A Note on Google's PageRank

According to Google, google-search on a given topic results in a listing of most relevant web pages related to the topic. Google ranks the importance of webpages according to an eigenvector of a weighted link matrix. The following offers an insight into how this is done and is a basic application of the eigenvalue problem from linear algebra. It is based on the *Bryan, Leise paper* and on `http://www.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture3/lecture3.html`.

A Tiny Web Example

- Core idea: in assigning a score to any given web page, the page's score (ranking) is derived from the links made *to* that page from other web pages.
- The links *to* a given page are called the *backlinks* for that page
- The web is represented as a directed graph $G = (V, E)$ with vertices being the web pages and edges the links. There is a directed edge from page i to page j if page i contains a hyperlink to page j.
- Denote the importance score of page k by x_k ($x_i > x_j$ means that page i is more important)
- As in the paper, the approach that *doesn't work* is to take x_k as the number of backlinks for page k, e.g. here $x_1 = 2, x_2 = 1, x_3 = 3, x_4 = 2$:

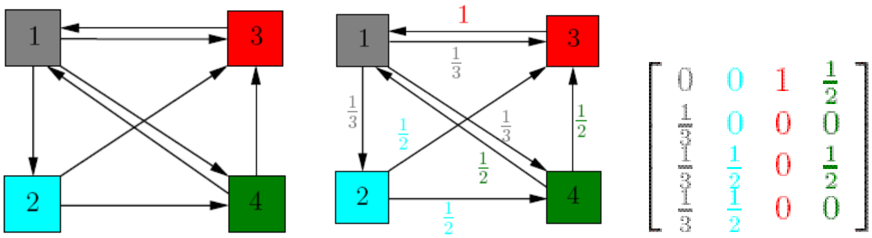


Figure 1: An example web graph, its weighted version, and its link matrix (transpose of the adjacency matrix of the weighted graph).

But we want a link to page k from an important page to boost page k's importance score more than a link from an unimportant page. (A page's importance is presumably higher when, say, the CNN's webpage links to it than when just Joe Blow's web page links to it.)

E.g., in the above graph, pages 1 and 4 have the same score, but one of page 1's backlinks is from the seemingly important page 3 (which seems important because everybody else links to it), while one of page 4's backlinks is from the relatively unimportant page 1. Thus, we'd be rating page 1's importance higher than page 4's.

- In an attempt to fix this, we can try to compute the score of page j as the *sum* of the scores of all pages linking to page j.

  For example, the score of page 1 would be determined by the relation $x_1 = x_3 + x_4$, because pages 3 and 4 are 1's backlilnks and their scores are x_3 and x_4.

- However, there's a bit of a problem with this: we don't want a single individual webpage to gain influence merely by casting multiple votes (just as in elections, we don't want a single individual to gain undue influence by casting multiple votes)

- So we make a correction: if page j contains n_j *out*links, one of which links to page k, then we boost page k's score by x_j/n_j rather than by x_j.

- Notice that in this scheme each web page gets a total of one vote, weighted by that web page's score, that is evenly divided up among all of its outgoing links.

  Let $L_k \subseteq \{1, 2, \ldots, n\}$ denote the set of pages with a link to page k, that is, L_k is the set of k's backlinks. For each k require:

$$x_k = \sum_{j \in L_k} \frac{x_j}{n_j} \tag{1}$$

  where n_j is the set of outgoing links from page j.

Assigning a Score to a Page, an Example

- **Linear algebra point of view:** For the web in the above figure, using the outlined scheme, we have:

$$\begin{aligned} x_1 &= x_3/1 + x_4/2 \\ x_2 &= x_1/3 \\ x_3 &= x_1/3 + x_2/2 + x_4/2 \\ x_4 &= x_1/3 + x_2/2 \end{aligned}$$

- These linear equations can be written as $\mathbf{Ax} = \mathbf{x}$:

$$\begin{bmatrix} 0 & 0 & 1 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

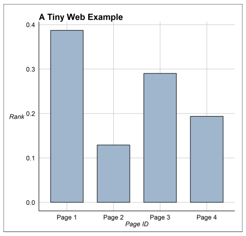
- *Thus, we have reduced the web ranking problem to the problem of finding an eigenvector for the link matrix* $\mathbf{A}$*:* $\mathbf{Ax} = \lambda\mathbf{x}$*. In particular, we are looking for the eigenvector corresponding to the eigenvalue* $\lambda = 1$. (Note that $\mathbf{A}$ is not the graph adjacency matrix. $\mathbf{A}^T$ is the graph adjacency matrix.)
- **Definition:** The *link matrix* $\mathbf{A}$ is such that $A_{ij} = 1/n_j$ if page j links to page i and 0 otherwise[1].
- **Definition:** A matrix $\mathbf{B}$ is called *column-stochastic* if the sum of entries of each column of $\mathbf{B}$ is one.
- If the web graph has no 'dangling' nodes (a dangling webpage is a page that has no outgoing links), e.g., see Fig. 2, then the j-th column of $\mathbf{A}$ contains n_j non-zero entries summing up to 1 (each is $1/n_j$). So $\mathbf{A}$'s columns all sum up to 1 and hence $\mathbf{A}$ is column-stochastic. This implies that A always has 1 as its eigenvalue as we will show now:

  Claim: Every column-stochastic matrix has 1 as an eigenvalue.

  Proof: Let $\mathbf{e}$ be a vector of all ones. Obviously $\mathbf{A}^T\mathbf{e} = \mathbf{e}$ holds, since the rows of $\mathbf{A}^T$ add up to one. Thus 1 is an eigenvalue of $\mathbf{A}^T$. Recalling that the eigenvlues of $\mathbf{A}^T$ and $\mathbf{A}$ are the same, proves the claim.
- In this small example the eigenvector corresponding to $\lambda = 1$ is easy to find 'by hand'. The following `MATLAB` code also finds the eigenvalus and eigenvectors of $\mathbf{A}$:

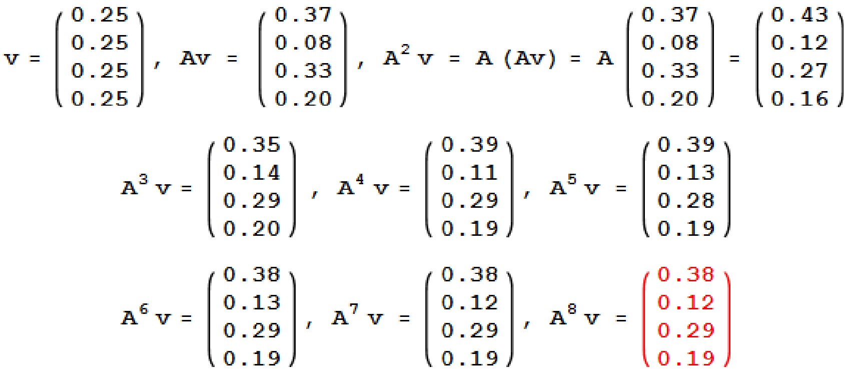
```
A = [   0     0     1    1/2;
       1/3    0     0     0 ;
       1/3   1/2    0    1/2;
       1/3   1/2    0     0  ];

[V D]  = eig(A);

x = V(:,1)/sum(V(:,1))   % scale 1'st eigenvector to sum to 1
```

We obtain $x = \frac{1}{31}[12\;4\;9\;6] \approx [0.38\;0.12\;0.29\;0.19]$. We call x the PageRank vector of our web graph. Note that its components sum up to 1. We can plot the rankings as well:



[1] The number of non-zero entries in the ith row of $\mathbf{A}$ is the *in-degree* of node i - i.e. how many other pages link to it. And the number of non-zero entries in the jth column is the *out-degree* of node j - i.e. how many other pages j links to.

- **Dynamical systems point of view:** Suppose that initially the importance is uniformly distributed among the 4 nodes, each getting 1/4. Denote by $\mathbf{v}$ the initial rank vector, having all entries equal to 1/4. Each incoming link increases the importance of a web page, so at step 1, we update the rank of each page by adding to the current value the importance of the incoming links. This is the same as multiplying the matrix $\mathbf{A}$ with $\mathbf{v}$. At step 1, the new importance vector is $\mathbf{v}_1 = \mathbf{A}\mathbf{v}$. We can iterate the process, thus at step 2, the updated importance vector is $\mathbf{v}_2 = \mathbf{A}(\mathbf{A}\mathbf{v}) = \mathbf{A}^2\mathbf{v}$. Numeric computations give:



$$v = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}, \quad Av = \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix}, \quad A^2 v = A\,(Av) = A \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix} = \begin{pmatrix} 0.43 \\ 0.12 \\ 0.27 \\ 0.16 \end{pmatrix}$$

$$A^3 v = \begin{pmatrix} 0.35 \\ 0.14 \\ 0.29 \\ 0.20 \end{pmatrix}, \quad A^4 v = \begin{pmatrix} 0.39 \\ 0.11 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^5 v = \begin{pmatrix} 0.39 \\ 0.13 \\ 0.28 \\ 0.19 \end{pmatrix}$$

$$A^6 v = \begin{pmatrix} 0.38 \\ 0.13 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^7 v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^8 v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}$$

  We notice that the sequences of iterates $\mathbf{v}, \mathbf{A}\mathbf{v}, ..., \mathbf{A}^k\mathbf{v}$ tend to the equilibrium value $\mathbf{v}^*$, which in this case is $\mathbf{v}^* \approx \mathbf{A}^8\mathbf{v}$. We observe that $\mathbf{v}^*$ is equal to the eigenvector $\mathbf{x}$ computed above.

- **Probabilistic point of view:** Since the importance of a web page is measured by its popularity (how many incoming links it has), we can view the importance of page i as the probability that a random surfer on the Internet that opens a browser to any page and starts following hyperlinks, visits the page i. We can interpret the weights we assigned to the edges of the graph in a probabilistic way: A random surfer that is currently viewing web page 2, has 1/2 probability to go to page 3, and 1/2 probability to go to page 4. We can model the process as a random walk on graphs. Each page has equal probability 1/4 to be chosen as a starting point. So, the initial probability distribution is given by the column vector $\mathbf{x} = [\frac{1}{4}\ \frac{1}{4}\ \frac{1}{4}\ \frac{1}{4}]^T$. The probability that page i will be visited after one step is equal to $\mathbf{A}\mathbf{x}$, and so on. The probability that page i will be visited after k steps is equal to $\mathbf{A}^k\mathbf{x}$. The sequence $\mathbf{A}\mathbf{x}, \mathbf{A}^2\mathbf{x}, \mathbf{A}^3\mathbf{x}, ..., \mathbf{A}^k\mathbf{x}, ...$ converges in this case to a unique probabilistic vector $\mathbf{v}^*$. In this context $\mathbf{v}^*$ is called the stationary distribution. Moreover, the ith entry in $\mathbf{v}^*$ is simply the probability that at each moment a random surfer visits page i. The computations are identical to the ones we did in the dynamical systems interpretation, only the meaning we attribute to each step being slightly different.

Dangling Nodes

- If the web graph has dangling nodes, they can be easily repaired. A column in the link matrix that represents a dangling node has all zero entries, since a dangling node has no outgoing edges. We can replace each such column with a column vector with all entries $1/n$, where n is the total number of graph nodes, see Fig. 2. In this way, the importance of the node is equally redistributed among the other nodes of the graph, instead of being lost. Since it is equally redistributed, the relative ranking of the other nodes did not change. For example, we can fix the dangling node E in Fig. 2 by replacing the column corresponding to E with a column vector with all entries 1/5.

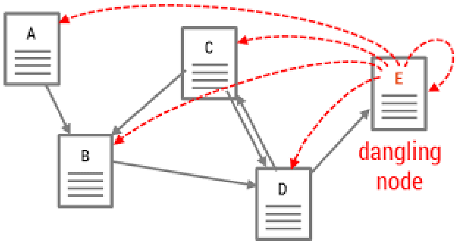
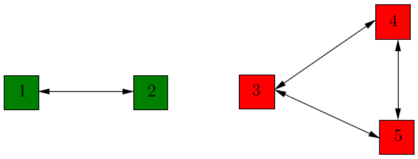


Figure 2: Example of a dangling node.

Case of Disconnectd SubWebs

- There maybe more than one eigenvector that solves $\mathbf{Ax} = \mathbf{x}$ if the web graph is not connected. In this case the solution is not unique, and hence the dimensionality of the eigenspace $V_1(\mathbf{A})$ corresponding to $\lambda = 1$ is larger that one. For example, there are two eigenvectors that solve $\mathbf{Ax} = \mathbf{x}$ in Fig. 3.



$$A = \left[\begin{array}{cc|ccc} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{array}\right] \qquad v = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad u = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Figure 3: Example of a disconnected web graph, its link matrix with two different eigenvectors corresponding to eigenvalue 1.

- To solve this problem Page and Brin replaced the matrix $\mathbf{A}$ with the matrix:
$$\mathbf{M} = (1-m)\mathbf{A} + m\mathbf{S},$$
where $\mathbf{S}$ is $n \times n$ matrix with all entries 1/n. This means we add 'weak' links form every webpage to every other. The value of m originally used by Google is reportedly 0.15. For any $m \in [0,1]$ the matrix $\mathbf{M}$ is column-stochastic and we can show that $V_1(\mathbf{M})$ is always one-dimensional if $m \in (0,1]$ if there is no dangling nodes.

- Observe that:
$$\mathbf{Mx} = (1-m)\mathbf{Ax} + m\mathbf{Sx} = (1-m)\mathbf{Ax} + m\mathbf{s},$$
where $\mathbf{s}$ is a column vector with all entries 1/n, since $\mathbf{Sx} = \mathbf{s}$ for any vector $\mathbf{x} = [x_1, \ldots, x_n]^T$ such that $\sum_{i=1}^n x_i = 1$.

- Hence, the equation $\mathbf{x} = \mathbf{Mx}$ can also be cast as $\mathbf{x} = (1-m)\mathbf{Ax} + m\mathbf{s}$.

PageRank Foundations

- By the following theorem, matrix $\mathbf{M}$ has a unique eigenvector, corresponding to the eigenvalue 1, which is called the PageRank vector for the web graph with link matrix $\mathbf{A}$.

  Perron-Frobenius Theorem: If $\mathbf{M}$ is a positive, column stochastic matrix, then

  1. 1 is an eigenvalue of multiplicity one,
  2. 1 is the largest eigenvalue: all the other eigenvalues have absolute value smaller than 1, and
  3. the eigenvectors corresponding to the eigenvalue 1 have either only positive entries or only negative entries. In particular, for the eigenvalue 1 there exists a unique eigenvector with the sum of its entries equal to 1.

- From the mathematical point of view, once we have M, computing the eigenvector corresponding to the eigenvalue 1 is, at least in theory, a straightforward task. But when the matrix $\mathbf{M}$ has size 30 billion (as it does for the real Web graph), even mathematical software such as Matlab or Mathematica are clearly overwhelmed. An alternative way of computing the eigenvector is given by the Power Method. The theorem below guarantees that the method works for positive, column stochastic matrices.

  Power Method Convergence Theorem: Let $\mathbf{M}$ be a positive, column stochastic $n \times n$ matrix. Denote by $\mathbf{v}^*$ its eigenvector corresponding to the eigenvalue 1. Let $\mathbf{v}$ be the column vector with all entries equal to 1/n. Then the sequence $\mathbf{v}, \mathbf{Mv}, ..., \mathbf{M}^k\mathbf{v}$ converges to vector $\mathbf{v}^*$.

- In the random surfer model, we argued that the iteration process corresponds to the way importance distributes over the net following the link structure. Computationally speaking, it is much more easier, starting from the vector with $\mathbf{v}$, to multiply $\mathbf{v}, \mathbf{Mv}, ..., \mathbf{M}^k\mathbf{v}$ until convergence then it is to compute the eigenvectors of $\mathbf{M}$. In fact, in this case, one needs only compute the first couple of iterates in order to get a good approximation of the PageRank vector.

- For a random matrix, the power method is in general known to be slow to converge. What makes it work fast in this case however is the fact that the web graph is sparse. This means that a node i has a small number of outgoing links (a couple of hundred at best, which is extremely small corresponding to the 30 billion nodes it could theoretically link to). Hence the link matrix $\mathbf{A}$ has a lot of entries equal to 0.