The Use of the Linear Algebra by Web Search Engines

Amy N. Langville and Carl D. Meyer
Department of Mathematics, North Carolina State University
Raleigh, NC 27695-8205

October 19, 2004

1 Introduction

Nearly all the major Web search engines today use link analysis to improve their search results. That's exciting news for linear algebraists because link analysis, the use of the Web's hyperlink structure, is built from fundamentals of matrix theory. Link analysis and its underlying linear algebra have helped revolutionize Web search, so much so that the pre-link analysis search (before 1998) pales in comparison to today's remarkably accurate search.

HITS [13] and PageRank [2, 3] are two of the most popular link analysis algorithms. Both were developed around 1998 and both have dramatically improved the search business. In order to appreciate the impact of link analysis, recall for a minute the state of search prior to 1998. Because of the immense number of pages on the Web, a query to an engine often produced a very long list of relevant pages, sometimes thousands of pages long. A user had to sort carefully through the list to find the most relevant pages. The order of presentation of the pages was little help because spamming was so easy then. In order to trick a search engine into producing rankings higher than normal, spammers used meta-tags liberally, claiming their page used popular search terms that never appeared in the page. Meta-tags became useless for search engines. Spammers also repeated popular search terms in invisible text (white text on a white background) to fool engines.

2 The HITS Algorithm

HITS [13], a link analysis algorithm developed by Jon Kleinberg from Cornell University during his postdoctoral studies at IBM Almaden, aimed to focus this long, unruly query list. The HITS algorithm is based on a pattern Kleinberg noticed among Web pages. Some pages serve as hubs or portal pages, i.e., pages with many outlinks. Other pages are authorities on topics because they have many inlinks. Kleinberg noticed that *good hubs seemed to point to good authorities and good authorities were pointed to by good hubs.* So he decided to give each page i both an hub score h_i and an authority score a_i. In fact, for every page i he defined the hub score at iteration k, $h_i^{(k)}$, and the authority score, $a_i^{(k)}$, as

$$a_i^{(k)} = \sum_{j:e_{ji}\in E} h_j^{(k-1)} \quad \text{and} \quad h_i^{(k)} = \sum_{j:e_{ij}\in E} a_j^{(k)} \quad \text{for } k = 1, 2, 3, \ldots,$$

where e_{ij} represents a hyperlink from page i to page j and E is the set of hyperlinks. To compute the scores for a page, he started with uniform scores for all pages, i.e., $h_i^{(1)} = 1/n$ and $a_i^{(1)} = 1/n$ where n is the number of pages in a so-called neighborhood set for the query list. The neighborhood set consists of

all pages in the query list plus all pages pointing to or from the query pages. Depending on the query, the neighborhood set could contain just a hundred pages or a hundred thousand pages. (The neighborhood set allows latent semantic associations to be made.) The hub and authority scores are iteratively refined until convergence to stationary values.

Using linear algebra we can replace the summation equations with matrix equations. Let $\mathbf{h}$ and $\mathbf{a}$ be column vectors holding the hub and authority scores. Let $\mathbf{L}$ be the adjacency matrix for the neighborhood set. That is, $\mathbf{L}_{ij} = 1$ if page i links to page j, and 0, otherwise. These definitions show that

$$\mathbf{a}^{(k)} = \mathbf{L}^T\mathbf{h}^{(k-1)} \quad \text{and} \quad \mathbf{h}^{(k)} = \mathbf{L}\mathbf{a}^{(k)}.$$

Using some algebra, we have

$$\begin{aligned} \mathbf{a}^{(k)} &= \mathbf{L}^T\mathbf{L}\mathbf{a}^{(k-1)} \\ \mathbf{h}^{(k)} &= \mathbf{L}\mathbf{L}^T\mathbf{h}^{(k-1)}. \end{aligned}$$

These equations make it clear that Kleinberg's algorithm is really the power method applied to the positive semi-definite matrices $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$. $\mathbf{L}^T\mathbf{L}$ is called the hub matrix and $\mathbf{L}\mathbf{L}^T$ is the authority matrix. Thus, HITS amounts to solving the eigenvector problems $\mathbf{L}^T\mathbf{L}\mathbf{a} = \lambda_1\mathbf{a}$ and $\mathbf{L}\mathbf{L}^T\mathbf{h} = \lambda_1\mathbf{h}$, where λ_1 is the largest eigenvalue of $\mathbf{L}^T\mathbf{L}$ (and $\mathbf{L}\mathbf{L}^T$), and $\mathbf{a}$ and $\mathbf{h}$ are corresponding eigenvectors.

While this is the basic linear algebra required by the HITS method, there are many more issues to be considered. For example, important issues include convergence, existence, uniqueness, and numerical computation of these scores [5, 7, 14]. Several modifications to HITS have been suggested, each bringing various advantages and disadvantages [4, 6, 8]. A variation of Kleinberg's HITS concept is at the base of the search engine TEOMA (`http://www.teoma.com`), which is owned by Ask Jeeves, Inc.

3 The PageRank Algorithm

PageRank, the second link analysis algorithm from 1998, is the heart of Google. Both PageRank and Google were conceived by Sergey Brin and Larry Page while they were computer science graduate students at Stanford University. Brin and Page use a recursive scheme similar to Kleinberg's. Their original idea was that a *page is important if it is pointed to by other important pages.* That is, they decided that the importance of your page (its PageRank score) is determined by summing the PageRanks of all pages that point to yours. In building a mathematical definition of PageRank, Brin and Page also reasoned that when an important page points to several places, its weight (PageRank) should be distributed proportionately. In other words, if YAHOO! points to your Web page, that's good, but you shouldn't receive the full weight of YAHOO! because they point to many other places. If YAHOO! points to 999 pages in addition to yours, then you should only get credit for 1/1000 of YAHOO!'s PageRank.

This reasoning led Brin and Page to formulate a recursive definition PageRank. They defined

$$r_i^{(k+1)} = \sum_{j \in I_i} \frac{r_j^{(k)}}{|O_j|},$$

where $r_i^{(k)}$ is the PageRank of page i at iteration k, I_i is the set of pages pointing into page i and $|O_j|$ is the number of outlinks from page j. Like HITS, PageRank starts with a uniform rank for all pages, i.e., $r_i^{(0)} = 1/n$ and successively refines these scores, where n is the total number of Web pages.

Like HITS, we can write this process using matrix notation. Let the row vector $\boldsymbol{\pi}^{(k)T}$ be the PageRank vector at the k^{th} iteration. As a result, the summation equation for PageRank can be written compactly as

$$\boldsymbol{\pi}^{(k+1)T} = \boldsymbol{\pi}^{(k)T}\mathbf{H},$$

where $\mathbf{H}$ is a row normalized hyperlink matrix, i.e., $h_{ij} = 1/|O_i|$, if there is a link from page i to page j, and 0, otherwise. Unfortunately, this iterative procedure has convergence problems—it can cycle or the limit may be dependent on the starting vector.

To fix these problems, Brin and Page revised their basic PageRank concept. Still using the hyperlink structure of the Web, they build an irreducible aperiodic Markov chain characterized by a primitive (irreducible with only one eigenvalue on the spectral circle) transition probability matrix. The irreducibility guarantees the existence of a unique stationary distribution vector $\boldsymbol{\pi}^T$, which becomes the PageRank vector. The power method with a primitive stochastic iteration matrix will always converge to $\boldsymbol{\pi}^T$ independent of the starting vector, and the asymptotic rate of convergence is governed by the magnitude of the subdominant eigenvalue λ_2 of the transition matrix [19].

Here's how Google turns the hyperlink structure of the Web into a primitive stochastic matrix. If there are n pages in the Web, let $\mathbf{H}$ be the $n \times n$ matrix whose element h_{ij} is the probability of moving from page i to page j in one click of the mouse. The simplest model is to take $h_{ij} = 1/|O_i|$, which means that starting from any Web page we assume that it is equally likely to follow any of the outgoing links to arrive at another page.

However, some rows of $\mathbf{H}$ may contain all zeros, so $\mathbf{H}$ is not necessarily stochastic. This occurs whenever a page contains no outlinks; many such pages exist on the Web and are called *dangling nodes.* An easy fix is to replace all zero rows with $\mathbf{e}^T/n$, where $\mathbf{e}^T$ is the row vector of all ones. The revised (now stochastic) matrix $\mathbf{S}$ can be written as a rank-one update to the sparse $\mathbf{H}$. Let $\mathbf{a}$ be the dangling node vector in which

$$a_i = \begin{cases} 1 & \text{if page } i \text{ is a dangling node,} \\ 0 & \text{otherwise.} \end{cases}$$

Then,

$$\mathbf{S} = \mathbf{H} + \mathbf{a}\mathbf{e}^T/n.$$

Actually, any probability vector $\mathbf{p}^T > 0$ with $\mathbf{p}^T\mathbf{e} = 1$ can be used in place of the uniform vector $\mathbf{e}^T/n$.

We're not home yet because the adjustment that produces the stochastic matrix $\mathbf{S}$ isn't enough to insure the existence of a *unique* stationary distribution vector (needed to make PageRank well defined). Irreducibility on top of stochasticity is required. But the link structure of the Web is reducible—the Web graph is not strongly connected. Consequently, an adjustment to make $\mathbf{S}$ irreducible is needed. This last adjustment brings us to the *Google matrix*, which is defined to be

$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)\mathbf{E},$$

where $0 \leq \alpha \leq 1$ and $\mathbf{E} = \mathbf{e}\mathbf{e}^T/n$. Google eventually replaced the uniform vector $\mathbf{e}^T/n$ with a more general probability vector $\mathbf{v}^T$ (so that $\mathbf{E} = \mathbf{e}\mathbf{v}^T$) to allow them the flexibility to make adjustments to PageRanks as well as to personalize them. See [10, 15] for more about the personalization vector $\mathbf{v}^T$.

Because $\mathbf{G}$ is a convex combination of the two stochastic matrices $\mathbf{S}$ and $\mathbf{E}$, it follows that $\mathbf{G}$ is both stochastic and irreducible. Furthermore, every node is now directly connected to every other node (although the probability of transition may be very small in some cases), so $\mathbf{G} > 0$. Consequently, $\mathbf{G}$ is a primitive matrix, and this insures that the power method $\boldsymbol{\pi}^{(k+1)T} = \boldsymbol{\pi}^{(k)T}\mathbf{G}$ will converge, independent of the starting vector, to a unique stationary distribution $\boldsymbol{\pi}^T$ [19]. This is the mathematical part of Google's PageRank vector.

3.1 The Power Method

While it doesn't always excite numerical analysts, the power method has been Google's computational method of choice, and there are some good reasons for this. First, consider iterates of the power method applied to $\mathbf{G}$ (a completely dense matrix, were it to be formed explicitly). If we take $\mathbf{E} = \mathbf{e}\mathbf{v}^T$, then

$$\boldsymbol{\pi}^{(k)T} = \boldsymbol{\pi}^{(k-1)T}\mathbf{G} = \alpha\boldsymbol{\pi}^{(k-1)T}\mathbf{S} + (1-\alpha)\mathbf{v}^T = \alpha\boldsymbol{\pi}^{(k-1)T}\mathbf{H} + (\alpha\boldsymbol{\pi}^{(k-1)T}\mathbf{a} + (1-\alpha))\mathbf{v}^T,$$

Written in this way, it becomes clear that the power method applied to $\mathbf{G}$ can be implemented with vector-matrix multiplications on the extremely sparse $\mathbf{H}$, and $\mathbf{G}$ and $\mathbf{S}$ are never formed or stored. A matrix-free method such as the power method is required due to the size of the matrices and vectors involved (Google's index is currently 4.3 billion pages). Fortunately, since $\mathbf{H}$ is sparse, each vector-matrix multiplication required by the power method can be computed in $nnz(\mathbf{H})$ flops, where $nnz(\mathbf{H})$ is the number of nonzeros in $\mathbf{H}$. And since the average number of nonzeros per row in $\mathbf{H}$ is significantly less than 10, we have $O(nnz(\mathbf{H})) \approx O(n)$. Furthermore, at each iteration, the power method only requires the storage of one vector, the current iterate, whereas other accelerated matrix-free methods, such as restarted GMRES or BiCGStab, require storage of at least several vectors, depending on the size of the subspace chosen. Finally, the power method applied in this way converges quickly. Brin and Page report success using only 50 to 100 power iterations [3]. This is due in large part to the fact that it can be proven [15] that the subdominant eigenvalue of $\mathbf{G}$ satisfies $|\lambda_2| \leq \alpha$, and Google originally set $\alpha = .85$.

Like HITS, the basic concepts of PageRank are simple, but there are many subtle issues that lurk just below the surface. For example, there are complicated and unresolved issues concerning personalization, computation, accelerated computation, sensitivity, and updating—more information is available in [7, 11, 12, 21, 15, 17, 18, 20].

This brief introduction describes only the mathematical component of Google's ranking system. However, it's known that there are non-mathematical "metrics" that are also considered when Google responds to a query, so the results seen by a user are in fact PageRank tempered by other metrics. While Google is secretive about these other metrics, they state on their Web site (`http://www.google.com/technology`) that "The heart of our software is PageRank... ."

4 Books

SIAM is publishing a second edition of the popular *Understanding Search Engines: Mathematical Modeling and Text Retrieval* [1] by Michael W. Berry and Murray Browne in 2005. The new edition contains a chapter devoted to link analysis. As a result, readers can see how link analysis and ranking algorithms fit into the overall search process.

Also due out in 2005 is our book, *Understanding Web Search Engine Rankings: Google's PageRank, Teoma's HITS, and other ranking algorithms* [16]. This book from Princeton University Press will contain over 250 pages devoted to link analysis algorithms with several introductory chapters, examples, and code, as well as chapters dealing with more advanced issues in Web search ranking.

References

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

[2] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 33:107–117, 1998.

[3] Sergey Brin, Lawrence Page, R. Motwami, and Terry Winograd. The PageRank citation ranking: bringing order to the web. Technical report, Computer Science Department, Stanford University, 1998.

[4] Chris Ding, Xiaofeng He, Hongyuan Zha, and Horst Simon. PageRank, HITS and a unified framework for link analysis. In *Proceedings of the 25th ACM SIGIR Conference*, pages 353–354, Tampere, Finland, August 2002.

[5] Chris H. Q. Ding, Hongyuan Zha, Xiaofeng He, Parry Husbands, and Horst D. Simon. Link analysis: hubs and authorities on the World Wide Web. *SIAM Review*, 46(2):256–268, 2004.

[6] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, F. Schaefer, and Lesley A. Ward. Modifications of Kleinberg's HITS algorithm using matrix exponentiation and web log records. In *ACM SIGIR Conference*, pages 444–445, September 2001.

[7] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, and Lesley A. Ward. Existence and uniqueness of ranking vectors for linear link analysis. In *ACM SIGIR Conference*, September 2001.

[8] Francois Fouss, Jean-Michel Renders, and Marco Saerens. Some relationships between kleinberg's hubs and authorities, correspondence analysis, and the Salsa algorithm. In *Proceedings of the 7th International Conference on the Statistical Analysis of Textual Data (JADT 2004)*, pages 445–455, 2004.

[9] Taher H. Haveliwala and Sepandar D. Kamvar. The second eigenvalue of the Google matrix. Technical report, Stanford University, 2003.

[10] Taher H. Haveliwala, Sepandar D. Kamvar, and Glen Jeh. An analytical comparison of approaches to personalizing PageRank. Technical report, Stanford University, 2003.

[11] Sepandar D. Kamvar and Taher H. Haveliwala. The condition number of the PageRank problem. Technical report, Stanford University, 2003.

[12] Sepandar D. Kamvar, Taher H. Haveliwala, and Gene H. Golub. Adaptive methods for the computation of PageRank. Technical report, Stanford University, 2003.

[13] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46, 1999.

[14] Amy N. Langville and Carl D. Meyer. A survey of eigenvector methods of web information retrieval. *The SIAM Review*, 2003. Accepted in December 2003.

[15] Amy N. Langville and Carl D. Meyer. Deeper inside PageRank. *Internet Mathematics Journal*, 2004. Accepted in February 2004.

[16] Amy N. Langville and Carl D. Meyer. *Understanding Web Search Engine Rankings: Google's PageRank, Teoma's HITS, and other ranking algorithms*. Princeton University Press, Princeton, 2005.

[17] Chris Pan-Chi Lee, Gene H. Golub, and Stefanos A. Zenios. Partial state space aggregation based on lumpability and its application to PageRank. Technical report, Stanford University, 2003.

[18] Ronny Lempel and Shlomo Moran. Rank-stability and rank-similarity of link-based web ranking algorithms in authority-connected graphs. In *Second Workshop on Algorithms and Models for the Web-Graph (WAW 2003)*, Budapest, Hungary, May 2003.

[19] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, 2000.

[20] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link analysis, eigenvectors and stability. In *Seventh International Joint Conference on Artificial Intelligence*, 2001.

[21] Twelfth International World Wide Web Conference. *Extrapolation Methods for Accelerating PageRank Computations*, 2003.