Reading Course

Ulm University
Institute of Stochastics

Lecture Notes
Dr. Tim Brereton
Winter Term 2015 - 2016

Ulm, 2015

Contents

Chapter 1

The Poisson Process

The *Poisson process* is one of the fundamental stochastic processes in probability. We can use it to build many complex and interesting stochastic objects. It is a very simple model for processes such as the arrival of people in a queue, the number of cars arriving at a red traffic light, and the occurrence of insurance claims.

1.1 Point Processes on $[0, \infty)$

A Poisson process is a *point process* on $[0, \infty)$. We will not consider such point processes in their most general form, but rather a straightforward and easy to work with subset of point processes. We can use a number of equivalent definitions. The first of these is to see the point process in terms of a counting process. A counting process, $\{N_t\}_{t\geq 0}$ counts the number of events that have happened by time t.

Definition 1.1.1 (Point Process on $[0, \infty)$: Counting Process Version)**.** A point process on $[0, \infty)$ is a process $\{N_t\}_{t\in[0,\infty)}$ taking values in $\mathbb{N}$ that:

(i) Vanishes at 0. That is, $N_{0^-} = N_0 = 0$, where $N_{t^-} = \lim_{u\uparrow t} N_u$.

(ii) Is non-decreasing. That is, $N_0 \leq N_s \leq N_t$ for $0 \leq s \leq t$.

(iii) Is right continuous. That is, $N_t = N_{t^+}$, where $N_{t^+} = \lim_{u\downarrow t} N_u$.

(iv) Has unit jumps. That is, $N_t - N_{t^-} \in 0, 1$. Technically, a process where only one jump can occur at a given time t is called a *simple point process*.

(v) Has an infinite limit. That is, $\lim_{t\to\infty} N_t = \infty$.

Note, again, that some of these requirements can be relaxed.

Definition 1.1.2 (Point Process on $[0, \infty)$: Jump Instances Version)**.** A point process on $[0, \infty)$ is defined by a sequence $\{T_n\}_{n\geq 1}$ of random variables that are positive and increasing to infinity. That is,

$$0 < T_1 < T_2 < \cdots < \infty \quad \text{and} \quad \lim_{n\to\infty} T_n = \infty.$$

This defines the counting process

$$N_t = \sum_{n\geq 1} \mathbb{I}(T_n \leq t) = \sup\{n \geq 1 : T_n \leq t\}.$$

Definition 1.1.3 (Point Process on $[0, \infty)$: Inter-arrivals Version)**.** A point process on $[0, \infty)$ is defined by a sequence $\{S_n\}_{n\geq 1}$ of positive random variables such that $\sum_{n\geq 1} S_n = \infty$. These define a sequence of jump instances by $S_1 = T_1$ and $S_n = T_n - T_{n-1}$ for $n \geq 2$.

These three definitions of point processes suggest three possible ways to simulate them.

(i) We can simulate the counting process $\{N_t\}_{t\geq 0}$ (or its increments).

(ii) We can simulate the jump times $\{T_n\}_{n\geq 1}$.

(iii) We can simulate the inter-arrival times $\{S_n\}_{n\geq 1}$.

The key properties of a Poisson process (aside from being a point process) are that it has *stationary increments* and *independent increments*.

Definition 1.1.4 (Stationary Increments)**.** A stochastic process $\{X_t\}_{t\geq 0}$ has stationary increments if the distribution of $X_{t+h} - X_t$ depends only on h for $h \geq 0$.

Definition 1.1.5 (Independent Increments)**.** A stochastic process $\{X_t\}_{t\geq 0}$ has independent increments if the random variables $\{X_{t_{i+1}} - X_{t_i}\}_{i=1}^n$ are independent whenever $0 \leq t_1 < t_2 < \cdots < t_n$ and $n \geq 1$.

A process that has stationary and independent increments is attractive from a simulation perspective because we can simulate it in 'parts' (the increments).

1.2 Poisson Process

Equipped with the necessary definitions, we can now define a Poisson process.

Definition 1.2.1 (Poisson Process on $[0,\infty)$)**.** A (homogenous) Poisson process on $[0,\infty)$ with parameter $\lambda > 0$ is a point process on $[0,\infty)$ with stationary and independent increments and $N_t \sim \mathsf{Poi}(\lambda t)$ for all $t \geq 0$. That is,

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

Actually, we can define a Poisson process in a number of different ways.

Theorem 1.2.2 (Poisson process)**.** The following definitions are equivalent definitions.

(i) A Poisson process is a point process, $\{N_t\}_{t\geq 0}$, with stationary and independent increments with $N_t \sim \mathsf{Poi}(\lambda t)$ for all $t \geq 0$.

(ii) A Poisson process is a point process, $\{N_t\}_{t\geq 0}$, with independent increments and the property that, as $h \downarrow 0$, uniformly in t

  (a) $\mathbb{P}(N_{t+h} - N_t = 0) = 1 - \lambda h + o(h)$.

  (b) $\mathbb{P}(N_{t+h} - N_t = 1) = \lambda h + o(h)$.

  (c) $\mathbb{P}(N_{t+h} - N_t > 1) = o(h)$.

(iii) A Poisson process is a point process defined by its inter-arrival times, $\{S_n\}_{n\geq 1}$, which are i.i.d. $\mathsf{Exp}(\lambda)$.

Before we prove anything, we need a few results.

Lemma 1.2.3 (Memoryless Property)**.** We say that exponential random variables have the *memoryless property*. That is, for $t, s \geq 0$, if $X \sim \mathsf{Exp}(\lambda)$, then

$$\mathbb{P}(X > t + s \,|\, X > s) = \mathbb{P}(X > t).$$

Proof. We can write $\mathbb{P}(X > t + s \,|\, X > s)$ as

$$\frac{\mathbb{P}(X > t + s, X > s)}{\mathbb{P}(X > s)}.$$

As $s < t$ then $\mathbb{P}(X > t + s, X > s) = \mathbb{P}(X > t)$, so

$$\frac{\mathbb{P}(X > t + s, X > s)}{\mathbb{P}(X > s)} = \frac{\mathbb{P}(X > t + s)}{\mathbb{P}(X > s)} = \frac{e^{-\lambda(t+s)}}{e^{-\lambda s}} = e^{-\lambda(t)} = \mathbb{P}(X > t).$$

□

Theorem 1.2.4 (Markov Property)**.** If $\{N_t\}_{t\geq 0}$ is a Poisson process with rate $\lambda > 0$, then, for any $s > 0$, $\{N_{t+s} - N_s\}_{t\geq 0}$ is also a Poisson process with rate λ. Furthermore, this process is independent of $\{N_r\}_{r\leq s}$.

Proof. First, note that the event $\{N_s = i\}$ can be written as

$$\{N_s = i\} = \{T_i \leq s\} \cap \{S_{i+1} > s - T_i\}$$

and, given this, for $r \leq s$,

$$N_r = \sum_{j=1}^{i} \mathbb{I}(S_j \leq r).$$

Define the process starting at time s by $\widetilde{N}_t = N_{t+s} - N_s$. Then, given $\{N_s = i\}$, $\widetilde{S}_1 = S_{i+1} - (s - T_i)$ and $\widetilde{S}_n = S_{i+n}$ for $n \geq 2$. Now, by the memoryless property, $\widetilde{S}_1$ is an $\mathsf{Exp}(\lambda)$ random variable. Thus the $\{\widetilde{S}_n\}_{n\geq 1}$ are i.i.d. $\mathsf{Exp}(\lambda)$ random variables independent of $S_1, \ldots, S_n$. Thus, conditional of $\{N_s = i\}$, $\{\widetilde{N}_t\}_{t\geq 0}$ is a Poisson process independent of $\{X_r\}_{r\leq s}$. □

Now, we can prove Theorem 1.2.2.

Proof. I give a number of proofs of equivalences here. The rest will be left for exercises or self-study.

Part 1. First, we will show that (i) implies (ii). Now, as the increments are stationary, we know $N_{t+h} - N_t$ has the same distribution as $N_h - N_0$. So,

$$\mathbb{P}(N_{t+h} - N_t = 0) = \mathbb{P}(N_h - N_0 = 0) = \mathbb{P}(N_h = 0) = e^{-\lambda h}.$$

Taking a Taylor expansion of the exponential function, we get $e^{-\lambda h} = 1 - \lambda h + o(h)$. Likewise,

$$\mathbb{P}(N_{t+h} - N_t = 1) = \mathbb{P}(N_h = 1) = \lambda h e^{-\lambda h} = \lambda h + o(h).$$

and $\mathbb{P}(N_{t+h} - N_t > 1) = o(h)$.

Part 2. We will show that (ii) implies (i). We do this by solving some differential equations. First, let us define $p_j(t) = \mathbb{P}(N_t = j)$. Then,

$$p_0(t+h) = \mathbb{P}(N_{t+h} = 0) = \mathbb{P}(N_{t+h} - N_t = 0)\mathbb{P}(N_t = 0) = (1 - \lambda h + o(h))p_0(t).$$

Rearranging, we have

$$\frac{p_0(t+h) - p_0(t)}{h} = -\lambda p_0(t) + \frac{o(h)}{h}.$$

Because this holds for all t, we also get

$$\frac{p_0(t) - p_0(t-h)}{h} = -\lambda p_0(t-h) + \frac{o(h)}{h}.$$

Letting $h \to 0$ shows that $p_0(t)$ has a derivative (as the limit exists). This gives us

$$p_0(t)' = -\lambda p_0(t) \Rightarrow p_0(t) = Ae^{-\lambda t}.$$

Now, because $N_0 = 0$, we know that $p_0(0) = 1$ so $A = 1$ (i.e., $p_0(t) = e^{-\lambda t}$). Doing the same for $p_j(t)$ we have

$$\begin{aligned} p_j(t+h) &= \sum_{i=0}^{j} \mathbb{P}(N_{t+h} - N_t = i)\mathbb{P}(N_t = j-i) \\ &= \mathbb{P}(N_{t+h} - N_t = 0)\mathbb{P}(N_t = j) + \mathbb{P}(N_{t+h} - N_t = 1)\mathbb{P}(N_t = j-1) \\ &\quad + \sum_{i=2}^{j} \mathbb{P}(N_{t+h} - N_t = i)\mathbb{P}(N_t = j-i) \\ &= (1 - \lambda h + o(h))p_j(t) + (\lambda h + o(h))p_{j-1}(t) + o(h). \end{aligned}$$

Rearranging, we have

$$\frac{p_j(t+h) - p_j(t)}{h} = -\lambda p_j(t) + \lambda p_{j-1}(t) + \frac{o(h)}{h}.$$

By a similar argument to the one above, we get

$$p_j(t)' = -\lambda p_j(t) + \lambda p_{j-1}(t).$$

Now, remember that the product rule tells us that $(fg)' = f'g + g'f$. If we use the integrating factor $e^{\lambda t}$, we get

$$\begin{aligned} &p_j(t)' = -\lambda p_j(t) + \lambda p_{j-1}(t) \Rightarrow e^{\lambda t}p_j(t) = -\lambda e^{\lambda t}p_j(t) + \lambda e^{\lambda t}p_{j-1}(t) \\ &\Rightarrow e^{\lambda t}p_j(t)' + \lambda e^{\lambda t}p_j(t) = \lambda e^{\lambda t}p_{j-1}(t) \Rightarrow \left(e^{\lambda t}p_j(t)\right)' = \lambda e^{\lambda t}p_{j-1}(t). \end{aligned}$$

We can solve this by induction. We start with $j = 1$. This gives

$$\left(e^{\lambda t}p_1(t)\right)' = \lambda e^{\lambda t}p_0(t) = \lambda e^{\lambda t}e^{-\lambda t} = \lambda.$$

Integrating, we get

$$e^{\lambda t}p_1(t) = \lambda t + A \Rightarrow p_1(t) = t\lambda e^{\lambda t} + Ae^{\lambda t}.$$

Now, $p_j(0) = 0$ for $j > 0$, so $A = 0$ and $p_1(t) = t\lambda e^{\lambda t}$. Repeating in this way, we get

$$p_j(t) = \mathbb{P}(N_t = j) = e^{-\lambda t}\frac{(\lambda t)^j}{j!}.$$

Part 3. We show that (iii) implies (ii). This is just like the proof that (i) implies (ii). Using the Markov property (which was based on the interarrivals definition) we observe that $N_{t+h} - N_t$ has the same distribution as N_h, so

$$\mathbb{P}(N_{t+h} - N_t = 0) = \mathbb{P}(N_h = 0) = \mathbb{P}(S_1 > h) = e^{-\lambda h} = 1 - \lambda h + o(h),$$

and

$$\begin{aligned}&\mathbb{P}(N_{t+h} - N_t = 1) = \mathbb{P}(N_h = 0) = \mathbb{P}(S_1 < h, S_1 + S_2 > h)\\ &= \int_0^h e^{-\lambda(h-u)}\lambda e^{-\lambda u}\,\mathrm{d}u = \int_0^h \lambda e^{-\lambda h}\,\mathrm{d}u = \lambda h e^{-\lambda h} = \lambda h + o(h).\end{aligned}$$

It is also straightforward to see that

$$\mathbb{P}(N_{t+h} - N_t > 1) = \mathbb{P}(S_1 < h, S_1 + S_2 < h) \leq \mathbb{P}(S_1 < h)\mathbb{P}(S_2 < h) = o(h).$$

□

1.2.1 Order Statistics and the Distribution of Arrival Times

Order Statistics

Consider identically distributed random variables $X_1, \ldots, X_n$ with distribution $F(x)$ (that is $\mathbb{P}(X < x) = F(x)$. The *order statistics* of these random variables are simply the random variables ordered from smallest to largest. We write these as $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(n)}$. Two of the most important order statistics are the minimum, $X_{(1)}$, and the maximum, $X_{(n)}$.

Lemma 1.2.5. The distribution of the minimum is given by

$$\mathbb{P}\left(X_{(1)} \leq x\right) = 1 - (1 - F(x))^n.$$

Proof. We have

$$\begin{aligned}&\mathbb{P}\left(X_{(1)} \leq x\right) = 1 - \mathbb{P}\left(X_{(1)} > x\right) = 1 - \mathbb{P}(X_1 > x, \ldots, X_n > x)\\ &= 1 - \mathbb{P}(X_1 > x)\cdots\mathbb{P}(X_n > x) = 1 - (\mathbb{P}(X > x))^n = 1 - (1 - F(x))^n.\end{aligned}$$

□

Lemma 1.2.6. The distribution of the maximum is given by

$$\mathbb{P}\left(X_{(n)} \leq x\right) = F(x)^n$$

Proof. We have

$$\mathbb{P}\left(X_{(n)} \leq x\right) = \mathbb{P}(X_1 \leq x, \ldots, X_n \leq x) = \mathbb{P}(X \leq x)^n = F(x)^n.$$

□

Lemma 1.2.7. The density of the order statistics of n random variables $U_1, \ldots, U_n$, with distribution $\mathcal{U}(a, b)$, is given by

$$f(u_1, \ldots, u_n) = \frac{n!}{(b-a)^n} \mathbb{I}(a \leq u_1 \leq \cdots \leq u_n \leq b)$$

Proof. It is easiest to begin with the distribution and then take derivatives to get the density. We wish to calculate $\mathbb{P}\left(U_{(1)} \leq u_1, \ldots, U_{(n)} \leq u_n\right)$. Note that there are $n!$ orderings of the uniform random variables, each of which is equally likely. So,

$$\begin{aligned}\mathbb{P}\left(U_{(1)} \leq u_1, \ldots, U_{(n)} \leq u_n\right) &= n!\,\mathbb{P}(U < u_1) \cdots \mathbb{P}(U < u_n) \\ &= n! \frac{u_1 - a}{b-a} \cdots \frac{u_n - a}{b-a}.\end{aligned}$$

Taking derivatives, we get

$$\begin{aligned}f(u_1, \ldots, u_n) &= \frac{\partial}{\partial u_1} \cdots \frac{\partial}{\partial u_n} \mathbb{P}\left(U_{(1)} \leq u_1, \ldots, U_{(n)} \leq u_n\right) \\ &= \frac{n!}{(b-a)^n} \mathbb{I}(a \leq u_1 \leq \cdots \leq u_n \leq b).\end{aligned}$$

□

1.2.2 Distribution of Arrival Times

As it turns out, given that we know how many arrivals a Poisson process has had in an interval $[0, t]$ (that is, we know N_t), the arrival times will be uniformly distributed in the interval. This implies that the points of a Poisson process have very little structure to them (in some sense, it is a process that puts points as arbitrarily as possible on a line).

Theorem 1.2.8. Let $\{N_t\}_{t \geq 0}$ be a Poisson process. Then, conditional on $\{N_t = n\}$, $T_1, \ldots, T_n$ have the joint density function

$$f(t_1, \ldots, t_n) = \frac{n!}{t^n} \mathbb{I}(0 \leq t_1 \leq \cdots \leq t_n \leq t).$$

This is the density of the order statistics of i.i.d. uniform random variables on the interval $[0, t]$. This means the arrival times are distributed uniformly on this interval.

Proof. Consider the event $\{T_1 = t_1, \ldots, T_n = t_n, N_t = n\}$. Because there is a bijection between arrival times and inter-arrival times, this should have the same probability density as the event

$$\{S_1 = t_1, S_2 = t_2 - t_1, \ldots, S_n = t_n - t_{n-1}, S_{n+1} > t - t_n\}.$$

Because this is the joint density of i.i.d. exponential random variables, we can write this explicitly as

$$\lambda e^{-\lambda u_1} \lambda e^{-\lambda(u_2 - u_1)} \cdots \lambda e^{-\lambda(u_n - u_{n-1})} e^{-\lambda(t-u_n)} = \lambda^n e^{-\lambda t}.$$

We then get the conditional density we wish by dividing by the probability of the event $\{N_t = n\}$,

$$\begin{aligned} f(t_1, \ldots, t_n) &= \frac{\lambda^n e^{-\lambda t}}{(\lambda t)^n e^{-\lambda t}/n!} \mathbb{I}(0 \le t_1 \le \cdots \le t_n \le t) \\ &= \frac{n!}{t^n} \mathbb{I}(0 \le t_1 \le \cdots \le t_n \le t). \end{aligned}$$

□

1.3 Simulating Poisson Processes

As already mentioned, there are at least three ways of simulating a Poisson process. These follow directly from the different definitions we have used.

1.3.1 Using the Infinitesimal Definition to Simulate Approximately

The infinitesimal definition gives us a way to simulate the counting process $\{N_t\}_{t \ge 0}$ directly. This simulation is approximate but becomes increasingly good as $h \downarrow 0$. The idea is to slice the interval $[0, t]$ up into little pieces of length (sometimes called mesh size) h. In each one of these slices, we increase $\{N_t\}_{t \ge 0}$ with probability λh.

Listing 1.1: Matlab code

```
lambda = 4; t = 1; h = 0.0001;
mesh = 0:h:t; N = zeros(1, length(mesh));
S = []; jump_indices = [];

N(1) = 0;

```

```
for i = 2:length(mesh)
    if rand < lambda * h
        jump_indices = [jump_indices i];
        N(i) = N(i-1) + 1;
    else
        N(i) = N(i-1);
    end
end

if isempty(jump_indices)==0
    Ts = (jump_indices - 1)*h;
    S(1) = Ts(1);
    if length(jump_indices) > 1
        for i = 2:length(jump_indices)
            S(i) = Ts(i) - Ts(i-1);
        end
    end
end
```

1.3.2 Simulating the Arrival Times

The idea of this approach is to simulate the arrival times directly. Given an interval $[0, t]$, we know that we have a $\mathsf{Poi}(\lambda t)$ random number of arrivals. These are then distributed uniformly in $[0, t]$.

Listing 1.2: Matlab code

```
t = 5; lambda = 2;

T = [];
n = poissrnd(lambda * t);

if n~=0
    T = sort(t * rand(n,1));
    S = zeros(n,1);
    S(1) = T(1);
    if n > 1
        for i = 2:n
            S(i) = T(i) - T(i-1);
        end
    end
end
```

1.3.3 Simulating the Inter-Arrival Times

The idea here is to simulate the inter-arrival times, which are i.i.d. $\mathsf{Exp}(\lambda)$ random variables. Note that, if $U \sim \mathcal{U}(0,1)$, then $-\log(U)/\lambda$ is $\mathsf{Exp}(\lambda)$.

Listing 1.3: Matlab code

```
lambda = 4; h = 0.0001; t = 1;

s = 0;

S = []; Ts = [];

while s <= t
    inter_time = - log(rand) / lambda;;
    s = s + inter_time;
    if s > t
        break;
    else
        Ts = [Ts s];
        S = [S inter_time];
    end
end
```

1.4 Inhomogenous Poisson Processes

For many of the processes that Poisson processes are used to model, such as queues and traffic, the assumption that events occur at a constant rate (i.e., λ is constant) is very unrealistic. If you think about traffic (either on the internet or on a road) it tends to be heavier in some time periods and lighter in others. Inhomogenous Poisson processes modify the definition of a Poisson process so that it can incorporate time-dependent arrivals. Inhomogenous Poisson processes can be defined in a number of ways. Note, however, that it is no longer straightforward to use a definition based on inter-arrival times.

Definition 1.4.1 (Inhomogenous Poisson Process)**.** A point process $\{N_t\}_{t\geq 0}$ is said to be an inhomogenous Poisson process with intensity function $\lambda(t) \geq 0 \, \forall t \geq 0$.

(i) $N_0 = 0$.

(ii) For each $t \geq 0$, N_t has a Poisson distribution with paramater $\Lambda = \int_0^t \lambda(s)\,\mathrm{d}s$.

(iii) For each $0 \leq t_1 < t_2 < \cdots < t_m$, the random variables $N_{t_1}, \ldots, N_{t_m} - N_{t_{m-1}}$ are independent (that is, $\{N_t\}_{t\geq 0}$ has independent increments).

Definition 1.4.2 (Inhomogenous Poisson Process (Infinitesimal Definition))**.** A point process $\{N_t\}_{t\geq 0}$ is said to be an inhomogenous Poisson process with intensity function $\lambda(t) \geq 0 \forall t \geq 0$ if, as $h \downarrow 0$,

(i) $\{N_t\}_{t\geq 0}$ has independent increments.

(ii) $\mathbb{P}(N_{t+h} - N_t = 0) = 1 - \lambda(t)h + o(h)$.

(iii) $\mathbb{P}(N_{t+h} - N_t = 1) = \lambda(t)h + o(h)$.

(iv) $\mathbb{P}(N_{t+h} - N_t > 1) = o(h)$.

1.5 Simulating an Inhomogenous Poisson Process

There are at least two ways to simulate an inhomogenous Poisson process.

1.5.1 Acceptance-Rejection

One way to simulate an inhomogenous Poisson process on an interval $[0, t]$ is to simulate a homogenous Poisson process with parameter

$$\lambda^* = \max\{\lambda(s) : 0 \leq s \leq t\}$$

then 'thin' this process by only accepting arrivals with a certain probability. If an arrival / jump occurs at time T_i it should only be accepted with probability $\lambda(T_1)/\lambda^*$. It is not hard to check that this method works using the infinitesimal definition.

Listing 1.4: Matlab code

```
t = 10; lambda_star = 1;

T = [];
n = poissrnd(lambda_star * t);

if n~=0
    point_count = 0;
    for i = 1:n
        T_temp = t * rand;
        if rand < sin(T_temp)^2 / lambda_star
```

```
            point_count = point_count + 1;
            T(point_count) = T_temp;
        end
    end
    if point_count ~= 0
        T = sort(T);
        S = zeros(point_count,1);
        S(1) = T(1);
        if point_count > 1
            for i = 2:point_count
                S(i) = T(i) - T(i-1);
            end
        end
    end
end
```

1.5.2 Infinitesimal Approach (Approximate)

As in the homogenous Poisson process case, we can simulate an inhomogenous Poisson process approximately using its infinitesimal definition. This approximation becomes better as $h \downarrow 0$.

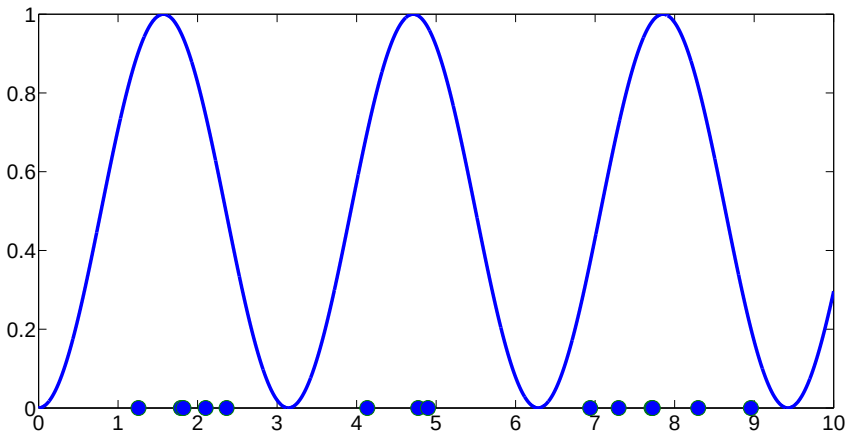

Figure 1.5.1: A realization of a inhomogenous Poisson process with the intensity function $\lambda(t) = 3\sin^2(t)$ plotted.

Listing 1.5: Matlab code

```
lambda = 1; t = 10; h = 0.0001;
```

```
mesh = 0:h:t; N = zeros(1, length(mesh));
S = []; jump_indices = [];

N(1) = 0;

for i = 2:length(mesh)
    if rand < 3*sin(h*(i-1))^2 * h
        jump_indices = [jump_indices i];
        N(i) = N(i-1) + 1;
    else
        N(i) = N(i-1);
    end
end

if isempty(jump_indices)==0
    Ts = (jump_indices - 1)*h;
    S(1) = Ts(1);
    if length(jump_indices) > 1
        for i = 2:length(jump_indices)
            S(i) = Ts(i) - Ts(i-1);
        end
    end
end
```

1.6 Compound Poisson Processes

A very useful extension of a Poisson process is what is called a *compound Poisson process*. A compound Poisson process replaces the unit jumps of a homogenous Poisson process with random jump sizes.

Definition 1.6.1 (Compound Poisson Process)**.** Given a homogenous Poisson process, $\{N_t\}_{t\geq 0}$ and a jump distribution G, we say

$$X_t = \sum_{i=1}^{N_t} J_i,$$

is a compound Poisson process, where the jumps $\{J_n\}_{n\geq 0}$ are i.i.d. draws from the distribution G.

Example 1.6.2. A street musician plays the accordion in the main street of Ulm for three hours. He hopes to earn enough for a beer, which costs €3.50. Throughout the three hours, people give him coins at random. There does

not seem to be any pattern to when people give him money, so a Poisson process is a good model. The amount of money each person gives is random, with distribution

$$\begin{aligned}\mathbb{P}(€0.05) &= 2/5\\ \mathbb{P}(€0.10) &= 2/5\\ \mathbb{P}(€0.20) &= 1/5.\end{aligned}$$

On average, 5 people per hour give the street musician money. This implies that the Poisson process has intensity $\lambda = 5$. What is the probability the musician gets his beer? That is, what is $\ell = \mathbb{P}(X_3 \geq 3.50)$. We can estimate this easily using Monte Carlo.

Listing 1.6: Matlab code

```
t = 3; lambda = 5; N = 10^6;
beer = zeros(N,1); beer_price = 350;

for i = 1:N

    n = poissrnd(lambda * t);

    if n~=0
        coins = zeros(n,1);
        for j = 1:n
            U = rand;
            coins(j) = (U <= 2/5)*5 + ...
                (U > 2/5 && U <= 4/5)*10 + (U > 4/5)*20;
        end
    end

    beer(i) = (sum(coins) >= beer_price);
end

ell_hat = mean(beer)
re_hat = std(beer) / (ell_hat * sqrt(N))
```

Chapter 2

Gaussian Processes

Gaussian processes are a reasonably large class of processes that have many applications, including in finance, time-series analysis and machine learning. Certain classes of Gaussian processes can also be thought of as spatial processes. We will use these as a vehicle to start considering more general spatial objects.

Definition 2.0.3 (Gaussian Process)**.** A stochastic process $\{X_t\}_{t\geq 0}$ is Gaussian if, for any choice of times $t_1, \ldots, t_n$, the random vector $(X_{t_1}, \ldots, X_{t_n})$ has a *multivariate normal distribution*.

2.1 The Multivariate Normal Distribution

Because Gaussian processes are defined in terms of the multivariate normal distribution, we will need to have a pretty good understanding of this distribution and its properties.

Definition 2.1.1 (Multivariate Normal Distribution)**.** A vector $\mathbf{X} = (X_1, \ldots, X_n)$ is said to be multivariate normal (multivariate Gaussian) if all linear combinations of $\mathbf{X}$, i.e. all random variables of the form

$$\sum_{k=1}^{n} \alpha_k X_k$$

have univariate normal distributions.

This is quite a strong definition. Importantly, it implies that, even if all of its components are normally distributed, a random vector is not necessarily multivariate normal.

Example 2.1.2 (A random vector with normal marginals that is not multivariate normal)**.** Let $X_1 \sim \mathsf{N}(0,1)$ and

$$X_2 = \begin{cases} X_1 & \text{if } |X_1| \leq 1 \\ -X_1 & \text{if } |X_1| > 1 \end{cases}.$$

Note that $X_2 \sim \mathsf{N}(0,1)$. However, $X_1 + X_2$ is not normally distributed, because $|X_1 + X_2| \leq 2$, which implies $X_1 + X_2$ is bounded and, hence, cannot be normally distributed.

Linear transformations of multivariate normal random vectors are, again, multivariate normal.

Lemma 2.1.3. Suppose $\mathbf{X} = (X_1, \ldots, X_n)$ is multivariate normal and A is an $m \times n$ real-valued matrix. Then, $\mathbf{Y} = A\mathbf{X}$ is also multivariate normal.

Proof. Any linear combination of $Y_1, \ldots, Y_m$ is a linear combination of linear combinations of $X_1, \ldots, X_n$ and, thus, univariate normal. □

Theorem 2.1.4. A multivariate normal random vector $\mathbf{X} = (X_1, \ldots, X_n)$ is completely described by a mean vector $\boldsymbol{\mu} = \mathbb{E}\mathbf{X}$ and a covariance matrix $\Sigma = \mathrm{Var}(\mathbf{X})$.

Proof. The distribution of $\mathbf{X}$ is described by its characteristic function which is

$$\mathbb{E}\exp\left\{i\boldsymbol{\theta}^\intercal \mathbf{X}\right\} = \mathbb{E}\exp\left\{i\sum_{i=1}^n \theta_i X_i\right\}.$$

Now, we know $\sum_{i=1}^n \theta_i X_i$ is a univariate normal random variable (because $\mathbf{X}$ is multivariate normal). Let $m = \mathbb{E}\sum_{i=1}^n \theta_i X_i$ and $\sigma^2 = \mathrm{Var}\left(\sum_{i=1}^n \theta_i X_i\right)$. Then,

$$\mathbb{E}\left\{i\sum_{i=1}^n \theta_i X_i\right\} = \exp\left\{im - \frac{1}{2}\sigma^2\right\}.$$

Now

$$m = \mathbb{E}\sum_{i=1}^n \theta_i X_i = \sum_{i=1}^n \theta_i \mu_i$$

and

$$\sigma^2 = \mathrm{Var}\left(\sum_{i=1}^n \theta_i X_i\right) = \sum_{i=1}^n \sum_{j=1}^n \theta_i \theta_j \mathrm{Cov}(X_i, X_j) = \sum_{j=1}^n \sum_{j=1}^n \theta_i \theta_j \Sigma_{i,j}.$$

So, everything is specified by $\boldsymbol{\mu}$ and Σ. □

There is nothing too difficult about dealing with the mean vector $\boldsymbol{\mu}$. However, the covariance matrix makes things pretty difficult, especially when we wish to simulate a high dimensional random vector. In order to simulate Gaussian processes effectively, we need to exploit as many properties of covariance matrices as possible.

2.1.1 Symmetric Positive Definite and Semi-Positive Definite Matrices

Covariance matrices are members of a family of matrices called *symmetric positive definite matrices*.

Definition 2.1.5 (Positive Definite Matrices (Real-Valued))**.** An $n \times n$ real-valued matrix, A, is positive definite if and only if

$$\boldsymbol{x}^\intercal A \boldsymbol{x} > 0 \text{ for all } \boldsymbol{x} \neq \mathbf{0}.$$

If A is also symmetric, then A is called symmetric positive definite (SPD). Some important properties of SPD matrices are

(i) $\mathsf{rank}(A) = n$.

(ii) $|A| > 0$.

(iii) $A_{i,i} > 0$.

(iv) A^{-1} is SPD.

Lemma 2.1.6 (Necessary and sufficient conditions for an SPD)**.** The following are necessary and sufficient conditions for an $n \times n$ matrix A to be SPD

(i) All the eigenvalues $\lambda_1, \ldots, \lambda_n$ of A are strictly positive.

(ii) There exists a unique matrix C such that $A = CC^\intercal$, where C is a real-valued lower-triangular matrix with positive diagonal entries. This is called the *Cholesky decomposition* of A.

Definition 2.1.7 (Positive Semi-definite Matrices (Real-Valued))**.** An $n \times n$ real-valued matrix, A, is positive semi-definite if and only if

$$\boldsymbol{x}^\intercal A \boldsymbol{x} \geq 0 \text{ for all } \boldsymbol{x} \neq \mathbf{0}.$$

If A is also symmetric, then A is called symmetric positive semi-definite (SPSD). Note that if A is an SPSD then there is a real-valued decomposition

$$A = LL^\intercal,$$

though it is not necessarily unique and L may have zeroes on the diagonals.

Lemma 2.1.8. Covariance matrices are SPSD.

Proof. Given an $n \times 1$ real-valued vector $\mathbf{x}$ and a random vector $\mathbf{Y}$ with covariance matrix Σ, we have

$$\mathrm{Var}\left(\mathbf{x}^\intercal \mathbf{Y}\right) = \mathbf{x}^\intercal \mathrm{Var}\left(\mathbf{Y}\right)\mathbf{x}.$$

Now this must be non-negative (as it is a variance). That is, it must be the case that

$$\mathbf{x}^\intercal \mathrm{Var}\left(\mathbf{Y}\right)\mathbf{x} \geq 0,$$

so Σ is positive semi-definite. Symmetry comes from the fact that $\mathrm{Cov}(X, Y) = \mathrm{Cov}(Y, X)$. □

Lemma 2.1.9. SPSD matrices are covariance matrices.

Proof. Let A be an SPSD matrix and $\mathbf{Z}$ be a vector of random variables with $\mathrm{Var}(\mathbf{Z}) = I$. Now, as A is SPSD, $A = LL^\intercal$. So,

$$\mathrm{Var}(L\mathbf{Z}) = L\mathrm{Var}(\mathbf{Z})L^\intercal = LIL^\intercal = A,$$

so A is the covariance matrix of $L\mathbf{Z}$. □

COVARIANCE POS DEF ...

2.1.2 Densities of Multivariate Normals

If $\mathbf{X} = (X_1, \ldots, X_n)$ is $\mathsf{N}(\boldsymbol{\mu}, \Sigma)$ and Σ is positive definite, then $\mathbf{X}$ has the density

$$f(\mathbf{x}) = f(x_1, \ldots, x_n) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\intercal \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})\right\}.$$

2.1.3 Simulating Multivariate Normals

One possible way to simulate a multivariate normal random vector is using the Cholesky decomposition.

Lemma 2.1.10. If $\mathbf{Z} \sim \mathsf{N}(\mathbf{0}, I)$, $\Sigma = AA^\intercal$ is a covariance matrix, and $\mathbf{X} = \boldsymbol{\mu} + A\mathbf{Z}$, then $\mathbf{X} \sim \mathsf{N}(\boldsymbol{\mu}, \Sigma)$.

Proof. We know from lemma 2.1.3 that $A\mathbf{Z}$ is multivariate normal and so is $\boldsymbol{\mu} + A\mathbf{Z}$. Because a multivariate normal random vector is completely described by its mean vector and covariance matrix, we simple need to find $\mathbb{E}\mathbf{X}$ and $\mathrm{Var}(\mathbf{X})$. Now,

$$\mathbb{E}\mathbf{X} = \mathbb{E}\boldsymbol{\mu} + \mathbb{E}A\mathbf{Z} = \boldsymbol{\mu} + A\mathbf{0} = \boldsymbol{\mu}$$

and

$$\mathrm{Var}(\mathbf{X}) = \mathrm{Var}(\boldsymbol{\mu} + A\mathbf{Z}) = \mathrm{Var}(A\mathbf{Z}) = A\mathrm{Var}(\mathbf{Z})A^\intercal = AIA^\intercal = AA^\intercal = \Sigma.$$

□

Thus, we can simulate a random vector $\mathbf{X} \sim \mathsf{N}(\boldsymbol{\mu}, \Sigma)$ by

(i) Finding A such that $\Sigma = AA^\intercal$.

(ii) Generating $\mathbf{Z} \sim \mathsf{N}(\mathbf{0}, I)$.

(iii) Returning $\mathbf{X} = \boldsymbol{\mu} + A\mathbf{Z}$.

Matlab makes things a bit confusing because its function 'chol' produces a decomposition of the form $B^\intercal B$. That is, $A = B^\intercal$. It is important to be careful and think about whether you want to generate a row or column vector when generating multivariate normals.

The following code produces column vectors.

Listing 2.1: Matlab code

```
mu = [1 2 3]';
Sigma = [3 1 2; 1 2 1; 2 1 5];
A = chol(Sigma);
X = mu + A' * randn(3,1);
```

The following code produces row vectors.

Listing 2.2: Matlab code

```
mu = [1 2 3];
Sigma = [3 1 2; 1 2 1; 2 1 5];
A = chol(Sigma);
X = mu + randn(1,3) * A;
```

The complexity of the Cholesky decomposition of an arbitrary real-valued matrix is $O(n^3)$ floating point operations.

2.2 Simulating a Gaussian Processes Version 1

Because Gaussian processes are defined to be processes where, for any choice of times $t_1, \ldots, t_n$, the random vector $(X_{t_1}, \ldots, X_{t_n})$ has a multivariate normal density and a multivariate normal density is completely describe by a mean vector $\boldsymbol{\mu}$ and a covariance matrix Σ, the probability distribution of Gaussian process is completely described if we have a way to construct the mean vector and covariance matrix for an arbitrary choice of $t_1, \ldots, t_n$.

We can do this using an *expectation function*

$$\mu(t) = \mathbb{E}X_t$$

and a covariance function

$$r(s,t) = \mathrm{Cov}(X_s, X_t).$$

Using these, we can simulate the values of a Gaussian process at fixed times $t_1, \ldots, t_n$ by calculating $\boldsymbol{\mu}$ where $\mu_i = \mu(t_1)$ and Σ, where $\Sigma_{i,j} = r(t_i, t_j)$ and then simulating a multivariate normal vector.

In the following examples, we simulate the Gaussian processes at evenly spaced times $t_1 = 1/h, t_2 = 2/h, \ldots, t_n = 1$.

Example 2.2.1 (Brownian Motion)**.** *Brownian motion* is a very important stochastic process which is, in some sense, the continuous time continuous state space analogue of a simple random walk. It has an expectation function $\mu(t) = 0$ and a covariance function $r(s,t) = \min(s,t)$.

Listing 2.3: Matlab code

```
%use mesh size h
h = 1000; t = 1/h : 1/h : 1;
n = length(t);

%Make the mean vector
mu = zeros(1,n);
for i = 1:n
    mu(i) = 0;
end

```

```
%Make the covariance matrix
Sigma = zeros(n,n);
for i = 1:n
    for j = 1:n
        Sigma(i,j) = min(t(i),t(j));
    end
end

%Generate the multivariate normal vector
A = chol(Sigma);
X = mu + randn(1,n) * A;

%Plot
plot(t,X);
```

Example 2.2.2 (Ornstein-Uhlenbeck Process)**.** Another very important Gaussian process is the Ornstein-Uhlenbeck process. This has expectation function $\mu(t) = 0$ and covariance function $r(s,t) = e^{-\alpha|s-t|/2}$.

Listing 2.4: Matlab code

```
%use mesh size h
h = 1000; t = 1/h : 1/h : 1;
n = length(t);

%paramter of OU process
alpha = 10;

%Make the mean vector
mu = zeros(1,n);
for i = 1:n
    mu(i) = 0;
end

%Make the covariance matrix
Sigma = zeros(n,n);
for i = 1:n
    for j = 1:n
        Sigma(i,j) = exp(-alpha * abs(t(i) - t(j)) / 2);
    end
end

%Generate the multivariate normal vector
A = chol(Sigma);
```

```
X = mu + randn(1,n) * A;

%Plot
plot(t,X);
```

Example 2.2.3 (Fractional Brownian Motion)**.** Fractional Brownian motion (fBm) is a generalisation of Brownian Motion. It has expectation function $\mu(t) = 0$ and covariance function $\text{Cov}(s,t) = 1/2(t^{2H} + s^{2H} - |t-s|^{2H})$, where $H \in (0,1)$ is called the Hurst parameter. When $H = 1/2$, fBm reduces to standard Brownian motion. Brownian motion has independent increments. In contrast, for $H > 1/2$ fBm has positively correlated increments and for $H < 1/2$ fBm has negatively correlated increments.

Listing 2.5: Matlab code

```
%Hurst parameter
H = .9;

%use mesh size h
h = 1000; t = 1/h : 1/h : 1;
n = length(t);

%Make the mean vector
mu = zeros(1,n);
for i = 1:n
    mu(i) = 0;
end

%Make the covariance matrix
Sigma = zeros(n,n);
for i = 1:n
    for j = 1:n
        Sigma(i,j) = 1/2 * (t(i)^(2*H) + t(j)^(2*H)...
            - (abs(t(i) - t(j)))^(2 * H));
    end
end

%Generate the multivariate normal vector
A = chol(Sigma);
X = mu + randn(1,n) * A;

%Plot
plot(t,X);
```

2.3 Stationary and Weak Stationary Gaussian Processes

Gaussian processes (and stochastic processes in general) are easier to work with if they are *stationary stochastic processes*.

Definition 2.3.1 (Stationary Stochastic Process)**.** A stochastic process $\{X_t\}_{t\geq 0}$ is said to be stationary if the random vectors $(X_{t_1}, X_{t_2}, \ldots, X_{t_n})$ and $(X_{t_1+s}, X_{t_2+s}, \ldots, X_{t_n+s})$ have the same distribution for all choices of s, n and $t_1, t_2, \ldots, t_n$.

An example of such a process would be an irreducible positive recurrent continuous time Markov chain started from its stationary distribution.

The conditions for a stochastic process to be stationary are quite strict. Often, a slightly weaker version of stationarity, called *weak stationarity* is required instead.

Definition 2.3.2 (Weak Stationary Stochastic Process)**.** A stochastic process $\{X_t\}_{t\geq 0}$ is said to be weak stationary (sometimes *wide sense stationary* or *second order stationary*) if $\mathbb{E}X_t = c$ for all $t \geq 0$ and $\mathrm{Cov}(X_t, X_{t+s})$ does not depends on t.

An example of a weak stationary process is the Ornstein-Uhlenbeck process, as $\mathbb{E}X_t = \mu(t) = 0$ and $\mathrm{Cov}(X_t, X_{t+s}) = r(t, t+s) = e^{-\alpha|t-(t+s)|/2} = e^{-\alpha s/2}$.

Lemma 2.3.3. Gaussian processes that are weak stationary are stationary.

Proof. We know from theorem 2.1.4 that Gaussian distributions are entirely determined by their mean vector and covariance matrix. Since the mean and covariance of a weakly stationary process do not change when the times are all shifted by s, a weakly stationary Gaussian process is stationary. □

An Ornstein-Uhlenbeck process is a weak stationary Gaussian process, so it is a stationary stochastic process.

2.4 Finite Dimensional Distributions

A very nice property of Gaussian processes is that we know their *finite dimensional distributions*.

Definition 2.4.1 (Finite Dimensional Distributions)**.** The finite dimensional distributions of a stochastic process $\{X_t\}_{t\geq 0}$ are the distributions of all vectors of the form $(X_{t_1}, \ldots, X_{t_n})$ with $n > 0$ and $0 \leq t_1 \leq \cdots \leq t_n$.

The finite dimensional distributions tell us a lot about the behaviour of a stochastic process. It is worth noting, however, that they do not fully specify stochastic processes. For example, Brownian motion is almost surely continuous but this property does not follow simply from specifying the finite dimensional distributions.

We can simulate the finite dimensional skeletons of a Gaussian process exactly. That is, we can generate $X_{t_1}, \ldots, X_{t_n}$ for any choice of n and $t_1, \ldots, t_n$. This is not always true for other stochastic processes. However, we do encounter a new form of error, *discretization error*.

Definition 2.4.2 (Discretization Error)**.** Discretization error is the error that arises from replacing a continuous object with a discrete object.

For example, if we wish to calculate the variance of the proportion of time that a Gaussian process spends above 0, or the expectation of the first time a process hits a set, A, then we will encounter an error in considering the process only at a fixed number of points.

Discretization error needs to be considered when we decide on a sampling budget. Consider, for example, an evenly spaced mesh of points $t_1 = 1/m, t_2 = 2/m, \ldots t_n = 1$. As m gets bigger, the mesh gets finer and the discretization error gets smaller. We still have statistical error, however, so we need to make sure we generate a large enough sample of realizations of the stochastic process (determined by the sample size N). The total work done by our simulation is then given by

$$\mathsf{work} = \text{number of samples} \times \text{work to make one sample} = Nf(m).$$

Usually, f grows at least linearly in m. In the case of Cholesky decomposition, for example, it is $O(m^3)$. For a fixed level of work, we need to decide on how much effort to allocate to reducing discretization error (how large m should be) and how much effort to allocate to reducing statistical error (how large N should be). Finding the optimal tradeoff can be difficult. We will consider such tradeoffs in a number of situations.

2.5 Marginal and Conditional Multivariate Normal Distributions

The multivariate normal distribution has many attractive properties. In particular, its marginal distributions are also multivariate normal. In addition, if we condition on part of the a multivariate normal vector, the remaining values are also multivariate normal.

To see this, we need to write normal random vectors in the appropriate form. Let $\mathbf{X} \sim \mathsf{N}(\boldsymbol{\mu}, \Sigma)$. We can decompose $\mathbf{X}$ into two parts, $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)^\intercal$. We can then write $\boldsymbol{\mu}$ as $(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)^\intercal$ and Σ as

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}.$$

Theorem 2.5.1 (Multivariate Normal Marginal Distributions)**.** Given $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2) \sim \mathsf{N}(\boldsymbol{\mu}, \Sigma)$,

$$\mathbf{X}_1 \sim \mathsf{N}(\boldsymbol{\mu}_1, \Sigma_{11}),$$

and

$$\mathbf{X}_2 \sim \mathsf{N}(\boldsymbol{\mu}_2, \Sigma_{22}).$$

Theorem 2.5.2 (Multivariate Normal Conditional Distributions)**.** Given $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2) \sim \mathsf{N}(\boldsymbol{\mu}, \Sigma)$, $\mathbf{X}_2$ conditional on $\mathbf{X}_1$ is multivariate normal with

$$\mathbb{E}[\mathbf{X}_2 \,|\, \mathbf{X}_1] = \boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\mathbf{X}_1 - \boldsymbol{\mu}_1)$$

and

$$\mathrm{Var}(\mathbf{X}_2 \,|\, \mathbf{X}_1) = \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}.$$

2.6 Interpolating Gaussian Processes

Because we know the finite dimensional distributions of Gaussian processes and know a lot about working with the multivariate normal distribution, we are able to interpolate between already simulated points of a Gaussian process. We can do this by simply drawing the new points conditional on the values that we have already generate. For example, if we have generate values for the process at 0.5 and 1, then we can generate values at the points 0.25 and 0.75 conditional on $X_{0.5}$ and X_1.

There are a number of reasons why interpolation might be useful. One is that it might not make sense to simulate the whole stochastic process on a fine mesh (which is expensive), but rather to simulate a coarse path of the stochastic process first and then focus our efforts on a particular region of this path. For example, if we are trying to estimate the first time a stochastic process hits a particular value, then we might want to focus our simulation efforts on the part of the stochastic process that is closest to this value. We should be careful, however, as simulating in this way could introduce a bias.

Example 2.6.1 (Iteratively Updating Brownian Motion)**.** Consider an example where we update Brownian motion in an iterative fashion. Remember, Brownian motion has mean function $\mu(t) = 0$ and covariance function

$r(s,t) = \min(s,t)$. We interpolate between points to simulate a process on an increasingly fine mesh.

Listing 2.6: Matlab code

```
num_levels = 10;

%Make the first two points (at 0.5 and 1)
t = [.5 1]; n = length(t);
Sigma = zeros(n,n);
for i = 1:n
    for j = 1:n
        Sigma(i,j) = min(t(i),t(j));
    end
end
X = chol(Sigma)' * randn(2,1);

plot([0; t'],[0; X]);
axis([0 1 -2.5 2.5]);

%Interpolate
for level = 2:num_levels
    %Make the additional mesh points
    t_new = 1/2^level : 2/2^level : (2^level-1)/(2^level);
    n_new = length(t_new);

    %Record the time points for the whole process
    t_temp = [t t_new];
    n_temp = length(t_temp);

    %Make a covariance matrix for the whole thing
    Sigma_temp = zeros(n_temp,n_temp);
    for i = 1:n_temp
        for j = 1:n_temp
            Sigma_temp(i,j) = min(t_temp(i),t_temp(j));
        end
    end

    %Make the separate Sigma components
    Sigma_11 = Sigma;
    Sigma_21 = Sigma_temp(n+1:n_temp, 1:n);
    Sigma_12 = Sigma_temp(1:n, n+1:n_temp);
    Sigma_22 = Sigma_temp(n+1:n_temp, n+1:n_temp);

```

```
    temp_mean = Sigma_21 * inv(Sigma_11) * X;
    Sigma_new = Sigma_22 - Sigma_21 * inv(Sigma_11) * Sigma_12;
    X_new = temp_mean + chol(Sigma_new)' * randn(n_new,1);
    X = [X; X_new];
    t = t_temp;
    n = n_temp;
    Sigma = Sigma_temp;
    [dummy index] = sort(t);
    another_dummy = waitforbuttonpress;

    plot([0; t(index)'],[0; X(index)]);
    axis([0 1 -2.5 2.5]);
end
```

2.7 Markovian Gaussian Processes

If a Gaussian process, $\{X_t\}_{t\geq 0}$, is Markovian, we can exploit this structure to simulate the process much more efficiently. Because $\{X_t\}_{t\geq 0}$ is Markovian, we can use we only need to know the value of X_{t_i} in order to generate $X_{t_{i+1}}$. Define

$$\sigma_{i,i+1} = \mathrm{Cov}(X_{t_i}, X_{t_{i+1}})$$

and

$$\mu_i = \mathbb{E}X_{t_i}.$$

By theorem 2.5.1, we know that $(X_{t_i}, X_{t_{i+1}})^\intercal$ has a multivariate normal distribution. In particular,

$$\begin{pmatrix} X_{t_i} \\ X_{t_{i+1}} \end{pmatrix} \sim \mathsf{N}\left(\begin{pmatrix} \mu_i \\ \mu_{i+1} \end{pmatrix}, \begin{pmatrix} \sigma_{i,i} & \sigma_{i,i+1} \\ \sigma_{i,i+1} & \sigma_{i+1,i+1} \end{pmatrix} \right).$$

Using theorem 2.5.2, we have

$$X_{t_{i+1}} \,|\, X_{t_i} = x_i \sim \mathsf{N}\left(\mu_i + \frac{\sigma_{i,i+1}}{\sigma i, i}(x_i - \mu_i)\,,\, \sigma_{i+1,i+1} - \frac{\sigma_{i,i+1}^2}{\sigma_{i,i}} \right).$$

Algorithm 2.7.1 (Generating a Markovian Gaussian Process)**.**

(i) Draw $Z \sim \mathsf{N}(0,1)$. Set $X_{t_1} = \mu_1 + \sqrt{\sigma_{i,i}} Z$.

(ii) For $i = 1, \ldots, m-1$ draw $Z \sim \mathsf{N}(0,1)$ and set

$$X_{t_{i+1}} = \mu_i + \frac{\sigma_{i,i+1}}{\sigma i, i}(x_i - \mu_i) + \left(\sqrt{\sigma_{i+1,i+1} - \frac{\sigma_{i,i+1}^2}{\sigma_{i,i}}} \right) Z.$$

Example 2.7.1 (Brownian Motion)**.** Consider Brownian motion, which is a Markov process. Now,

$$\mu_i = \mathbb{E}X_{t_i} = 0$$

and

$$\sigma_{i,i+1} = \min(t_i, t_{i+1}) = t_i.$$

So, our updating formula is

$$X_{t_{i+1}} = X_{t_i} + \left(\sqrt{t_{i+1} - t_i}\right) Z.$$

Listing 2.7: Matlab code

```
m = 10^4; X = zeros(m,1);
h = 1/m;
X(1) = sqrt(h)*randn;

for i = 1:m-1
    X(i+1) = X(i) + sqrt(h) * randn;
end

plot(h:h:1,X);
```

Example 2.7.2 (Ornstein-Uhlenbeck Process)**.** The Ornstein-Uhlenbeck process has

$$\mu_i = \mathbb{E}X_{t_i} = 0$$

with

$$\sigma_{i,i+1} = \exp\{\alpha|t_i - t_{i+1}|/2\}$$

and $\sigma_{i,i} = 1$. So, our updating formula is

$$X_{t_{i+1}} = \exp\{-\alpha|t_i - t_{i+1}|/2\}X_{t_i} + \left(\sqrt{1 - \exp\{\alpha|t_i - t_{i+1}|\}}\right) Z.$$

Listing 2.8: Matlab code

```
alpha = 50; m = 10^4;

X = zeros(m,1); h = 1/m;
X(1) = randn;

for i = 1:m-1
    X(i+1) = exp(-alpha * h / 2)*X(i)...
        + sqrt(1 - exp(-alpha * h ))*randn;
end

plot(h:h:1,X);
```

2.8 Brownian Motion

One of the most fundamental stochastic processes. It is a Gaussian process, a Markov process, a Lévy process, a Martingale and a process that is closely linked to the study of harmonic functions (do not worry if you do not know all these terms). It can be used as a building block when considering many more complicated processes. For this reason, we will consider it in much more depth than any other Gaussian process.

Definition 2.8.1 (Brownian Motion)**.** A stochastic process $\{W_t\}_{t\geq 0}$ is called Brownian motion (a Wiener process) if:

(i) It has independent increments.

(ii) It has stationary increments.

(iii) $W_t \sim \mathsf{N}(0,t)$ for all $t \geq 0$.

(iv) It has almost surely continuous sample paths. That is,

$$\mathbb{P}(\{\omega : X(t,\omega) \text{ is continuous in } t\}) = 1.$$

This definition implies that the increments of Brownian motion are normally distributed. Specifically, $W_{t+s} - W_t \sim \mathsf{N}(0,s)$. This implies the following simualtion scheme.

Listing 2.9: Matlab code

```
m = 10^3; h = 1/m;
X = cumsum(sqrt(h)*randn(m,1));
plot(h:h:1,X);
```

The first three parts of the definition of Brownian motion are equivalent to saying Brownian motion is a Gaussian process with $\mathrm{Cov}(X_t, X_s) = \min(t,s)$ and $\mathbb{E}X_t = 0$. However, the almost sure continuity of the paths of Brownian motion does not follow from this.

Theorem 2.8.2. The following two statements are equivalent for a stochastic process $\{X_t\}_{t\geq 0}$.

(i) $\{X_t\}_{t\geq 0}$ has stationary independent increments and $X_t \sim \mathsf{N}(0,t)$.

(ii) $\{X_t\}_{t\geq 0}$ is a Gaussian process with $\mu(t) = \mathbb{E}X_t = 0$ and $r(s,t) = \mathrm{Cov}(X_t, X_s) = \min(t,s)$.

Proof.

Part 1. First, we show (i) implies (ii). In order to show this, we need to show that $(X_{t_1}, \ldots, X_{t_n})$ is multivariate normal for all choices of $0 \leq t_1 \leq \cdots \leq t_n$ and $n \geq 1$. In order to $X_{t_1}, \ldots, X_{t_n}$ to be multivariate normal, we need $\sum_{k=1}^n \alpha_k X_{t_k}$ to be univariate normal (for all choices of n etc.). Now,

$$\begin{aligned}
\sum_{k=1}^n \alpha_k X_{t_k} &= \alpha_1 X_{t_1} + \sum_{k=2}^n \alpha_k X_{t_k} \\
&= \alpha_1 X_{t_1} + \sum_{k=2}^n \alpha_k \left(\sum_{j=1}^k X_{t_j} - \sum_{j=1}^{k-1} X_{t_j} \right) \\
&= \alpha_1 X_{t_1} + \sum_{k=2}^n \alpha_k \left(\sum_{j=1}^k X_{t_j} - \sum_{j=2}^{k} X_{t_{j-1}} \right) \\
&= \sum_{k=1}^n \alpha_k X_{t_1} + \sum_{k=2}^n \sum_{j=2}^k \alpha_k \left(X_{t_j} - X_{t_{j-1}} \right) \\
&= \left(\sum_{k=1}^n \alpha_k \right) X_{t_1} + \sum_{j=2}^n \left(\sum_{k=j}^n \alpha_k \right) \left(X_{t_j} - X_{t_{j-1}} \right) \\
&= \beta_1 X_{t_1} + \sum_{j=2}^n \beta_j \left(X_{t_j} - X_{t_{j-1}} \right).
\end{aligned}$$

Because X_{t_1} and $X_{t_2} - X_{t_1}, X_{t_3} - X_{t_2}, \ldots$ are independent random variables, it follows that the final equation results in a univariate normal random variable (sums of independent normals are normal). Now, we just need to check the expectation and covariance. It is easy to see that $\mathbb{E}X_t = 0$. In order to calculate the covariance, assume that $s < t$. We have that,

$$\mathrm{Cov}(X_s, X_t) = \mathbb{E}X_s X_t - \mathbb{E}X_s \mathbb{E}X_t = \mathbb{E}X_s X_t,$$

and

$$\mathbb{E}X_s X_t = \mathbb{E}X_s(X_t - X_s + X_s) = \mathbb{E}X_s^2 + \mathbb{E}X_s(X_t - X_s)$$

and, as the independent increments property implies $\mathbb{E}X_s(X_t - X_s) = 0$,

$$\mathrm{Cov}(X_s, X_t) = \mathbb{E}X_s^2 = s.$$

Repeating this with $t \geq s$, we get $\mathrm{Cov}(X_s, X_t) = \min(s, t)$.

Part 2. We show that (ii) implies (i). It follows immediately from the definition in (ii) that $X_t \sim \mathsf{N}(0, t)$. It is also immediate that $X_{t+s} - X_t$ is univariate normal (as this is a linear combination of multivariate normal

random variables). We can thus show this increment is stationary by showing that mean is constant and the variance only depends on s. Now, $\mathbb{E}X_{t+s} - X_t = 0$ and

$$\text{Var}(X_{t+s} - X_s) = \text{Var}(X_t) + \text{Var}(X_s) - 2\text{Cov}(X_s, X_t) = t + s - 2t = s.$$

Thus, the increments are stationary. Because the increments are multivariate normal, we can show the increments of the process are independent if we can show the covariance between increments is 0. So, take $u < v \leq s < t$. Then,

$$\begin{aligned}
&\text{Cov}(X_v - X_u, X_t - X_s) \\
&= \text{Cov}(X_v, X_t) - \text{Cov}(X_v, X_s) - \text{Cov}(X_u, X_t) + \text{Cov}(X_u, X_s) \\
&= \min(v, t) - \min(v, s) - \min(u, t) + \min(u, s) \\
&= v - v - u + u = 0.
\end{aligned}$$

So, non-overlapping increments are independent.

□