

# A Mathematical View of Latent Semantic Indexing: Tracing Term Co-occurrences

April Kontostathis  
Lehigh University  
CSE Department  
19 Memorial Drive West  
Bethlehem, PA 18015  
apk5@lehigh.edu

William M. Pottenger, Ph.D.  
Lehigh University  
CSE Department  
19 Memorial Drive West  
Bethlehem, PA 18015  
billp@cse.lehigh.edu

## ABSTRACT

Current research in Latent Semantic Indexing (LSI) shows improvements in performance for a wide variety of information retrieval systems. We propose the development of a theoretical foundation for understanding the values produced in the reduced form of the term-term matrix. We assert that LSI's use of higher orders of co-occurrence is a critical component of this study. In this work we present experiments that precisely determine the degree of co-occurrence used in LSI. We empirically demonstrate that LSI uses up to fifth order term co-occurrence. We also prove mathematically that a connectivity path exists for every nonzero element in the truncated term-term matrix computed by LSI. A complete understanding of this term transitivity is key to understanding LSI.

## 1. INTRODUCTION

The use of co-occurrence information in textual data has led to improvements in performance when applied to a variety of applications in information retrieval, computational linguistics and textual data mining. Furthermore, many researchers in these fields have developed techniques that explicitly employ second and third order term co-occurrence. Examples include applications such as literature search [14], word sense disambiguation [12], ranking of relevant documents [15], and word selection [8]. Other authors have developed algorithms that implicitly rely on the use of term co-occurrence for applications such as search and retrieval [5], trend detection [14], and stemming [17]. In what follows we refer to various degrees of term transitivity as orders of co-occurrence – first order if two terms co-occur, second order if two terms are linked only by a third, etc. An example of second order co-occurrence follows.

Assume that a collection has one document that contains the terms *Java* and *applet*. Also, assume that a different document in the collection contains the terms *Java* and *e-commerce*. Furthermore, assume that *applet* and *e-commerce* do not co-occur elsewhere in the collection. It may be useful to gather these three terms in one unique class and infer that *e-commerce* and *applet* are actually related despite the fact that they do not co-occur in any item in the given collection. In other words, due to the strength of the statistical association of these terms, a limited degree of transitivity is imputed to the co-occurrence relation. This example of second-order co-occurrence can be extended to third, fourth, or  $n^{\text{th}}$  order co-occurrence.

We propose the development of a theoretical foundation for understanding LSI, particularly the values in the reduced form of the term-term matrix. We assert that LSI's use of higher orders of co-occurrence is a critical component of this study. To develop this theoretical foundation, we will study the Singular Value Decomposition (SVD) algorithm that forms the basis for LSI.

In this work we present experiments that precisely determine the degree of term transitivity used in LSI. We will show empirically that LSI uses up to fifth order term co-occurrence. In addition, we show that the values in the term-term matrix are strongly correlated with the number of second order co-occurrence paths. We also prove mathematically that a connectivity path exists for every nonzero element of the term-term matrix.

In section 2 we present a simple example of the use of third-order co-occurrence in LSI. Section 3 discusses related work. In section 4 we present our empirical methodology and results. Section 5 presents a mathematical proof of term transitivity within LSI. Section 6 discusses the importance of these findings and describes future work.

## 2. COOCCURRENCE IN LSI - EXAMPLE

The data for the following example is taken from [5]. In this paper, the authors describe an example with 12 terms and 9 documents. The term by document matrix is shown as table 1, and the corresponding term-by-term matrix is shown in table 2.

**Table 1: Deerwester Term by Document Matrix**

Terms	Documents								
	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

**Table 2: Deerwester Term by Term Matrix**

	human	interface	computer	user	system	response	time	EPS	Survey	trees	graph	minors
human	x	1	1	0	2	0	0	1	0	0	0	0
interface	1	x	1	1	1	0	0	1	0	0	0	0
computer	1	1	x	1	1	1	1	0	1	0	0	0
user	0	1	1	x	2	2	2	1	1	0	0	0
system	2	1	1	2	x	1	1	3	1	0	0	0
response	0	0	1	2	1	x	2	0	1	0	0	0
time	0	0	1	2	1	2	x	0	1	0	0	0
EPS	1	1	0	1	3	0	0	x	0	0	0	0
Survey	0	0	1	1	1	1	1	0	x	0	1	1
trees	0	0	0	0	0	0	0	0	0	x	2	1
graph	0	0	0	0	0	0	0	0	1	2	x	2
minors	0	0	0	0	0	0	0	0	1	1	2	x

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 number of dimensions is  $\min(t, d)$  where  $t$  = number of terms and  $d$  = number of documents. The original matrix can be obtained, through matrix multiplication of  $TSD^T$ . The reader is referred to Deerwester, et al – 1990 [5] for the T, S, and D matrices. In the LSI system, the T, S and D matrices are truncated to  $k$  dimensions. The dimensionality reduction reduces “noise” in the term-term matrix resulting in a richer word relationship structure that reveals latent semantics present in the collection. After dimensionality reduction the term-term matrix can be approximated using the formula  $T_k S_k (T_k S_k)^T$ . A representation of the term-term matrix, after reduction to 2 dimensions ( $k=2$ ) is shown in table 3.

**Table 3: Deerwester Term by Term Matrix, truncated to two dimensions**

	human	interface	computer	user	system	response	time	EPS	Survey	trees	graph	minors
human	0.62	0.54	0.56	0.94	1.69	0.58	0.58	0.84	0.32	-0.32	-0.34	-0.25
interface	0.54	0.48	0.52	0.87	1.50	0.55	0.55	0.73	0.35	-0.20	-0.19	-0.14
computer	0.56	0.52	0.65	1.09	1.67	0.75	0.75	0.77	0.63	0.15	0.27	0.20
user	0.94	0.87	1.09	1.81	2.79	1.25	1.25	1.28	1.04	0.23	0.42	0.31
system	1.69	1.50	1.67	2.79	4.76	1.81	1.81	2.30	1.20	-0.47	-0.39	-0.28
response	0.58	0.55	0.75	1.25	1.81	0.89	0.89	0.80	0.82	0.38	0.56	0.41
time	0.58	0.55	0.75	1.25	1.81	0.89	0.89	0.80	0.82	0.38	0.56	0.41
EPS	0.84	0.73	0.77	1.28	2.30	0.80	0.80	1.13	0.46	-0.41	-0.43	-0.31
Survey	0.32	0.35	0.63	1.04	1.20	0.82	0.82	0.46	0.96	0.88	1.17	0.85
trees	-0.32	-0.20	0.15	0.23	-0.47	0.38	0.38	-0.41	0.88	1.55	1.96	1.43
graph	-0.34	-0.19	0.27	0.42	-0.39	0.56	0.56	-0.43	1.17	1.96	2.50	1.81
minors	-0.25	-0.14	0.20	0.31	-0.28	0.41	0.41	-0.31	0.85	1.43	1.81	1.32

For our purposes, 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, 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* (see table 2); therefore, *user* and *human* co-occur. This demonstrates the value of the LSI system, since queries on the term *user* would correctly result in documents containing the term *human* in the context of this collection.

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 (although 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 in the reduced term-term matrix is an example of third order co-occurrence.

Note also the negative values produced in the term-term matrix. We will assume a negative value in position (i,j) represents an *anti-similarity*. Note that the anti-similarity values also have a connectivity path in the matrix. For example, the value -.25 for *minors* and *human* has the path *minors* to *survey* to *interface* to *human*, an example of third order co-occurrence. In section 4 we present trend data that explains term-term matrix values in terms of the number of connectivity paths between terms.

### 3. RELATED WORK

The algebraic foundation for LSI was first described in [5] and has been further discussed by Berry, et al. in [2][3]. 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. The T and D matrices represent term and document vectors, respectively. In [16], Wiemer-Hastings shows that the power of LSI comes primarily from the SVD algorithm.

Other researchers have proposed theoretical approaches to understanding LSI. [18] describes LSI in terms of a subspace model and proposes a statistical test for choosing the optimal number of dimensions for a given collection. [13] discusses LSI's relationship to statistical regression and Bayesian methods. [6] constructs a dual probability 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 LSI [11]. We expand upon this work, showing here that SVD exploits higher order term co-occurrence in a collection, and providing insight into the origin of the values produced in the term-term matrix.

## 4. EXPERIMENTAL RESULTS

We chose the MED, CRAN, CISI and LISA collections for our study of the higher order co-occurrence in LSI. MED is a commonly studied collection of medical abstracts. It consists of 1033 documents and 5823 terms. CISI is a set of 1460 information science abstracts containing 5609 terms. The MED and CISI collections were used for the landmark Deerwester paper [5]. CRAN is the acronym for the Cranfield collection, one of the earliest collections available to information retrieval researchers. CRAN consists of 1398 documents and 4612 terms.

Experiments on a larger collection, LISA, were also performed. The LISA (Library and Information Science Abstracts) test collection was developed by the University of Sheffield, England. 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 [10] 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.

### 4.1 Methodology

Our experiments captured three main features of these data sets:

- The order of co-occurrence for each pair of terms in the truncated term-term matrix (shortest path length). The order of co-occurrence for each pair of terms in the original (not truncated) matrix was also determined.
- The distribution of the similarity values produced in the term-term matrix, categorized by order of co-occurrence.
- 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 [1] that provides eight algorithms for decomposition was selected because it was readily available, as well as thoroughly tested. The sis1 algorithm of the SVDPACK suite was used in our experiments.

The SVDPACK algorithms take input in Harwell-Boeing sparse matrix format. The MED, CRAN and CISI data was already in this format and was used without further preprocessing. As mentioned above, the HDDI™ Collection Builder [10] was used to extract the concepts for the LISA Noun Phrase collection. A customized process was developed to extract the words from the LISA data for the LISA Words collection. A preprocessor was developed to transform both data sets into the Harwell-Boeing sparse matrix format.

The sis1 program was modified slightly to produce the vectors and singular values in an output file. The sis1 program was then executed for each collection to find the singular values and vectors for  $k = 100$ .

The singular values and vectors were then input into the Trace Co-occurrence program, which identifies the level of co-occurrence for each pair of co-occurring terms. An additional utility was developed to create the term-by-term co-occurrence matrix for the collection (corresponding to table 2), which is also used by the Trace Co-occurrence program.

### 4.2 Trace Co-occurrence Algorithm

The Trace Co-occurrence algorithm is outlined in figure 1. The algorithm takes as input the vectors and singular values that were produced by the SVD program (the D vectors are read, but are not currently used), as well as the original co-occurrence matrix. The truncated matrix is then produced, using standard matrix multiplication. The program then traverses through the truncated matrix, and identifies the order of co-occurrence of each pair of terms.

```

Read the original term by term matrix
Read the T, S, and D vectors (output from sis1)
Compute the k-dim term-by-term matrix ST(ST)T // k = 100 for our experiments
FOR each (ti, tj) pair in the truncated matrix
  IF the (ti, tj) pair is also in the original matrix
    Set the co-occurrenceLevel = 1
    Update statistics
  ELSE
    Set workingLevel = 1
    Add the ti co-occurrences from the original matrix to SearchList
    Do until co-occurrenceLevel is found
      Do until SearchList is empty
        Let tk = top SearchList
        IF tk co-occurs with tj in original matrix
          set co-occurrenceLevel = workingLevel + 1
          update statistics
        ELSE
          Add all terms that co-occur with tk to NewSearchList
        ENDIF
        pop SearchList
      ENDDO
      increment workingLevel
      set SearchList = NewSearchList
    ENDDO
  ENDIF
END FOR
output co-occurrence statistics

```

**Figure 1: Trace Co-occurrence Algorithm**

Referring back to the example presented in section 2, the *human – interface* pair with value .54 in the truncated matrix (table 3) is the first co-occurrence encountered. The program immediately notes that the *human – interface* pair also appears in the original term-by-term matrix (table 2), and a co-occurrence level of 1 is stored in the statistics summary.

Further on, the co-occurrence pair *human – user* with value .94 is processed. In this instance, a check of the original matrix indicates that the *human – user* pair did not co-occur. Thus all of the terms that co-occur with human are added to the search list (*interface*, *computer*, *system*, and *EPS*). The program then processes the search list, starting with the term *interface*. Since *interface* co-occurs with *user* (table 1), the co-occurrence level is set to 2 and the statistics updated.

### 4.3 Results

The order of co-occurrence summary from the TraceCo-occurrence program for all of the collections is shown in table 4. Fifth order co-occurrence was the highest degree of transitivity observed. It is interesting to note that the noun phrase collection

is the only collection that resulted in a co-occurrence order higher than 3. It is important also to note that the order 2 and order 3 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-term matrix. Note that LSI finds over 99% of term co-occurrences present in the data for the first four collections, and 95% for the LISA Noun Phrase collection.

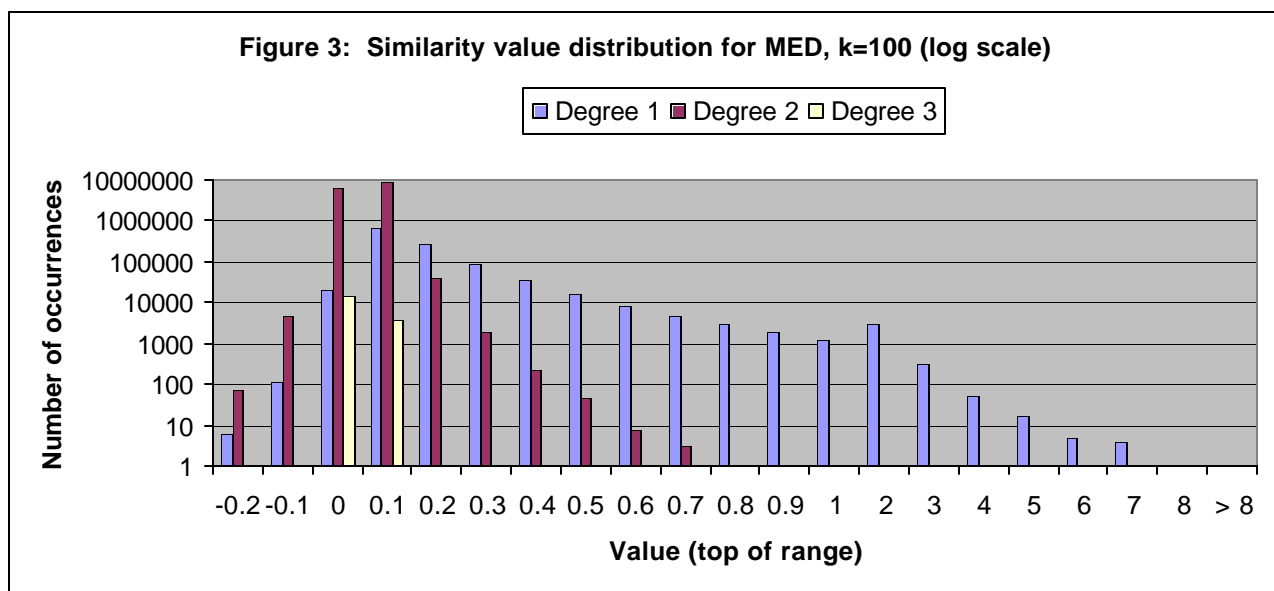
**Table 4: Order of Co-occurrence Summary Data**

k =100 for all Collections

Collection	Number of pairs with order of co-occurrence					
	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

Figure 3 shows the weight distribution for the MED data, for k=100. Tables 5 and 6 show this data for CRAN and LISA Words in tabular format. For MED, the order 2 weights range from -.3 to .8. These co-occurrences will result in significant differences in document matching when applied in a search and retrieval application. However, the weights for the degree 3 transitivity pairs are between -.1 and .1, effectively adding a relatively small degree of variation to the final results. Also note that many of the original term-by-term co-occurrences are reduced to nearly zero, while others are significantly larger (up to 7).

Tables 5 and 6 show the average number of paths by term-term value range for CRAN and LISA Words. MED and CISI showed similar trends. Clearly the degree 2 pairs that have a very low similarity (between -.1 and .1) have a much smaller average number of paths than the pairs with either higher negative or positive similarities, for all collections studied. This data provides insight into understanding the values produced by SVD in the truncated term-term matrix.



Notice that the degree 1 pairs with higher average number of paths tend to have negative similarity values, pairs with few co-occurrences 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, while de-emphasizing terms that co-occur frequently with many other terms (reduces ‘noise’). On the other hand, degree 2 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 2 pairs with high values can be considered the ‘latent semantics’ that are emphasized by LSI.

In the next section, we present a mathematical proof that a non-zero entry in position (i,j) of the truncated term-term matrix has a corresponding connectivity path between terms i and j. Our experimental data shows that there is often more than one path, and the value and number of paths are correlated.

**Table 5: Average Number of paths by term-term value for CRAN, k=100**

Term Term Matrix Value	CRAN Collection k = 100			Average No	Average No
	Degree 1 Pairs	Degree 2 Pairs	Degree 2 Paths	Paths for Degree 1 pairs	Paths for Degree 2 pairs
less than -0.2	68	632	323,952	4,764	513
-0.2 to -0.1	1,026	13,980	5,388,156	5,252	385
-0.1 to 0.0	52,734	7,416,342	598,493,140	11,349	81
0.0 to 0.1	1,256,054	11,292,268	1,500,874,348	1,195	133
0.1 to 0.2	607,618	89,546	298,811,456	492	3,337
0.2 to 0.3	229,274	4,166	135,964,358	593	32,637
0.3 to 0.4	107,380	366	76,171,768	709	208,120
0.4 to 0.5	57,808	46	47,004,230	813	1,021,831
0.5 to 0.6	34,208	8	31,258,040	914	3,907,255
0.6 to 0.7	21,790	2	21,734,644	997	10,867,322
0.7 to 0.8	15,040	-	15,991,638	1,063	-
0.8 to 0.9	10,180	-	11,607,916	1,140	-
0.9 to 1.0	7,466	-	9,000,714	1,206	-
1.0 to 2.0	23,352	-	32,179,792	1,378	-
2.0 to 3.0	3,374	-	5,595,238	1,658	-
3.0 to 4.0	734	-	1,293,838	1,763	-
4.0 to 5.0	250	-	480,136	1,921	-
5.0 to 6.0	82	-	156,506	1,909	-
6.0 to 7.0	48	-	90,186	1,879	-
7.0 to 8.0	20	-	33,366	1,668	-
over 8.0	14	-	33,386	2,385	-

**Table 6: Average Number of paths by term-term value for LISA Words, k=100**

Term Term Matrix Value	LISA Words Collection k = 100			Average No Paths	Average No Paths
	Degree 1 Pairs	Degree 2 Pairs	Degree 2 Paths	for Degree 1 pairs	for Degree 2 pairs
less than -0.2	21,946	186,066	66,323,200	3,022	356
-0.2 to -0.1	10,012	422,734	59,418,198	5,935	141
-0.1 to 0.0	76,968	127,782,170	1,587,147,584	20,621	12
0.0 to 0.1	1,670,542	175,021,904	4,026,560,130	2,410	23
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

## 5. TRANSITIVITY AND THE SVD

We have shown empirically in Section 4 that LSI uses higher orders of co-occurrence. In this section we present mathematical proof that the SVD 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 with  $m$  rows (terms) and  $g$  columns (documents). 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  $A_k = T_k S_k D_k^T$  [4].

The term-term co-occurrence matrices for the full matrix and the truncated matrix are [4]:

$$(3a) \quad B = TSST^T$$

$$(3b) \quad Y = T_k S_k S_k^T T_k^T$$

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,  $b_{ij} > 0$ . Straight forward matrix multiplication results in equations 4a and 4b for the  $ij^{\text{th}}$  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  $p^{\text{th}}$  largest singular value.

$$(4a) \quad b_{ij} = \sum_{p=1}^m s_p^2 u_{ip} u_{jp}$$

$$(4b) \quad y_{ij} = \sum_{p=1}^k s_p^2 u_{ip} u_{jp}$$

$B^2$  can be represented in terms of the matrices  $T$  and  $S$  as follows:

$$B^2 = (TSST^T)(TSST^T) = TSS(T^T T)SST^T \\ = TSSSST^T = TS^4 T^T$$

An inductive proof can be used to show:

$$(5) \quad B^n = TS^{2n} T^T$$

And the element  $b_{ij}^n$  can be written:

$$(6) \quad b_{ij}^n = \sum_{p=1}^m s_p^{2n} u_{ip} u_{jp}$$

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  $\leq n$  between the terms, iff the  $ij^{\text{th}}$  element of  $B^n$  is nonzero.*

*Proof:* Let the path be noted by the sequence  $f_0, f_1, \dots, f_n$  where  $f_0 = i$ , and  $f_n = j$ . Furthermore, by the definition of a transitivity path we have that  $f_0$  co-occurs with  $f_1$ ,  $f_1$  co-occurs with  $f_2$ ,  $\dots$ ,  $f_{n-1}$  co-occurs with  $f_n$ . In other words,  $b_{f_0 f_1} \neq 0$ ,  $b_{f_1 f_2} \neq 0$ ,  $\dots$ ,  $b_{f_{n-1} f_n} \neq 0$ . Furthermore,  $b_{f_0 f_2}^2 \neq 0$ , since

$$b_{f_0 f_2}^2 = \sum_{p=1}^m b_{f_0 p} b_{p f_2} = \sum_{p=1}^{f_1-1} b_{f_0 p} b_{p f_2} + b_{f_0 f_1} b_{f_1 f_2} + \sum_{p=f_1+1}^m b_{f_0 p} b_{p f_2}$$

and the  $b_{f_0 f_1} b_{f_1 f_2}$  term of the summation is  $> 0$ , since both  $b_{f_0 f_1} > 0$  and  $b_{f_1 f_2} > 0$ . Furthermore all other elements must be 0 or positive, therefore,  $b_{f_0 f_2}^2 > 0$ . We can continue this process iteratively to show that  $b_{f_0 f_n}^n \neq 0$ . Since  $f_0 = i$  and  $f_n = j$ , we have shown that the  $ij^{\text{th}}$  element of  $B^n \neq 0$ . Using the same argument we can show that  $b_{f_0 f_2}^2 > 0$  only if there is a path of length  $\leq 2$  between terms  $f_0$  and  $f_2$ , again using induction, we can show that  $b_{f_0 f_2}^n > 0$  only if there is a path of length  $\leq n$  between terms  $f_0$  and  $f_n$ .

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

*Proof:* Assume there is no transitivity path between terms  $i$  and  $j$ . Then clearly  $b_{ij}^1 = 0$ , since  $i$  and  $j$  do not co-occur in the collection. Assume  $b_{ij}^n = 0$ . Then  $b_{ij}^{n+1} = \sum_{p=1}^m b_{ip}^n b_{pj}$ . Look at each product in the summation, and assume that  $b_{ip}^n b_{pj} > 0$  for some  $p$ . Then lemma 1 says there is a path of length  $n$  between terms  $i$  and  $p$  and a path of length 1 between terms  $p$  and  $j$ . But, in this case, we have a path of length  $n+1$  between  $i$  and  $j$ , which contradicts our hypothesis. Therefore,  $b_{ij}^{n+1}$  must equal 0.

We are now ready to present our theorem.

*Theorem 1: If the  $ij^{\text{th}}$  element of the truncated term by 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  $q_1, \dots, q_n$   $n \geq 0$  such that  $b_{iq_1} \neq 0, b_{q_1 q_2} \neq 0, \dots, b_{q_n j} \neq 0$ . 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 (4b) represents the  $y_{ij}$  element. Assume that  $S_1 > S_2 > S_3 > \dots > S_m > 0$ . By lemma 2,  $b_{ij}^n = 0$  for all  $n$ . Dividing (6) by  $s_1^{2n}$ , we conclude that

$$u_{ii}u_{jj} + \sum_{p=2}^m \left(\frac{S_p}{S_1}\right)^{2n} u_{ip}u_{jp} = 0$$

We take the limit of this equation as  $n \rightarrow \infty$ , and note that  $(s_p/s_1) < 1$  when  $2 \leq p \leq m$ . Then as  $n \rightarrow \infty$ ,  $(s_p/s_1)^{2n} \rightarrow 0$  and the summation term reduces to zero. We conclude that  $u_{ii}u_{jj} = 0$ . Substituting back into  $b_{ij}^n$  we have:

$$\sum_{p=2}^m S_p^{2n} u_{ip}u_{jp} = 0$$

Dividing by  $s_2^{2n}$  yields:

$$u_{i2}u_{j2} + \sum_{p=3}^m \left(\frac{S_p}{S_2}\right)^{2n} u_{ip}u_{jp} = 0 \text{ for all } n.$$

Taking the limit as  $n \rightarrow \infty$ , we have that  $u_{i2}u_{j2} = 0$ . If we apply the same argument  $m$  times we will obtain  $u_{ip}u_{jp} = 0$  for all  $p$  such that  $1 \leq p \leq m$ . Substituting back into (4b) shows that  $y_{ij} = 0$  for all  $k$ .

The argument thus far depends on our assumption that  $S_1 > S_2 > S_3 > \dots > S_m$ . When using SVD it is customary to truncate the matrices by removing all dimensions whose singular value is below a given threshold [7]; however, for our discussion, we will merely assume that, if  $s_1 > s_2 > \dots > s_{z-1} > s_z = s_{z+1} = s_{z+2} = \dots = s_{z+w} > s_{z+w+1} > \dots > s_m$  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  $z \dots z+w$  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  $z \dots z+w$  dimensions have not been truncated and rewrite equation (6) as:

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

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. When we divide the remainder of the equation by  $s_z^{2n}$ , we have:

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

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

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

Again using the argument above, we can show that  $u_{ip} u_{jp} = 0$  for  $z+w+1 \leq p \leq m$ . Furthermore,

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

And our proof is complete.

## 6. CONCLUSIONS AND FUTURE WORK

Higher order co-occurrences play a key role in the effectiveness of systems used for information retrieval. 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 and mathematical studies prove that term co-occurrence plays a crucial role in LSI. The theoretical work shown here will find many practical applications.

This work is the first to study the values produced in the truncated term-term matrix, and we have discovered an explanation for why certain term pairs receive a high similarity value, while others receive low (and even negative) values. Thus we have discovered the basis for the claim that is frequently made for LSI: LSI emphasizes important semantic distinctions (latent semantics) while reducing noise in the data. The correlation between the number of connectivity paths between terms and the value produced in the truncated term-term matrix is another important component in the theoretical foundation for LSI.

Future plans include development of an approximation algorithm for LSI. Our goal is to approximate the LSI term-term matrix using a faster algorithm. This matrix can then be used in place of the LSI matrix in a variety of applications. We also plan to study the higher order co-occurrence used implicitly and explicitly in other systems such as those employing neural networks, vector space and clustering models. Our long-term goal is to construct a theory based on the existence of symmetric, reflexive and transitive relations that determine equivalence classes of terms.

## 7. ACKNOWLEDGMENTS

The authors gratefully acknowledge the assistance of Dr. Kae Kontostathis and Dr. Wei-Min Huang in developing the proof of the transitivity in the SVD as well as in reviewing drafts of this article. The authors also would like to express their gratitude to Dr. Michael Berry for his comments on an early draft. Co-author William M. Pottenger also gratefully acknowledges his Lord and Savior, Jesus Christ, for His guidance in his life.

## 8. REFERENCES

- [1] Berry, Michael, Theresa Do, Gavin O'Brien, Vijay Krishna, and Sowmini Varadhan. 1993. SVDPACKC (Version 1.0) User's Guide. April 1993.
- [2] Berry, Michael, Z. Drmac and E.R. Jessup. 1999. Matrices, Vector Spaces, and Information Retrieval. *SIAM Review*41:2. 1999. pp. 335-362
- [3] Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). "Using linear algebra for intelligent information retrieval." *SIAM Review*, 37(4), 1995, 573-595.
- [4] Chung, Yi-Ming, William M. Pottenger, and Bruce R. Schatz. 1998. Automatic Subject Indexing Using an Associative Neural Network. *Proceedings of Digital Libraries '98*. Pittsburgh, PA., June 1998.
- [5] Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391-407.
- [6] Ding, Chris H.Q. 1999. A similarity-based Probability Model for Latent Semantic Indexing. *Proc of 22<sup>nd</sup> ACM SIGIR '99 Conference*. pp. 59-65.
- [7] Dumais, Susan T. 1993. LSI meets TREC: A status report. *The First Text REtrieval Conference (TREC1)*, D. Harman (Ed.), National Institute of Standards and Technology Special Publication 500-207, pp. 137-152.
- [8] Edmonds, Philip. 1997. Choosing the Word Most Typical in Context Using a Lexical Co-occurrence Network. *Proceedings of the 35<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*. Pages 507-509.
- [9] Forsythe, George E, Michael A. Malcolm, and Cleve B. Moler. 1977. *Computer Methods for Mathematical Computations*. pages 192-235.
- [10] Pottenger, William M., Yong-Bin Kim and Daryl D Meling. 2001. HDDI™: Hierarchical Distributed Dynamic Indexing. In *Data Mining for Scientific and Engineering Applications*. July 2001.
- [11] Schütze, Hinrich. 1992. Dimensions of Meaning. *In Proceedings of Supercomputing '92*.
- [12] Schütze, Hinrich. 1998. Automatic Word Sense Disambiguation. *Computational Linguistics*, Volume 24, number 1.
- [13] 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(03) pp. 329-344.
- [14] Swanson, D.R. 1991. Complementary Structures in disjoint science literatures. In A. Bookstein, et al (Eds), *SIGIR91: Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*. pp 280-289.
- [15] Veling, Anne and Peter van der Weerd. 1999. Conceptual grouping in word co-occurrence networks. *Proceedings of the IJCAI '99*. Volume 2. Pages 694-699.
- [16] Wiemer-Hastings, P. (1999) How Latent is Latent Semantic Analysis? Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, Aug 1999, pp. 932-937. San Francisco: Morgan Kaufmann.
- [17] Xu, Jinxi and W. Bruce Croft. 1998. Corpus-Based Stemming Using Co-occurrence of Word Variants. In *ACM Transactions on Information Systems*, Volume 16, No 1. January 1998. Pages 61-81.
- [18] Zha, Hongyuan. 1998. *A Subspace-Based Model for Information Retrieval with Applications in Latent Semantic Indexing*, Technical Report No. CSE-98-002, Department of Computer Science and Engineering, Pennsylvania State University, 1998.