s_p^{2n} u_{ip}u_{jp} + \sum_{p=z}^{z+w} s_z^{2n} u_{ip}u_{jp} + \sum_{p=z+w+1}^{k} s_p^{2n} u_{ip}u_{jp} = 0 \tag{15}$$

6 Conclusions and Future Work

Higher-order co-occurrences play a key role in the effectiveness of systems used for information retrieval and text mining. We have explicitly shown use of higher orders of co-occurrence in the Singular Value Decomposition (SVD) algorithm and, by inference, on the systems that rely on SVD, such as LSI. Our empirical studies and mathematical analysis prove that term co-occurrence plays a crucial role in LSI. The work shown here will find many practical applications. Below we describe our own research activities that were directly influenced by our discovery of the relationship between SVD and higher-order term co-occurrence.

Our first example is a novel approach to term clustering. Our algorithm defines term similarity as the distance between the term vectors in the T_kS_k matrix. We conclude from Section 4 that this definition of term similarity is more directly correlated to improved performance than is use of the reduced dimensional term-to-term matrix values. In (Kontostathis et al., 2004) we describe the use of these term clusters in a system which detects emerging trends in a corpus of time stamped textual documents. Our approach enables the detection of 92% of the emerging trends, on average, for the collections we tested.

Our second, and more ambitious, application of these results is the development of an algorithm for approximating LSI. LSI runtime performance is significantly poorer than vector space performance for two reasons. First, the decomposition must be performed and it is computationally expensive. Second, the matching of queries to documents in LSI is also computationally expensive. The original document vectors are very sparse, but the document by dimension vectors used in LSI retrieval are dense, and the query must be compared to each document vector. Furthermore, the optimal truncation value (k) must be discovered for each collection. We believe that the correlation data presented here can be used to develop an algorithm that approximates the performance of an optimal LSI system while reducing the computational overhead.

7 Acknowledgements

This work was supported in part by National Science Foundation Grant Number EIA-0087977. The authors gratefully acknowledge the assistance of Dr. Kae Kontostathis and Dr. Wei-Min Huang in developing the proof of the transitivity in LSI as well as in reviewing drafts of this article. The authors also would like to express their gratitude to Dr. Brian D. Davison for his comments on a draft. The authors gratefully acknowledge the assistance of their colleagues in the Computer Science and Engineering Department at Lehigh University and the Department of Mathematics and Computer Science at Ursinus College in completing this work. Co-author William M. Pottenger also gratefully acknowledges his Lord and Savior, Yeshua (Jesus) the Messiah, for His continuing guidance in and salvation of his life.

References

Berry, M. W., Do, T., O'Brien, G., Krishna, V., Varadhan, S., 1993. SVDPACKC (version 1.0) User's Guide. Tech. rep.

Berry, M. W., Drmac, Z., Jessup, E. R., 1999. Matrices, vector spaces, and information retrieval. SIAM Review 41 (2), 335–362.

Berry, M. W., Dumais, S. T., O'Brien, G. W., 1995. Using linear algebra for intelligent information retrieval. SIAM Review 37 (4), 575–595.
URL `citeseer.nj.nec.com/berry95using.html`

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., Harshman, R. A., 1990. Indexing by latent semantic analysis. Journal of the American Society of Information Science 41 (6), 391–407.
URL `citeseer.nj.nec.com/deerwester90indexing.html`

Ding, C. H. Q., 1999. A similarity-based probability model for latent semantic indexing. In: Proceedings of the Twentysecond Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval. pp. 59–65.

Dominich, S., 2003. Connectionist interaction information retrieval. Information Processing and Management 39 (2), 167–194.

Dumais, S. T., 1992. LSI meets TREC: A status report. In: Harman, D. (Ed.), The First Text REtrieval Conference (TREC-1), National Institute of Standards and Technology Special Publication 500-207. pp. 137–152.
URL `citeseer.nj.nec.com/dumais93lsi.html`

Dumais, S. T., 1994. Latent semantic indexing (LSI) and TREC-2. In: Harman, D. (Ed.), The Second Text REtrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215. pp. 105–116.
URL `citeseer.nj.nec.com/19248.html`

Dumais, S. T., 1995. Using LSI for information filtering: TREC-3 experiments. In: Harman, D. (Ed.), The Third Text REtrieval Conference (TREC-3), National Institute of Standards and Technology Special Publication 500-225. pp. 219–230.

Edmonds, P., 1997. Choosing the word most typical in context using a lexical co-occurrence network. In: Proceedings of the Thirtyfifth Annual Meeting of the Association for Computational Linguistics. pp. 507–509.

Kontostathis, A., Holzman, L. E., Pottenger, W. M., 2004. Use of term clusters for emerging trend detection. Preprint.

Kontostathis, A., Pottenger, W. M., 2002. A mathematical view of latent semantic indexing: Tracing term co-occurrences. Technical report, LU-CSE-02-006, Dept. of Computer Science and Engineering, Lehigh University.

Martin, D. I., Berry, M. W., 2004. Principal component analysis for information retrieval. Handbook of Parallel Computing and Statistics, Marcel Dekker Series: Statistics: Textbooks and Monographs.To appear.

Pottenger, W. M., Kim, Y.-B., Meling, D. D., 2001. Hddi$_{TM}$: Hierarchical distributed dynamic indexing. Data Mining for Scientific and Engineering Applications.

Pottenger, W. M., Yang, T., 2001. Detecting emerging concepts in textual data mining. Computational Information Retrieval.

Schütze, H., 1992. Dimensions of meaning. In: Proceedings of Supercomputing.

Schütze, H., 1998. Automatic word sense discrimination. Computational Linguistics 24 (1), 97–124.
URL `citeseer.nj.nec.com/article/schutze98automatic.html`

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), 329–344.

van Rijsbergen, C., 1979. Information Retrieval. Department of Computer Science, University of Glasgow.

Wiemer-Hastings, P., 1999. How latent is latent semantic analysis? In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. pp. 932–937.

Xu, J., Croft, W. B., 1998. Corpus-based stemming using co-occurrence of word variants. ACM Transactions on Information Systems 16 (1), 61–81.

Zelikovitz, S., Hirsh, H., 2001. Using LSI for text classification in the presence of background text. In: Paques, H., Liu, L., Grossman, D. (Eds.), Proceedings of CIKM-01, Tenth ACM International Conference on Information and Knowledge Management. ACM Press, New York, US, Atlanta, US, pp. 113–118.
URL `citeseer.nj.nec.com/zelikovitz01using.html`

Zha, H., Simon, H., 1998. A subspace-based model for latent semantic indexing in information retrieval. In: Proceedings of the Thirteenth Symposium on the Interface. pp. 315–320.

Zhang, X., Berry, M., Raghavani, P., 2000. Level search schemes for information filtering and retrieval. Information Processing and Management 37 (2), 313–334.