Chapter 12

# Page Ranking for a Web Search Engine

When a search is made on the Internet using a search engine, there is first a traditional text processing part, where the aim is to find all the Web pages containing the words of the query. Due to the massive size of the Web, the number of hits is likely to be much too large to be of use. Therefore, some measure of quality is needed to filter out pages that are assumed to be less interesting.

When one uses a Web search engine it is typical that the search phrase is underspecified.

**Example 12.1.** A Google[28] search conducted on September 29, 2005, using the search phrase *university*, gave as a result links to the following well-known universities: *Harvard*, *Stanford*, *Cambridge*, *Yale*, *Cornell*, *Oxford*. The total number of Web pages relevant to the search phrase was more than 2 billion. ∎

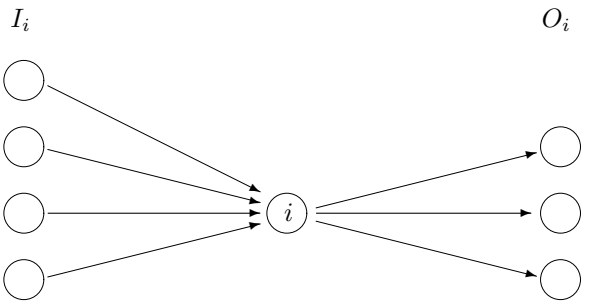
Obviously Google uses an algorithm for ranking all the Web pages that agrees rather well with a common-sense quality measure. Somewhat surprisingly, the ranking procedure is based not on human judgment but on the link structure of the Web. Loosely speaking, Google assigns a high rank to a Web page if it has inlinks from other pages that have a high rank. We will see that this self-referencing statement can be formulated mathematically as an eigenvalue equation for a certain matrix.

## 12.1 Pagerank

It is of course impossible to define a generally valid measure of relevance that would be acceptable for all users of a search engine. Google uses the concept of *pagerank* as a quality measure of Web pages. It is based on the assumption that the number of links to and from a page give information about the importance of a page. We will give a description of pagerank based primarily on [74] and [33]. Concerning Google, see [19].

[28] http://www.google.com/.

Let all Web pages be ordered from 1 to $n$, and let $i$ be a particular Web page. Then $O_i$ will denote the set of pages that $i$ is linked to, the *outlinks*. The number of outlinks is denoted $N_i = |O_i|$. The set of *inlinks*, denoted $I_i$, are the pages that have an outlink to $i$.




In general, a page $i$ can be considered as more important the more inlinks it has. However, a ranking system based only on the number of inlinks is easy to manipulate:[29] when you design a Web page $i$ that (e.g., for commercial reasons) you would like to be seen by as many users as possible, you could simply create a large number of (informationless and unimportant) pages that have outlinks to $i$. To discourage this, one defines the rank of $i$ so that if a highly ranked page $j$ has an outlink to $i$, this adds to the importance of $i$ in the following way: the rank of page $i$ is a weighted sum of the ranks of the pages that have outlinks to $i$. The weighting is such that the rank of a page $j$ is divided evenly among its outlinks. Translating this into mathematics, we get

$$r_i = \sum_{j \in I_i} \frac{r_j}{N_j}. \tag{12.1}$$

This preliminary definition is recursive, so pageranks cannot be computed directly. Instead a fixed-point iteration might be used. Guess an initial ranking vector $r^0$. Then iterate

$$r_i^{(k+1)} = \sum_{j \in I_i} \frac{r_j^{(k)}}{N_j}, \quad k = 0, 1, \ldots. \tag{12.2}$$

There are a few problems with such an iteration: if a page has no outlinks, then in the iteration process it accumulates rank only via its inlinks, but this rank is never distributed further. Therefore it is not clear if the iteration converges. We will come back to this problem later.

More insight is gained if we reformulate (12.1) as an eigenvalue problem for a matrix representing the graph of the Internet. Let $Q$ be a square matrix of dimension $n$. Define

$$Q_{ij} = \begin{cases} 1/N_j & \text{if there is a link from } j \text{ to } i, \\ 0 & \text{otherwise.} \end{cases}$$

[29] For an example of attempts to fool a search engine, see [96] and [59, Chapter 5].

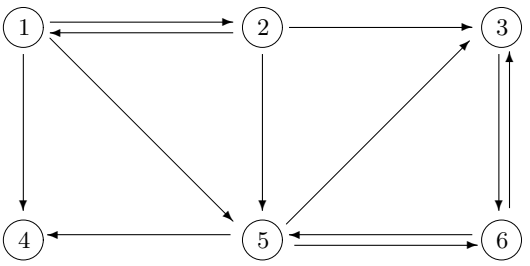
This means that row $i$ has nonzero elements in the positions that correspond to inlinks of $i$. Similarly, column $j$ has nonzero elements equal to $N_j$ in the positions that correspond to the outlinks of $j$, and, provided that the page has outlinks, the sum of all the elements in column $j$ is equal to one. In the following symbolic picture of the matrix $Q$, nonzero elements are denoted $*$:

$$\begin{array}{c} \\ i \\ \\ \end{array} \begin{array}{c} \begin{array}{c} j \end{array} \\ \begin{pmatrix} & & * & & \\ 0 & * & 0 & * & * \\ & & \vdots & & \\ & & * & & \\ & & 0 & & \\ & & * & & \end{pmatrix} \end{array} \begin{array}{l} \\ \leftarrow \text{ inlinks} \\ \\ \end{array}$$

$$\uparrow$$

outlinks

**Example 12.2.** The following link graph illustrates a set of Web pages with outlinks and inlinks:




The corresponding matrix becomes

$$Q = \begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & \frac{1}{3} & 0 \end{pmatrix}.$$

Since page 4 has no outlinks, the corresponding column is equal to zero. ∎

Obviously, the definition (12.1) is equivalent to the scalar product of row $i$ and the vector $r$, which holds the ranks of all pages. We can write the equation in

matrix form,

$$\lambda r = Qr, \qquad \lambda = 1, \tag{12.3}$$

i.e., $r$ is an *eigenvector* of $Q$ with *eigenvalue* $\lambda = 1$. It is now easily seen that the iteration (12.2) is equivalent to

$$r^{(k+1)} = Qr^{(k)}, \qquad k = 0, 1, \ldots,$$

which is the *power method* for computing the eigenvector. However, at this point it is not clear that pagerank is well defined, as we do not know if there exists an eigenvalue equal to 1. It turns out that the theory of Markov chains is useful in the analysis.

## 12.2 Random Walk and Markov Chains

There is a random walk interpretation of the pagerank concept. Assume that a surfer visiting a Web page chooses the next page among the outlinks with equal probability. Then the random walk induces a Markov chain (see, e.g., [70]). *A Markov chain is a random process in which the next state is determined completely from the present state; the process has no memory.* The transition matrix of the Markov chain is $Q^T$. (Note that we use a slightly different notation than is common in the theory of stochastic processes.)

The random surfer should never get stuck. In other words, our random walk model should have no pages without outlinks. (Such a page corresponds to a zero column in $Q$.) Therefore the model is modified so that zero columns are replaced with a constant value in all positions. This means that there is equal probability to go to any other Internet page. Define the vectors

$$d_j = \begin{cases} 1 & \text{if } N_j = 0, \\ 0 & \text{otherwise,} \end{cases}$$

for $j = 1, \ldots, n$, and

$$e = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^n. \tag{12.4}$$

The modified matrix is defined

$$P = Q + \frac{1}{n} e d^T. \tag{12.5}$$

With this modification the matrix $P$ is a proper *column-stochastic matrix*: It has nonnegative elements, and the elements of each column sum up to 1. The preceding statement can be reformulated as follows.

**Proposition 12.3.** *A column-stochastic matrix $P$ satisfies*

$$e^T P = e^T, \tag{12.6}$$

*where $e$ is defined by* (12.4).

**Example 12.4.** The matrix in the previous example is modified to

$$P = \begin{pmatrix} 0 & \frac{1}{3} & 0 & \frac{1}{6} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{6} & 0 & 0 \\ 0 & \frac{1}{3} & 0 & \frac{1}{6} & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & \frac{1}{6} & \frac{1}{3} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{6} & 0 & \frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{6} & \frac{1}{3} & 0 \end{pmatrix}. \quad \blacksquare$$

In analogy to (12.3), we would like to define the pagerank vector as a *unique* eigenvector of $P$ with eigenvalue 1,

$$Pr = r.$$

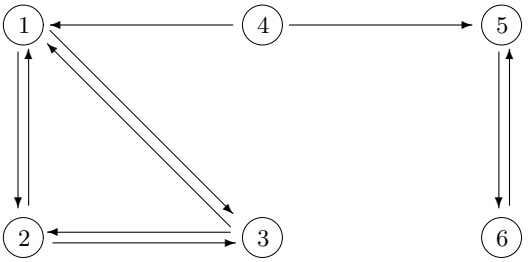
The eigenvector of the transition matrix corresponds to a stationary probability distribution for the Markov chain. The element in position $i$, $r_i$, is the probability that after a large number of steps, the random walker is at Web page $i$. However, the existence of a unique eigenvalue with eigenvalue 1 is still not guaranteed. To ensure uniqueness, the matrix must be *irreducible*; cf. [53].

**Definition 12.5.** *A square matrix $A$ is called* reducible *if there is a permutation matrix $P$ such that*

$$PAP^T = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix}, \tag{12.7}$$

*where $X$ and $Z$ are both square. Otherwise the matrix is called* irreducible.

**Example 12.6.** To illustrate the concept of reducibility, we give an example of a link graph that corresponds to a *reducible* matrix:



A random walker who has entered the left part of the link graph will never get out of it, and similarly will get stuck in the right part. The corresponding matrix is

$$P = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{12.8}$$

which is of the form (12.7). Actually, this matrix has two eigenvalues equal to 1 and one equal to $-1$; see Example 12.10. ∎

The directed graph corresponding to an irreducible matrix is *strongly connected*: given any two nodes $(N_i, N_j)$, in the graph, there exists a path leading from $N_i$ to $N_j$.

The uniqueness of the largest eigenvalue of an irreducible, positive matrix is guaranteed by the *Perron–Frobenius theorem*; we state it for the special case treated here. The inequality $A > 0$ is understood as all the elements of $A$ being strictly positive. By *dominant eigenvalue* we mean the largest eigenvalue in magnitude, which we denote $\lambda_1$.

**Theorem 12.7.** *Let $A$ be an irreducible column-stochastic matrix. The dominant eigenvalue $\lambda_1$ is equal to 1. There is a unique corresponding eigenvector $r$ satisfying $r > 0$, and $\|r\|_1 = 1$; this is the only eigenvector that is nonnegative. If $A > 0$, then $|\lambda_i| < 1$, $i = 2, 3, \ldots, n$.*

***Proof.*** Because $A$ is column stochastic, we have $e^T A = e^T$, which means that 1 is an eigenvalue of $A$. The rest of the statement can be proved using the Perron–Frobenius theory [70, Chapter 8]. □

Given the size of the Internet, we can be sure that the link matrix $P$ is reducible, which means that the pagerank eigenvector of $P$ is not well defined. To ensure irreducibility, i.e., to make it impossible for the random walker to get trapped in a subgraph, one adds, artificially, a link from every Web page to all the others. In matrix terms, this can be made by taking a convex combination of $P$ and a rank-1 matrix,

$$A = \alpha P + (1 - \alpha)\frac{1}{n} e e^T, \tag{12.9}$$

for some $\alpha$ satisfying $0 \leq \alpha \leq 1$. It is easy to see that the matrix $A$ is column-stochastic:

$$e^T A = \alpha e^T P + (1 - \alpha)\frac{1}{n} e^T e e^T = \alpha e^T + (1 - \alpha) e^T = e^T.$$

The random walk interpretation of the additional rank-1 term is that in each time step the surfer visiting a page will jump to a random page with probability $1-\alpha$ (sometimes referred to as *teleportation*).

We now see that the pagerank vector for the matrix $A$ is well defined.

**Proposition 12.8.** *The column-stochastic matrix $A$ defined in* (12.9) *is irreducible* (*since $A > 0$*) *and has the dominant eigenvalue $\lambda_1 = 1$. The corresponding eigenvector $r$ satisfies $r > 0$.*

For the convergence of the numerical eigenvalue algorithm, it is essential to know how the eigenvalues of $P$ are changed by the rank-1 modification (12.9).

**Theorem 12.9.** *Assume that the eigenvalues of the column-stochastic matrix $P$ are $\{1, \lambda_2, \lambda_3 \dots, \lambda_n\}$. Then the eigenvalues of $A = \alpha P + (1-\alpha)\frac{1}{n}ee^T$ are $\{1, \alpha\lambda_2, \alpha\lambda_3, \dots, \alpha\lambda_n\}$.*

***Proof.*** Define $\hat{e}$ to be $e$ normalized to Euclidean length 1, and let $U_1 \in \mathbb{R}^{n\times(n-1)}$ be such that $U = \begin{pmatrix}\hat{e} & U_1\end{pmatrix}$ is orthogonal. Then, since $\hat{e}^T P = \hat{e}^T$,

$$\begin{aligned} U^T P U &= \begin{pmatrix}\hat{e}^T P \\ U_1^T P\end{pmatrix}\begin{pmatrix}\hat{e} & U_1\end{pmatrix} = \begin{pmatrix}\hat{e}^T \\ U_1^T P\end{pmatrix}\begin{pmatrix}\hat{e} & U_1\end{pmatrix} \\ &= \begin{pmatrix}\hat{e}^T\hat{e} & \hat{e}^T U_1 \\ U_1^T P\hat{e} & U_1^T P^T U_1\end{pmatrix} = \begin{pmatrix}1 & 0 \\ w & T\end{pmatrix}, \end{aligned} \tag{12.10}$$

where $w = U_1^T P\hat{e}$ and $T = U_1^T P^T U_1$. Since we have made a similarity transformation, the matrix $T$ has the eigenvalues $\lambda_2, \lambda_3, \dots, \lambda_n$. We further have

$$U^T v = \begin{pmatrix}1/\sqrt{n}\, e^T v \\ U_1^T v\end{pmatrix} = \begin{pmatrix}1/\sqrt{n} \\ U_1^T v\end{pmatrix}.$$

Therefore,

$$\begin{aligned} U^T A U &= U^T(\alpha P + (1-\alpha)ve^T)U = \alpha\begin{pmatrix}1 & 0 \\ w & T\end{pmatrix} + (1-\alpha)\begin{pmatrix}1/\sqrt{n} \\ U_1^T v\end{pmatrix}\begin{pmatrix}\sqrt{n} & 0\end{pmatrix} \\ &= \alpha\begin{pmatrix}1 & 0 \\ w & T\end{pmatrix} + (1-\alpha)\begin{pmatrix}1 & 0 \\ \sqrt{n}\,U_1^T v & 0\end{pmatrix} =: \begin{pmatrix}1 & 0 \\ w_1 & \alpha T\end{pmatrix}. \end{aligned}$$

The statement now follows immediately. ∎

Theorem 12.9 implies that even if $P$ has a multiple eigenvalue equal to 1, which is actually the case for the Google matrix, the second largest eigenvalue in magnitude of $A$ is always equal to $\alpha$.

**Example 12.10.** We compute the eigenvalues and eigenvectors of the matrix $A = \alpha P + (1-\alpha)\frac{1}{n}ee^T$ with $P$ from (12.8) and $\alpha = 0.85$. The MATLAB code

```
LP=eig(P)';
e=ones(6,1);
A=0.85*P + 0.15/6*e*e';
[R,L]=eig(A)
```

gives the following result:

```
LP = -0.5    1.0   -0.5    1.0   -1.0    0

R = 0.447   -0.365   -0.354    0.000    0.817    0.101
    0.430   -0.365    0.354   -0.000   -0.408   -0.752
    0.430   -0.365    0.354    0.000   -0.408    0.651
    0.057   -0.000   -0.707    0.000    0.000   -0.000
    0.469    0.548   -0.000   -0.707    0.000    0.000
    0.456    0.548    0.354    0.707   -0.000   -0.000

diag(L) = 1.0 0.85 -0.0 -0.85 -0.425 -0.425
```

It is seen that the first eigenvector (which corresponds to the eigenvalue 1), is the only nonnegative one, as stated in Theorem 12.7. ∎

Instead of the modification (12.9) we can define

$$A = \alpha P + (1 - \alpha)ve^T,$$

where $v$ is a nonnegative vector with $\| v \|_1 = 1$ that can be chosen to make the search biased toward certain kinds of Web pages. Therefore, it is sometimes referred to as a *personalization vector* [74, 48]. The vector $v$ can also be used for avoiding manipulation by so-called link farms [57, 59].

## 12.3 The Power Method for Pagerank Computation

We want to solve the eigenvalue problem

$$Ar = r,$$

where $r$ is normalized $\| r \|_1 = 1$. In this section we denote the sought eigenvector by $t_1$. Dealing with stochastic matrices and vectors that are probability distributions, it is natural to use the 1-norm for vectors (Section 2.3). Due to the sparsity and the dimension of $A$ (of the order billions), it is out of the question to compute the eigenvector using any of the standard methods described in Chapter 15 for dense matrices, as those methods are based on applying orthogonal transformations to the matrix. The only viable method so far is the *power method*.

Assume that an initial approximation $r^{(0)}$ is given. The power method is given in the following algorithm.

**The power method for $Ar = \lambda r$**

**for** $k = 1, 2, \ldots$ until convergence

$$q^{(k)} = Ar^{(k-1)}$$
$$r^{(k)} = q^{(k)} / \| q^{(k)} \|_1$$

The purpose of normalizing the vector (making it have 1-norm equal to 1) is to avoid having the vector become either very large or very small and thus unrepresentable in the floating point system. We will see later that normalization is not necessary in the pagerank computation. In this context there is no need to compute an eigenvalue approximation, as the sought eigenvalue is known to be equal to one.

The convergence of the power method depends on the distribution of eigenvalues. To make the presentation simpler, we assume that $A$ is diagonalizable, i.e., there exists a nonsingular matrix $T$ of eigenvectors, $T^{-1}AT = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$. The eigenvalues $\lambda_i$ are ordered $1 = \lambda_1 > |\lambda_2| \geq \cdots \geq |\lambda_n|$. Expand the initial approximation $r^{(0)}$ in terms of the eigenvectors,

$$r^{(0)} = c_1 t_1 + c_2 t_2 + \cdots + c_n t_n,$$

where $c_1 \neq 0$ is assumed[30] and $r = t_1$ is the sought eigenvector. Then we have

$$\begin{aligned} A^k r^{(0)} &= c_1 A^k t_1 + c_2 A^k t_2 + \cdots + c_n A^k t_n \\ &= c_1 \lambda_1^k t_1 + c_2 \lambda_2^k t_2 + \cdots + c_n \lambda_n^k t_n = c_1 t_1 + \sum_{j=2}^{n} c_j \lambda_j^k t_j. \end{aligned}$$

Obviously, since for $j = 2, 3, \ldots$ we have $|\lambda_j| < 1$, the second term tends to zero and the power method converges to the eigenvector $r = t_1$. The rate of convergence is determined by $|\lambda_2|$. If this is close to 1, then the iteration is very slow. Fortunately this is not the case for the Google matrix; see Theorem 12.9 and below.

A stopping criterion for the power iteration can be formulated in terms of the residual vector for the eigenvalue problem. Let $\hat{\lambda}$ be the computed approximation of the eigenvalue and $\hat{r}$ the corresponding approximate eigenvector. Then it can be shown [94], [4, p. 229] that the optimal error matrix $E$, for which

$$(A + E)\hat{r} = \hat{\lambda}\hat{r},$$

exactly, satisfies

$$\| E \|_2 = \| s \|_2,$$

where $s = A\hat{r} - \hat{\lambda}\hat{r}$. This means that if the residual $\|s\|_2$ is small, then the computed approximate eigenvector $\hat{r}$ is the exact eigenvector of a matrix $A + E$ that is close

[30] This assumption can be expected to be satisfied in floating point arithmetic, if not at the first iteration, then after the second, due to round-off.

to $A$. Since in the pagerank computations we are dealing with a positive matrix, whose columns all add up to one, it is natural to use the 1-norm instead [55]. As the 1-norm and the Euclidean norm are equivalent (cf. (2.6)), this does not make much difference.

In the usual formulation of the power method the vector is normalized to avoid underflow or overflow. We now show that this is not necessary when the matrix is column stochastic.

**Proposition 12.11.** *Assume that the vector $z$ satisfies $\| z \|_1 = e^T z = 1$ and that the matrix $A$ is column stochastic. Then*

$$\| Az \|_1 = 1. \tag{12.11}$$

***Proof.*** Put $y = Az$. Then

$$\| y \|_1 = e^T y = e^T Az = e^T z = 1$$

since $A$ is column stochastic ($e^T A = e^T$). ∎

In view of the huge dimensions of the Google matrix, it is nontrivial to compute the matrix-vector product $y = Az$, where $A = \alpha P + (1-\alpha)\frac{1}{n} e e^T$. Recall that $P$ was constructed from the actual link matrix $Q$ as

$$P = Q + \frac{1}{n} e d^T,$$

where the row vector $d$ has an element 1 in all those positions that correspond to Web pages with no outlinks (see (12.5)). This means that to form $P$, we insert a large number of full vectors into $Q$, each of the same dimension as the total number of Web pages. Consequently, we cannot afford to store $P$ explicitly. Let us look at the multiplication $y = Az$ in more detail:

$$y = \alpha \left( Q + \frac{1}{n} e d^T \right) z + \frac{(1-\alpha)}{n} e (e^T z) = \alpha Q z + \beta \frac{1}{n} e, \tag{12.12}$$

where

$$\beta = \alpha d^T z + (1-\alpha) e^T z.$$

We do not need to compute $\beta$ from this equation. Instead we can use (12.11) in combination with (12.12):

$$1 = e^T (\alpha Q z) + \beta e^T \left( \frac{1}{n} e \right) = e^T (\alpha Q z) + \beta.$$

Thus, we have $\beta = 1 - \| \alpha Q z \|_1$. An extra bonus is that we do not use the vector $d$ at all, i.e., we need not know which pages lack outlinks.

The following MATLAB code implements the matrix vector multiplication:

```
yhat=alpha*Q*z;
beta=1-norm(yhat,1);
y=yhat+beta*v;
residual=norm(y-z,1);
```

Here $v = (1/n)\, e$, or a personalized teleportation vector; see p. 154. To save memory, we should even avoid using the extra vector `yhat` and replace it with `y`.

From Theorem 12.9 we know that the second eigenvalue of the Google matrix satisfies $\lambda_2 = \alpha$. A typical value of $\alpha$ is 0.85. Approximately $k = 57$ iterations are needed to make the factor $0.85^k$ equal to $10^{-4}$. This is reported [57] to be close to the number of iterations used by Google.

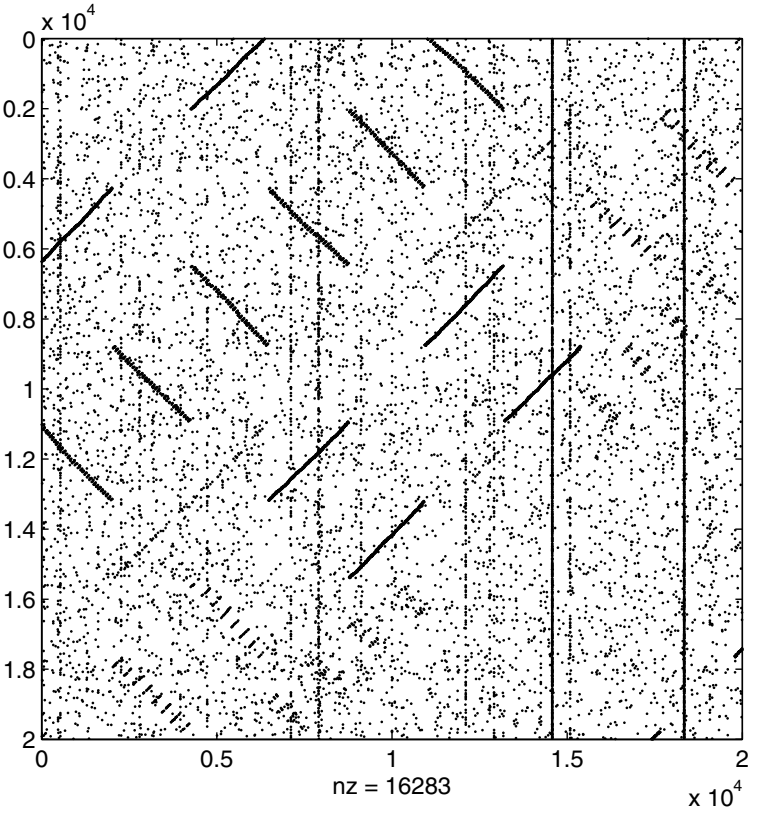



**Figure 12.1.** *A* $20000 \times 20000$ *submatrix of the stanford.edu matrix.*

**Example 12.12.** As an example we used the matrix $P$ obtained from the domain stanford.edu.[31] The number of pages is 281903, and the total number of links is 2312497. Part of the matrix is displayed in Figure 12.1. We computed the pagerank

[31] http://www.stanford.edu/~sdkamvar/research.html.

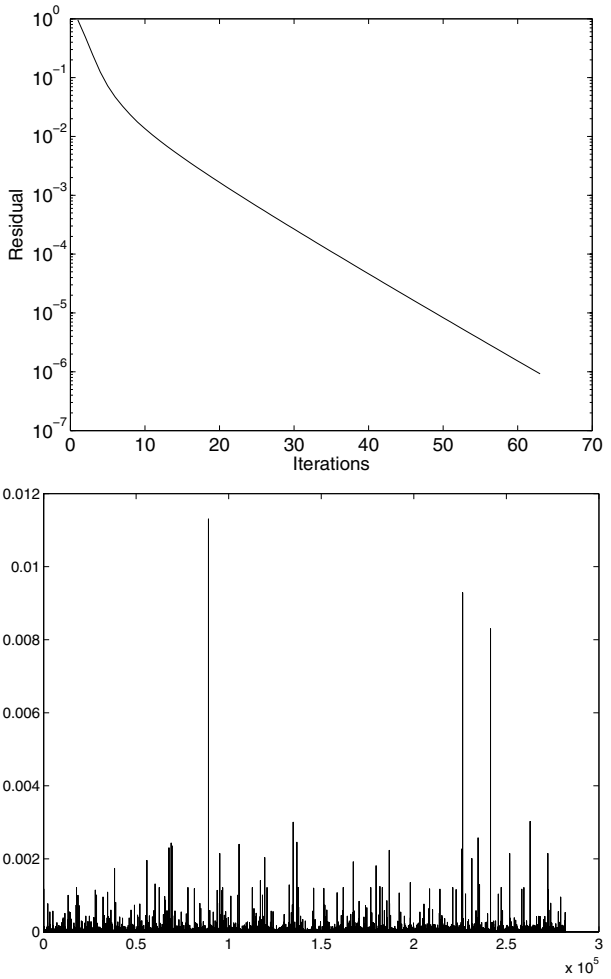



**Figure 12.2.** *The residual in the power iterations (top) and the pagerank vector (bottom) for the stanford.edu matrix.*

vector using the power method with $\alpha = 0.85$ and iterated 63 times until the 1-norm of the residual was smaller than $10^{-6}$. The residual and the final pagerank vector are illustrated in Figure 12.2. ∎

Because one pagerank calculation can take several days, several enhancements of the iteration procedure have been proposed. In [53] an adaptive method is described that checks the convergence of the components of the pagerank vector and avoids performing the power iteration for those components. Up to 30% speed-up has been reported. The block structure of the Web is used in [54], and speed-ups of a factor of 2 have been reported. An acceleration method based on Aitken extrapo-

lation is described in [55]. Aggregation methods are discussed in several papers by Langville and Meyer and in [51].

When computing the pagerank for a subset of the Internet, say, one particular domain, the matrix $P$ may be of a dimension for which one can use methods other than the power method, e.g., the Arnoldi method; see [40] and Section 15.8.3. It may even be sufficient to use the MATLAB function `eigs`, which computes a small number of eigenvalues and the corresponding eigenvectors of a sparse matrix using an Arnoldi method with restarts.

A variant of pagerank is proposed in [44]. Further properties of the pagerank matrix are given in [84].

## 12.4 HITS

Another method based on the link structure of the Web was introduced at the same time as pagerank [56]. It is called HITS (Hypertext Induced Topic Search) and is based on the concepts of *authorities* and *hubs*. An authority is a Web page with many inlinks, and a hub has many outlinks. The basic idea is that *good hubs point to good authorities and good authorities are pointed to by good hubs.* Each Web page is assigned both a hub score $y$ and an authority score $x$.

Let $L$ be the adjacency matrix of the directed Web graph. Then two equations are given that mathematically define the relation between the two scores, based on the basic idea:

$$x = L^T y, \qquad y = Lx. \tag{12.13}$$

The algorithm for computing the scores is the power method, which converges to the left and right singular vectors corresponding to the largest singular value of $L$. In the implementation of HITS, the adjacency matrix not of the whole Web but of all the pages relevant to the query is used.

There is now an extensive literature on pagerank, HITS, and other ranking methods. For overviews, see [7, 58, 59]. A combination of HITS and pagerank has been proposed in [65].

Obviously, the ideas underlying pagerank and HITS are not restricted to Web applications but can be applied to other network analyses. A variant of the HITS method was recently used in a study of Supreme Court precedent [36]. HITS is generalized in [17], which also treats synonym extraction. In [72], generank, which is based on the pagerank concept, is used for the analysis of microarray experiments.