Recap: Weighted graphs: Traveling salesman problem
Shortest path ~ Dijkstra's algorithm
Minimal spanning trees
Transport networks (max. flow / min. " cut)

D3. Random walks on graphs.

Notes by Malcolm Brooks,
expanded from notes of Pierre Portal
with influences from Judy-anne Osborn .

Unfortunately, Random walks are not covered in our text by Epp, nor in the books by Johnsonbaugh or Kolman *et. al.*

Announcements: Poll to appear on Wattle to select review lecture topics. (for next week)

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_{ij}.
[If $(i, j) \notin E$, then, of course, $p_{ij} = 0$.]

Random walks: definition

Associated with an n-vertex directed graph G, let $T = (p_{ij})_{1 \leq i,j \leq n})$ be an $n \times n$ stochastic matrix satisfying $p_{i,j} = 0 \ \forall (i,j) \notin E(G)$.

Random walks: definition

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

For any given n let B_n denote the set of **basis vectors** $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ where $\mathbf{e}_i$ 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 $n = 3$:

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

Random walks: definition

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

For any given n let B_n denote the set of **basis vectors** $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ where $\mathbf{e}_i$ 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 $n = 3$: $\mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

For $X_0 = \mathbf{e}_i \in B_n$ 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 $\mathbf{e}_i$"), with transition matrix T.

Random walks: definition

Associated with an n-vertex directed graph G, let $T = (p_{i\,j})_{1\le i,j\le n})$ be an $n \times n$ stochastic matrix satisfying $p_{i,j} = 0\ \forall (i,j) \notin E(G)$.

For any given n let B_n denote the set of **basis vectors** $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ where $\mathbf{e}_i$ 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 $n = 3$: $\mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$

For $X_0 = \mathbf{e}_i \in B_n$ 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 $\mathbf{e}_i$"), with transition matrix T.

Then $X_k = (T')^k \mathbf{e}_i = (q_j)_{1\le j\le n}$ 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.*

$$T'S = S.$$

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.*

$$T'S = S.$$

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.*

$$T'S = S.$$

i.e. $T^k X \not\to S$ for any X.

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,

$$\underbrace{\frac{1}{N}\sum_{k=0}^{N-1}}_{\text{average over time}} \underbrace{T^k X}_{\text{state at time } k.}$$

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.*

$$T'S = S.$$

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.* is at j $100q_j\%$ of the time.

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.*

$$T'S = S.$$

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.* is at j $100q_j\%$ of the time.

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.*

$$T'S = S.$$

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,

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

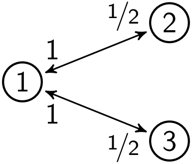
approaches S as N gets large, so, on average, the walk is at vertex j with probability q_j, *i.e.* is at j $100q_j$% of the time.

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

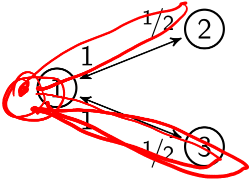
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$



Example 1

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

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

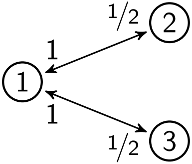


Our random walker alternates between vertex 1 and the other other two vertices, with neither of these two vertices being favoured over the other.

Example 1

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

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



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

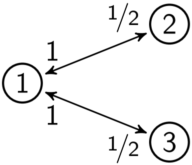
On average, the walker is at 1 half of the time and at 2, 3 a quarter of the time each, so the steady state vector is $S = \begin{bmatrix} 1/2 \\ 1/4 \\ 1/4 \end{bmatrix}$.

Example 1

$$T'S = \begin{bmatrix} 0 & 1 & 1 \\ \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} \\ \frac{1}{4} \\ \frac{1}{4} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{4} \\ \frac{1}{4} \end{bmatrix} = S$$

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

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



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

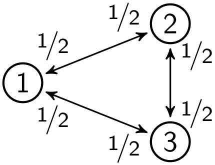
On average, the walker is at 1 half of the time and at 2, 3 a quarter of the time each, so the steady state vector is $S = \begin{bmatrix} 1/2 \\ 1/4 \\ 1/4 \end{bmatrix}$.

This is confirmed by checking that $T'S = S$.

Example 2

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

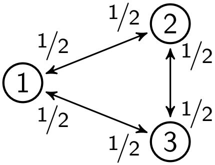
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$



Example 2

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

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

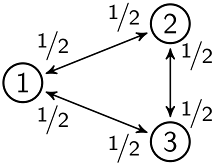


We see that no one vertex is favoured over any other.
On average, the walker will spend equal time at each vertex, so the steady state vector is $S = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$.

Example 2

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

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



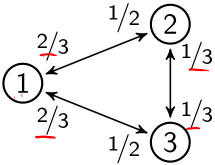
We see that no one vertex is favoured over any other.
On average, the walker will spend equal time at each vertex, so the steady state vector is $S = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$.

Again, this is confirmed by checking that $T'S = S$.

Example 3

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

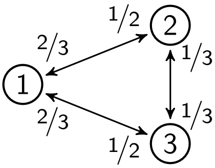
$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 2/3 & 0 & 1/3 \\ 2/3 & 1/3 & 0 \end{bmatrix}$$



Example 3

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

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

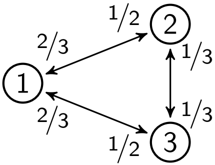


Now our random walker slightly favours vertex 1 over the other two, with neither of these other two being favoured over the other.

Example 3

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

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



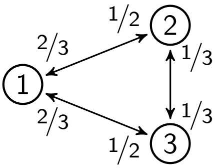
Now our random walker slightly favours vertex 1 over the other two, with neither of these other two being favoured over the other.

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

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



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 $S = \begin{bmatrix} p \\ q \\ q \end{bmatrix}$. steady state

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 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 by 1.

However, in this case the equations boil down to:

$$\begin{aligned} -p + \tfrac{4}{3}q &= 0 \\ p + 2q &= 1 \end{aligned}.$$

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 by 1.

However, in this case the equations boil down to: $\begin{array}{rcl} -p + \frac{4}{3}q & = & 0 \\ p + 2q & = & 1 \end{array}$.

Thus $\begin{bmatrix} -1 & \frac{4}{3} \\ 1 & 2 \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, so $\begin{bmatrix} p \\ q \end{bmatrix} = \dfrac{1}{-10/3} \begin{bmatrix} 2 & -(\frac{4}{3}) \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4/10 \\ 3/10 \end{bmatrix}$.

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 by 1.

However, in this case the equations boil down to: $\begin{array}{rcl} -p + \frac{4}{3}q & = & 0 \\ p + 2q & = & 1 \end{array}.$

Thus $\begin{bmatrix} -1 & \frac{4}{3} \\ 1 & 2 \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, so $\begin{bmatrix} p \\ q \end{bmatrix} = \dfrac{1}{-10/3} \begin{bmatrix} 2 & -(\frac{4}{3}) \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4/10 \\ 3/10 \end{bmatrix}$.

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 by 1.

However, in this case the equations boil down to: $\begin{array}{rcl} -p + \frac{4}{3}q & = & 0 \\ p + 2q & = & 1 \end{array}.$

Thus $\begin{bmatrix} -1 & \frac{4}{3} \\ 1 & 2 \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, so $\begin{bmatrix} p \\ q \end{bmatrix} = \dfrac{1}{-10/3} \begin{bmatrix} 2 & -(\frac{4}{3}) \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4/10 \\ 3/10 \end{bmatrix}$.

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, $T'S = S$.

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 at least one 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 (and became mega rich) 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 (and became mega rich) 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.

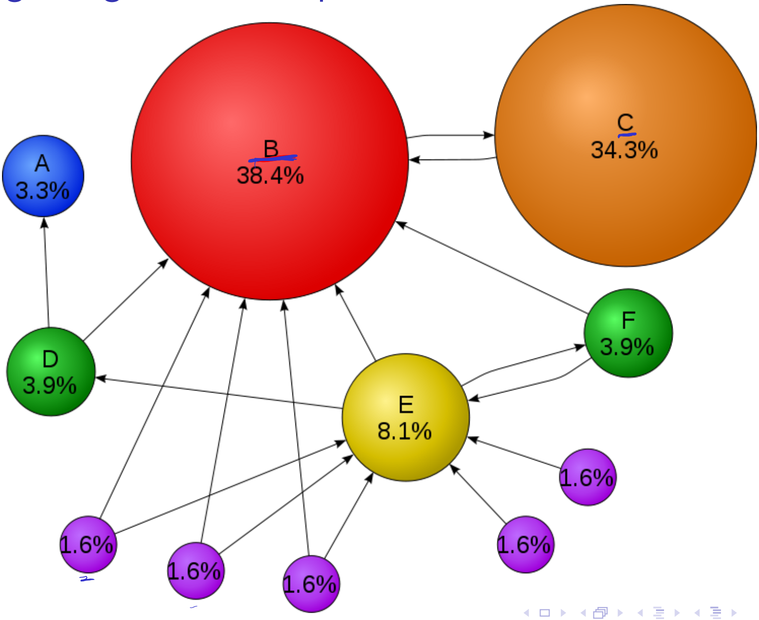
Larry Page solved this problem (and became mega rich) by defining the importance of page j as p_j in the steady state vector S for a random walk on the webgraph using a specially designed transition matrix M.

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.

Google PageRank Example

http://en.wikipedia.org/wiki/PageRank



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.

1

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 let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is linked:

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

1

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 let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is linked:

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

links from page

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

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

[1] When n is large (as in the WWW) this line can be simplified to: $1/n$ if $n_i = 0$.

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 let n be the number of vertices (pages) and for each vertex i let n_i be the number vertices to which i is linked:

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

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

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

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

---

[1] When n is large (as in the WWW) this line can be simplified to: $1/n$ if $n_i = 0$.

Example 4A

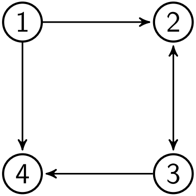
Here is a tiny example of basic transition probabilities - with just $n = 4$ vertices :

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

Example 4A

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

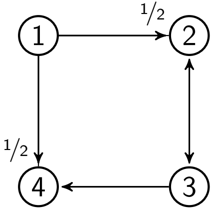
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

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

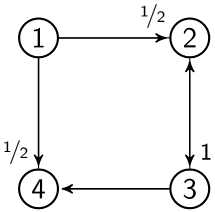
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

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

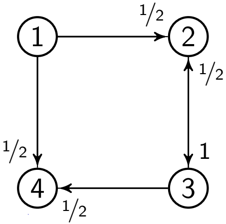
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

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

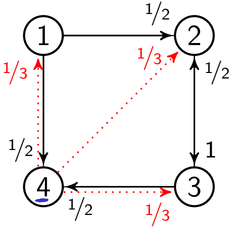
$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



Example 4A

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

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



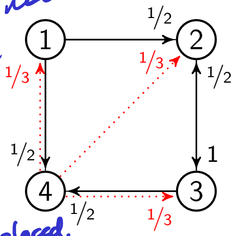
Example 4A

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

need to be scaled to make a transition matrix

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

needs to be replaced.

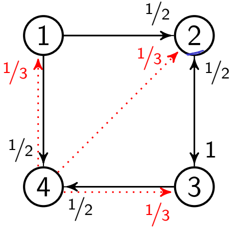


$$T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$$

Example 4A

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

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



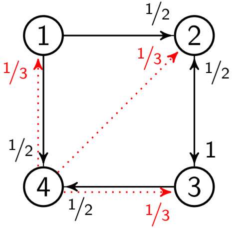
$$T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$$

Solving, by computer, $(T' - I)S = 0$ with the usual replacement last equation, 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 :

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



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

Solving, by computer, $(T' - I)S = 0$ with the usual replacement last equation, 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.

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 i to j is

$$m_{ij} = \underbrace{\alpha/n}_{\text{random jump}} + (1 - \alpha)\underbrace{p_{ij}}_{\text{follow hyperlinks.}}.$$

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 i to j is

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

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

The modified transition matrix M and PageRank vector PR

The modified transition probabilities lead to a modified transition matrix

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

The modified transition matrix M and PageRank vector PR

The modified transition probabilities lead to a modified transition matrix

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

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

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

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** $\underset{\sim}{\mathcal{PR}}$, 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

$$M' PR = PR.$$

Calculating PR

Expanding the defining equation $M' PR = PR$ gives

$$
\begin{aligned}
PR &= ((\alpha/n)U + (1-\alpha)T')PR \qquad \text{since } U' = U \\
&= (\alpha/n)UPR + (1-\alpha)T' \; PR \qquad (\star\star)
\end{aligned}
$$

$$
U\,PR = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & & \\ 1 & & \ddots & \\ \vdots & & & \end{bmatrix} \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} = \begin{bmatrix} p_1 + \dots + p_n \\ \vdots \\ p_1 + \dots + p_n \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}
$$

Calculating PR

Expanding the defining equation $M'PR = PR$ gives

$$
\begin{aligned}
PR &= ((\alpha/n)U + (1-\alpha)T')PR \qquad \text{since } U' = U \\
&= (\alpha/n)UPR + (1-\alpha)T')PR \quad (\star\star)
\end{aligned}
$$

Now each entry of the product UPR is the sum of all the entries in PR, and since PR is a probability vector that sum is 1. Hence $UPR = \mathbf{1}$ where $\mathbf{1}$ is the $n \times 1$ vector of all 1's.

Calculating $\mathcal{P}R$

Expanding the defining equation $M'\mathcal{P}R = \mathcal{P}R$ gives

$$\begin{aligned} \mathcal{P}R &= ((\alpha/n)U + (1-\alpha)T')\mathcal{P}R \qquad \text{since } U' = U \\ &= (\alpha/n)U\mathcal{P}R + (1-\alpha)T')PR \quad (\star\star) \\ &= \alpha/n \, \mathbf{1} + (1-\alpha)T'\mathcal{P}R \end{aligned}$$

Now each entry of the product $U\mathcal{P}R$ is the sum of all the entries in $\mathcal{P}R$, and since $\mathcal{P}R$ is a probability vector that sum is 1. Hence $U\mathcal{P}R = \mathbf{1}$ where $\mathbf{1}$ is the $n \times 1$ vector of all 1's. So equation $(\star\star)$ can be rearranged as

$$\boxed{(I - (1-\alpha)T')\mathcal{P}R = (\alpha/n)\mathbf{1}}$$

(subtract $(1-\alpha)T'\mathcal{P}R$ from both sides)

Calculating PR

Expanding the defining equation $M'PR = PR$ gives

$$\begin{aligned} PR &= ((\alpha/n)U + (1-\alpha)T')PR \qquad \text{since } U' = U \\ &= (\alpha/n)UPR + (1-\alpha)T')PR \quad (\star\star) \end{aligned}$$

Now each entry of the product UPR is the sum of all the entries in PR, and since PR is a probability vector that sum is 1. Hence $UPR = \mathbf{1}$ where $\mathbf{1}$ is the $n \times 1$ vector of all 1's. So equation $(\star\star)$ can be rearranged as

$$\boxed{(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}}$$

For small to moderate n this equation can be solved directly (by computer).

Calculating PR

Expanding the defining equation $M'PR = PR$ gives

$$\begin{aligned} PR &= ((\alpha/n)U + (1-\alpha)T')PR \qquad \text{since } U' = U \\ &= (\alpha/n)UPR + (1-\alpha)T')PR \quad (\star\star) \end{aligned}$$

Now each entry of the product UPR is the sum of all the entries in PR, and since PR is a probability vector that sum is 1. Hence $UPR = \mathbf{1}$ where $\mathbf{1}$ is the $n \times 1$ vector of all 1's. So equation $(\star\star)$ can be rearranged as

$$\boxed{\underbrace{(I - (1-\alpha)T')}_{\text{invertible if } \alpha \neq 0} PR = (\alpha/n)\mathbf{1}}$$

For small to moderate n this equation can be solved directly (by computer).

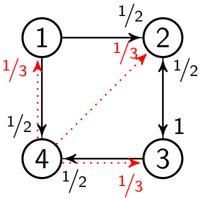
Notes:

- When $\alpha = 0$ this equation reverts to the basic steady state equation.
- When $\alpha \neq 0$ the fact that we have used $UPR = \mathbf{1}$ means that it is no longer necessary, nor appropriate, to replace the last row of this matrix equation by all 1's, as in the case of $\alpha = 0$.

Example 4B

For example 4A we had:

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

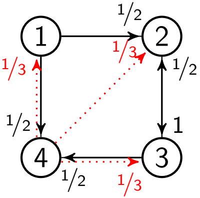


$$T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

Example 4B

For example 4A we had:

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



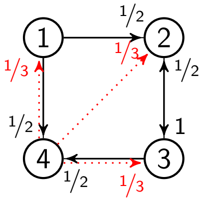
$$T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix}$$

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:

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



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

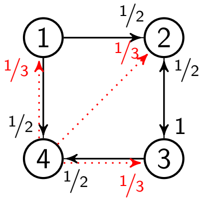
Let's see what happens if we apply a damping factor of 90%; *i.e.* $\alpha = 0.1$.

We need to solve the equation $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ where:

Example 4B

For example 4A we had:

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



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

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

We need to solve the equation $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ where:

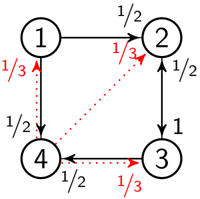
$$(I - (1-\alpha)T') = I - (0.9)T'$$

$$= \begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix}$$

Example 4B

For example 4A we had:

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



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

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

We need to solve the equation $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ where:

$$(I - (1-\alpha)T') = I - (0.9)T'$$

$$= \begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix}$$

$$(\alpha/n)\mathbf{1} = \frac{0.1}{4}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$

Example 4B (cont.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$

on the computer gives,

Example 4B (cont.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$

on the computer gives, to two decimal places,

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

Example 4B (cont.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$

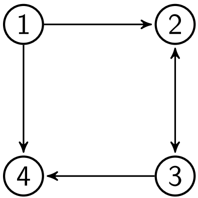
on the computer gives, to two decimal places,

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

Example 4B (cont.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$



on the computer gives, to two decimal places,

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

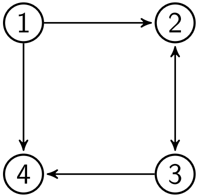
The PageRank of vertex 1 increases because it now has teleporting ‘inputs’ from all vertices, not just vertex 4.

PR is a "smoothed out" version of S.

Example 4B (cont.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$



on the computer gives, to two decimal places,

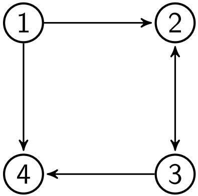
$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

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.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$



on the computer gives, to two decimal places,

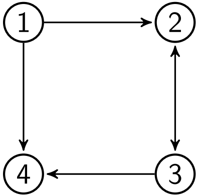
$$PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix} \quad \text{compared to} \quad S = \begin{bmatrix} .08 \\ .31 \\ .38 \\ .23 \end{bmatrix} \quad \text{without damping.}$$

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.)

Solving the equation

$$\begin{bmatrix} 1 & 0 & 0 & -0.3 \\ -0.45 & 1 & -0.45 & -0.3 \\ 0 & -0.9 & 1 & -0.3 \\ -0.45 & 0 & -0.45 & 1 \end{bmatrix} PR = \begin{bmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{bmatrix}$$

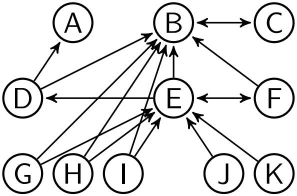


on the computer gives, to two decimal places,

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

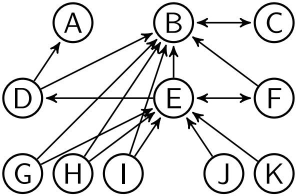
The PageRank of vertex 1 increases because it now has teleporting 'inputs' from all vertices, not just vertex 4. This increase is at the expense of the stronger vertices 2 and 3. Vertex 4 gains about as much as it loses. This is what you expect with damping.

Example 5



At left is the Wikipedia example we saw earlier of a miniweb of 11 pages and 17 hyperlinks. Colours, variable sizes and PageRanks have been removed and the bottom five vertices have been labelled G to K, following the given labelling of the top six vertices. The layout is similar to that in the original.

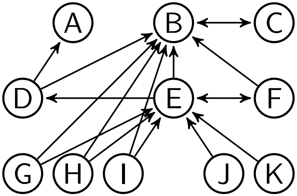
Example 5



At left is the Wikipedia example we saw earlier of a miniweb of 11 pages and 17 hyperlinks. Colours, variable sizes and PageRanks have been removed and the bottom five vertices have been labelled G to K, following the given labelling of the top six vertices. The layout is similar to that in the original.

Using the steady state method we have been discussing, we will derive the PageRanks given on the Wikipedia diagram.

Example 5

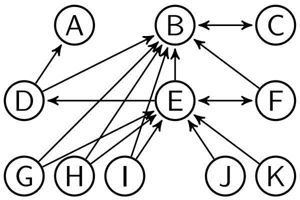


At left is the Wikipedia example we saw earlier of a miniweb of 11 pages and 17 hyperlinks. Colours, variable sizes and PageRanks have been removed and the bottom five vertices have been labelled G to K, following the given labelling of the top six vertices. The layout is similar to that in the original.

Using the steady state method we have been discussing, we will derive the PageRanks given on the Wikipedia diagram.

Step 1: Compile the adjacency matrix A.

Example 5



At left is the Wikipedia example we saw earlier of a miniweb of 11 pages and 17 hyperlinks. Colours, variable sizes and PageRanks have been removed and the bottom five vertices have been labelled G to K, following the given labelling of the top six vertices. The layout is similar to that in the original.

Using the steady state method we have been discussing, we will derive the PageRanks given on the Wikipedia diagram.

Step 1: Compile the adjacency matrix A.

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

Example 5 (cont.)

Step 2: Compile the basic transition matrix T.

Example 5 (cont.)

Step 2: Compile the basic transition matrix T.

In each row i of A, count the number n_i of 1's in the row then:

- If $n_i \neq 0$ replace each 1 with ${}^{1}\!/_{n_i}$.

Example 5 (cont.)

Step 2: Compile the basic transition matrix T.

In each row i of A, count the number n_i of 1's in the row then:

- If $n_i \neq 0$ replace each 1 with $1/n_i$.
- If $n_i = 0$ then
  - if n (the total number of pages) is small (less than 10 say) replace all but the i-th (diagonal) entry by $1/(n\text{-}1)$ (we did this in Example 4B)
  - but for $n \geq 10$ (as here and for WWW) replace *every* entry by $1/n$.

Example 5 (cont.)

Step 2: Compile the basic transition matrix T.

In each row i of A, count the number n_i of 1's in the row then:

- If $n_i \neq 0$ replace each 1 with $1/n_i$.
- If $n_i = 0$ then
  - if n (the total number of pages) is small (less than 10 say) replace all but the i-th (diagonal) entry by $1/(n-1)$ (we did this in Example 4B)
  - but for $n \geq 10$ (as here and for WWW) replace *every* entry by $1/n$.

For our Wikipedia example we get

$$T = \begin{bmatrix}
1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 & 1/11 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Example 5 (cont.)

Step 3: Compile the matrix $(I - (1 - \alpha)T')$

Example 5 (cont.)

Step 3: Compile the matrix $(I - (1-\alpha)T')$
For each row i of T

1. multiply each entry by $(1-\alpha)$ and put the *negative* of this in the corresponding position in the i-th *column* of $(I - (1-\alpha)T')$;
2. add 1 to each diagonal entry of the resulting matrix.

Example 5 (cont.)

Step 3: Compile the matrix $(I - (1-\alpha)T')$
For each row i of T

1. multiply each entry by $(1-\alpha)$ and put the *negative* of this in the corresponding position in the i-th *column* of $(I - (1-\alpha)T')$;
2. add 1 to each diagonal entry of the resulting matrix.

Google uses an 85% damping factor, so we set $(1-\alpha) = 0.85$.

Example 5 (cont.)

Step 3: Compile the matrix $(I-(1-\alpha)T')$

For each row i of T

1. multiply each entry by $(1-\alpha)$ and put the *negative* of this in the corresponding position in the i-th *column* of $(I-(1-\alpha)T')$;
2. add 1 to each diagonal entry of the resulting matrix.

Google uses an 85% damping factor, so we set $(1-\alpha)=0.85$.

$$(I-(1-\alpha)T')=$$

$$\begin{bmatrix}
.92\overline{27} & 0 & 0 & -.425 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-.07\overline{72} & 1 & -.85 & -.425 & -.28\overline{3} & -.425 & -.425 & -.425 & -.425 & 0 & 0 \\
-.07\overline{72} & -.85 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 1 & -.28\overline{3} & 0 & 0 & 0 & 0 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 1 & -.425 & -.425 & -.425 & -.425 & -.85 & -.85 \\
-.07\overline{72} & 0 & 0 & 0 & -.28\overline{3} & 1 & 0 & 0 & 0 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
-.07\overline{72} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Thus $\alpha/n \;=\; \frac{.15}{11} \;=\; .01\overline{36}$

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Thus $\alpha/n = \frac{.15}{11} = .01\overline{36}$ and hence

$$(\alpha/n)\mathbf{1} = \begin{bmatrix} .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \end{bmatrix}$$

Example 5 (cont.)

Step 4: Compile vector $({}^{\alpha}\!/\!_{n})\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Thus ${}^{\alpha}\!/\!_{n} = \frac{.15}{11} = .01\overline{36}$ and hence

$$({}^{\alpha}\!/\!_{n})\mathbf{1} = \begin{bmatrix} .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \end{bmatrix}$$

Step 5: Use a computer to solve $(I - (1-\alpha)T')PR = ({}^{\alpha}\!/\!_{n})\mathbf{1}$

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Thus $\alpha/n = \frac{.15}{11} = .01\overline{36}$ and hence

$$(\alpha/n)\mathbf{1} = \begin{bmatrix} .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \end{bmatrix}$$

Step 5: Use a computer to solve $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$

For example 'Gauss-Jordan Elimination' in the *Matrix Reshish* online matrix calculator,

Example 5 (cont.)

Step 4: Compile vector $(\alpha/n)\mathbf{1}$.

Since $(1-\alpha) = .85$ use $\alpha = .15$.

Thus $\alpha/n = \frac{.15}{11} = .01\overline{36}$ and hence

$$(\alpha/n)\mathbf{1} = \begin{bmatrix} .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \\ .01\overline{36} \end{bmatrix}$$

Step 5: Use a computer to solve $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$

For example 'Gauss-Jordan Elimination' in the *Matrix Reshish* online matrix calculator, gives

$$PR = \begin{bmatrix} 0.032919 \\ 0.384644 \\ 0.343127 \\ 0.039112 \\ 0.080937 \\ 0.039112 \\ 0.016180 \\ 0.016180 \\ 0.016180 \\ 0.016180 \\ 0.016180 \end{bmatrix} \approx \begin{bmatrix} 3.3\% \\ 38.4\% \\ 34.3\% \\ 3.9\% \\ 8.1\% \\ 3.9\% \\ 1.6\% \\ 1.6\% \\ 1.6\% \\ 1.6\% \\ 1.6\% \end{bmatrix}$$

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

A computationally simpler method starts from the defining equation:

$$M'PR = PR.$$

Recall that $M = (\alpha/n)U + (1-\alpha)T$ is the modified transition matrix.

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

A computationally simpler method starts from the defining equation:

$$M' PR = PR.$$

Recall that $M = (\alpha/n)U + (1-\alpha)T$ is the modified transition matrix.

As with Markov chains in general we can attempt to find PR by iteratively calculating the chain of probability vectors $P_0, P_1, \ldots$ where
P_0 is arbitrary and $P_k = M' P_{k-1}$ for $k \geq 1$ (so $P_k = (M')^k P_0$).

A steady state is reached when $P_k \approx P_{k-1}$. Then declare that $PR \approx P_k$.

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

A computationally simpler method starts from the defining equation:

$$M'PR = PR.$$

Recall that $M = (\alpha/n)U + (1-\alpha)T$ is the modified transition matrix.

As with Markov chains in general we can attempt to find PR by iteratively calculating the chain of probability vectors $P_0, P_1, \ldots$ where P_0 is arbitrary and $P_k = M'P_{k-1}$ for $k \geq 1$ (so $P_k = (M')^k P_0$).

A steady state is reached when $P_k \approx P_{k-1}$. Then declare that $PR \approx P_k$.

Now $$P_k = M'P_{k-1} = [(\alpha/n)U + (1-\alpha)T']P_{k-1} = (\alpha/n)\mathbf{1} + (1-\alpha)T'P_{k-1}$$
and it is natural to start with all ranks equal.

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

A computationally simpler method starts from the defining equation:

$$M' PR = PR.$$

Recall that $M = (\alpha/n)U + (1-\alpha)T$ is the modified transition matrix.

As with Markov chains in general we can attempt to find PR by iteratively calculating the chain of probability vectors $P_0, P_1, \ldots$ where
P_0 is arbitrary and $P_k = M' P_{k-1}$ for $k \geq 1$ (so $P_k = (M')^k P_0$).

A steady state is reached when $P_k \approx P_{k-1}$. Then declare that $PR \approx P_k$.

Now
$$\begin{aligned} P_k = M' P_{k-1} &= [(\alpha/n)U + (1-\alpha)T']P_{k-1} \\ &= (\alpha/n)\mathbf{1} + (1-\alpha)T' P_{k-1} \end{aligned}$$
and it is natural to start with all ranks equal. So the iterative scheme is

$$\boxed{P_0 = (1/n)\mathbf{1}; \quad P_k = \alpha P_0 + (1-\alpha)T' P_{k-1}, \ k \geq 1.}$$

Iterative Approximation method

When n is huge, as it is with the WWW, solving the the $n \times n$ linear system $(I - (1-\alpha)T')PR = (\alpha/n)\mathbf{1}$ becomes computationally infeasible.

A computationally simpler method starts from the defining equation:

$$M'PR = PR.$$

Recall that $M = (\alpha/n)U + (1-\alpha)T$ is the modified transition matrix.

As with Markov chains in general we can attempt to find PR by iteratively calculating the chain of probability vectors $P_0, P_1, \ldots$ where P_0 is arbitrary and $P_k = M'P_{k-1}$ for $k \geq 1$ (so $P_k = (M')^k P_0$).

A steady state is reached when $P_k \approx P_{k-1}$. Then declare that $PR \approx P_k$.

Now
$$\begin{aligned} P_k = M'P_{k-1} &= [(\alpha/n)U + (1-\alpha)T']P_{k-1} \\ &= (\alpha/n)\mathbf{1} + (1-\alpha)T'P_{k-1} \end{aligned}$$
and it is natural to start with all ranks equal. So the iterative scheme is

$$\boxed{P_0 = (1/n)\mathbf{1}; \quad P_k = \alpha P_0 + (1-\alpha)T'P_{k-1}, \ k \geq 1.}$$

Each iteration takes a weighted average of teleporting and hyperlinking.

Example 4C

In Example 4B we used the equation-solving method to find PR.

For $T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$ and $\alpha = 0.1$ we found $PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2d.p.

Example 4C

In Example 4B we used the equation-solving method to find PR.

For $T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$ and $\alpha = 0.1$ we found $PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2d.p.

Let's try the same problem using iterative approximation:

$$P_0 = \begin{bmatrix} .25 \\ .25 \\ .25 \\ .25 \end{bmatrix} \qquad P_k = \begin{bmatrix} .025 \\ .025 \\ .025 \\ .025 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & .3 \\ .45 & 0 & .45 & .3 \\ 0 & .9 & 0 & .3 \\ .45 & 0 & .45 & 0 \end{bmatrix} P_{k-1}$$

Example 4C

In Example 4B we used the equation-solving method to find $\mathcal{PR}$.

For $T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$ and $\alpha = 0.1$ we found $\mathcal{PR} = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2d.p.

Let's try the same problem using iterative approximation:

$$P_0 = \begin{bmatrix} .25 \\ .25 \\ .25 \\ .25 \end{bmatrix} \qquad P_k = \begin{bmatrix} .025 \\ .025 \\ .025 \\ .025 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & .3 \\ .45 & 0 & .45 & .3 \\ 0 & .9 & 0 & .3 \\ .45 & 0 & .45 & 0 \end{bmatrix} P_{k-1}$$

Results of the first ten iterations, rounded to 2d.p. Calcs used 15d.p.

k	1	2	3	4	5	6	7	8	9	10
P_k	$\begin{bmatrix} .10 \\ .33 \\ .33 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .29 \\ .39 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .35 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .38 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .30 \\ .37 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$

Example 4C

In Example 4B we used the equation-solving method to find PR.

For $T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$ and $\alpha = 0.1$ we found $PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2d.p.

Let's try the same problem using iterative approximation:

$$P_0 = \begin{bmatrix} .25 \\ .25 \\ .25 \\ .25 \end{bmatrix} \qquad P_k = \begin{bmatrix} .025 \\ .025 \\ .025 \\ .025 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & .3 \\ .45 & 0 & .45 & .3 \\ 0 & .9 & 0 & .3 \\ .45 & 0 & .45 & 0 \end{bmatrix} P_{k-1}$$

Results of the first ten iterations, rounded to 2d.p. Calcs used 15d.p.

k	1	2	3	4	5	6	7	8	9	10
P_k	$\begin{bmatrix} .10 \\ .33 \\ .33 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .29 \\ .39 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .35 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .38 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .30 \\ .37 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$

Pretty good after just 2 iterations! Within 1%-point after 4 iterations.

Example 4C

In Example 4B we used the equation-solving method to find PR.

For $T = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$ and $\alpha = 0.1$ we found $PR = \begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$ to 2d.p.

Let's try the same problem using iterative approximation:

$$P_0 = \begin{bmatrix} .25 \\ .25 \\ .25 \\ .25 \end{bmatrix} \qquad P_k = \begin{bmatrix} .025 \\ .025 \\ .025 \\ .025 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & .3 \\ .45 & 0 & .45 & .3 \\ 0 & .9 & 0 & .3 \\ .45 & 0 & .45 & 0 \end{bmatrix} P_{k-1}$$

Results of the first ten iterations, rounded to 2d.p. Calcs used 15d.p.

k	1	2	3	4	5	6	7	8	9	10
P_k	$\begin{bmatrix} .10 \\ .33 \\ .33 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .29 \\ .39 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .35 \\ .25 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .38 \\ .22 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .31 \\ .36 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$	$\begin{bmatrix} .09 \\ .30 \\ .37 \\ .24 \end{bmatrix}$	$\begin{bmatrix} .10 \\ .30 \\ .37 \\ .23 \end{bmatrix}$

Pretty good after just 2 iterations! Within 1%-point after 4 iterations.

END OF SECTION D3

End of Course Notes.