Latent Semantic Analysis and Fiedler Retrieval

Bruce Hendrickson

Discrete Algorithms & Math Dept.
Sandia National Labs
Albuquerque, New Mexico
Also, CS Department, UNM

Sandia National Laboratories

Informatics & Linear Algebra

- **Eigenvectors of graphs** (convergence of iterative process)
  - » Bibliometrics
  - » PageRank, HITS and descendents
  - » TrustRank, etc.

- **Singular vectors of data matrix** (Rank reduction techniques)
  - » Latent semantic analysis (LSA/LSI)
  - » Text retrieval, image recognition, etc.
  - » Tensor techniques, etc.

Yet Another Matrix

- Laplacian matrix of a graph
  - Widely used in spectral graph theory
  - Less common in informatics
    - Some usage in clustering (e.g. Dhillon'01)

- Goal of this talk:
  - Identify connection between LSA and eigenvectors of Laplacian matrices
  - Suggest new applications enabled by this connection
    - e.g. unified link and textual analysis

Outline

- **Review of Latent Semantic Analysis (LSA)**
- **New Problem – Embedding a graph**
  - » **“Fiedler embedding”**
- **Essential equivalence to LSA**
- **New generalizations of LSA**

Vector Space Model of Information

- **Developed by Gerald Salton**

- **Start with Term-Document matrix $A \in R^{t \times d}$**
  - » **Scaled version $B = D_t\, A\, D_d$**

- **Document similarities = $B^T B$**

- **Query is a vector q of term values**
  - » **Answer is *similar* documents, i.e. large entries in $B^T q$**
    - – **Angular similarity common, normalize appropriately**

Latent Semantic Analysis

- LSA uses **truncated SVD** for dimension reduction
  - » $B \approx B_k = U_k \Sigma_k V_k^T$
  - » Best rank-k approximation to B in the Frobenius norm
    - – Eckart-Young theorem
- Document similarities
  - » $B_k^T B_k = V_k \Sigma_k^2 V_k^T$
- Query: large entries in
  - » $\Sigma_k^{1/2} U_k^T q$

(Seemingly) Different Problem

- **Embedding a Graph in *k*-Space**

- **Given graph $G=(V,E)$, with edge weights $w_{i,j}$**
  - » **Weights encode *similarity* of two vertices**

- **Place vertices in *k*-space to keep similar vertices near each other**
  - » **That is, keep edge-lengths short**
  - » **Let p_r be the location of vertex r in k-space**
  - » **Minimize $\Sigma_{(r,s)\in E}\, w_{r,s}|p_r - p_s|^2$**

Matrix Interpretation

- **Minimize $\Sigma_{(r,s)\in E}\, w_{r,s}|p_r - p_s|^2$**

- **Laplacian matrix**
  - $$L(i,j) = \begin{cases} -w_{i,j} & \text{If } (i, j) \text{ is an edge} \\ \sum_k w_{i,k} & \text{For diagonal entry } (i,i) \\ 0 & \text{Otherwise} \end{cases}$$

- **After some algebra:**
  - **Minimize$_P$ Trace $(P^T L P)$**
  - **Where $P \in R^{n \times k}$ is matrix of n positions**

Need Constraints

- **Minimize Trace $(P^T L P)$**

- **Solution invariant under translations**
  - » **Place center of mass at origin**
  - » **(Constraint 1) $P^T 1_n = 0_k$**

- **Trivial solution of all points at origin**
  - » **(Constraint 2) for $i=1,\ldots,k$ $P_i^T P_i = \gamma_i$**

- **Coordinates should be distinct**
  - » **(Constraint 3) for $i \neq j$ $P_i^T P_j = 0$**

Fiedler Embedding

- **Minimize Trace** $(P^T L P)$
  - » **Such that:**
    - – $P^T 1_n = 0_k$
    - – $P^T P = \Gamma$ **(diagonal)**
- **Laplacian Eigenvectors**
  - » 1_n **is eigenvector with smallest eigenvalue (zero)**
- **Solution:**
  - » **Columns of** P **are eigenvectors 2 through** $k+1$ **of** L**.**
  - » **Scaled by** $\sqrt{\Gamma_{i,i}}$
  - » $$P = \Gamma^{\frac{1}{2}} W_{\hat{k}}$$

Adding New Items to k-Space

- **Given new item with some similarities to current items, place it in *k*-space**
  - **This is the heart of an LSA query q**

- **Find p_x to Minimize $\Sigma_{(r,x)\in E}\, w_{r,x}|p_r - p_x|^2$**

- **Solution**

$$p_x = \frac{\sum w_{s,x} p_s}{\sum w_{s,x}} = \frac{Z^T q}{\|q\|_1} = \frac{\Gamma^{\frac{1}{2}} W_{\hat{k}}^{\ T} q}{\|q\|_1}$$

- **Recall LSA query: $\Sigma_k^{1/2}\, U_k^T\, q$**

Term-Document Embedding

- **Apply Laplacian embedding to information analysis**
  - » **Start with canonical term-document example**
- **Let objects be terms *and* documents**
  - » $L \in R^{(t+d) \times (t+d)}$
- **Graph is bipartite:**
  - » **No term-term or document-document edges**
- **Think of entries *B* as term-document similarities**

- **Embedding involves eigenvectors of**

$$L = \begin{pmatrix} D_1 & -B^T \\ -B & D_2 \end{pmatrix}$$

Eigenvectors & Singular Vectors

- **LSA works with largest singular vectors of *B***

- **Equivalent to largest eigenvectors of**

$$
\begin{array}{cc}
 & \begin{array}{cc} d & t \end{array} \\
M = & \begin{pmatrix} 0 & B^T \\ B & 0 \end{pmatrix}
\end{array}
$$

- **That is**
  - » **if *(u, σ, v)* comprises a singular triplet of *B*,**
  - » **Then *(σ, v:u)* is an eigenpair of *M.***

Scaling

- Recall, $B = D_t\, A\, D_d$
- Choose D_t and D_d to make B doubly stochastic
  - » (row/column sums equal 1)
  - » E.g. Sinkhorn algorithm
- LSA Matrix: $M = \begin{pmatrix} 0 & B^T \\ B & 0 \end{pmatrix}$
- Laplacian: $L = \begin{pmatrix} I & -B^T \\ -B & I \end{pmatrix} = I - M$
- Leading eigenvectors of M = trailing eigenvectors of L.

Essential Equivalence

- **Theorem:**
  - » **If B is doubly stochastic and $\Gamma = \Sigma$, then LSA embedding is identical to Laplacian embedding**
  - » **Caveat: Laplacian discards trivial first vector**
- **Theorem:**
  - » **If query vector has 1-norm of one, geometry of LSA queries are identical to Laplacian queries**
  - » **Caveat: LSA typically uses angular distance, whereas Laplacian approach most naturally uses Euclidean**

Advantages I

- New way of thinking about LSA
  - » Optimal placement to minimize distances
  - » Alternative intuition
- Terms & Documents live in same space
  - » Principled method for adding document-document similarities or term-term similarities to embedding
    - – E.g. former from dictionary, latter from co-citation analysis or hyperlinks
    - – Unified text and link analysis

$$L = \begin{pmatrix} G_1 & -B^T \\ -B & G_2 \end{pmatrix}$$

Advantages II

- **Supports more complex queries**
  - **"similar to these documents and these terms"**

- **Supports extensions to more classes of objects.**
  - **E.g., instead of just term-document, could do term-document-author.**

$$
\begin{array}{cc}
 & \begin{array}{ccc} \quad d \quad & \quad t \quad & \quad a \quad \end{array} \\
L = & \begin{pmatrix} D_1 & -B^T & -C^T \\ -B & D_2 & -E^T \\ -C & -E & D_3 \end{pmatrix}
\end{array}
$$

Alternative to Tensors

- Tensors are higher dimensional generalizations of matrices
  - E.g. terms-by-document-by-author
  - Active area for informatics research
- Drawbacks
  - No factorization with all the SVD properties
  - Lack of efficient algorithms
- Current approach has some of the advantages of tensors, without the limitations

Conclusions

- New algebraic/geometric approach for information retrieval
- Closely related to LSA
- Supports novel enhancements and extensions in a principled way
  - » Unified text and link analysis
  - » More complex types of queries

Acknowledgements

- Thanks to Erik Boman, Brett Bader, Tammy Kolda, Liz Jessup, Inderjit Dhillon, and Petros Drineas.

- bah@sandia.gov
- www.cs.sandia.gov/~bahendr

- Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the US DOE under contract DE-AC-94AL85000. This work was funded by Sandia's LDRD Program.