1 Simulating Markov chains

Many stochastic processes used for the modeling of systems in engineering are *Markovian*, and this makes it relatively easy to simulate from them. Recall, for example, the Binomial Lattice Model for risky assets (stock), or the FIFO delay sequence for the GI/GI/1 queue; those are Markov chains.

Here we present a brief introduction to the simulation of Markov chains in general. Our emphasis is on discrete-state chains both in discrete and continuous time, but some examples with a general state space will be discussed too.

1.1 Definition of a Markov chain

We shall assume that the state space $\mathcal{S}$ of our Markov chain is $\mathcal{S} = \mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, the integers, or a proper subset of the integers. Typical examples are $\mathcal{S} = \mathbb{N} = \{0, 1, 2 \ldots\}$, the non-negative integers, or $\mathcal{S} = \{0, 1, 2 \ldots, a\}$, or $\mathcal{S} = \{-b, \ldots, 0, 1, 2 \ldots, a\}$ for some integers $a, b > 0$, in which case the state space is finite.

Definition 1.1 *A stochastic process $\{X_n : n \geq 0\}$ is called a Markov chain if for all times $n \geq 0$ and all states $i_0, \ldots, i, j \in \mathcal{S}$,*

$$(1) \qquad \begin{aligned} P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) &= P(X_{n+1} = j | X_n = i) \\ &= P_{ij}. \end{aligned}$$

P_{ij} denotes the probability that the chain, whenever in state i, moves next (one unit of time later) into state j, and is referred to as a *one-step transition probability.* The square matrix $\mathbf{P} = (P_{ij}),\ i, j \in \mathcal{S}$, is called the *one-step transition matrix*, and since when leaving state i the chain must move to one of the states $j \in \mathcal{S}$, *each row sums to one (e.g., forms a probability distribution):* For each i

$$\sum_{j \in \mathcal{S}} P_{ij} = 1.$$

We are assuming that the transition probabilities do not depend on the time n, and so, in particular, using $n = 0$ in (1) yields

$$P_{ij} = P(X_1 = j | X_0 = i).$$

(Formally we are considering only *time homogenous* MC's meaning that their transition probabilities are time-homogenous (*time stationary*).)

The defining property (1) can be described in words as *the future is independent of the past given the present state.* Letting n be the present time, the future after time n is $\{X_{n+1}, X_{n+2}, \ldots\}$, the present state is X_n, and the past is $\{X_0, \ldots, X_{n-1}\}$. If the value $X_n = i$ is known, then the future evolution of the chain only depends (at most) on i, in that it is stochastically independent of the past values $X_{n-1}, \ldots, X_0$.

Markov Property:

> Conditional on the rv X_n, the future sequence of rvs $\{X_{n+1}, X_{n+2}, \ldots\}$ is independent of the past sequence of rvs $\{X_0, \ldots, X_{n-1}\}$.

The defining Markov property above does not require that the state space be discrete, and in general such a process possessing the Markov property is called a *Markov chain* or *Markov process.*

Remark 1.1 A Markov chain with non-stationary transition probabilities is allowed to have a different transition matrix $\mathbf{P}_n$, for each time n. This means that given the present state X_n and the present time n, the future only depends (at most) on (n, X_n) and is independent of the past.

Simulation of a two-state Markov chain

The general method of Markov chain simulation is easily learned by first looking at the simplest case, that of a two-state chain. So consider a Markov chain $\{X_n : n \geq 0\}$ with only two states, $\mathcal{S} = \{0, 1\}$, and transition matrix

$$\mathbf{P} = \begin{pmatrix} 0.30 & 0.70 \\ 0.50 & 0.50 \end{pmatrix}.$$

Suppose that $X_0 = 0$, and we wish to simulate X_1. We only need to generate a rv X distributed as the first row of $\mathbf{P}$, $P(X = 0) = P_{0,0} = 0.30,\ P(X = 1) = P_{0,1} = 0.70$, and set $X_1 = X$. To generate such an X we use the discrete inverse transform method: Generate U. Set $X = 0$, if $U \leq 0.30$; set $X = 1$, if $U > 0.30$. Now that we have X_1, we can simulate X_2 as follows: If $X_1 = 0$, then independently generate X just as before, $P(X = 0) = 0.30,\ P(X = 1) = 0.70$, and set $X_2 = X$; otherwise if $X = 1$, independently generate X according to the second row of $\mathbf{P}$, $P(X = 0) = P_{1,0} = 0.50,\ P(X = 1) = P_{1,1} = 0.50$, and set $X_2 = X$. In general, once we have simulated X_n, we simulate X_{n+1} as follows: If $X_n = i$, then (independently), generate X as $P(X = 0) = P_{i,0},\ P(X = 1) = P_{i,1}$ via set $X = 0$, if $U \leq P_{i,0}$; set $X = 1$, if $U > P_{i,0}$ and set $X_{n+1} = X$.

In the end we see that to sequentially simulate the first n steps, $X_1, \ldots, X_n$, we only need n iid uniforms $U_1, \ldots U_n$.

General algorithm

For a Markov chain with transition matrix $\mathbf{P} = (P_{ij}),\ i, j \in \mathcal{S}$, let Y_i denote a rv distributed as the i^{th} row of the matrix, that is,

$$P(Y_i = j) = P_{i,j},\ j \in \mathcal{S}. \tag{2}$$

Let us assume we use inversion to generate such a Y_i. For example, if $\mathcal{S} = \{0, 1, 2, \ldots\}$, then we generate Y_i via generating a U and setting

1. $Y_i = 0$, if $U \leq P_{i,0}$;
2. $Y_i = j$, if
$$\sum_{k=0}^{j-1} P_{i,k} < U \leq \sum_{k=0}^{j} P_{i,k},\ j \geq 1.$$

In the following algorithm, whenever we say "generate a Y_i", we mean doing so using this inverse transform method using an independent uniform.

Algorithm for simulating a Markov chain up to the first N steps:

1. Choose an initial value, $X_0 = i$. Set $n = 1$
2. Generate Y_i, and set $X_1 = Y_i$.
3. If $n < N$, then set $i = X_n$, generate Y_i, set $n = n + 1$ and set $X_n = Y_i$; otherwise output $(X_1, \dots, X_N)$ and stop.
4. Go back to 3.

Examples of Markov chains

1. *Random walk:* Let $\{\Delta_n : n \geq 1\}$ denote any iid sequence (called the *increments*), and define

   $$X_n \overset{\text{def}}{=} \Delta_1 + \cdots + \Delta_n, \; X_0 = 0. \tag{3}$$

   The Markov property follows since $X_{n+1} = X_n + \Delta_{n+1}, \; n \geq 0$ which asserts that the future, given the present state, only depends on the present state X_n and an independent (of the past) r.v. Δ_{n+1}. Such a chain is easily simulated by sequentially generating the increments and using the recursion $X_{n+1} = X_n + \Delta_{n+1}$.

   When $P(\Delta = 1) = p, \; P(\Delta = -1) = 1-p$, then the random walk is called a *simple random walk*. When $p = 1/2$ the process is called the simple *symetric* random walk. Since the chain can only go up or down by 1 at each step, we see that $P_{i,i+1} = p$, $P_{i,i-1} = 1 - p$ and all other transition probabilities are zero.

   More generally, if the increment distribution is discrete with probability mass function $P(\Delta = j) = q(j), \; j \in \mathbb{Z}$, then $P_{i,j} = P(\Delta = j - i) = q(j - i)$.

   Requiring that $X_0 = 0$ is not necessary, we can start with any deterministic state $X_0 = i$ in which case the process is called a random walk started from state i, and is constructed via $X_n = i + \Delta_1 + \cdots + \Delta_n, \; n \geq 1$.

   Random walks are fundamental building blocks for many stochastic processes and they even lead to the construction of Brownian motion, as a limit as the step size gets smaller and smaller in time while the number of steps gets larger and larger.

2. *State-dependent random walk:*

   Instead of iid increments, suppose that whenever $R_n = i \in \mathbb{Z}$, then the increment distribution is a discrete distribution on the integers, that depends on the state i, with probability mass function $q_i(j), \; j \in \mathbb{Z}$. This means that if $R_n = i$, then $R_{n+1} = i + j$ with probability $q_i(j)$ and so $P_{i,j} = P(R_{n+1} = j \mid R_n = i) = q_i(j - i)$.

3. *Reflected random walk:* In this case, the random walk is not allowed to become negative:

   $$X_{n+1} = (X_n + \Delta_n)^+, \; n \geq 0, \tag{4}$$

   where $x^+ = \max\{0, x\}$ denotes the positive part of x. We assume that the Δ_n are iid with distribution $F(x) = P(\Delta \leq x), \; x \in \mathbf{R}$. If F is a discrete distribution with probability mass function $q(i) = P(\Delta = i), \; i \in \mathbb{Z}$, then we can compute the transition probabilities as follows: the state space is $\mathcal{S} = \mathbb{N} = \{0, 1, 2 \dots\}$, and $P_{i,j} = P(X_1 = j \mid X_0 = i) = P((i + \Delta)^+ = j))$.

When $j > 0$ the transitions are the same as for a regular (non-reflected) random walk:
$P_{i,j} = P((i + \Delta)^+ = j))$
$= P(i + \Delta = j)$
$= P(\Delta = j - i) = q(j - i)$.
Otherwise, $j = 0$: $P_{i,0} = P((i + \Delta)^+ = 0))$
$= P(i + \Delta \leq 0)$
$= P(\Delta \leq -i) = \sum_{k \leq -i} q(k) = F(-i)$.

A special case of this example is FIFO customer delay $\{D_n\}$ for a GI/GI/1 queue, in which case (4) takes the form

$$D_{n+1} = (D_n + S_n - T_n)^+,\ n \geq 0,$$

where S_n denotes the service of the n^{th} customer, and $T_n = t_{n+1} - t_n$ denotes the inter-arrival time between customers n and $n+1$. D_n then denotes the length of time that the n^{th} customer waits in the line (queue) before entering service. It is assumed that both $\{S_n\}$ and $\{T_n\}$ are iid sequences independent of one another. $\Delta_n = S_n - T_n$ then forms an iid sequence.

1.2 Markov chains as recursions

Let $f(x, v)$ be a real-valued function of two variables and let $\{V_n : n \geq 0\}$ be an iid sequence of random variables. We let V denote a typical such random variable.

Then the recursion

$$X_{n+1} = f(X_n, V_n),\ n \geq 0,$$

defines a Markov chain. (We of course must specify X_0, making sure it is chosen independent of the sequence $\{V_n : n \geq 0\}$.)

That this is so is immediate almost by definition: Given $X_n = x$, $X_{n+1} = f(x, V_n)$ only depends on x and some completely independent (of the past) random variable V_n; hence the Markov property holds.

Recursions make it very easy to simulate from: choose an initial value, X_0, then sequentially generate a V_n and set $X_{n+1} = f(X_n, V_n)$. We only need to be able to generate the iid V_n.

In the discrete state case, the transition probabilities from a recursively defined MC are determined via $P_{ij} = P(f(i, V) = j)$.

Proposition 1.1 *Every Markov chain can in fact be represented in the form of a recursion*

$$X_{n+1} = f(X_n, V_n),\ n \geq 0,$$

for some function $f = f(x, v)$ and an iid sequence $\{V_n\}$. In fact $\{V_n\}$ can be chosen as an iid sequence of uniforms on $(0, 1)$, $\{U_n\}$.

In the case when the chain is discrete-valued, with transition matrix $P = (P_{ij})$ the proof is a consequence of the inverse transform method and our general algorithm above for simulating a Markov chain: Letting F_i denote the cdf of the i^{th} row of the transition matrix and $F_i^{-1}(y) = \inf\{x : F(x) \geq y\}$ its generalized inverse function, define $f(i, u) = F_i^{-1}(u),\ i \in \mathbb{Z},\ u \in (0, 1)$; we have our desired f. The point is that if we want to generate a rv Y_i distributed as the i^{th} row of P, we can do so by setting $Y_i = f(i, U) = F_i^{-1}(U)$.

1.2.1 Recursive Examples

Here we illustrate Proposition 1.1 with some examples.

1. *Random walk:* The random walk with iid increments $\{\Delta_n : n \geq 1\}$, defined in (3) was already seen to be in recusive form, $X_{n+1} = X_n + \Delta_{n+1}$. Letting $V_n = \Delta_{n+1},\ n \geq 0$, $f(x, v) = x + v$ is the desired function. Thus $P_{ij} = P(i + \Delta = j) = P(\Delta = j - i)$.

   Letting F^{-1} denote the generalized inverse of $F(x) = P(\Delta \leq x)$, allows us to use uniforms and obtain the recursion $X_{n+1} = X_n + F^{-1}(U_n)$; $f(x, u) = x + F^{-1}(u)$.

2. *Binomial lattice model (BLM):* $S_n = S_0 Y_1 \times \cdots \times Y_n$, where the Y_i are iid distributed as $P(Y = u) = p,\ P(Y = d) = 1 - p$, where $0 < d < 1 + r < u$, with r the risk-free interest rate. In recursive form, $S_{n+1} = S_n Y_{n+1}$, which letting $V_n = Y_{n+1}$ leads to the recursion, $S_{n+1} = S_n V_n$, and $f(x, v) = xv$.

   Here the state space depends on the initial state S_0 and is given by the *lattice* of points,

   $$\mathcal{S} = \{S_0 u^k d^m \ : \ k \geq 0,\ m \geq 0\}.$$

   Since Y can be generated via $Y = uI\{U \leq p\} + dI\{U > p\}$, we can write the recursion using uniforms with the function $f(x, u) = x[uI\{u \leq p\} + dI\{u > p\}]$.

3. *Max and Min of iid sequences:* For $\{Y_n : n \geq 0\}$ any iid sequence, both $M_n = \max\{Y_0, \ldots, Y_n\}$ and $m_n = \min\{Y_0, \ldots, Y_n\}$ are Markov chains: $V_n = Y_{n+1}$ and $f(x, v) = \max(x, v)$, $f(x, v) = \min(x, v)$ respectively yields the desired recursive representation.

We now compute the transition probabilities for M_n above. Suppose that $j > i$. Then $P_{ij} = P(M_{n+1} = j | M_n = i) = P(\max(i, Y_{n+1}) = j) = P(Y = j)$, (where Y denotes a typical Y_n). Note that if $j < i$, then $P(M_{n+1} = j | M_n = i) = 0$ since the maximum can never decrease in value.

Finally, if $j = i$, then $P(M_{n+1} = i | M_n = i) = P(Y \leq i)$; the maximum remains constant at its current value i if the next Y value is less than or equal to i. A similar analysis yields the transition probabilities for m_n.

1.3 Markov chains in continuous time

Consider a discrete-time discrete space Markov chain, but suppose now that whenever it enters state $i \in \mathcal{S}$, independent of the past, the length of time spent in state i is a continuous, strictly positive random variable H_i called the *holding time* in state i, which we assume has an exponential distribution at rate a_i. When the holding time ends, the process then makes a transition into state j according to transition probability P_{ij}, independent of the past, and so on. $P_{ii} > 0$ is allowed, meaning that a transition back into state i from state i can ocurr. Each time this happens though, a new H_i, independent of the past, determines the new length of time spent in state i. Letting $X(t)$ denote the state at time t, we end up with a continuous-time stochastic process $\{X(t) : t \geq 0\}$ with state space $\mathcal{S}$ that has piecewise constant sample paths. It is easily deduced (because of the memoryless property of the exponential distribution) that this process satisfies the Markov property, *the future,* $\{X(s+t) : t \geq 0\}$*, given the present state,* $X(s)$*, is independent of the past,* $\{X(u) : 0 \leq u < s\}$.

The formal definition is given by

Definition 1.2 *A stochastic process* $\{X(t) : t \geq 0\}$ *is called a continuous-time Markov chain (CTMC) if for all* $t \geq 0,\ s \geq 0,\ i \in \mathcal{S},\ j \in \mathcal{S}$,

$$P(X(s + t) = j | X(s) = i, \{X(u) : 0 \leq u < s\}) = P(X(s + t) = j | X(s) = i) = P_{ij}(t).$$

$P_{ij}(t)$ represents the probability that the chain will be in state j, t time units from now, given it is in state i now.

Letting X_n denote the state *right after* the n^{th} transition, yields the underlying discrete-time Markov chain, called the *embedded Markov chain*; it moves according to a transition matrix $P = (P_{ij})$.

Thus a CTMC can simply be described by a transition matrix $P = (P_{ij})$, describing how the chain changes state step-by-step at transition epochs, together with a set of rates $\{a_i : i \in \mathcal{S}\}$, the holding time rates. Each time state i is visited, the chain spends, on average, $E(H_i) = 1/a_i$ units of time there before moving on.

One of the simplest cases of a CTMC that we aleady have learned to simulate from is a Poisson counting process at rate λ, a non-negative process, in which case $a_i = \lambda,\ i \geq 0$ and $P_{i,i+1} = 1,\ i \geq 0$.

In any case, we already know how to simulate a MC $\{X_n : n \geq 0\}$, and we already know how to generate exponential rvs. Putting this together yields (recall the definition of Y_i in (2)):

Algorithm for simulating a continuous-time Markov chain up to time $t = T$:

1. Choose an initial value, $X_0 = i$. Set $n = 0$ and $t = t_0 = 0$.
2. Generate $H_i \sim exp(a_i)$, set $t = t_1 = H_i$.
3. If $t < T$, then set $i = X_n$, generate Y_i, then set $n = n + 1$, $i = Y_i$, $X_n = i$, and generate $H_i \sim exp(a_i)$ and set $t = t + H_i$, $t_n = t$; otherwise set $N = n$ and output $(t_1, \ldots, t_N)$ and $(X_1, \ldots, X_N)$ and stop.
4. Go back to 3.

Letting $N(t) = \max\{n : t_n \leq T\}$ denote the number of transitions during $(0, T]$, note that the above algorithm generates all the values of $X_n,\ 0 \leq n \leq N(T)$, and the corresponding times $t_1, \ldots, t_{N(T)}$ at which the chain makes the transitions. Thus one could even graph the entire trajectory $\{X(t) : 0 \leq t \leq T\}$.

1.4 Semi-Markov processes

It is easily deduced that if the holding times H_i do not have an exponential distribution, then the resulting process $\{X(t) : t \geq 0\}$ will not in general possess the Markov property; it will not be a CTMC. Instead it is called a *semi-Markov* process: It makes transitions according to a discrete-time MC, but upon entering state i, it remains there, independent of the past, for an amount of time H_i with a general distribution F_i. Simulating such a process is as easy as simulating a CTMC, as long as we can easily generate from each F_i. Here is the algorithm:

Algorithm for simulating a semi-Markov process up to time $t = T$:

1. Choose an initial value, $X_0 = i$. Set $n = 0$ and $t = t_0 = 0$.
2. Generate $H_i \sim F_i$, set $t = t_1 = H_i$.
3. If $t < T$, then set $i = X_n$, generate Y_i, then set $n = n + 1$, $i = Y_i$, $X_n = i$, and generate $H_i \sim F_i$ and set $t = t + H_i$, $t_n = t$; otherwise set $N = n$ and output $(t_1, \ldots, t_N)$ and $(X_1, \ldots, X_N)$ and stop.
4. Go back to 3.

1.5 Other Markov processes

The defining Markov property *the future, $\{X(s+t) : t \geq 0\}$, given the present state, $X(s)$, is independent of the past, $\{X(u) : 0 \leq u < s\}$*, holds for Brownian motion (BM) because of the stationary and independent increments: $X(s+t) = X(s) + (X(t) - X(s))$ and $X(s)$ is independent of $(X(t) - X(s))$ with a $N(\mu(t-s), \sigma^2(t-s))$ distribution. This is why it is so easy to sequentially simulate BM at a sequence of points $0 < t_1 < t_2 < \cdots < t_k$. BM is an example of what is called a continuous-time *Markov process* (CTMP).

If $\{X_n : n \geq 0\}$ (or $\{X(t) : t \geq 0\}$) is a Markov process then the process defined by $Y_n = f(X_n)$ (or $Y(t) = f(X(t))$) is also a Markov process if the function $f : \mathcal{S} \to \mathbf{R}$ is a bijection onto its image, that is, if it is invertible. The point is that $X_n = i$ if and only if $f(X_n) = f(i)$. For example $f(x) = e^x$ is a such a function, $f^{-1}(y) = e^{-y}$. This is another way to see that geometric BM is a Markov process too: $S(t) = S_0 e^{X(t)}$ where $\{X(t) : t \geq 0\}$ is BM. We previously knew it was Markovian since it satisfies the nice recursion: for any $0 \leq s < t$, $S(t) = S(s)e^{X(t)-X(s)} = S_0 e^{X(s)} \times e^{X(t)-X(s)}$. This is why it is so easy to sequentially simulate GBM at a sequence of points $0 < t_1 < t_2 < \cdots < t_k$. In short, the recursive nature of Markov processes lends itself nicely to the simulation of such processes.

Finally, a Markov process need not be one-dimensional. For example, random walks can be defined in $\mathbf{R}^m$, as can BM. Once again their recursive properties make simulations from them easy.

1.6 Markov chain Monte Carlo simulation

Because Markov chains are relatively easy to simulate from, they can be used in very clever ways to sample from an a priori unknown (and very complicated) distribution. The general idea: Suppose we wish to generate a sample from a probability mass function $q(i)$, $i \in \mathcal{S}$. Further suppose that we do not have an explicit formula for the $q(i)$, we might for example know the values $a_i = cq(i)$ where c is the normalizing constant $c = \sum_{i \in \mathcal{S}} a_i$, but computing c is not possible a priori. Now suppose we could define a positive recurrent irreducible and aperiodic Markov chain, $\{X_n : n \geq 0\}$ with transition matrix $\mathbf{P}$ such that q is the stationary (limiting as $n \to \infty$) distribution, that is, it is the unique probability solution to $\pi = \pi\mathbf{P}$, and satisfies $\pi_i = \lim_{n\to\infty} P(X_n = i)$, $i \in \mathcal{S}$. Then we could simulate the chain out to (a very large) time n and use X_n as being (approximately) distributed as q: For n large, $q(i) \approx P(X_n = i)$. Such methods are particularly useful when the probability mass function q is multi-dimensional, with a complicated joint distribution structure. "Monte Carlo" simulation would arise in this context if we needed to generate iid copies from the desired distribution q; we would simulate the chain, out to time n, over and over again, to obtain the copies.

We will go into this method in more detail at a future time.