

Two-Hop Walks Indicate PageRank Order

Ying Tang^{a,1,*}

^a*College of Cyber Security, Chengdu University of Technology, Chengdu, 610059, People's Republic of China.*

Abstract

This paper shows that pairwise PageRank orders emerge from two-hop walks. The main tool used here refers to a specially designed sign-mirror function and a parameter curve, whose low-order derivative information implies pairwise PageRank orders with high probability. We study the pairwise correct rate by placing the Google matrix \mathbf{G} in a probabilistic framework, where \mathbf{G} may be equipped with different random ensembles for model-generated or real-world networks with sparse, small-world, scale-free features, the proof of which is mixed by mathematical and numerical evidence. We believe that the underlying spectral distribution of aforementioned networks is responsible for the high pairwise correct rate. Moreover, the perspective of this paper naturally leads to an $O(1)$ algorithm for any single pairwise PageRank comparison if assuming both $\mathbf{A} = \mathbf{G} - \mathbf{I}_n$, where \mathbf{I}_n denotes the identity matrix of order n , and \mathbf{A}^2 are ready on hand (e.g., constructed offline in an incremental manner), based on which it is easy to extract the top k list in $O(kn)$, thus making it possible for PageRank algorithm to deal with super large-scale datasets in real time.

Keywords: Spectral Ranking, PageRank, Two-Hop.

*Corresponding author

Email address: mathtygo@gmail.com (Ying Tang)

¹Postal Address: Room 5602, NanYi Building, Chengdu University of Technology, ErXianQiao East 3rd Road No.1, Chengdu, Sichuan, China. Post code: 610059; Tel: +86 18602852948; Fax: +86 28 84078903.

1. Introduction

The PageRank algorithm and related variants have attracted much attention in many applications of practical interests [1, 2, 3], especially known for their key role in the Google’s search engine. These principal eigenvector (the one corresponding to the largest eigenvalue) based algorithms share the same spirit and were rediscovered again and again by different communities from 1950’s. PageRank-type algorithms have appeared in the literatures on bibliometrics [4, 5, 6], sociometry [7, 8], econometrics [9], or web link analysis [10], etc. Two excellent historical reviews on this technique can be found in [11, 12].

Regardless of various motivations, this family of algorithms stand on the similar observations: an entity (person, page, node, etc) is important if it is pointed by other important entities, thus the resulting importance score should be computed in a recursive manner. More precisely, given a n -dimensional matrix \mathbf{G} with its element g_{ij} encoding some form of endorsement sent from the j^{th} entity to the i^{th} entity (both \mathbf{G} and the transpose of \mathbf{G} are alternately used in literatures, but which introduces no essential difference. Here, the former is adopted for convenience), then the importance score vector \mathbf{r} is defined as the solution of the linear system:

$$\mathbf{G}\mathbf{r} = \mathbf{r}. \quad (1)$$

However, some constraints are required for \mathbf{G} such that there exists an unique and nonnegative solution in (1). In the PageRank algorithm, \mathbf{G} is constructed by [10, 13]

$$\mathbf{G} = \alpha(\widehat{\mathbf{G}} + \mathbf{u}\mathbf{d}^T) + (1 - \alpha)\mathbf{v}\mathbf{1}^T, \quad (2)$$

where $\widehat{\mathbf{G}}$ is the column-normalized adjacent matrix of the web graph, i.e., the $(i, j)^{\text{th}}$ element of $\widehat{\mathbf{G}}$ is one divided by the outdegree of the j^{th} page if there is an link from the j^{th} page to the i^{th} page (zero otherwise), $\mathbf{1}$ is the all-ones vector, \mathbf{d} is the indicator vector of *dangling nodes* (those having no outgoing edges), \mathbf{u} and \mathbf{v} are nonnegative and have unit l_1 norm (known as *the dangling-node* and *personalization vectors*, respectively. By default $\mathbf{u} = \mathbf{v} = \mathbf{1}/n$), and $\alpha \in [0, 1)$ is the *damping factor* (had better not be too close to 1. Usually

$\alpha = 0.85$ by default) for avoiding the “sink effect” caused by the modules with in- but no out-links [14]. Then, it is easy to verify that \mathbf{G} constructed as above is a markov matrix with each column summing to one, and has an unrepeated largest eigenvalue valued 1 corresponding to the left eigenvector $\mathbf{1}$ (the modulus of the second largest eigenvalue of \mathbf{G} is upper-bounded by α [15]). Due to the Perron–Frobenius theorem [9], this means that the (right) positive principal eigenvector of \mathbf{G} actually is the unique PageRank vector in (1). Note that such a solution is only defined up to a positive scale, but introducing no harm in the ranking context.

1.1. Related Work

The humongous size of the World Wide Web and its fast growing rate make the evaluation of the PageRank vector one of the most demanding computational tasks ever, which causes the main obstacle of applying the PageRank algorithm to real-world applications since current principal eigenvector solvers for matrices of order over hundreds of thousands are still prohibitive in both time and memory. Much effort for accelerating the PageRank algorithm has been carried out from different view, such as Monte Carlo method [16], random walk [17], power method or general linear system [18, 19], graph theory [20, 21, 22], Schrödinger equation [23], and quantum networks [24, 25]. More recent related advances on this topic can be found in [26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36]. However, it seems that one important fact is totally ignored when achieving speed-up: the exact value of the PageRank vector is generally immaterial and what is really interesting is the ranking list, especially the top k list in general. To the best of our knowledge, no research has been carried out on this way. The problem addressed in this paper will follow this direction for extracting the pairwise PageRank order in $O(1)$ using a very different insight if assuming both $\mathbf{G} - \mathbf{I}_n$ and $(\mathbf{G} - \mathbf{I}_n)^2$ are ready in memory, based on which it is straightforward to obtain the top k list in $O(kn)$. Our proposed algorithm avoids any effort of computing the exact value of the principle eigenvector of \mathbf{G} .

1.2. Outline of Our Algorithm

In this paper, we always assume that \mathbf{G} is a nonnegative real matrix with the spectral radius 1, and 1 is an unique eigenvalue. We will use $\mathbf{r} = [r_1, \dots, r_n]^T$ to denote arbitrary nonnegative principal eigenvector of \mathbf{G} although the PageRank vector may be defined up to a positive scale. Let $\mathbf{A} = \mathbf{G} - \mathbf{I}_n = (a_{ij})$, where \mathbf{I}_n is the unit matrix of order n . The main tool used in this paper is a specially designed curve $\mathbf{F}(\mathbf{A}, \mathbf{w}, t) = [F_1(t), \dots, F_n(t)] \in \mathbb{R}^n$, where \mathbf{A}, \mathbf{w} and t are three parameters. Here, we drop the dependency of $F_k(t)$ on \mathbf{A} and \mathbf{w} to make notations less cluttered. Throughout the paper, we will indicate vectors and matrices using bold faced letters.

We expect $\mathbf{F}(\mathbf{A}, \mathbf{w}, t)$ to have the following properties: (a). For any positive \mathbf{w} , the curve converges to the positive principal eigenvector of \mathbf{A} (thus converges to \mathbf{r}) as $t \rightarrow \infty$. Let $\Delta_{ij}(t) = F_i(t) - F_j(t)$, thus the task of comparing the PageRank score between the i^{th} and j^{th} nodes is reduced to determining the sign of $\Delta_{ij}(\infty) = F_i(\infty) - F_j(\infty) = r_i - r_j$; (b). Denote by $\mathbf{F}^{(m)}(\mathbf{A}, \mathbf{w}, t)$ the m^{th} -order derivative of \mathbf{F} w.r.t. t , which had better be a simple function of \mathbf{w} and \mathbf{A} such that evaluating it at $t = 0$ causes relatively low computational cost; (c). Around the neighbourhood of $t = 0$, the shape of $(F_i(t), F_j(t)), i \neq j$, on the $\overline{x_i x_j}$ plane (spanned by the i^{th} and j^{th} axes in \mathbb{R}^n) can be flexibly controlled by \mathbf{w} and $\mathbf{F}^{(k)}(\mathbf{A}, \mathbf{w}, t), k = 1, \dots, m$.

With a carefully chosen \mathbf{w} , it is possible to find a scale function $\phi_{ij}(\mathbf{A}, \mathbf{w}, \mathbf{F}^{(1)}(\mathbf{A}, \mathbf{w}, 0), \dots, \mathbf{F}^{(m)}(\mathbf{A}, \mathbf{w}, 0))$, simplified as ϕ_{ij} , such that the probability $\pi_{ij} = \Pr(\phi_{ij} \Delta_{ij}(\infty) > 0)$ is sufficiently close to one. We call ϕ_{ij} the sign-mirror function for $\Delta_{ij}(\infty)$ since it reflects the sign of $\Delta_{ij}(\infty)$ in a probabilistic sense shown as above, although ϕ_{ij} itself only contains the local information of $\mathbf{F}(\mathbf{A}, \mathbf{w}, t)$ around $t = 0$. Furthermore, to avoid unnecessary computational cost, we also expect that small m can do this job.

Section 2 provides a curve equipped with the above properties with $m \geq 2$. There we also construct the corresponding sign-mirror function ϕ_{ij} and formulate π_{ij} as a function of θ , an angle variable dependent on the eigenvalue distribution of \mathbf{A} . In the same section, we discuss some extensions of the al-

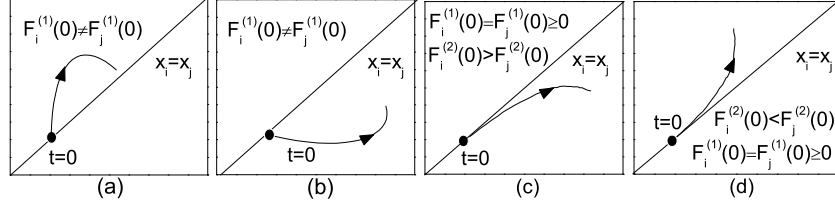


Figure 1: Four possible trajectories of $(F_i(t), F_j(t))$ on the $\overline{x_i x_j}$ plane with the x_i and x_j axes along horizontal and direction, respectively. In all the cases, \mathbf{w} 's are picked out such that $(F_i(0), F_j(0))$'s locate on the line $x_i = x_j$. The last two are our desired cases, where both trajectories are tangent to $x_i = x_j$ with nonequal acceleration at $t = 0$. Roughly speaking, this is due to the fact that in such cases we have more confidence to predict the sign of $F_i(\infty) - F_j(\infty)$ only based on \mathbf{A}, \mathbf{w} , and $\mathbf{F}^{(k)}(\mathbf{A}, \mathbf{w}, 0), k = 1, \dots, m$.

gorithms. Section 3 checks the numerical properties of θ , then verifies that π_{ij} keeps small for variant types of model-generated or real-world graphs (sparse, scale-free, small-world, etc). This means that with a high probability the proposed algorithm succeeds to extract the true pairwise PageRank order for those common types of graphs mentioned as above. Then, it is relatively straightforward to develop a top k list extraction algorithm based on partial (not total) pairwise orders, which will be discussed in section 4.

Nevertheless, it will be helpful to roughly imagine how such a curve possibly looks. Fig. 1 plots four possible trajectories of $(F_i(t), F_j(t))$ on the $\overline{x_i x_j}$ plane. Intuitively, $(F_i(t), F_j(t))$'s plotted in Fig. 1(a) and (b) are unpredictable in the sense that intuitively we have no confidence to predict whether they will cross the line $x_i = x_j$ at some $t > 0$ or not. On the contrary, $(F_i(t), F_j(t))$'s shown in Fig. 1(c) or (d) seem more revealing due to the following facts: with higher probability, those two curves will not cross the line $x_i = x_j$ again for $t > 0$ since both have been tangent to the line $x_i = x_j$ at $t = 0$, and will locally move away from the line $x_i = x_j$ soon since they have unequal acceleration along axes at $t = 0$. In fact, the imagined Fig. 1(c) and (d) do motivate us to construct an eligible sign-mirror function from a geometric view.

Finally, we point out that the algorithm of this paper is not only valid for the Google matrix defined in (2), which can even be applied to the non-markov

matrix \mathbf{G} as long as \mathbf{G} meets the two conditions presented at the beginning of this subsection.

2. Model

Let $i = \sqrt{-1}$ be the imaginary unit and $\mathbf{diag}[\mathbf{A}_1, \dots, \mathbf{A}_s]$ be a block diagonal matrix, where $\mathbf{A}_k, k = 1, \dots, s$, is a square matrix at the i^{th} diagonal block. Unless specially mentioned, in this paper $\mathbf{A} = \mathbf{G} - \mathbf{I}_n$ and \mathbf{G} is defined at the beginning of subsection 1.2. From a practical view, we also assume that \mathbf{G} (thus \mathbf{A}) is diagonalizable since any matrix can be perturbed into a diagonalizable one with perturbation arbitrary small. Thus, \mathbf{A} is real and diagonalizable, and all the eigenvalues of \mathbf{A} except the unrepeated zero eigenvalue have negative real parts.

2.1. Designing Curve

Lemma 1. [37] For any real and diagonalizable matrix \mathbf{A} of order n , there is an invertible matrix \mathbf{P} such that

$$\mathbf{A} = \mathbf{P} \cdot \mathbf{diag}[\underbrace{\lambda_1, \dots, \lambda_r}_r, \underbrace{\mathbf{A}_{r+1}, \dots, \mathbf{A}_{r+s}}_s] \cdot \mathbf{P}^{-1}, \quad r+2s = n, \quad 0 \leq r \leq n, \quad (3)$$

where

$$\mathbf{P} = [\underbrace{\mathbf{p}_1, \dots, \mathbf{p}_r}_{r \text{ real eigenvectors}}, \underbrace{\mathbf{p}_{R,r+1}, \mathbf{p}_{I,r+1}, \dots, \mathbf{p}_{R,r+s}, \mathbf{p}_{I,r+s}}_{s \text{ pairs of complex eigenvectors}}],$$

$$\mathbf{A}_{r+k} = \begin{pmatrix} \lambda_{R,r+k} & \lambda_{I,r+k} \\ -\lambda_{I,r+k} & \lambda_{R,r+k} \end{pmatrix}, k = 1, \dots, s.$$

In the above equation, $\lambda_k, k = 1, \dots, r$, are r real eigenvalues of \mathbf{A} sorted in descending order, corresponding to the r real eigenvectors \mathbf{p}_k , and $\lambda_{R,r+k} \pm i\lambda_{I,r+k}, k = 1, \dots, s$, are s pairs of complex eigenvalues of \mathbf{A} (sorted descendingly w.r.t. the real parts) corresponding to the s pairs of complex eigenvectors $\mathbf{p}_{R,r+k} \pm i\mathbf{p}_{I,r+k}$, respectively.

In this paper, there exists $\lambda_1 = 0$, and all the other λ_k 's ($k = 2, \dots, r$) as well as $\lambda_{R,k}$'s ($k = r+1, \dots, r+s$) are negative. Moreover, we will use

\mathbf{p}_k to denote the k^{th} column of \mathbf{P} in *lemma 1* for convenience, i.e., $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_r, \mathbf{p}_{R,r+1}, \mathbf{p}_{I,r+1}, \dots, \mathbf{p}_{R,r+s}, \mathbf{p}_{I,r+s}] = [\mathbf{p}_1, \dots, \mathbf{p}_n]$. Since $\mathbf{p}_1, \dots, \mathbf{p}_n$, are linearly independent, any \mathbf{w} takes the form as

$$\mathbf{w} = \sum_{k=1}^n w_k \mathbf{p}_k. \quad (4)$$

Let $\mathbf{V} = [\mathbf{v}_1^T, \dots, \mathbf{v}_n^T]^T = \mathbf{P}^{-1}$, and define for $k = 1, \dots, s$,

$$\mathbf{B}_{r+k} = \mathbf{p}_{r+2k-1} \mathbf{v}_{r+2k-1}^T + \mathbf{p}_{r+2k} \mathbf{v}_{r+2k}^T, \quad \mathbf{C}_{r+k} = \mathbf{p}_{r+2k-1} \mathbf{v}_{r+2k}^T - \mathbf{p}_{r+2k} \mathbf{v}_{r+2k-1}^T.$$

Then it is ready to construct the following curve with the desired properties given in subsection 1.2:

$$\mathbf{F}(\mathbf{A}, \mathbf{w}, t) = \left(\sum_{k=1}^r e^{\lambda_k t} \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s e^{\lambda_{R,r+k} t} [\cos(\lambda_{I,r+k} t) \mathbf{B}_{r+k} + \sin(\lambda_{I,r+k} t) \mathbf{C}_{r+k}] \right) \mathbf{w}, \quad (5)$$

where $t \geq 0$ is the time parameter and \mathbf{w} is the n -dimensional “shape adjusting” vector. Although \mathbf{p}_k and \mathbf{v}_k appear in (5), it is not necessary to compute them throughout our algorithm, which will be clear in the sequel.

Lemma 2. There exist $\mathbf{F}(\mathbf{A}, \mathbf{w}, 0) = \mathbf{w}$ and $\mathbf{F}(\mathbf{A}, \mathbf{w}, \infty) = w_1 \mathbf{p}_1$, where w_1 is the projection of \mathbf{w} on \mathbf{p}_1 .

Proof. Noting $\mathbf{F}(\mathbf{A}, \mathbf{w}, 0) = (\sum_{k=1}^n \mathbf{p}_k \mathbf{v}_k^T) \mathbf{w}$ and $\mathbf{P} \mathbf{V} = \mathbf{I}_n$, thus the first equality holds. Since $\lambda_1 = 0$, $\lambda_k < 0$ for $k = 2, \dots, r$, and $\lambda_{R,r+k}$ for $k = 1, \dots, s$, there exists $\mathbf{F}(\mathbf{A}, \mathbf{w}, \infty) = \mathbf{p}_1 \mathbf{v}_1^T \mathbf{w}$. Due to $\mathbf{V} \mathbf{P} = \mathbf{I}_n$, thus $\mathbf{v}_k^T \mathbf{p}_k = 1$ and $\mathbf{v}_k^T \mathbf{p}_h = 0$ for $\forall k \neq h$, which yields

$$\mathbf{F}(\mathbf{A}, \mathbf{w}, \infty) = \mathbf{p}_1 \mathbf{v}_1^T \sum_{k=1}^n w_k \mathbf{p}_k = w_1 \mathbf{p}_1.$$

thus proving the second equality.

Clearly, $w_1 \neq 0$ with probability 1, thus let us assume $w_1 \neq 0$. In the sequel, we will also restrict \mathbf{w} to be nonnegative, from which it is easy to see that $w_1 \mathbf{p}_1$ is always nonnegative, regardless of \mathbf{p}_1 being the nonpositive or nonnegative principal eigenvector of \mathbf{G} . Based on the above analysis and lemma 2, we can write $\mathbf{F}(\mathbf{A}, \mathbf{w}, \infty) = \mathbf{r}$, which verifies the property (a) presented in subsection

1.2. Thus, the task of comparing the PageRank score for the pair of $(i, j)^{\text{th}}$ pages is equivalent to determining the sign of $\Delta_{ij}(\infty) = F_i(\infty) - F_j(\infty)$.

The next lemma shows that both the first- and second-order derivatives of $\mathbf{F}(\mathbf{A}, \mathbf{w}, t)$ have a neat relation w.r.t. \mathbf{A} and \mathbf{w} at $t = 0$, which coincides with the highly desired property (b) given in subsection 1.2.

Lemma 3. There exist $\mathbf{F}^{(1)}(\mathbf{A}, \mathbf{w}, 0) = \mathbf{A}\mathbf{w}$ and $\mathbf{F}^{(2)}(\mathbf{A}, \mathbf{w}, 0) = \mathbf{A}^2\mathbf{w}$.

Proof. From (3), we have

$$\begin{aligned}\mathbf{A} &= \sum_{k=1}^r \lambda_k \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s [(\lambda_{\text{R}, r+k} \mathbf{p}_{r+2k-1} - \lambda_{\text{I}, r+k} \mathbf{p}_{r+2k}) \mathbf{v}_{r+2k-1}^T \\ &\quad + (\lambda_{\text{I}, r+k} \mathbf{p}_{r+2k-1} + \lambda_{\text{R}, r+k} \mathbf{p}_{r+2k}) \mathbf{v}_{r+2k}^T] \\ &= \sum_{k=1}^r \lambda_k \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s (\lambda_{\text{R}, r+k} \mathbf{B}_{r+k} + \lambda_{\text{I}, r+k} \mathbf{C}_{r+k}).\end{aligned}\quad (6)$$

Similarly, from the equality $\mathbf{A}^2 = \mathbf{P} \cdot \text{diag}[\lambda_1^2, \dots, \lambda_r^2, \mathbf{A}_{r+1}^2, \dots, \mathbf{A}_{r+s}^2] \cdot \mathbf{P}^{-1}$, a simple computation shows that

$$\mathbf{A}^2 = \sum_{k=1}^r \lambda_k^2 \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s [(\lambda_{\text{R}, r+k}^2 - \lambda_{\text{I}, r+k}^2) \mathbf{B}_{r+k} + 2\lambda_{\text{R}, r+k} \lambda_{\text{I}, r+k} \mathbf{C}_{r+k}]. \quad (7)$$

Based on the definition of $\mathbf{F}(\mathbf{A}, \mathbf{w}, t)$ as in (5), a direct computation yields

$$\mathbf{F}^{(1)}(\mathbf{A}, \mathbf{w}, 0) = \frac{d\mathbf{F}(\mathbf{A}, \mathbf{w}, t)}{dt} \Big|_{t=0} = \left(\sum_{k=1}^r \lambda_k \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s (\lambda_{\text{R}, r+k} \mathbf{B}_{r+k} + \lambda_{\text{I}, r+k} \mathbf{C}_{r+k}) \right) \mathbf{w} \stackrel{(6)}{=} \mathbf{A}\mathbf{w},$$

$$\begin{aligned}\mathbf{F}^{(2)}(\mathbf{A}, \mathbf{w}, 0) &= \frac{d^2\mathbf{F}(\mathbf{A}, \mathbf{w}, t)}{dt^2} \Big|_{t=0} \\ &= \left(\sum_{k=1}^r \lambda_k^2 \mathbf{p}_k \mathbf{v}_k^T + \sum_{k=1}^s (\lambda_{\text{R}, r+k}^2 \mathbf{B}_{r+k} + 2\lambda_{\text{R}, r+k} \lambda_{\text{I}, r+k} \mathbf{C}_{r+k} - \lambda_{\text{I}, r+k}^2 \mathbf{B}_{r+k}) \right) \mathbf{w} \stackrel{(7)}{=} \mathbf{A}^2\mathbf{w}.\end{aligned}$$

thus proving the lemma.

2.2. Designing the Sign-Mirror Function

Let $\mathbf{F}_k^{(m)}(\mathbf{A}, \mathbf{w}, 0)$ and $(\mathbf{A}^m \mathbf{w})_k$, $m = 1, 2$, be the k^{th} element of $\mathbf{F}^{(m)}(\mathbf{A}, \mathbf{w}, 0)$ and $\mathbf{A}^m \mathbf{w}$, respectively. In this subsection, we will focus on the key part of our eigenvector-computation-free algorithm: constructing the sign-mirror function

ϕ_{ij} for $\Delta_{ij}(\infty)$ (recall the notations defined in subsection 1.2). Obviously, the bigger π_{ij} is, with more confidence $\Delta_{ij}(\infty)$ and ϕ_{ij} share the same sign, In such a manner, we say that the sign of $\Delta_{ij}(\infty)$, which indicates the PageRank score order for the pair of the $(i, j)^{\text{th}}$ pages, is mirrored by the sign of ϕ_{ij} . As mentioned before, Fig. 1 suggests an intuition for constructing the sign-mirror function as follows: Let $\phi_{ij} = \mathbf{F}_i^{(2)}(\mathbf{A}, \mathbf{w}, 0) - \mathbf{F}_j^{(2)}(\mathbf{A}, \mathbf{w}, 0)$, under the constraints $\mathbf{F}_i^{(1)}(\mathbf{A}, \mathbf{w}, 0) = \mathbf{F}_j^{(1)}(\mathbf{A}, \mathbf{w}, 0)$, $\mathbf{w} \geq 0$ and $w_i = w_j$. From *lemma 3*, the above equations can be rewritten into

$$\phi_{ij} = (\mathbf{A}^2 \mathbf{w})_i - (\mathbf{A}^2 \mathbf{w})_j, \quad \text{with } w_i = w_j, \mathbf{w} \geq 0, (\mathbf{A} \mathbf{w})_i = (\mathbf{A} \mathbf{w})_j, \quad (8)$$

which possibly is the simplest form for ϕ_{ij} to adapt in practice. Although other more sophisticated candidates may be considered, ϕ_{ij} constructed as above has worked well enough for our goal.

Note that there exit many choices for \mathbf{w} meeting the constraints in (8). For reducing computational cost, in this paper we suggest to restrict \mathbf{w} in the type of vectors only composed of three different values.

Let $\mathcal{J} \subsetneq \{1, \dots, n\}$ be an index subset containing i and j such that $\sum_{k \in \mathcal{J}} (a_{ik} - a_{jk}) \neq 0$. Clearly, \mathcal{J} does not exist if and only if $a_{ii} + a_{ij} = a_{ji} + a_{jj}$ and $a_{ik} = a_{jk}, \forall k \neq i, j$, which corresponds to an event with zero probability if regarding \mathbf{A} as a random matrix. In what follows we assume the existence of \mathcal{J} .

Let $h \notin \mathcal{J}$ be any index such that $a_{ih} - a_{jh}$ has the opposite sign to that of $\sum_{k \in \mathcal{J}} (a_{ik} - a_{jk})$ (the exceptional case where h does not exist will be discussed later). Then, let $\zeta_{ij} = \sum_{k \notin \{h\} \cup \mathcal{J}} (a_{jk} - a_{ik})$ and define \mathbf{w} by

$$w_k = \frac{-q(a_{ih} - a_{jh}) - \zeta_{ij}}{\sum_{k \in \mathcal{J}} (a_{ik} - a_{jk})} \triangleq z, \forall k \in \mathcal{J}; \quad w_h = \varepsilon + \max(0, \frac{\zeta_{ij}}{a_{ih} - a_{jh}}) \triangleq q; \quad \text{otherwise } w_k = 1. \quad (9)$$

where ε is an adjustable positive constant ($\varepsilon = 10^{-5}$ is used in our simulation).

It is easy to verify that \mathbf{w} constructed as above meets all the constraints in (8).

Let b_{ij} be the $(i, j)^{\text{th}}$ element of $\mathbf{B} = \mathbf{A}^2 = (b_{ij})$. A simple simplification shows that with \mathbf{w} as in (9) ϕ_{ij} can be rewritten into:

$$\phi_{ij} = z \sum_{k \in \mathcal{J}} (b_{ik} - b_{jk}) + q(b_{ih} - b_{jh}) + \sum_{k \notin \mathcal{J} \cup \{h\}} (b_{ik} - b_{jk}).$$

Input: $\mathbf{A} = \mathbf{G} - \mathbf{I}_n$ and its square \mathbf{B} , where \mathbf{G} is constructed as (2).

- 1 Randomly choose \mathcal{J} satisfying $i, j \in \mathcal{J} \subseteq \{1, \dots, n\}, \sum_{k \in \mathcal{J}} (a_{ik} - a_{jk}) \neq 0$.
- 2 Randomly choose $h \notin \mathcal{J}$ satisfying $(a_{ih} - a_{jh}) \sum_{k \in \mathcal{J}} (a_{ik} - a_{jk}) < 0$.
- 3 Compute $\phi_{ij} = z \sum_{k \in \mathcal{J}} (b_{ik} - b_{jk}) + q(b_{ih} - b_{jh}) + \sum_{k \notin \mathcal{J} \cup \{h\}} (b_{ik} - b_{jk})$,
with z and q defined in (9).

Output: $\phi_{ij} > 0 \Rightarrow r_i > r_j$ or $\phi_{ij} < 0 \Rightarrow r_i < r_j$, where r_i and r_j are the estimated PageRank score of nodes i and j .

Algorithm 1: Comparing the PageRank score between nodes i and j .

Specially, in the case of $\mathcal{J} = \{i, j\}$, i.e., $a_{ii} + a_{ij} \neq a_{ji} + a_{jj}$, which corresponds to an almost sure event in practice, let us denote by $\text{sum}_k(\mathbf{A})$ and $\text{sum}_k(\mathbf{B})$ the sum of the k^{th} row of \mathbf{A} and \mathbf{B} , respectively. In this case, ϕ_{ij} takes a more computation-friendly form:

$$\phi_{ij} = \text{sum}_i(\mathbf{B}) - \text{sum}_j(\mathbf{B}) + (z-1)(b_{ii} + b_{ij} - b_{ji} - b_{jj}) + (q-1)(b_{ih} - b_{jh}), \quad (10)$$

where q and $z = \frac{\text{sum}_j(\mathbf{A}) - \text{sum}_i(\mathbf{A}) + (1-q)(a_{ih} - a_{jh})}{a_{ii} + a_{ij} - a_{ji} - a_{jj}} + 1$ are computed from (9) with $\mathcal{J} = \{i, j\}$. Now, we conclude our pairwise PageRank ranking algorithm as follows:

$$\phi_{ij} > 0 \Rightarrow r_i > r_j \quad \text{or} \quad \phi_{ij} < 0 \Rightarrow r_i < r_j. \quad (11)$$

The whole algorithm flow is depicted in **Algorithm 1**. As for the exceptional case that no index h exists, i.e., $a_{ik} - a_{jk}, k \neq i, j$, are all positive (or negative), which is an almost null event in practice, it is intuitive to claim $r_i > r_j$ (or $r_i < r_j$) due to the PageRank principle.

Finally, we provide a complexity analysis for single run of (11). If \mathbf{A} and $\mathbf{B} = \mathbf{A}^2$ (constructed offline) are ready in memory, the time cost comes from two parts: time for finding the index h plus a dozen of simple algebraic computation involved in (9) and (10). Given $\sum_{k \in \mathcal{J}} (a_{ik} - a_{jk})$, let p be the probability that $a_{ih} - a_{jh}$ has the same sign as that of $\sum_{k \in \mathcal{J}} (a_{ik} - a_{jk})$ for a randomly chosen $h \notin \mathcal{J}$. Then, the mean number of sampling h equals to $\lim_{n \rightarrow \infty} \sum_{k=1}^n k(1-p)p^{k-1} = \frac{1}{(1-p)^2}$, just a small constant. Thus, the time complexity for single run of (11) is $O(1)$. Moreover, it is easy to see that both \mathbf{A} and \mathbf{B} can be constructed

incrementally. Actually, the whole algorithm (11) is almost ready to work in an incremental fashion with slight modifications, which is omitted here.

2.3. Evaluating π_{ij}

Here, we study the probability $\pi_{ij} = \Pr(\phi_{ij}\Delta_{ij}(\infty) > 0)$ (recall the notations defined in subsection 1.2) given ϕ_{ij} constructed in (8), which determines the correct rate of our algorithm (11). Let $\mathbf{p}_k = [p_{1k}, \dots, p_{nk}]^T, k = 1, \dots, n$, and $\tau_k^{ij} = p_{ik} - p_{jk}$, thus $\Delta_{ij}(\infty) = w_1\tau_1^{ij}$ from the second equality in *lemma 2*. Based on (4), the constraint $w_i = w_j$ in (8) means $\sum_{k=1}^n w_k\tau_k^{ij} = 0$, i.e.,

$$\Delta_{ij}(\infty) = w_1\tau_1^{ij} = -\sum_{k=2}^n w_k\tau_k^{ij}. \quad (12)$$

Based on (4) and (6), the constraint $(\mathbf{A}\mathbf{w})_i = (\mathbf{A}\mathbf{w})_j$ in (8) indicates

$$\begin{aligned} 0 &= \sum_{k=2}^r \lambda_k w_k \tau_k^{ij} + \sum_{k=1}^s [\lambda_{R,r+k} (w_{r+2k-1} \tau_{r+2k-1}^{ij} + w_{r+2k} \tau_{r+2k}^{ij}) \\ &\quad + \lambda_{I,r+k} (w_{r+2k} \tau_{r+2k-1}^{ij} - w_{r+2k-1} \tau_{r+2k}^{ij})]. \end{aligned} \quad (13)$$

where we use the fact $\lambda_1 = 0$, $\mathbf{v}_k^T \mathbf{p}_k = 1$ and $\mathbf{v}_k^T \mathbf{p}_h = 0$ for $\forall k \neq h$. Similarly, using (4) and (7), $\phi_{ij} = (\mathbf{A}^2 \mathbf{w})_i - (\mathbf{A}^2 \mathbf{w})_j$ can be rewritten into

$$\begin{aligned} \phi_{ij} &= \sum_{k=2}^r \lambda_k^2 w_k \tau_k^{ij} + \sum_{k=1}^s [(\lambda_{R,r+k}^2 - \lambda_{I,r+k}^2) (w_{r+2k-1} \tau_{r+2k-1}^{ij} + w_{r+2k} \tau_{r+2k}^{ij}) \\ &\quad + 2\lambda_{R,r+k} \lambda_{I,r+k} (w_{r+2k} \tau_{r+2k-1}^{ij} - w_{r+2k-1} \tau_{r+2k}^{ij})]. \end{aligned} \quad (14)$$

Next, we want to eliminate one *redundant item* from both (12) and (14) with the help of (13). This *redundant item* corresponds to $(w_{r+1} \tau_{r+1}^{ij} + w_{r+2} \tau_{r+2}^{ij})$ if there exists $\lambda_{R,r+1}$ (i.e, there are at least one pair of complex eigenvalues, called *case 1*), or to $w_2 \tau_2^{ij}$ if there exists λ_2 (i.e, there are two or more real eigenvalues, called *case 2*). A direct computation gives the following theorem:

Theorem 4. Given any pair of (i, j) , we have $\phi_{ij}\Delta_{ij}(\infty) = (\hat{\boldsymbol{\lambda}}_1^T \boldsymbol{\beta}^{ij})(\hat{\boldsymbol{\lambda}}_2^T \boldsymbol{\beta}^{ij})$.

In case 1, there exists

$$\begin{aligned}
\beta^{ij} &= [\underbrace{w_2 \tau_2^{ij}, \dots, w_r \tau_r^{ij}}_{r-1}, \underbrace{\gamma_{r+2}^{ij}, \dots, \gamma_{r+2s}^{ij}}_{2s-1}]^T \in \mathbb{R}^{n-2}, \\
\bar{\mathbf{X}}_1 &= [\underbrace{\frac{\lambda_2}{\lambda_{R,r+1}} - 1, \dots, \frac{\lambda_r}{\lambda_{R,r+1}} - 1}_{r-1}, \underbrace{\frac{\lambda_{I,r+1}}{\lambda_{R,r+1}}, \frac{\lambda_{R,r+2}}{\lambda_{R,r+1}} - 1, \frac{\lambda_{I,r+2}}{\lambda_{R,r+1}}, \dots, \frac{\lambda_{R,r+s}}{\lambda_{R,r+1}} - 1, \frac{\lambda_{I,r+s}}{\lambda_{R,r+1}}}_{2s-1}]^T \in \mathbb{R}^{n-2}, \\
\bar{\mathbf{X}}_2 &= [\underbrace{d_2, \dots, d_r}_{r-1}, \underbrace{e_{r+1} - c\lambda_{I,r+1}, f_{r+2} - c\lambda_{R,r+2}, e_{r+2} - c\lambda_{I,r+2}, \dots, f_{r+s} - c\lambda_{R,r+s}, e_{r+s} - c\lambda_{I,r+s}}_{2s-1}]^T \in \mathbb{R}^{n-2}, \quad (15)
\end{aligned}$$

where $\gamma_{r+2k-1}^{ij} = w_{r+2k-1} \tau_{r+2k-1}^{ij} + w_{r+2k} \tau_{r+2k}^{ij}$, $\gamma_{r+2k}^{ij} = w_{r+2k} \tau_{r+2k-1}^{ij} - w_{r+2k-1} \tau_{r+2k}^{ij}$, $k = 1, \dots, s$, $c = (\lambda_{R,r+1}^2 - \lambda_{I,r+1}^2) / \lambda_{R,r+1}$, $d_k = \lambda_k(\lambda_k - c)$ for $k = 2, \dots, r$, $e_{r+k} = 2\lambda_{R,r+k} \lambda_{I,r+k}$ for $k = 1, \dots, s$, and $f_{r+k} = \lambda_{R,r+k}^2 - \lambda_{I,r+k}^2$ for $k = 1, \dots, s$.

In case 2, there exists

$$\begin{aligned}
\beta^{ij} &= [\underbrace{w_3 \tau_3^{ij}, \dots, w_r \tau_r^{ij}}_{r-2}, \underbrace{\gamma_{r+1}^{ij}, \dots, \gamma_{r+2s}^{ij}}_{2s}]^T \in \mathbb{R}^{n-2}, \\
\bar{\mathbf{X}}_1 &= [\underbrace{\frac{\lambda_3}{\lambda_2} - 1, \dots, \frac{\lambda_r}{\lambda_2} - 1}_{r-2}, \underbrace{\frac{\lambda_{R,r+1}}{\lambda_2} - 1, \frac{\lambda_{I,r+1}}{\lambda_2}, \frac{\lambda_{R,r+2}}{\lambda_2} - 1, \dots, \frac{\lambda_{R,r+s}}{\lambda_2} - 1, \frac{\lambda_{I,r+s}}{\lambda_2}}_{2s}]^T \in \mathbb{R}^{n-2}, \\
\bar{\mathbf{X}}_2 &= [\underbrace{d_3, \dots, d_r}_{r-2}, \underbrace{f_{r+1} - c\lambda_{R,r+1}, e_{r+1} - c\lambda_{I,r+1}, f_{r+2} - c\lambda_{R,r+2}, \dots, f_{r+s} - c\lambda_{R,r+s}, e_{r+s} - c\lambda_{I,r+s}}_{2s}]^T \in \mathbb{R}^{n-2},
\end{aligned}$$

Here, all variables are same to those in (15) except $c = \lambda_2$.

It is worthy noting that $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$ are two $(n-2)$ -dimensional random vectors only dependent on the eigenvalue distribution of $\mathbf{A} = \mathbf{G} - \mathbf{I}_n$, and β^{ij} is a $(n-2)$ -dimensional random vector w.r.t. the eigenvector distribution of \mathbf{A} and the projections of \mathbf{w} along eigenvectors. From now on, will treat the Google matrix \mathbf{G} as a random one that encodes the topological structure of a model-generated or real-world networks following different ensembles, e.g., scale-free [38], or small-world [39], etc.

Denote by θ the angle between $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$. The above theorem provides a geometric interpretation for π_{ij} . Imagining the bounded subspace in \mathbb{R}^{n-2} where β_{ij} lives, as depicted in Fig. 2, *theorem 4* shows that the event $\phi_{ij} \Delta_{ij}(\infty) \leq 0$ corresponds to two dark spherical wedges enclosed by the two $(n-2)$ -dimensional hyperplanes V_1 and V_2 whose normal vectors are $\hat{\lambda}_1$ and $\hat{\lambda}_2$, respectively. Hence,

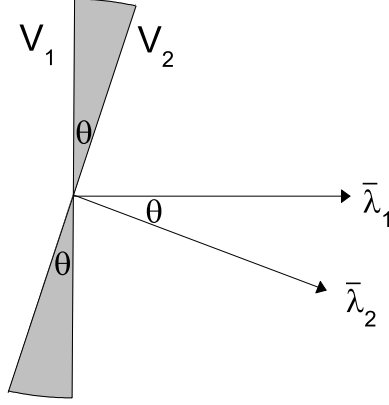


Figure 2: Two dark spherical wedges correspond to the event $\phi_{ij}\Delta_{ij}(\infty) \leq 0$, meaning that $\pi_{ij} = 1 - \frac{\theta}{180^\circ}$.

in principle we can write $\pi_{ij} = 1 - \text{Vol}(\text{dark})/\text{Vol}(\text{all})$, where $\text{Vol}(\text{dark})$ and $\text{Vol}(\text{all})$ denote the weighted volumes of two dark wedges and the total subspace, respectively. Here, the volume is weighted by the probability density function of β_{ij} , denoted by ρ . In general, it is impossible to obtain the analytical form of ρ for the purpose of evaluating π_{ij} , but it is interesting to note that when $\theta \rightarrow 0$, there approximately exists $\pi_{ij} \rightarrow 1$, regardless of the exact form of ρ and the direction of $\hat{\lambda}_k, k = 1, 2$. Note that in this claim we use an intuitive assumption that the support of ρ is not extremely concentrated around any low-dimensional hyperplane, which seems true from a practical view and will be discussed more later.

The above analysis also explains why we try eliminating an *redundant item* from $\Delta_{ij}(\infty)$ and ϕ_{ij} . The reason is that the resulting θ in such a manner would be close to a small angle as $n \rightarrow \infty$ (e.g., $n > 1000$, in what follows n is always assumed to be large enough unless specially stated) for a variant types of common networks, which will be detailedly verified in the next section.

Nevertheless, let us first look at some special cases to reveal the hidden motivation. To this end, let us consider the following types of undirected graphs as examples: 1). $\overline{\mathbf{G}} \leftarrow \text{abs}(\text{rand}(n))$ or $\text{abs}(\text{randn}(n))$, followed by $\overline{\mathbf{G}} \leftarrow \overline{\mathbf{G}} + \overline{\mathbf{G}}'$ or $\overline{\mathbf{G}} * \overline{\mathbf{G}}'$, where “ $\text{randn}(\cdot)$ ” and “ $\text{rand}(\cdot)$ ” are the functions, respectively for

generating matrices whose elements follow the standard normal $\mathcal{N}(0,1)$ and uniform distribution in $[0,1]$, and “abs(·)” denote the absolute value function in Matlab; 2). $\overline{\mathbf{G}}$ is the adjacent matrix for an Erdős-Reényi (ER) graph [40]. We normalize each column of $\overline{\mathbf{G}}$ to get $\hat{\mathbf{G}}$, then obtain \mathbf{G} following (2) with default parameters.

In such a way, although \mathbf{G} is generally asymmetric, $\|\mathbf{G} - \mathbf{G}'\|$ is very small with high probability since the sums of each column of $\overline{\mathbf{G}}$ are equal with high probability (thus $\hat{\mathbf{G}}$ is almost a constant scale of $\overline{\mathbf{G}}$), meaning that \mathbf{G} is “asymptotically symmetric”, i.e., all the eigenvalues of \mathbf{G} are real with high probability. Surprisingly, in all of our experiments based on the above graphs generated by different n (We also varied the sparse density for ER graphs), no complex eigenvalue appears at all! (however, it has been proven in [41] that for matrices with the elements following the standard normal distribution, the number of real eigenvalues scales with \sqrt{n} , instead of n). In a word, all the eigenvalues in these example typically are real, corresponding to the *case 2* in *theorem 4*, and $\overline{\lambda}_k, k = 1, 2$, will now take a more clean form as

$$\begin{aligned}\overline{\lambda}_1 &= [\lambda_3 - \lambda_2, \dots, \lambda_n - \lambda_2]^T / \lambda_2, \quad \overline{\lambda}_2 = [\lambda_3(\lambda_3 - \lambda_2), \dots, \lambda_n(\lambda_n - \lambda_2)]^T, \\ \beta^{ij} &= [w_3 \tau_3^{ij}, \dots, w_n \tau_n^{ij}]^T.\end{aligned}\tag{16}$$

At first glance, if assuming all the eigenvalues are equally spaced on the real line (however, it is not true in general), a direct computation shows $\theta \rightarrow \arccos(\sqrt{15}/4) = 14.74^\circ$. In fact, smaller θ may be expected. Note that the diagonal elements of \mathbf{G} in these examples should have the same expectation, so should the off-diagonal elements. Therefore, the *Theorem 1* in [42] or *Theorem 1.1* in [43] can be applied here, which says that the gap between λ_1 and λ_2 is $O(n)$, and that between λ_i and $\lambda_{i+1}, i \geq 2$, is only $O(\sqrt{n})$. Such unbalanced gap distribution was also observed in [44, 45] (e.g., refer to Fig. 1 in [44]) for Google matrices constructed from the Albert-Barabási model [38] or randomized real-world University networks [46]. Based on the above analysis, roughly speaking, there are $O(\sqrt{n})$ eigenvalues λ_k satisfying $\lambda_k/\lambda_3 \rightarrow 1$, meaning from (16) that the first $O(\sqrt{n})$ coordinates of $\overline{\lambda}_1$ and $\overline{\lambda}_2$ tend to be “collinear”. It is such an

$n = 100$							$n = 1000$						$n = 2000$					
T_1	T_2	T_3	T_4	T_5	T_6		T_1	T_2	T_3	T_4	T_5	T_6	T_1	T_2	T_3	T_4	T_5	T_6
$E(\theta)$	1.42°	0.16°	1.84°	0.15°	8.51°	6.17°	0.46°	0.09°	0.6°	0.01°	3.22°	2.19°	0.32°	0.04°	0.43°	0.008°	2.33°	1.57°
$\text{Var}(\theta)$	10^{-4}	10^{-6}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-6}	10^{-9}	10^{-6}	10^{-8}	10^{-4}	10^{-5}	10^{-8}	10^{-10}	10^{-7}	10^{-10}	10^{-6}	10^{-6}

Table 1: Mean and variance of the angle between $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$ in (16) averaged over 800 runs. Here, the Google matrix \mathbf{G} is constructed as (2) with default parameters, where $\hat{\mathbf{G}}$ is the column-normalized version of $\bar{\mathbf{G}}$ corresponding to six types of graphs T_k . T_1 (or T_2) corresponds to $\bar{\mathbf{G}} \leftarrow \text{abs}(\text{rand}(n))$ followed by $\bar{\mathbf{G}} \leftarrow \bar{\mathbf{G}} + \bar{\mathbf{G}}'$ (or $\bar{\mathbf{G}} * \bar{\mathbf{G}}'$). T_3 (or T_4) is generated similarly to T_1 (or T_2), but with “rand(n)” replaced by “randn(n)”. T_5 and T_6 correspond to ER graphs with the sparse density valued at 0.1 and 0.2, respectively.

intrinsic “collinear effect” that forces θ close to zero.

Table. 1 depicts the mean and variance of θ in aforementioned examples over 800 sequential runs for each case, from which we see that in all the cases θ is concentrated around 0° while the variance approaches to zero as n increases. Generally, if assuming that ρ is approximately constant along arbitrary direction in \mathbb{R}^{n-2} , there exists

$$\pi_{ij} \approx 1 - \frac{\theta}{180^\circ}. \quad (17)$$

We will show in the next section that the above formula is highly agrees with our experimental results especially for large n , even although θ is not so close to 0 (as shown in the next section, typically θ is a small angle less than 10° in most cases).

2.4. Higher-Order Sign-Mirror Functions

Motivated by the ϕ_{ij} constructed in (8), it is natural to consider its higher-order version: $\phi_{ij} = (\mathbf{A}^m \mathbf{w})_i - (\mathbf{A}^m \mathbf{w})_j$, satisfying

$$w_i = w_j, \quad \mathbf{w} \geq 0, \quad (\mathbf{A}^k \mathbf{w})_i = (\mathbf{A}^k \mathbf{w})_j, \quad k=1, \dots, m-1, \quad (18)$$

where m is a preassigned positive integer. However, the practical algorithm in form keeps unchanged as (11).

To study π_{ij} in this case, similar to what we do previously for the case of $m=2$, we first expand $\mathbf{A}^k \mathbf{w}$, $k=1, \dots, m$, using (3) and (4). Then, from the constraint $(\mathbf{A}^k \mathbf{w})_i = (\mathbf{A}^k \mathbf{w})_j$, $k=1, \dots, m-1$, we obtain $m-1$ linear equations w.r.t.

$n = 100, m = 3$		$n = 100, m = 4$		$n = 1000, m = 3$		$n = 1000, m = 4$		$n = 2000, m = 3$		$n = 2000, m = 4$		
T_5	T_6	T_5	T_6	T_5	T_6	T_5	T_6	T_5	T_6	T_5	T_6	
$E(\theta)$	5.97°	4.46°	4.61°	3.48°	2.42°	1.68°	1.93°	1.34°	1.76°	1.20°	1.41°	0.97°
$\text{Var}(\theta)$	10^{-2}	10^{-3}	10^{-2}	10^{-3}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-6}	10^{-6}	10^{-6}	10^{-6}

Table 2: Mean and variance of θ based on higher-order sign-mirror functions with $m = 3, 4$, averaged over 500 runs in each case, where T_5 and T_6 are the same graph ensembles used in Table. 1.

$w_2\tau_2^{ij}, \dots, w_r\tau_r^{ij}, \gamma_{r+1}^{ij}, \dots, \gamma_{r+2s}^{ij}$, thus we can represent $w_2\tau_2^{ij}, \dots, w_m\tau_m^{ij}$ using the linear combination of $w_{m+1}\tau_{m+1}^{ij}, \dots, w_r\tau_r^{ij}, \gamma_{r+1}^{ij}, \dots, \gamma_{r+2s}^{ij}$ (here, without loss of generality we assume $\lambda_m \geq \lambda_{R,r+1}$). Next, let us eliminate $w_2\tau_2^{ij}, \dots, w_m\tau_m^{ij}$, from the expressions of $\Delta_{ij}(\infty)$ in (12) and $\phi_{ij} = (\mathbf{A}^m \mathbf{w})_i - (\mathbf{A}^m \mathbf{w})_j$ in (18), finally leading to a tight form $\phi_{ij}\Delta_{ij}(\infty) = (\hat{\boldsymbol{\lambda}}_1^T \boldsymbol{\beta}^{ij})(\hat{\boldsymbol{\lambda}}_2^T \boldsymbol{\beta}^{ij})$ which formally is the same to that in *theorem 4*. However, $\hat{\boldsymbol{\lambda}}_k, k = 1, 2$, and $\boldsymbol{\beta}^{ij}$ are of order $n - m$ in this case, and $\hat{\boldsymbol{\lambda}}_k$ takes a more complex dependence on the eigenvalues. The biggest benefit own to higher-order sign-mirror functions lies in our numerical observations, as shown in Table. 2, that θ gets closer to zero as m increases, which we believe is due to the stronger ‘‘collinear effect’’ between $\hat{\boldsymbol{\lambda}}_1$ and $\hat{\boldsymbol{\lambda}}_2$ for bigger m . However, this benefit is at the expense of more computational cost since up to m -order power of \mathbf{A} is required. Moreover, to ensure the existence of \mathbf{w} meeting the constraints in (18), there should be at least $m - 1$ free variables in \mathbf{w} (generally, the constraint $\mathbf{w} \geq 0$ requires a few additional free variables involved in \mathbf{w}), which is different from the case of $m = 2$ where \mathbf{w} can be composed of three different values as shown in (9).

At the end of this section, we provide a direct extension of (18) to multiple pairwise comparisons in one pass through the algorithm. Let \mathcal{S} be the set containing the indexes of nodes in question. Then, for any $i, j \in \mathcal{S}$, consider $\phi_{ij} = (\mathbf{A}^m \mathbf{w})_i - (\mathbf{A}^m \mathbf{w})_j$ with the following constraints:

$$w_{k'} = w_{k''}, k', k'' \in \mathcal{S}, \quad \mathbf{w} \geq 0, \quad (\mathbf{A}^k \mathbf{w})_{k'} = (\mathbf{A}^k \mathbf{w})_{k''}, k = 1, \dots, m - 1. \quad (19)$$

In such a manner, single calculation of \mathbf{w} that meets the above constraints resolves all the ϕ_{ij} ’s, $i, j \in \mathcal{S}$, i.e., via single evaluating \mathbf{w} all the pairwise orders induced from \mathcal{S} emerge based on (11). To guarantee the existence of \mathbf{w} in (19),

Parameter	T ₇ T ₈	Parameter	T ₇ T ₈
ST: γ , exponent in scale-free target degree distribution	1.5 1	CM: η , probability that a new node is assigned a new color	.01 .02
KL: q , number of random connections to add per node	2 4	PR: d , mean degree	2 4
SM: p , probability of adding a shortcut in a given row	.2 .5	RA: λ , fixed base of geometric decaying factor	.9 .95

Table 3: Alterable parameters “T₇” and “T₈” in six graph models. Parameter markers used here coincide with those used in Matlab codes [50].

\mathbf{w} should at least contain $|\mathcal{S}|(m - 1)$ free variables (plus additional freedom for satisfying $\mathbf{w} \geq 0$), where $|\mathcal{S}|$ denotes the size of \mathcal{S} .

3. Numerical Verification for θ

This section provides numerical evidence to support the concentration property of θ near to small angles along with its universality on various types of directed (DI) or undirected (UD) graphs generated by UD Stickiness (ST) model [47], UD Kleinberg’s model (KL) [48], DI Color Model (CM) [44], DI Preferential Attachment (PR) model [38], DI Small-World (SM) model [39], and DI Range Dependent (RA) model [49], as well as several real-world networks. The Matlab toolbox for generating six model based graphs can be downloaded from [50], and all the input parameters were set to default unless specially mentioned.

For six model based graphs, experiments were carried out using twelve different groups of parameter settings, say, for each fixed $n = 100, 1000$ or 2000 (the number of nodes) and $m = 2$ or 4 (the order of the sign-mirror function), we performed experiments using two different parameters “T₇” and “T₈” to control the sparseness of graphs, the actual meaning of which varies with the type of graphs as shown in Table. 3. Table. 4 depicts the mean and variance of θ (averaged over 800 runs in each case), the correct rate of pairwise comparisons based on the algorithm (11) (averaged over 5×10^7 comparisons in each case), and the estimate of π_{ij} from (17). In all the cases, the correct rate corresponding to $m = 4$ is slightly bigger (generally no more than 2%) than that corresponding to $m = 2$, thus we omit it in the table for clear view. From Table. 4, we see that: (a). The mean of θ is observably concentrated around a small angle (less than 11° in all the cases) while the variance is much smaller. Typically, it decreases

	$n = 100, m = 2$		$n = 100, m = 4$		$n = 1000, m = 2$		$n = 1000, m = 4$		$n = 2000, m = 2$		$n = 2000, m = 4$	
	T_7	T_8	T_7	T_8	T_7	T_8	T_7	T_8	T_7	T_8	T_7	T_8
$E_{st}(\theta)$	10.77°	9.18°	7.11°	6.18°	9.10°	8.68°	7.88°	7.69°	8.63°	8.42°	8.34°	8.26°
$Var_{st}(\theta)$	1.62°	4.21°	3.56°	4.45°	3.92°	5.60°	3.07°	2.91°	5.46°	6.37°	2.62°	2.53°
$1 - E_{st}(\theta)/180$	94.01%	94.89%	96.04%	96.56%	94.93%	95.17%	95.61%	95.72%	95.20%	95.32%	95.36%	95.40%
Correct rate	92.47%	96.84%	-	-	92.54%	97.73%	-	-	95.32%	97.86%	-	-
$E_{cm}(\theta)$	3.58°	2.61°	1.83°	1.34°	2.28°	2.13°	1.07°	1.02°	2.35°	2.26°	1.04°	1.01°
$Var_{cm}(\theta)$	0.03°	2.09°	0.02°	0.51°	1.06°	1.38°	0.15°	0.48°	1.20°	1.18°	0.30°	0.77°
$1 - E_{cm}(\theta)/180$	98.01%	98.55%	98.98%	99.26%	98.73%	98.82%	99.41%	99.43%	98.69%	98.74%	99.42%	99.44%
Correct rate	97.72%	98.61%	-	-	97.94%	99.51%	-	-	98.58%	99.47%	-	-
$E_{kl}(\theta)$	9.69°	9.14°	5.68°	5.43°	9.25°	9.26°	5.50°	5.53°	9.30°	9.30°	5.56°	5.57°
$Var_{kl}(\theta)$	0.02°	0.31°	0.01°	0.08°	0.40°	0.37°	0.12°	0.12°	0.40°	0.39°	0.13°	0.13°
$1 - E_{kl}(\theta)/180$	94.61%	94.91%	96.83%	96.98%	94.85%	94.85%	96.94%	96.92%	94.83%	94.82%	96.90%	96.90%
Correct rate	92.43%	93.46%	-	-	92.80%	95.59%	-	-	91.29%	94.85%	-	-
$E_{pr}(\theta)$	10.51°	10.13°	5.80°	5.25°	10.20°	10.14°	5.28°	5.26°	10.17°	10.14°	5.28°	5.27°
$Var_{pr}(\theta)$	0.04°	1.15°	0.01°	0.17°	1.17°	1.16°	0.19°	0.18°	1.17°	1.16°	0.18°	0.18°
$1 - E_{pr}(\theta)/180$	93.60%	94.37%	96.77%	97.08%	94.33%	94.36%	97.02%	97.07%	94.34%	94.36%	97.06%	97.07%
Correct rate	82.82%	91.69%	-	-	96.22%	98.75%	-	-	98.71%	99.67%	-	-
$E_{sm}(\theta)$	10.77°	10.79°	5.43°	4.53°	10.80°	10.82°	5.53°	5.54°	10.82°	10.83°	5.54°	5.55°
$Var_{sm}(\theta)$	10 ⁻³ °	0.01°	10 ⁻³ °	0.01°	0.01°	0.01°	0.01°	0.01°	0.01°	0.01°	0.01°	0.01°
$1 - E_{sm}(\theta)/180$	96.92%	96.91%	93.43%	93.42%	96.91%	96.91%	93.45%	93.44%	96.97%	96.92%	93.44%	93.43%
Correct rate	84.11%	88.09%	-	-	89.45%	93.81%	-	-	91.59%	94.70%	-	-
$E_{ra}(\theta)$	7.11°	6.20°	4.74°	3.99°	6.22°	6.20°	4.02°	4.03°	6.21°	6.21°	4.05°	4.05°
$Var_{ra}(\theta)$	0.01°	0.84°	0.01°	0.55°	0.84°	0.84°	0.58°	0.57°	0.84°	0.84°	0.59°	0.58°
$1 - E_{ra}(\theta)/180$	96.04%	96.55%	97.36%	97.77%	96.54%	96.55%	97.72%	97.75%	96.54%	96.54%	97.74%	97.74%
Correct rate	95.04%	97.87%	-	-	96.56%	96.74%	-	-	96.47%	97.11%	-	-

Table 4: Mean and variance of θ (averaged over 800 runs in each case), pairwise correct rate (averaged over 5×10^7 comparisons in each case), and estimate of π_{ij} based on (17) for six types of graphs with 12 groups of different parameter settings. Meaning of “ T_7 ” and “ T_8 ” for each model can be found in Table. 3. In all the cases, the correct rate corresponding to $m = 4$ is slightly larger (generally no more than 2%) than that corresponding to $m = 2$, which is omitted here for clear view.

with the increasing of the size of graphs; (b). The mean of θ based on $m = 4$ tends to be smaller than that based on $m = 2$ although the resulting pairwise correct rate has no significant difference; (c). The pairwise correct rate is well approximated by π_{ij} in most cases, especially when n is relatively large; (d). The pairwise correct rate is over 90% in almost all the cases corresponding to $n = 1000, 2000$. In fact, we believe that θ and π_{ij} becomes less random for large n , and the potential principle is mainly due to the special structure (16) and “the large number law for random matrices”.

Next, we perform simulations on a set of real-world networks. Here, eight datasets were used here including four (“Roget”, “ODLIS”, “CSphs” and “Net-

	Roget	ODLIS	CSphd	Networktheory	EMN	PGP	p2p-Gnutella08	p2p-Gnutella09
Properties	DI/UW	DI/UW	DI/UW	DI/WI	DI/UW	DI/UW	DI/UW	DI/UW
Number of nodes	1022	2909	1882	1589	453	10680	6301	8114
Number of edges	5075	18419	1740	2742	4596	24340	20777	26013
θ	9.02°	8.70°	1.95°	1.29°	3.31°	5.56°	3.10°	2.86°
Correct rate	90.95%	91.71%	95.92%	93.59%	93.09%	92.77%	98.34 %	98.47%

Table 5: Pairwise correct rate (evaluated after all the possible pairs pass through the algorithm) and θ on eight real-world graphs.

worktheory”) taken from [51], two (“p2p-Gnutella08” and “p2p-Gnutella09”) in the SNAP collection [52], and two (“Elegans Metabolic Network (EMN)” and “PGP”) taken from [53]. Table. 5 depicts the properties (direct/undirect and weighted/unweighted, respectively abbreviated by DI/UD and WE/UW in the table. See more information in the dataset documents), number of nodes and edges along with θ and the pairwise correct rate computed from all possible pairwise comparisons. From the figure, we see the consistent performance due to the small θ and high pairwise correct rate.

4. From Pairwise Order to Top k List

This section provides an $O(kn)$ algorithm for extracting the top k list. Let n_i be the number of remaining nodes after the i^{th} iteration with $n_0=n$. In the $(i+1)^{\text{th}}$ iteration, n_i nodes are randomly divided into $\frac{n_i}{m_1 k}$, $m_1 > 1$, subgroups such that there are $m_1 k$ nodes in each subgroup. Then we run the naive Subgroup Ranking Algorithm (SRA, will be discussed shortly) to obtain the whole ranking list for each subgroup, which performs $m_1 k(m_1 k - 1)/2$ comparisons using (11) in each subgroup, thus totally leading to $(m_1 k - 1)n_i/2$ runs of (11). Thereafter, only the top $m_2 k$, $1 \leq m_2 < m_1$, nodes in each subgroup are kept for the follow-up processing, meaning $n_{i+1} = m_2 n_i / m_1$. Thus, we can compute the total number required for pairwise ranking operators as

$$\frac{n(m_1 k - 1)}{2} \left[1 + \frac{m_2}{m_1} + \left(\frac{m_2}{m_1} \right)^2 + \dots \right] = \frac{kn}{2} \cdot \frac{m_1^2 - \frac{m_1}{k}}{m_1 - m_2} = \frac{kn}{2} \cdot \left[m_1 - m_2 + \frac{m_2^2 - \frac{m_2}{k}}{m_1 - m_2} + 2m_2 - \frac{1}{k} \right].$$

The above equation reaches its minimum $kn[\sqrt{m_2^2 - m_2/k} + m_2 - (1/2k)] \approx 2m_2 kn$ (generally m_2/k is small) iff $m_1 = m_2 + \sqrt{m_2^2 - m_2/k} \approx 2m_2$. Since single pairwise

ranking based on (11) causes $O(1)$ cost averagely, our top k extraction algorithm totally has an $O(kn)$ time complexity. Note that when SRA perfectly computing the ranking list for each subgroup, we can let $m_2=1$ since every element in the final top k list surely belongs to any of the top k lists for those temporarily generated subgroups containing that element during iterations.

Subgroup Ranking Algorithm (SRA): Denote by v_1, \dots, v_{m_1k} , the nodes in a subgroup, and associate v_i with a score f_i (initialized to zero). For each pair of (v_i, v_j) , let $f_i \leftarrow f_i + 1$ if v_i is ranked higher than v_j based on (11), otherwise $f_j \leftarrow f_j + 1$. Repeat the above processing until $m_1k(m_1k-1)/2$ pairs pass through the algorithm. Finally, the ranking list is constructed based on f_i 's.

Clearly, SRA is specially well-qualified on relatively clean pairwise orders, which is just the case here. More sophisticated variants can be found in [54] and the references therein, but most of which are specially designed for noisy cases thus lead to higher computational cost.

Without loss of generality, assume $f_1 \geq \dots \geq f_{m_1k}$. Some interesting issues emerge in SRA: (a). In the ideal case that (11) generates 100% correct outputs for all the pairs, we have $f_i = m_1k - i$, thus the ranking list based on f_i 's perfectly matches the truth; (b). With the probability π_{ij} (over 90% for various types of graphs as shown in the last section), our algorithm outputs a correct pairwise order. Thus, f_i may diverge a little from its ideal value $m_1k - i$, causing potential disorders in the ranking list. A typical case is that there possibly exists $f_i = f_j, i \neq j$, such that we can not rank v_i and v_j using their scores. In practice, we just put the nodes with equal score together as a chunk, not to tell the precedence between them. However, f_i will not to diverge too much from $m_1k - i$ since π_{ij} is high enough, neither will the ranking list. Thus, we suggest to choose m_2 slightly more than one in practice, e.g, $m_2 = 1/\pi_{ij} \approx 1.15$ is used in the following simulation.

Finally, we use an application to demonstrate the performance of the algorithm of this paper based on the top k list extraction while comparing it with other four iteration based methods, i.e., the popular principle eigenvector solver *Power Method* (PM) [37], and three Power-Method-originated PageRank solver:

Power-Inner-Outer (PIO) [27], *Power-Arnoldi* (PA) and *GMRES-Power* (GP) methods [36]. In what follows, the PM, PA, PIO, and GP are called as the iteration based methods for convenience. Note that among all the algorithms for computing the PageRank score, the algorithm of this paper seems to be the only one explicitly avoiding eigenvector computations, thus there exists no trivial way for other PageRank solvers to directly achieve the top k list extraction. The common way on this task for the iteration based methods is first to run iterations up to v rounds, then reports the top k elements in the resulting vector as an approximation for the ground truth.

In this simulation, all the experiments were carried out on a Matlab 2015b/2.4 GHz/32 GB RAM platform. Two large-scale networks are employed here. The first one with the dimension of 20×10^4 was generated by the Color Model [44] (the mean degree was chosen around 7), an extension of the Preferential Attachment model [38], which is a more popular choice for artificially imitating the real WWW networks. Another is the sparse *Web-Stanford* web networks with 281,903 nodes and 2,312,497 links [55] from the real world. Since a larger α in (2) leads to a more challenging problem [15], here we set $\alpha = 0.99$, which is similarly to that used in [36]. All the parameters of five algorithms were set to their default values, e.g., $m = 2$ was used in our proposed algorithm, the restart number valued at 6 was used in GP [36], all-one initial vectors were used in the iteration based methods, and the tolerance $\tau = 10^{-8}$ was used for measuring the convergence. That is to say, when the 2-norm of the difference between two successive iterative vectors is less than 10^{-8} , the iteration based methods are regarded to get converged. Here, we not only ran the iteration based methods until convergence, we but also ran them in a fixed number v of iterations, i.e., we also compared the performance achieved by those four algorithms in the context of early stopping before convergence. Such an experimental design is due to the considerations that we want to compare our proposed algorithm to the iteration based methods that are equipped with an ability to freely choose the tradeoff between the running time and precision. The following index was

employed to measure the precision of the algorithms:

$$\text{precision} = \frac{\#\{\text{The computed top } k \text{ list} \cap \text{The ground truth}\}}{k},$$

where $\#\{\cdot\}$ denotes the number of elements in a set.

Table 6 shows for the iteration based methods the running time in seconds and the corresponding precision for various combinations of $v = 1, 5, 10, 20, 40$ and $k = 20, 50, 100$, on two aforementioned networks, as well as those after convergence, which is depicted in the three sub-columns tied to the “*Stable*” symbol. Since the algorithm of this paper, denoted by the “*Our*” symbol in the table, is not an iteration based one, its performance is thus only shown in the “*Stable*” column, where we also illustrate the iteration steps required for the four iteration based methods to get converged. Note that in the iteration based methods the iterations substantially dominated the running time while that consumed by sorting is negligible, thus we only response to v in the table. On the contrary, the time consumed by our proposed algorithm is clearly related to k since it is based on pairwise comparisons with the complexity $O(kn)$. In addition, each of five algorithms is described in the table by two successive rows, respectively corresponding to the running time or the precision in the upper or lower row. The performance on the Color Model generated networks were averaged on 800 successive runs of algorithms. From the table, we see that: (1). All the four iteration based methods got converged in both experiments, e.g., it took 23.84/33.24 and 7.14/9.96 seconds, respectively for the PM and GP methods, to perfectly extract the top $k = 20/50/100$ list; (2). Among the four iteration based methods, the GP achieved the best precision with the fastest speed before convergence. It reached the precision of 95.7%/94.2%/93.3% in 5.41 seconds on the Color Model generated networks, and 93.3%/92.5%.91.4% in 8.11 seconds on the *Web-Stanford* web networks, respectively for $k = 20/50/100$; (3). Our proposed method did the work with the precision of 98.9%/98.9%/98.8% for $k = 20/50/100$ on the Color Model generated networks only in less than 0.5 seconds, and with the precision of 97.4%/97.3%/97.3% for $k = 20/50/100$ on *Web-Stanford* web networks only in less than 0.7 seconds, which definitely shows

the essential improvement on the running speed of our proposed algorithm for approximating the top k list with super precision.

As the end of this section, we point out that more studies are necessary to go deeper along this way, e.g., more advanced knowledge from other communities, especially from the insights of random matrices, will definitely be helpful to this line. Moreover, more efficient top k extraction algorithms based on noisy pairwise comparisons will do much benefit in practice. We hope that this paper casts the first stone for penetrating the PageRank related algorithms from a probability point of view while not computing the exact value of eigenvectors.

5. Conclusion

This paper provides an $O(1)$ algorithm for pairwise comparisons of PageRank score from a probabilistic view, based on which the top k list can be extracted in $O(kn)$. It is not necessary to compute the exact values of the principle eigenvectors of the Google matrix based on our proposed frameworks because pairwise PageRank orders naturally emerge from two-hop walks. The key tool used in this paper is a specially designed sign-mirror function and a parameter curve, whose low-order derivative information implies pairwise PageRank orders with high probability, which is essentially due to the underlying spectral distribution law of random matrices. Although more quantitative analysis from the communities of random matrices is required to get deeper insight, this paper has shed the first light on this direction and the algorithm of this paper has made it possible for PageRank to deal with super large-scale datasets in real time.

References

References

- [1] D. Cvetković and S. Simić, Graph spectra in computer science, *Linear Algebra and its Applications*. 434 (6) (2011) 1545–1562.

v	10			30			50			70			100			Stable		
k	20	50	100	20	50	100	20	50	100	20	50	100	20	50	100	20	50	100
Color Model																		
PM	0.17			0.48			0.83			1.15			1.66			23.84, $v = 987$		
	.655	.628	.591	.662	.636	.600	.671	.643	.608	.678	.652	.616	.686	.665	.629	1.00	1.00	1.00
PIO	0.36			1.03			1.68			2.37			3.40			16.59, $v = 361$		
	.701	.658	.623	.718	.682	.658	.734	.701	.665	.751	.716	.680	.776	.744	.715	1.00	1.00	1.00
PA	0.83			2.42			4.09			5.71			8.17			11.72, $v = 125$		
	.739	.702	.681	.781	.750	.732	.823	.798	.779	.871	.841	.834	.933	.915	.901	1.00	1.00	1.00
GP	0.54			1.59			2.70			3.75			5.41			7.14, $v = 107$		
	.754	.731	.709	.801	.784	.763	.846	.835	.818	.892	.881	.872	.957	.942	.933	1.00	1.00	1.00
Our	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.09	0.25	0.48
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	.989	.989	.988
Web-Stanford																		
PM	0.25			0.73			1.22			1.67			2.39			33.24, $v = 1461$		
	.621	.604	.568	.625	.611	.572	.632	.615	.581	.636	.622	.587	.644	.629	.596	1.00	1.00	1.00
PIO	0.52			1.54			2.52			3.56			5.09			24.82, $v = 505$		
	.652	.635	.597	.669	.645	.616	.681	.671	.641	.684	.679	.652	.716	.696	.670	1.00	1.00	1.00
PA	1.24			3.63			6.14			8.57			12.25			16.15, $v = 138$		
	.701	.658	.632	.742	.715	.689	.783	.765	.739	.831	.805	.786	.896	.883	.875	1.00	1.00	1.00
GP	0.81			2.38			4.05			5.63			8.11			9.96, $v = 119$		
	.725	.693	.648	.781	.754	.708	.818	.792	.763	.863	.842	.829	.933	.925	.914	1.00	1.00	1.00
Our	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.14	0.35	0.67
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	.974	.973	.973

Table 6: The running time in seconds and the precision of four iteration based methods on two networks based on the preassigned iteration number $v = 10, 30, 50, 70, 100$ for $k = 20, 50, 100$, or until convergence (corresponding to the "stable" column, where we also show the performance of our proposed algorithm since it is a non-iteration based algorithm). Each of five algorithms is described in the table by two successive rows, respectively corresponding to the running time or the precision in the upper or lower row. The performance on the Color Model generated networks were averaged on 800 successive runs.

- [2] D. A. Spielman, Spectral graph theory and its applications, in: FOCS, 2007, pp. 29–38.
- [3] R. Singh, J. Xu, and B. Berge, Global alignment of multiple protein interaction networks with application to functional orthology detection, Proceedings of the National Academy of Sciences. 105 (2008) 12763–12768.
- [4] G. Pinski and F. Narin, Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics, Information Processing & Management. 12 (5) (1976) 297–312.

- [5] J. Bollen, M. A. Rodriguez, and H. V. de Sompel, Journal status, *Scientometrics*. 69 (3) (2006) 669–687.
- [6] J. R. Seeley, The net of reciprocal influence: A problem in treating sociometric data, *The Canadian Journal of Psychology*. 3 (1949) 234–240.
- [7] L. Katz, A new status index derived from sociometric analysis, *Psychometrika*. 18 (1) (1953) 39–43.
- [8] C. H. Hubbell, An input-output approach to clique identification, *Sociometry*. 28 (1) (1965) 377–399.
- [9] S. U. Pillai, T. Suel, and S. Cha, The Perron-Frobenius theorem: some of its applications, *IEEE Signal Processing Magazine*. 22 (2) (2005) 62–75.
- [10] S. Brin and L. Page, The anatomy of a large-scale hypertextual Web search engine, *Computer Networks and ISDN Systems*. 30 (1-7) (1998) 107–117.
- [11] M. Franceschet, PageRank: standing on the shoulders of giants, *Communications of the ACM*. 54 (6) (2011) 92–101.
- [12] S. Vigna, Spectral ranking. 2009, arXiv preprint arXiv:0912.0238.
- [13] A. N. Langville and C. D. Meyer, Deeper inside pagerank, *Internet Mathematics*. 1 (3) (2004) 335–380.
- [14] P. Boldi, M. Santini, and S. Vigna, PageRank: functional dependencies, *ACM Transactions on Information Systems*. 27 (4) (2009) 1–23.
- [15] T. Haveliwala, S. Kamvar, The second eigenvalue of the Google matrix, *Stanford University Technical Report*, 2003.
- [16] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova, Monte Carlo methods in PageRank computation: when one iteration is sufficient, *SIAM Journal on Numerical Analysis*. 45 (2) (2007) 890–904.
- [17] G. Jeh and J. Widom, Scaling personalized web search, in: *WWW*, 2003, pp. 271–279.

- [18] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub, Extrapolation methods for accelerating PageRank computations, in: WWW, 2003, pp. 261–270.
- [19] S. Kamvar, T. Haveliwala, and G. Golub, Adaptive methods for the computation of PageRank, *Linear Algebra and its Applications*. 386 (2004) 51–65.
- [20] [4] R. Andersen and C. Fan, Local graph partitioning using PageRank vectors, in: FOCS, 2006, pp. 475–486.
- [21] A. D. Sarma, S. Gollapudi, and R. Panigrahy, Estimating pagerank on graph streams, *Journal of the ACM*. 58 (3) (2011) 1–18.
- [22] A. Vattani, L. Jolla, and M. Gurevich, Preserving personalized pagerank in subgraphs, in: ICML, 2011, pp. 793–800.
- [23] N. Perra, V. Zlatić, A. Chessa, C. Conti, D. Donato, and G. Caldarelli, PageRank equation and localization in the WWW, *Europhysics Letters*. 88 (4) (2009) 48002.
- [24] G. D. Paparo and M. A. Martin-Delgado, Google in a quantum network, *Scientific Reports*. 2 (2012) 1–12.
- [25] S. Garnerone, P. Zanardi, and D. A. Lidar, Adiabatic quantum algorithm for search engine ranking, *Physical Review Letters*. 108 (23) (2012) 230506.
- [26] Y. Jing and S. Baluja, Visualrank: Applying pagerank to large-scale image search, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 30 (11) (2008) 1877–1890.
- [27] D. Gleich, A. Gray, C. Greif, and T. Lau, An inner-outer iteration method for computing PageRank, *SIAM Journal on Scientific Computing*, 32 (1) (2010) 349–371.
- [28] Z. Li, J. Liu, C. Xu, and H. Lu, Mlrank: Multi-correlation learning to rank for image annotation, *Pattern Recognition*. 46 (10) (2013) 2700–2710.

- [29] R. Martn-Flez and T. Xiang, Uncooperative gait recognition by learning to rank, *Pattern Recognition*. 47 (12) (2014) 3793–3806.
- [30] T. Celik, Spatial mutual information and PageRank-based contrast enhancement and quality-aware relative contrast measure, *IEEE Transactions on Image Processing*. 25 (10) (2016) 4719–4728.
- [31] L. Liu, L. Sun, S. Chen, M. Liu, and J. Zhong, K-PRSCAN: A clustering method based on PageRank, *Neurocomputing*. 175 (2016) 65–80.
- [32] Y. Tang and Y. Li, Pairwise comparisons in spectral ranking, *Neurocomputing*. 216 (2016) 561–569.
- [33] Z. Li, F. Nie, X. Chang, L. Nie, H. Zhang, and Y. Yang, Rank-constrained spectral clustering with flexible embedding, *IEEE Transactions on Neural Networks and Learning Systems*. 29 (12) (2018) 6073–6082.
- [34] X. Shi, M. Sapkota, F. Xing, F. Liu, L. Cui, and L. Yang, Pairwise based deep ranking hashing for histopathology image classification and retrieval, *Pattern Recognition*. 81 (2018) 14–22.
- [35] M. Buzzanca, V. Carchiolo, A. Longheu, M. Malgeri, and G. Mangioni, Black hole metric: Overcoming the pagerank normalization problem, *Information Sciences*. 438 (2018) 58–72.
- [36] C. Gu, X. Jiang, C. Shao, and Z. Chen, A GMRES-Power algorithm for computing PageRank problems, *Journal of Computational and Applied Mathematics*. 343 (2018) 113–123.
- [37] G. H. Golub and C. F. Van Loan, *Matrix computations*, 3rd edition, JHU Press, 2012.
- [38] A. L. Barabási and R. Albert, Emergence of scaling in random networks, *Science*. 286 (5439) (1999) 509–512.
- [39] D. J. Watts and S. H. Strogatz, Collective dynamics of ‘small-world’ networks, *Nature*. 393 (6684) (1998) 440–442.

- [40] P. Erdős and A. Rényi, On random graphs, *Publ. Math. Debrecen.* 6 (1959) 290–297.
- [41] A. Edelman, E. Kostlan, and M. Shub, How many eigenvalues of a random matrix are real? *Journal of the American Mathematical Society.* 7 (1) (1994) 247–267.
- [42] Z. Füredi and J. Komlós, The eigenvalues of random symmetric matrices, *Combinatorica.* 1 (3) (1981) 233–241.
- [43] M. Krivelevich and B. Sudakov, The largest eigenvalue of sparse random graphs, *Combinatorics Probability and Computing.* 12 (1) (2003) 61–72.
- [44] B. Georgeot, O. Giraud, and D. L. Shepelyansky, Spectral properties of the Google matrix of the World Wide Web and other directed networks, *Physical Review E.* 81 (5) (2010) 056109.
- [45] O. Giraud, B. Georgeot, and D. L. Shepelyansky, Delocalization transition for the Google matrix, *Physical Review E.* 80 (2) (2009) 026107.
- [46] Academic Web Link Database Project,
<http://cybermetrics.wlv.ac.uk/database/>.
- [47] N. Pržulj and D. J. Higham, Modelling protein-protein interaction networks via a stickiness index, *Journal of the Royal Society Interface.* 3 (10) (2006) 711–716.
- [48] J. M. Kleinberg, Navigation in a small world, *Nature.* 406 (6798) (2000) 845.
- [49] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S. M. Kim, and D. Eisenberg, DIP The Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions, *Nucleic Acids Research.* 30 (1) (2002) 303–305.
- [50] <http://www.mathstat.strath.ac.uk/outreach/contest/toolbox.html>.

- [51] V. Batagelj and A. Mrvar, Pajek datasets, <http://vlado.fmf.uni-lj.si/pub/networks/data/>.
- [52] <http://snap.stanford.edu/data/>.
- [53] <http://deim.urv.cat/~aarenas/data/welcome.htm>.
- [54] F. L. Wauthier and M. I. Jordan, Efficient ranking from pairwise comparisons, in: ICML, 2013, pp. 109–117.
- [55] <http://www.cise.ufl.edu/research/sparse/matrices/groups.html>