Chapter 10 : Finite-State Markov Chains

10.2 : The Steady-State Vector and Page Rank

Topics and Objectives

Topics

1. Review of Markov chains
2. Theorem describing the steady state of a Markov chain
3. Applying Markov chains to model website usage.
4. Calculating the PageRank of a web.

Learning Objectives

1. Determine whether a stochastic matrix is regular.
2. Apply matrix powers and theorems to characterize the long-term behaviour of a Markov chain.
3. Construct a transition matrix, a Markov Chain, and a Google Matrix for a given web, and compute the PageRank of the web.

Where is Chapter 10?

- The material for this part of the course is covered in Section 10.2
- Chapter 10 is not included in the **print** version of the book, but it is in the **on-line version**.
- If you read 10.2, and I recommend that you do, you will find that it requires an understanding of 10.1.
- You are not required to understand the material in 10.1.

Steady State Vectors

Recall the car rental problem from our Section 4.9 lecture.

Problem

A car rental company has 3 rental locations, A, B, and C.

		rented from		
		A	B	C
returned to	A	.8	.1	.2
	B	.2	.6	.3
	C	.0	.3	.5

There are 10 cars at each location today, what happens to the distribution of cars after a long time?

Long Term Behaviour

Can use the transition matrix, P, to find the distribution of cars after 1 week:

$$\vec{x}_1 = P\vec{x}_0$$

The distribution of cars after 2 weeks is:

$$\vec{x}_2 = P\vec{x}_1 = PP\vec{x}_0$$

The distribution of cars after n weeks is:

Long Term Behaviour

To investigate the long-term behaviour of a system that has a regular transition matrix P, we could:

1. compute the **steady-state vector**, $\vec{q}$, by solving $\vec{q} = P\vec{q}$.
2. compute $P^n\vec{x}_0$ for large n.
3. compute P^n for large n, each column of the resulting matrix is the steady-state

Theorem 1

If P is a regular $m \times m$ transition matrix with $m \geq 2$, then the following statements are all true.

1. There is a stochastic matrix Π such that

$$\lim_{n \to \infty} P^n = \Pi$$

2. Each column of Π is the same probability vector $\vec{q}$.
3. For any initial probability vector $\vec{x}_0$,

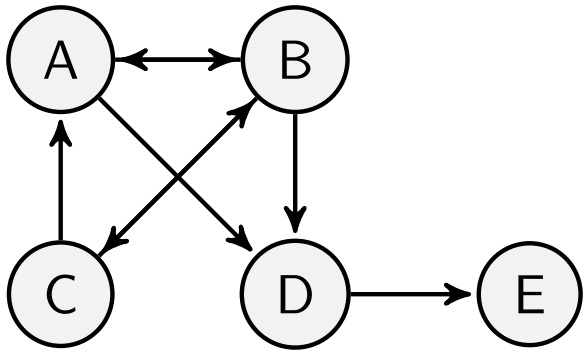
$$\lim_{n \to \infty} P^n \vec{x}_0 = \vec{q}$$

4. P has a unique eigenvector, $\vec{q}$, which has eigenvalue $\lambda = 1$.
5. The eigenvalues of P satisfy $|\lambda| \leq 1$.

We will apply this theorem when solving PageRank problems.

Example 1

A set of web pages link to each other according to this diagram.



Page A has links to pages _____ .

Page B has links to pages _____ .

We make two assumptions:

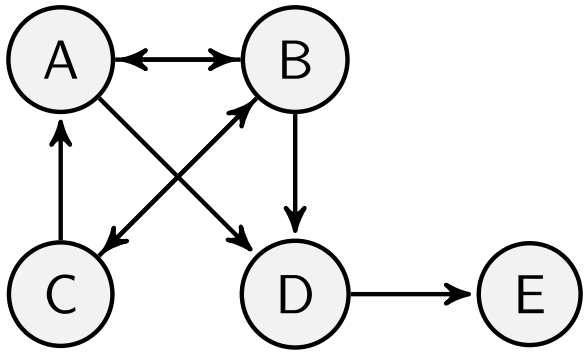
a) A user on a page in this web is equally likely to go to any of the pages that their page links to.

b) If a user is on a page that does not link to other pages, the user stays at that page.

Use these assumptions to construct a Markov chain that represents how users navigate the above web.

Solution

Use the assumptions on the previous slide to construct a Markov chain that represents how users navigate the web.



Transition Matrix, Importance, and PageRank

- The square matrix we constructed in the previous example is a **transition matrix**. It describes how users transition between pages in the web.
- The steady-state vector, $\vec{q}$, for the Markov-chain, can characterize the long-term behavior of users in a given web.
- If $\vec{q}$ is unique, the **importance** of a page in a web is given by its corresponding entry in $\vec{q}$.
- The **PageRank** is the ranking assigned to each page based on its importance. The highest ranked page has PageRank 1, the second PageRank 2, and so on.
- Two pages with same importance receive the same PageRank (some other method would be needed to resolve ties)

Is the transition matrix in Example 1 a regular matrix?

Adjustment 1

Adjustment 1

If a user reaches a page that does not link to other pages, the user will choose any page in the web, with equal probability, and move to that page.

Let's denote this modified transition matrix as P_*. Our transition matrix in Example 1 becomes:

Adjustment 2

Adjustment 2

A user at any page will navigate to any page among those that their page links to with equal probability p, and to any page in the web with equal probability $1 - p$. The transition matrix becomes

$$G = pP_* + (1 - p)K$$

All the elements of the $n \times n$ matrix K are equal to $1/n$.

p is referred to as the **damping factor**, Google is said to use $p = 0.85$.

With adjustments 1 and 2, our the Google matrix is:

Computing Page Rank

- Because G is stochastic, for any initial probability vector $\vec{x}_0$,

$$\lim_{n\to\infty} G^n \vec{x}_0 = \vec{q}$$

- We can obtain steady-state evaluating $G^n\vec{x}_0$ for large n, by solving $G\vec{q} = \vec{q}$, or by evaluating $\vec{x}_n = G\vec{x}_{n-1}$ for large n.
- Elements of the steady-state vector give the importance of each page in the web, which can be used to determine PageRank.
- Largest element in steady-state vector corresponds to page with PageRank 1, second largest with PageRank 2, and so on.

On an exam,

- problems that require a calculator will not be on your exam
- you may construct your G matrix using factions instead of decimal expansions

There is (of course) Much More to PageRank

The PageRank Algorithm currently used by Google is under constant development, and tailored to individual users.

- When PageRank was devised, in 1996, Yahoo! used humans to provide a "index for the Internet, " which was 10 million pages.
- The PageRank algorithm was produced as a competing method. The patent was awarded to Stanford University, and exclusively licensed to the newly formed Google corporation.
- Brin and Page combined the PageRank algorithm with a webcrawler to provide regular updates to the transition matrix for the web.
- The explosive growth of the web soon overwhelmed human based approaches to searching the internet.

WolframAlpha and MATLAB/Octave Syntax

Suppose we want to compute

$$\begin{pmatrix} .8 & .1 & .2 \\ .2 & .6 & .3 \\ .0 & .3 & .5 \end{pmatrix}^{10}$$

- At wolframalpha.com, we can use the syntax:

  `MatrixPower[{{.8,.1,.2},{.2,.6,.3},{.0,.3,.5}},10]`
- In MATLAB, we can use the syntax

  `[.8 .1 .2 ;.2 .6 .3;.0 .3 .5]^10`
- Octave uses the same syntax as MATLAB, and there are several free, online, Octave compilers. For example: https://octave-online.net.

You will need to compute a few matrix powers in your homework, and in your future courses, depending on what courses you end up taking.

Example 2 (if time permits)

Construct the Google Matrix for the web below. Which page do you think will have the highest PageRank? How would your result depend on the damping factor p? Use software to explore these questions.