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.

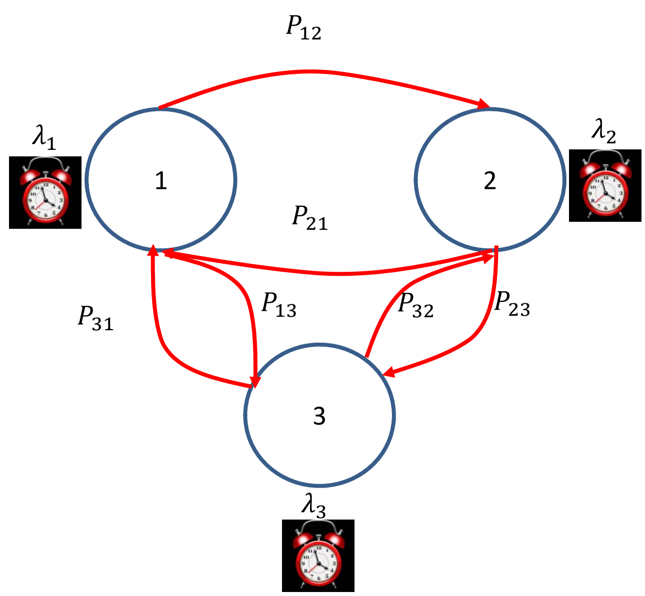

Figure 10.1: A 3-state continuous time Markov chain

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:

$t = 0.5$		
0.6482	0.1759	0.1759
0.1759	0.6482	0.1759
0.1759	0.1759	0.6482

$t = 2$		
0.3665	0.3167	0.3167
0.3167	0.3665	0.3167
0.3167	0.3167	0.3665

$t = 10$		
0.3333	0.3333	0.3333
0.3333	0.3333	0.3333
0.3333	0.3333	0.3333

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

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