Practice 5
Web search: PageRank algorithm

Contents

1 Introduction

The algorithms concerning web search, such as Google's PageRank, are excellent applications of basic tools of matrix algebra.

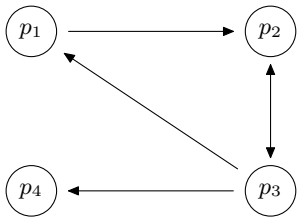
PageRank is the calculation method used by the founders of Google (Sergey Brin and Lawrence Page) to rank web pages according to their importance. The purpose of the method is to obtain a vector, the PageRank vector, which provides the "importance" of the pages (that is, it assigns, to each web page, a "degree of importance", giving a ranking of all sites according to the "importance"). This vector is calculated from the structure of the web connections. We will describe here only the basics of the PageRank method, although it has been (and it is nowadays) subject to continuous improvements.

2 Modeling the World Wide Web

Imagine that you are looking for the best book on Linear Algebra. You probably would try a web search engine such as Google. This lists pages ranked by importance. The ranking is defined in such a way that a page is important if other important pages link to it: "a page can have a high PageRank if there are many pages that point to it, or if there are enough important

pages that point to it." But, how can we tell whether a page is important without first deciding on the important pages? That's the question.

We will model the World Wide Web (that is a collection of pages connected by links) as a directed graph whose vertices correspond to the pages and whose arcs represent the links among them. For a better explanation, instead of the whole World Wide Web we shall consider, as a simplified model, a small network of web pages whose links are represented by means of the following directed graph:



(The double arrow must be interpreted as two arrows).

The key idea is that pages should be highly ranked if either they are cited often by other pages or they are cited by important enough pages. That is, we raise the importance of a page p_i (denoted by $I(p_i)$) each time it is linked from page p_j. The increment depends on the importance of the linking page p_j divided by how many links a_j are on that page:

$$I(p_i) = \sum_{\text{pages } p_j \text{ linking to } P_i} \frac{I(p_j)}{a_j}$$

In this example notice that

- The page p_1 has only one link (to p_2) and, therefore, $a_1 = 1$.
- The page p_2 has also only one link and, therefore, $a_2 = 1$.
- The page p_3 has 3 links and, therefore, $a_3 = 3$.
- The page p_4 has no link and, therefore, $a_4 = 0$.

Hence:

$I(p_1) = \frac{1}{3}I(p_3)$
$I(p_2) = I(p_1) + \frac{1}{3}I(p_3)$
$I(p_3) = I(p_2)$
$I(p_4) = \frac{1}{3}I(p_3)$

Observe that this is equivalent to the equality

$$\underbrace{\begin{bmatrix} I(p_1) \\ I(p_2) \\ I(p_3) \\ I(p_4) \end{bmatrix}}_{\vec{I}} = \underbrace{\begin{bmatrix} 0 & 0 & 1/3 & 0 \\ 1 & 0 & 1/3 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/3 & 0 \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} I(p_1) \\ I(p_2) \\ I(p_3) \\ I(p_4) \end{bmatrix}}_{\vec{I}} \tag{1}$$

So, the "vector of importances" $\vec{I}$ (that is what we want to compute) satisfies the equality $G\vec{I} = \vec{I}$, where $G = [g_{ij}]$ is the 4×4 matrix such that

$$g_{ij} = \frac{1}{\text{number of links of } p_j}$$

if p_j links to p_i, and $g_{ij} = 0$ if p_j does not link to p_i. The video given by the following link will help you to understand this:

http://www.youtube.com/watch?v=ZstQKxUW7oM

Equality (1) means that we want a *stationary vector* $\vec{I}$ of the matrix G (that is, a vector that remains invariant if we multiply it by G). We want, in addition, that $\vec{I}$ be a *probability vector*, which means that all the components of $\vec{I}$ are nonnegative and their sum is 1. A vector $\vec{I}$ satisfying all these properties would *rank* the pages assigning, to each one, a number between 0 and 1 that "means" its "degree of importance". But, in addition, there should be a **unique** vector with these conditions (two different vectors would mean two different ways for ranking the pages, and we do not want this). Summarizing, we want the existence of a vector $\vec{I}$ satisfying the following properties:

(1) $\vec{I}$ is a *stationary vector* for G, that is, $G\vec{I} = \vec{I}$,

(2) $\vec{I}$ is a *probability vector*, that is, its components are all nonnegative and their sum is equal to 1.

(3) $\vec{I}$ is the **unique** vector satisfying (1) and (2).

Does the above matrix G satisfies these requirements? In this case the answer is NO (the unique stationary vector is $\vec{0}$). However, as we will see, we can modify slightly the matrix G in a clever way to force it to satisfy the 3 conditions.

3 Stochastic matrices (revisited)

Recall that an $n \times n$ matrix is called *stochastic* if all its entries are nonnegative real numbers and the sum of the entries in each column is 1. The stochastic matrices are specially interesting for us because of the following property (that we saw in Practice 3)

Theorem 1. Any stochastic matrix has, at least, a stationary probability vector.

It follows from the definition of G that the sum of the entries in its nonzero columns is 1. However the last one is a zero column and, therefore, G is not stochastic; the problem is the *existence of zero columns*, and this occurs because there is a page (p_4) without any link (it is a *sink* page). The simplest way to avoid this problem is to imagine that, when somebody is "surfing through the web" and gets to a page like this, then he/she goes to a next page entirely at random. This means that one can think that a sink page has a link to any page. So, we can redefine the matrix G replacing the zero columns by vectors $(1/n, 1/n, \ldots, 1/n)$, where n is the number of pages (4, in this case):

$$G = \begin{bmatrix} 0 & 0 & 1/3 & 1/4 \\ 1 & 0 & 1/3 & 1/4 \\ 0 & 1 & 0 & 1/4 \\ 0 & 0 & 1/3 & 1/4 \end{bmatrix} \tag{2}$$

Due to the above theorem, this matrix (and any matrix defined in this way coming from any web of pages) satisfies Conditions (1) and (2). In this example, we are lucky and, as you can check, Condition (3) is also satisfied but, in general, this is not true. So we need to modify again the definition of G to assure that the stationary probability vector is always unique. The following result (that we saw with more generality in Practice 3 and is a consequence of the *Perron-Fröbenius Theorem*) gives the property that we need[1]:

Theorem 2. If G is a stochastic matrix such that all its entries are strictly positive then G has a **unique** stationary probability vector.

Our matrix G (and, in practice, any matrix G obtained in the same way) has zero entries and, therefore, we cannot apply to it this theorem. However there is a clever way of modifying G in such a way that all its entries be strictly positive:

Imagine a person that is "surfing" through the web following the links: each time he/she visits a page p_i, he/she goes to another page by using one of the links of p_i; if p_i is a sink page, he/she goes to a random page of the web by writing a random address in the browser address bar. Perhaps, from time to time (and independently whether the visited page p_i is a sink or not) he/she might want to put, in the browser address bar, the address of a random web page instead of "following" the links of the actual web page. That is, he/she might "want to follow", instead of the matrix G, the following one (which we shall call **matrix of randomness**):

$$E := \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{bmatrix};$$

[1]The statement and the proof of the Perron-Fröbenius Theorem use, as key ingredients, the concepts of eigenvalue and eigenvector, that will be introduced in Lesson 6.

this means that all web pages have the same probability of being chosen.

Now, we shall assume the following: each time that a "surfer" visits a site, he/she has two possibilities: either "to follow" the matrix G (that is, to follow one of the links of the actual site), or "to follow" the **matrix of randomness** E (that is, to choose a random web page).

Fix a real number $\alpha \in]0,1[$ denoting the probability of surfing "following" the matrix G. Then, the probability of surfing following the matrix of randomness E is $1-\alpha$. α measures, in some sense, the "degree of freedom" that we allow the surfer to "follow" each one of the two matrices. (Notice that, since we are interested in giving prominence to the matrix G, α should be near 1). The new assumption we have done is equivalent to consider, instead of the matrix G, this new one:

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

It is easy to check that this matrix is stochastic and that, moreover, all its entries are strictly positive. Then we can apply to it the theorems we have seen deducing that there is a vector $\vec{I}$ satisfying the above conditions (1), (2) and (3). This is the PageRank vector we are looking for.

A way for computing this vector $\vec{I}$ is to solve the following system of linear equations[2]:

$$(\mathbf{G} - I_{4\times 4})\vec{x} = \vec{0}. \tag{3}$$

The role of the parameter α is an important one. Notice that if $\alpha = 1$ then $\mathbf{G} = G$. This means that we are working with the original hyperlink structure of the web. However, if $\alpha = 0$ then $\mathbf{G} = E$; in other words, we are considering that any page has a link to any other page, and we have lost the original hyperlink structure of the web. Clearly, we would like to take α close to 1 so that we hyperlink structure of the web is weighted heavily into the computation.

However, there is another consideration. In practice, the actual matrix $\mathbf{G}$ corresponding with the WWW is an extremely huge matrix. As a consequence of this, to solve a system like (3) using the usual methods is not a good business[3]. Instead of this, in practice, the vector $\vec{I}$ is computed using an iterative method known as *power method*. *Essentially*, it consists of the computation, by iteration, of the limit vector of the following Markov chain[4] :

$$\vec{x}_0, \;\; \vec{x}_1 = \mathbf{G}\vec{x}_0, \;\; \vec{x}_2 = \mathbf{G}\vec{x}_1, \ldots$$

where $\vec{x}_0$ is *any* initial probability vector.

We omit here more explanations about this method except the fact that, when the parameter α is close to 1, the convergence of this method is very slow. As a compromise between these two competing interests, Sergey Brin and Larry Page, the creators of PageRank, chose a value of alpha around $\alpha = 0.85$.

[2]Observe that $\mathbf{G}\vec{x} = \vec{x} \;\Leftrightarrow\; \mathbf{G}\vec{x} = I_{4\times 4}\vec{x} \;\Leftrightarrow\; \mathbf{G}\vec{x} - I_{4\times 4}\vec{x} = \vec{0} \;\Leftrightarrow\; (\mathbf{G} - I_{4\times 4})\vec{x} = \vec{0}$

[3]The accumulation of rounding errors is one reason. Another is that, since the number web pages is so huge, the matrix of randomness E is extremely close to be the zero matrix!

[4]In practice, it is used a modified version of the *power method* that allows us to compute explicitly only the matrix G instead of $\mathbf{G}$, avoiding the use of E in the computations.

4 Worked example with Scilab

To illustrate the method, we are going now to compute the PageRank vector corresponding to the above example with the help of Scilab. First, we introduce the matrix G given in (2), that comes from the graph that describe the links but replacing the zeroes of the zero columns by $1/n$, where n is the number of pages ($n = 4$ in this case):

```
-->G=[0 0 1/3 1/4; 1 0 1/3 1/4; 0 1 0 1/4; 0 0 1/3 1/4]
 G  =

    0.    0.    0.3333333    0.25
    1.    0.    0.3333333    0.25
    0.    1.    0.           0.25
    0.    0.    0.3333333    0.25
```

Next we define the "matrix of randomness" E:

```
-->E=1/4*ones(4,4)
 E  =

    0.25    0.25    0.25    0.25
    0.25    0.25    0.25    0.25
    0.25    0.25    0.25    0.25
    0.25    0.25    0.25    0.25
```

Now we define the *Google matrix* $\mathbf{G}$ taking $\alpha = 0.85$:

```
-->G=0.85*G+(1-0.85)*E
 G  =

    0.0375    0.0375    0.3208333    0.25
    0.8875    0.0375    0.3208333    0.25
    0.0375    0.8875    0.0375       0.25
    0.0375    0.0375    0.3208333    0.25
```

Then we solve the system (3), that is, we compute the kernel of the matrix $\mathbf{G} - I_{4\times 4}$:

```
-->x=kernel(G-eye(4,4))
 x  =

    0.3254602
    0.6021013
    0.6523997
    0.3254602
```

The unique probability vector that is a solution of the system (3) can be computed easily dividing the above obtained generator of the kernel by the sum of its components:

```
-->x/sum(x)
 ans  =

    0.1708075
    0.3159938
    0.3423913
    0.1708075
```

This is the PageRank vector. This means that the pages are ranked, in non-increasing degree of importance, as follows:

$$Page\ 3$$

$$Page\ 2$$

$$Page\ 1$$

$$Page\ 4$$

In this case, since we have only 4 pages, it is easy to order the pages "by hand". However, in cases with a big number of pages, it may be useful to write the following Scilab command:

```
-->[w,k]=gsort(x/sum(x))
 k  =

    3.
    2.
    1.
    4.
 w  =

    0.3423913
    0.3159938
    0.1708075
    0.1708075
```

The returned vector w writes, in non-increasing order, the PageRank of the pages. The vector k gives directly the ordered list of the pages.