Markov Chains

Rebecca Weber, 2007

OSRIC How is't, Laertes?
  You cannot speak of this elegant
thimble"; and, when it grunted again, and put it more clearly,"
Alice replied very politely, "if I had seen the mobbled queen'
HAMLET 'The mobbled queen'?
POLONIUS That's very true, my lord.
HAMLET Then you live about her pet: "Dinah's our cat. And
she's such a very noble youth. Mark.
LAERTES What ceremony else?
PRIEST
  Her obsequies have been so with us a story."

What Is a Markov Chain?

A *probability vector* is a vector with nonnegative entries that add up to 1. A *stochastic matrix* is a square matrix whose columns are all probability vectors.

A *Markov chain* is a sequence of probability vectors $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \dots$ together with a stochastic matrix P such that

$$\mathbf{x}_1 = P\mathbf{x}_0, \quad \mathbf{x}_2 = P\mathbf{x}_1, \quad \dots, \quad \mathbf{x}_{n+1} = P\mathbf{x}_n, \quad \dots$$

What can these mean? Essentially, they represent a series of states of a system and a uniform way to change from one state to the next. The important part of the set-up is that wherever you happen to be, you get to the next step in exactly the same way.

An Example

We might imagine a laboratory setup where a lab rat has three levers available to it. Lever I results in a mild electric shock. Lever II makes a loud noise. Lever III causes a piece of food to appear. Originally the rat has equal probability of choosing each of the three levers.

If it chooses lever I, it has only a 10% probability of choosing that lever the next time. If it chooses lever II, it has a 25% probability of choosing that lever the next time. If it chooses lever III, it has a 80% probability of choosing that lever the next time. In each case the probability of choosing the remaining two levers is equally split.

How do we represent this as a Markov chain?

Example, Continued

We are making the simplifying assumption that what the rat does next only depends on the most recent lever pressing – no longer-term memory.

The n^{th} state vector represents how likely it is that the rat presses each lever in the n^{th} round. Therefore,

$$\mathbf{x}_0 = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3} \right).$$

The next time, we have a linear combination of vectors. A third of the time, the rat will press lever 1 and the next vector will be $(0.1, 0.45, 0.45)$. A third of the time, it will press lever II and the next vector will be $(0.375, 0.25, 0.375)$. The remaining third of a time it will press lever III and the next vector will be $(0.1, 0.1, 0.8)$.

Example, Part 3

We want to take the weighted average of those vectors, the linear combination with weights corresponding to the entries in the previous vector. In other words, we want the matrix-vector product

$$\begin{bmatrix} 0.1 & 0.375 & 0.1 \\ 0.45 & 0.25 & 0.1 \\ 0.45 & 0.375 & 0.8 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 0.18975 \\ 0.264 \\ 0.53625 \end{bmatrix}$$

This gives the linear combination we want, and notice that the weights will always be the probability vector from the previous step and the vectors being combined will always be the columns of the matrix above. That is, that matrix is the stochastic matrix for this Markov chain.

Return of Son of Example

What are the rat's probabilities of choosing each of the three levers on its third press? On its fourth press? We multiply $\mathbf{x}_0$ by the stochastic matrix to get $\mathbf{x}_1$, then again to get $\mathbf{x}_2$, and so forth.

$$\mathbf{x}_1 = \begin{bmatrix} 0.18975 \\ 0.264 \\ 0.53625 \end{bmatrix}, \ \mathbf{x}_2 = \begin{bmatrix} 0.1716 \\ 0.205 \\ 0.6134 \end{bmatrix}, \ \mathbf{x}_3 = \begin{bmatrix} 0.1554 \\ 0.1898 \\ 0.6448 \end{bmatrix}$$

On press 3, $\mathbf{x}_2$ is the appropriate vector. The rat has a more than 60% chance of pressing the food lever, and about a 17% and 20% chance of pressing the shock and noise levers, respectively. On press 4 the food probability has gone up a little to almost 65%, and the others have dropped to about 15% and 19%.
In fact if you keep multiplying you find the vectors start looking very much the same, approximately $(0.148, 0.177, 0.665)$.

Steady-State Vectors

The way the state vectors in our example seem to home in on a single vector is neither imaginary nor an artifact of the example.

A stochastic matrix P is *regular* if some power of P contains only strictly positive entries.

Stochastic Matrix Theorem: If P is a square regular stochastic matrix, there is a unique vector $\mathbf{q}$ such that $P\mathbf{q} = \mathbf{q}$. This $\mathbf{q}$ is called the *steady-state vector* (or *equilibrium vector*) for P. Furthermore, is $\mathbf{x}_0$ is any initial state, and the Markov chain $\{\mathbf{x}_k\}$ is generated from $\mathbf{x}_0$ by P, then $\{\mathbf{x}_k\}$ converges to $\mathbf{q}$ as $k \to \infty$.

Finding Steady-State Vectors

To find an equilibrium vector, notice that $P\mathbf{x} = \mathbf{x} = I\mathbf{x}$ can be solved as $(P - I)\mathbf{x} = \mathbf{0}$. We know how to solve for $\mathbf{x}$ in that case, obtaining the null space of $P - I$. Within that null space, if P is regular, there will be a probability vector (note $\mathbf{0}$ is not a probability vector). That probability vector is $\mathbf{q}$.

For our example,

$$P - I = \begin{bmatrix} -0.9 & 0.375 & 0.1 \\ 0.45 & -0.75 & 0.1 \\ 0.45 & 0.375 & -0.2 \end{bmatrix}.$$

Finding Steady-State Vectors II

Our $P - I$ reduces to

$$\begin{bmatrix} -0.45 & 0 & 0.1 \\ 0 & -0.5625 & 0.15 \\ 0 & 0 & 0 \end{bmatrix} \text{ and then to } \begin{bmatrix} 1 & 0 & -2/9 \\ 0 & 1 & -4/15 \\ 0 & 0 & 0 \end{bmatrix}.$$

Hence the null space is spanned by the vector $(\frac{2}{9}, \frac{4}{15}, 1)$. Find a scalar multiple of that which is a probability vector: $c(\frac{2}{9} + \frac{4}{15} + 1) = 1$ when $c = \frac{45}{67}$, so we take $\mathbf{q} = (\frac{10}{67}, \frac{12}{67}, \frac{45}{67})$.

One could check that $P\mathbf{q} = \mathbf{q}$. If we write $\mathbf{q}$ in decimals, we get $(0.149, 0.179, 0.672)$, very close to the $(0.148, 0.177, 0.665)$ that appeared from computation to be the steady state (most likely working the example with 0.33 instead of $\frac{1}{3}$ made the difference).

Text Generation

One can create toys with Markov chains to generate nonsense or parody text. The title slide is from “Alice in Elsinore”, generated by a Markov algorithm from the texts of *Alice in Wonderland* and *Hamlet*.

Running that algorithm on a single text will generate something that cosmetically or stylistically resembles the original text, but upon reading reveals itself to have nonsensical twists and turns (this may or may not differentiate it from the original).

How Does It Work?

The pieces we are working with are *n*-tuples of words found in the source text. Suppose we are working with pairs (2-tuples) and the following text, ignoring punctuation.

I scream, you scream, we all scream for ice cream!

Then our 2-tuples are, in order of appearance:

I scream, scream you,
you scream, scream we,
we all, all scream,
scream for, for ice,
ice cream

We could also wrap around at the end, adding one more pair.

The Stochastic Matrix

A probability vector has entries corresponding to the tuples, so in our example it would be a vector in $\mathbb{R}^9$ (or $\mathbb{R}^{10}$ with wrap-around).

The stochastic matrix represents the probability for each possible word w_{n+1} if the current n-tuple of words is $w_1, w_2, \ldots, w_n$. It does so by assigning a probability of 0 to each n-tuple that doesn't begin $w_2, \ldots, w_n$, and assigning probabilities to the tuples that *do* begin appropriately according to their frequency in the text.

[Technical note: the probability is the number of times the $(n+1)$-tuple $w_1, \ldots, w_{n+1}$ appears divided by the number of times the n-tuple $w_1, \ldots, w_n$ appears. However, for our example we will simply use the number of time w_2, w_3 appears divided by the number of times w_2 appears, because if we include w_1 our hands are tied: the only phrase we can produce is the original.]

Recombining Tuples

In our example the stochastic matrix will have a lot of zeros (it will be *sparse*), because not many of the pairs overlap. In fact, almost every column (each of which corresponds to one starting pair) will be a column of the identity matrix, because there is only one word that could follow. The columns corresponding to pairs ending in "scream" will have nonzero entries in the rows corresponding to "scream for", "scream we", and "scream you", each with value $\frac{1}{3}$.

Here's the algorithm: Start with a vector with a 1 in the first position and 0s elsewhere (so the new text begins with the same tuple as the original). Multiply by the stochastic matrix to get a probability vector. Use a random value to pick the next tuple using the new vector. Then discard that vector and start anew with a vector that has a 1 in the position corresponding to the tuple you just picked and 0s elsewhere.

A New Jingle

Since our matrix is simple, we can explain our procedure without writing it out. Start with the pair "I scream". Pick a random number from 1 to 100: 41. This is between 33 and 66 so the next pair is "scream we". Now we're fixed for a while: "we all", "all scream". Now pick another random number: 11. This is below 33 so the next pair is "scream you". Fixed for one pair: "you scream". Another random number: 71. This is above 66 so we pick "scream for". Now we're fixed to the end: "for ice", "ice cream". No pair begins with "cream" (unless we wrap around) so we've hit a dead end.

What did we get? Inserting punctuation, the new phrase is

I scream, we all scream, you scream for ice cream!

A Better Example

Our toy example isn't so exciting. Using the program that gave the title slide text, a colleague of mine combined an early 20th century cookbook with a text on semiotics. The output had to be significantly culled, but yielded gems such as the following:

> *Make a cream of one paradigm (e.g. a particular paradigm rather than a workable alternative was used in a dripping pan; put into pie plates; grease the tops of loaves over with butter. This will make it equal.*
>
> *[In this recipe, the term "mango" refers to its socio-cultural and personal associations (ideological, emotional etc.). Roland Barthes argued that such paired paradigms consist of an 'unmarked' and a tablespoon of whole cloves, and nutmeg to taste.*

A More Practical Application

How does Google decide which webpages should show up on page 1 of your search results and which on page 10?

> *In this paper, we have taken on the audacious task of condensing every page on the World Wide Web into a single number, its PageRank.*

Page, Brin, Motwani, and Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Technical Report, 1999.

Setting up a Matrix

PageRank begins with a matrix A with rows and columns corresponding to webpages (row i and column i both correspond to the same page). The entries are determined by whether there is a link from the row page to the column page, and how many links are on the row page.

Let $A = [a_{uv}]$. If there is no link from page u to page v, $a_{uv} = 0$. Otherwise, let n_u be the total number of pages that u links to, and set $a_{uv} = 1/n_u$.

Dividing by the number of links on the page decreases the effect of pages that are nothing but lists of links, which might otherwise artificially inflate a page's rank.

Using the Matrix

The PageRank vector $\mathbf{r}$ has as entries the ranks of every webpage. It is the equilibrium vector of the matrix A.

Why the equilibrium vector? We would like a page that itself has high rank to share that high rank with the pages it links to (having a link on the main Dartmouth website should count for more than having a link on the linear algebra course website). We do this by taking powers of A until the multiplication no longer changes the ranks of webpages.

Technical Issues

- The matrix A is not necessarily a stochastic matrix to begin with. To make sure the total rank of all pages stays constant (e.g., stays at 1 so the vector $\mathbf{r}$ is a probability vector) we must multiply by a scaling factor c, so $\mathbf{r}$ actually satisfies $cA\mathbf{r} = \mathbf{r}$.
- Technical problems are introduced by circular pointers (say, two webpages that point to each other but nowhere else) and dangling links (links to pages which have no outgoing links, or to pages Google has not downloaded yet). These are fairly straightforward to solve.
- Most entries of A will be 0. This means much storage space is saved by remembering not A, but instead just its nonzero entries and their positions.

More Technical Issues

- The equilibrium vector of this matrix is found by multiplying repeatedly by the matrix, rather than finding a probability vector in the null space of $P - I$. This is for three reasons:
  1. Because the matrix is very sparse, multiplication is quicker than solving the enormous system of equations.
  2. An approximation of the rank vector is good enough for Google's purposes.
  3. When Google updates their information, the matrix changes. The previous rank vector is still a decent approximation to the rank, so using it as the initial state and multiplying several times by the matrix allows for a fast update of the rank vector.
- By assigning the link weights more cleverly than $1/n_u$, one can get better results and decrease the ability of people to artificially boost their pages' rankings.

Resources

Text on title page is from
`www.eblong.com/zarf/markov/alice_in_elsinore.txt`
(about halfway down)

Main page: `www.eblong.com/zarf/markov/`

PageRank paper is available from
`dbpubs.stanford.edu/pub/1999-66`

A PageRank example: `www.mathworks.com/company/newsletters/news_notes/clevescorner/oct02_cleve.html`

Markov Toys

Random Word Generator:
`www.fourteenminutes.com/fun/words/`

Mark V. Shaney text generator:
`www.yisongyue.com/shaney/`

Markov chains for secret messages:
`www.zentastic.com/entries/200503031618.html`

Random numbers obtained from `www.random.org`