from *SIAM News*, Volume 32, Number 6

Atlanta Organizers Put Mathematics to Work For the Math Sciences Community

Calling on their experience in developing algorithms and software for the automated indexing model known as LSI, the organizers of SIAM's Atlanta meeting made short work of what can be a frustrating, time-consuming task: organizing the meeting's contributed paper sessions.

By Michael Berry and Jack Dongarra

If you have ever been on the organizing committee for a large meeting, you know that organizers encounter many rewards and frustrations—before, during, and after the meetings. The story we relate here, part of the before side of the 1999 SIAM Annual Meeting, held in Atlanta in May, turned out to be a terrific experience, one that could become standard procedure for future meetings.

One of the pre-meeting chores of the organizing committee is to assemble the contributed paper sessions. This involves reading the abstracts and trying to form groupings of related papers. Having spent several years developing algorithms and software for an automated indexing approach known as Latent Semantic Indexing (or LSI), we decided to automate the grouping of talks, especially since we were running close to the program deadline.

Research in information retrieval has followed several parallel, yet similar, developmental paths. Latent Semantic Indexing, because of the way it represents terms and documents in a term-document space, is considered a vector-space information retrieval model. LSI is one of many candidate methods that can be used to encode terms (keywords) and documents for semantic-based retrieval, i.e., retrieval of relevant information that may or may not share common terminology with an information need (query).

Jack Dongarra, co-chair (with Fan Chung Graham) of the organizing committee for the 1999 SIAM Annual Meeting. Using numerical linear algebra to meet a program deadline, he says, "turned out to be a terrific experience."

Latent Semantic Indexing

The Latent Semantic Indexing information retrieval model [4] builds on earlier research in information retrieval and matrix decompositions, such as use of the singular value decomposition to reduce the dimensions of the term-document space. High-dimensional subspaces (on the order, say, of all the indexed terms in a collection) tend to encode or cluster documents that share terms, whether the documents are related or not. Reductions in dimension (or rank) are an attempt to solve the problems of synonymy and polysemy that plague automatic information retrieval systems (see [2]). Such problems involve terms whose meaning can change from document to document—*bank*, for example, refers sometimes to a financial institution and at other times to a partition of computer memory. Synonyms can also present problems for lexical-matching–based systems—*sprouts*, *toddlers*, *rugrats*, and *kids* are terms used in different situations to designate young children. LSI explicitly represents both terms and documents in a more modest number of dimensions so that the underlying (or latent) semantic relationships between terms and documents can be both accurately and efficiently exploited for query matching.

LSI relies on the constituent terms of a document to suggest the document's semantic content. However, the LSI model views the terms in a document as somewhat unreliable indicators of the concepts contained in the document. The model assumes that the variability of word choice partially obscures the semantic structure of the document. By reducing the dimensionality of the term-document space, LSI reveals the underlying, semantic relationships between documents and eliminates much of the *noise* (differences in word usage, terms that do not help distinguish documents, etc.). LSI statistically analyses the patterns of word usage across the entire document collection, placing documents with similar word-usage patterns near each other in the term-document space, and allowing semantically related documents to be near each other even though they may not share terms.

LSI differs from previous attempts to use reduced-space models for information retrieval in several ways. Most notably, LSI can represent documents in a high-dimensional space whose order is user-specified. Secondly, both terms and documents are explicitly represented in the same space. Thirdly, no attempt is made to interpret the meaning of a dimension. Each dimension is merely assumed to represent one or more semantic relationships in the term-document space. Finally, because of the limits imposed

(mostly) by the computational demands of vector-space approaches to information retrieval, previous attempts focused on relatively small document collections. LSI is able to represent and manipulate large data sets, making it viable for real-world applications.

Compared with other information retrieval techniques, LSI performs surprisingly well. In one test [5], LSI provided more related documents than standard word-based retrieval techniques when searching the standard MED collection. Over five standard document collections, the same study indicated that LSI, on average, performed better than lexical retrieval techniques. In addition, LSI is fully automatic and easy to use, requiring no complex expressions or syntax to represent the query. Because terms and documents are explicitly represented in the space, relevance feedback can be seamlessly integrated with the LSI model, providing even better overall retrieval performance.

In the LSI model, terms and documents are represented by an m by n incidence (or term-by-document) matrix A. Each of the m unique terms in the document collection is assigned a row in the matrix, while each of the n documents in the collection is assigned a column in the matrix. A non-zero element a_{ij} , where

$$A = a_{ij},$$

indicates not only that term i occurs in document j, but also the number of times the term appears in that document. Since the number of terms in a given document is typically far lower than the number of terms in the entire document collection, A is usually very sparse.

Once the m by n matrix A has been created and properly weighted, an orthogonal decomposition known as the singular value decomposition (SVD) is used to compute a rank-k approximation ($k<<\min(m,n)$) to A, A_k. By weighting, we mean that the actual matrix entries (a_{ij}) are rational values, usually defined by

$$a_{ij} = l_{ij} * g_i,$$

where l_{ij} is the local term weight for term i in document j and g_i is the global weighting for term i in the entire collection. Local and global term weightings are typically used either to emphasize or to de-emphasize terms within and across documents, respectively. For our SIAM Annual Meeting abstracts, we used logarithmic local and entropy global term weighting (see [5]):

$$l_i = \log_2(f_{ij} + 1),$$

$$g_i = 1 + \sum [\, p_{ij} \log2(p_{ij})] \,/\, [\, \log_2(n)],$$

$$p_{ij} = f_{ij} \,/\, f_i,$$

where f_{ij} is the frequency of term i in document j and f_i is the global frequency (all counts) of term i in a collection of n documents.

The SVD of the matrix A is defined as the product of three matrices,

$$A = USV^T,$$

where the columns of U and V are the left and right singular vectors, respectively, corresponding to the monotonically decreasing (in value) diagonal elements of S, which are called the singular values of the matrix A. As illustrated in Figure 1, the first k columns of the U and V matrices and the first (largest) k singular values of A are used to construct a rank-k approximation to A via $A_k = U_k S_k V_k^T$. The columns of U and V are orthogonal, such that $U^TU = V^TV = I$, where r is the rank of the matrix A. The A_k, constructed from the k largest singular triplets of A (a singular value and its corresponding left and right singular vectors are referred to as a singular triplet), is the closest rank-k approximation (in the least-squares sense) to A.

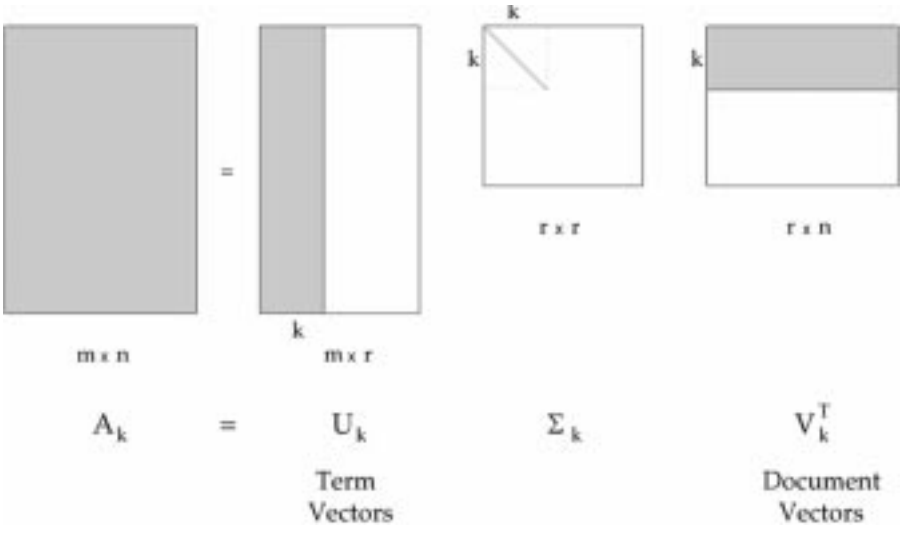

Figure 1. *Pictorial representation of the singular value decomposition. The shaded areas of U and V, along with the diagonal line Σ, represent A_k , the reduced representation of the original term-document matrix A.*

With regard to LSI, A_k is the closest k-dimensional approximation to the original term-document space represented by the incidence matrix A. As stated earlier, reducing the dimensionality of A is believed to result in elimination of much of the noise that causes poor retrieval perfor-mance. Thus, although a high-dimension-al representation appears to be required for good retrieval performance, care must be taken to

avoid reconstructing A. If A is nearly reconstructed, the noise caused by variability of word choice and terms that span or nearly span the document collection will not be eliminated, and retrieval performance will be poor.

In LSI, the left and right singular vectors specify the locations of the terms and documents, respectively, in the reduced term-document space. The singular values are often used to scale the term and docu-ment vectors, allowing clusters of terms and documents to be more readily identi-fied. Within the reduced space, semantical-ly related terms and documents presumab-ly lie near each other, since the SVD at-tempts to derive the underlying, semantic structure of the term-document space.

Query Projection and Matching

In the LSI model, queries are formed into pseudo-documents that specify the location of the query in the reduced term-document space. Given q, a vector whose non-zero elements contain the weighted (by the same local and global weighting schemes applied to the document collection being searched) term-frequency counts of the terms that appear in the query, the pseudo-document, $\underline{q}$, can be represented by

$$\underline{q} = q^T U_k S_k^{-1}.$$

Thus, the pseudo-document consists of the sum of the term vectors ($q^T U_k$) corresponding to the terms specified in the query scaled by the inverse of the singular values (S_k^{-1}). The singular values are used to individually weight each dimension of the term-document space.

Once the query has been projected into the term-document space, one of several similarity measures can be applied to compare the position of the pseudo-document with the positions of the terms or documents in the reduced term-document space. One popular measure, the cosine similarity measure, finds the angle between the pseudo-document and the terms or documents only in the reduced space; as a result, the lengths of the documents, which can affect the distance between the pseudo-document and the documents in the space, are normalized. Once the similarities between the pseudo-document and all the terms and documents in the space have been computed, the terms or documents are ranked according to the results of the similarity measure and the highest-ranking terms or documents, or all the terms and documents exceeding some threshold value, are returned to the user.

Assigning Contributed Papers To Meeting Sessions

To organize the contributed sessions for the SIAM meeting, we constructed a term-by-abstract matrix A of order 1149 by 128, corresponding to the 128 submitted papers. A reduced LSI model of rank $k = 44$ (A_{44}) was generated with SVDPACKC (see [3]) to produce both term and document (abstract) vector representations. Each of the 19 different themes identified in the preliminary announcement of the meeting was used as a query (q) into the vector-space model, and the 10 top-ranked abstracts (in cosine similarity to q) were recorded. Among the most popular themes were combinatorial optimization, high-performance computing, numerical linear algebra, PDEs and control of PDEs, and mathematical biology.

As discussed by Berry and Browne [1], automated indexing cannot be expected to generate the same results as a human indexer. Some judgments were made in the final assignments of the papers to sessions, but the entire construction of contributed paper program was basically done in two days (yes, that's correct!). Had we been required to read each abstract word for word, the process would certainly have taken much longer to complete.

One additional advantage of using LSI for an exercise of this type is that we quickly got a sense of how well the contributed papers actually matched the advertised themes. Abstracts that did not match very well to any of the proposed themes (queries) were then compared with each other (i.e., cosines between document vectors were computed) to create new sessions with papers of similar content. Again, some human judgments were needed during this phase, to control the total number of sessions allocated, and similar intrusions were needed to balance the numbers of papers assigned to individual sessions. In one sense, then, the entire process was not totally automated, but the use of applied mathematics (or, in this case, numerical linear algebra) did play a major role in meeting the dead-line handed to us by SIAM. Using mathematics to organize mathematics seems somewhat aesthetically pleasing, doesn't it?

Acknowledgment

The authors would like to thank Tammy Kolda for a stimulating discussion of LSI and its uses.

Further Reading

For more details on LSI and related computational models for information retrieval, we recommend the new (1999) book [1] by Berry and Browne , published in the SIAM Book Series *Software, Environments, and Tools*, and a recent *SIAM Review* article [2] by Berry, Drmac, and Jessup.

References

[1] M.W. Berry and M. Browne, *Understanding Search Engines: Mathematical Modeling and Text Retrieval,* SIAM, Philadelphia, 1999.

[2] M.W. Berry, Z. Drmac, and E.R. Jessup, *Matrices, vector spaces, and information retrieval,* SIAM Rev., 41:2, 1999, 335–362.

[3] M. Berry, T.Do, G. O'Brien, V. Krishna, and S. Varadhan, *SVDPACKC: Version 1.0 User's Guide,* Tech. Report CS-93-194, University of Tennessee, Knoxville, October 1993.

[4] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, *Indexing by latent semantic analysis,* J. Amer. Soc. for Inform. Sc.,

41, 1990, 391–407.
[5] S.T. Dumais, *Improving the retrieval of information from external sources*, Behavior Res. Meth., Instru., & Comp., 23, 1991, 229–236.

Michael Berry is an associate professor of computer science at the University of Tennessee, Knoxville. Jack Dongarra is a professor of computer science at the University of Tennessee, Knoxville, and a scientist at Oak Ridge National Laboratory.