Markov Chains

Previously, we solve problems such as:

> *Two teams, A and B, are playing a match: the fist team to win 2 games wins the match. If team A has a 70% chance of winning any given game, what is the probability that team A wins the match?*

With this problem, the maximum number of games you play is finite: at most you'll play 5 games. A slightly different problem is requiring that a team be *up* 3 games to win the match

> *Two teams, A and B, are playing a match: the fist team to win* ***by*** *2 games wins the match. If team A has a 70% chance of winning any given game, what is the probability that team A wins the match?*

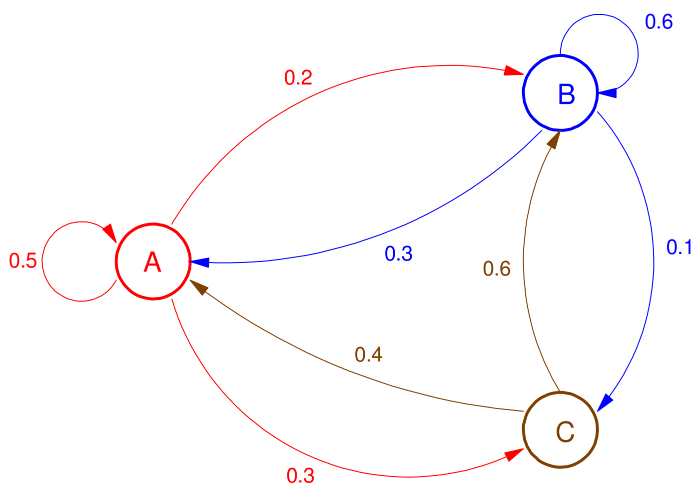
This is a *totally* different problem. If team A wins then B wins, you're back were you started. Likewise, this match could in theory go on to infinity.

To solve the latter problem, we need a different approach. Markov chains are one such approach.

A Markov chain is a discrete-time probability function where

- X(k) is the state of the system at time k, and
- X(k+1) = A X(k)

A classic problem with Markov chains is as follows:



Markov Chain: Three people toss a ball back and forth with probability of A tossing to B being defined.

Three people, A, B, and C, are playing ball. Every second they pass the ball at random:

- When A has the ball, he/she
  - Keeps the ball 50% of the time
  - Passes it to B 20% of the time, and
  - Passes it to C 30% of the time
- When B has the ball, he/she
  - Passes it to A 30% of the time

  - Keeps it 60% of the time, and
  - Passes it to C 10% of the time
- When C has the ball, he/she
  - Passes it to A 40% of the time, and
  - Passes it to B 60% of the time.

Assume at t=0, A has the ball.

- What is the probability that B will have the ball after k tosses?
- After infinite tosses?

This lecture covers three different methods to analyze problems of this sort:

- Matrix multiplication
- Eigenvalues and Eigenvectors, and
- z-Transforms.

Solution #1: Matrix Multiplication

Let X(k) be the probability that A, B, and C have the ball at time k:

$$x(k) = \begin{bmatrix} p(a) \\ p(b) \\ p(c) \end{bmatrix}$$

Then from the problem statement, X(k+1) is related to X(k) by a state-transition matrix:

$$x(k+1) = \begin{bmatrix} 0.5 & 0.3 & 0.4 \\ 0.2 & 0.6 & 0.6 \\ 0.3 & 0.1 & 0 \end{bmatrix} x(k) = A\ x(k)$$

Note that

- The columns are the probabilities in the above problem statement, and
- The columns must add up to 1.000 (all probabilities add to one)

Initially, A has the ball:

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

Starting with the initial condition, you can determine x(1), x(2), x(3), etc. simply by multiplying by the state transition matrix:

The probability of each player having the ball after 1, 2, 3 tosses is (using Matlab)

```
A = [0.5,0.2,0.3 ; 0.3,0.6,0.1 ; 0.4,0.6,0]'

    0.5000    0.3000    0.4000
    0.2000    0.6000    0.6000
    0.3000    0.1000         0


X = [1;0;0]         % k = 0

    1.0000
    0.0000
    0.0000

X = A*X             % k = 1

    0.5000
    0.2000
    0.3000

X = A*X             % k = 2

    0.4300
    0.4000
    0.1700

X = A*X             % k = 3

    0.4030
    0.4280
    0.1690
```

time passes

```
X = A*X             % k = 100

    0.3953
    0.4419
    0.1628
```

Eventually X quits changing. This is the steady-state solution.

If you want to find the steady-state solution, you can simply raise A to a large number (like 100) and solve in one shot:

```
X0 = [1;0;0]

     1
     0
     0

X20 = A^100 * X0

    0.3953
    0.4419
    0.1628
```

You can also solve for the steady-state solution by finding x(k) such that

$$x(k+1) = A\ x(k)$$

subject to

$$x(k+1) = x(k)$$

Solving:

$$(A - I)x(k) = 0$$

$$\left(\begin{bmatrix} 0.5 & 0.3 & 0.4 \\ 0.2 & 0.6 & 0.6 \\ 0.3 & 0.1 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0$$

$$\begin{bmatrix} -0.5 & 0.3 & 0.4 \\ 0.2 & -0.4 & 0.6 \\ 0.3 & 0.1 & -1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0$$

Assume c = 1

$$\begin{bmatrix} -0.5 & 0.3 \\ 0.2 & -0.4 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = -\begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}$$

```
ab = -inv([-0.5,0.3;0.2,-0.4])*[0.4;0.6]

    2.4286
    2.7143
```

So an (invalid) solution is

```
X = [ab;1]

    2.4286
    2.7143
    1.0000
```

This is invalid since the sum isn't 1.0000 (all probabilities add to one). Scaling to make the sum one:

```
X = X / sum(X)

    0.3953
    0.4419
    0.1628
```

which is the same answer we got before.

Solution #2: Eigenvalues and Eigenvectors

The problem we're trying to solve is

$$x(k+1) = A\ x(k)$$

subject to

$$x(0) = X_0$$

This is actually an eigenvalue / eigenvector problem.

- Eigenvalues tell you how the system behaves,
- Eigenvectors tell you what behaves that way.

Since this system has three states, the generalized solution for x(k) will be:

$$x(k) = a_1\Lambda_1\lambda_1^k + a_2\Lambda_2\lambda_2^k + a_3\Lambda_3\lambda_3^k$$

where

- λ_i is the ith eigenvalue,
- Λ_i is the ith eigenvector, and
- a_i is a constant depending upon the initial condition.

At k = 0:

$$x(0) = \left[\begin{array}{ccc} \Lambda_1 & \Lambda_2 & \Lambda_3 \end{array}\right]\left[\begin{array}{c} a_1 \\ a_2 \\ a_3 \end{array}\right]$$

The excitation of each eigenvector is then

```
X0 = [1;0;0]

      1
      0
      0

A123 = inv(M)*X0

     0.6149
     0.5482
     0.9354
```

meaning

$$x(k) = 0.6149\left[\begin{array}{c} 0.6430 \\ 0.7186 \\ 0.2468 \end{array}\right](1)^k + 0.5482\left[\begin{array}{c} 0.2222 \\ 0.5693 \\ -0.7915 \end{array}\right](-0.1562)^k + 0.9543\left[\begin{array}{c} 0.5151 \\ -0.8060 \\ 0.2989 \end{array}\right](0.2562)^k$$

or adding the scalars to the eigenvectors:

```
W = inv(M)*X0

    0.6149
    0.5482
    0.9354

M * diag(W)

    0.3953    0.1218    0.4828
    0.4419    0.3121   -0.7539
    0.1628   -0.4339    0.2711
```

$$x(k) = \begin{bmatrix} 0.3953 \\ 0.4419 \\ 0.1628 \end{bmatrix}(1)^k + \begin{bmatrix} 0.1218 \\ 0.3121 \\ -0.4339 \end{bmatrix}(-0.1562)^k + \begin{bmatrix} 0.4828 \\ -0.7539 \\ 0.2711 \end{bmatrix}(0.2562)^k$$

As k goes to infinity, the first eigenvector is all that remains.

Note that the steady-state solution is simply the eigenvector associated with the eigenvalue of 1.000. If you are interested in the steady-state solution, then this is the only thing you really care about.

In Matlab, determine the eigenvalues and eigenvectors:

```
A =

    0.5000    0.3000    0.4000
    0.2000    0.6000    0.6000
    0.3000    0.1000         0

[M,V] = eig(A)

M =     eigenvectors

    0.6430    0.2222    0.5161
    0.7186    0.5693   -0.8060
    0.2648   -0.7915    0.2898


V =      eigenvelues

    1.0000         0         0
         0   -0.1562         0
         0         0    0.2562
```

The eigenvector associated with the eigenvalue of 1.000 is the one we want (shown in blue). Scale this so that it is a valid probability (i.e. the sum of its values is 1.000) and you have the steady-state solution.

```
X = M(:,1);
X = X / sum(X)

    0.3953
    0.4419
    0.1628
```

Solution #3: z-Transforms

Again, the problem we are trying to solve is

$$x(k+1) = \begin{bmatrix} 0.5 & 0.3 & 0.4 \\ 0.2 & 0.6 & 0.6 \\ 0.3 & 0.1 & 0 \end{bmatrix} x(k)$$

subject to

$$x(0) = X_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

This can be written as

$$x(k) = A\ x(k-1) + X_0\ \delta(k)$$

or

$$x(k+1) = A\ x(k) + X_0\ \delta(k+1)$$

Take the z-transform

$$zX = AX + zX_0$$

To determine the probability that B has the ball at time k, look at the second state

$$Y = CX = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} X$$

Solving for Y then gives the z-transform for b(k)

$$zX = AX + zX_0$$

$$(zI - A)X = zX_0$$

$$X = z(zI - A)^{-1} X_0$$

$$Y = zCX = zC(zI - A)^{-1}X_0$$

For our 3x3 example,

$$Y(z) = z\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}\left(\begin{bmatrix} z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & z \end{bmatrix} - \begin{bmatrix} 0.5 & 0.3 & 0.4 \\ 0.2 & 0.6 & 0.6 \\ 0.3 & 0.1 & 0 \end{bmatrix}\right)^{-1}\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

This is somewhat painful to compute by hand. Fortunately, there's Matlab to the rescue.

- G = ss(A, B, C, D, T) *input a dynamic system into matlab*
- Y = tf(G) *calculate and express the z-transform of Y in transfer function form*
- Y = zpk(G) *calculate and express the z-transform of Y in factored form*

The matlab command *ss* allows you to input a discrete-time system of the form

$$zX = AX + BU$$

$$Y = CX + DU$$

with sampling rate T (T = 1 here). Putting this system in the same form

$$zX = AX + zX_0$$

$$Y = CX$$

In matlab:

```
A = [0.5,0.3,0.4;0.2,0.6,0.6;0.3,0.1,0]

    0.5000    0.3000    0.4000
    0.2000    0.6000    0.6000
    0.3000    0.1000         0

X0 = [1;0;0]

     1
     0
     0
C = [0,1,0]

     0     1     0

Bz = ss(A, X0, C, 0, 1);

tf(Bz)

        0.2 z + 0.18
-----------------------------
z^3 - 1.1 z^2 + 0.06 z + 0.04
```

```
Sampling time (seconds): 1
```

Multiply by z to get the B(z)

```
zpk(Bz)

          0.2 (z+0.9)
----------------------------
(z-1) (z-0.2562) (z+0.1562)

Sampling time (seconds): 1
```

Again, multily by z to get B(z). This means that the probability that B has the ball at time k is

$$B(z) = \left(\frac{0.2z(z+0.9)}{(z-1)(z-0.2562)(z-0.1562)} \right)$$

To find b(k), factor out a z and expand this using partial fractions:

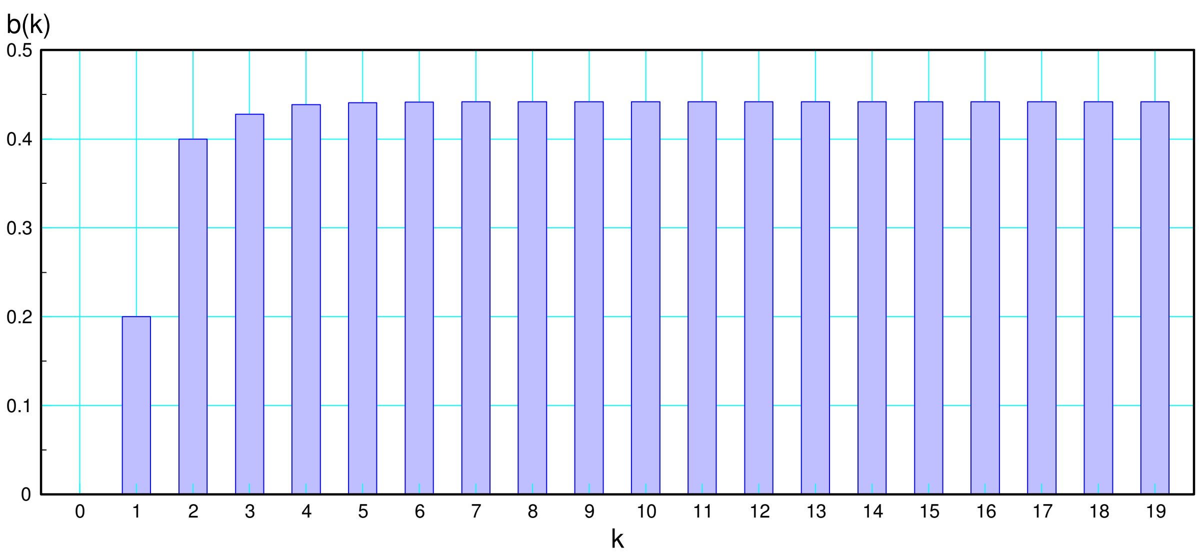
$$B(z) = \left(\left(\frac{0.6054}{z-1} \right) + \left(\frac{-3.1089}{z-0.2562} \right) + \left(\frac{2.5034}{z-0.1562} \right) \right) z$$

This isn't in the table of z-transforms, so multiply both sides by z

$$B = \left(\frac{0.6054z}{z-1} \right) + \left(\frac{-3.1089z}{z-0.2562} \right) + \left(\frac{2.5034z}{z-0.1562} \right)$$

Take the inverse z-transform (i.e. use the table of z-transforms

$$b(k) = \left(0.6054 - 3.1089(0.2562)^k + 2.5034(0.1562)^k \right) u(k)$$



Probability that player B has the ball after toss k

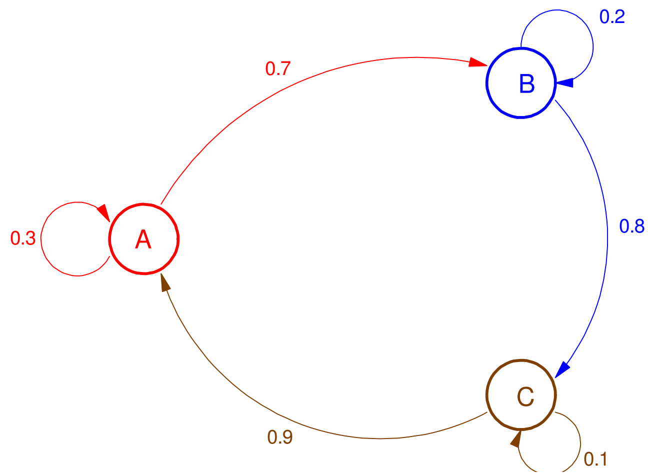
z-Transform with Complex Poles

You can get complex poles. If you do, use entry in the z-transform table:

$$\left(\frac{(a\angle\theta)z}{z-b\angle\phi}\right)+\left(\frac{(a\angle-\theta)z}{z-b\angle-\phi}\right)\rightarrow 2a\,b^{k}\,\cos(\phi k-\theta)\,u(k)$$

For example, suppose player A, B, and C toss the ball as:

- A keeps the ball 30% of the time and passes it to B 70% of the time
- B keeps the ball 20% of the time and passes it to C 80% of the time, and
- C keeps the ball 10% of the time and passes it to A 90% of the time



Ball tossing game with complex poles

Suppose A starts with the ball at k = 0. Determine the probability that B has the ball after k tosses.

Solving using z-transforms: express in matrix form

$$zX=\begin{bmatrix}0.3 & 0 & 0.9\\ 0.7 & 0.2 & 0\\ 0 & 0.8 & 0.1\end{bmatrix}X \qquad X(0)=\begin{bmatrix}1\\0\\0\end{bmatrix}$$

$$Y=p(B)=\begin{bmatrix}0 & 1 & 0\end{bmatrix}X$$

Find B(z) using Matlab

```
A = [0.3,0,0.9;0.7,0.2,0;0,0.8,0.1]

    0.3000         0    0.9000
    0.7000    0.2000         0
         0    0.8000    0.1000

X0 = [1;0;0];
C = [0,1,0];
Bz = ss(A,X0,C,0,1);
zpk(Bz)

       0.7 (z-0.1)
-------------------------
(z-1) (z^2 + 0.4z + 0.51)


Sampling time (seconds): 1
```

or (multiplying by z)

$$B(z) = \left(\frac{0.7z(z-0.1)}{(z-1)\left(z-0.7142\angle 106^0\right)\left(z-0.7142\angle -106^0\right)} \right)$$

Pull out a z and expand using partial fractions

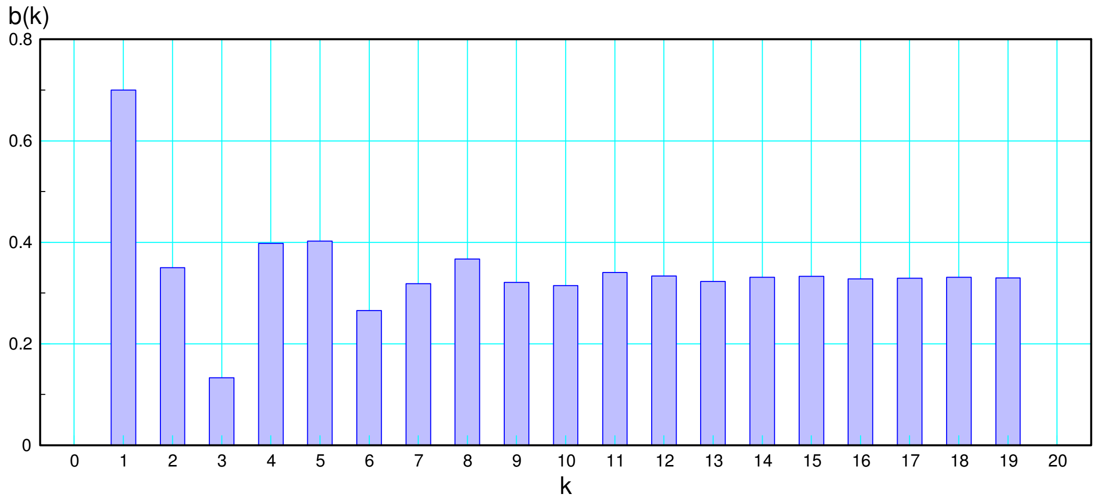
$$B(z) = \left(\left(\frac{0.3298}{(z-1)} \right) + \left(\frac{0.2764\angle -126.8^0}{\left(z-0.7142\angle 106^0\right)} \right) + \left(\frac{0.2764\angle 126.8^0}{\left(z-0.7142\angle -106^0\right)} \right) \right) z$$

Multiply both sides by z

$$B = \left(\frac{0.3298z}{(z-1)} \right) + \left(\frac{z0.2764\angle -126.8^0}{\left(z-0.7142\angle 106^0\right)} \right) + \left(\frac{z0.2764\angle 126.8^0}{\left(z-0.7142\angle -106^0\right)} \right)$$

Take the inverse z-transform

$$b(k) = \left(0.3298 + 0.5527(0.7142)^k \cos\left(k \cdot 106^0 + 126.8^0\right)\right) u(k)$$



probability that player B has the ball after k tosses