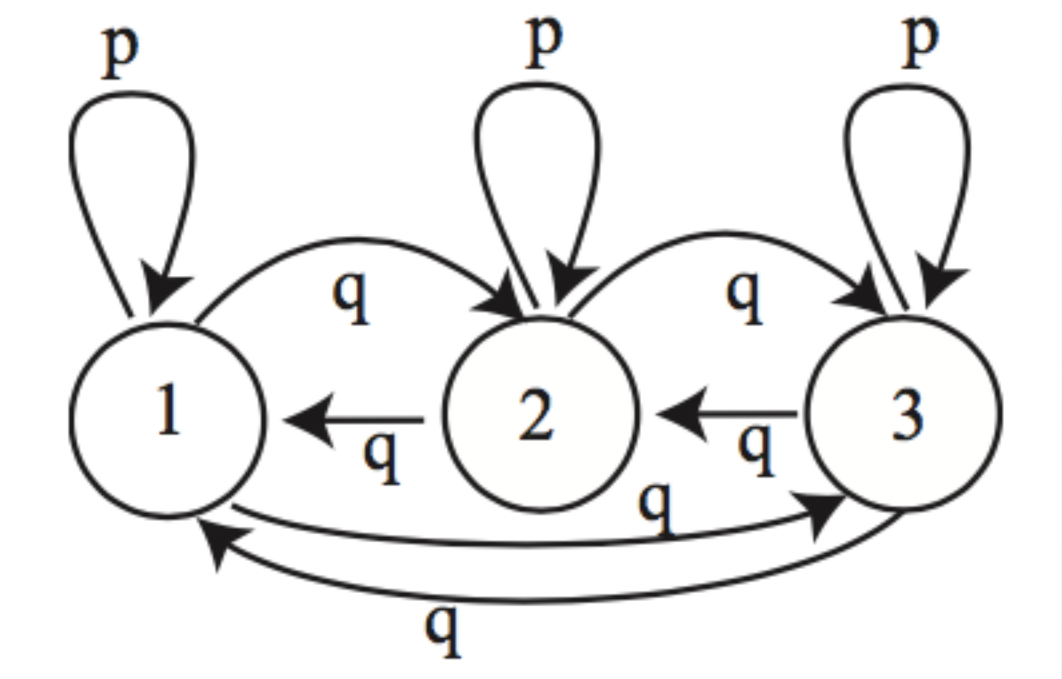
October 10, 2017

CS 361: Probability & Statistics

Markov chains & simulation

Markov Chains



Picture a bug being placed on one of the nodes of this weighted directed graph. Its initial placement is determined by some probability distribution. Then, it walks from node to node according to the transition probabilities indicated in the arrows.

As it walks, the states that it visits (state 1 through state k) are a sequence of random variables. The random variable X_n=j if the bug was on state *j* at time step *n*

Markov chains

If we have such a setup—a distribution for the initial state, a set of transition probabilities for each state, and a sequence of random variables that take values indicating the state of the process at time *n*—and one additional constraint is met we have a Markov chain

The additional constraint is that the probability that the process is in state *j* at time *n* depends only on the state that the process is in at time *n-1*

$$P(X_n = j | \text{all previous states visited}) = P(X_n = j | X_{n-1})$$

Markov chains

Any model built by putting probabilities on the transitions of a finite state diagram is a Markov Chain but this isn't the only way to build or represent them

Another representation is to use a matrix to encode the transition probabilities from state *i* to state *j*

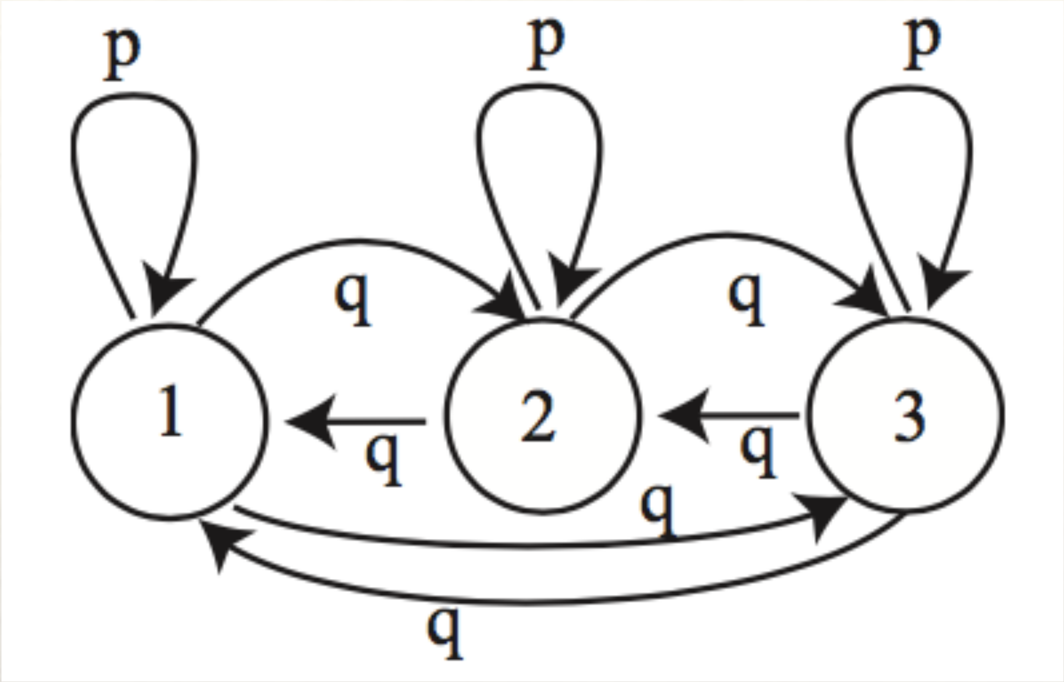
Define a matrix *P* such that

$$p_{ij} = P(X_n = j | X_{n-1} = i)$$

Note that this matrix will satisfy $p_{ij} \geq 0$ and $\sum_j p_{ij} = 1$

Example

Write the transition matrix for the virus example with a=0.2



p = 1-a

q = a/2

Recall:

$$p_{ij} = P(X_n = j | X_{n-1} = i)$$

$$P= \begin{pmatrix} 0.8 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{pmatrix}$$

Markov chains

Write π for the k dimensional row vector corresponding to the probability distribution of the initial state at time 0

So for example, when we write π_i it means $P(X_0 = i)$

Then if we want to calculate a probability distribution for the state at time 1, we have

$$\begin{aligned} P(X_1 = j) &= \sum_i P(X_1 = j, X_0 = i) && \text{law of total probability} \\ &= \sum_i P(X_1 = j | X_0 = i) P(X_0 = i) && \text{Bayes rule} \\ &= \sum_i p_{ij} \pi_i. && \text{Transition matrix and initial distribution} \end{aligned}$$

Markov chains

If we write $\mathbf{p}^{(n)}$ to represent the probability distribution over states at time n then what we figured out on the last slide is that

$$\mathbf{p}_j^{(1)} = \sum_i p_{ij}\pi_i \qquad \text{or} \qquad \mathbf{p}^{(1)} = \pi \mathcal{P}$$

Now consider time 2

$$\begin{aligned} P(X_2 = j) &= \textstyle\sum_k P(X_2 = j, X_1 = k) \\ &= \textstyle\sum_k P(X_2 = j | X_1 = k) P(X_1 = k) \\ &= \textstyle\sum_k P(X_2 = j | X_1 = k) \mathbf{p}_k^{(1)} \\ &= \textstyle\sum_k p_{kj} \sum_i p_{ik}\pi_i \end{aligned}$$

I.e. we have

$$\mathbf{p}^{(2)} = \pi P^2$$

Markov chains

And in general for Markov chains with transition probabilities given by P and initial state given by π we have

$$\mathbf{p}^{(n)} = \pi P^n$$

Example

With our earlier example for viruses, for a=0.2, we can compute the distribution of states after two state transitions, given that the virus starts in strain 1 as

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}^2 = \begin{bmatrix} 0.66 & 0.17 & 0.17 \end{bmatrix}$$

We also have

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}^{20} = \begin{bmatrix} 0.3339 & 0.3331 & 0.3331 \end{bmatrix}$$

Over time, the process seems to "forget" which state it starts in

Stationary distribution

This forgetting is not atypical. As long as it is possible to reach any state in a Markov chain from any other state, the chain is called **irreducible**. And irreducible Markov chains exhibit the following

For any initial state distribution there is a unique vector *s* called the **stationary distribution** of the Markov chain, given by

$$\lim_{n \to \infty} \pi \mathcal{P}^{(n)} = \mathbf{s}.$$

Simulation

Simulating for expectation

If we want to compute the expected value of a function of random variables, the law of large numbers tells us that we can use simulation

If our x_i come from a random number generator and f(x_i) = f_i, then defining

$$F_N = \frac{\sum_{i=1}^{N} f_i}{N}.$$

we have

$$\lim_{N \to \infty} P(\{\| F_N - \mathbb{E}[F] \| > \epsilon\}) = 0.$$

Example

Let X be a random variable uniformly distributed in the range 0 to 1. Let Y be a random variable uniformly distributed from 0 to 10. For Z=(Y-5X)^3 - X^2, what is Var(Z)?

```
import random

def variance(data):
    n = len(data)
    total = 0.0
    totalsq = 0.0
    for value in data:
        total += value
        totalsq += value*value
    return totalsq/n - pow(total/n, 2)

if __name__ == "__main__":
    data = []
    for i in range(100000):
        x = random.random()
        y = random.uniform(0,10)
        z = pow((y-5*x), 3) - pow(x,2)
        data.append(z)
    print variance(data)
```

Using

$$Var(Z) = E[Z^2] - E[Z]^2$$

gives Var(Z) is approximately 27,000

Probabilities with simulations

Recall the indicator random variable

Definition: 5.16 *Indicator functions*

An indicator function for an event is a function that takes the value zero for values of X where the event does not occur, and one where the event occurs. For the event $\mathcal{E}$, we write

$$\mathbb{I}_{[\mathcal{E}]}(X)$$

for the relevant indicator function.

We have $\mathbb{E}[\mathbb{I}_{[\mathcal{E}]}] = P(\mathcal{E})$ which means we can use our insight for calculating expectations in order to calculate probabilities

By counting occurrences of the event in our simulations $\frac{\#(\mathcal{E})}{N}$

Example

Let X be a random variable uniformly distributed in the range 0 to 1. Let Y be a random variable uniformly distributed from 0 to 10. For Z=(Y-5X)^3 - X^2, what is P(Z>3)?

```
import random

def probability(data):
    n = len(data)
    event_count = 0
    for value in data:
        if value > 3:
            event_count += 1
    return float(event_count) / n

if __name__ == "__main__":
    data = []
    for i in range(100000):
        x = random.random()
        y = random.uniform(0,10)
        z = pow((y-5*x), 3) - pow(x,2)
        data.append(z)
    print probability(data)
```

Count how many times Z > 3

Giving P(Z > 3) is approximately 0.6

Simulations as random variables

Our estimates for expectations and probabilities are themselves random variables since they are a number associated with a random process or experiment

Each time we run our simulation, we will get a different estimate

Given that our simulation-derived estimates are sums of independent random variables, our estimates follow a normal distribution. In other words, if we do multiple simulations and record the estimates we get each time, the estimates will be normal data

This is good because it means that the standard deviation of a set of estimates is informative about what range of values we would be likely to observe if we ran the simulation a very large number of times

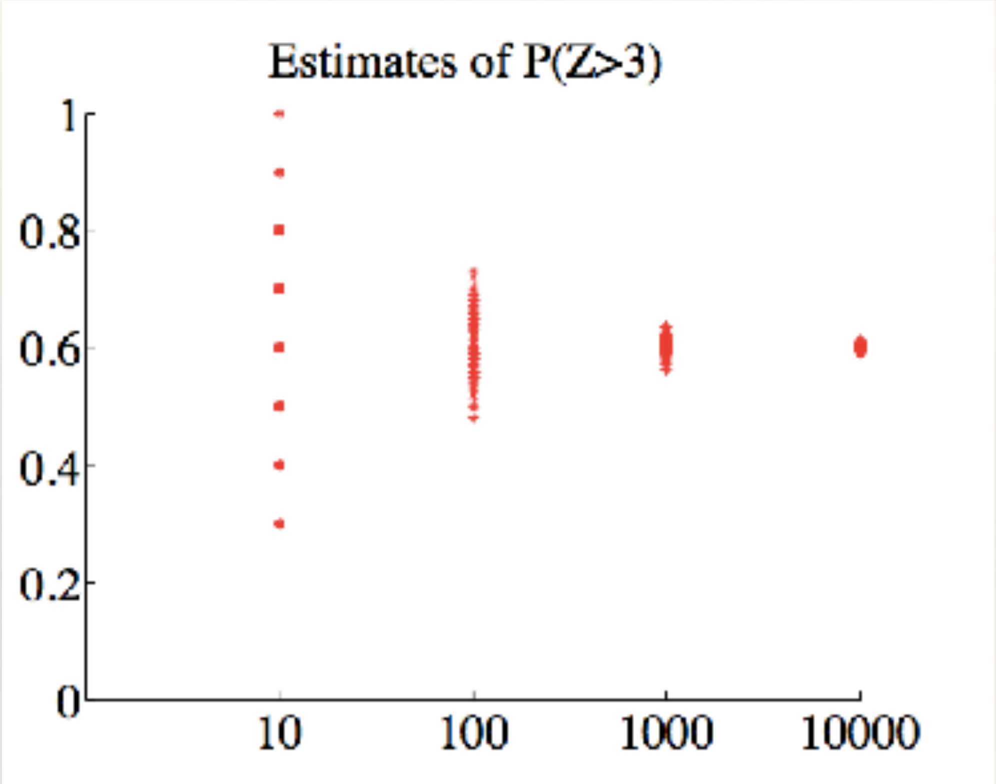
Simulations as random variables

In general, if N is the number of samples in our simulation, the standard deviation of our estimates should be dictated by

$$\frac{C}{\sqrt{N}}$$

which means, i.e. to double our estimate's accuracy we need 4X as many samples in our simulation

C is hard to compute and depends on the problem. It will be much larger for computing the probability of low probability events, for instance



Example

Worked example 7.11 *Getting 14's with 20-sided dice*

You throw 3 fair 20-sided dice. Estimate the probability that the sum of the faces is 14 using a simulation. Use $N = [1e1, 1e2, 1e3, 1e4, 1e5, 1e6]$. Which estimate is likely to be more accurate, and why?

Example

Worked example 7.14 *A Queue*

A bus is supposed to arrive at a bus stop every hour for 10 hours each day. The number of people who arrive to queue at the bus stop each hour has a Poisson distribution, with intensity 4. If the bus stops, everyone gets on the bus and the number of people in the queue becomes zero. However, with probability 0.1 the bus driver decides not to stop, in which case people decide to wait. If the queue is ever longer than 15, the waiting passengers will riot (and then immediately get dragged off by the police, so the queue length goes down to zero). What is the expected time between riots?