Discrete Mathematical Models

Lecture 29

Kane Townsend

Semester 2, 2024

D3: Random walks on graphs

Introduction

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Imagine that you are walking on this digraph.
Travelling through an edge takes you one unit of time.

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Imagine that you are walking on this digraph.
Travelling through an edge takes you one unit of time.

At time 0, you are at vertex x.
At time 1 you are at a vertex $y \in V$, with $(x, y) \in E$.

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Imagine that you are walking on this digraph.
Travelling through an edge takes you one unit of time.

At time 0, you are at vertex x.
At time 1 you are at a vertex $y \in V$, with $(x, y) \in E$.
At time 2 you are at a vertex $z \in V$ with $(y, z) \in E$.

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Imagine that you are walking on this digraph.
Travelling through an edge takes you one unit of time.

At time 0, you are at vertex x.
At time 1 you are at a vertex $y \in V$, with $(x, y) \in E$.
At time 2 you are at a vertex $z \in V$ with $(y, z) \in E$.
At time k you are at a vertex t and there is a walk of length k from x to t.

Random walks: idea

Let G be a digraph with n vertices $V = V(G) = \{1, ..., n\}$ and (directed) edge set $E = E(G)$.

[We will stick to these meanings for G, n, V and E throughout this final section of the notes.]

Imagine that you are walking on this digraph.
Travelling through an edge takes you one unit of time.

At time 0, you are at vertex x.
At time 1 you are at a vertex $y \in V$, with $(x, y) \in E$.
At time 2 you are at a vertex $z \in V$ with $(y, z) \in E$.
At time k you are at a vertex t and there is a walk of length k from x to t.

Before each step, you choose where to go next probabilistically :
If you are at a vertex i you go to vertex j with probability $p_{i\,j}$.
[If $(j, i) \notin E$, then, of course, $p_{i\,j} = 0$.]

Random walks: definition

Associated to an n-vertex directed graph G, let $T = (p_{i\,j})_{1 \leq i,j \leq n}$ be a matrix s.t. $p_{i,j} = 0 \; \forall (j,i) \notin E(G)$

Random walks: definition

Associated to an n-vertex directed graph G, let $T = (p_{i\,j})_{1 \leq i,j \leq n}$ be a matrix s.t. $p_{i,j} = 0 \; \forall (j,i) \notin E(G)$

For any given n let B_n denote the set of **basis vectors** $\{e_1, \ldots, e_n\}$ where e_i is the $n \times 1$ vector with 1 as the i-th entry (*i.e.* in row i) and all other entries zero. *E.g*, for $n = 3$:

$$e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Random walks: definition

Associated to an n-vertex directed graph G, let $T = (p_{i\,j})_{1 \le i,j \le n}$ be a matrix s.t. $p_{i,j} = 0 \; \forall (j,i) \notin E(G)$

For any given n let B_n denote the set of **basis vectors** $\{\mathrm{e}_1, \ldots, \mathrm{e}_n\}$ where e_i is the $n \times 1$ vector with 1 as the i-th entry (*i.e.* in row i) and all other entries zero. *E.g*, for $n = 3$:

$$\mathrm{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

For $X_0 = \mathrm{e}_i \in B_n$ the Markov chain $(X_k)_{k \in \mathbb{N}^\star}$ specified by G and T is called the **random walk** on G starting at vertex i (or "at e_i"), with transition matrix T.

Random walks: definition

Associated to an n-vertex directed graph G, let $T = (p_{i\,j})_{1 \le i,j \le n}$ be a matrix s.t. $p_{i,j} = 0\ \forall (j,i) \notin E(G)$

For any given n let B_n denote the set of **basis vectors** $\{e_1, \ldots, e_n\}$ where e_i is the $n \times 1$ vector with 1 as the i-th entry (*i.e.* in row i) and all other entries zero. *E.g*, for $n = 3$:

$$e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

For $X_0 = e_i \in B_n$ the Markov chain $(X_k)_{k \in \mathbb{N}^\star}$ specified by G and T is called the **random walk** on G starting at vertex i (or "at e_i"), with transition matrix T.

Then $X_k = T^k e_i = (q_j)_{1 \le j \le n}$ say gives, for $1 \le j \le n$, the probability q_j of being at the vertex j after k steps, starting from vertex i.

Steady States

Let $S = (q_j)_{1 \leq j \leq n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

Steady States

Let $S = (q_j)_{1 \leq j \leq n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk?

Steady States

Let $S = (q_j)_{1 \leq j \leq n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk? We have seen that a Markov process does not necessarily approach a steady state from every initial state.

Steady States

Let $S = (q_j)_{1 \leq j \leq n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk? We have seen that a Markov process does not necessarily approach a steady state from every initial state. However it can be proved that, for any initial probability vector X,

$$\frac{1}{N} \sum_{k=0}^{N-1} T^k X$$

approaches S as N gets large,

Steady States

Let $S = (q_j)_{1 \leq j \leq n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk? We have seen that a Markov process does not necessarily approach a steady state from every initial state. However it can be proved that, for any initial probability vector X,

$$\frac{1}{N} \sum_{k=0}^{N-1} T^k X$$

approaches S as N gets large, so, on average, the walk is at vertex j with probability q_j, *i.e.* $100q_j\%$ of the time is at j.

Steady States

Let $S = (q_j)_{1 \le j \le n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk? We have seen that a Markov process does not necessarily approach a steady state from every initial state. However it can be proved that, for any initial probability vector X,

$$\frac{1}{N} \sum_{k=0}^{N-1} T^k X$$

approaches S as N gets large, so, on average, the walk is at vertex j with probability q_j, *i.e.* $100q_j\%$ of the time is at j.

Idea for the Google algorithm: If G is the Web graph, then, for some suitable transition matrix T, q_j is the relative importance of the page j.

Steady States

Let $S = (q_j)_{1 \le j \le n} \in \mathbb{Q}^n$ be a stochastic vector, which is a steady state vector for T, *i.e.*

$$TS = S.$$

What does S represent in relation to our random walk? We have seen that a Markov process does not necessarily approach a steady state from every initial state. However it can be proved that, for any initial probability vector X,

$$\frac{1}{N} \sum_{k=0}^{N-1} T^k X$$

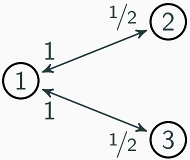
approaches S as N gets large, so, on average, the walk is at vertex j with probability q_j, *i.e.* $100q_j\%$ of the time is at j.

Idea for the Google algorithm: If G is the Web graph, then, for some suitable transition matrix T, q_j is the relative importance of the page j.
We will explore this idea later, but first some examples of random walks.

Example 1

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

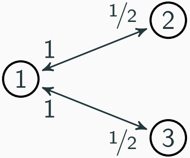
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$



Example 1

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$

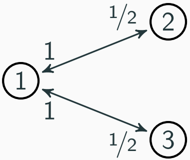


Our random walker alternates between vertex 1 and the other other two vertices, with neither of these two vertices being favoured over the other.

Example 1

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$



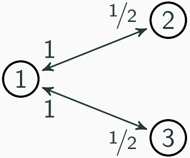
Our random walker alternates between vertex 1 and the other other two vertices, with neither of these two vertices being favoured over the other.

On average, the walker is at 1 half of the time and at 2, 3 a quarter of the time each, so the steady state vector is $S = \begin{bmatrix} 1/2 \\ 1/4 \\ 1/4 \end{bmatrix}$.

Example 1

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 1 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$



Our random walker alternates between vertex 1 and the other other two vertices, with neither of these two vertices being favoured over the other.

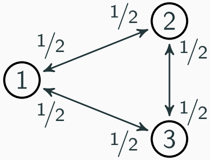
On average, the walker is at 1 half of the time and at 2, 3 a quarter of the time each, so the steady state vector is $S = \begin{bmatrix} 1/2 \\ 1/4 \\ 1/4 \end{bmatrix}$.

This is confirmed by checking that $TS = S$.

Example 2

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

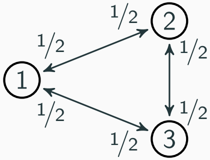
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$



Example 2

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$

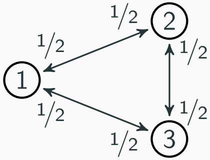


We see that no one vertex is favoured over any other.
On average, the walker will spend equal time at each vertex, so the steady state vector is $S = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$.

Example 2

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$



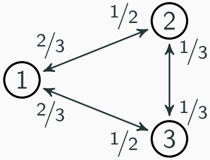
We see that no one vertex is favoured over any other.
On average, the walker will spend equal time at each vertex, so the steady state vector is $S = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$.

Again, this is confirmed by checking that $TS = S$.

Example 3

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

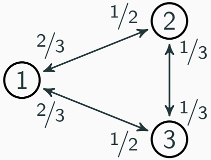
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 2/3 & 2/3 \\ 1/2 & 0 & 1/3 \\ 1/2 & 1/3 & 0 \end{bmatrix}$$



Example 3

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 2/3 & 2/3 \\ 1/2 & 0 & 1/3 \\ 1/2 & 1/3 & 0 \end{bmatrix}$$

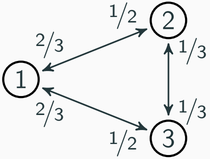


Now our random walker slightly favours vertex 1 over the other two, with neither of these other two being favoured over the other.

Example 3

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 2/3 & 2/3 \\ 1/2 & 0 & 1/3 \\ 1/2 & 1/3 & 0 \end{bmatrix}$$



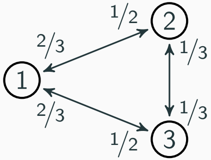
Now our random walker slightly favours vertex 1 over the other two, with neither of these other two being favoured over the other.

So the steady state vector should have the form $S = \begin{bmatrix} p \\ q \\ q \end{bmatrix}$.

Example 3

Consider a graph G with adjacency matrix A and a random walk on G with transition matrix T, where

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 2/3 & 2/3 \\ 1/2 & 0 & 1/3 \\ 1/2 & 1/3 & 0 \end{bmatrix}$$



Now our random walker slightly favours vertex 1 over the other two, with neither of these other two being favoured over the other.

So the steady state vector should have the form $S = \begin{bmatrix} p \\ q \\ q \end{bmatrix}$.

But what should the probabilities p and q be?
Can you guess?

Example 3 (concluded)

To find p and q we could use the method we used when investigating Markov chains earlier in the course.

Example 3 (concluded)

To find p and q we could use the method we used when investigating Markov chains earlier in the course.

That is, we could use the computer to solve the system of linear equations represented by the matrix equation $(T - I)S = 0$ modified by replacing all entries in the last row of both $T - I$ and $[0, 0, 0]'$ by 1.

Example 3 (concluded)

To find p and q we could use the method we used when investigating Markov chains earlier in the course.

That is, we could use the computer to solve the system of linear equations represented by the matrix equation $(T - I)S = 0$ modified by replacing all entries in the last row of both $T - I$ and $[0, 0, 0]'$ by 1.

Example 3 (concluded)

To find p and q we could use the method we used when investigating Markov chains earlier in the course.

That is, we could use the computer to solve the system of linear equations represented by the matrix equation $(T - I)S = 0$ modified by replacing all entries in the last row of both $T - I$ and $[0, 0, 0]'$ by 1.

On average, our walker spends 40% of the time at vertex 1 and 30% at each of the other two vertices.

Is that what you guessed?

Example 3 (concluded)

To find p and q we could use the method we used when investigating Markov chains earlier in the course.

That is, we could use the computer to solve the system of linear equations represented by the matrix equation $(T - I)S = 0$ modified by replacing all entries in the last row of both $T - I$ and $[0, 0, 0]'$ by 1.

On average, our walker spends 40% of the time at vertex 1 and 30% at each of the other two vertices.

Is that what you guessed?

The steady state vector is $S = \begin{bmatrix} 4/10 \\ 3/10 \\ 3/10 \end{bmatrix}$. As you can check, $TS = S$.

Webgraphs

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web.

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.
[**Source:** `http://en.wikipedia.org/wiki/Web_graph`]

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.
[**Source:** `http://en.wikipedia.org/wiki/Web_graph`]

We will use "webgraph" more generally to refer to any simple digraph.

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.
[**Source:** `http://en.wikipedia.org/wiki/Web_graph`]

We will use "webgraph" more generally to refer to any simple digraph.

PageRank is a link analysis algorithm, named after Larry Page, co-founder of Google, used by the Google Internet search engine.

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.

[**Source:** `http://en.wikipedia.org/wiki/Web_graph`]

We will use "webgraph" more generally to refer to any simple digraph.

PageRank is a link analysis algorithm, named after Larry Page, co-founder of Google, used by the Google Internet search engine. It assigns a numerical weighting to each element of a hyperlinked set of documents with the purpose of "measuring" its relative importance within the set. The numerical weight that it assigns to any given element E is referred to as the PageRank of E.

The Webgraph and PageRank

The **webgraph** describes the directed links between pages of the World Wide Web. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.

[**Source:** `http://en.wikipedia.org/wiki/Web_graph`]

We will use "webgraph" more generally to refer to any simple digraph.

PageRank is a link analysis algorithm, named after Larry Page, co-founder of Google, used by the Google Internet search engine. It assigns a numerical weighting to each element of a hyperlinked set of documents with the purpose of "measuring" its relative importance within the set. The numerical weight that it assigns to any given element E is referred to as the PageRank of E.

The name "PageRank" is a trademark of Google, and the PageRank process has been patented (U.S. Patent 6,285,999).

[**Source:** `http://en.wikipedia.org/wiki/PageRank`]

Rationale of PageRank

The idea is that the importance of a page is indicated by the number of links to it, modified by by the idea that links from important pages should carry more weight than links from less important pages.

Rationale of PageRank

The idea is that the importance of a page is indicated by the number of links to it, modified by by the idea that links from important pages should carry more weight than links from less important pages.

Of course this idea is recursive (self-referential) and the first problem is how to make it into an operational definition.

Rationale of PageRank

The idea is that the importance of a page is indicated by the number of links to it, modified by by the idea that links from important pages should carry more weight than links from less important pages.

Of course this idea is recursive (self-referential) and the first problem is how to make it into an operational definition.

Larry Page solved this problem by defining the importance of page j as p_j in the steady state vector S for a random walk on the webgraph using a specially designed transition matrix M.

Rationale of PageRank

The idea is that the importance of a page is indicated by the number of links to it, modified by by the idea that links from important pages should carry more weight than links from less important pages.

Of course this idea is recursive (self-referential) and the first problem is how to make it into an operational definition.

Larry Page solved this problem by defining the importance of page j as p_j in the steady state vector S for a random walk on the webgraph using a specially designed transition matrix M.

We will look at the design of M shortly, but first an example using a baby web with only 11 pages and a total of 17 hyperlinks.

Rationale of PageRank

The idea is that the importance of a page is indicated by the number of links to it, modified by by the idea that links from important pages should carry more weight than links from less important pages.

Of course this idea is recursive (self-referential) and the first problem is how to make it into an operational definition.

Larry Page solved this problem by defining the importance of page j as p_j in the steady state vector S for a random walk on the webgraph using a specially designed transition matrix M.

We will look at the design of M shortly, but first an example using a baby web with only 11 pages and a total of 17 hyperlinks.

In the following diagram the sizes of the vertices indicate their importance, as calculated by the PageRank algorithm. The only input to the algorithm was the digraph itself plus a 'damping' factor of 85%, to be discussed later.

Basic transition probabilities

The PageRank algorithm assumes that our random walker (random *surfer*!) is equally likely to follow any link on a page, and, if there are no links, is equally likely to 'teleport' to any other page on the web.

Basic transition probabilities

The PageRank algorithm assumes that our random walker (random *surfer*!) is equally likely to follow any link on a page, and, if there are no links, is equally likely to 'teleport' to any other page on the web.

Formally, for any webgraph G we construct G^+ by adding to edges from any vertex that has no links to all other vertices (we remove the sinks). Let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is adjacent in G^+, that is

$$\begin{aligned} n &= |V(G^+)| \\ n_i &= |\{j \; : \; (i,j) \in E(G^+)\}| \end{aligned}$$

Basic transition probabilities

The PageRank algorithm assumes that our random walker (random *surfer*!) is equally likely to follow any link on a page, and, if there are no links, is equally likely to 'teleport' to any other page on the web.

Formally, for any webgraph G we construct G^+ by adding to edges from any vertex that has no links to all other vertices (we remove the sinks). Let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is adjacent in G^+, that is

$$\begin{aligned} n &= |V(G^+)| \\ n_i &= |\{j \,:\, (i,j) \in E(G^+)\}| \end{aligned}$$

Then the basic probability $p_{i\,j}$ of a transition from vertex j to i is given by

$$p_{i\,j} = \begin{cases} {}^1/_{n_i} & \text{if } n_i \neq 0 \text{ and } (j,i) \in E(G^+) \\ 0 & \text{otherwise} \end{cases}$$

Basic transition probabilities

The PageRank algorithm assumes that our random walker (random *surfer*!) is equally likely to follow any link on a page, and, if there are no links, is equally likely to 'teleport' to any other page on the web.

Formally, for any webgraph G we construct G^+ by adding to edges from any vertex that has no links to all other vertices (we remove the sinks). Let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is adjacent in G^+, that is

$$\begin{aligned} n &= |V(G^+)| \\ n_i &= |\{j \ : \ (i,j) \in E(G^+)\}| \end{aligned}$$

Then the basic probability $p_{i\,j}$ of a transition from vertex j to i is given by

$$p_{i\,j} = \begin{cases} 1/n_i & \text{if } n_i \neq 0 \text{ and } (j,i) \in E(G^+) \\ 0 & \text{otherwise} \end{cases}$$

The basic transition matrix is $T = (p_{i\,j})_{1 \leq i,j \leq n}$.

Example 4A

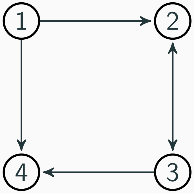
Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

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

Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

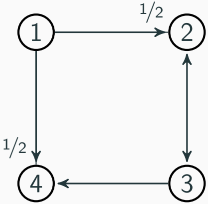
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

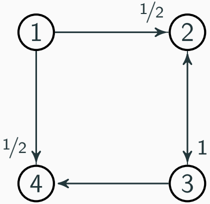
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

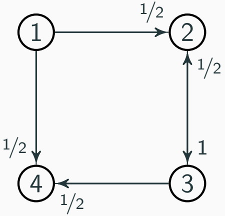
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

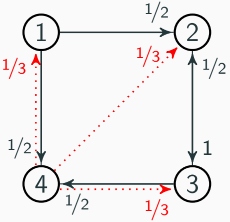
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

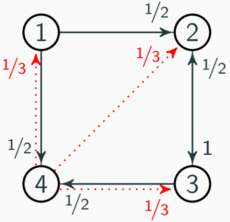
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

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

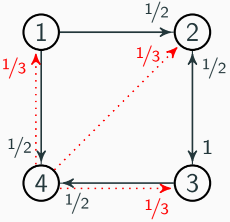


$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix}$$

Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

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



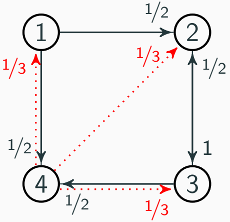
$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix}$$

Solving, by computer, $(T - I)S = 0$ with the usual shortcut method, gives steady state solution $S = \dfrac{1}{13}\begin{bmatrix} 1 \\ 4 \\ 5 \\ 3 \end{bmatrix} \approx \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$.

Example 4A

Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

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



$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix}$$

Solving, by computer, $(T - I)S = 0$ with the usual shortcut method, gives steady state solution $S = \dfrac{1}{13}\begin{bmatrix} 1 \\ 4 \\ 5 \\ 3 \end{bmatrix} \approx \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$.

So on this basis, vertex 3 is most important and vertex 1 least.

Damping

The damping factor $(1-\alpha)$

The PageRank algorithm assumes that, at any time k, there is a small probability α that, irrespective of what links are available at the current page, the surfer chooses to teleport randomly to any page on the web;
i.e. the surfer acts as if there were *no* links from the current page.

The damping factor $(1 - \alpha)$

The PageRank algorithm assumes that, at any time k, there is a small probability α that, irrespective of what links are available at the current page, the surfer chooses to teleport randomly to any page on the web;
i.e. the surfer acts as if there were *no* links from the current page.

For convenience, we include the current page amongst the possibilities for this random choice, so that the probability that the surfer takes the teleport option and lands on any particular page is $\alpha(1/n) = \alpha/n$.

The damping factor $(1-\alpha)$

The PageRank algorithm assumes that, at any time k, there is a small probability α that, irrespective of what links are available at the current page, the surfer chooses to teleport randomly to any page on the web;
i.e. the surfer acts as if there were *no* links from the current page.

For convenience, we include the current page amongst the possibilities for this random choice, so that the probability that the surfer takes the teleport option and lands on any particular page is $\alpha(1/n) = \alpha/n$.

Correspondingly, at any time k, the probability that the surfer takes the standard 'non-teleport' option is $(1-\alpha)$, thus reducing the basic probabilities $p_{i\,j}$ by a **damping factor** $(1-\alpha)$.

The damping factor $(1-\alpha)$

The PageRank algorithm assumes that, at any time k, there is a small probability α that, irrespective of what links are available at the current page, the surfer chooses to teleport randomly to any page on the web;
i.e. the surfer acts as if there were *no* links from the current page.

For convenience, we include the current page amongst the possibilities for this random choice, so that the probability that the surfer takes the teleport option and lands on any particular page is $\alpha({}^1\!/_n) = {}^\alpha\!/_n$.

Correspondingly, at any time k, the probability that the surfer takes the standard 'non-teleport' option is $(1-\alpha)$, thus reducing the basic probabilities p_{ij} by a **damping factor** $(1-\alpha)$.

Thus the modified probability for transition from vertex j to i is

$$m_{ij} = {}^\alpha\!/_n + (1-\alpha)p_{ij}.$$

The damping factor $(1-\alpha)$

The PageRank algorithm assumes that, at any time k, there is a small probability α that, irrespective of what links are available at the current page, the surfer chooses to teleport randomly to any page on the web;
i.e. the surfer acts as if there were *no* links from the current page.

For convenience, we include the current page amongst the possibilities for this random choice, so that the probability that the surfer takes the teleport option and lands on any particular page is $\alpha(1/n) = \alpha/n$.

Correspondingly, at any time k, the probability that the surfer takes the standard 'non-teleport' option is $(1-\alpha)$, thus reducing the basic probabilities p_{ij} by a **damping factor** $(1-\alpha)$.

Thus the modified probability for transition from vertex j to i is

$$m_{ij} = \alpha/n + (1-\alpha)p_{ij}.$$

In practice, Google uses a damping factor of 85%, *i.e.* $\alpha = 0.15$.

What are we doing?

We start with a web graph G (directed graph). At each vertex that is a sink in G we add edges to all other vertices to get G^+. We then say that T is our **basic transition matrix** for the webgraph G.

What are we doing?

We start with a web graph G (directed graph). At each vertex that is a sink in G we add edges to all other vertices to get G^+. We then say that T is our **basic transition matrix** for the webgraph G.

By adding in the damping factor, in essence we take our G^+ and construct a complete directed graph with loops $G^+_{Complete}$ with probabilities corresponding to the **modified transition matrix** $M = (\alpha/n)U + (1-\alpha)T$, where U is the $n \times n$ all-1's matrix. We say that M is the modified transition matrix for the webgraph G.

What are we doing?

We start with a web graph G (directed graph). At each vertex that is a sink in G we add edges to all other vertices to get G^+. We then say that T is our **basic transition matrix** for the webgraph G.

By adding in the damping factor, in essence we take our G^+ and construct a complete directed graph with loops $G^+_{Complete}$ with probabilities corresponding to the **modified transition matrix** $M = (\alpha/n)U + (1-\alpha)T$, where U is the $n \times n$ all-1's matrix. We say that M is the modified transition matrix for the webgraph G.

The damping factor allows the surfer to sometimes jump to another page without requiring a link in G or to be at a sink of G.

What are we doing?

We start with a web graph G (directed graph). At each vertex that is a sink in G we add edges to all other vertices to get G^+. We then say that T is our **basic transition matrix** for the webgraph G.

By adding in the damping factor, in essence we take our G^+ and construct a complete directed graph with loops $G^+_{Complete}$ with probabilities corresponding to the **modified transition matrix** $M = (\alpha/n)U + (1-\alpha)T$, where U is the $n \times n$ all-1's matrix. We say that M is the modified transition matrix for the webgraph G.

The damping factor allows the surfer to sometimes jump to another page without requiring a link in G or to be at a sink of G.

So without damping we do a random walk on G^+ assigning probabilities in a particular way (equal likely for each edge from the vertex). Now with the damping we do a random walk on $G^+_{Complete}$ assigning probabilities to the edges as seen in M above.

The modified transition matrix M and PageRank vector PR

The modified transition probabilities lead to a modified transition matrix

$$M = (m_{ij})_{1 \le i,j \le n} \quad = \quad ({}^{\alpha}\!/_{n} + (1-\alpha)p_{ij})_{1 \le i,j \le n}$$

The modified transition matrix M and PageRank vector PR

The modified transition probabilities lead to a modified transition matrix

$$\begin{aligned} M = (m_{ij})_{1 \le i,j \le n} &= ({}^{\alpha}/_{n} + (1-\alpha)p_{ij})_{1 \le i,j \le n} \\ &= ({}^{\alpha}/_{n})U + (1-\alpha)T, \end{aligned}$$

where U is the $n \times n$ all-1's matrix and T is the basic transition matrix.

The modified transition matrix M and PageRank vector $\mathcal{PR}$

The modified transition probabilities lead to a modified transition matrix

$$\begin{aligned} M = (m_{ij})_{1\leq i,j\leq n} &= (\alpha/n + (1-\alpha)p_{ij})_{1\leq i,j\leq n} \\ &= (\alpha/n)U + (1-\alpha)T, \end{aligned}$$

where U is the $n\times n$ all-1's matrix and T is the basic transition matrix.

As indicated earlier, the PageRank algorithm defines the rank of page i of the webgraph to be the i-th entry in the **PageRank vector** $\mathcal{PR}$, which in turn is defined as the steady state vector for the random walk on the webgraph with transition matrix M.

The modified transition matrix M and PageRank vector PR

The modified transition probabilities lead to a modified transition matrix

$$\begin{aligned} M = (m_{ij})_{1\le i,j\le n} &= (\alpha/n + (1-\alpha)p_{ij})_{1\le i,j\le n} \\ &= (\alpha/n)U + (1-\alpha)T, \end{aligned}$$

where U is the $n \times n$ all-1's matrix and T is the basic transition matrix.

As indicated earlier, the PageRank algorithm defines the rank of page i of the webgraph to be the i-th entry in the **PageRank vector** PR, which in turn is defined as the steady state vector for the random walk on the webgraph with transition matrix M.

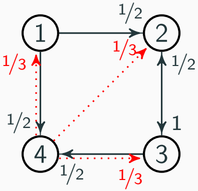
Thus PR is defined as the probability vector solution to the equation

$$M\,PR = PR.$$

Example 4B

For example 4A we had:

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

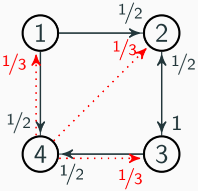


$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix} \qquad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

Example 4B

For example 4A we had:

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



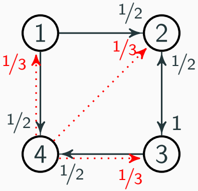
$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix} \qquad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

Let's see what happens if we apply a damping factor of 90%; *i.e.* $\alpha = 0.1$.

Example 4B

For example 4A we had:

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



$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix} \qquad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

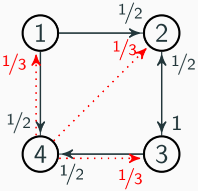
Let's see what happens if we apply a damping factor of 90%; *i.e.* $\alpha = 0.1$.

We need to solve the equation $MPR = PR$ where:

Example 4B

For example 4A we had:

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



$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix} \qquad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

Let's see what happens if we apply a damping factor of 90%; *i.e.* $\alpha = 0.1$.

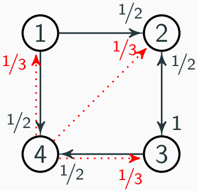
We need to solve the equation $MPR = PR$ where:

$$M = (0.1/4)U + (1 - 0.1)T =$$

$$= \begin{bmatrix} 0.025 & 0.025 & 0.025 & 0.325 \\ 0.475 & 0.025 & 0.475 & 0.325 \\ 0.025 & 0.925 & 0.025 & 0.325 \\ 0.475 & 0.025 & 0.475 & 0.025 \end{bmatrix}$$

For example 4A we had:

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



$$T = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 0 & 1 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix} \qquad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

Let's see what happens if we apply a damping factor of 90%; *i.e.* $\alpha = 0.1$.

We need to solve the equation $M\,PR = PR$ where:

$$M = (0.1/4)U + (1 - 0.1)T =$$

$$= \begin{bmatrix} 0.025 & 0.025 & 0.025 & 0.325 \\ 0.475 & 0.025 & 0.475 & 0.325 \\ 0.025 & 0.925 & 0.025 & 0.325 \\ 0.475 & 0.025 & 0.475 & 0.025 \end{bmatrix}$$

Using the shortcut method we find that $PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2dp.

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$$

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

The PageRank of vertex 1 increases because it now has teleporting 'inputs' from all vertices, not just vertex 4.

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

The PageRank of vertex 1 increases because it now has teleporting 'inputs' from all vertices, not just vertex 4. This increase is at the expense of the stronger vertices 2 and 3.

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

The PageRank of vertex 1 increases because it now has teleporting 'inputs' from all vertices, not just vertex 4. This increase is at the expense of the stronger vertices 2 and 3. Vertex 4 gains about as much as it loses.

Example 4B (cont.)

Comparing the Page rank to the steady state without damping we have:

$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

The PageRank of vertex 1 increases because it now has teleporting 'inputs' from all vertices, not just vertex 4. This increase is at the expense of the stronger vertices 2 and 3. Vertex 4 gains about as much as it loses.

This is what you expect with damping.

Plan for next 2 weeks

Week 10 Lecture 3: Finish off Google page rank with more complex examples.

Plan for next 2 weeks

Week 10 Lecture 3: Finish off Google page rank with more complex examples.

Week 11: Work through a past exam paper. I will also post another exam paper to Wattle with full solutions.

Plan for next 2 weeks

Week 10 Lecture 3: Finish off Google page rank with more complex examples.

Week 11: Work through a past exam paper. I will also post another exam paper to Wattle with full solutions.

Week 12: Cover some proofs and interesting applications of what we have learnt in this course.

Plan for next 2 weeks

Week 10 Lecture 3: Finish off Google page rank with more complex examples.

Week 11: Work through a past exam paper. I will also post another exam paper to Wattle with full solutions.

Week 12: Cover some proofs and interesting applications of what we have learnt in this course.

Final exam on the 2nd November, which is a Saturday!

Plan for next 2 weeks

Week 10 Lecture 3: Finish off Google page rank with more complex examples.

Week 11: Work through a past exam paper. I will also post another exam paper to Wattle with full solutions.

Week 12: Cover some proofs and interesting applications of what we have learnt in this course.

Final exam on the 2nd November, which is a Saturday!

I will announce some extra consultation hours for the lead up to the exam.