UNIVERSITY OF LEEDS

The Perron-Frobenius Theorem and the Pagerank Algorithm From a Mathematical Perspective

Student: Bradley Neve - 201242617
Supervisor: Prof. Paul Martin
06/05/2022

The University of Leeds
School of Mathematics

Abstract

This paper considers the mathematics behind the Google Pagerank algorithm. It will initially serve as a learning resource for the mathematics required to fully understand the algorithm. There will be a large focus on the Perron-Frobenius theorem and its implications to Pagerank. This will represent a shift towards the underlying mathematics of the algorithm that allows it to work. It will also look into usage of the algorithm for another purpose - ranking NFL teams. Through statistical analysis we will aim to evaluate its usage as a predictor of future success of teams in the NFL.

Contents

1 Introduction

The Google search engine plays a huge part in almost every individuals life. Quickly after founding, Google became a leader amongst search engines and has remained so since, currently holding almost 92% of the search engine market share across the world [StatCounter, 2022]. So what sets it apart from other search engines? The Pagerank algorithm is the algorithm used to rank web-pages in their search engine. This algorithm and the originality of it is without doubt a large part of Google's great success.
This paper will attempt to provide a comprehensive mathematical discussion around the algorithm. Within this discussion we aim to fulfill three key purposes; to teach the required mathematics to understand HOW the algorithm works, to investigate the underlying mathematics of the algorithm i.e. WHY it works and to evaluate the algorithms usage for an alternative purpose.
Within this we will cover some advanced mathematical concepts. As such, it is recommended readers have some mathematical background. However, we will cover the teaching from a very basic level, so to not have particular expertise in the fields we will use will not be of particular detriment. The mathematics we use to explain the machinery of the algorithm in particular will be taught very much from the ground up.
The workings on why the algorithm works and the underlying mathematics assumes some basic understanding of linear algebra, specifically a rudimentary knowledge of matrices and eigenvalues. For those wishing for a better understanding of this, you may find useful the book 'Linear algebra: A modern introduction' by [Poole, 2014], in particular chapters 3 and 4.
In a 2008 article by [Govan et al., 2008] Govan et al proposed a generalised version of the PageRank algorithm as a method to rank NFL teams. In the last section of this paper we will present the methods proposed in this article. We will then attempt to further this work through computation, and subsequent analysis of the output, of their method, for a recent season in the NFL.
In this section, we will use some statistical methods to analyse the results. We will not teach the methods we use, however we will explain in detail what the results of these methods imply. Thus, an understanding of statistical methods is not required to appreciate the interpretation of the results. However, for those who wish to understand the methods used you may find useful the lecture notes on probability and statistics by [Aykroyd, 2018], and then the information on the web-page [Statistics, 2018].

2 Background Mathematics

2.1 Graph Theory

One of the foundations for the Pagerank Algorithm is the modelling of the internet as a graph. As such, to begin this paper, we will look at some of the key aspects of graph theory.
Across the literature, there exists many definitions of graphs [Tutte and Tutte, 2001], [Chen, 2012]. We shall use the one found in the University of Leeds lecture notes on Graph Theory [Gambino, 2020].

Definition 2.1.1 *(Graph)*

A ***graph****, G, is a pair of sets (V,E) such that $E \subseteq V^{\{2\}}$ (ie. for all $e \in E$ we have $e = \{v_i, v_j\}$ where $v_i, v_j \in V$).*

Notation and Nomenclature

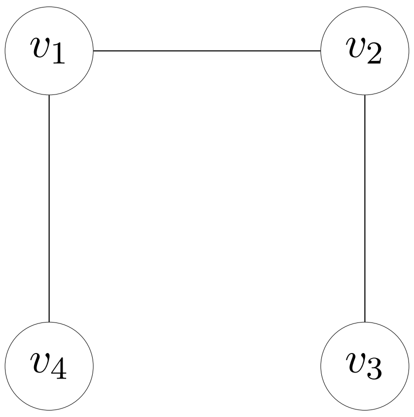
- We will use V(G) and E(G) to refer to the sets V and E that make up the graph, G.
- We will refer to the sets V(G) and E(G) as the sets of **vertices** and **edges** of G respectively. An element of V(G), v, is a vertex of G and an element of E(G), e, is an edge of G.
- An edge of a graph consists of two distinct vertices, which we call the **endpoints** of the edge.
- The edge containing the vertices, u,v, is represented by e = $\{u, v\}$, however for simplicity, we may call it e = uv.

Note: We will also refer to vertices as **nodes**.

Example
Let $G_{un} = (V, E)$ with;

$$V(G_{un}) = \{v_1, v_2, v_3, v_4\}, E(G_{un}) = \{v_1v_2, v_1v_4, v_2v3\}.$$

We can represent G_{un} pictorially as below.



Now we will introduce a concept to allow us to work algebraically with graphs. Again we will use a definition from Gambino's work.

Definition 2.1.2 *(Adjacency Matrices) [Gambino, 2020]*

*Let G be a graph with $\|V(G)\|= n$. Let each vertex be labeled v_i where $i \in \{1, ..., n\}$. The **adjacency matrix** A(G) is the n x n matrix with entries $A(G)_{ij}$ given by*

$$A(G)_{ij} = \begin{cases} 1 & \textit{if } v_i v_j \in E(G) \\ 0 & \textit{otherwise} \end{cases}$$

Example
Using the same example as above, we get;

$$A(G_{un}) = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Previously we have not considered the direction of edges, i.e. we have not considered any distinction between the edge uv and the vu. We will now introduce a type of graph which does indeed differeniate between the two

Definition 2.1.3 *(Directed Graphs) [Guichard, 2017]*

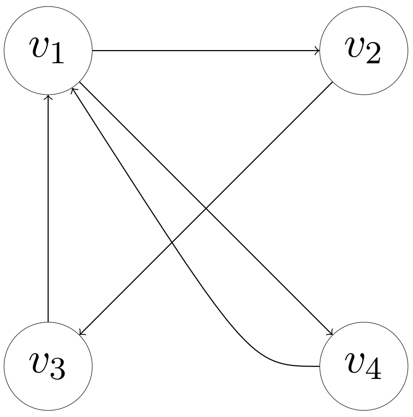
*A **Directed Graph**, G, is a pair of sets (V, E) where E(G) is a set of ordered pairs of the form (u,v) with u,v $\in V(G)$.*

This is a similar definition to that of a graph, however the word ordered separates the edge uv and vu.

Example
Let $G_{di} = (V, E)$ with;

$$V(G_{di}) = \{v_1, v_2, v_3, v_4\}, E(G_{di}) = \{v_1v_2, v_1v_4, v_2v_3, v_3v_1, v_4v_1\}.$$

This can be represented pictorially as below



Here we have arrows representing the direction of the edges, and we take the direction as going out of the first node in the pair and into the second node. The adjacency matrix of the above graph is given by;

$$A(G_{un}) = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Definition 2.1.4 *(Path) [Gambino, 2020]*

Let G = (V,E) be a graph. A path in G is a sequence of distinct vertices $v_0v_1...v_n$ such that $v_iv_{i+1} \in E$ for every $0 \leq i < n$.

Definition 2.1.5 *(Strongly connected) [Guichard, 2017]*

A directed graph, (V,E) is strongly connected if for all pairs of vertices, v,u $\in$ E there is a path from v to u and a path from u to v.

Example Let us consider two simple examples,

(i) Let G_1 be the graph represented below,

$$\begin{tikzcd}
v_0 \arrow[r] & v_1 \arrow[d] \\
v_3 \arrow[u] & v_2 \arrow[l]
\end{tikzcd}$$

Here we have an example of a strongly connected graph. Let v_i, v_j be any two distinct vertices of G_1, there is always a path from v_i to v_j given by $v_i v_{(i+1)mod4}...v_j$. These vertices are general, so there also exists a path from v_j to v_i. As there is a path from any vertex v_i to any vertex v_j and back, G_1 is strongly connected.

(ii)

$$\begin{tikzcd}
v_0 \arrow[r] & v_1 \arrow[d] \\
v_3 & v_2 \arrow[l]
\end{tikzcd}$$

Here we see the vertex v_3 has no outgoing nodes. Thus, there is no path from v_3 to any other node, and G_2 is not strongly connected.

Definition 2.1.6 *(Weighted Graph) [Weisstein, 2022c]*

*A **weighted** graph is a graph where the edges are assigned a numerical value, which we refer to as a weight.*

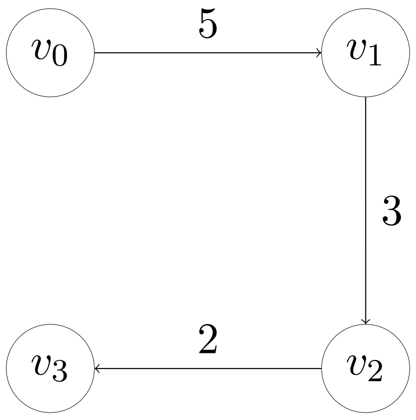
Let e be an edge belonging to a weighted graph. We will use w(e) to refer the weighting of e. Pictorially, we will label the edge with the weighting.

Definition 2.1.7 *(Cost Matrix)*

Similar to an adjacency matrix for unweighted graphs, we can represent a weighted graph in matrix form using what we will call a ***cost matrix****.*
Let G be a weighted graph with $\|V(G)\| = n$*. Let each vertex be labeled* v_i *where* $i \in \{1, ..., n\}$*. The cost matrix C(G) is the n x n matrix with entries* $C(G)_{ij}$ *given by*

$$C(G)_{ij} = \begin{cases} w(v_i v_j) & \text{if } v_i v_j \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

Example
Let G = (V, E) be the graph represented below



Here we see V(G) = $\{v_0, v_1, v_2, v_3\}$ and E(G) = $\{v_0v_1, v_1v_2, v_2, v_3\}$. These edges have the corresponding weights;

$$w(v_0v_1) = 5$$
$$w(v_1v_2) = 3$$
$$w(v_2v_3) = 2$$

We can then represent G with the following cost matrix

$$C(G) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \end{bmatrix}$$

2.2 Perron-Frobenius Theorem

2.2.1 Intro

There is a multitude of versions of the Perron-Frobenius theorem across the literature. The different forms all have the same idea, stating the existence and uniqueness of the largest real eigenvalue of a specific matrix type. However, they often differ in the type of matrix they state this for and as such some properties of this eigenvalue change. We will look at some of these forms and decide which are applicable to a model of the internet.

Definition 2.2.2 *(Types of matrices)*

- ***Non-negative matrix*** *- A matrix with all non-negative entries.*
- ***Positive matrix*** *- A matrix with all positive entries.*
- ***Square matrix*** *- A matrix with the same number as rows as columns - we will only deal with matrices of this kind in this paper, as such shall omit reference in future.*

Definition 2.2.3 *(Identity matrix) adapted from [Weisstein, 2022a]*

The ***identity matrix*** *of size n is the n x n matrix with 1s on the diagonal entries and 0s elsewhere.*

Example

The 3 x 3 identity matrix is given by;

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

The next two definitions have been adapted from [Poole, 2014]

Definition 2.2.4 *(Characteristic Polynomial)*

For a square matrix, A, the ***characteristic polynomial*** *is the equation given by the left hand side of;*

$$det(A\text{-}I\lambda) = 0, \text{ where } I \text{ is the identity matrix.}$$

The roots of this equation are given by the eigenvalues of the matrix A.

Definition 2.2.5 *(Multiplicity) [Poole, 2014]*

Let λ be an eigenvalue of the matrix A. The multiplicity of λ is given by the number of times it appears in the characteristic polynomial of A.

Example
Let,

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

Then,

$$A - I\lambda = \begin{bmatrix} 1-\lambda & 2 & 3 \\ 6 & 4-\lambda & 2 \\ 1 & 2 & 3-\lambda \end{bmatrix}$$

The determinant of this is given by det(A-Iλ) $= \lambda^2(8-\lambda)$.
Setting equal to zero, we get $\lambda^2(8-\lambda) = 0$. We see this equation has two roots, $\lambda_1 = 8$ and $\lambda_2 = 0$.
The root 8 appears only once in the characteristic polynomial, whereas the root 0 appears twice, and so we have $\lambda_1 = 8$ has multiplicity 1 and $\lambda_2 = 0$ has multiplicity 2.

As mentioned, the Perron-Frobenius theorem appears in many forms over the literature. The versions and forms we give next are taken from a variety of sources and combined/adapted to contain the information we will require later. [Higham, 2021] [Armstrong, 2012] [Chang et al., 2008] [Ninio, 1976]

Theorem 2.2.6 *(Perron-Frobenius):*
Let A be a positive, square matrix, and let $\rho(A)$ be the absolute value of the largest eigenvalue of A. Then,

(i) $\rho(A)$ is an eigenvalue of A.

(ii) $\rho(A)$ is greater than zero.

(iii) There exists an eigenvector, x, with $x > 0$, (i.e. each entry of x is positive) such that $Ax = \rho(A)x$

(iv) $\rho(A)$ has multiplicity 1

*We may refer to $\rho(A)$ as the **Perron root** or the **dominant eigenvalue**. We will refer to its corresponding eigenvector as the **dominant eigenvector**.*

Example
Let,

$$G_{po} = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

Here we have a positive matrix. We will check the Perron-Frobenius holds for this matrix.

From the example in **2.2.5**, we know this matrix has 2 eigenvalues, $\lambda_1 = 8$ and $\lambda_2 = 0$.

We see the absolute value of the largest eigenvalue, $\rho(\mathrm{G}_{po}) = |\lambda_1| = 8$. Now let us check the rules of **2.2.6**;

(i) $\rho(\mathrm{G}_{po}) = 8 = \lambda_1$, so (i) is satisfied.

(ii) $\rho(\mathrm{G}_{po}) = 8 > 0$ so (ii) is satisfied.

(iii) To verify (iii) we solve $\mathrm{G}_{po} \cdot x = 8 \cdot x$, or $(\mathrm{G}_{po} - 8I)x = 0$, where I is the identity matrix.
Solving this, we find $x = (1, 2, 1)^T$ and so we find x exists, and is positive. Thus (iii) is satisfied.

(iv) From the example in **2.2.5**, we know $\lambda_1 = 8$ has multiplicity 1. So, (iv) holds.

Thus, the theorem holds for our example.
We will omit a general proof of the Perron-Frobenius theorem in this paper, and instead focus on its implications to the PageRank algorithm. However for those interested you can find a short proof in the book "Potts Models And Related Problems In Statistical Mechanics" by Paul Purdon Martin.
As previously mentioned there exist similar theorems for other types of matrices. We may want to consider the case were the matrix has some zero entries. Would the theorem hold? We will evaluate a non-negative matrix and see if the theorem holds.

Example

Let,

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

We have,

$$G - I\lambda = \begin{bmatrix} -\lambda & 0 \\ 1 & -\lambda \end{bmatrix}$$

Now, let us solve det(G-Iλ) = 0 to find the eigenvalues of G.

$$det(G - I\lambda) = -\lambda \cdot -\lambda - 1 \cdot 0 = \lambda^2$$

Setting equal to zero, we have $\lambda^2 = 0$.
Thus, we see G has one eigenvalue, $\lambda = 0$, with multiplicity 2. This violates rule (ii) and (iv) from (**2.2.6**) and so the theorem doesn't hold.
However, we do see rule (i) is satisfied, and in fact, as is rule (iii), with eigenvector v= $(0,1)^T$.

These two rules hold for all non-negative matrices, giving us the following, weaker version of the theorem for non-negative matrices.

Theorem 2.2.7 *(Perron-Frobenius - non-negative version)*
Let A be a non-negative, square matrix, and let $\rho(A)$ be the absolute value of the largest eigenvalue of A. Then,

(i) $\rho(A)$ *is an eigenvalue of* A.

(ii) There exists an eigenvector, x, *with* $x > 0$ *such that* $Ax = \rho(A)x$

Now, there exist some useful extensions of these theorems for positive or non-negative matrices with some specific properties. First let us define these properties.

Definition 2.2.8 *(Irreducible Matrix) [Lewis, 2013]*

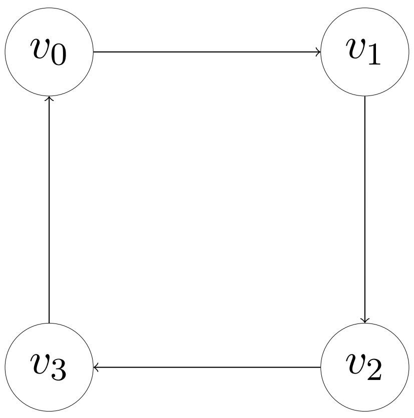
Across different sources, an irreducible matrix is defined in a multitude of ways. For our purposes, the specific definition is not of utmost importance, and we will be better suited with a focus on a feature which implies irreducible.

A square matrix G is ***irreducible*** *if and only if the associated directed graph is strongly connected.*

Example
Consider the matrix,

$$G_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

This matrix is represented graphically as below,

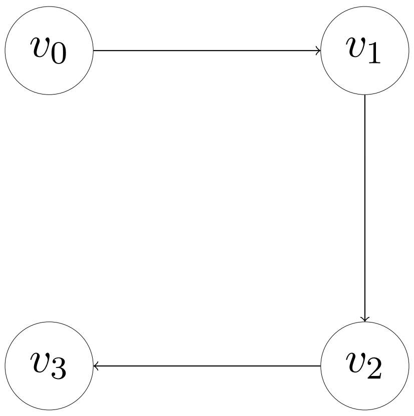


From the example in **2.1.5** we know this graph is strongly connected. Thus, we have the matrix G_1 is irreducible.

Now let us consider the matrix,

$$G_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

This matrix is represented graphically as below,



This is another example from **2.1.5**. We know this graph is not strongly connected. Thus we have the matrix G_2 is reducible.

Definition 2.2.9 *(Stochastic Matrix) [Weisstein, 2022b]*

Let A be a square, non-negative matrix with the entries for each column summing to 1. Then, we say A is a ***stochastic matrix****.*

We can can give a more complete version of the theorem for non-negative matrices if they are also irreducible.

Theorem 2.2.10 *(Irreducible extension of non-negative Perron-Frobenius): Let A be a non-negative, irreducible matrix, and let $\rho(A)$ be the absolute value of the largest eigenvalue of A. Then,*

(i) $\rho(A)$ is an eigenvalue of A.

(ii) $\rho(A)$ is greater than zero.

(iii) There exists an eigenvector, x, with $x > 0$ such that $Ax = \rho(A)x$.

(iv) $\rho(A)$ has multiplicity 1.

Lemma 2.2.11 *(Stochastic extension)*
Let A be a stochastic, irreducible matrix, and let $\rho(A)$ be the absolute value of the largest eigenvalue of A. Then, the properties from ***2.2.7*** *hold and,*

(i) $\rho(A) = 1$.

(ii) *The sum of the entries of the eigenvector, x, corresponding to $\rho(A)$ is 1.*

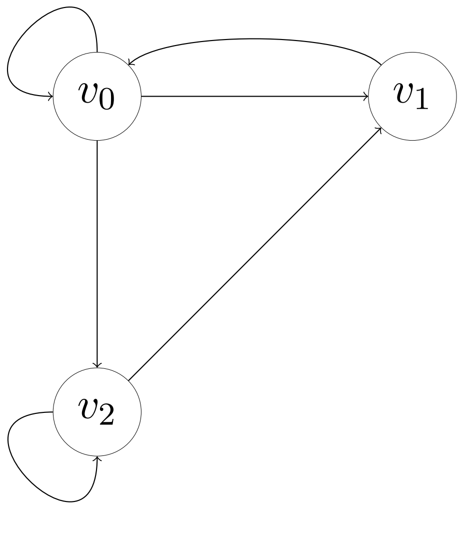
Example

Let,

$$G_{st} = \begin{bmatrix} \frac{1}{3} & 1 & 0 \\ \frac{1}{3} & 0 & 0.5 \\ \frac{1}{3} & 0 & 0.5 \end{bmatrix}$$

We clearly see the sums of all columns sum to 1, so G$_{st}$ is stochastic. Next we check it is irreducible.

Let us consider the graph of G$_{st}$.



We see this graph is strongly connected, for similar reasons to our example in **2.1.5**. Thus, we know G$_{st}$ is irreducible, and we can apply **Lemma 2.2.11**. Now let us find the eigenvalues, λ_i and corresponding eigenvectors, v$_i$. To do this we first find G$_{st}$ - Iλ,

$$G_{st} - I\lambda = \begin{bmatrix} \frac{1}{3} - \lambda & 1 & 0 \\ \frac{1}{3} & -\lambda & 0.5 \\ \frac{1}{3} & 0 & 0.5 - \lambda \end{bmatrix}$$

Then, we consider det($G_{st} - I\lambda$). This is given by,

$$det(G_{st} - I\lambda) = (\frac{1}{3} - \lambda)(-\lambda(0.5 - \lambda) - (\frac{1}{3}(0.5 - \lambda) - \frac{1}{3} \cdot \frac{1}{2})$$

This rearranges to,

$$det(G_{st} - I\lambda) = -\lambda(\lambda - 1)(\lambda + \frac{1}{6})$$

Setting equal to zero and solving, we find λ_1 = 1, λ_2 = -1/6 or λ_3 = 0

We see we have the absolute value of the largest eigenvalue, $\rho(G_{st}) = |\lambda_1| = 1$. We also will need the corresponding eigenvector, v_1 of the dominant eigenvalue. So, let us solve;

$$(G_{st} - I\lambda_1)v_1 = (G_{st} - I)v_1 = 0$$

We have,

$$(G_{st} - I)v_1 = \begin{bmatrix} -\frac{2}{3} & 1 & 0 \\ \frac{1}{3} & -1 & 0.5 \\ \frac{1}{3} & 0 & -0.5 \end{bmatrix} (v_{1_i}, v_{1_j}, v_{1_k})^T$$

Setting to zero, we have,

$$\begin{bmatrix} -\frac{2}{3} & 1 & 0 \\ \frac{1}{3} & -1 & 0.5 \\ \frac{1}{3} & 0 & -0.5 \end{bmatrix} (v_{1_i}, v_{1_j}, v_{1_k})^T = 0$$

Then, using matrix multiplication we get,

$$(-\frac{2}{3}v_{1_i} + v_{1_j}, \frac{1}{3}v_{1_i} - v_{1_j} + 0.5v_{1_k}, \frac{1}{3}v_{1_i} - 0.5v_{1_k})^T = 0$$

This gives us simultaneous equations;

$$-\frac{2}{3}v_{1_i} + v_{1_j} = 0,$$
$$\frac{1}{3}v_{1_i} - v_{1_j} + 0.5v_{1_k} = 0,$$
$$\frac{1}{3}v_{1_i} - 0.5v_{1_k} = 0$$

Solving these we have $v_{1_i} = \frac{3}{2}v_{1_j} = \frac{3}{2}\ v_{1_k}$, thus we have the stochastic vector $v_1 = (\frac{3}{7}, \frac{2}{7}, \frac{2}{7})$.
Now let us check the rules of **2.2.10**;

(i) $\rho(\mathrm{G}_{st}) = 1 = \lambda_1$, so (i) is satisfied.

(ii) $\rho(\mathrm{G}_{st}) = 1 > 0$ so (ii) is satisfied.

(iii) We see x exists with x = v_1. $\mathrm{v}_1 > 0$, and so (iii) is satisfied.

(iv) We know G_{st} has characteristic polynomial $-\lambda(\lambda - 1)(\lambda + \frac{1}{6})$, so we see that $\rho(\mathrm{G}_{st})$ has multiplicity 1, and so (iv) is satisfied.

We see our example abides by the irreducible extension, so let us now check the stochastic extension, **2.2.11**.

(i) $\rho(\mathrm{G}_{st}) = 1$ as required.

(ii) $\mathrm{v}_1 = (\frac{3}{7}, \frac{2}{7}, \frac{2}{7})$. We see the sum of these entries is indeed 1, and so (ii) is satisfied.

Thus we see the stochastic extension also holds for our example. Again we will now consider a proof for general cases.

2.3 The Power Method

We have spoken about some features of the dominant eigenvalue and eigenvector, so we have an idea these may be useful. The power method is an iterative procedure which is used to find the dominant eigenvalue and eigenvector (if one exists). The following theorems and proof are adapted from the paper "On the mathematical background of Google PageRank algorithm" [Peretti and Roveda, 2014].
It begins with an arbitrary estimate of the dominant eigenvector, and uses an iterative formula to produce a sequence that converges to the true value.

For a square, N x N matrix, A, iterations of the power method are calculated as follows;

$$\begin{aligned} x_1 &= \frac{Ax_0}{||Ax_0||} \\ x_2 &= \frac{Ax_1}{||Ax_1||} \\ \vdots &= \vdots \\ x_i &= \frac{Ax_{i-1}}{||Ax_{i-1}||} \end{aligned}$$

So in each stage, we multiply our matrix A by the previous iteration, and normalise the resulting vector. This process yields some interesting results.

Theorem 2.3.1 *(The Power Method)*
Let A be a square $N \times N$ matrix, with dominant eigenvector v_1 and corresponding eigenvalue λ_1 and n linearly independent eigenvectors, v_1,v_2,...,v_n. From the sequence given by;

$$x_i = \frac{Ax_{i-1}}{||Ax_{i-1}||}$$

we can then calculate the dominant eigenvector and eigenvalue as follows;

$$\begin{aligned} v_1 &= \lim_{i\to\infty} x_i \\ \lambda_1 &= \lim_{i\to\infty} ||Ax_i|| \end{aligned}$$

Example
We will consider our example from **2.2.6**.

Let,

$$G_{po} = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

From **2.2.6**, we know this matrix has dominant eigenvector (1, 2, 1)^T, with corresponding eigenvalue 8.
We will now apply the power method to G_{po} and see the outcome.

Let x_0 = (1, 1, 1)^T then we have,

$$x_1 = \frac{G_{po}x_0}{||G_{po}x_0||}$$

Breaking this down we have,

$$\begin{aligned} G_{po}x_0 &= \begin{bmatrix} 1 & 2 & 3 \\ 6 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix} (1,1,1)^T \\ &= \begin{pmatrix} 6 \\ 12 \\ 6 \end{pmatrix} \end{aligned}$$

and,

$$\begin{aligned} ||G_{po}x_0|| &= ||(6,12,6)^T|| \\ &= \sqrt{6^2 + 12^2 + 6^2} \\ &= \sqrt{216} \end{aligned}$$

So, after one iteration we get;

$$x_1 = \frac{1}{\sqrt{216}} \begin{pmatrix} 6 \\ 12 \\ 6 \end{pmatrix}$$

Next we have,

$$x_2 = \frac{G_{po}x_1}{||G_{po}x_1||}$$

$$\begin{aligned} G_{po}x_1 &= \begin{bmatrix} 1 & 2 & 3 \\ 6 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix} \frac{1}{\sqrt{216}} \begin{pmatrix} 6 \\ 12 \\ 6 \end{pmatrix} \\ &= \begin{pmatrix} 48 \\ 96 \\ 48 \end{pmatrix} \frac{1}{\sqrt{216}} \end{aligned}$$

Giving

$$\begin{aligned} ||G_{po}x_1|| &= || \begin{pmatrix} 48 \\ 96 \\ 48 \end{pmatrix} \frac{1}{\sqrt{216}} || \\ &= \sqrt{\frac{48^2 + 96^2 + 48^2}{216}} \\ &= \sqrt{64} \\ &= 8 \end{aligned}$$

This gives

$$\begin{aligned} x_2 &= \begin{pmatrix} 48 \\ 96 \\ 48 \end{pmatrix} \frac{1}{8} \frac{1}{\sqrt{216}} \\ &= \frac{1}{\sqrt{216}} \begin{pmatrix} 6 \\ 12 \\ 6 \end{pmatrix} \\ &= x_1 \end{aligned}$$

So we see in this example we have convergence after 1 iteration, as $x_1 = x_2$ and so each iteration past x_1 will equal x_1.

Thus we see from this method that

$$\begin{aligned}\lim_{i\to\infty} x_i &= x_1\\ &= \frac{1}{\sqrt{216}}\begin{pmatrix}6\\12\\6\end{pmatrix}\\ &= v_1\end{aligned}$$

Which can be divided through by $\frac{6}{\sqrt{216}}$ to give

$$v_1 = (1,2,1)^T$$

as required. We also have

$$\begin{aligned}\lim_{i\to\infty} ||G_{po}x_i|| &= ||G_{po}x_1||\\ &= 8\end{aligned}$$

as required.

Proof
As v_1,v_2,...,v_n are linearly independent eigenvectors, we can write;

$$x_0 = c_1v_1 + c_2v_2 + ... + c_nv_n$$

where $c_1c_2, ..., c_n$ are some scalar constants. We can multiply both sides by A^i to get;

$$A^ix_0 = A^i(c_1v_1 + c_2v_2 + ... + c_nv_n)$$

then, as v_1,v_2,...,v_n are eigenvectors of A, we have $A^iv_k = \lambda_k^iv_k$ for k = 1,2,...,n
Thus

$$\begin{aligned}A^ix_0 &= A^i(c_1v_1 + c_2v_2 + ... + c_nv_n)\\ &= c_1\lambda_1^iv_1 + c_2\lambda_2^iv_2 + ... + c_n\lambda_n^iv_n\end{aligned}$$

Then, taking out a factor of λ_1^i, we get

$$A^ix_0 = \lambda_1^i(c_1v_1 + c_2\frac{\lambda_2^i}{\lambda_1^i}v_2 + ... + c_n\frac{\lambda_n^i}{\lambda_1^i}v_n)$$

As λ_1 is our dominant eigenvalue, we have $\lambda_1 > \lambda_k$ for k = 2,...,n. Thus as $\frac{\lambda_2^i}{\lambda_1^i} = \frac{\lambda_2}{\lambda_1}^i$ we have as $i \to \infty$

$$A^ix_0 = x_i = \lambda_1^ic_1v_1$$

So we have as $i \to \infty$, x_i is a scalar multiple of the v_1. As scaling of eigenvectors is unimportant we have x_i, as i $\to \infty$ is a dominant eigenvector of A. As normalisation of a vector only multiplies it by a scalar we can say;

$$\frac{Ax_i}{||Ax_i||} = \alpha x_i = \alpha \lambda_1^i c_1 v_1$$

where α is some scalar constant.

Thus, once more we can say $\frac{Ax_i}{||Ax_i||}$, as $i \to \infty$ is a dominant eigenvector of A.
Furthermore, as $x_i = x_{i+1}$ as $i \to \infty$, we have

$$\begin{aligned} x_i &= x_{i+1} \\ &= \frac{Ax_i}{||Ax_i||} \\ &= \frac{\lambda_1 x_i}{||Ax_i||} \\ &\Rightarrow \lambda_1 = ||Ax_i|| \text{ as } i \to \infty. \end{aligned}$$

As required.
As we saw in the proof, it is possible to find the dominant eigenvector prior to the normalisation steps, and we can use the following result.

Lemma 2.3.2 *(Lesser Result)*
Let A be a square $N \times N$ matrix, with dominant eigenvector v_1 and corresponding eigenvalue λ_1 and n linearly independent eigenvectors, $v_1, v_2, ..., v_n$. From the sequence given by;

$$x_i = Ax_{i-1}$$

we can then calculate the dominant eigenvector and eigenvalue as follows;

$$v_1 = \lim_{i \to \infty} x_i$$

2.4 Some Probability Theory

Finally we will look at some elements of probability theory, in particular we will look at Markov Chains. We aim to understand a specific example of a Markov Chain that is used in the formulation of the Pagerank Algorithm. The following two definitions are adapted from Statistics Glossary v1.1 [Easton and McColl, 1997].

Definition 2.4.1 *(Random Variable)*

*A **random variable** is a variable who's numerical value is determined by the outcome of some random event, e.g the outcome of a dice roll.*

Random variables can be discrete or continuous. We will consider solely **discrete** random variables, ie. random variables with a finite set of outcomes.

Definition 2.4.2 *(Stochastic Process)*

*A **stochastic process** is defined as a collection of random variables, X=$\{X_t : t \in T\}$ defined on a common probability space, taking values in a common set S (the state space), and indexed by a set T, thought of as time.*

We will again consider **discrete** stochastic processes, ie. processes where T is a countable set, e.g T={0,1,2,3...}, and the set S is finite.

Definition 2.4.3 *(Markov Chains) [Coolen, 2009]*

*A **Markov chain** is a type of stochastic process that abides the following probabilistic rules;*

- *A discrete state space (finite S).*
- *A discrete time space (countable T). This part can vary across the literature, however we will use this definition.*
- *The system abides by the 'Markov Property'.*

*The **Markov property** states that the probability of future events depend solely on the current state of the process. This means the process is memoryless - past states have no impact on the likelihood of future states.*

We can write this mathematically as;

$$P(X_{n+1}|X_0, ...X_n) = P(X_{n+1}|X_n)$$

Example
We will first consider a very simple example of a Markov chain.

Imagine a number line from -5 to 5. Consider a situation where you begin at 0 and each 'turn' you have 50% chance of going up 1 and 50% of going down 1.
At 5, there is a 100% chance of going down to 4 and at -5 there is a 100% chance of going up to -4.
Let X_t be the your location on the tth turn.
Here we have a stochastic process, with T = {0,1,2,...}, S= $\{k \in \mathbb{Z}$: -5 $< k <$ 5$\}$ and X=$\{X_t : t \in T\}$. We know $X_0 = 0$.

Consider the case where we have;

$$\begin{aligned}X_0 &= 0\\X_1 &= 1\\X_2 &= 2\\X_3 &= 3\\X_4 &= 4\\X_5 &= 5\end{aligned}$$

So, what will happen on the next turn? We know at turn 5, we are at 5, and so turn 6 has a 100% chance of being 4, or $\mathrm{P}(X_6 = 4) = 1$. Now, consider the case;

$$\begin{aligned}X_0 &= 0\\X_1 &= 1\\X_2 &= 0\\X_3 &= 1\\X_4 &= 2\\X_5 &= 3\\X_6 &= 4\\X_7 &= 5\end{aligned}$$

Here we have a different path to 5, so what will happen here? The next turn still has a 100% chance of being 4. We see this as the probabilities of the next turn are described solely by the current turn. This is what we mean by the Markov property, in that only the current state impacts the probabilities of the future state. Thus we see an example of a Markov chain.

Definition 2.4.4 *(Transition Matrix) [Wang and Swendsen, 2002]*

A transition matrix of a Markov chain is a matrix that describes the transitions between states in the chain.

Assume we have a Markov chain with n possible states, indexed 1 to n. Then the transition matrix of the Markov chain is the n x n matrix given by;

$$P = (p_{ij})$$

where p_{ij} is the probability of a transition from state j to state i.

Example
We will consider the same example we used for a Markov chain. In this example we had S= $\{k \in \mathbb{Z} :$ -5 $< k <$ 5$\}$. We will call -5 state 1, -4 state 2 and so on. For $1 < j < 11$ we have;

$$p_{ij} = \begin{cases} \frac{1}{2} & \text{if j} \in \{i-1, i+1\} \\ 0 & \text{otherwise} \end{cases}$$

and for j $\in$ {1,11} we have;

$$p_{ij} = \begin{cases} 1 & \text{if } |j| = |i+1| \\ 0 & \text{otherwise} \end{cases}$$

Thus we have the transition matrix as below

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

2.4.5 Random Walks on a Graph

As previously stated, we will look at an example of type of Markov chain that is used in the formulation of the Pagerank algorithm.

The kind of Markov chain we will consider is that of random walks on graphs. In particular, we will look at random walks on directed graphs. The information in this section is taken from [Lawler and Limic, 2010] and [Coolen, 2009].

Definition 2.4.6 *(Random Walk on a Directed Graph)*

Let G = (V,E) be a directed graph.
A ***random walk*** *on G is a sequence of vertices, V_n, with $v_n \in V$ generated from a start vertex, v_1 by randomly selecting an edge of the form (v_1, v_2). v_2 is then the next vertex in the sequence, and the process is repeated from v_2.*

We must consider two possibilities;

(a) G is unweighted.

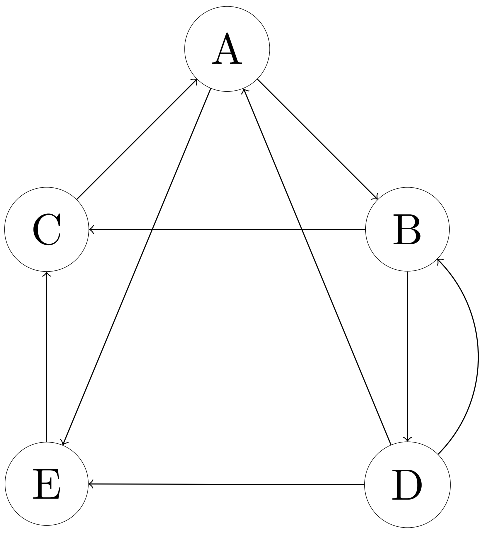
(b) G is weighted.

In the case where G is unweighted, each traversable edge is selected with equal probability. Thus this probability is $\frac{1}{N}$, where N is the number of traversable edges.
In the case where G IS weighted, each traversable edge is selected with probability proportional to its weighting.
Under these circumstances the probability of traversing an edge (v_n, v_{n+1}) is given by the weighting of (v_n, v_{n+1}) over the sum of the weightings of all edges leaving v_n or;

$$P(v_{n+1}|v_n) = \frac{w((v_n, v_{n+1}))}{\sum_{i=1}^{M} w((v_n, a_i))}$$

where M is the total number of vertices in G, and $a_i \in V$.

Example
First let us consider an example of an undirected graph.
Let G = (V,E) be the directed, unweighted graph represented by;



We will now consider the situation where each node is the current state of a random walk on G.

- At A, we can traverse either the edge AB or AE with probability $\frac{1}{2}$.
- At B, we can traverse BC or BD with probability $\frac{1}{2}$.
- At C, we must traverse CA with probability 1.
- At D, we can traverse DA, DB or DE, with probability $\frac{1}{3}$.

- At E, we must traverse EC with probability 1.

Now we have the transition probabilities for each possible state, we can create a transition matrix to represent a random walk on G. This is given by,

$$P = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{3} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 1 \\ 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} & 0 \end{bmatrix}$$

If we consider the adjacency matrix of G, shown below;

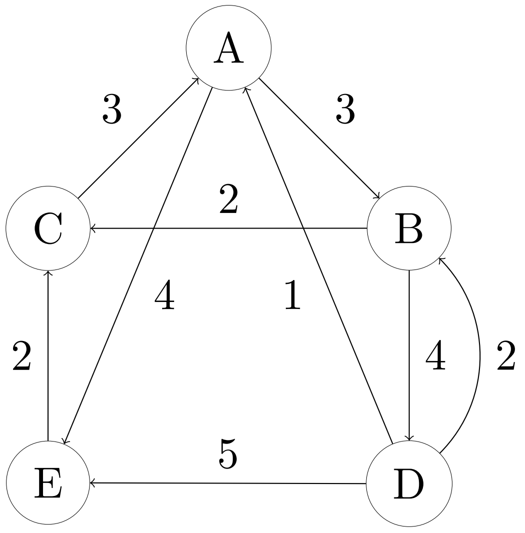
$$A = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

we see that the transition matrix of G can be found by dividing each column by the number of non 0 entries in the column.

Formally;

$$P_{ij} = \frac{A_{ij}}{\sum_{i=1} A_{ij}}$$

Now, let us consider a similar example, however adding weights to the edges of G.
Let G* = (V,E) be the directed, weighted graph represented by;



Now, we can similarly consider the situation where each node is the current state of a random walk on G*. However, we must now consider the weighting of the edges, with probabilities of traversing an edge given by,

$$P(v_{n+1}|v_n) = \frac{w((v_n, v_{n+1}))}{\sum_{i=1}^{M} w((v_n, a_i))}$$

Thus we have,

- At A, we can traverse the edge AB with probability $\frac{3}{3+4} = \frac{3}{7}$ or AE with probability $\frac{4}{3+4} = \frac{4}{7}$.
- At B, we can traverse BC with probability $\frac{2}{2+4} = \frac{2}{6}$ or BD with probability $\frac{4}{2+4} = \frac{4}{6}$.
- At C, we must traverse CA with probability 1.
- At D, we can traverse DA with probability $\frac{1}{1+2+5} = \frac{1}{8}$, DB with probability $\frac{2}{1+2+5} = \frac{2}{8}$ or DE, with probability $\frac{5}{1+2+5} = \frac{5}{8}$.
- At E, we must traverse EC with probability 1.

We can then create a transition matrix to represent a random walk on G*. This is given by,

$$P^* = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{8} & 0 \\ \frac{3}{7} & 0 & 0 & \frac{1}{4} & 0 \\ 0 & \frac{2}{6} & 0 & 0 & 1 \\ 0 & \frac{4}{6} & 0 & 0 & 0 \\ \frac{4}{7} & 0 & 0 & \frac{5}{8} & 0 \end{bmatrix}$$

Similarly to what we found for the adjacency matrix of G, if we consider the cost matrix for G*, we can form a relationship between the cost matrix of a weighted graph and the transition matrix.

$$C = \begin{bmatrix} 0 & 0 & 3 & 1 & 0 \\ 3 & 0 & 0 & 2 & 0 \\ 0 & 2 & 0 & 0 & 2 \\ 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \end{bmatrix}$$

We see that,

$$P_{ij} = \frac{C_{ij}}{\sum_{i=1} C_{ij}}$$

Definition 2.4.7 *(Probability Distribution Vector) [Easton and McColl, 1997]*

Let ρ_k be the row vector with entries given by;

$$\rho_{k_i} = P(v_k = a_i) \textit{ where } a_i \in V$$

Here we see the entries of ρ_k represent the probability distribution of v_k and we call ρ_k the ***probability distribution vector*** *of v_k.*

Let us now consider ρ_{k+1}. We have;

$$\begin{aligned} \rho_{k+1_i} &= P(v_{k+1} = a_i) \text{ where } a_i \in V \\ &= \sum_{a_j \to a_i} P(v_{k+1} = a_i | v_k = a_j) P(v_k = a_j) \\ &= \sum_{a_j \to a_i} p_{ij} \rho_{k_j} \end{aligned}$$

where $\sum_{a_j \to a_i}$ means the sum over the vertices a_j that have are adjacent to a_i. We see this is equivalent to the result of matrix multiplication between ρ_k and the transition matrix of the graph over which our random walk is taking place. Thus we have the following result.

Theorem 2.4.8 *(Reccurence)*
Let G = (V,E) be a graph, and V_k be a random walk on G. The probability distribution vector of the stages of a random walk forms a recurrence relation, given by;

$$\begin{aligned} \rho_{k+1} &= P\rho_k \\ &= PP\rho_{k-1} \\ &= \vdots \\ &= P^{k+1}\rho_0 \end{aligned}$$

or

$$\rho_k = P^k \rho_0$$

3 Pagerank

In this section we will use the mathematics we have learnt until now to thoroughly explain what pagerank is, how it works and why it works. The information within this section has been collated and adapted from multiple sources. The sources primarily used are as follows; [Amine, 2020], [Rogers, 2002], [Langville and Meyer, 2004] and [Yan and Ding, 2011].

3.1 What is Pagerank

Pagerank is the name given to the algorithm used to rank web pages when a search occurs in the Google search engine.
The aim of the algorithm is to assign a numerical weight describing the importance of each web page. It does this through counting hyperlinks into a page and determine the quality of these links. The idea is an important page will have many incoming links, from other important pages.

3.2 Modelling of the Internet

Before we discuss the algorithm itself, we must first consider the mathematical model used to represent the internet.
For Google's Pagerank algorithm we will consider a graphical representation of the internet.

We will think of the internet as a directed graph with;

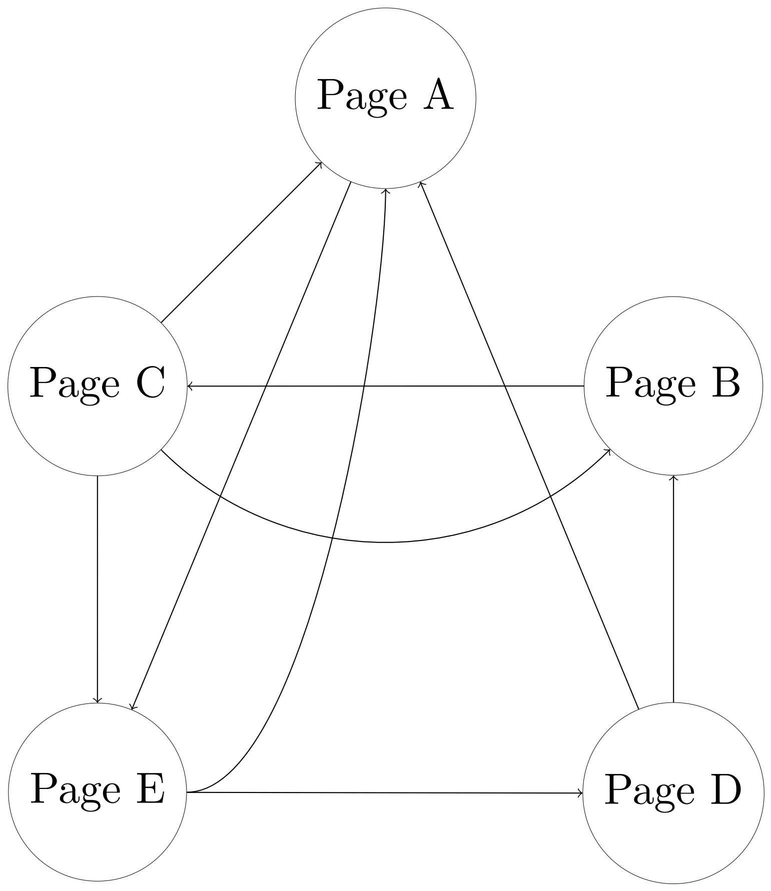
- Web pages represented by nodes.
- Hyperlinks from Page X to Page Y represented by an edge from X to Y. Mutliple links from one page to another are treated as one link. Links from a page to itself are ignored.

To clarify we will consider an example, using a small sample of the internet.

Example
Consider a sample of 5 web pages; Page A, Page B, Page C, Page D and Page E. These pages have hyperlinks given below.

Page	Links In	Links Out
A	C,D,E	E
B	C,D	C
C	B	A,B,E
D	E	A,B
E	A,C	A,D

This would be represented graphically as below:



Now we have a graphical representation of the internet, we can begin to use the graph theory and other mathematics we discussed earlier. For our example, we can consider the corresponding matrix to the graphical representation.

We see the graph has adjacency matrix given by

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

and thus, dividing through each column by the sums of itself we get the transition matrix

$$P = \begin{bmatrix} 0 & 0 & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} \\ 1 & 0 & \frac{1}{3} & 0 & 0 \end{bmatrix}$$

3.3 The Algorithm

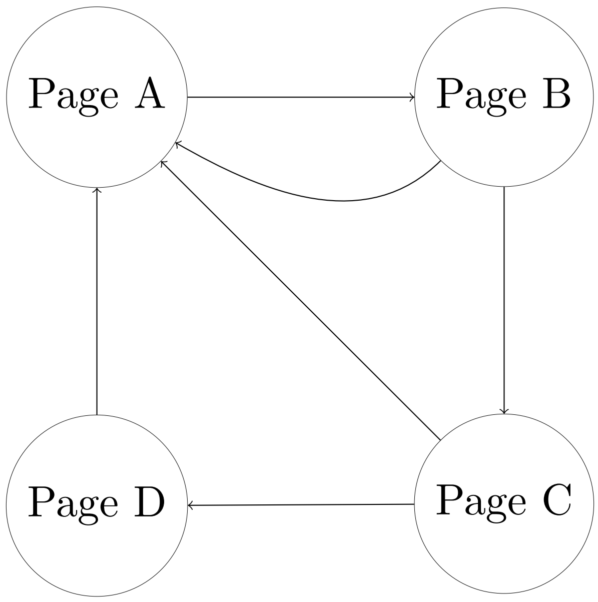
As previously discussed, the algorithm begins with a graphical modelling of the internet. We know we aim to provide a numerical ranking of each page, relating to the 'importance' of each page. In this subsection we will first present a basic version of the algorithm, primarily to unearth some issues that arise with this version. We will then present a complete version of the algorithm, with a more sophisticated mathematical thought process.

3.3.1 Basic/Simplified Idea

To show the steps of this idea, we will run the algorithm with a simple example.
Example 1
Consider a sample of 4 web pages; Page A, Page B, Page C and Page D. These pages have hyperlinks given below.

Page	Links In	Links Out
A	B, C, D	B
B	A	A, C
C	B	A, D
D	C	A

We can represent this graphically as



Earlier we stated "The idea is an important page will have many incoming links, from other important pages." Thus intuitively we may expect Page A to have the highest pagerank, due to it being the only page with multiple incoming links. Let us run the simplified algorithm to find out.

Step 1

We begin the algorithm by setting an initial pagerank value for each page. These are made equal, and to sum to 1. We will call the pagerank of Page i after k iterations $R_k(i)$. So we have;

$$R_0(A) = R_0(B) = R_0(C) = R_0(D) = \frac{1}{4}$$

Step 2

At each iteration, the current page rank of each page is shared out evenly amongst the pages it has an outgoing link to. So after one iteration of our example we have;

$$\begin{aligned}
R_1(A) &= \frac{R_0(B)}{2} + \frac{R_0(C)}{2} + R_0(D) = \frac{1}{2} \\
R_1(B) &= R_0(A) = \frac{1}{4} \\
R_1(C) &= \frac{R_0(B)}{2} = \frac{1}{8} \\
R_1(D) &= \frac{R_0(C)}{2} = \frac{1}{8}
\end{aligned}$$

Step 3

The third step (and fourth and fifth...) is to repeat iterations, until we have

$R_k(i) = R_{k+1}(i)$ for all i. For our example, it takes around 70 iterations to converge so we will not compute these by hand (instead using computational techniques - the code used will be included in the appendix).
The resulting pageranks are;

$$\begin{aligned} R(A) &= \frac{4}{11} \\ R(B) &= \frac{4}{11} \\ R(C) &= \frac{2}{11} \\ R(D) &= \frac{1}{11} \end{aligned}$$

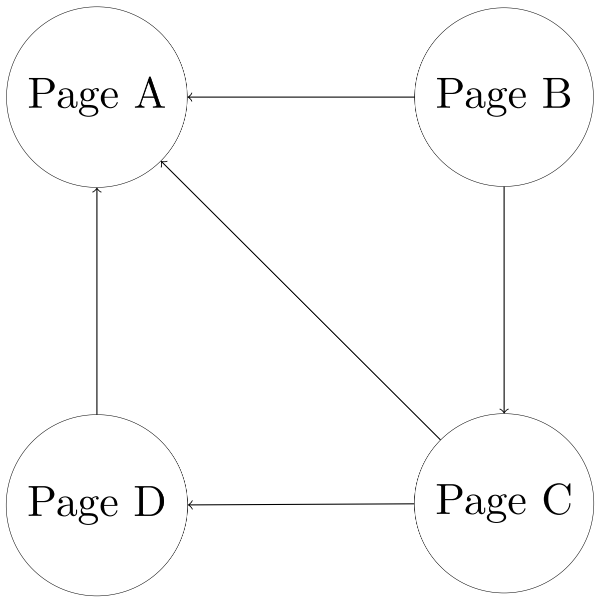
'Surprisingly' we see we end up with the pagerank of A only being joint highest, with a value equal to that of B. We see this highlights the importance of having links from other important pages, that have few outgoing links.

For this example, the algorithm works fine and does its job. However as stated earlier, there is a particular issue that arises with this method. We will consider another example to highlight this issue.

Example 2
Suppose from the sample of web pages in example 1, the hyperlink from page A to page B is removed. We are then left with the following hyperlinks;

Page	Links In	Links Out
A	B, C, D	-
B	-	A, C
C	B	A, D
D	C	A

and a graphical representation



Let us now perform the algorithm on this sample.
Step 1
Step one remains the same and we have

$$R_0(A) = R_0(B) = R_0(C) = R_0(D) = \frac{1}{4}$$

Step 2
We see that Page A has no outgoing links so it keeps its pagerank. Also Page B has no incoming links, so will gain no pagerank. Thus after one iteration we have

$$\begin{aligned}
&R_1(A) = \frac{R_0(B)}{2} + \frac{R_0(C)}{2} + R_0(D) + R_0(A) = \frac{3}{4} \\
&R_1(B) = 0 \\
&R_1(C) = \frac{R_0(B)}{2} = \frac{1}{8} \\
&R_1(D) = \frac{R_0(C)}{2} = \frac{1}{8}
\end{aligned}$$

Step 3 After a second iteration we get

$$\begin{aligned}
&R_2(A) = \frac{R_1(B)}{2} + \frac{R_1(C)}{2} + R_1(D) + R_1(A) = \frac{15}{16} \\
&R_2(B) = 0 \\
&R_2(C) = 0 \\
&R_2(D) = \frac{R_1(C)}{2} = \frac{1}{16}
\end{aligned}$$

And after a third we have

$$\begin{aligned}
R_3(A) &= \frac{R_2(B)}{2} + \frac{R_2(C)}{2} + R_2(D) + R_2(A) = 1 \\
R_3(B) &= 0 \\
R_3(C) &= 0 \\
R_3(D) &= 0
\end{aligned}$$

And so we see after 3 iterations, Page A has 'absorbed' the pagerank of the other pages. As Page A has no outgoing links, this will remain for any further iterations. We refer to this kind of page as a **dangling node**. This leaves us with values of 0 for the pagerank of all the other pages involved. As such there is no way to differentiate between them in the ranking, making the ranking pointless.

3.3.2 Random Surfer

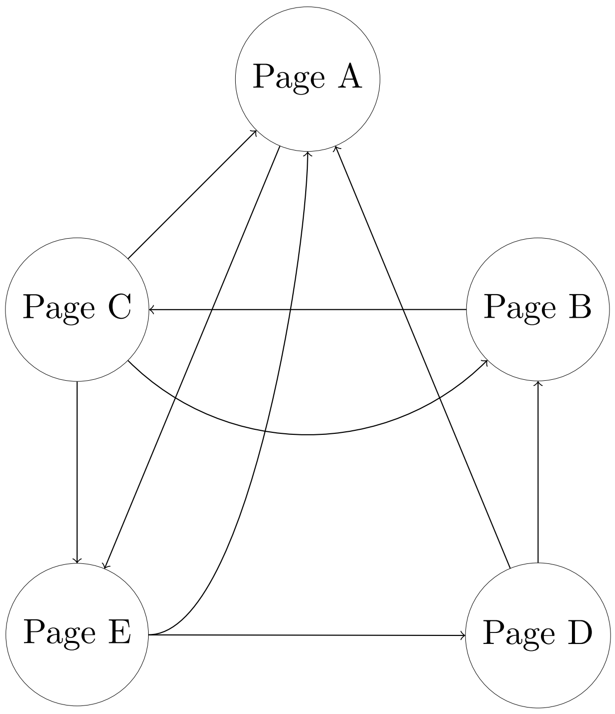
A more sophisticated way to think of this, and the basis of the algorithm, is the random surfer approach [Levene and Loizou, 2002].

Suppose we have a web user who surfs the web by starting at a random web page and randomly clicking outgoing links from that page, each with equal probability. We aim to find out the probability for each page that after a certain amount of clicks, the user has arrived at it.
Combined with our graphical model of the internet, we can model this situation with a random walk on a graph. We will show how we can work with this model to calculate these probabilities and ultimately determine the pagerank of each page.

Example
Recall our example from Section 3.2. We have 5 web pages; Page A, Page B, Page C, Page D and Page E with hyperlinks as below

Page	Links In	Links Out
A	C,D,E	E
B	C,D	C
C	B	A,B,E
D	E	A,B
E	A,C	A,D

a graphical representation,



and a transition matrix,

$$P = \begin{bmatrix} 0 & 0 & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} \\ 1 & 0 & \frac{1}{3} & 0 & 0 \end{bmatrix}$$

Let the graph be called G=(V,E). We have V = A, B, C, D, E, where the vertex i represents page i. E = AE, BC, CA, CB, CE, DA, DB, EA, ED
Let us consider a random walk, V_n, on G. As in the random surfer model the user starts at a random web page with equal probability, we have a probability distribution vector for v_0 given by,

$$\rho_0 = \begin{pmatrix} \frac{1}{5} \\ \frac{1}{5} \\ \frac{1}{5} \\ \frac{1}{5} \\ \frac{1}{5} \end{pmatrix}$$

From **Theorem 2.4.8** we know we can then calculate the probability distribution vector for any future stage using the recurrence formula;

$$\rho_{k+1} = P\rho_k$$

Recall **lemma 2.3.2**. This states for a square N x N matrix, A, the sequence;

$$x_{k+1} = Ax_k$$

converges to the dominant eigenvector of A as k grows.

We see clearly these two formulas are identical, with the A=P and $\rho_k = x_k$. Thus we have if the transition matrix, P, of a graph G has a dominant eigenvector, v_1, then the probability distribution vector of a random walk on G converges to v_1.

So what does this vector tell us? In the case of the random surfer, this vector would represent the probability distribution that the user is at each page after infinite clicks.

This is the idea Google use to rank their pages. Which page is a random surfer most likely to arrive at after infinite clicks? So to find a ranking for our pages, we can compute the power method to find the dominant eigenvector of P.

After one iteration of the power method, we have;

$$\rho_1 = \begin{pmatrix} \frac{4}{15} \\ \frac{1}{6} \\ 0.2 \\ 0.1 \\ \frac{4}{15} \end{pmatrix}$$

and so on... until after 37 iterations we get;

$$\rho_{37} = \begin{pmatrix} 0.28 \\ 0.12 \\ 0.12 \\ 0.16 \\ 0.32 \end{pmatrix}$$

and after 38 we have;

$$\rho_{38} = \begin{pmatrix} 0.28 \\ 0.12 \\ 0.12 \\ 0.16 \\ 0.32 \end{pmatrix}$$

And so we see after 37 iterations we get;

$$\begin{aligned} \rho_{37} &= \rho_{38} \\ &= P\rho_{37} \end{aligned}$$

$\Rightarrow \rho_{37}$ is an eigenvector of P by definition, and by lemma 2.3.2, we know this is the dominant eigenvector.

3.3.3 Damping Factor

With this method, we have still not addressed the issue of dangling nodes. Here we see a dangling node would cause a random surfer to get 'stuck' if he ever reached that page. As the number of clicks tends to infinity, the probability the user visits one of these pages, and thus gets stuck at this page, becomes 1.

To counter this, Google introduced something called a **damping factor** to their algorithm. The idea, in terms of the random surfer model, is that before each click their is a small probability the user gets 'bored' on his current path of clicking and goes to a new random page.

So we have a large probability - which we call the damping factor, d - that the user continues along his path of clicking links, and a small probability, 1-d, that he jumps to a random page. A probability of 1-d that the user jumps to a random page implies a probability of $\frac{1-d}{N}$ that the user jumps to each specific page.

We can alter our graphical model to show this.

- We first add weightings to our existing graph. Each already existing edge is weighted d+$\frac{1-d}{N}$. This represents the probability our user follows the link between two pages plus the probability he gets bored and randomly jumps between the two pages.
- New edges are then added between each page without an existing link (in each direction) all with weighting $\frac{1-d}{N}$.

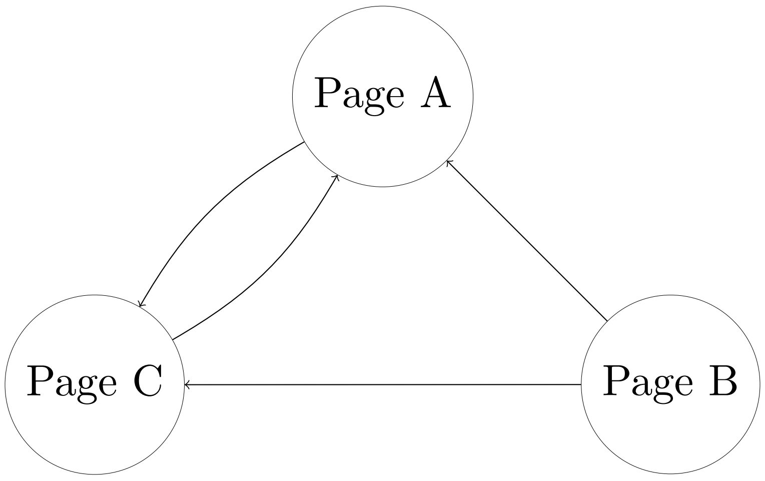
We will show how this looks with a small example (as any bigger example will be very messy).

Example

Consider a sample of 3 web pages, Page A, Page B and Page C with hyperlinks as below.

Page	Links In	Links Out
A	C,B	C
B	C	A,C
C	B	A,B

Originally, we would represent this graphically as below.



Now we will represent this accounting for the damping factor. For this example we will set d = 0.85. This is the baseline value Google uses.

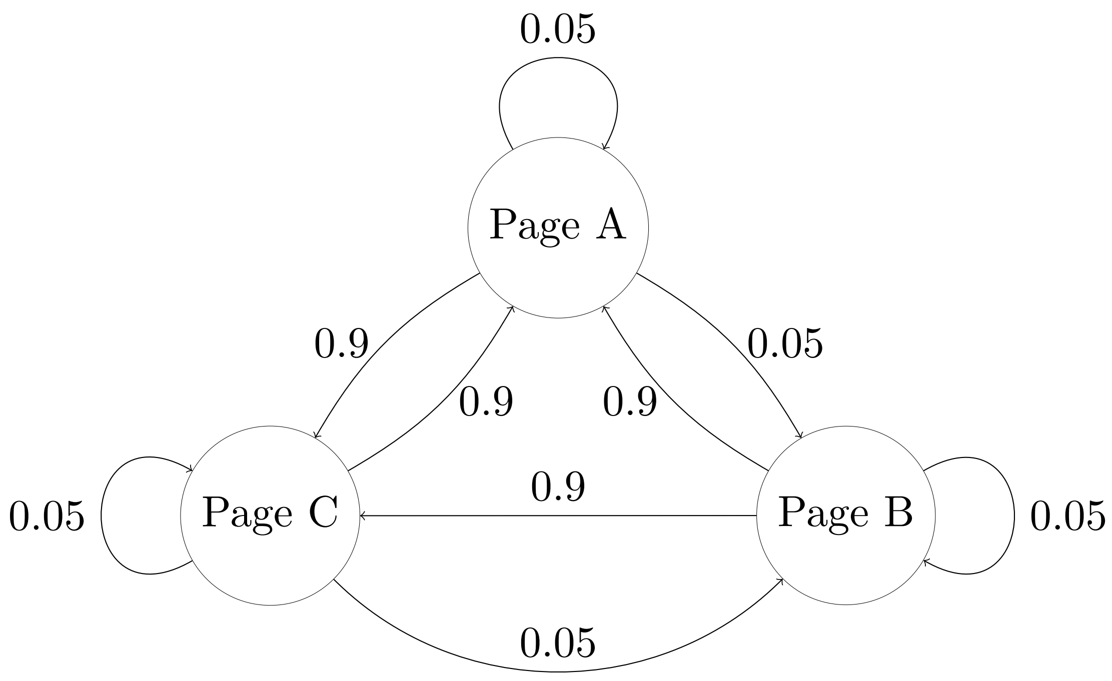
So for edges already in the graph above we will add a weighting of;

$$d + \frac{1-d}{N} = 0.85 + \frac{1-0.85}{3} = 0.9$$

and for the edges we add we will assign a weighting of;

$$\frac{1-d}{N} = \frac{1-0.85}{3} = 0.05$$

The resulting graph is below.



For the transition matrices of the graphs, this transformation is represented by;

$$G = dP + \frac{1-d}{N} J_N$$

where G is the transition matrix of our new graph, P is the transition matrix of the original graph and J_N is the N x N matrix with each entry equal to 1. In general we will refer to G as the google matrix.

3.3.4 Complete Algorithm

Now we have talked in depth over each stage of the algorithm, we can now put together a complete step by step.

For a group of web-pages of size N, the Google pagerank algorithm is computed as follows;

1. Index each page, from 1-N, and create an adjacency matrix, A, given by;

$$A_{ij} = \begin{cases} 1 & \text{if page j links to page i} \\ 0 & \text{otherwise} \end{cases}$$

2. Create a transition matrix, P, given by;

$$P_{ij} = \frac{A_{ij}}{\sum_{i=1} A_{ij}}$$

3. Create the Google matrix, G, given by;

$$G = dP + \frac{1-d}{N} J_N$$

   where d is the damping factor set by Google between 0 and 1 - usually 0.85.

4. Compute the power method to find the dominant eigenvector of G.

This vector then acts as a ranking for the pages, with the ith entry of the vector representing page i. A page with a higher entry in this vector is ranked higher.

3.4 Why it Works

3.4.1 Existence and Uniqueness of Ranking Vector

It is in this section we will reveal the true importance of the Perron-Frobenius theorem for this algorithm.
Consider the aim of our algorithm. We have a graphical model of the internet, and we wish to find the probability distribution vector representing the probabilities of a random walk of infinite length ending at each node. We know this is given by the dominant eigenvector of the transition matrix of the graph, if it has one. But how do we know this vector exists? And how do we know the ranking we get is unique?
Well let us think about the algorithm, and in particular the damping factor and what it does to our model.

In terms of our graphical model of the web, we know the damping factor adds an edge from each node to each other node it was not already connected to. Thus we end up with every node being adjacent to each other node. Trivially, this graph is then strongly connected. For each node u and v, we have a path from u to v, namely uv.
As our graph is strongly connected, from **2.2.8** we know our adjacency matrix, and thus our transition matrix, is irreducible. This allows us to apply the rules from **Theorem 2.2.10**.
These tell us the dominant eigenvalue both exists, and is unique.

3.4.2 Convergence of the Power Method and a Non-Stationary Damping Factor

Now we know our Google matrix has a unique, dominant eigenvector - our ranking vector we aim to find - all that remains is to find this. We know to do this we compute the power method on our Google matrix. However, in this process there remains one small caveat. Recall how we began **theorem 2.3.1**.

"Let A be a square N x N matrix, with dominant eigenvector v_1 and corresponding eigenvalue λ_1 and n linearly independent eigenvectors, $v_1, v_2, ..., v_n$..." We know our matrix has dominant eigenvector and eigenvalue, proved by the Perron-Frobenius theorem. But how do we know it has linearly independent eigenvectors? Actually, we do not.

Definition 3.4.3 *(Diagonalisable Matrix) [Bengtsson and Weisstein, 2022]*

An $n \times n$ matrix A is said to be diagonalisable if it can be written on the form;

$$A = PDP^{-1}$$

where D is a diagonal (only has entries on the main diagonal) $n \times n$ matrix with the eigenvalues of A as its entries and P is a non-singular (invertible) $n \times n$ matrix consisting of the eigenvectors corresponding to the eigenvalues in D.

Theorem 3.4.4 *[Mitchell, 1953]*

An $n \times n$ matrix A is diagonalisable if and only if A has n linearly independent eigenvectors.

Thus we require our Google vector to be diagonlisable. Now, as stated above this is not always the case. There do exist matrices that cannot be diagonalised. However we CAN say that the probability a randomly chosen matrix is diagonalisable is extremely high. In fact;

Theorem 3.4.5 *[Paranjape, 2002] [Hetzel et al., 2007]*
If we choose a matrix randomly (in a uniform distribution) from within a bounded region, then it will turn out to be diagonalisable over the complex numbers with probability 1.

So to counter the possibility that our given Google matrix is non-diagonalisible, Google use a clever trick. Recall when we spoke about damping factor, we said the value is USUALLY 0.85. This is because in the case where our matrix is non-diagonalisible, Google uses a 'ball' of values around 0.85. These values would not differ greatly from 0.85 and could be for example 0.8499999999999... The idea is that if the matrix given by using 0.85 as the damping factor is non-diagonalisable, Google randomly select a different damping factor from this ball of values. This gives us a different random Google matrix, and thus we know the probability of this matrix being diagonalisable is 1.

4 Pagerank as a Ranking System in Sports

Previously, we have observed and discussed the efficacy of Pagerank as a ranking algorithm for web-pages. In this section, we will look at an alternative use for the algorithm, with the purpose of evaluating the algorithm as a general ranking system. To do this we will consider the algorithm as a ranking system for NFL teams, and will compute some statistical analysis on the output to determine and discuss the quality of the results.

4.1 The GeM Method

The idea for this use of the pagerank algorithm is first used in a 2008 paper by Anjela Y.Govan, Carl D.Meyer and Russell Albright. In their paper they proposed the GeM method which we will present below [Govan et al., 2008].

4.1.1 Model

As with the pagerank algorithm, the initial requirement is to correctly model the things we wish to rank. For this method, the model used is as follows;

- Represent each team as a node on a directed weighted graph.
- Represent each game as an edge from loser to winner with weighting given by the positive score differential of the game.

In the case where teams play multiple games against each other, these games are treated as one game with the score given by the sum of the scores across both games.

4.1.2 Algorithm

1. Index each team, and create a cost matrix, A, given by;

$$A_{ij} = \begin{cases} w_{ij} & \text{if team j beat team i} \\ 0 & \text{otherwise} \end{cases}$$

   where w_{ij} is the positive score differential between the two teams.

2. Create a transition matrix, P, given by;

$$P_{ij} = \frac{A_{ij}}{\sum_{i=1} A_{ij}}$$

3. Create the GeM matrix, G, given by;

$$G = dP + \frac{1-d}{N} J_N$$

where d is the damping factor set by Google between 0 and 1 - usually 0.85.

4. Find the dominant eigenvector of G.

As with pages in the pagerank algorithm, this vector is then used to rank the teams.

Example
In order to show how this algorithm works, we will look at a small sample of games from the most current league season, the 2021/22 season. We will consider solely games within one division of the NFL - the NFC North. This includes only games between 4 teams; Minnesota Vikings, Green Bay Packers, Chicago Bears and Detroit Lions. We will abbreviate these as;

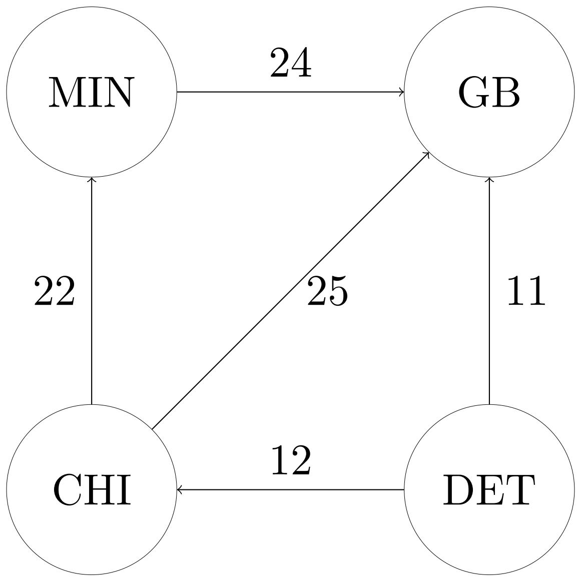
- Minnesota Vikings - MIN
- Green Bay Packers - GB
- Chicago Bears - CHI
- Detroit Lions - DET

The results of the games between these teams were as follows;

Home Team	Score	Away Team
MIN	19-17	DET
DET	29-27	MIN
MIN	34-31	GB
GB	37-10	MIN
MIN	31-17	CHI
CHI	9-17	MIN
DET	37-30	GB
GB	35-17	DET
DET	14-16	CHI
CHI	24-14	DET
GB	45-30	CHI
CHI	14-24	GB

As mentioned, when there is multiple games between the teams, we model this as one game with the result is given by the sum of the scores across both games. For example we would treat the games between MIN and DET as one game with total score MIN 46-46 DET. The games between MIN and GB would have total score GB 68-44 MIN.

We would thus model this sample graphically as below;



We can now enact the GeM method on this model.

Step 1

We will first apply the following indexing; MIN - 1, GB - 2, CHI - 3, DET - 4. We can then create a cost matrix for our graph given by;

$$A = \begin{bmatrix} 0 & 0 & 22 & 0 \\ 24 & 0 & 25 & 11 \\ 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2

We then create a transition matrix;

$$P = \begin{bmatrix} 0 & 0 & \frac{22}{47} & 0 \\ 1 & 0 & \frac{25}{47} & \frac{11}{23} \\ 0 & 0 & 0 & \frac{12}{23} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 3

Then we create the GeM matrix given by;

$$G = 0.85P + \frac{0.15}{4}J_4$$

Step 4

Finally, we can use the power method to find the dominant eigenvector of G.

This will produce a rating of each team and from there we can rank them.

Upon computation, we find the dominant eigenvector of G is given by;

$$\rho = \begin{pmatrix} 0.18982712 \\ 0.59188261 \\ 0.14282234 \\ 0.07546793 \end{pmatrix}$$

And so we have the final ranking;

Rank	Team
1	GB
2	MIN
3	CHI
4	DET

4.2 2017/18 Season

In this next section, we will run the algorithm on the the complete set of games from the 2017/18 NFL regular season. We will then compare the output of this algorithm with other methods of team ranking within the NFL. Our primary purpose here is to discuss whether this method is a useful tool to predict future success of NFL teams - i.e. does a high ranking from our GeM method suggest a teams will perform well in the next year.

4.2.1 Our Ranking

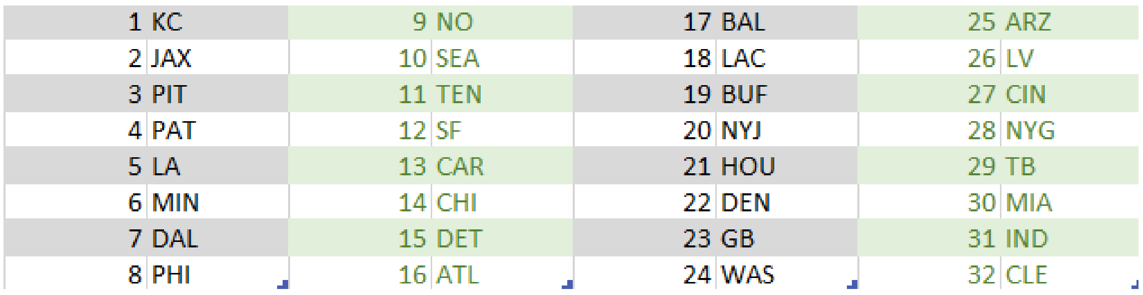
For this we will not go through the method again in any detail - however all code and steps used will be included in the appendix if you wish to see. As with our small example we will first give the abbreviations and index we will use for each team;

Index	Team	Abbreviation	Index	Team	Abbreviation
1	49ers	SF	17	Jaguars	JAX
2	Bears	CHI	18	Jets	NYJ
3	Bengals	CIN	19	Lions	DET
4	Bills	BUF	20	Packers	GB
5	Broncos	DEN	21	Panthers	CAR
6	Browns	CLE	22	Patriots	PAT
7	Buccanee	TB	23	Raiders	LV
8	Cardinals	ARZ	24	Rams	LAR
9	Chargers	LAC	25	Ravens	BAL
10	Chiefs	KC	26	Redskins	WAS
11	Colts	IND	27	Saints	NO
12	Cowboys	DAL	28	Seahawks	SEA
13	Dolphins	MIA	29	Steelers	PIT
14	Eagles	PHI	30	Texans	HOU
15	Falcons	ATL	31	Titans	TEN
16	Giants	NYG	32	Vikings	MIN

Upon completing the algorithm we get the following vector output;

$$\rho = \begin{pmatrix} 0.4913985 \\ 0.3629603 \\ 0.15639533 \\ 0.30725353 \\ 0.23631527 \\ 0.06121233 \\ 0.1379599 \\ 0.21000763 \\ 0.31984454 \\ 1 \\ 0.09491166 \\ 0.58282981 \\ 0.13630358 \\ 0.58016957 \\ 0.32653519 \\ 0.14266035 \\ 0.91609575 \\ 0.30270775 \\ 0.3402907 \\ 0.23434408 \\ 0.48035997 \\ 0.75324757 \\ 0.18420621 \\ 0.73595978 \\ 0.32206391 \\ 0.21507451 \\ 0.57131406 \\ 0.52499467 \\ 0.83642978 \\ 0.27313395 \\ 0.5211965 \\ 0.70045469 \end{pmatrix}$$

which corresponds to the following ranking;



1	KC	9	NO	17	BAL	25	ARZ
2	JAX	10	SEA	18	LAC	26	LV
3	PIT	11	TEN	19	BUF	27	CIN
4	PAT	12	SF	20	NYJ	28	NYG
5	LA	13	CAR	21	HOU	29	TB
6	MIN	14	CHI	22	DEN	30	MIA
7	DAL	15	DET	23	GB	31	IND
8	PHI	16	ATL	24	WAS	32	CLE

4.3 Comparison and Analysis

We will not delve too deep into statistical terminology in this section, but before we begin our analysis we will go through the measure we will use to analyse our ranking.

Definition 4.3.1 *(Spearman's Correlation Coefficient) [Myers and Sirois, 2004]*

The Spearman's Rank Correlation Coefficient which we will call r is the non-parametric statistical measure used to study the strength of association between the two ranked variables.

More information about this coefficient, including how to count it and a more rigorous definition can be found in the article referenced. However for our purposes we must understand some features of this coefficient.

- -1 $\leq r \leq$ 1
- r = 0 implies no correlation between rankings.
- r = 1 implies the rankings are identical
- r = -1 implies the rankings are opposite (i.e. rank 1 is ranked last rank 2 is ranked 2nd last)
- r closer to 1 implies stronger correlation between rankings.

4.3.2 Similarity to Other Rankings

In this section we aim to evaluate the viability of our ranking as a method to determine the best team. To do this we will compare our ranking to that of other methods. First I will present the other ranking methods, and discuss them briefly.

EPA

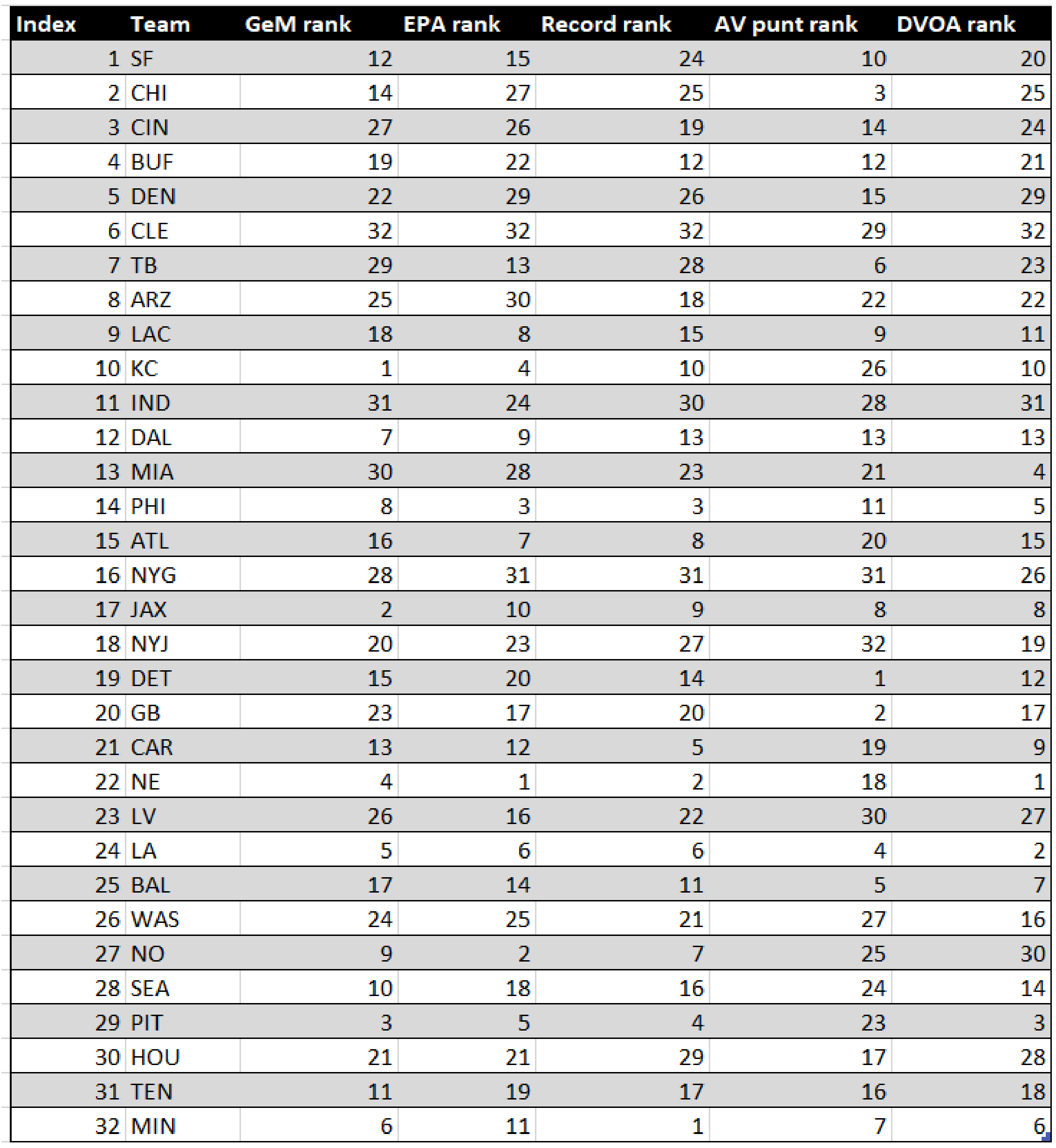
On the website NFELO, in an article about EPA, they define EPA as follows; ”On every play, a team has the potential to either increase or decrease their Expected Points. Expected Points Added is the difference between a team’s Expected Points at the end of a play and their Expected Points at the beginning of a play.” [Lourie, 2021a]. As this is a play by play method it is a useful evaluator for how good a team truly played, regardless of result. This makes it a useful evaluator for teams as it cares less about the small sample size things such as scoring and gives a deep analysis of team quality.

Team Record
Team record is a more self-explanatory ranking. Teams who won more games are ranked higher. If two teams won the same amount of games, we use the tie-breaker system found on the NFL website. Of course, team record is an important statistic as winning games is the ultimate goal of football. However, as an evaluator of teams it is far from perfect. It fails to take into account a variety of factors such as difficulty of schedule.

DVOA
DVOA is Football Outsiders signature statistic and according to their website ”DVOA is a method of evaluating teams, units, or players. It takes every single play during the NFL season and compares each one to a league-average baseline based on situation. DVOA measures not just yardage, but yardage towards a first down: Five yards on third-and-4 are worth more than five yards on first-and-10 and much more than five yards on third-and-12. Red zone plays are worth more than other plays. Performance is also adjusted for the quality of the opponent. DVOA is a percentage, so a team with a DVOA of 10.0 percent is 10 percent better than the average team, and a quarterback with a DVOA of -20.0 percent is 20 percent worse than the average quarterback.” [Outsiders, 2022]. Importantly for later, according to a studies, DVOA is a highly predictive statistic for future winning - i.e. chances of winning games in the next season [Lourie, 2021b].

Average Punt Distance
[SportsLingo, 2022] define a punt as follows. ”This is a term used in football to describe when the offensive team kicks the ball to the defensive team after they’ve exhausted all of their downs.” This statistic has been included as something of a control group. The idea is this statistic - a small, and widely thought of as ’unimportant’ part of the game - will be poor at ranking teams and will output dissimilar rankings to the other methods. It will rank teams based solely on the average distance of their ’punts’ that season.

Now we will use the Spearman test to analyse our ranking. First we will present the results of each previously mentioned ranking method from the 2017/18 regular season.



Index	Team	GeM rank	EPA rank	Record rank	AV punt rank	DVOA rank
1	SF	12	15	24	10	20
2	CHI	14	27	25	3	25
3	CIN	27	26	19	14	24
4	BUF	19	22	12	12	21
5	DEN	22	29	26	15	29
6	CLE	32	32	32	29	32
7	TB	29	13	28	6	23
8	ARZ	25	30	18	22	22
9	LAC	18	8	15	9	11
10	KC	1	4	10	26	10
11	IND	31	24	30	28	31
12	DAL	7	9	13	13	13
13	MIA	30	28	23	21	4
14	PHI	8	3	3	11	5
15	ATL	16	7	8	20	15
16	NYG	28	31	31	31	26
17	JAX	2	10	9	8	8
18	NYJ	20	23	27	32	19
19	DET	15	20	14	1	12
20	GB	23	17	20	2	17
21	CAR	13	12	5	19	9
22	NE	4	1	2	18	1
23	LV	26	16	22	30	27
24	LA	5	6	6	4	2
25	BAL	17	14	11	5	7
26	WAS	24	25	21	27	16
27	NO	9	2	7	25	30
28	SEA	10	18	16	24	14
29	PIT	3	5	4	23	3
30	HOU	21	21	29	17	28
31	TEN	11	19	17	16	18
32	MIN	6	11	1	7	6

As with earlier computation, the method used will be included in the abstract. However here we will simply present and discuss the results.

Results

In the following table we give the Spearman's correlation coefficient between our GeM ranking and each of the other rankings.

Ranking Method	Correlation with GeM Method
EPA	0.769061584
Record	0.794354839
Av Punt Dist	0.281891496
DVOA	0.636363636

We see the GeM method has strong correlation with EPA, DVOA and team record. These are the 3 rankings we thought were indicative of a 'good' team. We see much weaker correlation with average punt distance, which we thought would be less indicative of a 'good' team.

Conclusion
It is obvious our method, over our 2017/18 season sample, ranks teams similarly to other high quality rankings of teams.

4.3.3 Future Success

But what is the purpose of finding 'good' teams or rather what makes a team 'good'? If the aim is to win the most games, surely the only statistic of importance is team record. For one individual season, this is the case. However, as we take a step back, we see that other statistics are better suited to predict future success - i.e. a better team record in the next season. Thus when we say a 'good' team, we mean one more likely to win in the future.
As such, we can compute further analysis on our ranking and consider it as a predictor of future outcomes. Firstly we should say that the structure of the NFL is designed to promote balance between teams. It aims to make it harder for the better teams to improve and easier for the worse teams. Thus each season there is large changes in performance of teams, and as such even the best indicators of future success are far from perfect.

For this part of our analysis we will compute the Spearman test between our rankings we have discussed - including the GeM method - and the next seasons (2018/19) team record rankings.

Results
The results from this are then as follows;

Ranking Method	Correlation with 2018/19 Team Record
GeM	0.455645161
EPA	0.470307918
Record	0.373900293
Av Punt Dist	0.12170088
DVOA	0.255865103

We see our GeM method has the 2nd highest correlation with the 2018/19 seasons team record, with just slightly less than EPA. Average punt distance has almost no correlation, and DVOA and team record are somewhere between.

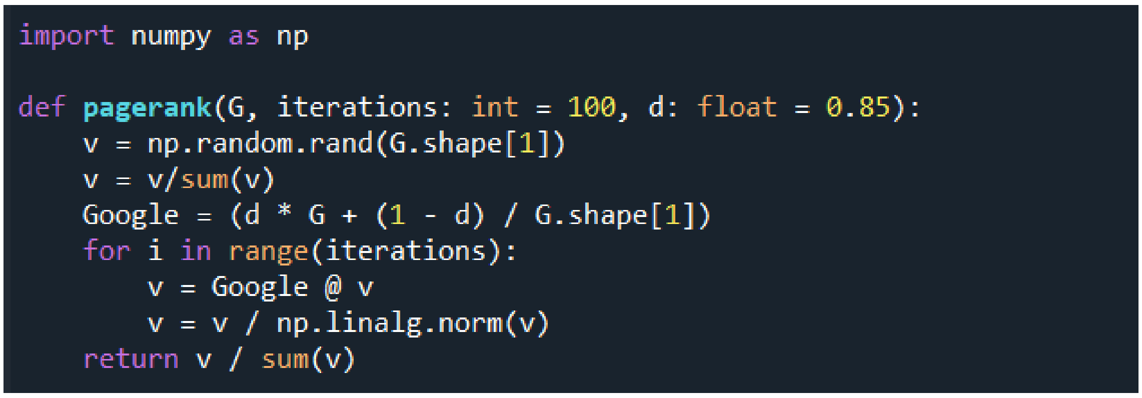
4.3.4 Conclusion

We see from the our tests, there is potential the GeM method could serve as a beneficial ranking method within the NFL. Over our sample season, when compared with other ranking methods, it predicted the next seasons ranking better than team record, DVOA and (unsurprisingly) average punt distance, whilst being only slightly worse than EPA. From these results, it is definitely worth considering further research over a greater sample of seasons to get a more accurate depiction of the quality of our ranking.

Appendices

A Code to Calculate Pagerank

Here is the code we used to calculate our values of PageRank throughout our workings. This code inputs a matrix, G, a number of iterations and a damping factor, d. It then outputs, for valid matrices, a stochastic vector which we know to be the dominant eigenvector and the ranking vector in the pagerank process.

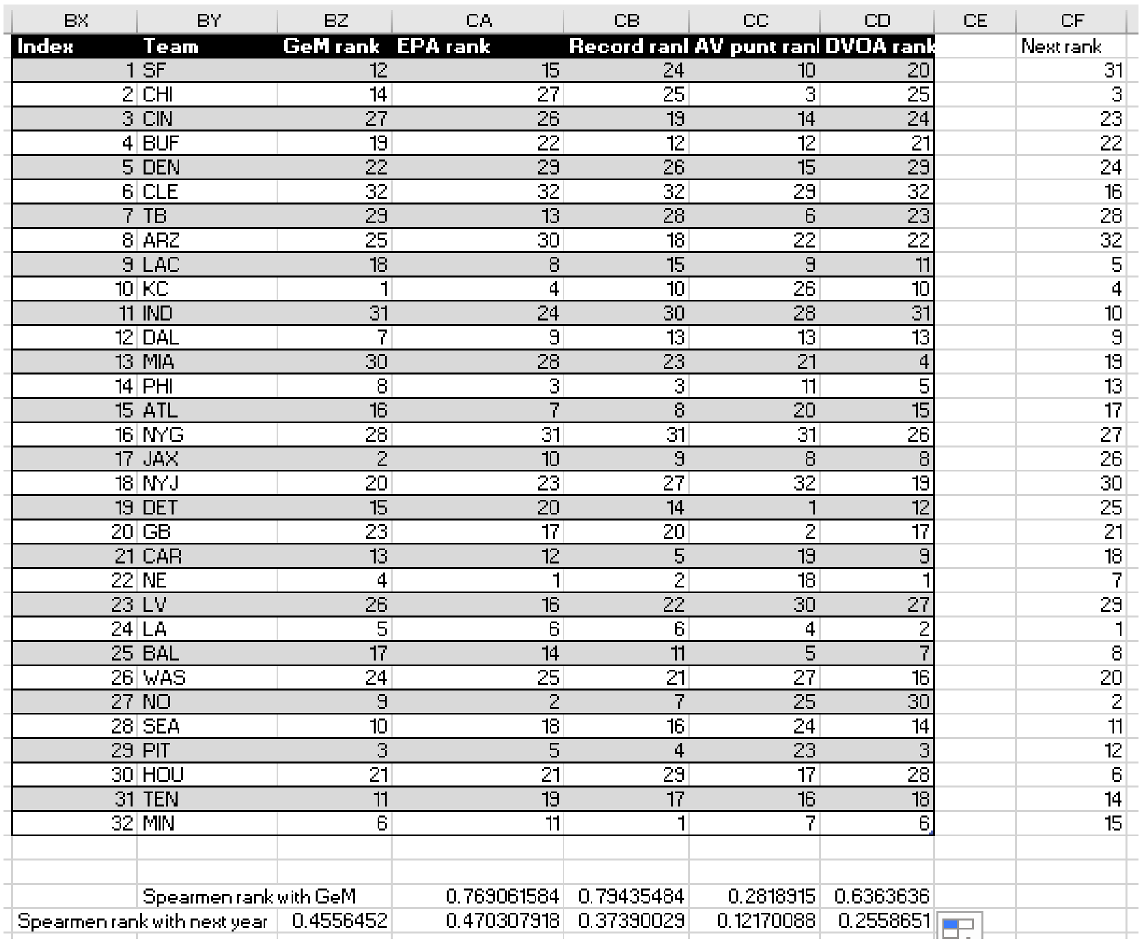


```
import numpy as np

def pagerank(G, iterations: int = 100, d: float = 0.85):
    v = np.random.rand(G.shape[1])
    v = v/sum(v)
    Google = (d * G + (1 - d) / G.shape[1])
    for i in range(iterations):
        v = Google @ v
        v = v / np.linalg.norm(v)
    return v / sum(v)
```

B NFL Data and Code

The data we used for the NFL team scores was accessed from a google drive file of NFL scores [Stats, 2019]. We then exported this data to excel so that we could get the score data in the form we require to import it to python. Once we had the data in python, we used our pagerank code to calculate the ranking between teams. Then we imported this ranking vector back to excel, where we indexed and sorted highest to lowest to find our ranking. Once we had our ranking, and the other rankings we wished to compare with in excel (shown in the table below) we could then begin to analyse this.



BX	BY	BZ	CA	CB	CC	CD	CE	CF
Index	Team	GeM rank	EPA rank	Record ranl	AV punt ranl	DVOA rank		Next rank
1	SF	12	15	24	10	20		31
2	CHI	14	27	25	3	25		3
3	CIN	27	26	19	14	24		23
4	BUF	19	22	12	12	21		22
5	DEN	22	29	26	15	29		24
6	CLE	32	32	32	29	32		16
7	TB	29	13	28	6	23		28
8	ARZ	25	30	18	22	22		32
9	LAC	18	8	15	9	11		5
10	KC	1	4	10	26	10		4
11	IND	31	24	30	28	31		10
12	DAL	7	9	13	13	13		9
13	MIA	30	28	23	21	4		19
14	PHI	8	3	3	11	5		13
15	ATL	16	7	8	20	15		17
16	NYG	28	31	31	31	26		27
17	JAX	2	10	9	8	8		26
18	NYJ	20	23	27	32	19		30
19	DET	15	20	14	1	12		25
20	GB	23	17	20	2	17		21
21	CAR	13	12	5	19	9		18
22	NE	4	1	2	18	1		7
23	LV	26	16	22	30	27		29
24	LA	5	6	6	4	2		1
25	BAL	17	14	11	5	7		8
26	WAS	24	25	21	27	16		20
27	NO	9	2	7	25	30		2
28	SEA	10	18	16	24	14		11
29	PIT	3	5	4	23	3		12
30	HOU	21	21	29	17	28		6
31	TEN	11	19	17	16	18		14
32	MIN	6	11	1	7	6		15
	Spearmen rank with GeM		0.769061584	0.79435484	0.2818915	0.6363636		
Spearmen rank with next year		0.4556452	0.470307918	0.37390029	0.12170088	0.2558651		

To find the correlation between the other rankings and the GeM ranking we used the CORREL formula as follows;

- =CORREL($BZ2:$BZ33, CA2:CA33) - returns the correlation with the EPA ranking.
- =CORREL($BZ2:$BZ33, CB2:CB33) - returns the correlation with the record ranking.
- =CORREL($BZ2:$BZ33, CC2:CC33) - returns the correlation with the average punt distance ranking.
- =CORREL($BZ2:$BZ33, CD2:CD33) - returns the correlation with the DVOA ranking.

Similarly to find the correlation between next years ranking and our rankings we use the following formulas;

- = CORREL(BZ2:BZ33,CF2 :CF33) - for GeM
- = CORREL(CA2:CA33,CF2 :CF33) - for EPA
- = CORREL(CB2:CB33,CF2 :CF33) - for record
- = CORREL(CC2:CC33,CF2 :CF33) - for average punt distance
- = CORREL(CD2:CD33,CF2 :CF33) - for DVOA

References

[Amine, 2020] Amine, A. (2020). Pagerank algorithm, fully explained.

[Armstrong, 2012] Armstrong, D. (2012). The perron-frobenius theorem.

[Aykroyd, 2018] Aykroyd, R. G. (2018). Probability and statistics [lecture notes accessed through minerva], math1710 probability and statistics i.

[Bengtsson and Weisstein, 2022] Bengtsson, V. and Weisstein, E. W. (2022). "diagonalizable matrix." from mathworld–a wolfram web resource.

[Chang et al., 2008] Chang, K.-C., Pearson, K., and Zhang, T. (2008). Perron-frobenius theorem for nonnegative tensors. *Communications in Mathematical Sciences*, 6(2):507–520.

[Chen, 2012] Chen, W.-K. (2012). *Applied graph theory*, volume 13. Elsevier.

[Coolen, 2009] Coolen, A. (2009). Markov chains.

[Easton and McColl, 1997] Easton, V. J. and McColl, J. H. (1997). Statistics glossary v1. 1.

[Gambino, 2020] Gambino, N. (2020). Chapter 1: Basic notions [lecture notes accessed through minerva], math3033 graph theory.

[Govan et al., 2008] Govan, A. Y., Meyer, C. D., and Albright, R. (2008). Generalizing google's pagerank to rank national football league teams. In *Proceedings of the SAS Global Forum*, volume 2008.

[Guichard, 2017] Guichard, D. (2017). An introduction to combinatorics and graph theory. *Whitman College-Creative Commons.*

[Hetzel et al., 2007] Hetzel, A. J., Liew, J. S., and Morrison, K. E. (2007). The probability that a matrix of integers is diagonalizable. *The American Mathematical Monthly*, 114(6):491–499.

[Higham, 2021] Higham, N. (2021). What is the perron–frobenius theorem?

[Langville and Meyer, 2004] Langville, A. N. and Meyer, C. D. (2004). Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380.

[Lawler and Limic, 2010] Lawler, G. F. and Limic, V. (2010). *Random walk: a modern introduction*, volume 123. Cambridge University Press.

[Levene and Loizou, 2002] Levene, M. and Loizou, G. (2002). Kemeny's constant and the random surfer. *The American mathematical monthly*, 109(8):741–745.

[Lewis, 2013] Lewis, F. (2013). Matrix analysis of graphs.

[Lourie, 2021a] Lourie, S. (2021a). What are expected points added (epa) in the nfl.

[Lourie, 2021b] Lourie, S. (2021b). Which stats are most predictable and predictive?

[Mitchell, 1953] Mitchell, B. (1953). Normal and diagonalizable matrices. *The American Mathematical Monthly*, 60(2):94–96.

[Myers and Sirois, 2004] Myers, L. and Sirois, M. J. (2004). Spearman correlation coefficients, differences between. *Encyclopedia of statistical sciences*, 12.

[Ninio, 1976] Ninio, F. (1976). A simple proof of the perron-frobenius theorem for positive symmetric matrices. *Journal of Physics A: Mathematical and General*, 9(8):1281.

[Outsiders, 2022] Outsiders, F. (2022). What is dvoa?

[Paranjape, 2002] Paranjape, K. H. (2002). Are all matrices diagonalisable?

[Peretti and Roveda, 2014] Peretti, A. and Roveda, A. (2014). On the mathematical background of google pagerank algorithm.

[Poole, 2014] Poole, D. (2014). *Linear algebra: A modern introduction*. Cengage Learning.

[Rogers, 2002] Rogers, I. (2002). The google pagerank algorithm and how it works.

[SportsLingo, 2022] SportsLingo (2022). Sports glossary; what is a punt?

[StatCounter, 2022] StatCounter (2022). Search engine market share worldwide.

[Statistics, 2018] Statistics, L. (2018). Spearman's rank order correlation.

[Stats, 2019] Stats, N. G. (2019). Nfl game stats.

[Tutte and Tutte, 2001] Tutte, W. T. and Tutte, W. T. (2001). *Graph theory*, volume 21. Cambridge university press.

[Wang and Swendsen, 2002] Wang, J.-S. and Swendsen, R. H. (2002). Transition matrix monte carlo method. *Journal of statistical physics*, 106(1):245–285.

[Weisstein, 2022a] Weisstein, E. W. (2022a). ”identity matrix” from mathworld–a wolfram web resource.

[Weisstein, 2022b] Weisstein, E. W. (2022b). ”stochastic matrix” from mathworld–a wolfram web resource.

[Weisstein, 2022c] Weisstein, E. W. (2022c). ”weighted graph” from mathworld–a wolfram web resource.

[Yan and Ding, 2011] Yan, E. and Ding, Y. (2011). Discovering author impact: A pagerank perspective. *Information processing & management*, 47(1):125–134.

UNIVERSITY OF LEEDS

School of Mathematics

Declaration of Academic Integrity
for Individual Pieces of Work

I am aware that the University defines plagiarism as presenting someone else's work as your own. Work means any intellectual output, and typically includes text, data, images, sound or performance.

I promise that in the attached submission I have not presented anyone else's work as my own and I have not colluded with others in the preparation of this work. Where I have taken advantage of the work of others, I have given full acknowledgement. I have read and understood the University's published rules on plagiarism and also any more detailed rules specified at School or module level. I know that if I commit plagiarism I can be expelled from the University and that it is my responsibility to be aware of the University's regulations on plagiarism and their importance.

I re-confirm my consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to monitor breaches of regulations, to verify whether my work contains plagiarised material, and for quality assurance purposes.

I confirm that I have declared all mitigating circumstances that may be relevant to the assessment of this piece of work and that I wish to have taken into account. I am aware of the School's policy on mitigation and procedures for the submission of statements and evidence of mitigation. I am aware of the penalties imposed for the late submission of coursework.

Student Signature BNeve Date 06/06/2022

Student Name Bradley Neve Student Number 201242617
