ORIE 5530: Modeling Under Uncertainty

Lecture 1 (Introduction)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

We dedicated the first class mostly to motivation and to get us thinking about probability – think of our first meeting as a warmup. The purpose of the warmup is to have us start thinking (and carefully so) in terms of mathematical models of uncertainty. We will embark on our analytical journey starting next class.

Below is a bullet-point list of some examples I discussed informally in this first class. These are examples for motivation and to get us thinking. I do not go into any of these in much detail but I will return to these examples (or variants thereof) once we can ground our answers in models.

1. What is probability?
   (a) Frequency-based approach (how many Heads out of 100 tosses of a coin) and models. Say I just tossed a coin a 100 times. What is the likelihood that the 101st toss comes out Heads? Well, depends on what I know about the coin ... . Coin tossing is a basic probability model. Rolling a die is another basic probability model. Approach can provide nice/helpful intuition.
   (b) Axiomatic approach and models. Addresses some disadvantages of frequency-based approach, forming basis of modern probability.
2. Birthday problem and exploiting $\mathbb{P}[A] = 1 - \mathbb{P}[A^c]$, where event A^c is the complement of event A.
3. Single-order, multi-period inventory model as an illustration of stochastic optimization. Distributions are the simplest of probabilistic models but are very useful. It is important to be familiar with a basic set that is widely applicable.
4. Dynamic inventory as an example of a dynamical system.
5. Queueing (waiting line) models and queueing network models.
6. Workforce management models.
7. Portfolio example: This is intended to emphasize that the details of the model matter but how much they matter depends also on our objective; see more details below. We will revisit this example in a future class once we cover *correlations* and multi-dimensional decisions.
8. Conditional probabilities
   (a) Simple example: $\mathbb{P}[A] = 30\%$, where $A = \{\text{baseball batter gets a hit}\}$. $\mathbb{P}[A|L] = 40\%$ and $\mathbb{P}[A|R] = 20\%$, where $L = \{\text{left-handed pitcher}\}$ and $R = \{\text{right-handed pitcher}\}$.
   (b) Monty Hall problem (donkey and car): conditional probabilities and information updating.
9. Emergency room: An informal illustration of using Markov chains in a realistic setting. Evaluation for given cost, benefit and pricing optimization.

Portfolio optimization: An example. Say you have a \$100 that you want to invest in one of two investments A or B. Each will give you, in expectation, a 5% return in a year. This means that if you put your 100 on A, you will have (in expectation) \$5 in a year from now. Same for B.

Where would you put your money? It depends on two things:

1. Do you care about expected return or also about risk? If you only care about expected return you should not care. However, things are different if you care about risk. It could be that A gives you 5% for sure while B gives you 10% return with a 50% chance but 0 otherwise. So you might get 10% but you also might end up getting nothing.
2. If you care about risk, what do you know about the correlation between the two investment products? You could utilize this correlation to your advantage (which is what finance models do).

ORIE 5530: Modeling Under Uncertainty

Lecture 2 (Events)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

We spent most of class today on counting. This is useful and fundamental when outcomes are discrete (such as the number on a die or the number of graphs). Some of this goes back to basics you might be familiar with and my objective is to present these with a bit more elaborate and interesting examples.

When all outcomes on a probability space, $\mathcal{S}$, are equally likely, the probability of an outcome (a single point in the sample space) is

$$\frac{1}{\#\text{ of points in the space}}.$$

In the case of a fair die, there are six possible (and equally likely) outcomes and the probability of each is 1/6.

If we throw two dice the number of possible outcomes is 36 and each of the outcomes (i.e., each pair of numbers between 1 and 6) has a likelihood of 1/36. If we want to know the likelihood of a more complex outcomes such as the event $A = \{\text{the sum of the two tosses is 4}\}$ we need to be able to **count** the number of such sample points. There are 3 of these (check that you agree). The likelihood is $|A|$ (the size of A, or the number of points in it) divided by the size, $|\mathcal{S}|$, of the sample space $\mathcal{S}$ (the number of points). So in this case

$$\mathbb{P}\{\text{the sum of the two tosses is 7}\} = \frac{|A|}{|S|} = \frac{3}{36} = \frac{1}{12}.$$

With dice counting is simple. However, it can get fairly complicated; fortunately, there are some basic facts we can use.

Say you have n **different** items.

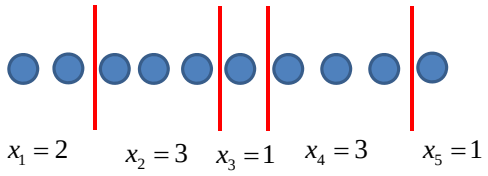
1. There are $n!$ ways of ordering the **n** different items. In other words, the number of ordered sets with n different items is $n!$.
2. There are $P(n,k) = \frac{n!}{(n-k)!}$ ways of obtaining an ordered subset of size $k \leq n$ items from the set of n items. $P(n,k)$ are called the number of k-permutations of n.
3. There are $C(n,k) = \frac{P(n,k)}{P(k,k)} = \frac{n!}{k!(n-k)!} = \binom{n}{k}$ ways of choosing k items out of n if you do not care about the order in which you choose them. $C(n,k) = \binom{n}{k}$ are called the binomial coefficients.
4. There are 2^n ways to choose a subset out of a set of n **different** items.

Two non-trivial examples

1. **Integral equations:** You have an equation

   $$x_1 + x_2 + x_3 + x_4 + x_5 = 10.$$

   The question is how many integer (i.e., in whole numbers) different solutions exist for this equation where all variables are **strictly positive**. For example, $x_1 = x_2 = x_3 = x_4 = x_5 = 2$ is a valid solution.



The key is to translate this question into the basic facts we have above. What are we choosing?

We have 10 balls and we are choosing where to place dividers. Everything on the left of the first divider goes to x_1, everything between the 1st and 2nd divider goes to x_2 and everything above the $4th$ divider goes to x_5; see the figure below for an example.

We have 9 spots where we can put the 4 dividers (after the first ball, after the second ball, ..., after the 9th ball). So we need to choose 4 out of 9 spots where to place the divider. The answer is thus $\binom{9}{4}$.

** **Bonus comment:** (I will use ** in my handouts for points/comments that are kind of "bonus" curiosities. You do not have to worry about these)

There is an interesting question as to what happens if we also allow some variables be 0. In that case, $x_1 = x_2 = x_3 = x_4 = 0$ and $x_5 = 10$ is a valid solution.

The simplest way to think about this, building on the previous result, is to have two steps:

- Choose which variables are going to be strictly positive. You are choosing a subset of the variables. There are, for example, 5 ways to choose a single variable to be positive, there are $\binom{5}{2}$ to choose 2, etc.
- For each option resolve the above. For example, if only 1 variables is strictly positive, once we chose this variable, it must equal to 10 for the sum to be 10. If two variables are strictly positive, given the chosen variables, there are (based on part 1) $\binom{9}{1}$ ways to make them strictly positive and sum to 10, etc.

So the solution would be

$$\sum_{i=1}^{5} \binom{5}{i} \binom{9}{i-1}$$

Notice that $\binom{5}{i} = \binom{5}{5-i}$ (the math says this but this is also intuitive – it is the same to choose i variables to be positive as to choose $5-i$ variables to be 0).

Then, the above is equal to

$$\sum_{i=1}^{5} \binom{5}{i} \binom{9}{i-1} = \sum_{i=1}^{5} \binom{5}{5-i} \binom{9}{i-1} = \binom{14}{4}.$$

If you want to convince yourself that this works, checkout the case where you have only two variables to start with. Then, there are 9 solutions where both variables are strictly positive (you choose the value of x_1 and then x_2 must equal $10 - x_1$) and you have 11 solutions in which one of them could be 0 (once you choose which one is 0, the other one must be 10).

There is an additional way, that requires a bit more creativity, to get to the same result.

2. **Graphs:** How many undirected graphs that have n nodes (individuals) can we build? A graph is defined by the nodes and the edges that connect them. So if the number of nodes is given, the question is only which edges do we include. This means we want to divide our analysis into two steps
   - How many possible edges are there?
   - How many graphs can we construct from these edges?

   Let's start from the second part. If there are m edges (let's use m as a placeholder and we will give it a value shortly), then building a graph is choosing a subset out of the m distinct edges. We know that there are 2^m ways of doing this. So,

   $$\text{Number of possible graphs} = 2^{\text{Number of possible edges}}.$$

   Now, an edge connects to individuals:

   $$\text{number of edges} = \text{number of distinct pairs of individuals.}$$

   How many pairs do we have? Namely, the ways to choose 2 individuals out of n. Hence, the number of edges m equals $\binom{n}{2}$.
   Combining things, we arrive at the conclusion that there are

   $$\text{Number of possible graphs} = 2^{\text{Number of possible edges}} = 2^{\binom{n}{2}}.$$

   The nice thing about this example is that it builds on two of the basic facts listed at the beginning of this note.

Recursions: We also did a couple of things that illustrate the value of counting through recursions. You will do one such question in the homework.

2.1 A refresher of sample spaces and set operations

The first step in analyzing a probabilistic outcome is to have a clear understanding of **all** the possible outcomes even if they have a small probability.

We use $\mathcal{S}$ for the *sample space* – the space of all possible outcomes. An item ω in the sample space is called a *sample point*. So in the toss of a die, the sample space has the 6 sample points $1, 2, \ldots, 6$.

An *event* is a subset of the sample space. In this simple example, $\{1\}, \{1, 2\}, \{2, 4, 6\}$ are all events (the last one is the event that the result of the die is even).

Below are some formal definitions of union and intersection (but really, these are very intuitive notions).

To make definitions easier, let us use the letter ω for a sample point.

If you have two events A and B, the union and intersection are defined by

- $A \cup B = \{w \text{ in the sample space that are either in } A \text{ or } B\}$,
- $A \cap B = \{w \text{ in the sample space that are in } \textbf{both } A \textbf{ and } B\}$

Some basic rules:

- Order of operations does not matter $(A \cup B) \cup C = A \cup (B \cup C)$.
- You can disaggregate operations ("the distributive law"):

  $$(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$$

  and

  $$(A \cap B) \cup C = (A \cup C) \cap (B \cup C).$$

ORIE 5530: Modeling Under Uncertainty

Lecture 3 (Basic Probability)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

So we said an event is a subset of the sample space. In essence, an event is a collection of possible outcomes. We say that an event happened if one of the outcomes in it happened. If $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, then $A = \{2, 4, 6\}$ corresponds to the event that we got an even outcome – one of the numbers $2, 4, 6$ was the outcome of the experiment.

So you could be slightly formal and define Ω to be the collection of all subsets of the sample space $\mathcal{S}$. For example if $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, Ω includes individual-outcome events like $\{1\}$ (this is the event the outcome was 1 on the die), it includes events of the form $\{3, 5\}$ (that we got 3 or 5), etc.; and all possible combinations up to and including the event $\{1, 2, 3, 4, 5, 6\}$, which is the whole space.

3.1 Probability on events

Now, once you listed the possible events you should be able to assign probabilities to them. There is a probability 1 that I get some number in the set $[6] := \{1, \ldots, 6\}$ when I roll the dice, so $\mathbb{P}(\mathcal{S}) = 1$. The event $A = \{2, 4, 6\}$ = "even outcome" has probability $1/2$ (if the die is fair, etc.).

It also is clear that a probability (likelihood) will always be a real number between 0 and 1.

Finally, it is intuitive to us (and correctly so) that the probability of $\{2, 4, 6\}$ in the roll of a die is the sum of the probabilities of getting 2, 4 and 6. This is because these are exclusive events (if we get 2 on the roll of a dice, that's it. We did not get a 4 or a 6).

So we want to be able to say that if two events are exclusive (meaning they cannot happen together – A and B are exclusive means $A \bigcap B = \emptyset$) then the probability that one (or both) of them happens should be the sum of the probabilities.

In summary, a probability is a function from Ω to $[0, 1]$ that has the properties:

i. $0 \leq \mathbb{P}(A) \leq 1$ for any event A;

ii. $\mathbb{P}(\mathcal{S}) = 1$; and

iii. For any collection of mutually exclusive events $A_1, A_2, \ldots, A_n$ (i.e., $A_i \bigcap A_j = \emptyset$, $\forall i, j$),

$$\mathbb{P}(\cup_{i=1}^{n} A_i) = \sum_{i=1}^{n} \mathbb{P}(A_i).$$

An event A and its complement A^c ("not A") are obviously exclusive and their union is $\mathcal{S}$ so that $\mathbb{P}(A) + \mathbb{P}(A^c) = 1$ (hence, $\mathbb{P}(A) = 1 - \mathbb{P}(A^c)$).

If two events A and B are not exclusive then

$$\mathbb{P}(A \bigcup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \bigcap B)$$

so that we always have the inequality

$$\mathbb{P}(A \bigcup B) \leq \mathbb{P}(A) + \mathbb{P}(B).$$

3.2 Conditional probability on events

The basic identity here is that (we use | for conditioning)

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(B \bigcap A)}{\mathbb{P}(A)}. \qquad (*)$$

Notice that this makes sense only if $\mathbb{P}(A) > 0$. (This make intuitive sense – If A is an event that has 0 probability, it will never happen so we will never care about what is the likelihood of B given A). Observe that $\mathbb{P}(A|B) \neq \mathbb{P}(B|A)$ unless $\mathbb{P}(A) = \mathbb{P}(B)$.

Now, from here follows a characterization of what it means for two events to be independent. Conceptually, you want to say that B and A are independent if knowing that A happened does not change the likelihood I assign to B. That is,

$$\mathbb{P}(B|A) = \mathbb{P}(B).$$

If you plug this to the identity above (replace the left hand side $\mathbb{P}(B|A)$ with $\mathbb{P}(B)$) you get
$\mathbb{P}(B) = \frac{\mathbb{P}(B \bigcap A)}{\mathbb{P}(A)}$ or, as we mathematically think of independence, we have

$$\mathbb{P}(B \bigcap A) = \mathbb{P}(B)\mathbb{P}(A). \qquad \text{(independence)}$$

If we have multiple events, say, $A_1, \ldots, A_m$, we say that they are all independent of each other if any sub-collection satisfies the above. That is, we definitely need that $\mathbb{P}(A_i \bigcap A_j) = \mathbb{P}(A_i)\mathbb{P}(A_j)$ for any $i \neq j$, but we also need $\mathbb{P}(A_i \bigcap A_j \bigcap A_l) = \mathbb{P}(A_i)\mathbb{P}(A_j)\mathbb{P}(A_l)$ for any $i \neq j \neq l$, etc. In general, for any n and any sub-collection $A_{i_1}, \ldots, A_{i_n}$ of size n with $i_1 \neq i_2 \neq \ldots \neq i_n$, we need that

$$\mathbb{P}(\bigcap_{k=1}^{n} A_{i_k}) = \mathbb{P}(A_{i_1})\mathbb{P}(A_{i_2}) \ldots \mathbb{P}(A_{i_{n-1}})\mathbb{P}(A_{i_n}).$$

Useful fact: If you take a partition of the world into B and B^c (not B), it is obvoius you can write a set A as the union of its intersection with B (what is in B and in A) and an intersection with B^c (what is in A but not in B). That is,

$$A = (A \bigcap B) \bigcup (A \bigcap B^c).$$

Because B and B^c are exclusive so are $(A \bigcap B)$ and $(A \bigcap B^c)$ and we have by the basic rule (iii) of probability that

$$\mathbb{P}(A) = \mathbb{P}(A \bigcap B) + \mathbb{P}(A \bigcap B^c).$$

Using (*) we can re-write this as

$$\mathbb{P}(A) = \mathbb{P}(A|B)\mathbb{P}(B) + \mathbb{P}(A|B^c)\mathbb{P}(B^c). \qquad (**)$$

More generally if $B_1, \ldots, B_m$ is some way to divide the world of outcomes (meaning it is a *partition* of $\mathcal{S}$: $B_i \bigcap B_j = \emptyset$ for all $i \neq j$ and $\bigcup_{i=1}^{m} B_i = \mathcal{S}$) then

$$\mathbb{P}(A) = \sum_{i=1}^{n} \mathbb{P}(A \bigcap B_i).$$

We used this fact in an example we had in class about the conditional likelihood of two boys.

Notice that because of the basic rule (*) we can re-write this as

$$\mathbb{P}(A) = \sum_{i=1}^{n} \mathbb{P}(A|B_i)\mathbb{P}(B_i).$$

Replacing $\mathbb{P}(B \bigcap A) = \mathbb{P}(A|B)\mathbb{P}(B)$ and replacing $\mathbb{P}(A)$ in the denominator using (**) we get the simplest version of Bayes rule:

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B)\mathbb{P}(B)}{\mathbb{P}(A|B)\mathbb{P}(B) + \mathbb{P}(A|B^c)\mathbb{P}(B^c)}.$$

The beauty of Bayes rule is in how it reverses the conditioning. We are interested in B conditional on A and we evaluate this by conditioning A and B and B^c.

More generally, if $B_1, \ldots, B_n$ are a partition of the set $\mathcal{S}$, then for each $j = 1, \ldots, n$ we have

$$\mathbb{P}(B_j|A) = \frac{\mathbb{P}(A|B_j)\mathbb{P}(B_j)}{\sum_{i=1}^{n} \mathbb{P}(A|B_i)\mathbb{P}(B_i)}.$$

ORIE 5530: Modeling Under Uncertainty

Lecture 4 (Random Variables 1)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

We move to random variables. We start with the world of discrete variables where things are kind of more intuitive but nevertheless most of the stuff can be covered. We will later move to continuous random variables. A random variable (RV) is *discrete* if it takes on a finite or countable number of possible values.

We are sometimes interested in a function of the uncertainty rather than in the uncertainty itself. For example, with the roll of two die, instead of being interested in the outcome pair, I might be interested in the sum of the two die. Then, I could define a random variable X for this sum and I will be able to compute the *probability mass function* (pmf) of X: $\mathbb{P}(X = k)$ for different values of k that the RV X can take on by aggregating the corresponding outcomes. So, for example,

$$\mathbb{P}(X = 5) = \mathbb{P}(\{(1, 4), (4, 1), (2, 3), (3, 2)\}) = \frac{4}{36}.$$

We can also compute the *distribution* of X which is $F(k) = \mathbb{P}(X \leq k)$ for each k. For example

$$\mathbb{P}(X \leq 5) = \sum_{k=2}^{5} \mathbb{P}(X = k).$$

This is often referred to as the *cumulative distribution function* (CDF). Of course, different random variables have different cumulative distribution functions. The distribution function captures all the information we need for a random variable. For example, the probability that a random variable X has values between x and y is given by

$$\mathbb{P}(x < X \leq y) = F(y) - F(x).$$

We could also recover the probability mass function by noting, in our example above, that

$$\mathbb{P}(X = k) = \mathbb{P}(k - 1 < X \leq k).$$

A legitimate distribution always has the property that the probability of being smaller than infinity (or greater than $-\infty$) is 1:

$$\lim_{y \to \infty} \mathbb{P}\left(X \leq y\right) = 1 \text{ and } \lim_{y \to -\infty} \mathbb{P}\left(X \leq y\right) = 0.$$

This is nothing mind blowing. It just means that ∞ or $-\infty$ are not an option, but also means that when you sum up the probabilities of all possible outcomes they must sum to 1: for a random variable X taking values $x_1, x_2, \ldots$,

$$\sum_k \mathbb{P}(X = x_k) = 1.$$

4.1 Some Important Distributions

Later we will also cover expectation and variance. In the list below I include those properties. I will use the common abbreviation w.p. for "with probability".

1. **Bernoulli:** $X \sim Bernoulli(p)$ is a RV where

$$X = \begin{cases} 1 & w.p. \;\; p \\ 0 & w.p. \;\; 1 - p \end{cases}$$

Hence, we have $\mathbb{P}(X=1)=p$ and $\mathbb{P}(X=0)=1-p$.
Mean and Variance: $\mathbb{E}[X]=p, \mathbb{V}\text{ar}(X)=p(1-p)$.

2. **Geometric:** $X \sim Geom(p)$ is the number of independent trials until a success, where p is the probability of success in each trial.

$$\mathbb{P}(X=1)=p, \mathbb{P}(X=2)=(1-p)\,p, \cdots, \mathbb{P}(X=k)=(1-p)^{k-1}\,p.$$

Mean and Variance: $\mathbb{E}[X]=1/p, \mathbb{V}\text{ar}(X)=(1-p)/p^2$.

3. **Binomial:** $X \sim Bin(n,p)$ is the number of successes in n independent trials, each having success probability p. $\mathbb{P}(X=k)$ is the probability of seeing k successes out of n Bernoulli trials.

$$\mathbb{P}(X=k)=\binom{n}{k}p^k\,(1-p)^{n-k}.$$

Mean and Variance: $\mathbb{E}[X]=np, \mathbb{V}\text{ar}(X)=np(1-p)$.

4. **Poisson:** $X \sim Poisson\,(\lambda)$ means

$$\mathbb{P}(X=k)=\frac{e^{-\lambda}\lambda^k}{k!} \text{ for } k \in \mathbb{Z}^+.$$

Mean and Variance: $\mathbb{E}[X]=\mathbb{V}\text{ar}(X)=\lambda$.

The binomial and Poisson distributions are closely related through the following result.

Poisson approximation to the binomial: Let $X_{n,p} \sim Bin(n,p)$. Consider a sequence of these random variables such that n is increasing along the sequence but p is decreasing so as to keep $np=\lambda$ constant. Then,

$$\mathbb{P}(X_{n,p}=k) \to \frac{e^{-\lambda}\lambda^k}{k!}.$$

The right hand side is the $Poisson(\lambda)$ distribution. What do we practically do with such a limit result? It says that *with a large number of independent experiments n, each with a small probability of success p*, it is reasonable to approximate the likelihood of k successes by $\mathbb{P}(Y=k)$ where $Y \sim Poisson(np)$.

4.2 Expectation

Definition 1 *Let X be a discrete r.v. taking on values $x_1, x_2, \ldots$. The expectation of X is defined as*

$$\mathbb{E}[X]=\sum_{k=1}^{\infty} x_k \mathbb{P}(X=x_k).$$

For a Binomial r.v, we have

$$\mathbb{E}[X]=\sum_{k=0}^{n} k\binom{n}{k}p^k\,(1-p)^{n-k}=np.$$

For a Geometric r.v, we have

$$\mathbb{E}[X]=\sum_{k=1}^{n} k\,(1-p)^{k-1}\,p=\frac{1}{p}.$$

Expectation of a function of a random variable: If X takes on values $x_1, x_2, \ldots$ and $g(X)$ is a function of the random variable (in our single-order, multi-period inventory example, D was the random variable

representing demand, Q was the inventory level (or quantity of units stocked), and we were interested in part in $g(D) = \max(Q - D, 0)$), it then obtains values $g(x_1), g(x_2), \ldots$ with respective probabilities $\mathbb{P}(X = x_1), \mathbb{P}(X = x_2), \ldots$. Thus,

$$\mathbb{E}[g(X)] = \sum_{k=1}^{\infty} g(x_k)\mathbb{P}(X = x_k).$$

We had examples in class of the application of expectations: For example, the single-order, multi-period inventory model, a.k.a. the newsvendor model (example for function of RVs and optimization).

The *variance* of a random variable X is also just the expectation of a function of the RV. Specifically, let $g(X) = (X - \mathbb{E}[X])^2$. Then,

$$\mathbb{V}\text{ar}(X) = \mathbb{E}[g(X)] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2.$$

(Convince yourself that the last inequality is correct by opening up the quadratic).

4.3 Joint distribution, independence and conditional probabilities for Discrete RVs

When we have two (or more) random variables, we can talk about their joint distribution. Consider two RVs X and Y taking values in $\mathcal{X} = \{x_1, \ldots, x_n, \ldots\}$ and $\mathcal{Y} = \{y_1, \ldots, y_n, \ldots\}$, respectively. Then, we can talk about the intersection of the events $A_i = \{X = x_i\}$ and $B_j = \{Y = y_j\}$. Recall that the two events would be independent if

$$\mathbb{P}(A_i \bigcap B_j) = \mathbb{P}(A_i)\mathbb{P}(B_j).$$

We would say that the random variables X and Y are independent if all events A_i and B_j are independent. That is, if

$$\mathbb{P}(\{X = x_i\} \bigcap \{Y = y_j\}) = \mathbb{P}(X = x_i)\mathbb{P}(Y = y_j), \text{ for all } x_i, y_j.$$

To simplify notation we write $\mathbb{P}\{X = x_i, Y = y_j\}$ instead of $\mathbb{P}\{\{X = x_i\} \bigcap \{Y = y_j\}\}$. Then, the collection of joint probabilities

$$\mathbb{P}\{X = x_i, Y = y_j\}, \quad \text{for all } x_i, y_j$$

is the *joint distribution* of (X, Y).

Notice that from the joint distribution we can recover the *marginal distribution*, meaning that of X or Y. Indeed, this is again like with events and the connection might be useful. Before we had that if $B_1, \ldots, B_n$ is a partition, then we can write

$$\mathbb{P}(A) = \sum_i \mathbb{P}(A \bigcap B_i).$$

The same holds here. Consider the event $A = \{X = x\}$ and let $B_j = \{Y = y_j\}$. Then,

$$\mathbb{P}(X = x) = \sum_j \mathbb{P}(X = x, Y = y_j).$$

Now (again similar to events) we can ask what is the likelihood of $X = x_i$ given that $Y = y_j$:

$$\mathbb{P}\left(X = x_i | Y = y_j\right) = \frac{\mathbb{P}\left(X = x_i, Y = y_j\right)}{\mathbb{P}\left(Y = y_j\right)}.$$

This follows from the identity for events $\mathbb{P}(A|B) = \frac{\mathbb{P}(A \bigcap B)}{\mathbb{P}(B)}$. Independence then implies that $\mathbb{P}\left(X = x_i | Y = y_j\right) = \mathbb{P}\left(X = x_i\right)$ for all x_i and y_j.

Just like we compute expectation from probability, we can compute conditional expectations from conditional probabilities:

$$\mathbb{E}[X|Y = y] = \sum_x x\mathbb{P}\{X = x|Y = y\}.$$

Notice that we can remove the conditions by multiplying then by the probability of Y. Meaning, we have the *de-conditioning*

$$\mathbb{P}\{X = x\} = \sum_y \mathbb{P}\{X = x|Y = y\}\mathbb{P}\{Y = y\},$$

and

$$\mathbb{E}[X] = \sum_y \mathbb{E}[X|Y = y]\mathbb{P}\{Y = y\}.$$

The following example illustrates how cleverly using conditional expectation (and de-conditioning) can help answer some non-trivial questions.

Example 1 Toss a coin with probability p of getting Heads (H). How many tosses, in expectation, do you need until you get k consecutive H? Let us define two things:

$N_k = \#$ of trials until k consecutive H, and

$M_k = \mathbb{E}[N_k]$.

Our question concerns what is the value of M_k? It is not trivial to determine how to answer this, but it seems we should be able to build a recursion. Specifically, if we know that it took n tosses until we saw a sequence of H of length $k-1$, then it should take at least $n+1$ until we see one of k but possibly more; that is, if the $(n+1)^{st}$ toss is a tail (which happens with probability $1-p$) then we have to re-start all over again so that we have N_k to add. More formally. we have that

$$\mathbb{E}[N_k|N_{k-1} = n] = n + 1 + (1-p)\mathbb{E}[N_k].$$

Upon de-conditioning, we obtain

$$\begin{aligned}
\mathbb{E}[N_k] &= \sum_n \mathbb{E}[N_k|N_{k-1} = n]\mathbb{P}\{N_{k-1} = n\} \\
&= \sum_n (n + 1 + (1-p)\mathbb{E}[N_k])\mathbb{P}\{N_{k-1} = n\} \\
&= \sum_n n\mathbb{P}\{N_{k-1} = n\} + \sum_n (1 + (1-p)\mathbb{E}[N_k])\mathbb{P}\{N_{k-1} = n\} \\
&= \mathbb{E}[N_{k-1}] + (1 + (1-p)\mathbb{E}[N_k]).
\end{aligned}$$

In the last line I just used the fact that $\sum_n \mathbb{P}\{N_{k-1} = n\} = 1$. So we have arrived at the recursion $M_k = M_{k-1} + 1 + (1-p)M_k$, which we can write as

$$M_k = \frac{1 + M_{k-1}}{p}.$$

Now let us apply this recursion with the initial condition $M_0 = 0$. Then, we have $M_1 = \frac{1}{p}$, $M_2 = \frac{(1+1/p)}{p} = \frac{1}{p} + \frac{1}{p^2}$, and so on to get the result for general k:

$$M_k = \frac{1}{p} + \frac{1}{p^2} + \ldots \frac{1}{p^k}.$$

■

Another useful fact is that, if X and Y are independent, then

$$\mathbb{E}[XY] = \mathbb{E}[X]\,\mathbb{E}[Y].$$

This is because

$$\begin{aligned}\mathbb{E}[XY] =& \sum_{(i,j)} x_i y_j \times \mathbb{P}(X = x_i, Y = y_j) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} x_i y_j \times \mathbb{P}(X = x_i, Y = y_j) \\ =& \sum_{i=0}^{\infty} x_i \sum_{j=0}^{\infty} y_j \mathbb{P}(X = x_i)\,\mathbb{P}(Y = y_j) = \sum_{i=0}^{\infty} x_i \mathbb{P}(X = x_i) \sum_{j=0}^{\infty} y_j \mathbb{P}(Y = y_j) \\ =& \mathbb{E}[X]\,\mathbb{E}[Y].\end{aligned}$$

Definition 1 *The covariance between two random variables X and Y is defined as*

$$\mathbb{C}ov(X,Y) = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y] = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])].$$

Covariance is not always an informative quantity because it does not allow for comparisons of relationship. Say you have a quantity Y that you are trying to predict and two dependent variables X_1, X_2, and suppose that $\mathbb{C}\text{ov}(Y, X_1) > 0$ and $X_2 = 10 \cdot X_1$. Then,

$$\mathbb{C}\text{ov}(Y, X_2) = \mathbb{C}\text{ov}(Y, 10 \cdot X_1) = 10 \cdot \mathbb{C}\text{ov}(Y, X_1).$$

Do you want to conclude that X_2 is a much better predictor of Y. Probably not. After all they are the same variable just scaled. Correlation corrects for this scale effect and allows you to compare normalized relationship.

Definition 2 *The correlation between two random variables X and Y is defined as*

$$\rho(X,Y) = \frac{\mathbb{C}ov(X,Y)}{\sqrt{\mathbb{V}ar(X)}\sqrt{\mathbb{V}ar(Y)}}.$$

Notice (check) that ρ is scale insensitive. Meaning that for $a, b > 0$

$$\rho(aX, bY) = \rho(X, Y).$$

So, going back to the example above, $\rho(Y, X_2) = \rho(Y, X_1)$ captures the fact that X_1 and X_2 would equally be good predictors of Y.

Further observe that if X and Y are independent, then $\mathbb{C}\text{ov}(X, Y) = 0$ and $\rho(X, Y) = 0$.

4.4 Some useful inequalities

The beauty of the inequalities below is that the more information you use the better bounds you get. These bounds can be extremely useful when the underlying structure of the problem is too complicated for us to actually derive the distribution, yet we can derive certain quantities like the expectation or standard deviation. Below are some basic examples we covered, followed by a more elaborate example. I will use the common abbreviation i.i.d. for "independent and identically distributed".

Theorem 1 (Markov inequality) *For a positive RV X,*

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}[X]}{a} \text{ for any } a > 0.$$

Proof: Note that

$$\mathbb{E}[X] = \sum_{k=1}^{\infty} k\mathbb{P}(X = k) = \sum_{k=1}^{a-1} k\mathbb{P}(X = k) + \sum_{k=a}^{\infty} k\mathbb{P}(X = k)$$
$$\geq \sum_{k=a}^{\infty} k\mathbb{P}(X = k) \geq a \sum_{k=a}^{\infty} \mathbb{P}(X = k) = a\mathbb{P}(X \geq k).$$

Rearranging the terms, we obtain the desired inequality. ■

Example 2 *Let $X \sim Bin(n, p)$. We have $\mathbb{P}(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$ and $\mathbb{E}[X] = np$. Suppose we have $n = 200$ and $p = 0.1$. Then, $\mathbb{E}[X] = np$ and, by the Markov inequality,*

$$\mathbb{P}(X \geq 120) \leq \frac{\mathbb{E}[X]}{120} = \frac{1}{6}.$$

Theorem 2 (Chebyshev Inequality) *If X is a RV with variance $\mathbb{V}ar(X)$, then*

$$\mathbb{P}(|X - \mathbb{E}[X]| \geq a) \leq \frac{\mathbb{V}ar(X)}{a^2}.$$

Proof: Since $(X - \mathbb{E}[X])^2$ is a nonnegative RV, applying Markov's inequality with $a = k^2$ renders $\mathbb{P}[(X - \mathbb{E}[X])^2 \geq k^2] \leq \frac{\mathbb{E}[(X-\mathbb{E}[X])^2]}{k^2}$. But since $(X - \mathbb{E}[X])^2 \geq k^2$ if and only if $|(X - \mathbb{E}[X])| \geq k$, this is equivalent to

$$\mathbb{P}[|X - \mathbb{E}[X]| \geq k] \leq \frac{\mathbb{E}\left[(X - \mathbb{E}[X])^2\right]}{k^2} = \frac{\mathbb{V}\text{ar}[X]}{k^2}.$$

■

Example 3 *For a random variable $X \sim Bin(n, p)$, $\mathbb{V}ar(X) = np(1-p)$. Instead of the $\frac{1}{6}$ in the previous example, we get a sharper bound this time:*

$$\mathbb{P}(X \geq 120) = \mathbb{P}(X - 20 \geq 100) \leq \mathbb{P}(|X - 20| \geq 100) \leq \frac{\mathbb{V}ar(X)}{100^2} = \frac{18}{100^2} = 0.0018.$$

Theorem 3 (Chernoff Bound) *Let X be a RV.*

$$\mathbb{P}(X \geq a) = \mathbb{P}(g(X) \geq g(a)) \leq \frac{\mathbb{E}[g(X)]}{g(a)}.$$

Let $g(x) = e^x$. We have

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}\left[e^X\right]}{e^a},$$

which is the Chernoff bound.

Example 4 *Let $X \sim Bin(n, p)$. Then*

$$\mathbb{E}\left[e^X\right] = \mathbb{E}[e^{\sum_{i=1}^n Z_i}] = \prod_{i=1}^{n} \mathbb{E}\left[e^{Z_i}\right] = (pe + 1 - p)^n.$$

where the Z_i are i.i.d. Bernoulli(p).

By Chernoff's bound, for $n = 200$ and $p = 0.1$, we have

$$\mathbb{P}(X \geq 120) \leq \frac{(0.1 \times e + 0.9)^2 00}{e^{120}} \approx 1.4 \times 10^{-44},$$

which is much sharper than the previous 2 bounds.

Using Chernoff's to evaluate random algorithms (an example)

The original formulation, which I used in class, concerns a set of n nodes and m subsets of the set of nodes. But you can think of the nodes as individuals and the subsets as groups or teams of individuals, in addition to many other applications.

You have a collection of n nodes U and m subsets of U: $S_1, \ldots, S_m$. The subsets can have non-empty intersections (there might be nodes that belong to multiple subsets). Now suppose I want to "paint" each of the n nodes in one of two colors, say Blue or Red, while making sure that the number of blue and red nodes in each subset S_i is as balanced as possible.

Specifically, the imbalance in subset S_i is given by

$$Disc(S_i) = |\#redsinS_i - \#blueinS_i|,$$

and I am interested in minimizing the largest discrepancy in my set of nodes. That is, I want an algorithm to minimize

$$\max_i Disc(S_i).$$

A deterministic algorithm could be very complicated. Instead, we use a "stupid" random algorithm that paints a node blue with probability 1/2 and red with probability 1/2. The question we then want to ask is *how bad/good is this algorithm?*

Analyzing this algorithm exactly is very complicated because there can be overlap between subsets, in which case there is dependency, and the structure of overlap can be arbitrarily crazy. **Chernoff's bound comes to the rescue here!!!**

With the random algorithm the outcome is going to be, well, random. Let us say I want to get a number so that I can claim with a certainty of $1 - 1/m$ that the max discrepancy $\max_i Disc(S_i)$ is below this number. In other words, find a number $K = 2a$, so that you can guarantee that, under the randomized algorithm,

$$\mathbb{P}\{\max_i Disc(S_i) > 2a\} \leq \frac{1}{m}.$$

What is this number m? I claim that $K = 2a = \sqrt{12n \log m}$ is this number. Namely, the following is true:

$$\mathbb{P}\{\max_i Disc(S_i) > \sqrt{12n \log m}\} \leq \frac{1}{m}.$$

What follows is a proof of this claim.

If $X \sim Bin(n, p)$, then $X = \sum_i Z_i$ where Z_i are i.i.d. Bernoulli RVs, and we apply the Chernoff bound to obtain

$$\mathbb{P}(X \geq a) \leq \frac{(pe + (1-p))^n}{e^a}.$$

It is useful, instead, to think of deviations from the mean: $|X - \mathbb{E}[X]|$. Using similar ideas we can derive the bound

$$\mathbb{P}(|X - \mathbb{E}[X]| \geq \delta \mathbb{E}[X]) \leq e^{\frac{-\delta^2 \mathbb{E}[X]}{3}}.$$

Let us focus on a single subset, say S_1, and suppose k is the number of nodes in S_1. Notice that (since $\#red = k - \#blue$), we have

$$Disc(S_1) = |\#red - \#blue| = 2|\#red - k/2|.$$

Thus, $Disc(S_1) \geq 2a$ if and only if $|\#red - k/2| \geq a$.

Let us define the random variable R_1 to be the number of red nodes in subset 1. This is a $Bin(k, 1/2)$ random varaible and $\mathbb{E}[R] = k/2$. Then,

$$\mathbb{P}(Disc(S_1) \geq 2a) = \mathbb{P}(|R_1 - k/2| \geq a) = \mathbb{P}(|R - \mathbb{E}[R]| \geq a) \leq e^{-\frac{a^2}{3\mathbb{E}[R]}} = e^{-\frac{2a^2}{3k}} \leq e^{-\frac{2a^2}{3n}}.$$

In the first inequality we used the Chernoff bound with $\delta = (a)/\mathbb{E}[R]$; in the second to last equality we used $\mathbb{E}[R] = k/2$; and in the last inequality we used the fact that $k \leq n$. We can repeat the same arguments for any subset. Thus, we have established that, for any subset i,

$$\mathbb{P}(Disc(S_i) \geq 2a) \leq e^{-\frac{2a^2}{3n}}.$$

Finally, notice that the event $\{\max_i Disc(S_i) \geq 2a\}$ is the union of the events $\{Disc(S_i) \geq 2a\}$ for $i = 1, \ldots, m$. This is a good point to recall as a useful fact. Next recall that, if you take the union of two events A and B, then $\mathbb{P}(A \bigcup B) \leq \mathbb{P}(A) + \mathbb{P}(B)$; this also applies to muliple events. So we have

$$\mathbb{P}(\max_i Disc(S_i) > 2a) \leq \sum_{i=1}^{m} \mathbb{P}(Disc(S_i) > 2a) \leq m \max_i \mathbb{P}(Disc(S_i) > 2a) \leq me^{-\frac{2a^2}{3n}}.$$

Now choose $a = \sqrt{3n \log m}$ (so that $2a = \sqrt{12n \log m}$). Then, you can verify that $e^{-\frac{2a^2}{3n}} = \frac{1}{m^2}$ and we have the desired bound.

4.5 Introduction to Decision Trees

The example covered in class is provided in the file Decision_Tree_Example.pdf. There are some minor differences in notation and terms as follows. The decisions at hand are whether to build a small-sized complex (denoted B_S in class and d_1 in the handout), a medium-sized complex (denoted B_M in class and d_2 in the handout), and a large-sized complex (denoted B_L in class and d_3 in the handout). The sources of uncertainty (called "states of nature" in the handout) are with respect to demand, where two (discrete) possibilities were identified for the model: strong demand (denoted D_S in class and s_1 in the handout); and weak demand (denoted D_W in class and s_2 in the handout). Table 4.1 provides the estimated rewards (profits) for the various possible model outcomes.

Decision	Strong Demand (D_S)	Weak Demand (D_W)
B_S	8	7
B_M	14	5
B_L	20	-9

Table 4.1: Expected Rewards for Decision Tree Example.

For the application of Bayes rule, you can recover the original formulation in terms of $\mathbb{P}[D_S|F]$, $\mathbb{P}[D_S|U]$, $\mathbb{P}[D_W|F]$, $\mathbb{P}[D_W|U]$ upon being given $\mathbb{P}[F|D_S] = 0.9$, $\mathbb{P}[U|D_S] = 0.1$, $\mathbb{P}[F|D_W] = 0.25$, $\mathbb{P}[U|D_W] = 0.75$ and recalling $\mathbb{P}[F] = 0.77$ and $\mathbb{P}[U] = 0.23$. (Convince yourself that this is indeed the case).

ORIE 5530: Modeling Under Uncertainty

Lecture 5 (Simulation and Continous Random Variables)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

5.1 Simulation of discrete random variables

In class we started from Bernoulli random variables and (quickly) built our way up. I just want to provide here the final algorithm. The premise of simulation is that you have a code command that generates a $U[0,1]$ (that is uniform on $[0,1]$) random variable. We want to generate a sample of size n from the distribution of a discrete random variable X. That is, we want to generate a sequence $X_1, \ldots, X_n$ but we can only generate a sequence $U_1, \ldots, U_n$ from a uniform distribution.

Here is the general algorithm: Suppose you have a random variable taking values $x_1, \ldots, x_n$. Let $x_0 = -\infty$. Notice that the distribution is given by $p_k = F(x_k) = \sum_{i=1}^{k} \mathbb{P}(X = x_k)$.

Then, do the following.

Algorithm 1 *For $i = 1, \ldots, n$:*

1. *Generate a $U[0,1]$ random variable U_i (independent of all previous RVs)*
2. *Find k such that $p_{k-1} < U_i \leq p_k$. Set $Y_i = x_k$.*

Why does this do the right thing? In other words, if I generated Y_i this way, is the distibution of Y_i the same as that of X? The answer is yes because:

$$\mathbb{P}(Y_i = x_k) = \mathbb{P}(p_{k-1} < U_i \leq p_k) = p_k - p_{k-1} = F(x_k) - F(x_{k-1}) = \mathbb{P}(X = x_k).$$

Example 1 *With the binomial distribution, the possible outcomes are $x_1 = 0, x_2 = 1, \ldots, x_{n+1} = n$. Further, $\mathbb{P}(X = i) = \binom{n}{i} p^i (1-p)^{n-i}$ and $p_k = \sum_{i=0}^{k-1} \mathbb{P}(X = i)$.*

5.2 Continuous random variables

The presentation of continuous random variables is aimed at relating the discrete to the continuous by showing that, at an intuitive level, whatever we did with the probability mass function (pmf) of a discrete random variable, we can translate to a continuous random variable by replacing the pmf with the probability density function (pdf).

Definition 1 *We say that X is a continuous random variable if there exists a function $f_X(x)$ such that for any interval $[a,b]$ of the real line*

$$\mathbb{P}(a < x \leq b) = \int_a^b f_X(x) dx.$$

We call $f_X(\cdot)$ the probability density function of X.

It may be useful to go back to thinking about sample spaces and events. The sample space of a normal random variable is $S = \{x : -\infty < x < \infty\}$ (namely, the real line). What are events of interest here? The "important" events are intervals of the real line. Namely, we have events of the form $A = [a, b]$ and their probability is determined by the normal density. That is, $\mathbb{P}(A) = \mathbb{P}(a < X \leq b)$ which can be computed as in the definition.

The easiest way to think about density is via the distribution. We define (as before) the distribution (CDF) of a random variable X as

$$F_X(x) = \mathbb{P}(X \leq x).$$

The density is how the distribution grows with x; i.e., it is the derivative:

$$F_X(x) = \int_{-\infty}^{x} f_X(y)dy \Rightarrow f_X(x) = F'_X(x).$$

In some way then f_X captures the local behavior. Remember your basic Taylor theorem? If you have a differentiable function g then $g(y) \approx g(x) + g'(x)(y - x)$ when y is sufficiently close to x. In this way $F_X(x + \epsilon) = F_X(x) + f_X(x)\epsilon$, or equivalently,

$$F_X(x + \epsilon) - F_X(x) = \mathbb{P}(x < X \leq x + \epsilon) \approx f_X(x)\epsilon.$$

Expectations: In the discrete case we had, for a random variable taking values in $\{x_1, \ldots, x_n\}$, $\mathbb{E}[X] = \sum_{k=1}^{n} x_k \mathbb{P}\{X = x_k\}$. Similarly here (with the pmf replaced by the pdf), we have

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} y f_X(y)dy.$$

In general, for a function g,

$$\mathbb{E}[g(X)] = \int_{-\infty}^{\infty} g(y) f_X(y)dy.$$

Taking this function to be $g(X) = X^2$ we can compute the variance

$$\mathbb{V}\text{ar}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 = \int_{-\infty}^{\infty} y^2 f_X(y)dy + \left(\int_{-\infty}^{\infty} y f_X(y)dy\right)^2.$$

Some important random variables

- *Uniform:* $X \sim U[a, b]$
  - Density: $f_X(x) = \frac{1}{b-a}$
  - Distribution: $F_X(x) = \frac{x-a}{b-a}$ for $x \in [a, b]$
  - Expectation: $\mathbb{E}[X] = \frac{b-a}{2}$
  - Variance: $\mathbb{V}\text{ar}(X) = \frac{1}{12}(b - a)^2$
- *Exponential:* $X \sim exp(\lambda)$
  - Density: $f_X(x) = \lambda e^{-\lambda x}$
  - Distribution: $F_X(x) = \int_0^x f_X(y)dy = 1 - e^{-\lambda x}$ for $x \geq 0$
  - Expectation: $\mathbb{E}[X] = \frac{1}{\lambda}$
  - Variance: $\mathbb{V}\text{ar}(X) = \frac{1}{\lambda^2}$
- *Normal:* $X \sim N(\mu, \sigma^2)$
  - Density: $f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

  - Distribution: $F_X(x) = \int_0^x f_X(y)dy$ (there is no closed form)
  - Expectation: $\mathbb{E}[X] = \mu$
  - Variance: $\mathbb{V}\text{ar}(X) = \sigma^2$

- *Log-Normal:* $X \sim Log\text{-}Normal(\mu, \sigma^2)$
  - Density: $f_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(log(x)-\mu)^2}{2\sigma^2}}$
  - Distribution: $F_X(x) = \int_0^x f_X(y)dy$ (there is no closed form)
  - Expectation: $\mathbb{E}[X] = e^{\mu+\frac{1}{2}\sigma^2}$
  - Variance: $\mathbb{V}\text{ar}(X) = e^{2\mu+\sigma^2}(e^{\sigma^2} - 1)$

The important thing about log-normal is that if X is $log\text{-}normal(\mu, \sigma^2)$ then $Y = log(X) \sim N(\mu, \sigma^2)$.

The most important case is the *standard normal* $X \sim N(0, 1)$ (that is, mean of 0 and variance of 1). Why is that? Because any other normal can be constructed from it. Specifically, if $X \sim N(\mu, \sigma^2)$ then $Y = aX + b \sim N(a\mu + b, a^2\sigma^2)$. To make sure we get the variance right, notice that by linearity of expectation $\mathbb{E}[aX + b] = a\mathbb{E}[X] + b$ and by laws of variance

$$\mathbb{V}\text{ar}(aX + b) = \mathbb{V}\text{ar}(aX) = a^2\mathbb{V}\text{ar}(X).$$

So, if you have $X \sim N(0, 1)$, then $Y = \mu + \sigma X \sim N(\mu, \sigma^2)$. Vice versa, if $Y \sim N(\mu, \sigma^2)$ then $X = (Y - \mu)/\sigma \sim N(0, 1)$.

Further note that, if X_1 and X_2 are two independent normal random variables, with means μ_1 and μ_2 and variances σ_1^2 and σ_2^2, then their sum $X_1 + X_2$ will also be normally distributed, with mean $\mu_1 + \mu_2$ and variance $\sigma_1^2 + \sigma_2^2$.

5.3 Joint distributions

Oftentimes you have random variables that are dependent. With dependent events, to be able to compute conditional probabilities, for example, we had

$$\mathbb{P}\{A|B\} = \frac{\mathbb{P}\{A \bigcap B\}}{\mathbb{P}\{B\}}.$$

So, letting D be the foot-traffic to a store and W be the weather (say temperatures tomorrow), if I wanted to compute the likelihood that demand exceeds 100 if the temperature is below 0 (say this is what the forecast says), then I want to compute

$$\mathbb{P}\{D > 100|W \leq 0\} = \frac{\mathbb{P}\{D > 100, W \leq 0\}}{\mathbb{P}\{W \leq 0\}}.$$

This means, I want to be able to compute things of the form

$$\mathbb{P}\{X \leq x, Y \leq y\}, \text{ for all } x, y.$$

This is the *joint distribution.*

If I have discrete random variables X and Y that take on values in $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$, respectively, then I can compute the joint distribution from the joint probabilities

$$\mathbb{P}\{X = x_l, Y = y_j\}, l = 1, \ldots, n; j = 1, \ldots, n, \tag{5.1}$$

because

$$\mathbb{P}\{X \leq x_k, Y \leq y_m\} = \sum_{l=1}^{k}\sum_{j=1}^{m} \mathbb{P}\{X = x_l, Y = y_j\}. \tag{5.2}$$

I can also recover the marginal distribution (i.e., of one RV) from the joint distribution. Returning to our Demand and Weather example

$$\mathbb{P}\{D = 100\} = \mathbb{P}\{D = 100, W < \infty\},$$

or similarly

$$\mathbb{P}\{D \leq 100\} = \mathbb{P}\{D \leq 100, W < \infty\}.$$

What in all of the above changes with continuous distributions? Not much really. Now we have a joint density $f_{X,Y}(x, y)$

Here, are the equivalents.

- The joint probabilities (5.1) are replaced by a joint density $f_{X,Y}(x, y)$ (see example of normal below).
- The joint distribution is then computed from the joint density in the same way that (5.2) is computed from (5.1) with integration instead of summation:

  $$F_{X,Y}(x, y) = \mathbb{P}\{X \leq x, Y \leq y\} = \int_{z=-\infty}^{x}\int_{h=-\infty}^{y} f_{X,Y}(z, h)dzdh.$$

  This also means that when you actually have the joint distribution, you can get the joint density by differentiating twice.

  $$f_{X,Y}(a, b) = \frac{\partial^2}{\partial x \partial y} F_{X,Y}(x, y)\Big|_{a,b}.$$

  Note: Do not be confused by this superscript X, Y. I just added it so that we remember that this is a joint distribution of the two variable X, Y.

- The marginal distribution can then be recovered by

  $$F_X(x) = \mathbb{P}\{X \leq x\} = F_{X,Y}(x, \infty).$$

A useful example is the normal distribution. Say (X, Y) is a two-variate normal with mean vector $\boldsymbol{\mu} = (\mathbb{E}[X], \mathbb{E}[Y]) = (\mu_X, \mu_Y)$, variance σ_X^2 and σ_Y^2, and correlation ρ. In this case, we have

$$f_{X,Y}(x, y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}}\, e^{-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_x)^2}{\sigma_X^2} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y}\right]}. \tag{5.3}$$

What can we do with these joint distributions? Well, sometimes we need them because reality has dependencies and correlation – we want to see the interaction between rain and demand or between the fact that a customer likes a pink shirt and the likelihood he/she will buy a foldable bike. We also want it because we can then (as in the beginning of these notes) build conditioning. **There will be in the third homework a couple of questions that will help you practice this.**

Independence: Two random variables are independent if $F_{X,Y}(x,y) = F_X(x)F_Y(y)$ for any values of x, y or in "words" if $\mathbb{P}\{X \leq x, Y \leq y\} = \mathbb{P}\{X \leq x\}\mathbb{P}\{Y \leq y\}$ for any x, y.

For the discrete case, this implies and is implied by the condition that $\mathbb{P}\{X = x_k, Y = y_l\} = \mathbb{P}\{X = x_k\}\mathbb{P}\{, Y = y_l\}$ for all possible values.

Similarly, for the continuous case, this implies and is implied by

$$f_{X,Y}(x,y) = f_X(x)f_Y(y)$$

(that is, the joint density is the product of the densities). Notice that this holds in the normal example above if you set the correlation to $\rho = 0$.

Bottom line: If you want to check independence it suffices that

- Discrete: $\mathbb{P}\{X = x_k, Y = y_l\} = \mathbb{P}\{X = x_k\}\mathbb{P}\{, Y = y_l\}$ for all possible values x_k, y_l.
- Continuous: $f_{X,Y}(x,y) = f_X(x)f_Y(y)$ for all possible x, y.

5.4 Simulation of continuous random variables

One idea of simulating continuous random variables is in fact very simple and has to do with inverting the distribution function. Specifically, say you want to create a family of independent random variables $X_1, \ldots, X_n$ that follow the exponential distribution $F(x) = 1 - e^{-\lambda x}$ and, as before, the only thing your computer knows how to do is generate uniform random variables.

The basic idea is as follows: if you take a family of independent $uniform[0,1]$ random variables $U_1, \ldots, U_n$, then $F^{-1}(U_1), \ldots, F^{-1}(U_n)$ follow the exponential distribution. (You can visualize this in excel: in column A you have a bunch of rand() commands and in column B you put the function $F^{-1}(A1)$ in the first row, $F^{-1}(A2)$ in the second row, etc.).

Why does it do the right thing? Put another way, why is it the case that $F^{-1}(U)$ is a random variable with the distribution $F(x)$?

This is because (recall $\mathbb{P}\{U \leq b\} = b$):

$$\mathbb{P}\{F^{-1}(U) \leq x\} = \mathbb{P}\{U \leq F(x)\} = F(x).$$

The only challenge with this is that, except for some simple cases, inverting also involves computational work. But, sometimes you can invert by hand as in the case of exponential:

$$y = F(x) = 1 - e^{-\lambda x} \Rightarrow x = -\frac{1}{\lambda}\log(1-y).$$

So we simulate a bunch of uniform random variables $U_1, \ldots, U_n$ and then the values we use are

$$-\frac{1}{\lambda}\log(1-U_1), \ldots, -\frac{1}{\lambda}\log(1-U_n).$$

In fact one can replace here $1 - U_i$ with U_i (convince yourself that if U is $Uniform[0,1]$ so is $1-U$).

Why the hell is this of any use? Well, say we want to simulate a queue where customer service times follow an exponential distribution. This will tell us how to create the service times.

A small remark: When I say that I am generating a draw from an exponential distribution, what does this mean? After all, I am just generating one number. The "meaning" here is the frequentist view just as the one we apply intuitively with a coin. In the case of a coin, "generating a draw" is just flipping the coin. When I say that this flip follows a Bernoulli distribution with $p = 1/2$, what I mean is that if I do this many times, I will get heads half of the time.

It is the same with the exponential distribution. If I simulate many independent exponential random

variables, I should have that

$$\%\text{ of draws } \leq x \approx \mathbb{P}\{X \leq x\} = 1 - e^{-\lambda x}.$$

Lastly, let us assume that we have code that generates **independent standard normal random variables**. How do we generate a vector of dependent normal random variables?

Definition 2 *A multivariate (d-dimensional) normal $X = (X_1, \ldots, X_d)$ with mean vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_d)$ and $d \times d$ covariance matrix Σ (where the entry Σ_{ij} is the covariance of X_i and X_j, and the entry $\Sigma_{ii} = \sigma_i^2$ is $\mathbb{V}ar(X_i)$) is a continuous RV with the density*

$$f_X(x) = \frac{1}{(\sqrt{2\pi})^d \sqrt{|\Sigma|}} e^{-\frac{1}{2}(x-\boldsymbol{\mu})^t \Sigma^{-1}(x-\boldsymbol{\mu})},$$

where $|\Sigma|$ is the determinant of the matrix Σ. We write $X \sim N(\boldsymbol{\mu}, \Sigma)$.

In the special bi-variate case (i.e., $d = 2$), the determinant of Σ equals $\sigma_1^2\sigma_2^2 - \rho^2\sigma_1^2\sigma_2^2 = (1-\rho^2)\sigma_1^2\sigma_2^2$ and the inverse of Σ is

$$\Sigma^{-1} = \frac{1}{|\Sigma|} \begin{bmatrix} \sigma_2^2 & -\rho\sigma_1\sigma_2 \\ -\rho\sigma_1\sigma_2 & \sigma_1^2 \end{bmatrix},$$

where ρ is the correlation and hence we obtain (5.3) from the above definition. Notice, in general, that it suffices if we are given the variances (hence standard deviations) and the correlation matrix $\rho = [\rho_{ij}]$ because then we have the covariance matrix $\Sigma_{ij} = \mathbb{C}\text{ov}(X_i, X_j) = \rho_{ij}\sigma_i\sigma_j$.

The basic idea for simulating correlated multivariate $X = (X_1, \ldots, X_d)$ from a vector of independent standard normals $N = (N_1, \ldots, N_d)$ is built on the multivariate version of the basic fact that, taking a standard normal N, $Y = aN + b$ is normal with mean b and variance a^2.

There is a bit of algebra involved below. To make our life easier as we move to the multidimensional case let us first assume $\boldsymbol{\mu} = 0$ (zero mean). Also, let us just look at the two-dimensional case first.

Let us consider a 2×2 matrix L (we will shortly say what this matrix L is) and define

$$X = LN = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} N_1 \\ N_2 \end{bmatrix} = \begin{array}{l} L_{11}N_1 + L_{12}N_2 \\ L_{21}N_1 + L_{22}N_2 \end{array}.$$

What is the covariance matrix of L? Using the independence of N_1 and N_2, we have
$\Sigma_{11} = \mathbb{V}\text{ar}(X_1) = L_{11}^2 + L_{12}^2$, $\Sigma_{22} = \mathbb{V}\text{ar}(X_2) = L_{21}^2 + L_{22}^2$ and

$$\begin{aligned} \Sigma_{12} = \Sigma_{21} &= \mathbb{C}\text{ov}(X_1, X_2) = \mathbb{C}\text{ov}(L_{11}N_1 + L_{12}N_2, L_{21}N_1 + L_{22}N_2) \\ &= L_{11}L_{21}\mathbb{C}\text{ov}(N_1, N_1) + L_{11}L_{22}\mathbb{C}\text{ov}(N_1, N_2) + L_{12}L_{21}\mathbb{C}\text{ov}(N_1, N_2) + L_{12}L_{22}\mathbb{C}\text{ov}(N_2, N_2) \\ &= L_{11}L_{21}\mathbb{V}\text{ar}(N_1) + L_{12}L_{22}\mathbb{V}\text{ar}(N_2) \\ &= L_{11}L_{21} + L_{12}L_{22}, \end{aligned}$$

where I use the rules of covariance:

- $\mathbb{C}\text{ov}(X, X) = \mathbb{V}\text{ar}(X)$;
- $\mathbb{C}\text{ov}(X, Y) = \mathbb{C}\text{ov}(Y, X)$;
- $\mathbb{C}\text{ov}(aX, bY) = ab\mathbb{C}\text{ov}(X, Y)$;
- $\mathbb{C}\text{ov}(X + Y, Z) = \mathbb{C}\text{ov}(X, Z) + \mathbb{C}\text{ov}(Y, Z)$;
- If X, Y are independent $\mathbb{C}\text{ov}(X, Y) = 0$.

So, if I am given a Σ, I would want to choose L such that

$$\Sigma = \begin{bmatrix} L_{11}^2 + L_{12}^2 & L_{11}L_{21} + L_{12}L_{22} \\ L_{11}L_{21} + L_{12}L_{22} & L_{21}^2 + L_{22}^2 \end{bmatrix} = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}^t = LL^t.$$

So, in the case of $\boldsymbol{\mu} = 0$, we have arrived at the following recipe to simulate a multivariate normal random variable with covariance matrix Σ.

1. Compute the matrix L such that $\Sigma = LL^t$.
2. Generate a standard normal vector $N_1, \ldots, N_d$.
3. Assign $X = LN$.

If you do this, $X = (X_1, \ldots, X_d)$ will follow the desired normal distribution. Having a non-zero mean $\boldsymbol{\mu}$ is now easy. Replace the last step with

3. Assign $X = \boldsymbol{\mu} + LN$.

A final word about implementation. How do we find the matrix L such that $\Sigma = LL^t$? Well, the matrix Σ is a special case of what is called a positive semi-definite matrix. It turns out that, for such matrices, you can always find a "square-root" matrix L as above. The procedure to do so is called Cholesky factorization and there is a command in Python's scipy.linalg library's that does exactly that. In that command you can choose whether you want to have L upper or lower triangular and you can always choose lower.

ORIE 5530: Modeling Under Uncertainty

Lecture 6 (Markov Chain Formalities)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

In this class we had formal definition of Markov chains, followed by a couple of elaborate examples detailed below. Estimation and simulation are also discussed below.

6.1 Markov chain formalities

We have a process $X_0, X_1, X_2, \ldots$ that takes a value at each time period. A simple example to think of is a weather example. Each day is rainy or sunny and X_n equals either R or S. The simplest model is that they are independent. That is, each day is rainy with some probability p_R and sunny with some probability p_S independent of the past. It is more reasonable that we have some sort of dependence of history. Markov chains allow for dependence on *immediate* history. Specifically, a sequence of random variables is a Markov chain if *the next state depends on history only through the current state*:

$$\mathbb{P}\{X_{n+1} = j | X_n = i_n, \ldots, X_0 = i_0\} = \mathbb{P}\{X_{n+1} = j | X_n = i_n\}. \qquad \text{(Markov property)}$$

It is a homogeneous Markov chain (and we will mostly be looking only at those) if the one step transition probability does not depend on the time index n. Namely,

$$\mathbb{P}\{X_{n+1} = j | X_n = i\} = \mathbb{P}\{X_1 = j | X_0 = i\} = p_{ij}. \qquad \text{(Homogeneity)}$$

We use p_{ij} for the probability of going *in one step* from i to j. It is very useful that these one step transitions do not depend on the time n because then we can capture all relevant information in a matrix P that has p_{ij} in row i, column j. In the case of the rainy-sunny Markov chain, this matrix is

$$P = \left[\begin{array}{c|c|c} & R & S \\ \hline R & p_{RR} & p_{RS} \\ S & p_{SR} & p_{SS} \end{array}\right]$$

Clearly, $p_{RS} = 1 - p_{RR}$. It is generally true that **each row has to sum up to 1** that is $\sum_j p_{ij} = 1$ for all rows i. Each row i is a discrete distribution that assigns probabilities to different next-step outcomes conditional on the current state i. The columns do not have this meaning and do not have to sum up to 1. Such a matrix is called a *stochastic* matrix.

6.2 Simulation

Taking the view that each row captures a distribution provides a useful starting point to think about simulation of a Markov chain using the knowledge we already have.

Let us re-visit the Rain-Sun example. If today it rains, then tomorrow is a random variable, lets call it Y^R, that has $\mathbb{P}\{Y^R = R\} = p_{RR} = 1 - \mathbb{P}\{Y^R = S\} = 1 - p_{RS}$. If today is sunny, then tomorrow is a random variable Y^S with $\mathbb{P}\{Y^S = R\} = p_{SR} = 1 - \mathbb{P}\{Y^S = S\} = 1 - p_{SS}$. We know how to simulate such binary random variables from Uniform random variables. In turn, we can simulate a path of the weather as follows

- For each step n
  - if $X_n = R$, simulate a value of Y^R and set $X_{n+1} = Y^R$.
  - if $X_n = S$, simulate a value of Y^S and set $X_{n+1} = Y^S$.

Recall that we would simulate Y^R for example by simulating a random variable $U \sim Uniform[0,1]$ and then setting $Y^R = R$ if $U \leq p_{RR}$ and $Y^R = S$ otherwise.

More generally, let us say we have a Markov chain with states $0, 1, \ldots, N$. Define Y^i to be a random variable with probability mass function

$$\mathbb{P}\{Y^i = j\} = p_{ij}.$$

Here, once again, p_{ij} is the probability of going from state i to state j in one step. Suppose we have (and we do) an algorithm to simulate a draw from X^i. Then:

- For each step n
  - For each state $i = 1, \ldots, N$
    - if $X_n = i$, simulate a value of Y^i and set $X_{n+1} = Y^i$.

6.3 Estimation

Suppose we are given historical data and we want to use the data to estimate the transition probabilities of an underlying Markov chain. A useful interpretation for $p_{ij} = P\{X_{n+1} = j | X_n = i\}$ is that it represents the fraction of those days where the state is i and then it is followed by the state j.

Based on the strong low of large numbers (which says that averages are close to the expectations and fractions to the true probabilities) one expects that a good estimate for p_{ij} is

$$\frac{\#\text{ of steps } n \text{ where } X_{n+1} = j \text{ and } X_n = i}{\#\text{ of steps } n \text{ where } X_n = i}.$$

In the case of rain and sun, p_{RR} can be estimated by counting the number of days where **it rains today and rains tomorrow** and dividing by the number of days it rains.

If we return to the way we constructed the simulation, we have Y^R with the distribution of X_{n+1} conditional on $X_n = R$. Every time we visit state R we have a new independent realization of Y^R, let's call it Y^R_k for the k^{th} visit to Rain. Let's say we have K rainy days within the first 300 days.

Then, what we are doing here in our estimation is saying that

$$\mathbb{P}\{Y^R = R\} \approx \frac{\text{Number of times } Y^R_k = R, \text{ over } k = 1, \ldots, K}{K} = \frac{\sum_{k=1}^{K} \mathbb{1}\{Y^R_k = R\}}{K}.$$

This now looks a bit like the strong law of large numbers. It is, as we intuitively argued, the number of days where **it rains today and rains tomorrow** and dividing by the number of day it rains.

For the strong law of large numbers to kick in, though, we need K to be a rather large number. If over 300 days it rains only $K = 2$ days, we do not really have a good sample size to work with.

6.4 Examples

6.4.1 A discrete queue

Here are the inputs:

- Time between consecutive arrivals is geometrically distributed with probability p_a. That is, if a customer arrives at time k, the next customer arrives at time $k + A$ where $A \sim Geom(p_a)$.
- Service time is geometrically distributed with parameter p_s. That is, if a customer begins service at time k, this customer completes service at time $k + S$ where $S \sim Geom(p_s)$.

There is a simple way to think about geometric interarrival and service times that is useful here.

- **Arrival coin:** Suppose that at each period n, we flip a coin (a biased one – with probability p_a of landing heads). If it falls *Head*, we have a customer arrival. If it falls *Tail* nothing happens. What would be the time until the first arrival. Well, this is the time until the first head in a sequence of independent experiments each with probability of success p_a.
- **Service coin:** Suppose that at each period n, we flip a coin (a biased one with probability p_s of landing heads). If it falls *Head*, we "kick out" the customer in service and free the server. Notice that by the same logic as above, the time it will take until we kick a customer out of service is geometrically distributed with probability of success p_s.

Notice that if there is one or more people in service, it means the server is busy serving the customer at the head of the line. If there are two or more people, it means that one person is in service and at least one is waiting in the line.

With the above interpretation it should be clear that all we need to track (in order to have a Markov chain) is the number of people in the system. If $i \geq 1$:

$$\begin{aligned} p_{i,i+1} &= \mathbb{P}\{X_{n+1} = i+1 | X_n = i\} = \mathbb{P}\{\text{arrival coin} = \textit{Head} \text{ and service coin} = \textit{Tail}\ \} \\ &= p_a(1-p_s) \end{aligned}$$

$$\begin{aligned} p_{i,i-1} &= \mathbb{P}\{X_{n+1} = i-1 | X_n = i\} = \mathbb{P}\{\text{arrival coin} = \textit{Tail} \text{ and service coin} = \textit{Head}\ \ \} \\ &= p_s(1-p_a). \end{aligned}$$

Since the probabilities have to sum up to 1 we also have the probability of staying in state i:

$$p_{i,i} = 1 - p_s(1-p_a) - p_a(1-p_s) = 1 - p_a - p_s + 2p_ap_s.$$

Lastly, if $i = 0$, we have $p_{0,1} = p_a$ and $p_{0,0} = 1 - p_a$, where all other $p_{i,j}$ are equal to 0.

6.4.2 An inventory model

You sell diapers. At the end of each day you go to your back room and check how many boxes of diapers remain. If there are 20 or fewer boxes, you order as many as you need to replenish the inventory up to 50. If, for example, the day ended up with 0 boxes, you will order 50. If there were 15 you will order 35. This is sometimes called the s-S inventory policy where little s is the inventory level that triggers an order (here 20), and big S is the "order up-to level" (here 50).

Demand is independent from one day to the next and follows some distribution. For concreteness let's say that $D \sim Poisson(10)$.

Let X_n stand for the number of boxes in inventory **at the end of day** n. This is the number of boxes before you place your order.

The key challenge in this question is to write down the transition probabilities. There are multiple scenarios here. Notice that at the end of the day your inventory could be any number between 0 and 50 but it can never be above 50. Also, if you finished your day n with $i > 20$ units in inventory, then this is the number with which you will start day $n + 1$. On the other hand, if $i \leq 20$, you will start day $n + 1$ with 50.

Overall, we have

$$\mathbb{P}\{X_{n+1}=j|X_n=i\}=\begin{cases}\mathbb{P}\{D_{n+1}=i-j\} & \text{if } i>20, \text{ and } 1\le j\le i,\\ \mathbb{P}\{D_{n+1}=50-j\} & \text{if } i\le 20, \text{ and } 1\le j\le 50,\\ \mathbb{P}\{D_{n+1}\ge i\} & \text{if } i>20, \text{ and } j=0,\\ \mathbb{P}\{D_{n+1}\ge 50\} & \text{if } i\le 20, \text{ and } j=0.\end{cases}$$

The last two cases correspond to the scenarios where your inventory is depleted by the end of day $n+1$. This means that the demand during the $(n+1)$st day was greater than your starting inventory on that day. If $X_n \le 20$, then you would have started your $(n+1)$st day with 50 units of inventory so reaching 0 means having a demand that is greater than or equal to 50. If $X_n = i > 20$ then you would have started day $n+1$ with i units so reaching 0 means having demand that equals or exceeds i.

The nice thing here is that if we could actually compute (or simulate) this Markov chain we would be able to answer, for example, on how many days some demand remains unsatisfied. Having unmet demand on day $n+1$, for example, could happen if either $X_n \le 20$ (so that day $n+1$ starts with 50 units in inventory) or if $X_n > 20$ (so that day $n+1$ starts with X_n). In total the probability of having unmet demand on day $n+1$ is

$$\begin{aligned}\mathbb{P}\{\text{Unmet demand on day } n+1\} &= \sum_{i\le 20}\mathbb{P}\{X_n=i, D_{n+1}>50\}+\sum_{i>20}\mathbb{P}\{X_n=i, D_{n+1}>i\}\\ &= \sum_{i\le 20}\mathbb{P}\{X_n=i\}\mathbb{P}\{D_{n+1}>50\}+\sum_{i>20}\mathbb{P}\{X_n=i\}\mathbb{P}\{D_{n+1}>i\}.\end{aligned}$$

In the last line, we use the fact that demand on period $n+1$ is independent of demand in all previous periods and hence from the state of inventory in previous periods.

Remark 1 Enlarging the state space to capture more history. Suppose that the probability of rain tomorrow depends not only on weather today but also on the weather yesterday. Specifically, suppose that

- $\mathbb{P}\{X_{n+1}=R|X_n=R, X_{n-1}=R\}=0.7$, in which case also $\mathbb{P}\{X_{n+1}=S|X_n=R, X_{n-1}=R\}=1-0.7=0.3$.
- $\mathbb{P}\{X_{n+1}=R|X_n=R, X_{n-1}=S\}=0.5$, in which case also $\mathbb{P}\{X_{n+1}=S|X_n=R, X_{n-1}=S\}=0.5$
- $\mathbb{P}\{X_{n+1}=R|X_n=S, X_{n-1}=R\}=0.4$, in which case also $\mathbb{P}\{X_{n+1}=S|X_n=S, X_{n-1}=R\}=1-0.4=0.6$
- $\mathbb{P}\{X_{n+1}=R|X_n=S, X_{n-1}=S\}=0.2$, in which case also $\mathbb{P}\{X_{n+1}=S|X_n=S, X_{n-1}=S\}=1-0.2=0.8$.

Notice that X_n itself is not a Markov chain because, for example, to know the likelihood of rain tomorrow you must know what happened also yesterday and not just today. If today was Rain and yesterday was Rain, tomorrow will be Rain with probability 0.7. But if today was rain and yesterday was Sun then this probability is 0.5.

However, what about $Y_n=(X_n, X_{n+1})$ (a moving window of two). Is this a Markov chain? The answer is yes and all you need to do is to verify that this is indeed the case; if I tell you the pair (today,yesterday) you will be able to give me a probability for (tomorrow,today). The state space however is much larger. It now includes pairs: RR,SR,RS,SS.

I leave it to you to compute the 4×4 one step function. In doing so, notice that there is no transition, for example, from $(X_{n-1}, X_n)=(S,R)$ to $(X_n, X_{n+1})=(S,R)$ (because X_n has to be the same).

ORIE 5530: Modeling Under Uncertainty

Lecture 7 (Performance evaluation of Markov chains)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

It will be useful to have a single example to be used throughout.

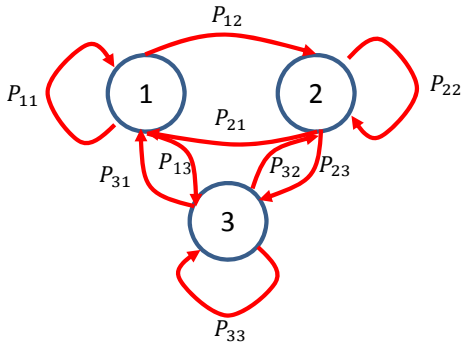


Figure 7.1: A 3-state Markov chain

7.1 Multi-step transition probability

In Figure 7.1 the probability of going in one step from state i to state j is P_{ij} – entry i, j of the one-step transition probability matrix. What, however, if we ask for the probability of doing so in n steps:

$$p_{ij}^n = \mathbb{P}\{X_n = j | X_0 = i\}.$$

Well, that can be more difficult. If $n = 2$ we can figure this out easily: you can get, for example, from state 1 to itself in two steps, by either staying there for two steps—with the corresponding probability $(P_{1,1})^2$—or by first going to state 2 and then back to state 1 which would have the probability $P_{12}P_{21}$.

But if I ask what happens in n steps it becomes a bit insane to count all possible trajectories. So let's build a simple formula that can be generalized to other Markov Chains. Starting with two steps, we just have to count all the options of where we could be in one step:

$$p_{12}^2 = \sum_{k=1}^{3} \mathbb{P}\{X_2 = 2, X_1 = k, X_0 = 1\} = \sum_{k=1}^{3} \mathbb{P}\{X_1 = k | X_0 = 1\}\mathbb{P}\{X_2 = 2 | X_1 = k, X_0 = 1\} = \sum_{k=1}^{3} P_{1k}P_{k2}.$$

Now notice that what we have obtained on the right hand side is the product of row 1 of the transition matrix P with the first column of P:

$$\begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}.$$

Thus, we have found that

$$p_{12}^2 = [P \times P]_{12} = [P^2]_{12}.$$

How about the three step probability $p_{12}^3 = \mathbb{P}\{X_3 = 2, X_0 = 1\}$?

$$p_{12}^3 = \sum_{k=1}^{3} \mathbb{P}\{X_3 = 2, X_1 = k, X_0 = 1\} = \sum_{k=1}^{3} \mathbb{P}\{X_1 = k | X_0 = 1\} \mathbb{P}\{X_3 = 2 | X_1 = k, X_0 = 1\}.$$

Notice that $\mathbb{P}\{X_3 = 2 | X_1 = k, X_0 = 1\} = \mathbb{P}\{X_3 = 2 | X_1 = k\} = \mathbb{P}\{X_2 = 2 | X_0 = k\} = p_{k2}^2$ is just the probability of getting from state k to state 2 in two steps. We already know that $p_{k2}^2 = [P^2]_{k2}$. Thus, we have that

$$p_{12}^3 = \sum_{k=1}^{3} \mathbb{P}\{X_1 = k | X_0 = 1\} \mathbb{P}\{X_3 = 2 | X_1 = k, X_0 = 1\} = \sum_{k=1}^{3} P_{1k} [P^2]_{k2} = [P^3]_{12}.$$

So we see that there is a pattern here that indicates

$$\mathbb{P}\{X_n = j | X_0 = i\} = [P^n]_{ij}. \qquad (n\text{-steps transitions})$$

This is rather beautiful because it says that all we need is matrix algebra.

Let's see a numerical example. Suppose that

$$P = \begin{bmatrix} 0.3 & 0.3 & 0.4 \\ 0.2 & 0.35 & 0.45 \\ 0.18 & 0.52 & 0.3 \end{bmatrix}.$$

Then,

$$P^2 = \begin{bmatrix} 0.198 & 0.403 & 0.375 \\ 0.159 & 0.3925 & 0.3405 \\ 0.1704 & 0.392 & 0.396 \end{bmatrix},$$

which, because the Markov chain has only three states, you can verify by yourself that it make sense. For example $[P^2]_{12}$ is the probability of getting in two steps from state 1 to state 2. We can do that in three ways: (i) first step back to 1 and second step to 2, which has probability $P_{11}P_{12} = 0.3 * 0.3$; (ii) first step to 2 and then from 2 to itself, which has probability $P_{12}P_{22} = 0.3 * 0.35$; (iii) from 1 to 3 and 3 to 2, which has probability $P_{13}P_{32} = 0.4 * 0.52$. The total is $0.3 * 0.3 + 0.3 * 0.35 + 0.4 * 0.52 = 0.403$.

So, even in two steps, the fact that we can just raise the matrix to the power of 2, saves us some work. Of course, it saves us a LOT of work if we want $P\{X_{10} = 3 | X_0 = 1\}$. But we can easily compute

$$P^{10} = \begin{bmatrix} 0.2137 & 0.4043 & 0.3820 \\ 0.2137 & 0.4043 & 0.3820 \\ 0.2137 & 0.4043 & 0.3820 \end{bmatrix},$$

and conclude that $P\{X_{10} = 3 | X_0 = 1\} = [P^{10}]_{13} = 0.382$. Interestingly, we see that (while the rows of P^2 were different than each other) those of P^{10} are equal to each other. We will be able to explain that soon.

7.2 Performance analysis

Suppose that, in the same Markov chain of Figure 7.1, we collect a reward $R(1)$ every time we visit state 1, a reward $R(2)$ every time we visit state 2, and a reward $R(3)$ when we visit state 3. You can think of the three Markov chain as capturing weather and $R(i)$ is the amount of sales I am making when the weather is in state i.

Consider discounting as a standard way of thinking about present value. If putting \$1 in the bank gives you an interest of r percent per day. Then, a dollar you will get tomorrow is worth $\beta = 1/(1 + r) < 1$ in

today's terms. A dollar received on day k is worth $1 \times \beta^k$ in todays terms. So if you seek to understand whether you will sell this product, you want to compare the investment you are making in today's dollars to the expected discounted reward

$$\sum_{k=0}^{\infty} \beta^k R(X_k).$$

This is random so you have to decided how you want to deal with this. It is standard just to look at expected present value. Suppose today (i.e., at time 0) $X_0 = 2$ so we are seeking to compute

$$\mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 2].$$

This seems difficult so let's go back to Figure 7.1. If you start at state 2, you collect today a reward $R(2)$ – that, for one, is certain.

Now, what happens from tomorrow and on. With probability P_{21} you will be in state 1 tomorrow and your expected reward from then on will be $\mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 1]$. With probability P_{22} you will be in state 2 tomorrow and your reward from then on will be $\mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 2]$. Finally, with probability P_{23} you will be in state 3 tomorrow and your reward from then and on will be $\mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 3]$.

But you also have to take into account that every dollar made from tomorrow is worth only β today. This logic is basically the proof that

$$\mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 2] = R(2) + \beta \left(P_{21} \mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 1] + P_{22} \mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 2] + P_{23} \mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = 3] \right).$$

To have a cleaner equation, let's define

$$V(i) = \mathbb{E}[\sum_{k=0}^{\infty} \beta^k R(X_k) | X_0 = i].$$

Then, what we proved is that

$$\begin{aligned} V(2) &= R(2) + \beta \left(P_{21} V(1) + P_{22} V(2) + P_{23} V(3) \right) \\ &= R(2) + \beta \sum_{i=1}^{3} P_{2i} V(i). \end{aligned}$$

From this we can in the same way create the equations for the cases that today we are in state 1 or 3 (instead of 2), i.e., $X_0 = 1$ or $X_0 = 3$. We overall have then the set of equations

$$\begin{aligned} V(1) &= R(1) + \beta \sum_{i=1}^{3} P_{1i} V(i) \\ V(2) &= R(2) + \beta \sum_{i=1}^{3} P_{2i} V(i) \\ V(3) &= R(3) + \beta \sum_{i=1}^{3} P_{3i} V(i). \end{aligned}$$

This is just a linear system of equations with the unknowns being $V(1), V(2), V(3)$. We can solve this system using linear programming that you are covering in the optimization class. Fortunately, in this case we can also directly compute the solution.

Re-writing this system in matrix form we have

$$\begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \end{bmatrix} + \beta \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix}.$$

Notice that this is the same as

$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \beta \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \right) \begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \end{bmatrix},$$

or inverting

$$\begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \beta \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \right)^{-1} \begin{bmatrix} R(1) \\ R(2) \\ R(3) \end{bmatrix}.$$

Inverting matrices is something that computers do beautifully. Here is a numerical example (you will be doing such a computation in the homework). Suppose that, $\beta = 0.95$, $R(i) = i^2$ and that, as before,

$$P = \begin{bmatrix} 0.3 & 0.3 & 0.4 \\ 0.2 & 0.35 & 0.45 \\ 0.18 & 0.52 & 0.3 \end{bmatrix}.$$

Then, the equation becomes

$$\begin{aligned} \begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix} &= \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - 0.95 \begin{bmatrix} 0.3 & 0.3 & 0.4 \\ 0.2 & 0.35 & 0.45 \\ 0.18 & 0.52 & 0.3 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix} \\ &= \begin{pmatrix} 0.715 & -0.285 & -0.38 \\ -0.19 & 0.6675 & -0.4275 \\ -0.171 & -0.494 & 0.715 \end{pmatrix}^{-1} \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix} \\ &= \begin{pmatrix} 5.151249685 & 7.579327435 & 7.26942288 \\ 4.045311986 & 8.63928523 & 7.315402784 \\ 4.026920024 & 7.781639014 & 8.191440962 \end{pmatrix} \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix} = \begin{matrix} 100.8933653 \\ 104.441078 \\ 108.8764447 \end{matrix} \end{aligned}$$

In this case, then, our total discounted reward is of the order of a 100 with small differences depending on where we start.

ORIE 5530: Modeling Under Uncertainty

Lecture 8 (Long-run averages)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

Let us stick to the example we had in the previous handout and which has three states.

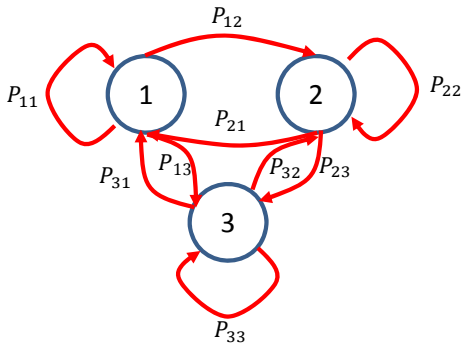


Figure 8.1: A 3-state Markov chain

In the last couple of lectures we saw how to derive multi-step transition probabilities (by taking powers of the one-step transition probability matrix) as well as how to evaluate infinite horizon discounted performance. In both cases we reduced performance evaluation to matrix algebra.

Specifically, we know now to compute the expectation of a function of the three-state Markov chain in Figure 8.1 at its k^{th} step:

$$\mathbb{E}[f(X_k)|X_0 = x] = \sum_{y=1}^{3} f(y)\mathbb{P}\{X_k = y|X_0 = x\} = \sum_{y=1}^{3} f(y)[P^k]_{xy},$$

and, in turn, we can compute cumulative finite horizon performance (over an horizon that has n periods)

$$\mathbb{E}[\sum_{k=0}^{n-1} f(X_k)|X_0 = x] = \sum_{k=0}^{n-1} \mathbb{E}[f(X_k)|X_0 = x] = \sum_{k=0}^{n-1}\sum_{y=1}^{3} f(y)[P^k]_{xy}. \qquad \text{(Finite horizon performance)}$$

This, notice, is just about taking powers of the matrix

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}.$$

We also know how to compute the infinite horizon discounted performance

$$V(x) = \mathbb{E}[\sum_{k=0}^{\infty} \beta^k f(X_k)|X_0 = x], \qquad \text{(Infinite-horizon discounted performance)}$$

using the matrix P. Specifically, it is the solution of the set of equation

$$\begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \end{bmatrix} + \beta \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} V(1) \\ V(2) \\ V(3) \end{bmatrix}.$$

See the previous handout.

One sort of performance analysis we turn to next is *long-run average performance*. Say, we are interested in the average over a very long horizon of n periods. We can reasonably approximate it by the infinite horizon average given by

$$\lim_{n\to\infty} \mathbb{E}[\frac{1}{n}\sum_{k=0}^{n-1} f(X_k)|X_0 = x]. \qquad \text{(Long-run average performance)}$$

Compare this to the finite horizon performance. Let us take one more step — using the fact that the sum of expectation equals the expectation of the sum — to write

$$\lim_{n\to\infty} \mathbb{E}[\frac{1}{n}\sum_{k=0}^{n-1} f(X_k)|X_0 = x] = \lim_{n\to\infty} \frac{1}{n}\sum_{k=0}^{n-1} \mathbb{E}[f(X_k)|X_0 = x].$$

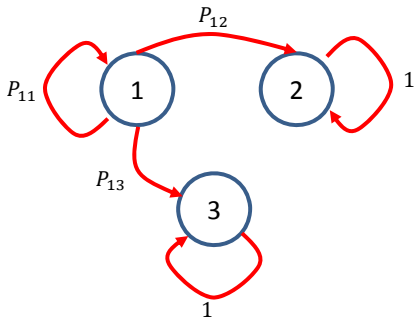
To study this criterion we need to introduce the notions of *stationary distributions* for Markov chains. First some preliminaries.

8.1 Long-run averages and stationary distributions

Suppose that in the three-state Markov chain of Figure 8.1 we have $P_{22} = P_{32} = 0$: once you reach 2 or 3 you get stuck there forever. The matrix is of the form

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

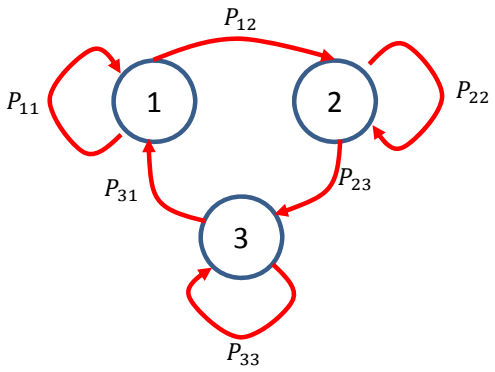
and the figure has the simple form



We could still talk about long-run average performance here. But the result will depend on where we start at time 0. If we start at 2 (that is $X_0 = 2$), we stay there forever and the long-run average is just $f(2)$. If we start at 3 we stay there forever so the long-run average is $f(3)$.

There are chains where the long-run average does not depend on where we start. These are *irreducible chains*. Simply, a chain is irreducible if from any pair of states x and y there is a path (of positive probability) from x to y in the graph.

Irreducibility does not require that you can go in one step from one state to another, rather just that there is a path. In yet another instance of the 3-state chain depicted below, you can get from any state to any other state but it will take, for example, at least two steps to get from 1 to 3.



If the chain is irreducible, there is, it turns out, an algebraic way to compute the long-run average. First, let's think just about the fraction of time the 3-state Markov chain spends in state 1. That is, you count step k if $X_k = 1$ (we write this as $\mathbb{1}\{X_k = 1\}$). You sum these up and divide by the number of steps:

$$\mathbb{E}\left[\frac{1}{n}\sum_{k=0}^{n-1}\mathbb{1}\{X_k = 1\}|X_0 = x\right].$$

We will take n to be large to get the long-run average. That is, we are looking for the long-run fraction of time, π_1, spent in state 1:

$$\pi_1 = \lim_{n\to\infty}\mathbb{E}\left[\frac{1}{n}\sum_{k=0}^{n-1}\mathbb{1}\{X_k = 1\}|X_0 = x\right].$$

We could similarly look for π_2 and π_3. Notice that we can write this "forgetting" about the fact that $X_0 = x$. In irreducible chains, the long-run average does not depend on whether we started at state 2, 3 or 1 (we can, in other words, drop the conditioning). The fundamental result here is that you can find these long-run fractions by solving a simple system of linear equations

$$\begin{aligned}\pi_1 &= \pi_1 P_{11} + \pi_2 P_{21} + \pi_3 P_{31},\\ \pi_2 &= \pi_1 P_{12} + \pi_2 P_{22} + \pi_3 P_{32},\\ \pi_3 &= \pi_1 P_{13} + \pi_2 P_{23} + \pi_3 P_{33},\end{aligned}$$

with the added requirements that $\pi_1, \pi_2, \pi_3 \geq 0$ and that $\pi_1 + \pi_2 + \pi_3 = 1$. These two latter conditions are obvious, you want fractions to be positive and sum up to 1. Define $\pi = (\pi_1, \pi_2, \pi_3)$ and e to be a (column) vector of all ones. Then, in matrix notation, the above is written as

$$\pi \geq 0, \quad \pi e = 1, \quad \pi = \pi P.$$

Before proceeding, let me convince you — still focusing on the 3-state chain and on the fraction of time spent in state 1 — that this makes sense. Let's first agree that we can partition the event that $X_k = 1$ into three mutually exclusive sub-events:

$$\{X_k = 1\} = \{X_k = 1, X_{k-1} = 1\} \cup \{X_k = 1, X_{k-1} = 2\} \cup \{X_k = 1, X_{k-1} = 3\}.$$

So, we can write

$$\begin{aligned}\mathbb{P}\{X_k = 1\} &= \mathbb{P}\{X_k = 1, X_{k-1} = 1\} + \mathbb{P}\{X_k = 1, X_{k-1} = 2\} + \mathbb{P}\{X_k = 1, X_{k-1} = 3\} \\ &= \mathbb{P}\{X_k = 1 | X_{k-1} = 1\}\mathbb{P}\{X_{k-1} = 1\} + \mathbb{P}\{X_k = 1 | X_{k-1} = 2\}\mathbb{P}\{X_{k-1} = 2\} \\ &\quad + \mathbb{P}\{X_k = 1 | X_{k-1} = 3\}\mathbb{P}\{X_{k-1} = 3\} \\ &= \sum_{j=1}^{3} \mathbb{P}\{X_{k-1} = j\} P_{j1}.\end{aligned}$$

This is very useful. Why? It means we can write the amount of time spent in state 1 as a weighted average of the other states, namely

$$\begin{aligned}\frac{1}{n}\sum_{k=1}^{n-1} \mathbb{E}\left[\mathbb{1}\{X_k = 1\}\right] &= \frac{1}{n}\sum_{k=1}^{n-1} \mathbb{P}\{X_k = 1\} \\ &= \frac{1}{n}\sum_{k=1}^{n-1} \mathbb{P}\{X_{k-1} = 1\} P_{11} + \frac{1}{n}\sum_{k=1}^{n-1} \mathbb{P}\{X_{k-1} = 2\} P_{21} + \frac{1}{n}\sum_{k=1}^{n-1} \mathbb{P}\{X_{k-1} = 3\} P_{31}\end{aligned}$$

But, recall that we defined $\pi_j = \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n-1} \mathbb{P}\{X_{k-1} = j\}$ (you could complain that actually in my definition I counted from $k = 0$ to $n - 1$, but as n grows, we can ignore the first term). So what we have shown above is that (map the colors)

$$\pi_1 = \pi_1 P_{11} + \pi_2 P_{21} + \pi_3 P_{31}.$$

This explains the equations we introduced for the long-run fractions.

Example computation for finite-space Markov chain As noted above, we can compute the desired long-run fractions π by solving a simple system of linear equations and there exists numerical software to do so. To illustrate one approach, as discussed in class, we can rewrite $\pi = \pi P$ as $\pi(I - P) = 0$, where I denotes the identity matrix. We can incorporate $\pi e = 1$ in this system of linear equations by defining Z to be the matrix $(I - P)$ with any column, say the last column, replaced by all ones (the vector e) and by defining b to be the (column) vector $(0, 0, \ldots, 0, 1)$. Then we have $\pi = bZ^{-1}$, from which we obtain π by inverting the matrix Z and multiplying the vector b on the right by this inverse matrix Z^{-1}.

From time-fractions to rewards Our original objective was to figure out the long-run rewards. When we are in state i we collect a reward $f(i)$. But what we had above is only a way to compute the fraction of steps we spend in state i. Well, obviously, if we know that we spend 20 percent of the steps in state 1, we also know that on 20 percent of the steps we collect a reward of $f(1)$, etc. Mathematically, for the three-state example:

$$\mathbb{E}[f(X_k) | X_0 = x] = \sum_{j=1}^{3} f(j)\mathbb{P}\{X_k = j | X_0 = x\}.$$

Thus, the long-run average reward is

$$\frac{1}{n}\sum_{k=0}^{n-1} f(X_k) = \frac{1}{n}\sum_{k=0}^{n-1}\sum_{j=1}^{3} f(j)1\{X_k = j\} = \sum_{j=1}^{3} f(j)\frac{1}{n}\sum_{k=0}^{n-1} 1\{X_k = j\}.$$

As n grows large we know that $\mathbb{E}[\frac{1}{n}\sum_{k=0}^{n-1} 1\{X_k = j\}]$ converges to π_j. Thus, we have

$$\lim_{n\to\infty} \mathbb{E}[\frac{1}{n}\sum_{k=0}^{n-1} f(X_k)] = \sum_{j=1}^{3} f(j) \lim_{n\to\infty} \mathbb{E}[\frac{1}{n}\sum_{k=0}^{n-1} 1\{X_k = j\}] = \sum_{j=1}^{3} f(j)\pi_j.$$

So, recalling that to solve for π_j all we need is to solve linear equations, this means that for any reward function we can build on those linear equations to evaluate the long-run average rewards.

Why is π called the stationary distribution? First notice that π is a legitimate distribution. It is non-negative ($\pi \geq 0$) and sums up to 1 ($\sum_{j=1}^{3} \pi_j = 1$). It turns out, *if you start at time $k = 0$ in π — that is $X_0 = j$ with probability π_j — then you get stuck there*, meaning that

$$\mathbb{P}\{X_k = j\} = \pi_j, \text{ for all times } k.$$

We can prove that. For example, if X_0 follows the distribution π then

$$\mathbb{P}\{X_k = 1\} = \sum_{j=1}^{3} \mathbb{P}\{X_k = 1|X_0 = j\}\mathbb{P}\{X_0 = j\} = \sum_{j=1}^{3} \pi_j P_{j1} = \pi_1.$$

The last equality is just from our system of equations for π.

In what practical contexts do we use long-run average? This is a question that we should open with but, actually, there is value in addressing this now that we have seen the analysis.

One family of settings where this is relevant concerns those where it is natural to think about long-run average and discounting does not make sense because the rewards, say, are not monetary. For example, say you run the emergency service (ambulance and fire department) for LA county. You might be interested in the fraction of ambulances that arrive to the location of the incident within 2 minutes over a year. It is not trivial to model such a network as a Markov chain. The point here is that long-run averages make sense here in contrast to discounting.

Another family is one where you are actually interested in performance over short time periods but long-run averages provide a reasonable approximation and one that is computationally simple **(remember – you only have to solve linear equations).**

8.2 Limit distributions

So π does not only capture long-run averages. It also is a stationary distribution: if you start there you stay there. It turns out, with some additional conditions, you do not even have to start there to get there. That is, no matter where you start, you will end up with π far enough in the future. In more mathematical terms: No matter what is the distribution of X_0,

$$\mathbb{P}\{X_k = j\} \approx \pi_j, \text{ for all } k \text{ large}$$

To have this, **in addition to irreducibility**, we have to require **aperiodicity**. It is useful to think here of yet another instance of our three-state Markov chain as illustrated in Figure 8.2.

Figure 8.2 is a trivial Markov chain where, with probability 1, you go from state 1 to state 2, from state 2 to state 3 and from state 3 to state 1. Here, you can return from 1 to itself in n steps only if n is divisible by 3 – 3 steps, 6 steps, 9 steps, etc. The probability of returning from 1 to itself is non-zero only if n is divisible by 3. We say that state 1 has period 3. In fact, this Markov chain is *periodic* with each state having period 3. The less trivial chain in Figure 8.3 is very different in this regard.

In Figure 8.3, we can return from state 1 to itself in 3 steps ($1- > 2- > 3- > 1$) but also in 4 steps ($1- > 2- > 2- > 3- > 1$ or $1- > 2- > 3- > 3- > 1$) and also in 5 steps, etc. Here **the period of state 1 is 1.**

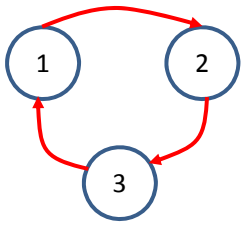


Figure 8.2: Simple periodic Markov chain

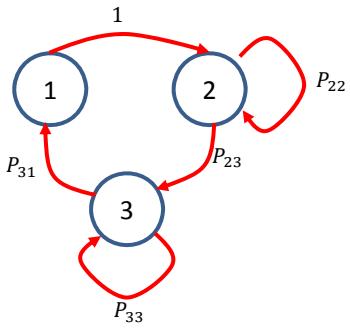


Figure 8.3: Simple aperiodic Markov chain

When all states have a period of 1, we say that the Markov chain is *aperiodic*. The Markov chain in Figure 8.3 is aperiodic. The fundamental result is then that **for irreducible aperiodic chains, π (computed by the linear system of equations) is a limit distribution** in the sense that

$$\mathbb{P}\{X_k = j\} \to \pi_j, \text{ as } k \to \infty.$$

Why is aperiodicity needed here? Well, let's look again at Figure 8.2.

My statement above is clearly not true here. If we start in state 1, $\mathbb{P}\{X_{3k} = 1|X_0 = 1\} = 1$ for all k (no matter how large) but $\mathbb{P}\{X_{2k} = 1|X_0 = 1\} = 0$ for all k. So, if we just "visit/sample" this chain at random periods, what we will see will not be really informative of the long-run average behavior of the chain.

If it is irreducible and aperiodic then we are on solid ground. Sampling the chain gives us a true understanding of its long-run average.

There are also cases where we are truly interested in the limit distribution (rather than the long-run average):

1. Macro economics: you introduce a new taxation policy that is designed to affect the distribution of income in society. It will take some time (indeed years) until this has a visible effect, but what you are making is a long-term decision and you are interested in how the distribution of income will eventually look.

2. Algorithms: Look at the case of applying the results in the next example based on page rank.

8.3 Some cool facts about moving on graphs: page rank

Consider a connected (undirected) graph that has a weight w_{ij} associated with the edge between i and j. **Notice that $w_{ij} = w_{ji}$ (there is no importance to order)**. Suppose that when in state i you move to state j with a probability that is proportional to the weight $w_{i,j}$. Specifically,

$$P_{ij} = \frac{w_{ij}}{\sum_k w_{ik}}.$$

Then, it turns out we have a beautifully simple expression for the stationary distribution π. It is given by

$$\pi_i = \frac{\sum_k w_{ik}}{\sum_{j,l} w_{jl}} = \frac{\text{total outgoing weight from node } i}{\text{total outgoing weight of all nodes}}.$$

We can verify that this is indeed correct. When we have a guess for a stationary distribution, all we have to do is verify that it satisfies the equations $\pi \geq 0$, $\sum_i \pi_i = 1$ and $\pi = \pi P$. It is trivial here that the sum is 1 and that all π_i are non-negative. Let's check $\pi_j = \sum_i \pi_i P_{ij}$:

$$\sum_i \pi_i P_{ij} = \sum_i \frac{\sum_k w_{ik}}{\sum_{kl} w_{kl}} \frac{w_{ij}}{\sum_k w_{ik}} = \frac{\sum_i w_{ij}}{\sum_{kl} w_{kl}} = \pi_j.$$

Now, let's use it on **page rank**. The basic version of page rank is that the "surfer" moves from webpage i to webpage j (linked to i) with probability $1/n_i$ where n_i is the number of outgoing links from page i. Suppose that the web we consider always has a back link. Namely that if i links to j then j links to i. So we have a movement on a graph. What are the corresponding weights though? Well, take $w_{ij} = 1$ for all edges. Then, consistent with what we want

$$P_{ij} = \frac{w_{ij}}{\sum_j w_{ij}} = \frac{1}{n_i}.$$

So, we can apply our results on graphs to argue that, in the long run, a fraction

$$\pi_i = \frac{\sum_j w_{ij}}{\sum_{kl} w_{kl}} = \frac{n_i}{\sum_k n_k} = \frac{\text{page } i \text{ outward links}}{\text{total web outward links}}$$

of a surfer's page-visits will be to page i. This is cool because it means that the frequency of page visits is proportional to the **outgoing links** of a page rather than ingoing links. Kind of surprising but not too much because you have to remember that, in our model, the more outward links you have the more inward links you have as well in our model. In the real web, you might not have *back links*.

ORIE 5530: Modeling Under Uncertainty

Lecture 9 (Continuous Time Markov Chains)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

We begin our journey into continuous-time Markov chains (CTMCs). Simply put, CTMCs are an extension of discrete-time Markov chains (DTMCs) where, instead of staying in each state for exactly one time unit, the chain remains in each state for a random amount of time that follows an exponential distribution.

There are two **equivalent** definitions of continuous time Markov chains. The first is based on defining state-clocks and transition probabilities and the second is what one can call a "competing-clocks" construction.

It is useful to have both as they can be handy in different settings.

Some of this material follows Chapter 6 in Ross's book. The first construction, for example, follows Section 6.2 there. However, Chapter 6 of Ross takes for granted that you had gone through his chapter 5 that is about the exponential distribution. We will summarize the needed essentials within the notes below.

9.1 First definition

Let's take as a starting point a three-state chain.

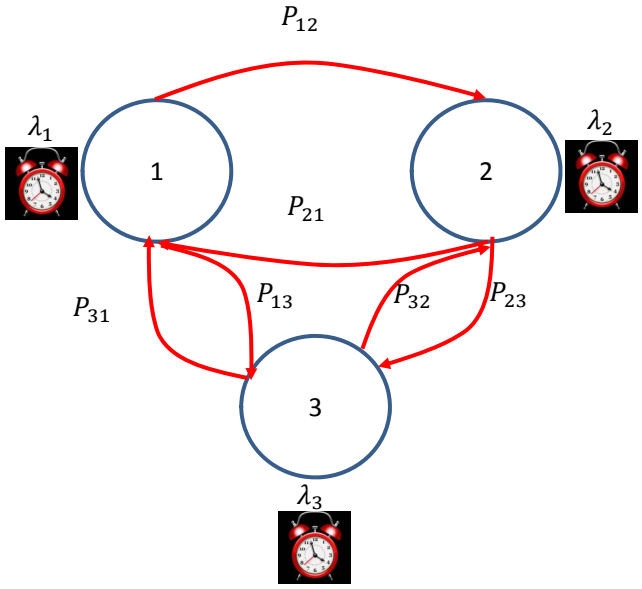


Figure 9.1: A 3-state continuous time Markov chain

Here is how the dynamics work. Suppose you start at time 0 in state 1. That is, $X_0 = 1$. You set up an alarm clock to go off after an exponential amount of time with rate λ_1 and hence expectation $1/\lambda_1$. We will refer frequently to the parameter of the exponential as its rate. When the alarm clock sounds, you move with probability P_{12} to state 2 and with probability $P_{13} = 1 - P_{12}$ to state 3. Say, you moved to

state 2. As soon as you enter the state, you activate an exp(λ_2) alarm clock. Once it sounds you move with probability P_{21} to state 1 and with probability $P_{23} = 1 - P_{21}$ to state 3, and so on in this manner.

Notice that this fully specifies the process: if someone asked you to simulate this process, you have enough information to do so, right? So all we need are two ingredients — the rates

$$\lambda = (\lambda_1, \lambda_2, \lambda_3)$$

of the alarm clocks and the transition probability matrix

$$P = \begin{bmatrix} 0 & P_{12} & P_{13} \\ P_{21} & 0 & P_{23} \\ P_{31} & P_{32} & 0 \end{bmatrix}.$$

In summary, a CTMC is a process having the property that each time it enters state i:

(1) the amount of time it spends in that state before making a transition into a different state is exponentially distributed with mean $1/\lambda_i$ and

(2) when the process leaves state i, it next enters state j with probability P_{ij}. The probabilities P_{ij} must satisfy $P_{ii} = 0$ for all i (no return in one step) and $\sum_j P_{ij} = 1$.

What makes this process that we just created appealing? Well, if you tell someone that the chain in Figure 9.1 is in state 1 now (say this is minute 4), then they can compute the probability that it will be in state 2 in 6 minutes from now. We do not need to know how much time has passed since you entered state 1. That is, even if you throw in all the history before minute 4, we do not care about that:

$$\mathbb{P}\{X_{10} = 2 | X_4 = 1, X_s = x_s; s < 4\} = \mathbb{P}\{X_{10} = 2 | X_4 = 1\}.$$

Why is that? This is driven by what is called the memoryless property of the exponential distribution. For the above, it implies that if you tell me that 30 seconds passed since the chain entered state 1, then the distribution of the remaining time until you move is still exp(λ_1) – in other words, the likelihood that is takes another 6 minutes until you move is as if you moved to state 1 just now, hence we can ignore when you entered.

Formally, if $Y \sim exp(\lambda)$,

$$\mathbb{P}\{Y \geq s + t | Y \geq t\} = \mathbb{P}\{Y \geq s\} = e^{-\lambda s}.$$

It should also be clear that, because nothing in the properties of the chain (the vector λ and P) changes with time, the probability of moving from state 1 to state 2 in 5 minutes depends only on the time that passes (6 minutes) and not on the running time:

$$\mathbb{P}\{X_{10} = 2 | X_4 = 1\} = \mathbb{P}\{X_6 = 2 | X_0 = 1\}.$$

9.2 Second definition: Competing clocks

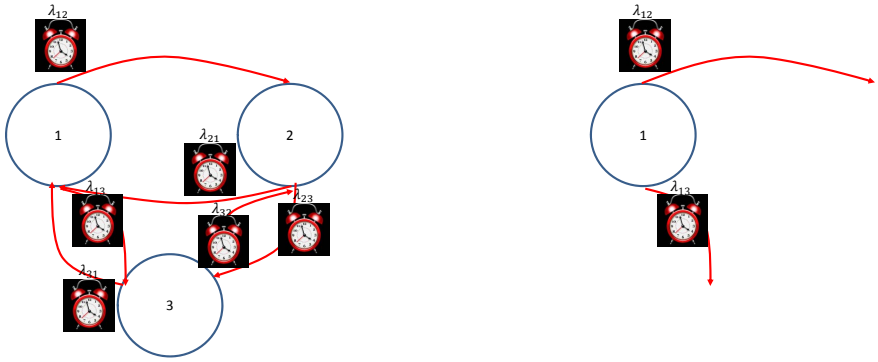


Figure 9.2: A 3-state continuous time Markov chain: competing clocks

Let us focus on the subset of the chain captured on the right-hand side of Figure 9.2. Here are the dynamics:

> Say you just entered state 1. You activate two clocks–the first exponential with rate λ_{12}—clock A—and the second exponential with rate λ_{13}—clock B. The first clock that rings determines where you go. If clock A rings first, you move to state 2 and if clock B rings first you move to state 3. That is, you move after the **minimum of the two exponentials with rates λ_{12} and λ_{13}**.
>
> When in state 2, you have two clocks with competing rates λ_{23} and λ_{21} and you do the same. Analogously for state 3.

Again, this is a complete specification meaning that the above description and the values λ_{ij} are sufficient information to simulate this process.

This is often (as we will see) a more useful construction than the first one. Essentially, however, they are both equivalent:

- How much time until the chain leaves state 1: It is the minimum of two independent exponentials—one, let's call it Y_{13} with rate λ_{13} and the other Y_{12} with rate λ_{12}. The following is known

  $$\min\{Y_{12}, Y_{13}\} \sim exp(\lambda_{12} + \lambda_{13}).$$

  That is the minimum is itself exponential with the sum of the rates.

  Letting $\lambda_i = \sum_j \lambda_{ij}$ we have that after entering state i the chain stays there for an exponential amount of time with rate λ_i.

- What is the likelihood that after leaving state 1, you go to state 2 rather than state 3? This is the likelihood that clock A rings before clock B and it turns out that

  $$\mathbb{P}\{Y_{12} < Y_{13}\} = \frac{\lambda_{12}}{\lambda_{12} + \lambda_{13}},$$

  and the other case is

  $$\mathbb{P}\{Y_{13} < Y_{12}\} = \frac{\lambda_{13}}{\lambda_{12} + \lambda_{13}}$$

  (the probability that $Y_1 = Y_2$ is zero for two independent continuous random variables).

Moreover, this probability is independent of how much time it took until the clocks rang. That is,

$$\mathbb{P}\{Y_{12} < Y_{13} | \min\{Y_{12}, Y_{13}\} = y\} = \frac{\lambda_{12}}{\lambda_{12} + \lambda_{13}}.$$

In summary, say someone gives you a construction with the parameters λ_{ij} as above. Define

$$\lambda_i = \sum_j \lambda_{ij}, \ P_{ij} = \frac{\lambda_{ij}}{\lambda_i}.$$

Then, it turns out that both properties we introduce in Section 1 hold:

(1) the amount of time the chain spends in that state before making a transition into a different state is exponentially distributed with mean $1/\lambda_i$ and

(2) when the process leaves state i, it next enters state j with probability P_{ij}. The probabilities P_{ij} must satisfy $P_{ii} = 0$ for all i (no return in one step) and $\sum_j P_{ij} = 1$.

WE ARE MORE OFTEN THAN NOT GOING TO USE THE SECOND CONSTRUCTION.

9.3 The practical approach to constructing rates: An Uber example

The practical approach to construct these CTMCs is to forget about the underlying math and think about the speeds (rates) (the math kicks-in in justifying that this simple way of thinking is correct).

In the single-server queue example we did in class — the speed of arrival was λ and the speed of service completions was μ. In other words, if each service takes an expontential time with expectation $1/\mu$ minutes, it means that we have completions at a speed (rate) of μ per minute, as long as the server remains busy.

Consider the Uber example we started in class but let us limit our attention to two regions A and B. Here are the ingredients:

Customers: Customers arrive to region A at speed λ_A and to region B at speed λ_B. A customer taking a cab in region A wants to get to a point in region A with probability P_{AA} and wants to get to a point in region B with probability P_{BB}. The speed (or rate) at which customers take a cab to go from region A to region B is then $\lambda_A P_{AB}$.

Ride time: A trip from a point in region A to a point in region B takes an exponential amount of time with mean $1/\mu_{AB}$. Similarly, we define μ_{AA}, μ_{BA} and μ_{BB}.

Driver self movement: When a ride is complete in region B the driver can choose to stay in this region or relocate. The driver will stay in region B with probability Q_{BB} after completing a ride or move to region A with probability Q_{BA}. We similarly define Q_{AA} and Q_{AB}.

The state is 8 dimensional as follows:

- Empty cabs in a region:
  - E_{AA} is the number of empty cabs in region A
  - E_{BB} is the number of empty cabs in region B
- Empty cabs in motion:
  - E_{AB} is the number of empty cabs in transition from A to B

  - E_{BA} is the number of empty cabs in transition from B to A

- Taken cabs in motion:
  - F_{AA} is the number of busy (i.e., taking a customer) cabs in transition from one point in A to another point in A
  - F_{BB} is the number of busy (i.e., taking a customer) cabs in transition from one point in B to another point in B
  - F_{AB} is the number of busy (i.e., taking a customer) cabs in transition from A to B
  - F_{BA} is the number of busy (i.e., taking a customer) cabs in transition from B to A

Our process is the **8-dimensional** chain

$$X(t) = (E_{AA}(t), E_{BB}(t), E_{AB}(t), E_{BA}(t), F_{AA}(t), F_{BB}(t), F_{AB}(t), F_{BA}(t)),$$

where the states are of the form

$$e = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}).$$

What are the transition speeds from state e^1 to state e^2. There are only certain types of transition possible

- Customers taking cabs:
  - A customer takes a cab in region A and goes to region B. In this case e_{AA} decreases by one (a cab is taken from region A) and f_{AB} increases by 1 (a cab is driving to region B). The speed at which customers arrive and are interested in such a ride is $\lambda_A P_{AB}$. So, if $e_{AA} > 0$ (there are cabs available), we have a transition

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{BA}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA} - 1, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB} + 1, f_{BA}, \end{aligned}$$

    with speed (rate) $\lambda_A P_{AB}$. That is $\lambda_{e^1 e^2} = \lambda_A P_{AB}$
    Similarly for a customer that takes a cab from region A and stays in A we have with rate $\lambda_A P_{AA}$ a transition

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA} - 1, e_{BB}, e_{AB}, e_{BA}, f_{AA} + 1, f_{BB}, f_{AB}, f_{BA}, \end{aligned}$$

    and for transitions from B to A and from B to itself we respectively have $\lambda_B P_{BA}$ and $\lambda_B P_{BB}$ with the corresponding state transitions.

- Customers completing rides and cabs moving:
  - Since the ride from A to B takes an exponential amount of time with rate μ_{AB}, when there are f_{AB} cabs driving from A to B the first of these cabs will be done with rate $\mu_{AB} f_{AB}$ (this is like in the multiserver queueing model). When this ride is done, the driver will stay in region B with probability Q_{BB} or will move back to A with probability Q_{BA}. This will increase by one the number of empty cabs going from B to A (this is e_{BA}).
    Thus, we have the transitions that move

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA}, e_{BB} + 1, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB} - 1, f_{BA}), \end{aligned}$$

    at a speed of $\mu_{AB} f_{AB} Q_{BB}$ and we move

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA}, e_{BB}, e_{AB}, e_{BA} + 1, f_{AA}, f_{BB}, f_{AB} - 1, f_{BA}), \end{aligned}$$

    at a speed of $\mu_{AB} f_{AB} Q_{BA}$.

- Empty cabs completing an empty transition:
  - The ride from A to B takes an exponential amount of time with rate μ_{AB}. By the same argument as above, when there are e_{AB} cabs transitioning from A to B, completions happen at rate $\mu_{AB}e_{AB}$ and hence we have transitions that move

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA}, e_{BB}+1, e_{AB}-1, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}), \end{aligned}$$

    at a speed of $\mu_{AB}e_{AB}$, and similarly moving

$$\begin{aligned} \text{From} \quad & e^1 = (e_{AA}, e_{BB}, e_{AB}, e_{BA}, f_{AA}, f_{BB}, f_{AB}, f_{BA}) \\ \text{To} \quad & e^2 = (e_{AA}+1, e_{BB}, e_{AB}, e_{BA}-1, f_{AA}, f_{BB}, f_{AB}, f_{BA}), \end{aligned}$$

    with rate $\mu_{BA}e_{BA}$.

ORIE 5530: Modeling Under Uncertainty

Lecture 10 (CTMC: Performance metrics)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

The topic of these two lectures is performance analysis of continuous time chains. The note has three parts. The first deals with long-run average, the second with infinite horizon discounted rewards and the last with finite-time horizon performance.

10.1 Long-run average performance

We collect a reward $f(j)$ when in state j and we want to capture the long-run average $\frac{1}{T}\mathbb{E}[\int_0^T f(X(s))ds]$ for T large. Let's get back to our base three-state example below.

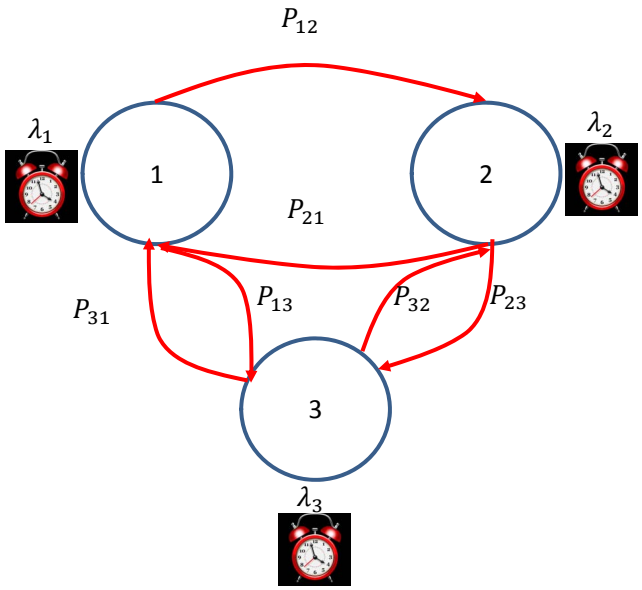


Figure 10.1: A 3-state continuous time Markov chain

If you just look at steps: first step, second step, etc., the movement essentially follows the discrete time Markov chain with transition probabilities P_{ij}. For that chain we know how to compute the long run average. The long-run average fraction of steps it spends in state i is π_i^D (D for discrete) where π^D solves

the system of equations

$$\pi_i^D = \sum_j \pi_j^D P_{ji}, \ i = 1, 2, 3,$$
$$\sum_i \pi_i^D = 1,$$
$$\pi_i^D \geq 0, \ i = 1, 2, 3.$$

(we have written the first equation often in matrix form as $\pi^D P = \pi^D$).

Then after N steps (N large), we can approximate the number of steps in state i as $N\pi_i^D$. Each time we visit state i we remain there for an exponential amount of time with parameter λ_i (and hence mean $1/\lambda_i$). Thus, after N steps we will be in state i for $N\pi_i^D \times \frac{1}{\lambda_i}$. The same applies to all three states. Hence, the fraction **of time** the continuous time chain spends in state 1 is given by

$$\pi_1 = \frac{\text{Total amount in 1}}{\sum_{j=1}^3 \text{Total amount in j}} = \frac{N\pi_1^D/\lambda_1}{\sum_{j=1}^3 N\pi_j^D/\lambda_j} = \frac{\pi_1^D/\lambda_1}{\sum_{j=1}^3 \pi_j^D/\lambda_j}.$$

Conclusion 1: The long-run fraction of time the continuous time chain spends in state i is given by

$$\pi_i = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T \mathbb{1}\{X(s) = i\}ds\right] = \frac{\pi_i^D/\lambda_i}{\sum_{j=1}^3 \pi_j^D/\lambda_j},$$

where π_i^D is the stationary distribution of the corresponding discrete time chain.

Equipped with this, we can then compute the long run average reward as

$$\begin{aligned}\lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T f(X(s))ds\right] &= \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T \sum_{j=1}^3 f(j)\mathbb{1}\{X(s) = j\}ds\right] \\ &= \sum_{j=1}^3 f(j) \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T \mathbb{1}\{X(s) = i\}ds\right] \\ &= \sum_{j=1}^3 f(j)\pi_j,\end{aligned}$$

where we would now plug in the expression for π from our **Conclusion 1**.

There is also a direct way to compute the long-run fraction by using the rate matrix (recall that $\lambda_{i,j} = \lambda_i P_{ij}$)

$$Q = \begin{bmatrix} -\lambda_1 & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & -\lambda_2 & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & -\lambda_3 \end{bmatrix}.$$

To see this, by the equations that define π^D, $\pi_1^D = \sum_{k=1}^3 \pi_k^D P_{k1}$ so that

$$\pi_1 = \frac{\pi_1^D/\lambda_1}{\sum_{j=1}^3 \pi_j^D/\lambda_j} = \frac{\frac{1}{\lambda_1}\sum_{k=1}^3 \lambda_k P_{k1}\pi_k^D/\lambda_k}{\sum_{j=1}^3 \pi_j^D/\lambda_j} = \frac{1}{\lambda_1}\sum_{k=1}^3 \pi_k \lambda_k P_{k1}$$

So that (recall that $P_{11} = 0$) we have

$$\lambda_1\pi_1 = \sum_{k=2}^{3} \lambda_k P_{k1}\pi_k = \sum_{k=2}^{3} \pi_k \lambda_{k1},$$

or for any $i = 1, 2, 3$

$$\lambda_i\pi_i = \sum_{k\neq i} \lambda_k P_{ki}\pi_k = \sum_{k\neq i} \pi_k \lambda_{ki}.$$

In matrix form this is exactly the system of equations

$$\begin{pmatrix} \pi_1 & \pi_2 & \pi_3 \end{pmatrix} \begin{bmatrix} -\lambda_1 & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & -\lambda_2 & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & -\lambda_3 \end{bmatrix} = 0.$$

(in matrix notation $\pi Q = 0$). So we have

Conclusion 2 The long run fraction of time in state i

$$\pi_i = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T \mathbb{1}\{X(s) = i\}ds\right]$$

can be found by solving the system of equatiosn

$$\pi Q = 0,\ \sum_i \pi_i = 1,\ \pi_i \geq 0.$$

10.2 Infinite horizon discounted performance

Here we have a discount factor $\beta > 0$. This will mean that 1 dollar we make an hour from now is worth $e^{-\beta} < 1$ dollars. This is, as we discuss below, the same view as taken when calculating net present value of a project/investment; see `https://en.wikipedia.org/wiki/Net_present_value`. Specifically, the continuous time version. We have a Markov chain $(X(t), t \geq 0)$ and we collect a reward $f(i)$ per minute (or whatever time unit you are working with) in state i. The reward is collected here continuously. If we spend a minute in state 1 we collect $f(1)$ if we spend 1.2 minutes we collect $1.2f(1)$. We want to know the long-term discounted reward given the starting state i

$$V(i) = \mathbb{E}\left[\int_0^\infty e^{-\beta s} f(X(s))ds | X(0) = i\right].$$

Let's focus on state 1 in our 3-state Markov chain. We stay in state 1 for an $exp(\lambda_1)$ amount of time and then we leave. Let's call this time T_1. There are two components to the reward: up to T_1 and from T_1 and on.

- Up to T_1: We collect (discounted) rewards at a rate of $f(1)$ per minute so the reward until the first move is

$$\mathbb{E}\left[\int_0^{T_1} f(1)ds\right] = \mathbb{E}\left[\int_0^{T_1} e^{-\beta s}\right] f(1).$$

- From T_1 till infinity: Anything collected after T_1 is worth $e^{-\beta T_1}$ in time 0 terms. Also, if at T_1, $X(T_1) = 2$ (that is the first move is to 2) then, what we collect after T_1 is just the infinite horizon discounted reward starting at 2. Thus, we have

$$\mathbb{E}[e^{-\beta T_1}]\sum_{j\neq 1} P_{1j}\mathbb{E}\left[\int_0^\infty e^{-\beta s} f(X(s))ds|X(0)=j\right] = \mathbb{E}[e^{-\beta T_1}]\sum_{j\neq 1} P_{1j}V(j).$$

We also can compute, using the fact that $T_1 \sim exp(\lambda_1)$ that

$$\mathbb{E}[\int_0^{T_1} e^{-\beta s}] = \frac{1}{\beta}(1-\mathbb{E}[e^{-\beta T_1}]) = \frac{1}{\lambda_1+\beta}, \text{ and } \mathbb{E}[e^{-\beta T_1}] = \frac{\lambda_1}{\lambda_1+\beta}.$$

Thus, we have

$$\begin{aligned} V(1) = &\text{ reward up to } T_1 + \text{ reward from } T_1 \text{ and on} \\ &= \frac{1}{\lambda_1+\beta} f(1) + \frac{\lambda_1}{\lambda_1+\beta}\sum_{j\neq 1} P_{1j}V(j). \end{aligned}$$

Next, using the definition $\lambda_{1j} = \lambda_1 P_{1j}$ and re-arranging terms a bit — multiply both sides by $\lambda_1+\beta$ and then move the term on the left to the right—we have

$$0 = f(1) + \sum_{j\neq 1}\lambda_{1j}V(j) - \lambda_1 V(1) - \beta V(1).$$

We can repeat the same steps for state 2 and 3 to obtain, for each i,

$$\begin{aligned} 0 &= f(i) + \sum_{j\neq i}\lambda_{ij}V(j) - \lambda_i V(i) - \beta V(i) \\ &= f(i) + (QV)_i - \beta V(i) \end{aligned}$$

So we have the system of equations

$$-\begin{pmatrix} f(1) \\ f(2) \\ f(3) \end{pmatrix} = \begin{bmatrix} -\lambda_1 & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & -\lambda_2 & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & -\lambda_3 \end{bmatrix} \begin{pmatrix} V(1) \\ V(2) \\ V(3) \end{pmatrix} - \beta \begin{pmatrix} V(1) \\ V(2) \\ V(3) \end{pmatrix},$$

which in matrix form is

$$-f = (Q-\beta I)V,$$

where $I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ is the identity matrix. Thus, for any reward function f, we can compute the infinite horizon discounted reward via simple linear algebra which is as simple as solving also the long-run averages.

10.3 Finite horizon

Say we start in state 1 and want to know what the performance is over a finite horizon. That is, we collect as before a reward $f(i)$ when in state i but are interested, for some finite time T, in $\mathbb{E}[\int_0^T f(X(t))dt|X(0)=1]$ since, under very mild assumptions,

$$\mathbb{E}[\int_0^T f(X(t))dt|X(0)=1] = \int_0^T \mathbb{E}[f(X(t))|X(0)=1]dt.$$

We therefore need the ability to compute $\mathbb{E}[f(X(t))|X(0)=1]$ for each $t \geq 0$. If we know how to compute the probabilities $\mathbb{P}\{X(t)=j|X(0)=1\}$ then we could compute the expectation

$$\mathbb{E}[f(X(t))|X(0)=1] = \sum_{j=1}^{3} f(j)\mathbb{P}\{X(t)=j|X(0)=1\}.$$

How do we do that? In the discrete time case our life was easier. To compute $\mathbb{P}\{X_n=j|X_0=1\}$ all we had to do was raise the one-step transition matrix to the power of n. In the continuous time chain, the process spends random (exponential) amounts of time in each state and there are just too many ways in which you can go from 1 to j in t time units.

We are going to slowly build up a differential equation whose solution will give us the values of

$$P_{ij}(t) = \mathbb{P}\{X(t)=j|X(0)=i\}.$$

Let's us again focus on our basic three-state example. First, the event that $\{X(t+s)=2\}$, for example, is the union of the exclusive events $\{X(t+s)=2, X(t)=k\}$ with the union taken over $k=1,2,3$; that is,

$$\{X(t+s)=2\} = \bigcup_{k=1}^{3} \{X(t+s)=2, X(s)=k\},$$

so that

$$\begin{aligned}
\mathbb{P}\{X(t+s)=2|X(0)=1\} &= \sum_{k=1}^{3} \mathbb{P}\{X(t+s)=2, X(s)=k|X(0)=1\} \\
&= \sum_{k=1}^{3} \mathbb{P}\{X(t+s)=2|X(s)=k\}\mathbb{P}\{X(s)=k|X(0)=1\} \\
&= \sum_{k=1}^{3} P_{1k}(s)P_{k2}(t).
\end{aligned}$$

In words, this is simply the fact that after s time units we must be in some state k and we are summing over all possible states.

In general we have the following

- **Conclusion 3:**

$$P_{ij}(t+s) = \sum_{\text{states } k} P_{ik}(s)P_{kj}(t).$$

We now put this to use. First, a simple manipulation separating $k \neq i$ from $k=i$ in the sum yields

$$P_{ij}(t+s) = \sum_{\text{states } k \neq i} P_{ik}(s)P_{kj}(t) + P_{ii}(s)P_{ij}(t).$$

Subtracting $P_{ij}(t)$ on both sides

$$P_{ij}(t+s) - P_{ij}(t) = \sum_{\text{states } k \neq i} P_{ik}(s)P_{kj}(t) - (1-P_{ii}(s))P_{ij}(t)$$

Let us divide by s on both sides then we have

$$\frac{P_{ij}(t+s)-P_{ij}(t)}{s} = \sum_{\text{states } k\neq i} \frac{P_{ik}(s)}{s}P_{kj}(t) - \frac{(1-P_{ii}(s))}{s}P_{ij}(t)$$

Notice that what we have on the left hand side, if we take s to approach 0 is the derivative of $P_{ij}(t)$ at t. As s approaches 0

$$\frac{P_{ik}(s)}{s} \to \lambda_{ik} \text{ and } \frac{1-P_{ii}(s)}{s} \to \lambda_i.$$

(WHY?) We have then that, taking s to approach 0 in the equation above:

$$P'_{ij}(t) = \sum_{k\neq i} \lambda_{ik}P_{kj}(t) - \lambda_i P_{ij}(t).$$

For the three state example we have then the system of differential equations

$$\begin{bmatrix} P'_{11}(t) & P'_{12}(t) & P'_{13}(t) \\ P'_{21}(t) & P'_{22}(t) & P'_{23}(t) \\ P'_{31}(t) & P'_{32}(t) & P'_{33}(t) \end{bmatrix} = \begin{bmatrix} -\lambda_1 & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & -\lambda_2 & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & -\lambda_3 \end{bmatrix} \begin{bmatrix} P_{11}(t) & P_{12}(t) & P_{13}(t) \\ P_{21}(t) & P_{22}(t) & P_{23}(t) \\ P_{31}(t) & P_{32}(t) & P_{33}(t) \end{bmatrix}.$$

In matrix notation we have the following for the general case.

> **Conclusion 4:** The transition probabilities $P_{ij}(t) = \mathbb{P}\{X(t) = j | X(0) = i\}$ satisfy the system of differential equations
>
> $$P'(t) = QP(t).$$

We can put these into a mathematical software [1]

Once we have the function $P_{ij}(t)$, we can compute the reward collected at a finite time point t given that we start in state i

$$\mathbb{E}[f(X(t))|X(0)=i] = \sum_j f(j)\mathbb{P}\{X(t)=j|X(0)=i\}.$$

BUT ... solving these can be a mess also for a computer. Notice that for the three state chain we have 9 simultaneous equations. In fact, we know something about how to approximate the solution $P(t)$ by sums of many summands. It turn out that one can write the matrix $P(t) = \{P_{ij}(t)\}$ as the infinite sum

$$P(t) = I + (tQ) + \frac{1}{2!}(tQ)^2 + \frac{1}{3!}(tQ)^3 + ... = \sum_{n=0}^{\infty} \frac{(tQ)^n}{n!}$$

Here tQ is simply the matrix Q multiplied by the time t and $(tQ)^n$ is this matrix raised to the power of n, that is $tQ \times tQ \times ... \times tQ$ n times. In the three-state example tQ is the matrix

$$tQ = \begin{bmatrix} -t\lambda_1 & t\lambda_{12} & t\lambda_{13} \\ t\lambda_{21} & -t\lambda_2 & t\lambda_{23} \\ t\lambda_{31} & t\lambda_{32} & -t\lambda_3 \end{bmatrix}.$$

Thus, if you are satisfied with approximating $P(t)$ you can take N large enough and compute

[1] Python has the odeint command as part of its ScyPy library`https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.integrate.odeint.html`

$$P(t) \approx \sum_{n=0}^{N} \frac{(tQ)^n}{n!},$$

which is not too bad. Here is a try for the three-state example, using a perfectly symmetric case ($\lambda_1 = \lambda_2 = \lambda_3 = 1$ and $P_{ij} = 1/2$ for all i and $j \neq i$). That is,

$$tQ = \begin{bmatrix} -t & t/2 & t/2 \\ t/2 & -t & t/2 \\ t/2 & t/2 & -t \end{bmatrix}.$$

For this case we have some intuition what to expect. Specifically, we expect that for each t the matrix will be symmetric: that is $P_{11}(t) = P_{22}(t) = P_{33}(t)$ as well as $P_{12}(t) = P_{21}(t)$, etc. We also expect that as t grows we will be equally likely to be in any state regardless of the initial condition.

Using the simple code (implemented in Matlab):

```
temp=zeros(3,3);
tQ=[-t,t/2,t/2;t/2,-t,t/2;t/2,t/2,-t];
for i=0:100
temp=temp+Qˆi/factorial(i);
end
return temp
```

the following approximations are obtained:

$t = 0.5$			$t = 2$			$t = 10$		
0.6482	0.1759	0.1759	0.3665	0.3167	0.3167	0.3333	0.3333	0.3333
0.1759	0.6482	0.1759	0.3167	0.3665	0.3167	0.3333	0.3333	0.3333
0.1759	0.1759	0.6482	0.3167	0.3167	0.3665	0.3333	0.3333	0.3333

We can use the approximation for the probabilities to compute an approximation for the reward. For example, for $t = 0.5$

$$\begin{aligned} \mathbb{E}[f(X(0.5))|X(0) = 1] &= f(1)P_{11}(0.5) + f(2)P_{12}(0.5) + f(3)P)13(0.5) \\ &\approx f(1) * 0.6482 + f(2) * 0.1759 + f(3) * 0.1759. \end{aligned}$$

ORIE 5530: Modeling Under Uncertainty

Lecture 11 (The Poisson Process)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

This is a short introduction to the Poisson process. The Poisson process is a very useful way to model inputs (arrivals to services, people browsing the internet, etc.). It is flexible enough to accommodate temporal variation, forecasting uncertainty and other realistic features. Here, we focus on some basics that will allow you to use and simulate a Poisson process. More discussion of modeling, forecast uncertainty, etc. are postponed to a service-operations class.

Chapter 5 of the Ross book covers the Poisson process with a bit more detail than we need but you can get an alternative view there.

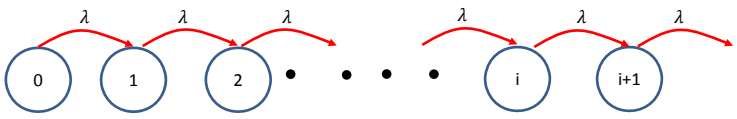
11.1 A Poisson process as a special case of a CTMC

The Poisson process models the following:

- No customers arrive before time 0
- Customers arrive one at a time
- The time between consecutive arrivals is $exp(\lambda)$. With T_i being the time of the i^{th} arrival, this means $T_{i+1} - T_i \sim exp(\lambda)$. Consecutive interarrival times are independent of each other.

  λ is then referred to as *the rate of the Poisson process.*

It is a process that evolves over time. It is, specifically, a continuous time Markov chain that jumps up by 1 every exponential amount of time. In a transition diagram this is



Let's call this Markov Chain $N(t)$. Then, the number of arrivals by time t is the state of this Markov chain at t. There is, of course, a reason that this is called a Poisson process.

Property 1. The number of arrivals by time t, $N(t)$, has a Poisson distribution with mean λt. That is,

$$\mathbb{P}\{N(t) = k\} = \frac{e^{-\lambda t}(\lambda t)^k}{k!}.$$

In particular, $\mathbb{E}[N(t)] = Var(N(t)) = \lambda t$.

Remark. You might ask yourself how to make this consistent with what we learned about finite horizon performance? What is written here is that $P_{0,k}(t) = e^{-\lambda t}(\lambda t)^k/k!$. We can check that it solves the differential equation $P'(t) = QP(t)$. Let us take the first item of this set of equations. On the left we have $P'_{0,k}(t)$ (the derivative of $P_{0,k}(t)$ at time t) and on the right we have the product of the first row of Q with the k^{th} column of $P(t)$. This (check) gives

$$P'_{0,k}(t) = -\lambda P_{0,k}(t) + \lambda P_{1,k}(t).$$

Let us check that our "guess" in property 1 actually satisfies this. With $P_{0,k}(t) = e^{-\lambda t}(\lambda t)^k/k!$ we have that

$$\begin{aligned} P'_{0,k}(t) &= -\lambda \frac{e^{-\lambda t}(\lambda t)^k}{k!} + \lambda \frac{e^{-\lambda t} k(\lambda t)^{k-1}}{k!} \\ &= -\lambda \frac{e^{-\lambda t}(\lambda t)^k}{k!} + \lambda \frac{e^{-\lambda t}(\lambda t)^{k-1}}{(k-1)!} \\ &= -\lambda P_{0,k}(t) + \lambda P_{1,k}(t). \end{aligned}$$

The last line follows because it is clear that in this Markov chain $P_{0,k-1}(t)$ is the same as $P_{1,k}(t)$ (its the probability of $k-1$ arrivals in t time units) and equals $e^{-\lambda t}(\lambda t)^{k-1}/(k-1)!$. Thus, our "guess" solves the necessary differential equation. ∎

Because of the homogeneous nature of this chain (for all states $\lambda_i = \lambda$ and $P_{i,i+1} = 1$) it is clear that

$$\mathbb{P}\{N(t+s) - N(s) = k | N(s) = \ell\} = \mathbb{P}\{N(t) = k\},$$

and does not depend on $N(s)$. In other words, for any interval of length t,

$$\mathbb{P}\{N(t+s) - N(s) = k\} = \frac{e^{-\lambda t}(\lambda t)^k}{k!}.$$

Exercise: People are hired into a workplace according to a Poisson process with rate $\lambda/week$. There is a learning curve. Let $X(t,z)$ be the number of employees at time t that have that have been with the firm for z weeks or more. What is the distribution of $X(t,z)$? ∎

Now, take $t_1 < t_2 \leq t_3 < t_4$ and consider arrivals on the intervals $(t_1, t_2]$ and $(t_3, t_4]$.

Property 2. (**independent increments**) $N(t_4) - N(t_3)$ is independent from $N(t_2) - N(t_1)$ meaning that for every k, ℓ,

$$\begin{aligned} \mathbb{P}\{N(t_4) - N(t_3) = k, N(t_2) - N(t_1) = \ell\} &= \mathbb{P}\{N(t_4) - N(t_3) = k\}\mathbb{P}\{N(t_2) - N(t_1) = \ell\} \\ &= \frac{e^{-\lambda(t_4 - t_3)}(\lambda(t_4 - t_3))^k}{k!} \frac{e^{-\lambda(t_2 - t_1)}(\lambda(t_2 - t_1))^\ell}{\ell!}. \end{aligned}$$

This should be clear again from homogeneity. $N(t_4) - N(t_3)$ is Poisson$(\lambda(t_4 - t_3))$ independently of anything that happens before t_3. In particular, it should not depend on $N(t_2) - N(t_1)$.

Remark 1 (simulation) Simulating a Poisson process is rather straightforward given the simulation you had already constructed in your homework for a more complicated continuous time Markov chains. Here is the basic pseudo code for simulating a Poisson process on $[0, T]$

- Set the time to $T_0 = 0$. Set the counter i to 0
- As long as $T_i \leq T$:
  - Generate an exponential random variable Y_i with parameter λ. (we do this, recall, by generating a uniform U and then computing $ln(U)/\lambda$.)

- Advance $i \leftarrow i+1$ Advance the time to $T_i \leftarrow T_{i-1} + Y_i$.
- Set $N(T_i) = i$.

Now, computers are not very good at keeping functions, such as $N(t)$. They are better at discrete things. So one way to keep this information is to generate a table where one records the times of events and where the Poisson process was (the same applies to any CTMC). Then, if you are asked what is the value of $N(t)$, you can search the table for the interval in which t falls.

Notice that if you are interested in just one specific t (rather than the sample path of the process – how the process evolves) you don't need this. Since $N(t) \sim Poisson(\lambda t)$, you can just simulate one random variable with this distribution.

Below, for example, is data from a simulation run for a Poisson process with rate $\lambda = 1$.

Arrival i	T_i
1	1.637013794
2	3.617584853
3	5.607789129
4	6.274811479
5	8.921637384
6	10.01218765
7	10.64001714
8	11.01940243
9	11.0229644
10	11.32889342
11	11.4233002
12	12.32754224
13	14.11642354
14	14.27324588
15	16.74586619
16	16.75468146
17	16.76373638
18	17.74894105
19	18.6708546
20	19.11551817

■

11.2 Splitting/Thinning a Poisson processes

Think about the following (see Figure 11.1): There is a total input process that follows a Poisson process with rate λ. Each of the customers is of type A with probability p and of type B with the remaining probability $1-p$. Customers are independent of each other for their types. What is the input process to queues A and B?

In total then there are λ customers arriving per hour and a fraction p of them are type A. So, on average, you expect (and you'd be correct) that the number of type A customers per hour is λp. But it turns out we can say more than that. The input to A and the input to B maintain the Poisson structure of the input process and are independent of each other. Let

$$N_A(t) = \#\text{of customers of type } A \text{ that arrived by time } t,$$

$$N_B(t) = \#\text{of customers of type } B \text{ that arrived by time } t.$$

Then, we have

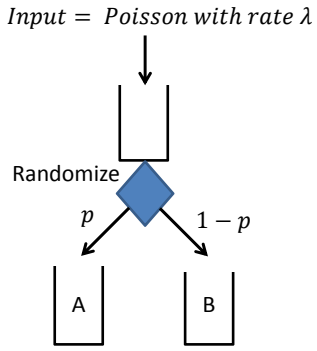


Figure 11.1: Splitting of Poisson process

Property 3. $N_A(t)$ is a Poisson process with rate λp and $N_B(t)$ is a Poisson process with rate $\lambda(1-p)$. In particular,

- $\mathbb{P}\{N_A(t) = k\} = e^{-\lambda pt}(\lambda pt)^k/k!$ and $\mathbb{P}\{N_B(t) = k\} = e^{-\lambda(1-p)t}(\lambda(1-p))^k/k!$
- The time between type A arrivals is exponential with parameter λp (and mean $\frac{1}{\lambda p}$).
- The time between type B arrivals is exponential with parameter $\lambda(1-p)$ (and mean $\frac{1}{\lambda(1-p)}$).

Moreover, the two processes are independent of each other. That is, for each t, k, ℓ

$$\mathbb{P}\{N_A(t) = k, N_B(t) = \ell\} = \mathbb{P}\{N_A(t) = k\}\mathbb{P}\{N_B(t) = \ell\}.$$

This is very useful because it tells us that the probabilistic structure is maintained. This makes it easier to simulate the system to which A and B are inputs. Moreover, we can treat whatever comes after the randomization node as independent processes.

The fact that the time between A-arrivals is exponential with parameters λp and hence mean $1/(\lambda p)$ makes intuitive sense: If we just had a type A customer, the number of customers until we get another one is Geometric with parameter p (because each customer is of type A with probability p), right? On average we have to wait then $1/p$ customers. The mean time between consecutive customers is $1/\lambda$ (the expectation of the exponential) and hence *the expected time between A-arrivals is* $\frac{1}{p}\frac{1}{\lambda} = \frac{1}{\lambda p}$.

But the result goes beyond average time between arrival. It says that this time — a $Geom(p)$ sum of $exp(\lambda)$ random variables — is exponential with parameters λp:

Implication. Let $M \sim Geom(p)$. Let $Y_1, Y_2, \ldots$ be independent $exp(\lambda)$ random variables (that are also independent of M). Then,

$$\sum_{i=1}^{M} Y_i \sim exp(\lambda p).$$

In particular,

$$\mathbb{P}\left\{\sum_{i=1}^{M} Y_i \leq x\right\} = 1 - e^{-(\lambda p)x}.$$

Example 1 (The infinite server queue) This idea of Poisson splitting is kind of powerful. As an example consider the following setting which is called "the infinite server queue". We have the following:

- Customers arrive according to a Poisson process with rate λ
- Customer i stays in the system for a service time S_i that follows a distribution G. That is, $\mathbb{P}\{S_i \leq s\} = G(s)$ and $\mathbb{P}\{S_i > s\} = 1 - G(s)$. We use the notation $\bar{G}(s) = 1 - G(s)$.

What can we say about the number of people in the system at time t. That is, how many people arrived by time t but did not leave the system yet.

Poisson thinning gives us an answer. Let's divide the time $[0, t]$ into buckets of a size h.

Since the arrival process is Poisson, the number of arrivals on $[0, h]$ is Poisson with mean λh. Out of those arriving on this interval, each customer will be still be around at time t if her service time is longer (roughly) than t. The fraction of those is $\mathbb{P}\{S_i > t\} = \bar{G}(t)$. Let's think of those customers as type A in our thinning example. The probability of being A is here $p = \bar{G}(t)$. Then, the number of type A customer arriving over $[0, h]$ is by thinning $N_0 = Poisson(\lambda h \bar{G}(t))$.

We can apply this also to the second time bucket and we will have that the number that arrived over $[h, 2h]$ that are still around at time t is $N_1 = Poisson(\lambda h \bar{G}(t-h))$, for the third $N_2 = Poisson(\lambda h \bar{G}(t-2h))$, and so on.

Now, we are going to add all these Poisson random variables. By *addition* of independent Poisson random variables we have that

$$\sum_{i=1}^{t/h} N_i \sim Poisson(\sum_{k=1}^{t/h} \lambda h \bar{G}(t - kh)).$$

As we shrink h to zero we get

$$\sum_{k=1}^{t/h} \lambda h \bar{G}(t - kh) \to \int_0^t \lambda \bar{G}(s) ds.$$

Thus, **we found that the number of people in the system at time t is a Poisson random variable with parameters (and hence mean) $\lambda \int_0^t \bar{G}(s) ds$.**

Exercise: The doors to a show open at 8:00pm. People start arriving at 6:00pm and lining up. Arrivals follow a Poisson process with rate $50/hour$ until doors open.

Those that arrive at 6:01 will wait 1:59 hours while those that arrive at 7:59 will wait just one minute. What is the expected total waiting time (across all customers). How much will the average customer wait?

ORIE 5530: Modeling Under Uncertainty

Lecture 12 (Dynamic Programming: Finite Horizon)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

In this lecture we embark on a short journey into dynamic optimization. In dynamic optimization (often referred to as dynamic programming) we change our decisions dynamically from one period to the next. This should be contrasted with static optimization where we make one decision, say how much money to put in a closed bank account. Many decisions in practice allow for dynamic adjustment. All airlines and many retailers adjust their prices dynamically in response to changes in the inventory of available seats on a plane or remaining inventory, etc.

12.1 Finite horizon discrete time Dynamic Programming

We start with an example. An "investment-consumption" problem. It is taken from very nice notes by Peter Glynn.

You have a finite number of periods $0, 1, \ldots, n$. At period i, you can consume some of your capital, the rest you can invest in the bank. The money that you put in the bank (that part that you do not consume) makes a return of R_{i+1} between periods i and $i+1$ where $R_1, \ldots, R_n$ are independent random variables. They are not necessarily identically distributed random variables. It could be, for example, that as you get closer to the date in which your account "expires" the returns are larger.

Thus, if you had X_i money at the end of period i and you consumed $\$C_i$, the amount you have available at period $i+1$ is

$$X_{i+1} = (X_i - C_i)R_{i+1}.$$

Of course, there is a reason we consume. We get some utility from it. Let's assume that if you consume $\$C$ you get an immediate utility that is equal to that value. You also have utility from having some money left at the end of the horizon (say, you give a gift to your kids). Let us assume that this terminal utility is just equal to the money you have remaining at the end.

You are trying to maximize your total utility over the horizon.

$$\mathbb{E}\left[\sum_{i=0}^{n-1} C_i + X_n\right]$$

What we are trying to do is **choose the optimal consumption level at each period**.

Let us define

$V_i(x) =$ maximal expected utility with i periods passed (so $n-i$ periods to go) and x available dollars.

The key is to express the value as an optimization problem where we maximize the expected *immediate reward* plus the best we can do in the future given the action we take now (and the state to which it takes us). Here this means we are going to have a recursion of the form

$$V_i(x) = \max_{0 \le c \le x} \{c + \mathbb{E}_x[V_{i+1}(X_{i+1})]\}.$$

Notice that given $X_i = x$ and we take an action c we can replace X_{i+1} with $(x-c)R_{i+1}$, so this recursion reduces here to

$$V_i(x) = \max_{0\le c\le x}\{c + \mathbb{E}[V_{i+1}((x-c)R_{i+1})]\}.$$

If we know what $V_n(x)$ is we could solve for all other periods i and dollars x *backward*. $V_n(x)$ is then the utility if we have 0 steps to go and x dollars so $V_n(x) = x$. There is no choice here. Let's now go one step backward – to step $n-1$. If we have x dollars in the beginning of step $n-1$ and we consume $c \le x$, then we get immediately a utility of c. We will have $(x-c)R_n$ in the last period at which point our utility is just the money we have and hence $(x-c)R_n$ or, in expectation, $\mathbb{E}[(x-c)R_n]$. Thus,

$$V_{n-1}(x) = \max_{0\le c\le x}\{c + \mathbb{E}[(x-c)R_n]\} = \max_{0\le c\le x}\{c + (x-c)\mathbb{E}[R_n]\}.$$

In general, in period i we obtain what we consume and then the expected utility with $i+1$ period passes (or $n-(i+1)$ to go) assuming we take the best possible actions given that we have $(x-c)R_{i+1}$ dollars left. That is,

$$V_i(x) = \max_{0\le c\le x}\{c + \mathbb{E}[V_{i+1}((x-c)R_{i+1})]\}.$$

This then gives us a solution via backward induction. In period $n-1$ we will put everything we have in investment if $\mathbb{E}[R_n] < 1$ in which case $V_{n-1}(x) = x$. Otherwise, we consume $c = 0$ and have $V_{n-1}(x) = x\mathbb{E}[R_n]$:

$$V_{n-1}(x) = \max\{x, x\mathbb{E}[R_n]\} = x\max\{1, \mathbb{E}[R_n]\}.$$

Define the notation $\gamma_n = \max\{1, \mathbb{E}[R_n]\}$. Then, $V_{n-1}(x) = \gamma_n x$. Proceeding by backward induction,

$$V_{n-2}(x) = \max_{0\le c\le x}\{c + \mathbb{E}[V_{n-1}((x-c)R_{n-1})]\} = \max_{0\le x\le c}\{c + \gamma_n\mathbb{E}[(x-c)R_{n-1}]\}.$$

By the same argument we would consume everything if $\gamma_n\mathbb{E}[R_{n-1}] < 1$ and nothing otherwise. Thus,

$$V_{n-2}(x) = \max\{x, \gamma_n\mathbb{E}[xR_{n-1}]\} = x\max\{1, \gamma_n\mathbb{E}[R_{n-1}]\}.$$

Letting $\gamma_{n-1} = \max\{1, \gamma_n\mathbb{E}[R_{n-1}]\}$ we can proceed in this way to get with $\gamma_i = \max\{1, \gamma_{i+1}\mathbb{E}[R_i]\}$ that we will consume all if $\gamma_i < 1$ and invest all otherwise.

Let's pause now to reflect on what we did and generalize this. We have the following elements:

- An expected immediate reward that depends on the action (let's call it δ). In general, it may depend also on the state. So we can write $r(\delta, x)$ for the reward if you take the action δ when in state x. In our example above $r(\delta, x) = \delta$. It is independent of x and the action is the consumption c and that is your reward.
- Constraints on acceptable actions. In the example above the control c could take only values between 0 and x, that is we require that $c \in \Delta(x) = \{y : 0 \le y \le x\}$. In general, we will write this as $\delta \in \Delta(x)$.
- Dynamics that govern the transition between the current state and the next, *given the action we take*. Above we could compute $X_{i+1} = (X_i - C_i)R_{i+1}$. This means for example that, given consumption c,

$$\mathbb{P}_c^i(x,y) = \mathbb{P}_c\{X_{i+1} = y | X_i = x\} = \begin{cases} 1 & \text{if } c = x, y = 0 \\ \mathbb{P}\{R_{i+1} = y/(x-c)\} & \text{otherwise.} \end{cases}$$

  The first row follows since if we invest all our money ($c = x$) the next state will be 0 with probability 1 (does not matter what interest the bank offers, we have no money remaining to

put there). Also, the only relevant transitions are from x to y of the form $(x-c)r$. Then, $\mathbb{E}_x[V_{i+1}(X_{i+1})] = \sum_y \mathbb{P}_c^i(x,y)V_{i+1}(y)$

Notice that here, because we allow R_i to have a different distribution per i, the transition probability depends on i.

- A recursion that says that the optimal decision in this period is to maximize the combination of what we get in this period plus the optimal utility starting at the state we move to:

$$V_i(x) = \max_{0\le c\le x}\{c + \sum_y \mathbb{P}_c^i(x,y)V_{i+1}(y)\}$$

This is the finite horizon *dynamic programming equation.*

This is clearly generalizable. Take any setting where there is

$$\text{a per-period reward function } r(A_i, X_i)$$

(that can depend on the action A_i in period i and on the state X_i). In the example we had $A_i = C_i$ as the consumption and $(X_i - C_i)$ as the investment.

We seek to choose actions $A_1, \ldots, A_n$ to maximize utility. We can change are actions dynamically – we can choose A_{n-1} after we have seen the states up to that time evolving. The utility we seek to maximize is

$$\mathbb{E}[\sum_{i=0}^{n-1} r(A_i, X_i) + u(X_n)],$$

where we allow for a terminal utility $u(X_n)$ that depends on the *terminal* state. We can write the dynamic programming equation

$$V_i(x) = \max_{\delta\in\Delta(x)} \left\{ r(\delta, x) + \sum_y \mathbb{P}_\delta^i(x,y)V_{i+1}(y) \right\}.$$

This might still be more general than needed in some cases. In the example we had, $R_1, R_2, \ldots, R_n$ were independent but did not have to be identically distributed. This is why we allow $\mathbb{P}_\delta$ to depend on the time i. If they were all identically distributed we could have $\mathbb{P}_\delta$ instead of $\mathbb{P}_\delta^i$. Also, we do not necessarily have to have a terminal value for the remaining X_n, that is, $u(X_n) = 0$ can also be the case. We have freedom within this framework.

Finally, notice that for each x in the state space and time i we are going to get from solving this equation a recommended action $\delta_i^*(x)$.

The fundamental result here is that

- This dynamic programming equation has a unique finite-valued solution.
- The action $A_i = \delta^*(X_i)$ is an optimal action at state X_i.

While the result is somewhat intuitive, there is depth to it. It says that we can find optimal actions that do not depend on all the history. We only need the current state and the time.

Example 1: Pricing inventory In this setting we have an initial inventory of x units. Think of this as, for example, the number of seats on a flight. One customer arrives per period. An arriving customer will buy the product (say, a ticket) at price p with probability $\lambda(p)$ (and not buy with probability $1-\lambda(p)$). If we had only one seat to sell we would choose the price that maximizes the expected immediate reward = price*probability of sale = $p * \lambda(p)$. If the arriving customers buys a unit of the product at the offered price then the inventory goes down by 1. Let $B_i(p)$ be a random variable that equals 1 if the i^{th} customer

buys the product when offered a price p. That is, $\mathbb{P}\{B_i(p) = 1\} = \lambda(p)$. We will assume customers are statistically identical (they all have the same probability of buying under a price p). Then, we seek to choose a price strategy (saying what price p_i to offer to the i^{th} customer given the inventory states) to maximize

$$\mathbb{E}_x \left[\sum_{i=1}^{n} p_i B_i(p_i) \right].$$

Here are the ingredients:

- **Immediate expected reward**: Under the action δ, the expected reward is given by
$$r(\delta, x) = \delta \mathbb{E}[B_i(\delta)] = \delta \lambda(\delta).$$
- **Transition Probability:** If we offer the price δ, the transition is
$$\mathbb{P}_\delta(x, x-1) = \mathbb{P}_\delta\{X_{i+1} = x-1 | X_i = x\} = \lambda(\delta),$$
and
$$\mathbb{P}_\delta(x, x) = \mathbb{P}_\delta\{X_{i+1} = x | X_i = x\} = 1 - \lambda(\delta).$$
All other transition have 0 probability.
- **Terminal value:** In the last period no customer arrives and the inventory we have remaining is useless. Thus, $V_{n+1}(x) = 0$ for all x and that serves for our recursion.

The dynamic programming equation is given by

$$V_i(x) = \max_\delta \{\delta\lambda(\delta) + \lambda(\delta)V_{i+1}(x-1) + (1-\lambda(\delta))V_{i+1}(x)\}.$$

To make things concrete, let us assume the following structure. Let us normalize the price to 1 (if you are counting in thousands of dollars, 1 will stand for one thousdand). If the price is 1 nobody buys, and everybody buys at \$0. Things are linear in between. Namely,

$$\lambda(\delta) = (1 - \delta).$$

Then, replacing $\lambda(\delta)$ with $(1-\delta)$, the equation is, **for** $x \geq 1$

$$V_i(x) = \max_\delta \{\delta(1-\delta) + (1-\delta)V_{i+1}(x-1) + \delta V_{i+1}(x)\}.$$

Notice that, given V_{i+1}, we know how to find the optimal δ by differentiating and comparing to 0. Differentiating $\delta(1-\delta) + (1-\delta)V_{i+1}(x-1) + \delta V_{i+1}(x)$ with respect to δ gives $1 - 2\delta + V_{i+1}(x) - V_{i+1}(x-1)$. Comparing to 0 gives

$$\delta^*(i, x) = \frac{1}{2}\left(1 + V_{i+1}(x) - V_{i+1}(x-1)\right).$$

There is a nice interpretation of this as an opportunity-cost correction to the base price of $1/2$. Now if we plug this back into our recursion equation we get the equation, **for** $x \geq 1$,

$$\begin{aligned} V_i(x) &= \delta^*(i,x)(1-\delta^*(i,x)) + (1-\delta^*(i,x))V_{i+1}(x-1) + \delta^*(i,x)V_{i+1}(x) \\ &= \left(\frac{1}{2}(1 + V_{i+1}(x) - V_{i+1}(x-1))\right)^2 + V_{i+1}(x-1). \end{aligned}$$

(Check the last transition yourselves.) So now we have a recursion that we can easily code starting from $V_{n+1}(x) = 0$. **Notice, also, that for** $x = 0$**,** $V_i(0) = 0$ **for all** i**.** See the worksheet in the excel file $FiniteHorizonPricing.xlsx$.

A final comment. What would you do if prices could only take on one of three values, say $1/4, 1/2$ or $3/4$. Then, we cannot solve for the optimal price via differentiation. In the code we will have to compare. Namely, when solving for $V_i(x)$, given $V_{i+1}(x)$ that we have from the backward induction, we will compare the different prices

$$V_i(x) = \max_{\delta \in \{1/4, 1/2, 3/4\}} \{\delta(1-\delta) + (1-\delta)V_{i+1}(x-1) + \delta V_{i+1}(x)\}.$$

We will compute the right hand side for each price. Do a comparison and pick the highest as $\delta^*(i, x)$.

Example 2: The multi-secretary problem. n items are presented one at a time. They have values $Z_1, Z_2, \ldots, Z_n$ that are independent and identically distributed from a discrete distribution over $\{1, 2, 3, 4\}$ with probabilities $\mathbb{P}\{Z_i = k\} = p_k$ for each value k.

Once a value is presented you have to decide whether to take it or not. You cannot go back and correct your actions. You have a finite budget $c < n$. So the total number of items you can take is c out of the n presented to you (if $c \geq n$ there would be no question to solve here — you would just take everything).

Let the remaining budget be c and the value we see in step i be a_i. At each step the policy is $\delta \in \{0, 1\}^4$. Meaning $\delta = (\delta_1, \delta_2, \delta_3, \delta_4)$. Having $\delta_1 = 1$ means that we will take the secretary if his value turns out to be 1, $\delta_2 = 1$ means we take the secretary if his value is 2. If both δ_1 and $\delta_2 = 1$ we will take the secretary if his values are either 1 or 2. You see where this is going. So the control specifies these vectors δ.

The state to track is the amount of budget we have left. The other elements of this are

- Expected immediate reward:
$$r(x, \delta) = \sum_{k=1}^{4} k p_k \delta_k$$
(why is that?)
- Transition: if $\delta_k = 1$, and the secretary turns out to be of type k, then our budget goes down by one. That is,
$$\mathbb{P}_\delta(x, x-1) = \sum_{k=1}^{4} p_k \delta_k.$$

The dynamic programming equation is given by

$$V_i(x) = \max_{\delta \in \{0,1\}^4} \left\{ \sum_{k=1}^{4} k p_k \delta_k + (\sum_{k=1}^{4} p_k \delta_k) V_{i+1}(x-1) + (1 - \sum_{k=1}^{4} p_k \delta_k) V_{i+1}(x) \right\}, \text{ for } x = 1, \ldots, n.$$

And we have the terminal condition, for all budget $c \geq 1$, $V_{n+1}(c) = 0$ (the expected value). This problem is very difficult to solve by hand but relatively easy to code. This is part of your last homework.

ORIE 5530: Modeling Under Uncertainty

Lecture 13 (Infinite Horizon Discounted Dynamic Programmin)

Professor Mark S. Squillante,
Adapted from Professor Itai Gurvich's original notes

First, a few words about static vs. dynamic optimization, building on the pricing and the secretary problems we saw.

Let us start with the secretary problem and think of a linear optimization based heuristic way to obtain the optimal hiring rule. Recall: we are interviewing n secretaries and have a hiring budget of k. There are secretaries of four possible qualities: the quality is k with probability p_k.

Say we solve the dynamic programming problem and have a policy.

Let's call

$$x_i = \text{ the average number of quality-}i\text{ secretaries that the policy hires in expectation.}$$

Then, these four variables x_1, x_2, x_4, x_4 must satisfy

i. $x_1 + x_2 + x_3 + x_4 \leq k$

ii. $x_i \leq np_i =$ expected number of quality-i candidates we will see in n interviews.

iii. The total reward we get from hiring quality-1 candidates is x_1, quality-2 candidates $2x_2$, etc. Thus, the optimal value equals
$$x_1 + 2x_2 + 3x_3 + 4x_4.$$

Pretending that we can freely set these, we have an upper bound equal to the solution of the *linear-optimization problem*

$$\begin{aligned} \max\ & x_1 + 2x_2 + 3x_3 + 4x_4 \\ \text{s.t.}\ & x_1 + x_2 + x_3 + x_4 \leq k, \\ & x_i \leq np_i, \text{ for } i = 1, 2, 3, 4, \\ & x_i \geq 0, \text{ for } i = 1, 2, 3, 4. \end{aligned}$$

Four questions:

- Why is the solution of this an upper bound on the optimal value but not necessarily equal to the optimal value.
- How can you interpret this policy to come up with a dynamic heuristic for hiring candidates?
- Will the heuristic achieve the upper bound value, why no?
- Why would a dynamic policy do better that this heuristic?

A first static analysis gives a good sense, however, into structural properties of good policies as well as an implementable simple heuristic.

Question: What is the static problem for the inventory-pricing problem?

13.1 Infinite horizon discounted dynamic programming

In the last class we saw how finite horizon formulations are solved via backward recursion. In finite horizon dynamic programs the optimal action depends on the time period (on how many steps to go) so there is never rarely policy. Also, they may be difficult to compute if the horizon n is large.

Infinite horizon discounted problems take care of some of these complications and they are also a proxy for finite horizon in that larger weights are put on what happens in the shorter term. By adjusting the discount constant β you can adjust how much you care about the far future. The discount factor β also has a natural monetary interpretation.

In infinite-horizon problems, we seek to find a dynamic and adaptive decision rule that maximizes the collected reward

$$\mathbb{E}_x[\sum_{k=0}^{\infty} \beta^k r(X_k, A_k)],$$

where X_k is the state of the process we are tracking at time k and A_k is the action we take in this period. The evolution of X_k will depend on the actions we take. As before the reward $r(X_k, A_k)$ is allowed to depend on the state and the action. In an inventory problem this might be the amount of inventory we have and how much we order.

Some of the what follows should be familiar. Suppose that the control is fixed and $X_k, k = 0, 1, 2, \ldots$ is a given (finite state space) Markov chain and you are studying

$$V(x) = \mathbb{E}_x[\sum_{k=0}^{\infty} \beta^k r(X_k)].$$

We saw in class that we can compute V by solving the system of equation

$$V(x) = r(x) + \beta \sum_y P(x, y)V(y), \tag{13.1}$$

which has the solution (in vector form) $V = (I - \beta P)^{-1} r$.

Thus, it seems that if the chain is not given but rather it is optimized, a variant of the equation (13.1) should give us a way to find the optimal control. Meaning that if the rewards depend on the control u, and thus the probability transition matrix, we should have that the optimal value satisfies the equations

$$V(x) = \max_{\delta \in \Delta(x)} \left\{ r(x, \delta) + \beta \sum_y P_\delta(x, y)V(y) \right\}. \tag{13.2}$$

In principle this is a so-called **fixed-point** equation. We have V on both sides. How do we solve this thing?

Example 1 *You have c units of inventory. A customer arrives every period. If you price the product at p, the customer buys an item with probability $\lambda(p)$. This the same example we used in the finite-horizon context. Here, the tradeoff is a bit different. If you wait, you are not necessarily going to be stuck with remaining inventory (because there is infinite horizon), BUT selling very far in the future brings a discounted value. You'd rather get the cash now! So you are trying to maximize*

$$V(x) = \mathbb{E}_x[\sum_{i=0}^{\infty} p_i R_i(p_i)].$$

In contrast to the finite horizon setting, notice that here we do not have a dependence of V on the period i. This means we also lose this terminal condition we had which said that $V_n(x) = 0$. The question is how does your price schedule look like if you wish to maximize the infinite horizon discounted reward? ∎

To solve the Bellman equation (13.2), we cannot invert as we did with a fixed control but we will have a rather simple algorithm. Let's return to our pricing example:

$$V(c) = \max_{p\geq 0}\{p\lambda(p) + (1-\lambda(p))V(c) + \lambda(p)(V(c-1))\}.$$

Let us put a concrete structure on the price so that we can actually do something here. Also, to make things interesting let's use here $\lambda(p) = e^{-p}$ instead of $\lambda(p) = 1-p$ that we used in the finite horizon case. Then,

$$\begin{aligned} V(c) &= \max_{p\geq 0}\{pe^{-p} + \beta(1-e^{-p})V(c) + \beta e^{-p}(V(c-1))\} \\ &= \beta V(c) + \max_{p\geq 0}\{pe^{-p} + \beta e^{-p}(V(c-1) - V(c))\}. \end{aligned}$$

How do we maximize the inside. We take a derivative and compare to 0. We get (fill in the details) that

$$p^*(c) = 1 - \beta(V(c-1) - V(c)) = 1 + \beta(V(c) - V(c-1)).$$

This give us a way to get the optimal price **once** we know the value. This expression make intuitive sense – the optimal price is the average valuation (which is 1) plus $\beta(V(c) - V(c-1))$ which is how much we are losing from not having that extra item a period from now. Why does it make sense that we never price below average valuation? Think if you had only one item at what price would you offer it ?

How do we get the function $V(c)$? We know $V(0) = 0$ so we can even do so in excel. Indeed, plugging in the expression $p^*(c)$ for the optimal price we have that

$$\begin{aligned} V(c) &= \beta V(c) + \max_{p\geq 0}\{pe^{-p} + \beta e^{-p}(V(c-1) - V(c))\} \\ &= \beta V(c) + e^{-p^*(c)}(p^*(c) + \beta(V(c-1) - V(c))) \\ &\qquad \text{use } p^*(c) = 1 + \beta(V(c) - V(c-1)) \\ &= \beta V(c) + e^{-p^*(c)} \end{aligned}$$

Plugging back the definition of $p^*(c)$ we have **the point equation**

$$(1-\beta)V(c) = e^{-(1+\beta(V(c)-V(c-1)}.$$

Using $V(0) = 0$ (if there is no inventory, there is no revenue), we can now solve this using even an excel solver. We have variables $V(1), V(2), \ldots$ and equations as above. See the worksheet called "optimization" in the excel file *RM_DP_Example.xlsx*

This is not a useful generalizable procedure because it depends on our ability in this simple example to express the optimal action in a clean form as a function of the value function.

In practice, we will rarely enjoy such simplicity. Instead, we use a procedure called value iteration.

13.2 Value Iteration

We start with a first guess $V^0(x)$ for the value function for each x. The easiest guess is to set $V^0(x) = 0$ for all states x. Then, we improve upon it by solving **for each x in the state space**

$$V^1(x) = \max_{\delta\in\Delta(x)}\{r(\delta, x) + \beta\sum_y P_\delta(x,y)V^0(y)\}.$$

We then continue iterating. For each n, we compute **for all state x**:

$$V^n(x) = \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^{n-1}(y)\}.$$

The beautiful fact is that this converges to the true value; namely, as $n \to \infty$,

$$V^n(x) \to V(x),$$

where V is the real optimal value. Once we have that, we know how to find the optimal policy.

Remark 1 1. You could have better candidates for the initial values $V^0(x)$ sometimes. The algorithm will converge regardless of your guess (that is the powerful result here) but the closer your guess to the true value, the faster the algorithm will converge.

2. *Only if you are curious: Why does it work?* This is in case you are interested in an informal proof of why this works. The formal proof follows something called "the contraction principle" but we can actually see this from a simpler argument.

   Take $V^0(x) = 0$ as we did and notice the following: Since $V^0(x) = 0$ and assuming the reward per $r(\delta, x)$ is always non-negative, we have

   $$V^1(x) = \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^0(y)\} = \max_{\delta \in \Delta(x)} \{r(\delta, x)\} \geq 0 = V^0(x).$$

   So we conclude that

   $$V^1(x) \geq V^0(x) \text{ for all } x.$$

   We are now going to use an induction argument. Suppose that $V^k(x) \geq V^{k-1}(x)$ for all $k \leq n-1$ and consider now the equation for $k = n$:

   $$\begin{aligned} V^n(x) &= \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^{n-1}(y)\} \\ &\geq \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^{n-2}(y)\} \\ &= V^{n-1}(x), \end{aligned}$$

   where the first inequality follows from the induction assumption and the last inequality from the definition of value-iteration by which $V^{n-1}(x) = \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^{n-2}(y)\}$.

   Thus, we conclude, for all n, that

   $$V^n(x) \geq V^{n-1}(x) \text{ for all } x.$$

   You now have to recall a basic fact from analysis that monotone increasing sequences converge to a number or diverge to infinity. If the per-period reward $r(\delta, x)$ is bounded, then the sequence is bounded and hence converges to some function

   $$f(x) = \lim_{n \to \infty} V^n(x).$$

   It is fair for you to ask now why is the limit the solution to the original Bellman (dynamic programming equation) but notice that since both $V^n(x) \to f(x)$ for all x and $V^{n-1}(y) \to f(y)$ as $n \to \infty$ then

   $$\begin{array}{ll} V^n(x) & = \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V^{n-1}(y)\} \\ \downarrow & \qquad\qquad\qquad\qquad\qquad\qquad \downarrow \\ f(x) & = \max_{\delta \in \Delta(x)} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) f(y)\} \end{array}.$$

   Hence, f solves the dynamic programming equation and $f = V$.

■

Let's again return to the single-product pricing example. We have reached the dynamic-programming (Bellman) equation:

$$V(c) = \beta V(c) + \max_{p \geq 0}\{pe^{-p} + \beta e^{-p}(V(c-1) - V(c))\}.$$

The value iteration is:

$$V^n(c) = \beta V^{n-1}(c) + \max_{p \geq 0}\{pe^{-p} + \beta e^{-p}(V^{n-1}(c-1) - V^{n-1}(c))\}.$$

Following similar derivation to what we did above, the optimal p **in the** n^{th} **iteration** is

$$p^{*,n}(c) = 1 + \beta(V^{n-1}(c) - V^{n-1}(c-1)),$$

and we can plug this back in to get

$$\begin{aligned} V^n(c) &= \beta V^{n-1}(c) + e^{-p^{*,n}(c)} \\ &= \beta V^{n-1}(c) + e^{-(1+\beta(V^{n-1}(c) - V^{n-1}(c-1)))}. \end{aligned}$$

Value iteration will look like this:

0. Take $V^0(c) = 0$ for all c.
1. Take $p^{*,1}(c) = 1 + \beta(V^0(c) - V^0(c-1)) = 1$ and
$$V^1(c) = e^{-1}.$$
2. Repeat step 1 for all n. For example, given $V^1(c)$ we compute the price $p^{*,2}(c) = 1 + \beta(V^1(c) - V^1(c-1))$ and compute
$$V^2(c) = \beta V^1(c) + e^{-(1+\beta(V^1(c) - V^1(c-1)))}.$$

See the worksheet "value iteration" in the file *RM_DP_Example.xlsx*.

In general, you will not be able to solve for the optimal equation in a clean way from the equation. You might actually have to solve an optimization problem for each n and x. Things get a bit simpler when you have a discrete set of possible actions – in that case you just have to compare values; see HW5 problem 2.

13.3 Synthesis of optimal policy

Value iteration gives you the optimal value $V(x)$ but **does not give you the optimal action**. The prices $p^{*,n}(c)$ that we computed while iterating are just instruments to find the optimal value. By themselves they do not mean anything. We want the optimal price that arises once we have the true value function $V(x)$. In the example above, we were able to identify in closed form

$$p^*(c) = 1 + \beta(V(c) - V(c-1)).$$

Once your algorithm converges and you have the value function V you can compute the optimal prices.

More generally, we recall that optimal actions are the maximizers in the equation

$$V(x) = \max_{\delta \in \Delta(x)}\{r(\delta, x) + \beta \sum_y P_\delta(x, y)V(y)\}.$$

Since we already have V from value iteration and do not need to solve for it, the optimal actions are the maximizers:

$$\delta^*(x) = \underset{\delta \in \Delta(x)}{\operatorname{argmax}} \{r(\delta, x) + \beta \sum_y P_\delta(x, y) V(y)\},$$

where $\delta^*(x)$ is a function from the state x to the best action. In the case that the set of actions is discrete and finite, we can plug in different actions and see which one returns the maximal value.

If the state space is discrete (like the level of inventory in our example) we can finally **represent** $\delta^*(x)$ **by a table** that specifies for each x what is the optimal action.

Here is a full matlab code for this problem where c is allowed to be as high as 50.

```
beta=0.95;

v=zeros(50);
num˙iter = 100;
for k=1:num˙iter
v(1)=beta* v(1)+exp(-(1+beta*v(1)));
for c=2:50
v(c)=beta*v(c) +exp(-(1+(beta*v(c)-v(c-1))));
end
k=k+1;
end
v this output v
```

policy synthesis

```
price=zeros(50);
price(1)= 1+beta*v(1);
for c=2:50
price(c)=1+beta*(v(c)-v(c-1));
end
delta this outputs the optimal price
.....
```

Using $\beta = 0.95$ gives for $c = 1$, 1.6 which is consistent with the excel file. The optimal price it gives for $c = 1$ for example is 2.52.