Applications

- Population dynamics
- Weather and election forecasting
- Internet webpage ranking
- Markov Chains and Baseball Statistics.

Topics

- Linear iterative systems (ALA 9.1)
    - Matrix powers, diagonalization, and stability
- Markov Processes (ALA 9.3, LAA 5th ed 4.9 and 10.1)
    - Perron-Frobenius Theorem and steady state probability vector.
- PageRank and the Google Matrix (LAA 5th ed 10.2).

Linear Iterative Systems (ALA 9.1)

So far, we have focused on continuous time dynamical systems for which the time index t of the solution $x(t)$ is continuous, i.e., $t \in \mathbb{R}$. This is a natural model to use for many systems, such as for example those evolving according to the laws of physics. However, in other instances, it is more natural to consider time in discrete steps. For example, when banks add interest to a savings account, they typically do so on a monthly or yearly basis. More specifically, suppose at period k, we have $x(k)$ dollars in our account, and an interest rate of r is applied. Then $x(k+1) = (1+r)x(k)$. This defines a scalar linear iterative system, which take the general form:

$$x(k+1) = \lambda x(k), \quad x(0) = a. \qquad (SD)$$

If we roll out the dynamics (SD), we easily compute:

$x(0) = a$, $x(1) = \lambda a$, $x(2) = \lambda^2 a$, $\ldots$, $x(k) = \lambda^k a$ for any positive integer k.

We therefore immediately conclude that there are three possibilties for $x(0) = a \neq 0$:

- Stable: If $|\lambda| < 1$, then $|x(k)| \longrightarrow 0$ as $k \to \infty$
- Marginaly Stable: If $|\lambda| = 1$, then $|x(k)| = |a|$ for all $k \in \mathbb{N}$ (online notes please define $\mathbb{N}$ as natural numbers $0, 1, 2, \ldots$)
- Unstable: If $|\lambda| > 1$, then $|x(k)| \to \infty$ as $k \to \infty$

Our goal is to extend this analysis to general linear iterative systems of the form:

$$\underline{x}(k+1) = T\,\underline{x}(k), \quad \underline{x}(0) = \underline{a}, \qquad (MD)$$

where $\underline{x}(k) \in \mathbb{R}^n$ and $T \in \mathbb{R}^{n \times n}$. We will then use these tools to study a very important class of linear iterative systems called Markov Chains, which can be used for everything from internet search (Google's search algorithm PageRank) to baseball statistics (DraftKings uses these ideas to set betting odds!).

Powers of Matrices

Rolling out the dynamics (MD), we again see a clear solution to (MD):

$\underline{x}(0) = \underline{a}$, $\underline{x}(1) = T\underline{a}$, $\underline{x}(2) = T^2\underline{a}$, $\ldots$, $\underline{x}(k) = T^k\underline{a}$ for all $k \in \mathbb{N}$.

However, unlike the scalar setting (SD), the qualitative behavior of $\underline{x}(k) = T^k\underline{a}$ as $k \to \infty$ is much less obvious. Since the scalar solution $x(k) = \lambda^k a$ is defined in terms of powers of λ, let's try a similar guess for the vector-valued setting: $\underline{x}(k) = \lambda^k \underline{v}$. Under what conditions on λ and $\underline{v}$ is $\underline{x}(k) = \lambda^k \underline{v}$ a solution to (MD)?

On the one hand, we have that $\underline{x}(k+1) = \lambda^{k+1}\underline{v}$. On the other, we have

$$T\underline{x}(k) = T(\lambda^k \underline{v}) = \lambda^k T\underline{v}.$$

These two expressions will be equal if and only if $T\underline{v} = \lambda \underline{v}$, i.e., if and only if $(\lambda, \underline{v})$ are an eigenvalue/vector pair of T.

Thus, for each eigenvalue/vector pair $(\lambda_i, \underline{v}_i)$ of T, we can construct a solution $\underline{x}_i(k) = \lambda_i^k \underline{v}_i$ to (MO). By linear superposition, we can use this observation to characterize all solutions for complete matrices.

Theorem: If the coefficient matrix T is complete, then the general solution to the linear iterative system $\underline{x}(k+1) = T\underline{x}(k)$ is given by

$$\underline{x}(k) = c_1 \lambda_1^k \underline{v}_1 + c_2 \lambda_2^k \underline{v}_2 + \cdots + c_n \lambda_n^k \underline{v}_n,$$

where $\underline{v}_1, \ldots, \underline{v}_n$ are linearly independent eigenvectors of T with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$. The coefficients $c_1, \ldots, c_n$ are scalars uniquely prescribed by the initial condition

$$\underline{x}(0) = [\underline{v}_1\ \underline{v}_2 \cdots \underline{v}_n]\underline{c} = \underline{a}.$$

We can extend this theorem to incomplete matrices using Jordan Blocks, but the idea remains the same.

Example: Consider the iterative system $\underline{x}(k+1) = T\underline{x}(k)$ defined by

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} .6 & .2 \\ .2 & .6 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}, \quad \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}.$$

T has eigenvalue/vector pairs:

$$\lambda_1 = .8,\ \underline{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \lambda_2 = .4,\ \underline{v}_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

Therefore, the eigensolutions are

$$\underline{x}_1(k) = \lambda_1^k \underline{v}_1 = (.8)^k \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \underline{x}_2(k) = \lambda_2^k \underline{v}_2 = (.4)^k \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

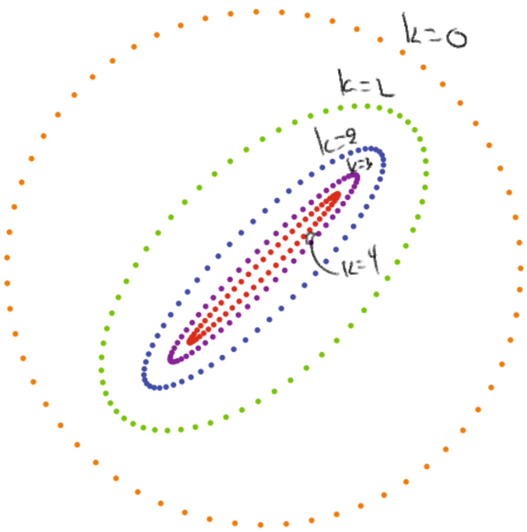
Thus, a general solution is given by $\underline{x}(k) = c_1 \underline{x}_1(k) + c_2 \underline{x}_2(k) = c_1 (.8)^k \begin{bmatrix} 1 \\ 1 \end{bmatrix} + c_2 (.4)^k \begin{bmatrix} -1 \\ 1 \end{bmatrix}$.

To determine c_1 and c_2, we solve $\underline{x}(0) = \begin{bmatrix} c_1 - c_2 \\ c_1 + c_2 \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \Rightarrow c_1 = \frac{a_1 + a_2}{2},\ c_2 = \frac{a_2 - a_1}{2}$

giving the specific solution:

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \frac{a_1+a_2}{2}(.8)^k \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \frac{a_2-a_1}{2}(.4)^k \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

We conclude that $\underline{x}(k) \to 0$ as $k \to \infty$, and that the slowest decaying direction is $\underline{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, which decays at rate .8 (by 80%) per time step.



Sampled initial conditions along unit circle (orange) then plot iterates (green, blue, purple, red)
k=1 k=2 k=3 k=4

Figure 9.2. Stable Iterative System.

ONLINE NOTES: Please include Example 9.7 and show how to obtain real solutions.

Diagonalization and Iteration

An alternative and equally efficient approach to solving (MD) in the case of complete matrices is based on the diagonalization of T. We start with the following observation (which we also saw when computing the matrix exponential). Let $V = [v_1 \cdots v_n]$ be an eigenbasis for T, and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ the diagonal matrix of the eigenvalues of T. Then:

$$T = V\Lambda V^{-1}, \quad T^2 = V\Lambda V^{-1} V \Lambda V = V\Lambda^2 V^{-1}, \quad T^3 = T(T^2) = V\Lambda V^{-1} V \Lambda^2 V^{-1}$$
$$= V\Lambda^3 V^{-1}$$

and in general, $T^k = V\Lambda^k V^{-1}$. Therefore, the solution to $\underline{x}(k+1) = T\underline{x}(k)$, with $\underline{x}(0) = \underline{a}$ is

$$\underline{x}(k) = T^k \underline{a} = V\Lambda^k V^{-1}\underline{a}.$$

If we define $\underline{c} = V^{-1}\underline{a}$, then we recover $\underline{x}(k) = c_1\lambda_1^k \underline{v}_1 + \cdots + c_n \lambda_n^k \underline{v}_n$.

ONLINE NOTES: Please include example 9.8.

Markov Chains (LAA 5th Edition, §4.9 and Ch. 10, ALA 9.3).

We will spend the rest of this lecture on Markov Chains, which are a widely used linear iterative model used to describe a wide variety of situations in biology, business, chemistry, engineering, physics, and elsewhere.

In each case, the model is used to describe an experiment or measurement that is performed many times in the same way. The outcome of an experiment can be one of several known possible outcomes, and importantly, the outcome of one experiment depends only on the experiment conducted immediately before it. Before introducing a general model for Markov chains, let's look at a simple example.

Example: Weather Prediction.

Suppose you would like to predict the weather in your city. Looking at local weather records over the past 10 years, you notice that:

(i) If today is sunny, tomorrow is sunny 70% of the time, and cloudy 30% of the time.

(ii) If today is cloudy, tomorrow is cloudy 80% of the time, and sunny 20% of the time.

Now, suppose today is sunny. What is the probability[1] that the weather 8 days from now will also be sunny?

[1] You will learn how to properly define probabilities in ESE 3010. For our purposes, you can think of it as confidences or likelihoods. So saying that 8 days from now will be sunny with probability 60% is the same as saying that the weather 10 days from now is determined by flipping a biased coin that comes up "sunny" 60% of the time and "cloudy" 40% of the time.

To formulate this problem mathematically, let's use $s(k)$ to denote the probability that day k is sunny and $c(k)$ the probability that it is cloudy. If these are the only two possibilities, then the individual probabilities must sum to 1 (1 represents 100% likely, .5 50% likely, etc.): $s(k)+c(k)=1$.

According to our historical data, the probability that day $k+1$ is sunny or cloudy can be expressed as:

$$s(k+1) = .7\,s(k) + .2\,c(k), \qquad c(k+1) = .3\,s(k) + .8\,c(k) \qquad (*)$$

For example, the equation says that if day k was sunny, i.e., $s(k)=1$ and $c(k)=0$ there is a 70% chance day $k+1$ is too; similarly, if day k was cloudy, i.e., $s(k)=0$ and $c(k)=1$, there is a 20% chance day $k+1$ is sunny.

We rewrite (*) as the linear iterative system $\underline{x}(k+1) = P\underline{x}(k)$, where

$$P = \begin{bmatrix} .7 & .2 \\ .3 & .8 \end{bmatrix} \quad \text{and} \quad \underline{x}(k) = \begin{bmatrix} s(k) \\ c(k) \end{bmatrix}.$$

We use P instead of T here as this is a typical convention for denoting the transition matrix of a Markov chain. The vector $\underline{x}(k)$ is called the k^{th} state vector.

Now, given that today is sunny, i.e., that $s(0) = 1$ and $c(0) = 1$, what is the probability that 8 days from now is sunny? We can answer this easily by iterating the system $\underline{x}(k+1) = P\underline{x}(k)$ to compute $\underline{x}(8)$!

$$\underline{x}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \underline{x}(1) = P\underline{x}(0) = \begin{bmatrix} .7 \\ .3 \end{bmatrix}, \quad \underline{x}(2) = P\underline{x}(1) = P^2\underline{x}(0) \approx \begin{bmatrix} .55 \\ .45 \end{bmatrix},$$

$$\underline{x}(3) = P^3\underline{x}(0) \approx \begin{bmatrix} .475 \\ .525 \end{bmatrix}, \quad \underline{x}(4) \approx \begin{bmatrix} .438 \\ .562 \end{bmatrix}, \quad \underline{x}(5) \approx \begin{bmatrix} .419 \\ .581 \end{bmatrix}, \quad \underline{x}(6) \approx \begin{bmatrix} .410 \\ .590 \end{bmatrix}$$

$$\underline{x}(7) \approx \begin{bmatrix} .405 \\ .595 \end{bmatrix}, \quad \underline{x}(8) \approx \begin{bmatrix} .402 \\ .598 \end{bmatrix}.$$

So we conclude that 40.2% of the time, if today is sunny, then 8 days from now is also sunny.

We make a few observations about the state vectors $\underline{x}(k)$ to motivate some of the new tools we'll introduce:

1) Every state vector $\underline{x}(k)$ is a probability vector, i.e., $x_1(k)$ and $x_2(k) \geq 0$ and $x_1(k) + x_2(k) = 1$
2) the iterates converge fairly quickly to $\underline{x}^* = \begin{bmatrix} .4 \\ .6 \end{bmatrix}$, which is a fixed point of $\underline{x}(k+1) = P\underline{x}(k)$, i.e., $\underline{x}^* = P\underline{x}^*$.
3) This convergence to $\underline{x}^*$ actually happens for any initial probability vector $\underline{x}(0)$. This means that in the long run, 40% of days are sunny and 60% are rainy.

Let's try to understand why this happens, and then we'll look at some interesting applications of Markov chains.

Our starting point is a general definition of a probability vector: a vector $\underline{x} \in \mathbb{R}^n$ is called a probability vector if $x_i \geq 0$ for $i = 1, \dots, n$ and $x_1 + \cdots + x_n = 1$. We interpret x_i as the probability, or likelihood, that the system is in state i.

In general, a Markov Chain is given by the first order linear iterative system

$$\underline{x}(k+1) = P\underline{x}(k) \qquad \text{(MC)}$$

whose initial state $\underline{x}(0)$ is a probability vector. The entries of the transition matrix T must satisfy

$$0 \leq p_{ij} \leq 1 \quad \text{and} \quad p_{1j} + \cdots + p_{nj} = 1. \qquad \text{(TM)}$$

for all $i, j = 1, \ldots, n$. The entry p_{ij} is the transition probability that the system will switch from state j to state i. Because this covers all possible transitions, this means each column sums to 1. Under these conditions, we can guarantee that if $\underline{x}(k)$ is a probability vector, so is $\underline{x}(k+1) = P\underline{x}(k)$. To see this, note that $\underline{1}^T P = [\underline{1}^T p_1 \cdots \underline{1}^T p_n] = [1 \cdots 1] = \underline{1}^T$ so that $\underline{1}^T \underline{x}(k+1) = \underline{1}^T P \underline{x}(k) = \underline{1}^T \underline{x}(k) = 1$. That $\underline{x}(k+1)$ is entrywise non-negative follows from P and $\underline{x}(k)$ being entry wise non-negative.

Next, let's investigate convergence properties. We first need to impose a very mild technical condition on the transition matrix P, namely we assume that it is regular. A transition matrix P (TM) is regular if for some power k, T^k contains no zero entries. This means that it is possible to get from one state to any other state in k steps.

The long-term behavior of a Markov chain with regular transition matrix P is governed by the Perron-Frobenius theorem, which we state next. The proof is quite involved, so we won't cover it, but if you're curious, check out the end of ALA 9.3.

Theorem: If P is a regular transition matrix, then it admits a unique probability vector $\underline{x}^*$ with eigenvalue $\lambda_1 = 1$. Moreover, a Markov chain with coefficient matrix P will converge to $\underline{x}^*$: $\underline{x}(k) \to \underline{x}^*$ as $k \to \infty$.

This is a very exciting development! It tells us that we can understand the long term behavior of a regular Markov chain by solving for the eigenvector $\underline{x}^*$ associated with the eigenvalue $\lambda_1 = 1$ of P.

Returning to our weather prediction example, we compute the steady state probability vector $\underline{x}^*$ by first solving $(P-I)\underline{v} = 0$:

$$(P-I)\underline{v} = \begin{bmatrix} -.3 & .2 \\ .3 & -.2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0 \Rightarrow v_1 = \frac{2}{3} v_2 \Rightarrow \underline{v} = \begin{bmatrix} 2/3 \\ 1 \end{bmatrix}$$

and then normalizing $\underline{v}$ so that its entries add up to 1:

$$\underline{x}^* = \frac{1}{1+\frac{2}{3}} \begin{bmatrix} 2/3 \\ 1 \end{bmatrix} = \begin{bmatrix} 2/5 \\ 3/5 \end{bmatrix} = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}.$$

This special eigenvector $\underline{x}^*$ tells us that no matter the initial state $\underline{x}(0)$, the long term behavior is that we are in state 1 (sunny) 40% of days and state 2 (cloudy) 60% of days.

Example: Get out the vote!

Suppose the voting results of a congressional election at a certain voting precinct are represented by a vector $\underline{x} \in \mathbb{R}^3$:

$$\underline{x} = \begin{bmatrix} \text{\% voting Democratic (D)} \\ \text{\% voting Republican (R)} \\ \text{\% voting Libertarian (L)} \end{bmatrix}.$$

We record the outcome of this election every two years by a vector of this type, and let's assume that the outcome of one election depends only on results of the previous one. Then the sequence $\underline{x}(k)$ of vectors that describe the votes in each election form a Markov chain. Suppose, using historical data, we estimate the following transition matrix P:

$$P = \begin{array}{c} \text{From:} \\ \begin{array}{ccc} D & R & L \end{array} \\ \begin{bmatrix} .7 & .1 & .3 \\ .2 & .8 & .3 \\ .1 & .1 & .4 \end{bmatrix} \end{array} \begin{array}{c} \text{To:} \\ D \\ R \\ L \end{array}$$

The entries in the first column, labeled D, describe what % of persons who voted D in the last election will vote D, R, and L in this one: in this example, 70% of prior D voters will vote D again, 20% will vote R, and 10% will vote L.

If we assume that P remains fixed across many elections, we can predict not only the next election's results, but long term elections as well. For example, if last election had results:

$$\underline{x}(0) = \begin{bmatrix} .55 \\ .45 \\ .05 \end{bmatrix}$$

then the next election will have a likely outcome of

$$\underline{x}(1) = P\underline{x}(0) = \begin{bmatrix} .44 \\ .445 \\ .115 \end{bmatrix}$$

and the following election will have likely outcome

$$\underline{x}(2) = P\underline{x}(1) = \begin{bmatrix} .387 \\ .4785 \\ .1345 \end{bmatrix}.$$

In the long run, we expect voters converge to the steady state distribution $\underline{x}^*$ satisfying $\underline{x}^* = P\underline{x}^*$, which we obtain by solving

$$(P-I)\underline{v} = 0$$

and setting $\underline{x}^* = \frac{1}{\underline{1}^T\underline{v}}\underline{v}$. In this case, this works out to:

$$\underline{x}^* \approx \begin{bmatrix} .321 \\ .536 \\ .143 \end{bmatrix},$$

which means that assuming voter patterns do not change, 32.1% of votes will go to D, 53.6% to R, and 14.3% to L in this precinct.

<u>Random Walks, Page Rank, and the Google Matrix</u>

An internet user's behaviour while surfing the web can be modeled as a Markov chain that captures a random walk on a graph. We start with a simple example of this, and then explain how it can be used to design a search engine.

Consider the following graph:

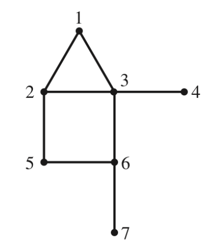


FIGURE 4
A graph with seven vertices.

Which has seven vertices interconnected by edges. Let's pretend this graph models a very simple internet! each vertex, or node, is a web page, and each edge is a hyperlink connecting pages to each other (for now we assume that if page i links to page j, then page j also links to page i, but this isn't necessary). We assume that if a user is on page i, they will click on one of the hyperlinks with equal probability. For example, in our simple internet, if a user is on page 5, they will visit page 2 next 50% of the time, and page 6 next 50% of the time. Similarly, if a user is on page 3, they will visit page 1, 2, or 4 next 33% of the time each.

We can model this user behaviour using a Markov Chain, which is called a simple random walk on a graph. The transition matrix for the graph above is given by:

$$P = \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array}\begin{array}{ccccccc|c} 1 & 2 & 3 & 4 & 5 & 6 & 7 & \\ \hline 0 & 1/3 & 1/4 & 0 & 0 & 0 & 0 & 1 \\ 1/2 & 0 & 1/4 & 0 & 1/2 & 0 & 0 & 2 \\ 1/2 & 1/3 & 0 & 1 & 0 & 1/3 & 0 & 3 \\ 0 & 0 & 1/4 & 0 & 0 & 0 & 0 & 4 \\ 0 & 1/3 & 0 & 0 & 0 & 1/3 & 0 & 5 \\ 0 & 0 & 1/4 & 0 & 1/2 & 0 & 1 & 6 \\ 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 7 \end{array}$$

This allows us to answer questions such as the following: Suppose 100 users start on page 6. After each user clicks on three hyperlinks, what % of users do we expect to find on each web page. The solution is given by setting our initial user distribution to:

$$\underline{x}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

and computing $\quad \underline{x}(3) = P\underline{x}(0) = \begin{bmatrix} .0833 \\ .0417 \\ .4028 \\ 0 \\ .2778 \\ 0 \\ .1944 \end{bmatrix}$.

This tells, for example, that 40.28% of users starting on page 6 end up on page 3 after 3 hyperlink clicks.

PageRank

The founders of Google, Sergey Brin and Lawrence Page, reasoned that "important" pages had links coming from other "important" pages, and thus, a typical internet user would spend more time on more important pages, and less time on less important pages. This can be captured by the steady state distribution $\underline{x}^*$ of the Markov chain we are using to model the internet: in the long run, a typical user will spend x_i^* % of their time on page i. This is precisely the observation used to define the PageRank algorithm, which Google uses to rank the importance of the web pages it catalogs.

Key idea: The importance of a webpage i can be measured by the corresponding entry x_i^* of the steady state vector $\underline{x}^*$ of the Markov chain describing the behaviour of a typical internet user

Now, if a typical transition matrix P describing the internet were regular, we would be done — we simply compute $x^* = Px^*$ and use x^* to rank websites. Unfortunately, typical models of the web are directed graphs which lead to non-regular transition matrices P. To address this, Google make two adjustments, which we illustrate with the following slight modification of our previous example:

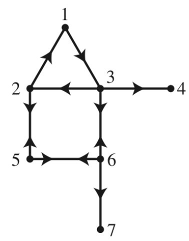


FIGURE 1
A seven-page Web.

$$P = \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 & 1/2 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 1/3 & 1 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 1/3 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/3 & 1 \end{bmatrix} \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}$$

The first issue that arises here is that pages 4 and 7 are dangling nodes: if a user ever ends up on page 4 or 7, they never leave as there are no outgoing links. This means that columns 4 and 7 never change as we compute P^k, and hence P cannot be regular. To handle dangling nodes, the following adjustment is made:

Adjustment 1: If an internet user reaches a dangling node, they will pick any page on the web with equal probability and move to that page.

This means that if state j is an absorbing state, we replace column j of P with the vector $(\frac{1}{n}, \frac{1}{n}, \dots, \frac{1}{n})$. In our example, our modified transition matrix is now:

$$P_* = \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 0 & 1/2 & 0 & 1/7 & 0 & 0 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 1/2 & 0 & 1/7 \\ 1 & 0 & 0 & 1/7 & 0 & 1/3 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 0 & 0 & 1/7 \\ 0 & 1/2 & 0 & 1/7 & 0 & 1/3 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 1/2 & 0 & 1/7 \\ 0 & 0 & 0 & 1/7 & 0 & 1/3 & 1/7 \end{bmatrix} \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}$$

While this helps eliminate dangling nodes, we may still not have a regular transition matrix, as there might still be cycles of pages. For example, if page i only links to page j, and page j only links to page i, a user entering either page is condemned to bouncing back and forth between the two pages: this means the corresponding columns of P_*^k will always have zeros in them, and hence P_* would not be regular.

Adjustment 2: Pick a number p between 0 and 1. If a user is on page j, then p proportion of the time, they will pick from all possible hyperlinks on that page with equal probability and move to that page. The other $1-p$ fraction of the time, they will pick any page on the web with equal probability and move to that page.

In terms of the modified transition matrix P_*, the new transition matrix will be

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

where $K \in \mathbb{R}^{n \times n}$ and $k_{ij} = 1/n$ for $i,j = 1, \ldots, n$. The matrix G is called the Google matrix. G is easily seen to be regular as all entries of G are positive.

Although any value of p is used, Google is thought to use $p = 0.85$. For our example, the Google matrix is

$$G = .85\begin{bmatrix} 0 & 1/2 & 0 & 1/7 & 0 & 0 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 1/2 & 0 & 1/7 \\ 1 & 0 & 0 & 1/7 & 0 & 1/3 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 0 & 0 & 1/7 \\ 0 & 1/2 & 0 & 1/7 & 0 & 1/3 & 1/7 \\ 0 & 0 & 1/3 & 1/7 & 1/2 & 0 & 1/7 \\ 0 & 0 & 0 & 1/7 & 0 & 1/3 & 1/7 \end{bmatrix}$$

$$+ .15\begin{bmatrix} 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \end{bmatrix}$$

$$= \begin{bmatrix} .021429 & .446429 & .021429 & .142857 & .021429 & .021429 & .142857 \\ .021429 & .021429 & .304762 & .142857 & .446429 & .021429 & .142857 \\ .871429 & .021429 & .021429 & .142857 & .021429 & .304762 & .142857 \\ .021429 & .021429 & .304762 & .142857 & .021429 & .021429 & .142857 \\ .021429 & .446429 & .021429 & .142857 & .021429 & .304762 & .142857 \\ .021429 & .021429 & .304762 & .142857 & .446429 & .021429 & .142857 \\ .021429 & .021429 & .021429 & .142857 & .021429 & .304762 & .142857 \end{bmatrix}$$

We can now compute the steady state vector $\underline{x}^* = G\underline{x}^*$, which is found to be:

$$\underline{x}^* = \begin{bmatrix} .116293 \\ .168567 \\ .191263 \\ .098844 \\ .164054 \\ .168567 \\ .092413 \end{bmatrix}$$

Thus, we can rank the pages in terms of descending importance as: 3, 2, 6, 5, 1, 4, 7.

Numerical Note: The Google matrix for the world wide web has over 8 billion rows and columns, and computing $\underline{x}^* = G\underline{x}^*$ is a very non-trivial task. An iterative approach known as the power method is used in practice, and typically takes several days for Google to compute a new $\underline{x}^*$, which it does every month.

ONLINE NOTES: Please provide links to accessible description of power method.