A Novel Node Ranking Algorithm Using Complex Numbers: A Comparison to Google PageRank

Keita Sugihara
Department of Mechatronics, Faculty of Science and Engineering, Nanzan University
E-Mail: sugihara@nanzan-u.ac.jp

Abstract
This study proposes the usage of complex numbers in node ranking calculations for a network. We calculate node scores in the graph by using the Hermitian adjacency matrix and the complex plane through an algorithm, called HermitianStatus. We then compare the result of the method to that of Google PageRank, which is a widely spread algorithm for search engines. In conclusion, HermitianStatus has advantages over PageRank.

Key words: Google PageRank, link analysis, Hermitian adjacency matrix

1. Introduction

This study proposes an algorithm, called HermitianStatus, for calculating the node score in a network based on the link relation among nodes. The method expresses the link relations between the node using complex numbers. HermitianStatus is then compared to Google PageRank.

2. PageRank

PageRank assigns authority weights to each node of a network based on the network link structure [1]. Google's algorithm gives a high score to a node depending on the following three characteristics: node with (i) an in-link to it from a high score page; (ii) many in-links to the node; and (iii) selected out-links to it [2]. To realize these characteristics, PageRank employs the traditional adjacency matrix to represent a network. This matrix for the graph $A = (V, E)$ is defined as follows, where u and v represent nodes in the graph :

$$A_{uv} = \begin{cases} 1 & \text{if} \quad uv \in E; \\ 0 & \text{othewise.} \end{cases}$$

In PageRank, the eigenvalue problem for matrix G, which is created from A, is solved using λ_1 to represent the maximal eigenvalue of G [3]. In the equation, N stands for the number of the nodes in the network, and, r_i represents the score of node n_i.

$$R = \frac{1}{\lambda_1} G R$$

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix} = \frac{1}{\lambda_1} \left(d \times \left(\begin{bmatrix} a_{11} & 0 & \dots a_{N1} \\ a_{12} & 0 & \dots a_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1N} & 0 & \dots a_{NN} \end{bmatrix} \begin{bmatrix} 1/\Sigma_j a_{1j} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 1/\Sigma_j a_{Nj} \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{N} & \dots & 0 \\ 0 & \frac{1}{N} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{1}{N} & 0 & 0 \end{bmatrix} \right) + (1-d) \times \begin{bmatrix} \frac{1}{N} & \frac{1}{N} & \dots & \frac{1}{N} \\ \frac{1}{N} & \frac{1}{N} & \dots & \frac{1}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{N} & \frac{1}{N} & \dots & \frac{1}{N} \end{bmatrix} \right) \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{bmatrix} \quad (1.1)$$

where, the damping factor $d(0 < d < 1)$ is used to create a strongly connected graph, G. In a strongly connected network, one can reach every node via nodes using the link directions.

The PageRank scores of network nodes are defined as the corresponding element values of the vector R. The damping factor value 0.85 is often used in PageRank. When applied to the net, PageRank's usage of the damping factor transforms the network to one strongly connected component. Figure 1 shows the PageRank ranking at a damping factor value of 0.85 for the nodes in the network shown by the colors.

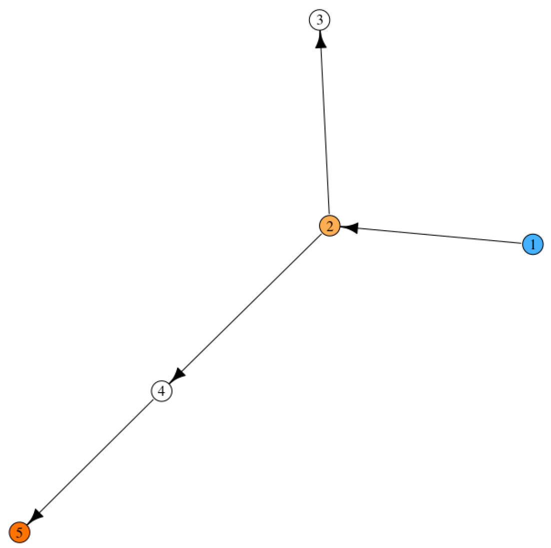

Figure 1: 5 node network with its PageRank result for the nodes (blue: low score and orange: high score).

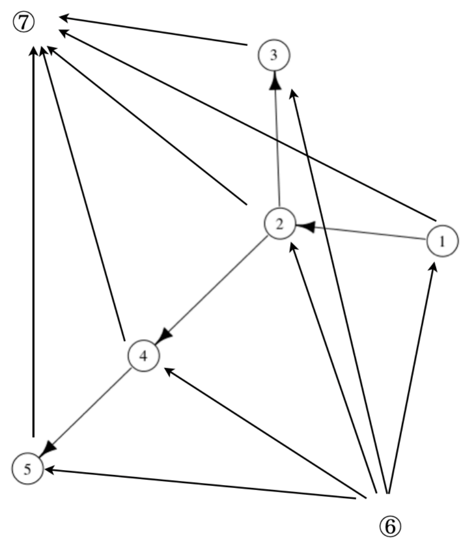

Figure 2: 7 node network transformed from the network in Figure 1.

3. HermitianStatus

3.1. Hermitian adjacency matrix

HermitianStatus utilizes the Hermitian adjacency matrix to represent a network. This Hermitian matrix for the graph $X = (V, E)$ is defined as follows using i to represent the imaginary unit [4] , where u and v represent nodes in the graph :

$$H_{uv} = \begin{cases} 1 & \text{if} \quad uv \in E \quad \text{and} \quad vu \in E; \\ i & \text{if} \quad uv \in E \quad \text{and} \quad vu \notin E; \\ -i & \text{if} \quad uv \notin E \quad \text{and} \quad vu \in E; \\ 0 & \text{otherwise.} \end{cases}$$

3.2. Algorithm

In HermitianStatus, we decompose the entire network into weakly connected networks. A weakly connected network means one can reach every node via nodes if or if not one ignores the link direction. We presuppose that in every weakly connected network, one node

exists without an in-link, and another exists without an out-link. With the algorithm, the lowest score is given for the former, while the highest score is assigned to the latter.
The HermitianStatus algorithm will be elaborated below using the network in Figure 1 as an example. In this case, the weakly connected network constitutes the entire network.

1. - Set the total node number N in the whole graph. In each weakly connected component of the entire graph, describe its node number as n and create the adjacency matrix A_0.
   -In the case of the example network, the adjacency matrix is given as follows:

$$A_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2. - Add the following two dummy nodes and links to the weakly connected component. First, the dummy node is $n + 1$ and links from the dummy to all network nodes. Second, the dummy node is $n + 2$ and links to the dummy from all nodes, excluding the first dummy. Create the adjacency matrix A for the transformed component.
   - This matrix for the transformed network in Figure 2 is presented below as an example.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3. - Create the Hermitian adjacency matrix H of A.
   - The matrix for the transformed example network is presented as follows:

$$H = \begin{bmatrix} 0 & i & 0 & 0 & 0 & -i & i \\ -i & 0 & i & i & 0 & -i & i \\ 0 & -i & 0 & 0 & 0 & -i & i \\ 0 & -i & 0 & 0 & i & -i & i \\ 0 & 0 & 0 & -i & 0 & -i & i \\ i & i & i & i & i & 0 & 0 \\ -i & -i & -i & -i & -i & 0 & 0 \end{bmatrix}$$

4. - In H, replace each i with 0 and change every $-i$ to $cos(-\frac{\pi}{2N}) + isin(-\frac{\pi}{2N})$ creating H_1.
   - The next matrix for the example is H_1.

$$H_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & 0 \end{bmatrix}$$

5. - In H_1, replace all diagonal elements 0s with -1, creating H_2.
   - The matrix in the example is presented below:

$$H_2 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & -1 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & -1 \end{bmatrix}$$

6. - Solve the linear equation $H_2 x = b,$ where $n \times 1$ vector b has the value of -1 for the element corresponding to the node without an in-link and the values 0s for the remaining elements.
   - In the example, the equation is presented as

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & -1 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & -1 & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & cos(-\frac{\pi}{2N})+isin(-\frac{\pi}{2N}) & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 0 \end{bmatrix} \qquad (3.1)$$

7. - Locate $x_1,\ x_2,$ and x_n on the complex plane as vectors.
   - Figure 3 depicts an example for this.

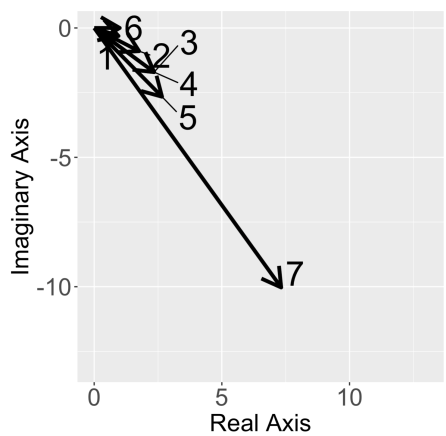

Figure 3: The fourth quadrant of the complex plane with the 7 vectors corresponding to the 7 nodes in the network transformed from the original network in Figure 1.

8. Define the node n_i's score, s_{i1}, on the complex plane.

$$s_{i1} = |x_i| \times \{\frac{2\pi - Arg(x_i)}{\frac{\pi}{2N}}\}^{k_1}$$

9. Multiply the s_{i1} value by the k_2th power of node x_i's in-link number, creating s_{i2} for n_i.

$$s_{i2} = \text{Indegree}(n_i)^{k2} \times s_{i1}$$

10. - Divide s_{i2} by the k3th power of the multiplication of each out-link number of the nod on the path from the node without an in-link to the node excluding itself, $n_1, \ldots, n_{i-2}, n_{i-1}$, creating s_{i3} for n_i.

$$s_{i3} = \frac{s_{i2}}{\{(\text{Outdegree}(n_1) \times \cdots \times (\text{Outdegree}(n_{i-2}) \times (\text{Outdegree}(n_{i-1})\}^{k_3}}$$

11. - Multiply the value of s_{i3} by the k_4th power of the node number of the component, n, setting s_{i4} for x_i.

$$s_{i4} = \frac{s_{i3}}{n^{k_4}}$$

12. Change the scores of the nodes without an in-link to zero.

Table 1: Purpose and Description.

1	Decomposition of the entire network into weakly connected components
2	Assuring even components without the lowest and highest score nodes to have them
3	Introducing imaginary numbers to be used on the complex plane.
4	Confining all the vectors corresponding to the nodes of the network into the fourth quadrant of the complex plane
5	For the liner equation in the next step
6	Earning the relative location of the vectors corresponding to the nodes on the complex plane and setting the vector corresponding to the node without in-link on the complex plane
7	Locating the vectors on the complex plane; each vector on the plane is the composition of $\frac{\pi}{2N}$ clock wise rotated other vectors corresponding to the nodes from which the node has in-links.
8	Adjusting k_1 (>0) for the first characteristic of PageRank: (i) an in-link to it from a high score page angle part of this definition corresponds to the number of links from the node with 0 in-link to the node, and, the vector length part of the formula represents the length of the vector.
9	Changing k_2 (>0) for the second characteristic of the Google algorithm: (ii) many in-links to the node.
10	Using k_3 (>0) for the third PageRank's characteristic :(iii) selected out-links to it.
11	The parameter k_4 (>0) refers to the node number of the weakly connected component the node is belonging to
12	Making all scores of the node without in-links 0.

The standard HermitianStatus scores of the network nodes in Figure 1 resulted in s_1 = 10.0000000, s_2 = 0.6308773, s_3 = 0.6178998, s_4 = 0.6178998, and s_5 = 1.0000000 with k_1 = 1, k_2 = 1, k_3 = 1, and k_4 = 1, respectively. This ranking is exactly the same as the PageRank result in Figure 1.

Table 1 presents the purposes and description of each step of the algorithm. Note that this algorithm is applied to each weakly connected component of the entire network. Moreover, we can compare the node scores of the nodes in different components, thanks to the node number of the entire network, *N*.

4. Experimental Results

The programing language R and its Rcpp package were used. The package facilitates extending R with the C++ programing language [5]. These tools have been used on Linux to write programs for testing the algorithm and compare it to PageRank. The power method is the favorite algorithm for the PageRank problem [6], while the Householder method is often used for linear equations. Accordingly, the power method for the PageRank and the Householder for HermitianStatus were used herein.

4.1. PageRank Reproductivity

Figure 4 shows the PageRank result at a damping factor value of 0.85 for a 60-node network. In this network, the strong orange color represents a high score, while the strong blue ones depict a low score. The Spearman's correlation coefficient of HermitianStatus compared to PageRank scores showed a high value of 0.9317358 achieved at the parameter values of k_1 = 10, k_2 = 7, k_3 = 1, and k_4 = 0.5. Figure 5 shows the scatter plot of the PageRank and HermitianStatus scores.

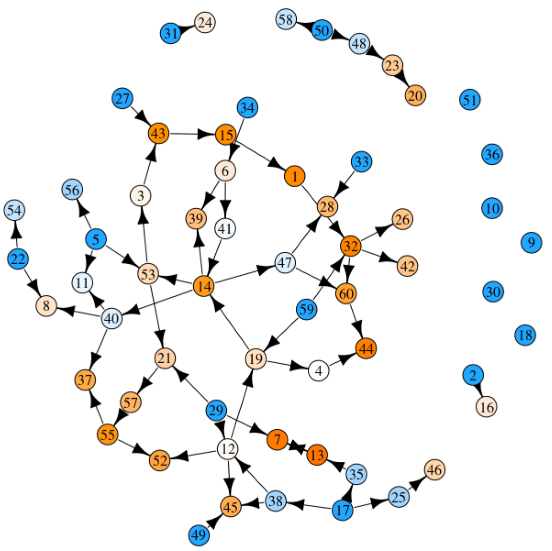

Figure 4: 60 node network with its PageRank result for the nodes (blue: low score and orange: high score).

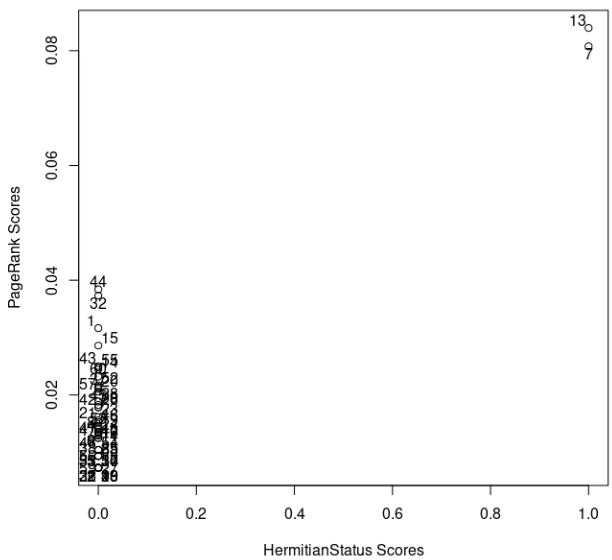

Figure 5: Scatter Plot of the PageRank scores and (standard) HermitianStatus scores of the nodes in the network in Figure 2.

4.2. Calculation Cost Efficiency

Ten networks were randomly created. Figure 6 illustrates the node numbers of these graphs. A probability value of 0.25 was used for the link between every node in each graph. The node calculation times for each network by PageRank and HermitianStatus were then compared.

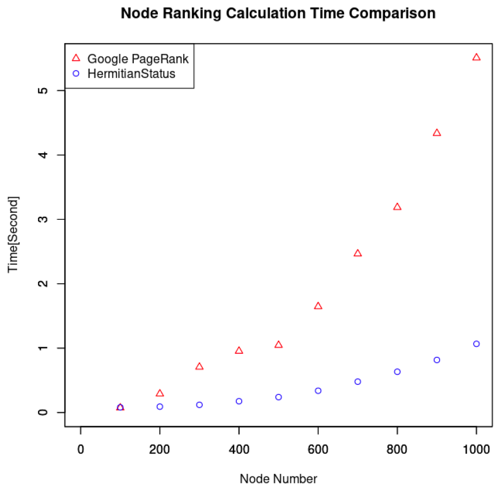

Figure 6: Comparison between PageRank and HermitianStatus.

5. Final Remarks

HermitianStatus reproduced the PageRank result well. Moreover, it achieved efficient calculations over PageRank. The future research will include showing that these points stand for larger networks. With large networks, showing systematical parameter changes leads to desirable results.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 20K11856.

This research was supported by Nanzan University Pache Research Subsidy I-A-2 for the 2020 academic year.

This paper includes a world wide patent application: PCT/JP2020/045268. This paper includes a patent in Japan: JP6502592.

References

[1] P. Berkhin "A Survey on PageRank Computing." *In Internet Mathematics* 2 (2005), 173-120.

[2] K. Sugihara ”Using Complex Numbers in Website Ranking Calculations: A Non-ad hoc Alternative to Google's PageRank" *Journal of Software*(14) (2019), 58-64.

[3] A. Langillw and C. D. Meyer *Google's Page Rank and beyond THE SCIENCE OF SEARCH ENGINE RANKINGS.* Princeton University Press (2006).

[4] K. J. Guo *Simple eigenvalues of graphs and digraphs.* Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in the Department of Mathematics of the Faculty of Science of Simon Fraser University (2015).
[5] D. Eddelbuttel *Seamless R and C++ Integration with Rcpp,* Springer (2013).
[6] S. Osborne, J. Rebaza, and E. Wiggins "On Accelerating the PageRank Computation." *Internet Mathematics*(6)*,* (2009), 157-171.