School of Mathematics

Simulating Sample Paths of Stochastic Processes Arising in Financial Engineering

Zhangdaihong Liu

A Masters project report submitted in partial fulfillment of the requirements for the award of the degree of MSc in Mathematical Finance of Loughborough University, September 2014

Author	Zhangdaihong Liu
Programme	Mathematical Finance
Academic Supervisor	Dr Dmitri Tseluiko School of Mathematics Loughborough University Loughborough Leics LE11 3TU
Copyright:	Zhangdaihong Liu 2014

Abstract

This report is devoted to methods for simulating paths of stochastic processes that arise in financial mathematics. It first shows three approaches of simulating paths of Brownian and geometric Brownian motions in one dimensional case - random walk, Brownian bridge and principal components constructions. Then there are examples of the application of geometric Brownian motion in option pricing - Asian option and lookback option. In the end, it talks about a simple model of processes with jumps - the jump-diffusion model and two methods of simulating this model. Programs and graphs are showed and all the algorithms are implemented by Matlab.

Acknowledgement

I would like to express my deepest gratitude to my project supervisor, Dr. Dmitri Tseluiko, who gave me a great help along the period of my project and was always very patient with all kinds of questions from me. Without his guidance and persistent help this dissertation would not have been possible. I would also like to thank my course mates and friends who provided me advise and assistance that helped me overcome the difficulties I met during the project.

Contents

1 Introduction

Simulations of stochastic processes have been widely used in the financial world. Such simulations help to value and analyze instruments, portfolios and investments by simulating the various sources of uncertainty affecting their value, and then determining their average value over the range of resultant outcomes [1]. This is the motivation for writting this report.

In this report, first comes a recap section that reminds us some concepts and definitions that will be used in the following sections. Straight after is the section introduces Brownian motion which is the main style of stochastic process that is discussed in this report. This section includes the definition, distribution, properties of Brownian motion and Brownian motion with drifts and diffusions. Then it comes to the main body of the report - simulating paths of stochastic processes.

We start with simulating the easiest style, the ordinary Brownian motion (with drift and diffusion). There are three methods discussed in this section. The first and most straightforward one is called random walk construction. It depends on the distribution of the process and every point on the path is generated from one random variable. Unlike the random walk construction, sampling the points on a path in a time order, the Brownian bridge construction always generates the midpoint between the start and end points. This method is like its name, building a bridge between the endpoints. It could be more convenient and time saving under some circumstances. The last construction that is considered is called principal components construction. This is a sort of optimisation method that is trying to use less random variables to sample a path. There are also algorithms, Matlab codes and explanations on how to implement the constructions.

The second main part investigates geometric Brownian motion. There is no big difference from the ordinary Brownian motion in simulating the paths. Therefore, our interests stay on its application in option pricing. The first example option that is applied to is Asian option. However there is no exact solution of this option. Therefore, we will also see another example, the lookback option. The result getting from numerical simulation is compared with the analytical solution. There are also algorithms and programs displayed to implement the simulation, and graphs are showed to better demonstrate the results.

The last section talks about the processes with jumps which is also a broadly used model in financial engineering. In this kind of process, a jump-diffusion model is mainly discussed. Two methods of simulating this process are introduced - simulating at fixed dates and simulating jump times. The algorithms are similar but simulating from different angles.

At the end of this report there is a conclusion section that summarizes all the constructions, methods, algorithms and results drawn from the whole project and discusses the possible future work.

2 Preliminaries

This section is a recap section. There will be some basic definitions and symbols that might be mentioned in the following sections. It will mainly focus on the concepts that relate to the Brownian motion.

2.1 Basic Concepts

Field and σ-Field[5]: Let $\mathscr{F}$ be a collection of subsets of a set Ω. Then $\mathscr{F}$ is called a field iff:
(1). $\Omega \in \mathscr{F}$, $\emptyset \in \mathscr{F}$;
(2). If $A \in \mathscr{F}$, then $A^c \in \mathscr{F}$;
(3). If $A_1, A_2, \dots, A_n \in \mathscr{F}$, then $\bigcup_{i=1}^{n} A_i \in \mathscr{F}$.
If we replace (3) with (3*):
(3*). If $A_1, A_2, \dots, A_n, \dots \in \mathscr{F}$, then $\bigcup_{i=1}^{\infty} A_i \in \mathscr{F}$,
then $\mathscr{F}$ is called a σ-field. The smallest σ-field consists of the empty set and the universal set i.e. $(\emptyset, \Omega)$.

Measure Space and Measurable[5]: A given set Ω, if $\mathscr{F}$ is a σ-field, the pair $(\Omega, \mathscr{F})$ is called a measure space; Consider two measure spaces $(\Omega_1, \mathscr{F}_1)$, $(\Omega_2, \mathscr{F}_2)$, the map $h : \Omega_1 \to \Omega_2$ is called measurable w.r.t. $\mathscr{F}_1$ and $\mathscr{F}_2$ iff for every $A \in \mathscr{F}_2$, $h^{-1}(A) \in \mathscr{F}_1$.
Remark (Borel Measurable): Let $(\Omega, \mathscr{F})$ be a measurable space and $h : \Omega \to \mathbb{R}$. If h is measurable relative to the σ-fields $\mathscr{F}$ and $\mathscr{B}(\mathbb{R})$ (Borel set), then h is called Borel measurable.

Probability Space[1]: A probability space has three elements: sample space Ω (a given set), set of events $\mathscr{F}$ (a σ-field) and probability measure $P : \mathscr{F} \to [0, 1]$. So a probability space is also a measure space with $P(\Omega) = 1$, the probability of entire events equal to 1. Denote it as triple $(\Omega, \mathscr{F}, P)$.
Remark1: Let X be any random variable. Its **expectation** is the integral with respect to (w.r.t.) probability P and we use the notation:

$$\mathbb{E}_P[X] = \int_\Omega x(\omega) dP(\omega), \quad \omega \in \Omega.$$

Remark2: The notation

$$\mathbb{E}_P[X^k] = \int_\Omega x^k(\omega) dP(\omega)$$

gives the kth moment of X where X is real-valued continuous random variable. Especially, the first moment is the expectation of X under the probability measure P i.e.

$$\mathbb{E}_P[X] = \int_\Omega x(\omega)dP(\omega).$$

Random Process (Stochastic Process)[4]: It is widely known as a collection of random variables [1]. It describes a random phenomenon of occurrence and this randomness is captured by probability space $(\Omega, \mathscr{F}, P)$. A common notation is $X = \{X_t : 0 \le t < \infty\}$ where X_t takes values on measure space $(\Omega, \mathscr{F})$ with probability P.

Probability Density Function (pdf) and Probability Distribution Function[3]: We call an absolutely integrable function $f_X : P \to [0, \infty)$ a pdf of random variable X if its integral on any set in the field equals to the probability distribution i.e. if

$$P_X(A) = \int_A f_X(x)dx, \quad A \in B.$$

It is equivalent to say $dP_X(x) = f_X(x)dx$.
The probability distribution function F_X is defined by $F_X(x) = P(X \le x)$. The relationship between pdf and probability distribution function is:

$$f_X(x) = \frac{dF_X(x)}{dx}.$$

Normal (Gaussian) Distribution and Normal Random Variable[3]: Random variable is like other variables in mathematics. It can take different values from a possible set associated with its possibility [1]; Normal random variable $X : \Sigma \to \mathbb{R}$ is the random variable with normal (Gaussian) distribution. It has pdf

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}},$$

where $\mu \in \mathbb{R}$ and $\sigma > 0$ are given numbers. The notation $N(\mu, \sigma^2)$ abbreviates the above formula which means the random variable having mean of μ and variance of σ^2.
Remark (Standard Normal Random Variable): It is the very special case of normal random variable with mean 0 and variance 1. Denote it as $N(0, 1)$. If $Z \sim N(0, 1)$ and $X \sim N(\mu, \sigma^2)$, then $X = \sigma Z + \mu$.

Filtration[3]: For probability space $(\Omega, \mathscr{F}, P)$, it is a family of nondecreasing sub-σ-field of $\mathscr{F}$ i.e. $\mathscr{F}_s \in \mathscr{F}_t$ if $s < t$, such collection $(\mathscr{F}_s)_{s \ge 0}$ is called a filtration.

The given measurable space endowed with a filtration is called a filtered space, having notation $(\Omega, \mathscr{F}, \mathscr{F}_s, P)$.

Adapted Process[3]: Let $(X_t)_{t\geq 0}$ be a random process and $(\Omega, \mathscr{F})$ be a filtered space with filtration $(\mathscr{F}_s)_{s\geq 0}$. We say $(X_t)_{t\geq 0}$ is an adapted process if X_t is $\mathscr{F}_t$-measurable for each $t \geq 0$.

Markov Process[3]: The process $(X_t)_{t\geq 0}$ is called a Markov process w.r.t $(\mathscr{F}_s)_{s\geq 0}$ on a filtered space $(\Omega, \mathscr{F}, \mathscr{F}_t, P)$ whenever

$$\mathbb{E}_P[f(X_t)|\mathscr{F}_s] = \mathbb{E}_P[f(X_t)|\sigma(X_s)], s \leq t, \tag{1}$$

where f is any bounded Borel function. It tells us $mathscrF_s$ and $\sigma(X_s)$ have the same information w.r.t to $f(X_t)$.

Markov Property for Brownian Motion[3]: This is a very important property of Brownian motion telling that the future value of Brownian motion does not rely on its past but only depends on its present. More generally speaking, Brownian motion has no 'memory'. The mathematical expression is

$$\mathbb{E}^x[f(B_{s+t_0}, \ldots, B_{s+t_n})|\mathscr{F}_s^B] = \mathbb{E}^{B_s}[f(B_{t_0}, \ldots, B_{t_n})], \forall x \in \mathbb{R} \tag{2}$$

for all bounded Borel function f, where $(B_t)_{t\geq 0}$ is a Brownian motion with its natural filtration $(\mathscr{F}_t^B)_{t\geq 0}$. It can be briefly stated as

$$\mathbb{E}^x[f(B_t)|\mathscr{F}_s^B] = \mathbb{E}^{B_s}[f(B_{t-s})], s \leq t. \tag{3}$$

Conditional Expectation[3]: Let X be a random variable on probability space $(\Omega, \mathscr{F}, P)$ and $\mathscr{G}$ is a sub-σ-field of $\mathscr{F}$. The expectation of X conditioning on $\mathscr{G}$ is defined by the unique $\mathscr{G}$-measurable random variable such that

$$\mathbb{E}_P[1_A X] = \mathbb{E}_P[1_A \mathbb{E}_P[X|\mathscr{G}]], \quad A \in \mathscr{G},$$

and the notation $\mathbb{E}_P[X|\mathscr{G}]$ is called the conditional expectation. It has the following commonly used properties:
(1).$\mathbb{E}[\mathbb{E}_P[X|\mathscr{G}]] = \mathbb{E}[X]$
(2).Tower property: $\mathbb{E}[\mathbb{E}_P[X|\mathscr{G}]|\mathscr{H}] = \mathbb{E}[X|\mathscr{H}]$, if $\mathscr{H} \subset \mathscr{G} \subset \mathscr{F}$.

Martingale[3]: A random process $(X_t)_{t\geq 0}$ is said to be a martingale w.r.t. probability space $(\Omega, \mathscr{F}, P)$ and its filtration $(\mathscr{F}_t)_{t\geq 0}$ if

(1). X_t is $\mathscr{F}_t$-measurable and integrable for each t.
(2). $\mathbb{E}[X_t|\mathscr{F}_s] = \mathbb{E}[X_s]$, for $s \leq t$.
This is another important property of Brownian motion which will be discussed in the next section.

$\mathbf{M^2[S,T]}$ **Space**[3]: This is a class of functions $f : \mathbb{R}^+ \times \Omega \to \mathbb{R}$ such that:
(1). $f \in \mathscr{B}(\mathbb{R}^+) \times \mathscr{B}(\Omega)/\mathscr{B}(\mathbb{R})$;
(2). $f(t,\omega)$ is $\mathscr{F}_t^B$-adapted;
(3). $\mathbb{E}[\int_S^T f(t,\omega)^2 dt] < \infty$.

Class $\mathbf{L_t^1}$ [3]: A random process $(X_t)_{t\geq 0}$ is said to be of class $\mathbf{L_t^1}$ if:
(1). $(X_t)_{t\geq 0}$ is $(\mathscr{F}_t^X)_{t\geq 0}$-adapted;
(2). $\int_0^t |X_s| ds < \infty, \quad P - a.s.$.

Itô Process[3]: An It$\hat{o}$ Process is a random process $(X_t)_{t\geq 0}$ such that:
(1). $t \mapsto X_t$ is almost surely continuous; (2). $X_t = X_0 + \int_0^t b_s ds + \int_0^t \sigma_s dB_s \quad P - a.s.$ for $\forall t \geq 0$, where $(b_s)_{s\geq 0}$ is a random process in $\mathbf{L_t^1}$ and $(\sigma_s)_{s\geq 0}$ is a random process in $\mathbf{M^2[0,t]}$.
In the form of differential equation, it can be written as:

$$dX_t = b_t dt + \sigma_t dB_t.$$

b_t is called the drift term and σ_t is called the diffusion term.

Stochastic Differential Equation (SDE)[3]: SDE has the form of

$$\begin{cases} dX_t = b_t X_t dt + \sigma_t X_t dB_t \\ X_0 = x, \quad x \in \mathbb{R} \end{cases}$$

$(b_s)_{s\geq 0}$ and $(\sigma_s)_{s\geq 0}$ are with the same conditions in Itô process. Solving this SDE is to find a random process $X_t)_{t\geq 0}$ satisfying the above equations.

Itô Formula[3]: Let $(X_t)_{t\geq 0}$ be an Itô process with drift term b_t and diffusion term σ_t. Furthermore, let $h : \mathbb{R}^+ \times \mathbb{R} \to \mathbb{R}$ be a function such that it is first order continuous with respect to the first variable and is second order continuous with respect to the second variable. Also it satisfies $\sigma_t \frac{\partial h}{\partial x}(t, X_t) \in \mathbf{M^2[0,t]}$, for all $t > 0$. Define an Itô process $Y_t = h(t, X_t)$ and we have

$$dY_t = \Big(\frac{\partial h}{\partial t}(t,X_t) + b_t \frac{\partial h}{\partial x}(t,X_t) + \frac{1}{2}\sigma t^2 \frac{\partial^2 h}{\partial x^2}(t,X_t)\Big)dt + \sigma_t \frac{\partial h}{\partial x}(t,X_t) dB_t. \qquad (4)$$

Exponential Distribution: A continuous random variable X is said to have an exponential (λ) distribution if it has probability density function:

$$f_X(x, \lambda) = \begin{cases} \lambda e^{\lambda x} & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases},$$

where $\lambda > 0$ is called the rate of the distribution.

Poisson Process: In probability theory, a Poisson process is a stochastic process that counts the number of events and the time points at which these events occur in a given time interval. The time between each pair of consecutive events has an exponential distribution with parameter λ and each of these inter-arrival times is assumed to be independent of other inter-arrival times.

Poisson Distribution: A discrete random variable X is said to have a Poisson distribution with parameter $\lambda > 0$, if, for $k = 0, 1, 2, \ldots$, the probability mass function of X is given by:

$$f(k, \lambda) = P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}.$$

The positive real number λ is equal to the expected value of X and also to its variance

$$\lambda = \mathbb{E}(X) = Var(X).$$

2.2 Notation

(1). $\sim$: equivalent to, similar to or more frequently in this paper, having the distribution of. e.g. $B \sim N(0, 1)$ means B has the standard normal distribution.
(2). $\wedge$: take the smaller value. e.g. $a \wedge b = \min(a, b)$.
(3). $\sum$: summation. e.g. $\sum_{x=0}^{10} S(x)$ means taking the sum value of S from $x = 0$ to $x = 10$.
(4). Let A be a matrix. A^T means its transposed matrix.
(5). $N(0, I)$: the standard normal distribution in multi dimensional space. I is the identity matrix.
(6). $(A - B)^+ = \max\{0, A - B\}$.

3 Brownian Motion

3.1 Definition

In reality, Brownian motion is the random motion of particles suspended in a fluid (a liquid or a gas) resulting from their collision with the quick atoms or molecules in the gas or liquid. While in mathematics, Brownian motion is also called Wiener process which has the definition that is a real-valued adapted random process $(B_t)_{(t\geq 0)}$ on a probability space $(\Omega, \mathscr{F}, P)$ with the following conditions:
(1) $P(B_0 = x) = 1$.
(2) the increments $(B_{t_{i+1}} - B_{t_i})_{0\leq i\leq n-1}$ are independent random variables for any integer n and any partition $0 = t_0 \leq t_1 \leq t_2 \leq ... \leq t_n$.
(3) $(B_t - B_s) \sim N(0, t-s)$ for any $0 \leq s < t$.
(4) $t \to B_t(\omega)$ is almost surely continuous for every $\omega \in \Omega$.

A standard Brownian motion (sBM) is a Brownian motion with $x = 0$ in condition (1) with $P(B_0 = 0) = 1$, i.e. an sBM almost surely starts from 0. The above definition also implies that for an sBM, $B_t \sim N(0,t)$, i.e. B_t is a normal random variable with mean 0 and variance t, and gives the pdf:

$$p_t(x) := \frac{1}{\sqrt{2\pi t}} e^{\frac{-x^2}{2t}}, \quad t > 0, x \in \mathbb{R}.$$

The probability measure P is called **Wiener measure** under which the coordinate process $B_t(\omega) = \omega(t), 0 \leq \infty$ is a sBM i.e.

$$P(B_0 = 0) = 1.$$

If for a general Brownian motion starting at x, the corresponding measure P^x is called the Wiener measure starting at x.

Consider a sBM and time points $t_1 < t_2 < t_3$, real numbers $a < b$. We have

$$\begin{aligned} P(B_{t_2} - B_{t_1} \in [a,b]) &= \int_{-\infty}^{\infty} dx \int_a^b \frac{1}{\sqrt{2\pi t_1}} e^{\frac{-x^2}{2t_1}} \frac{1}{\sqrt{2\pi(t_2 - t_1)}} e^{\frac{-y^2}{2(t_2-t_1)}} \\ &= \int_a^b \frac{1}{\sqrt{2\pi(t_2-t_1)}} e^{\frac{-y^2}{2(t_2-t_1)}}. \end{aligned}$$

This is consistent with the definition of Brownian motion that $(B_t - B_s) \sim N(0, t-s)$.

Denote $\mathbb{E}^x$ as the expectation with respect to measure P^x and $\mathbb{E}$ as the expectation with respect to P. We have

$$\mathbb{E}[f(B_t)] = \int_{-\infty}^{\infty} f(x) p_t(x) dx = \frac{1}{\sqrt{2\pi t}} \int_{-\infty}^{\infty} f(x) e^{\frac{-x^2}{2t}} dx \tag{5}$$

for any bounded Borel function f. Particularly,

$$\mathbb{E}[B_t] = \int_{-\infty}^{\infty} x p_t(x) dx = \frac{1}{\sqrt{2\pi t}} \int_{-\infty}^{\infty} x e^{\frac{-x^2}{2t}} dx. \tag{6}$$

The moments of sBM is given by straightforward Laplace transform of sBM that[3]

$$\phi(u) := \mathbb{E}[e^{-uB_t}] = e^{\frac{tu^2}{2}}, \quad u \geq 0.$$

This implicates that Brownian motion is exponentially integrable and all moments (see preliminaries) of Brownian motion exist.

The existence of Brownian motion can be proved by Kolmogorov's Theorem and Kolmogorov-Centsov Theorem. For the whole proof, please see reference [3].

3.2 Properties

We denote by $(B_t^x)_{t\geq 0}$ a Brownian motion starting at x, $x \in \mathbb{R}$, and by $(B_t)_{t\geq 0}$ the standard Brownian motion.
(1) sBM is a Gaussian process with mean and covariance:

$$E[B_t] = 0,$$

$$Cov[B_s, B_t] = E[B_s B_t] = s \wedge t,$$

where $s, t \in \mathbb{R}^+$.
(2) Spacial homogeneity: $B_t + x \sim B_t^x$.
(3) Reflection symmetry: $-B_t \sim B_t$.
(4) Scaling: $\forall c > 0, \sqrt{c} B_{\frac{t}{c}} \sim B_t$.
(5) Time inversion: if

$$W_t = \begin{cases} 0, & \text{if } t = 0, \\ tB_{\frac{1}{t}}, & \text{if } t > 0, \end{cases}$$

then $W_t \sim B_t$.
(6) Time reversibility: $B_s \sim B_t - B_{t-s}, \forall 0 \leq s \leq t$.

In the preliminaries, it was mentioned that martingale property is an important one of Brownian motion. Let $(B_t)_{t\geq 0}$ be sBM and $\mathscr{F}_s^B$ is its filtration. The *martingale property* tells that both $(B_t)_{t\geq 0}$ and $(B_t^2 - t)_{t\geq 0}$ are $(\mathscr{F}_s^B)_{s\geq 0}$-martingales.

3.3 Brownian motion with drift and diffusion

More generally, we consider a Brownian motion $X(t)$ with drift and diffusion. Let

$$X(t) = \mu t + \sigma B(t),$$

where constants $\mu, \sigma > 0$ and $B(t)$ is a standard Brownian motion. We call μ the drift coefficient and σ^2 the diffusion coefficient. Moreover,

$$\frac{X(t) - \mu t}{\sigma},$$

is a standard Brownian motion, therefore $X(t) \sim N(\mu t, \sigma^2 t)$.

Brownian motion has a wide range of applications in the real world especially in financial engineering like modeling of stock prices since it shows the randomness of the price trend. Also, it could be used as the model of random perturbations in physics, of certain behavior in biology, of management systems in economics etc.[4]

4 Generating Sample Paths

In this section, we will discuss three methods of simulating paths of a Brownian motion, random walk construction, Brownian bridge construction and principal components construction, in one dimensional case. We will also see a few examples of applying these methods and make a comparison between them.

4.1 Random Walk Construction

When talking about simulation of sample paths, it is natural firstly to think of simulating the values of a Brownian motion at fixed points (the discrete case) and then expand to the continuous case. The idea of random walk construction is generating n points in order (point by point from left to right) by making use of n random variables.

For a set of fixed time points, $0 = t_0 < t_1 < t_2 < \cdots < t_n$, since the increments of Brownian motion are independent, it is straightforward to generate the values by the formula

$$B(t_{i+1}) = B(t_i) + \sqrt{t_{i+1} - t_i} Z_{i+1}, \quad i = 0, \ldots, n-1, \tag{7}$$

where $B(t)$ is a standard Brownian motion, $Z(i), i = 0, \ldots, n-1$ are independent standard normal random variables.

For a Brownian motion with drift constant μ and diffusion constant σ^2, abbreviated $X \sim BM(\mu, \sigma^2)$, the formula becomes like

$$X(t_{i+1}) = X(t_i) + \mu(t_{i+1} - t_i) + \sigma\sqrt{t_{i+1} - t_i}Z_{i+1}, \quad i = 0, \dots, n-1, \tag{8}$$

where $X(0)$ is given.

Now, we try this method in Matlab. Consider a Brownian motion with drift μ and diffusion σ^2. The algorithm is: first divide a time interval of length 1 into 500 equal length subintervals, so the time increment dt is $1/500$. Then we input the values of μ and σ. Denote the difference of Brownian motion between two time points as dB and initiate it as a 0 vector consists of 500 components. Also we need to initialize the Brownian motion at $t = 0$ to be a zero vector. The first simulation is by defining $dB(t_1)$ as $\mu dt + \sigma\sqrt{dt}Z_1$ and $B(t_1) = dB(t_1)$ where Z_1 is some random variable generated by Matlab randomly. After the first approximation there will be a loop starting from 2 to the last step 500 with formulae like the following:

$$dB(t_i) = \mu dt + \sigma\sqrt{dt}Z_i,$$
$$B(t_i) = B(t_{i-1}) + dB(t_i).$$

The last step of the program produces the graph, showing the Brownian motion sampled by this algorithm.

Here is the program in Matlab. We assume μ is 0.5 and σ is 1:

```
-%randn('state',12) %Set initial state for repeatability;
-T=1;N=500;dt=T/N;           %choose time interval as 1 and divide
                              it into 500 subintervals
-mu=0.5; sigma=0.25;
-dB=zeros(1,N);              %give initial value 0 to dB
-B=zeros(1,N);               %give initial value 0 to B
-dB(1)=mu*dt+sigma*sqrt(dt)*randn;      %first approximation
-B(1)=dB(1);
```

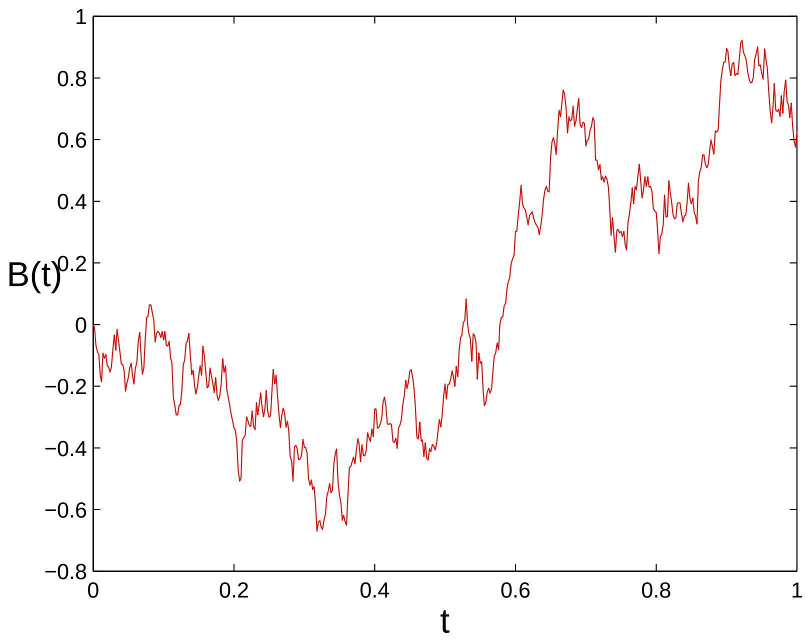

Figure 1: Brownian motion generated by random walk construction

```
-for j=2:N                  %the loop for the increment
    dB(j)=mu*dt+sigma*sqrt(dt)*randn;
    B(j)=B(j-1)+dB(j);
-end
-plot([0:dt:T],[0,B],'r-')  %plot B against t
-xlabel('t','FontSize',16)
-ylabel('B(t)','FontSize',16,'Rotation',0)
```

Figure 1 is the Brownian motion that is sampled from the program above. As we can see, this graph shows a good randomness of Brownian motion. It takes both negative and positive values and all the points locate centred by $B_t = 0.5$ with not too big variance.

4.2 Brownian Bridge Construction

Being different from random walk construction, generating $B(t_1), \dots, B(t_n)$ in time order, Brownian bridge construction provides a method of simulating $B(t_i)$ in any order we want. This relies on the conditional distribution from the values that we have already generated. For example, we can first generate the last point $B(t_n)$ and then generate $B(t_{n/2})$ conditional on $B(t_n)$. Use the values of $B(t_n)$ and $B(t_{n/2})$ to generate the intermediate points between the midpoint and the endpoint, etc. This method is like building a bridge between the endpoints of Brownian motion. We first choose the endpoints to sample and then fill in intermediate values conditioning on the endpoints and keep repeating this procedure until the process is constructed.

Now, we show the first step of this construction. Assume $(X_{[1]}, X_{[2]})$ is a vector with each element is a normally distributed random process and might be a vector as well. This vector has distribution:

$$\begin{pmatrix} X_{[1]} \\ X_{[2]} \end{pmatrix} \sim N\left(\begin{pmatrix} \mu_{[1]} \\ \mu_{[2]} \end{pmatrix}, \begin{pmatrix} \Sigma_{[11]} & \Sigma_{[12]} \\ \Sigma_{[21]} & \Sigma_{[22]} \end{pmatrix}\right), \tag{9}$$

with $\Sigma_{[22]}$ full rank. The **Conditioning Formula** gives that

$$(X_{[1]}|X_{[2]} = x) \sim N(\mu_{[1]} + \Sigma_{[12]}\Sigma_{[22]}^{-1}(x - \mu_{[2]}), \Sigma_{[11]} - \Sigma_{[12]}\Sigma_{[22]}^{-1}\Sigma_{[21]}). \tag{10}$$

This formula tells the distribution of $X_{[1]}$ conditional on $X_{[2]}$.

We set the endpoints as u, t and let $B(u) = x$, $B(t) = y$. Choose the intermediate point $s \in (u, t)$ to sample. From the process B is a Brownian motion we know that

$$\begin{pmatrix} B(u) \\ B(s) \\ B(t) \end{pmatrix} \sim N\left(0, \begin{pmatrix} u & u & u \\ u & s & s \\ u & s & t \end{pmatrix}\right). \tag{11}$$

The aim is to use the conditioning formula to get the distribution of $B(s)$ conditional on $B(u)$ and $B(t)$. Therefore rearrange the entries of vector $\mathbf{B}$ to get

$$\begin{pmatrix} B(s) \\ B(u) \\ B(t) \end{pmatrix} \sim N\left(0, \begin{pmatrix} s & u & s \\ u & u & u \\ s & u & t \end{pmatrix}\right). \tag{12}$$

It can be written as (3) if we set:

$$\begin{aligned}
&B(s) = X_{[1]}, (B(u), B(t))^T = X_{[2]}, \\
&\mu_{[1]} = 0, \mu_{[2]} = 0, \\
&\Sigma_{[11]} = s, \Sigma_{[12]} = (u \quad s), \\
&\Sigma_{[21]} = \begin{pmatrix} u \\ s \end{pmatrix}, \Sigma_{[22]} = \begin{pmatrix} u & u \\ u & t \end{pmatrix}.
\end{aligned}$$

Therefore, since $B(s)$ is a standard Brownian motion, applying formula (4), we find that the mean of $B(s)$ is

$$E[B(s)|B(u)=x,B(t)=y] = 0-(u \quad s)\begin{pmatrix} u & u \\ u & t \end{pmatrix}^{-1}\begin{pmatrix} x \\ y \end{pmatrix} = \frac{(t-s)x+(s-u)y}{t-u}, \tag{13}$$

and the variance is

$$s-(u \quad s)\begin{pmatrix} u & u \\ u & t \end{pmatrix}^{-1}\begin{pmatrix} u \\ s \end{pmatrix} = \frac{(s-u)(t-s)}{(t-u)}. \tag{14}$$

More generally, we can sample a point $B(t)$ conditional on k known values $B(t_1)=x_1, B(t_2)=x_2,\ldots,B(t_k)=x_k$. From the Conditioning Formula, or even more straightforwardly, from the Markov property of Brownian motion (see preliminaries), if $t_i < t < t_{i+1}$, $B(t)$ is independent of all the previous points that are earlier than t_i or the future points that are latter than t_{i+1}, i.e.

$$(B(t)|B(t_1)=x_1,B(t_2)=x_2,\ldots,B(t_k)=x_k) = (B(t)|B(t_i)=x_i,B(t_{i+1})=x_{i+1}).$$

Hence

$$(B(t)|B(t_1)=x_1,B(t_2)=x_2,\ldots,B(t_k)=x_k) = \\ N\left(\frac{(t_{i+1}-t)x_i+(t-t_i)x_{i+1}}{(t_{i+1}-t_i)}, \frac{(t_{i+1}-t)(t-t_i)}{(t_{i+1}-t_i)}\right).$$

When sampling the paths, we may set

$$B(t) = \mu + \sqrt{\sigma}Z, \tag{15}$$

where

$$\mu = \frac{(t_{i+1}-t)x_i+(t-t_i)x_{i+1}}{(t_{i+1}-t_i)}, \quad \sigma = \frac{(t_{i+1}-t)(t-t_i)}{(t_{i+1}-t_i)}, \tag{16}$$

with $t \in (t_i, t_{i+1})$ and $Z \sim N(0,1)$.

Particularly, the conditional mean at t is the linear interpolation between (t_i, x_i) and (t_{i+1}, x_{i+1}). This is to say that the conditional mean value at t lies on the segment connecting (t_i, x_i) and (t_{i+1}, x_{i+1}). All the values at t are normally distributed.

We now consider the implementation of this method on the standard Brownian motion. Firstly set the number of time indices to be equal to a power of 2. The program starts with generating the last point conditionally (or the last point can be given under some circumstances). Then we sample the midway point and by this recursion, generating the all the midway points step by step. For example, let the number of time indices be 16. We first sample $B(t_{16})$. Then we sample the first midpoint $B(t_8)$ and then $B(t_4)$ and $B(t_{12})$. Finally are $B(t_2)$, $B(t_6)$, $B(t_{10})$ and $B(t_{14})$. If the time index n is not a power of 2, we can choose the number $2^m < n$, using the same algorithm, then fulfill the rest points.

```
m=8
N=2^m;             %number of time indices
T=1; dt=T/N;       %choose time interval of length 1 and divide it into
                    N equal subintervals
for n=1:N
    Z(n)=randn;;    %generate N standard random variables
end
for i=1:N+1         %since Matlab requires the index of every variable
    t(i)=(i-1)*dt; to be positive, here define the time index from 1
end                 to N+1 of total still N time points.

h=N;
j_max=1; %the number of midpoints that need to be sampled at each step
B(h+1)=sqrt(t(h))*Z(h); %Sample the rightmost point
B(1)=0;   %define the first point of BM as 0

for k=1:8  %there are 8 steps of simulation
    i_min=h/2; i=i_min+1;    %the point to be sampled
    l=1;r=h+1;
    for j=1:j_max  %there are j_max midpoints at each step
        a=((t(r)-t(i))*B(l)+(t(i)-t(l))*B(r))/(t(r)-t(l));
        b=sqrt((t(i)-t(l))*(t(r)-t(i))/(t(r)-t(l)));
        B(i)=a+b*Z(i);  %expression of recursion
        i=i+h;
        l=l+h;
        r=r+h;
    end
```

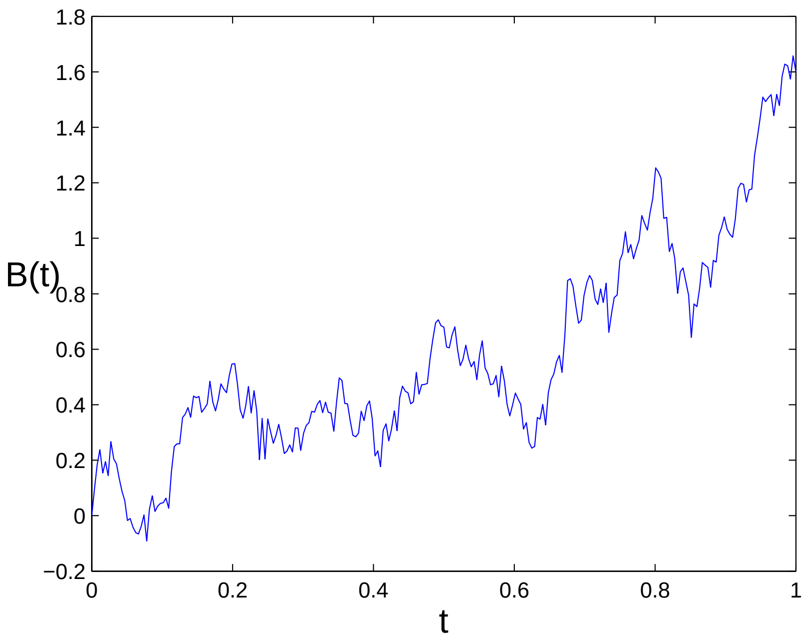

Figure 2: Brownian motion generated by Brownian bridge construction

```
    j_max=2*j_max; %the midpoints of next step is twice of it is
                    at last step
    h=i_min;
end
%plot the Brownian path against time t
figure
plot(t,B,'-');
```

Figure 2 shows a Brownian path that generated by Brownian bridge construction and we can see this graph is very similar to figure 1. This implicates the rationality of Brownian bridge construction.

This program could be modified for the construction of a Brownian motion with drift. Assume the points waiting to be sampled are $B(t_1), \ldots, B(t_m)$. The only difference in the program would be adding the term μt_h to $B(h+1)$, the 12th line in the program ($B(h+1)$ is used to show the last point in the program). It means when sampling the first rightmost point $B(t_m)$, the distribution would change from $N(0, t_m)$ to $N(\mu t_m, t_m)$. The rest distributions of $B(t_1), \ldots, B(t_{m-1})$ conditioning on

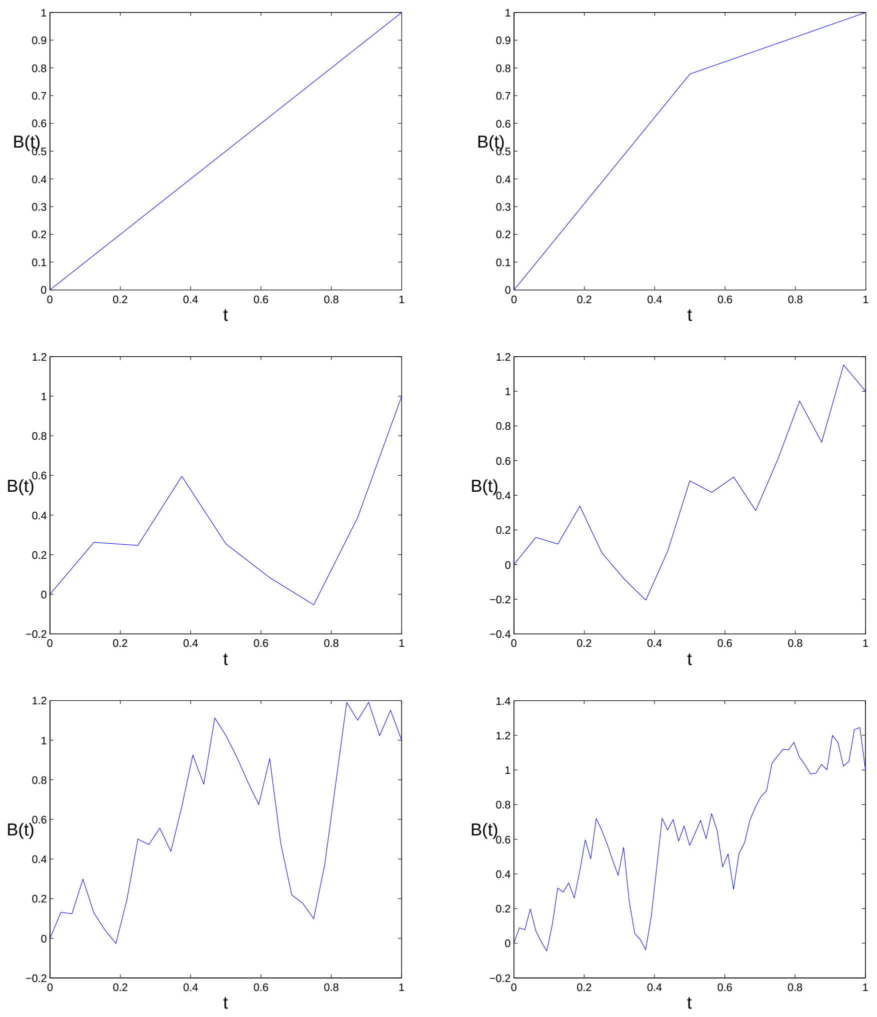

Figure 3: The Brownian path generated by Brownian bridge construction in the first 6 steps

$B(t_m)$ remain the same for all values of μ. Now consider the Brownian motion with diffusion coefficient σ^2. Since the conditional mean stays the same, the conditional variance is multiplied by σ^2. Then we may change the last point $B(h+1)$ (still the 12th line) from $\sqrt{t_h}Z_h$ to $\mu t_h + \sigma\sqrt{t_h}Z_h$ and multiply b, the standard deviation (the 18th line) by σ. Also the last output vector (the last line) needs to be multiplied by σ as well.

In the comparison of random walk construction, the Brownian bridge construction has no computational advantage. Both of these methods require $O(n)$ operations. However, we can see in figure 3 that generating a Brownian path by the Brownian bridge construction is like continuously adding details to the path step by step. Each step renews the whole path. So the advantage of this would appear when some variance reduction techniques are applied to the method. For example, the point that only matters is the last point, there is no need to simulate the full process. With the Brownian bridge construction, we may only need one random variable to get the point we want. But the random walk construction needs n random variables to get to the nth point. Therefore, the Brownian bridge construction could give a coarse simulation of Brownian motion when needed.

4.3 Principal Components Construction

4.3.1 Covariance matrix and the Cholesky factorization

Before talking about principal components, Let's introduce the concept of the covariance matrix of a Brownian motion $(B(t_1), B(t_2), \ldots, B(t_n))$. We have already known that the covariance of Brownian motion between $B(s)$ and $B(t)$ is

$$Cov[B(s), B(t)] = \min(s, t).$$

Denoting by $\mathbf{C}$ the covariance matrix of process $(B(t_1), B(t_2), \ldots, B(t_n))$, we therefore have

$$\mathbf{C}_{ij} = \min(t_i, t_j). \tag{17}$$

Assume the vector $(B(t_1), B(t_2), \ldots, B(t_n))$ have the distribution $N(0, \mathbf{C})$, with $\mathbf{C}$ as above. $\mathbf{C}$ can be factorized into $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ and the vector can be simulated by $\mathbf{AZ}$, where $\mathbf{Z} \sim N(0, \mathbf{I})$. As we can see, evaluating $\mathbf{AZ}$ is an $O(n^2)$ operation. The Cholesky factorization gives $\mathbf{A}$ as

$$\mathbf{A} = \begin{pmatrix} \sqrt{t_1} & 0 & \cdots & 0 \\ \sqrt{t_1} & \sqrt{t_2 - t_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{t_1} & \sqrt{t_2 - t_1} & \cdots & \sqrt{t_n - t_{n-1}} \end{pmatrix}. \tag{18}$$

More generally, for a random process X with normal distribution $N(\mu, \mathbf{\Sigma})$, there exists a corresponding process $\mathbf{Z}$ with $N(0, \mathbf{I})$ such that $X = \mu + \mathbf{AZ}$, where $\mathbf{\Sigma} = \mathbf{A}\mathbf{A}^T$. This is from the linear transformation property. Therefore, $X \sim N(\mu, \mathbf{A}\mathbf{A}^T)$ and the simulation of X has been transformed into the problem of finding a matrix $\mathbf{A}$ for which $\mathbf{\Sigma} = \mathbf{A}\mathbf{A}^T$.

4.3.2 Principal component

Now, we come to talk about the principal components. We could represent the Brownian motion we were talking about above as $\mathbf{B} = \mathbf{AZ}$. Since $\mathbf{B}$ is the vector with $N(0, \mathbf{C})$, $\mathbf{Z} \sim N(0, \mathbf{I})$, then

$$\begin{pmatrix} B(t_1) \\ B(t_2) \\ \vdots \\ B(t_n) \end{pmatrix} = \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{pmatrix} Z_1 + \begin{pmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{pmatrix} Z_2 + \cdots + \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{nn} \end{pmatrix} Z_n. \tag{19}$$

Denote $a_i = (a_{1i}, \ldots, a_{ni})^T$ and the n vectors a_i are the n columns of matrix $\mathbf{A}$. Assume the covariance matrix $\mathbf{C}$ is a $n \times n$ matrix and it is positive definite (for Brownian motion). Thus $\mathbf{C}$ has n nonnegative real eigenvalues $\lambda_1, \ldots, \lambda_n$ and n associated orthonormal eigenvectors $\upsilon_1, \ldots, \upsilon_n$, i.e.,

$$\mathbf{C}\upsilon_i = \lambda_i \upsilon_i. \tag{20}$$

It follows that $\mathbf{C} = \mathbf{V\Lambda V}^T$ where $\mathbf{V}$ is with columns $\upsilon_1, \ldots, \upsilon_n$ and $\mathbf{VV}^T = \mathbf{I}$, and Λ is the diagonal matrix with diagonal entries $\lambda_1, \ldots, \lambda_n$. Particularly, if we let $\mathbf{A} = \mathbf{V\Lambda}^{1/2}$, then $\mathbf{C}$ can be written in the form

$$\mathbf{AA}^T = \mathbf{V\Lambda V}^T = \mathbf{C}. \tag{21}$$

In equation (18), we call Z_i the independent factors driving the components of $\mathbf{B}$. If $\mathbf{C}$ has rank k, then $\mathbf{B}$ can be written as $\mathbf{B} = a_1 Z_1 + \cdots + a_k Z_k$ for some vectors $a_1, \ldots, a_k$ and random variables $Z_1, \ldots, Z_k$ are sufficient to represent $\mathbf{B}$. If $\mathbf{C}$ has full rank n, then we need $Z_1, \ldots, Z_n$ to show $\mathbf{B}$.

From what have been discussed above, if $\mathbf{C}$ has full rank and $\mathbf{C} = \mathbf{AA}^T$, then $\mathbf{A}$ has to be of full rank. By $\mathbf{B} = \mathbf{AZ}$, it follows that $\mathbf{Z} = \mathbf{A}^{-1}\mathbf{B}$. From the special case in (14), it can be shown that

$$\mathbf{A}^{-1} = \mathbf{\Lambda}^{-1/2}\mathbf{V}^T. \tag{22}$$

This implicates that Z_i is proportional to $\upsilon_i^T \mathbf{B}$ where υ_i is the ith eigenvector of $\mathbf{C}$. We want to find the linear combination $\omega^T X$ that best captures the variability of $\mathbf{B}$. If we put the eigenvalues of $\mathbf{C}$ in the descending order, i.e. $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_n$, the optimization problem would be solved by the first k eigenvectors if we want to approximate $\mathbf{B}$ by k factors. Besides, since

$$\upsilon_i^T \mathbf{C} \upsilon_i = \lambda_i,$$

then we have the expression of factor Z_i that

$$Z_i = \frac{1}{\sqrt{\lambda_i}} \upsilon_i^T \mathbf{B}. \tag{23}$$

Moreover, the representation $\mathbf{B} = \mathbf{AZ}$ gives the column $a_i = \sqrt{\lambda_i}v$. Thus, the optimal representation of $\mathbf{B}$ using k factors would be $\mathbf{B} = a_1Z_1 + \cdots + a_kZ_k$ with all the expression we have got above. At the same time, the square mean approximation error

$$\mathbf{E}\Big[\|\mathbf{B} - \sum_{i=1}^{k} a_iZ_i\|^2\Big], \quad (\|x\|^2 = x^Tx) \tag{24}$$

is minimized by setting $Z_i = \frac{1}{\sqrt{\lambda_i}}v_i^T\mathbf{B}$. We call the linear combination $v_i^T\mathbf{B}$ the *principal component* of Brownian motion $\mathbf{B}$ [1]. The variance explained by the first k principal components is the ratio

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_n}. \tag{25}$$

Since the eigenvalues $\lambda_1, \ldots, \lambda_n$ are in the descending order, then the variance that each of these eigenvalues explain is also in a descending order.

4.3.3 Principal components construction

The principal components construction is the application of principal components to discrete Brownian motion. From the Brownian bridge construction, we know that n random variables determine n points of a Brownian path. The principal components construction is a way for using less random variables to simulate as many points as possible.

For example, we consider a discrete Brownian motion with 200 steps and equal time intervals, i.e. the time increment is $t_i - t_{i-1} = 1/200$. The associated covariance matrix has entries $\mathbf{C}_{ij} = \min(i,j)/200, \quad i,j = 1,2,\ldots,200$. The five largest eigenvalues are $81.4632, 9.0518, 3.2589, 1.6629, 1.0061$. The sum of these five numbers explain more than 94% of the variability. As we can see, the eigenvalues decrease very rapidly which implicates the first few factors $Z_1, \ldots, Z_k$ would describe the most variability of the process. This is to say, we do not need all the eigenvalues and 200 random variables to simulate a path. The first 4 to 5 eigenvalues are suffice to describe the general trends of a Brownian path.

Figures 4 displays the first four eigenvectors that correspond to the first four largest eigenvalues in the above case. Every graph consists of 200 points with $\Delta t = 1/200$.

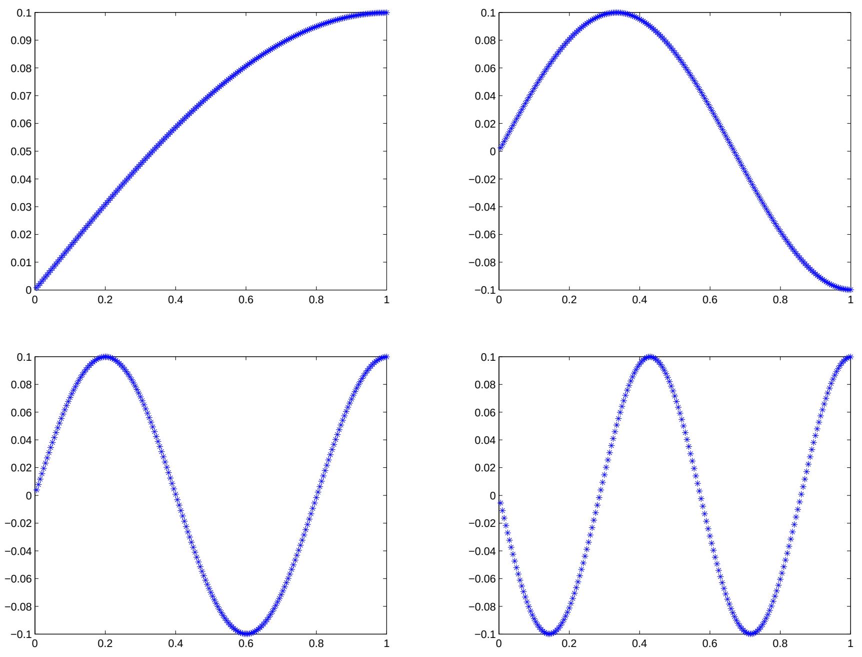

Figure 4: The eigenvectors associated with the first 4 largest eigenvalues of a 200-step Brownian motion

These figures seem to be sinusoidal. Actually, Akesson and Lehoczky [8] show that for an n-step path with equal spacing $t_{i+1} - t_i = \Delta t$,

$$v_i(j) = \frac{2}{\sqrt{2n+1}} \sin\left(\frac{2i-1}{2n+1}\right) j\pi, \quad j = 1, \ldots, n$$

and

$$\lambda_i = \frac{\Delta t}{4} \sin^{-2}\left(\frac{2i-1}{2n+1}\frac{2}{\pi}\right),$$

for $i = 1, \ldots, n$.

When implementing this method with Matlab, it is necessary to choose an appropriate number of principal components to simulate a path. This number could be obtained by calculating the eigenvalues or by experiments in Matlab using different steps. After calculating the eigenvalues with the step of $32, 100, 200...$, a conclusion drawn from this is that the first 4 eigenvalues would explain around 95% of the total variability (this could also be proved in algebra). Therefore, in the following experiments, 4 could be used as a proper number of components to simulate the path no matter how many time intervals are used. In the program, first of all, define the number of time subintervals to be 200, i.e. there will be 200 points in the final path. Like in the previous program, set the time increments to be equal lengths. Then define the covariance matrix with entries $\mathbf{C}_{ij} = \min(i,j)/200, i,j = 1,2,\ldots,200$, and get the eigenvalues and eigenvectors of this matrix. After all this, assign the initial value of B by a zero vector with 200 components. Last, the recursion formula of the Brownian motion is

$$B = \sum_{i=1}^{k} \sqrt{\lambda_i} v_i Z_i,$$

where k is the number of principal components, λ and v are eigenvalues and eigenvectors respectively. This is to say, the formula in the main loop is

$$B_{i+1} = B_i + \sqrt{\lambda_i} v_i Z_i.$$

Here let's choose k to be 4:

```
N=200; % number of subintervals
k=4; % number of principal components
t=1/N:1/N:1;
% generate the covariance matrix
C=zeros(N);
for i=1:N
    for j=1:N
        C(i,j)=min([i,j])/N;
    end
end
% get the eigenvalues and eigenvectors of the covariance matrix
[V,D]=eig(C);
% initiate B with a 200-row, 1-column zero vector
B=zeros(N,1);
% main loop for sampling a path
for i=1:k
    %D(N-i+1,N-i+1) means the eigenvalues taking from the largest one,
     V(:,N-i+1) means the corresponding eigenvectors.
    B=B+randn*sqrt(D(N-i+1,N-i+1))*V(:,N-i+1);
end
% plot B against time t
figure;
plot(t,B,'.-')
```

From the program above, the principal components construction gives a path of Brownian motion like in figure 5.

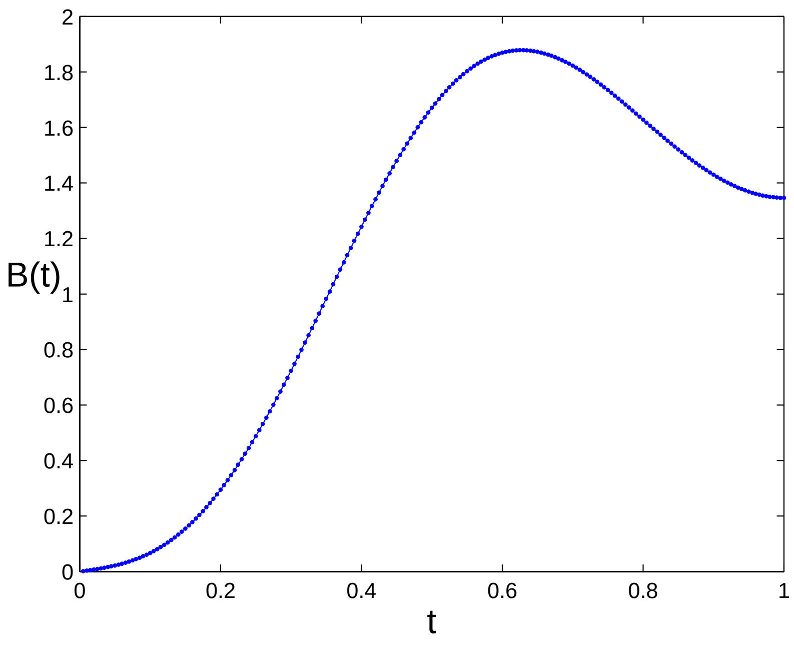

Figure 5: Brownian motion generated by principal components construction using 4 components. There are 200 points have been sampled.

Figure 5 gives a coarse trend of a Brownian path. Also, it contains most of the information that a path has, like the trace of the path and where the path going. However, this path showing in figure 5 does not look very random. This is because we abandoned most of the eigenvalues although these ones contain very little information, but still, they will add all the details to the path. This simulation with only 4 principal components is like a smooth approximation of a random path. We will see this by sampling the paths with more components. Before doing so, we need to make sure all the other factors stay the same only the number of components is different, i.e. all the random variables that are generated by Matlab are the same every time the program is running. This is achieved just by simply adding a command "randn('state',10)" at the beginning of the program.

The last graph in figure 6 is the full path simulated by this construction and it looks like the paths we obtained from other two constructions that have been introduced before. After seeing all these graphs, it can be concluded surely that the path sampling with 4 components does give the right and very close approximation of the full path. As the number of components increases, the path becomes more and more random.

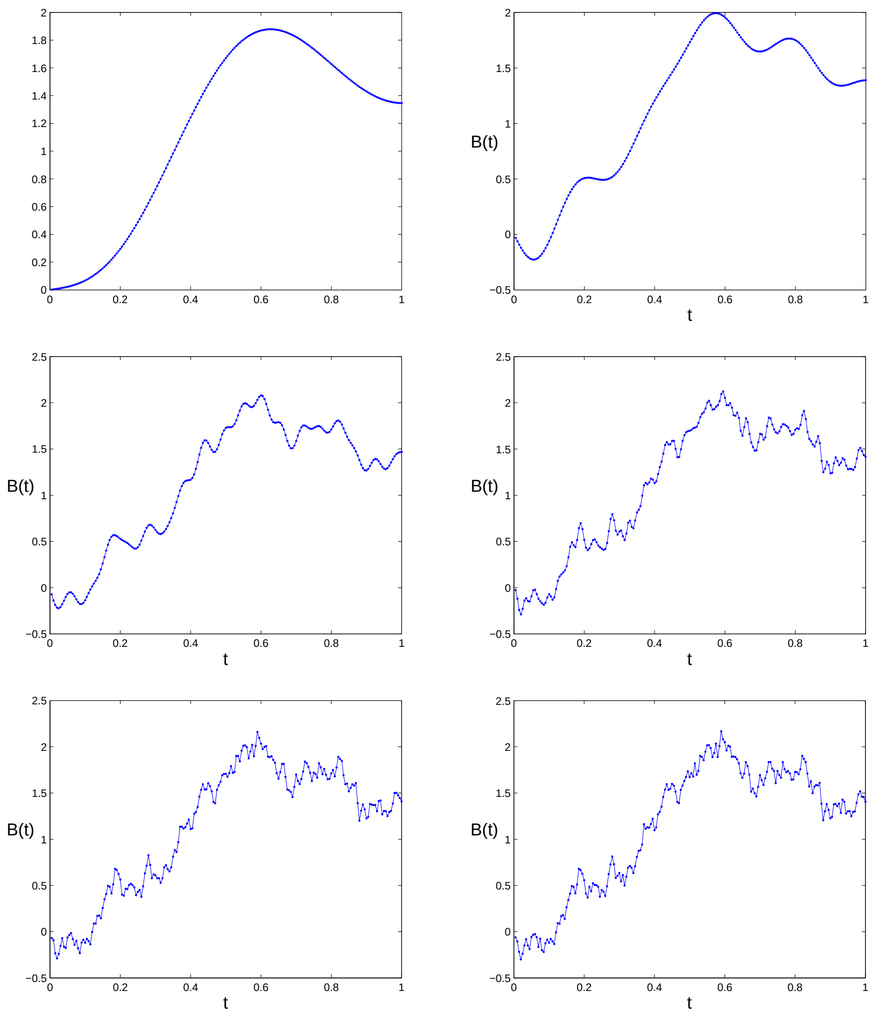

Figure 6: The Brownian path sampled by 4, 10, 40, 100, 160 and full components

By using graphs we will see the role of the largest eigenvalue (the first component) in determining the whole path. In order to see this, we abandon the first component and simulate the path from the second one and see how the path changes.

Figure 7 shows very obviously that after removing the largest eigenvalue, the path becomes very different no matter if it is using 4 components or all the rest of the components. It still cannot compensate the absence of the largest eigenvalue.

Comparing with random walk construction and Brownian bridge construction, the principal components construction mainly has two drawbacks. First is that this is an $O(n^2)$ operation, but random walk and Brownian bridge constructions only require $O(n)$ operations. The second disadvantage is none of the $B(t_i), i = 1, \ldots, n$, will be fully determined until all the random variables $Z_1, \ldots, Z_k$ have been processed,

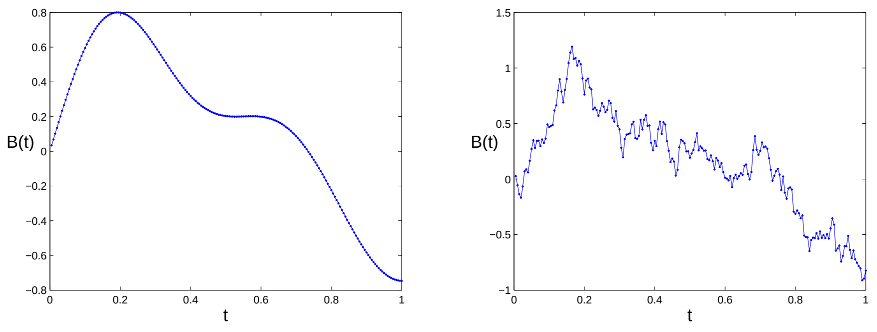

Figure 7: The Brownian path sampled from the second component to the fifth and to the last component. There are 200 points have been sampled.

i.e. even the simulation of $B(t_1)$ needs all the first k random variables to be determined. Being different from this, in either random walk construction or Brownian bridge construction, k points can be determined by k random variables. However, in principal components construction, the most shape of a Brownian motion could be described by a few random variables which is much fewer than what we need in either the random walk construction or the Brownian bridge construction. Also, the principal components construction is a very ideal method for explained variability of a Brownian motion.

In the discrete case, the relationship between eigenvalue and eigenvector is

$$\sum_j \min(t_i, t_j)v(j) = \lambda v(i)$$

In the continuous case, this relationship happens between eigenvalue and eigenfunction ψ and becomes the form of integral on $[0, 1]$,

$$\int_0^1 \min(s, t)\psi(s)ds = \lambda\psi(t).$$

The continuous expansion of Brownian motion called *Karhounen-Loève expansion* is

$$B(t) = \sum_{i=0}^{\infty} \sqrt{\lambda_i}\psi_i(t)Z_i, \quad 0 \le t \le 1, \tag{26}$$

where $Z_i, i = 1, 2, \ldots$ are standard normal random variables. This is the exact expression of the continuous Brownian motion and could be used as the counterpart of equation (18). The same algorithm with the discrete case, we could make use the first k terms to approximate a continuous Brownian path that explains the most variability.

5 Geometric Brownian Motion

Geometric Brownian motion is an exponentiated Brownian motion, i.e. if $S(t)$ is a geometric Brownian motion, then $\log S(t)$ is a Brownian motion. All the constructions of path simulation in Brownian motion could be applied to geometric Brownian motion just by simply exponentiating the process when sampling. The core of discussing geometric Brownian motion is that it is a fundamental model of asset price in finance. Before geometric Brownian motion had been used in finance, ordinary Brownian motion was used to describe the stock price as a model (Bachelier L. 1900). Then in the 1960s, Paul Samuelson introduced geometric Brownian motion to the stock market [2]. Geometric Brownian motion has more advantages than ordinary Brownian motion as the model of asset price since Brownian motion could take both negative and positive values, but geometric Brownian motion could only take positive values because of the exponentiation, just like the real price does.
Therefore, the simulation of paths of geometric Brownian motion refers to the previous sections. In this section, the emphasis will be on modeling of asset prices.

5.1 Properties of Geometric Brownian Motion

Unlike ordinary Brownian motion, the absolute changes $B(t_{i+1}) - B(t_i)$ are independent. The independence of geometric Brownian motion happens among the percentage changes, i.e.

$$\frac{S(t_2) - S(t_1)}{S(t_1)}, \frac{S(t_3) - S(t_2)}{S(t_2)}, ..., \frac{S(t_n) - S(t_{n-1})}{S(t_{n-1})}$$

are independent for $0 \leq t_1 < t_2 < ... < t_n$.

Consider a Brownian motion $(X_t)_{t\geq 0}$ with drift coefficient μ and diffusion coefficient σ^2. Define $f(X(t)) \equiv S(t) = S(0)e^{X(t)}$. Since $X_t \sim N(\mu, \sigma^2)$, then X satisfies

$$dX(t) = \mu dt + \sigma dB(t)$$

where $B(t)$ is an sBM. Moreover, based on Itô formula (see equation (4) in preliminaries), we have

$$dS(t) = S(t)(\mu + \frac{1}{2}\sigma^2)dt + S(t)\sigma dB(t). \tag{27}$$

However, in general, a geometric Brownian motion is often written in the form of an SDE as

$$\frac{dS(t)}{S(t)} = \mu dt + \sigma dB(t). \tag{28}$$

Comparing (27) and (28), it can be found that they are inconsistent in the sense of the drift term. (27) implicates μ is the drift of $\log S(t)$. This can be seen by applying the Itô formula to $\log S(t)$. While, in (28), $S(t)$ has the drift term $\mu S(t)$ and (28) is equivalent to

$$d\log S(t) = (\mu - \frac{1}{2}\sigma^2)dt + \sigma dB(t) \tag{29}$$

as this can also be obtained from Itô formula. It implicates the drift term of $\log S(t)$ is $\mu - \frac{1}{2}\sigma^2$.
Denote a geometric Brownian motion in (28) by $S \sim GBM(\mu, \sigma^2)$. μ in (28) is called the drift parameter although it is not the drift of neither $\log S(t)$ nor $S(t)$. σ in (28) is called the volatility parameter of $S(t)$. σ^2 is called the diffusion coefficient of $S(t)$.

Assume a geometric Brownian motion $S \sim GBM(\mu, \sigma^2)$ have initial value $S(0)$ and satisfy (29). Then we can write it in the form

$$S(t) = S(0)\exp([\mu - \frac{1}{2}\sigma^2]t + \sigma B(t)). \tag{30}$$

If we consider this geometric Brownian motion at time point $s < t$, then the above formula can be written in a more general form

$$S(t) = S(s)\exp([\mu - \frac{1}{2}\sigma^2](t-s) + \sigma(B(t) - B(s)). \tag{31}$$

Since the increments of standard Brownian motion are independent and normally distributed, then we get the general recursion of geometric Brownian motion for sampling a path

$$S(t_{i+1}) = S(t_i)\exp([\mu - \frac{1}{2}\sigma^2](t_{i+1} - t_i) + \sigma\sqrt{t_{i+1} - t_i}Z_{i+1}, i = 0, 1, ..., n-1, \tag{32}$$

where $Z_i, i = 1, ..., n$ are standard normal random variables. This is the formula that could be used in implementing the programs in Matlab.

The distribution of a geometric Brownian motion is called the lognormal distribution. This distribution is of the exponential of a normal random variable. If a random variable X has the distribution of $\exp(\mu + \sigma Z)$ and Z is a standard normal random variable, then it has notation $X \sim LN(\mu, \sigma^2)$. This notation does not mean that it has the mean value μ and variance σ^2. Whereas, the real mean value actually is larger than μ which just reflects the exponentiation.

In fact, if a geometric Brownian motion $S \sim GBM(\mu, \sigma^2)$, then

$$(S(t)/S(0)) \sim LN([\mu - \frac{1}{2}\sigma^2]t, \sigma^2 t) \tag{33}$$

and

$$\mathbb{E}[S(t)] = S(0)e^{\mu t}, \quad Var[S(t)] = S^2(0)e^{2\mu t}(e^{\sigma^2 t} - 1). \tag{34}$$

Moreover, from the Markov property, we also have

$$\mathbb{E}[S(t)|S(\tau), 0 \leq \tau \leq s] = \mathbb{E}[S(t)|S(s)] = S(s)e^{\mu(t-s)}, s < t. \tag{35}$$

The above conclusions tell that the expectation grows exponentially and μ actually acts as the average growth rate of geometric Brownian motion S. It is like a continuously compounded rate of return in the financial sense. Besides, $\mu - \frac{1}{2}\sigma^2$ behaves as the path growth rate of S.

5.2 Application of geometric Brownian motion in Options

Let us first briefly introduce the concept of option in financial sense. An option is a contract which gives the buyer the right, but not the obligation, to buy or sell an underlying asset or instrument at a specified pre-determined price (called strike price) on or before a specified date. The seller has the corresponding obligation to fulfill the transaction C that is to sell or buy C if the buyer "exercises" the option. An option which conveys to the buyer the right to buy an asset at a specific strike price is referred to as a call option; an option which conveys the buyer the right of to sell an asset at a specific strike price is referred to as a put option. The call situation is more often discussed.

We need to know how much an option worth initially, i.e. at time 0. Therefore, our interests of geometric Brownian motion mainly stay on its applications of option pricing. For path dependent options, when considering the payoffs of an underlying asset S, it is not simply a value at some specific exercise date while we consider the payoffs that rely on the whole path of $S(t)$. The price of an option is usually calculated by the discounted value of the expected payoff. The main idea of getting this price is that we simulate many paths of the underlying asset, evaluating the discounted payoff on each path, and then take the average value over all the paths.

5.2.1 Risk-neutral pricing

Then the question comes to the way that expected payoff are discounted. In the sense of geometric Brownian motion, this is way that the paths of underlying asset is generated. More specifically, it depends on how the drift parameter μ is chosen.

Assume there exists a constant continuously compounded interest rate r for riskless borrowing and lending. We could see this as borrowing or lending money from/to the bank. For example, deposit one pound to the bank at this rate at time 0. By the time of t, the money grows to $\beta(t) = e^{rt}$ pounds. Similarly, if a contract paying one pound at time t, it has value of e^{-rt} pound(s) at time 0. Let us then introduce the risk-neutral measure. For a discounted price process $(Z_t)_{t\geq 0}$, clearly, $\beta(t)$ is also a price process. Then the process $\beta(t)/Z(t)$ is a positive martingale and it has initial values $\beta(0)/Z(0) = 1$. Any positive martingale with an initial value 1 defines a change of probability measure. For a fixed interval $[0, T]$, the process $\beta(t)/Z(t)$ defines a new measure P_β through the likelihood ratio process

$$\Big(\frac{dP_\beta}{dP_0}\Big)_t = \frac{\beta(t)}{Z(t)}, \quad 0 \leq t \leq T. \tag{36}$$

More explicitly, this means that for any event $A \in \mathscr{F}_t$,

$$P_\beta(A) = \mathbb{E}_0\Big[\Big(\frac{dP_\beta}{dP_0}\Big)_t \cdot 1_A\Big] = \mathbb{E}_0\Big[\frac{\beta(t)}{Z(t)} \cdot 1_A\Big] \tag{37}$$

where 1_A denotes the indicator of the event A. Such probability measure is called *risk-neutral measure* [2],[3].

In pricing under the risk-neutral measure, we discount a payoff to be received at time t back to time 0 by dividing $\beta(t)$. As being discussed before, if S pays no dividends, the discounted price process $S(s)/\beta(s)$ is also a martingale such that

$$\frac{S(s)}{\beta(s)} = \mathbb{E}\Big[\frac{S(t)}{\beta(t)} | S(\tau), 0 \leq \tau \leq s\Big]. \tag{38}$$

In this case, if S is a geometric Brownian motion under the risk-neutral measure, then there is $\mu = r$ and

$$\frac{dS(t)}{S(t)} = r dt + \sigma dB(t). \tag{39}$$

This comes from the comparison between (38) and (35). Equation (35) explains that in the risk-neutral world, all the assets would have the same average rate of return. The drift parameter μ of geometric Brownian motion S would have the same value with risk-free risk r.

In the financial market, there are many options for investors to carry out their investments. All the options differ from the way they calculate the payoffs whereas all the payoffs depend on the paths of the value process $(S(t))_{t\geq 0}$. These kinds off payoffs are called *path-dependent payoffs*. We primarily focus on cases in which the payoffs depends on the values $S(t_1), \ldots, S(t_n)$ at fixed time points $t_1, \ldots, t_n$. In such cases, we could generate unbiased simulations. For the cases that the payoffs rely on the whole path of $S(t)$ over an interval $[0, T]$, pricing such options by simulation will often entail discretization bias. In the following examples, we will look at the Asian option (discrete case) and the look back option.

5.2.2 Asian option

A so called Asian option is an option on a time average of the underlying asset [2]. The call option has payoff

$$(S(T) - \bar{S})^+ \tag{40}$$

and the payoff of the put option is $(\bar{S} - S(T))^+$. Here T denotes the expiry time. $S(T)$ is the price of the underlying asset at expiry and

$$\bar{S} = \frac{1}{n} \sum_{i=1}^{n} S(t_i) \tag{41}$$

is the average price of the underlying asset over the discrete set of monitoring dates $t_1, \ldots, t_n (= T)$. There is no exact solution of the prices of the Asian option since the distribution of $\bar{S}$ is intractable. The Asian option can be seen as taking the average price of the underlying asset over the whole process, i.e. till time T, comparing it with the final price $S(T)$. If the average price is larger than the price at expiry, exercise the option to get payoff $S(T) - \bar{S}$. If not, don't exercise the option, then get nothing.

Now, let's try to price the call option numerically in Matlab. The main idea is like what was given at the beginning of section 5.2, taking the average value of a large number of the discounted payoffs. This averaged discounted payoff is the value of the option at time 0, i.e. the price of this option. In order to get the discounted payoffs, of course we need the expected payoffs which is given in formula (40). $\bar{S}$ comes from the price process $S(t)$ which is a geometric Brownian motion. Put all these in the programming order. What we need to do is, first of all, use the recursive procedure (32) to simulate a single path of geometric Brownian motion. Sum all the values up at every time point and divide by the number of time intervals to get a single average value of $\bar{S}$. Then apply (40) to attain one expected payoff. Repeat the whole procedure plenty of times to get a considerable number of expected payoffs. The last step is to discount the average value of all the payoffs we have got from previous steps using formula

$$\bar{V}(0) = e^{-rT}\bar{V}(T). \tag{42}$$

$\bar{V}(0)$ can also be obtained from the procedure that calculate $V(0)$ at each time of simulation and take the average over all the $V(0)$s. Therefore, the expression of $\bar{V}(0)$ becomes

$$V(0) = \mathbb{E}[e^{-rT}V(T)]. \tag{43}$$

We also want to observe how will the price change as the number of simulations increases. Whether it converges to a certain price or diverges.

At the beginning of the program, as always, set the number of time intervals and the length of the total time. Since the recursion (32) has terms μ and σ which are the drift parameter and volatility parameter, then they should be assumed as the beginning of the program as well. Together with all these variables is the initial value of the geometric Brownian motion. The plots come at the end of the program.

Assume μ to be 0.5 and σ to be 1. The number of time intervals is set to be 200:

```
randn('state',12)
M=100; % number of simulations
N=200; % number of subintervals
T=1; % total length of the time interval
dt=T/N; % time increment
mu=0.5; % drift parameter of GBM
sigma=1; % volatility parameter of GBM
S(1)=1; % initial value of GBM
```

```
% give the definition of every time point
for i=1:N+1
    t(i)=(i-1)*dt;
end

% main loop of calculating the expected payoffs
for m=1:M
    % sub-loop to generate a path of GBM
    for i=1:N
        S(i+1)=S(i)*exp((mu-0.5*sigma^2)*(t(i+1)-t(i))+
        sigma*sqrt(t(i+1)-t(i))*randn);
    end
    S_av(m)=sum(S)/(N+1); %the average value of the path
    VT(m)=max(0,S(N+1)-S_av(m)); %the payoff of Asian option
end

% calculate the discounted average of payoffs as the number
of simulations increases
for m=1:M
    V0(m)=exp(-r*T)*sum(VT(1:m))/m;
end

% plot the average value of GBM as simulation increases
figure
plot(1:M,S_av, '.')
% plot the expected payoff of GBM as simulation increases
figure
plot(1:M,VT,'.')
% plot the discounted average of payoff of GBM as simulation
increases
figure
plot(1:M,V0,'-k')
```

Now let us first see the graphs resulting from 100 times of simulation. That is to say, we take the average value of a 200-point geometric Brownian path by 100 times. Then calculate the discounted payoff at every time of simulaton and plot the

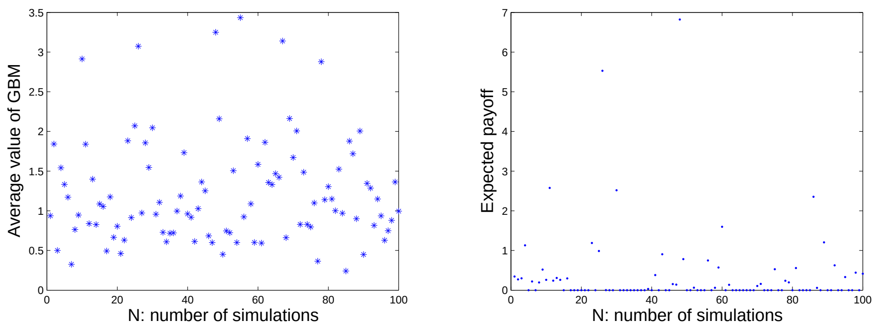

Figure 8: The left graph is the average value of the 200-point GBM at each simulation in total of 100 simulations; The graph to the right is the expected payoff at each simulation in total of 100 simulations of the Asian call option.

price value against the sample time.

From the left graph in figure 8, we can see that the average values of geometric Brownian motion take randomly but are all positive since this is the property of geometric Brownian motion, and the right graph tells the property of the payoff which is that it always takes non-negative values just like the expression of the Asian call option (40). Also, most of the discounted prices stay close to 0.

Figure 9 looks like a path of a random process. It does not show an obvious convergence. Thus, we need to see what happens if more simulations take place. Let us change the number of simulations to be 1000.

The most straightforward observation from the left graph in figure 10 is that, comparing with the left one in figure 8, this one shows a more regular distribution - most of the values locate between 0 and 1. They should around the mean value of this geometric Brownian motion which is given in (34). Since the initial value of the geometric Brownian motion and the drift parameter keep constant, then the mean value should also be a constant as it is always the same process being sampled on the same time interval.

From this figure, we can kind of tell the price it turned to be convergent, but still hasn't shown the convergent value. Therefore, we try 10000 simulations.

Finally, figure 12 shows a good sign of convergence and the option price is close to 0.32. There is a big fluctuation at the beginning of the figure. After the simulation times reach 2000, the path turns to gradually convergent. This is an expected result and indicates that the simulation of Asian call option is rational. However, it cannot be proved that the final value we got is the real price of the option. This is because there is no analytical solution for Asian option pricing since the average

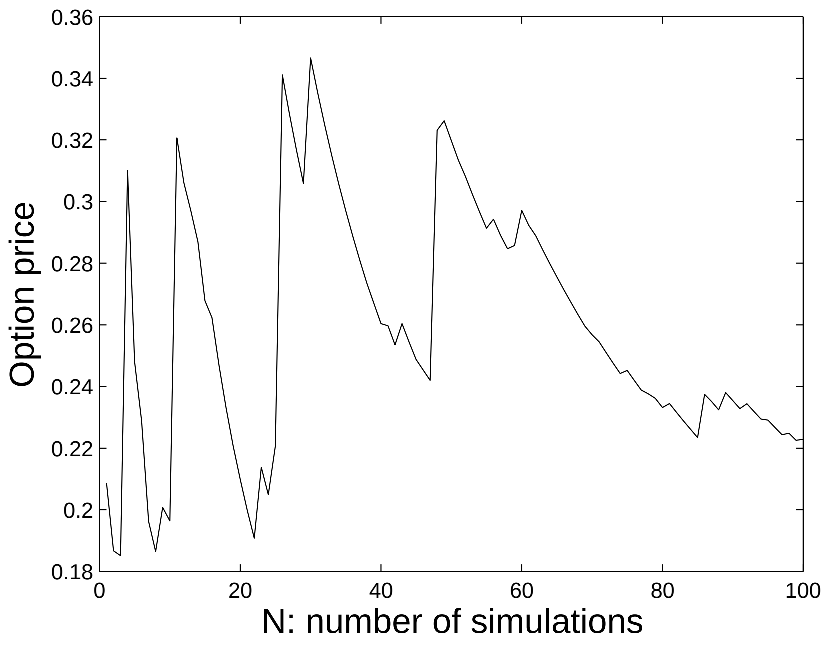

Figure 9: The dependence of the arithmetic average of the discounted payoff of the Asian call option on the number of simulations. The maximum number of simulations is 100. $\sigma = 1$ and $r = 0.5$.

value of geometric Brownian motion is not trackable. Now, let's see an example option which the price getting from the numerical simulation is testable.

5.2.3 Lookback option

An option which payoff is based on the maximum that the underlying asset price attains over some interval of time prior to expiration is called a lookback option. The lookback option depends on extremal values of the underlying asset price. The payoff of this option is the difference between the maximum price over the time between initiation and expiration and the asset price at the expiration. Hence, the expressions of Lookback calls and puts expiring at t_n are

$$(S(t_n) - \min_{i=1,\dots,n} S(t_i)) \quad and \quad (\max_{i=1,\dots,n} S(t_i) - S(t_n)) \tag{44}$$

respectively. A lookback call, for example, may be viewed as the profit from buying at the lowest price over $t_1, \dots, t_n (= T)$ and selling at the final price $S(T)$. Continuously monitored versions of these options are defined by taking the maximum or minimum over an interval rather than a finite set of points.

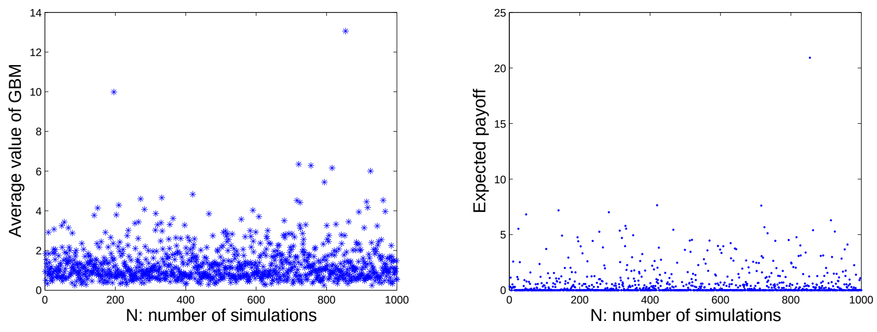

Figure 10: The left graph is the average value of the 200-point GBM at each simulation in total of 1000 simulations; The graph to the right is the expected payoff at each simulation in total of 1000 simulations of the Asian call option.

We may assume $Y(t)$ to be the maximum of the underlying asset price up to time t. Therefore, $Y(t)$ has the expression

$$Y(t) = \max_{0 \le u \le t} S(u) = S(0)e^{X(t)} \tag{45}$$

where $X(t)$ is as before a Brownian motion with drift μ and difussion σ. Therefore, the payoff of the lookback call has the value of

$$V(T) = Y(T) - S(T) \tag{46}$$

at expire time T. This payoff is non-negative since $Y(T)$ is always bigger or equal to $S(T)$.
At time $t \in [0, T]$, the risk-neutral price of lookback option is

$$V(t) = \mathbb{E}[e^{-r(T-t)}(Y(T) - S(T))|\mathscr{F}(t)]. \tag{47}$$

Particularly, at initiation i.e. $t = 0$,

$$V(0) = \mathbb{E}[e^{-rT}(Y(T) - S(T))|\mathscr{F}(0)]. \tag{48}$$

Because the pair of processes $(S(t), Y(t))$ has the Markov property, there must exist a function $v(t, x, y)$ such that

$$V(t) = v(t, S(t), Y(t)). \tag{49}$$

Set

$$\delta_{\pm}(\tau, s) = \frac{1}{\sigma\sqrt{\tau}}\big[\log s + \big(r \pm \frac{1}{2}\sigma^2\big)\tau\big], \tag{50}$$

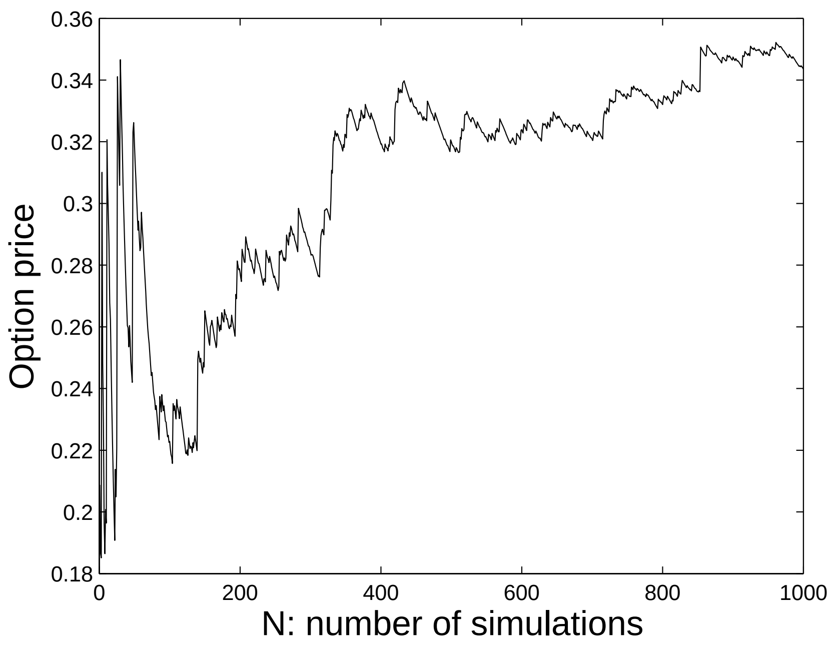

Figure 11: The dependence of the arithmetic average of the discounted payoff of the Asian call option on the number of simulations. The maximum number of simulations is 1000. $\sigma = 1$ and $r = 0.5$.

then the exact solution for the price of lookback option is

$$\begin{aligned} v(t,x,y) =& \big(1+\frac{\sigma^2}{2r}\big)xN\big(\delta_+\big(\tau,\frac{x}{y}\big)\big) + e^{-r\tau}yN\big(-\delta_-\big(\tau,\frac{x}{y}\big)\big) \\ & -\frac{\sigma^2}{2r}e^{-r\tau}\big(\frac{y}{x}\big)^{\frac{2r}{\sigma^2}}xN\big(-\delta_-\big(\tau,\frac{y}{x}\big)\big) - x, \quad 0 \le t < T, \quad 0 < x \le y, \end{aligned} \tag{51}$$

where $\tau = T - t$ and N is the cumulative distribution function with normal distribution. (The derivation of this solution please see [6], section 7.2.)

The implementation of pricing this option is very similar with the Asian option. Instead of calculating the average of the price process, lookback option calculates the maximum along the whole price process and use a different payoff expression (44). i.e. in the Matlab syntax of Asian option, substitute

```
S_av(m)=sum(S)/(N+1)
```

with

```
S_max(m)=max(S),
```

and

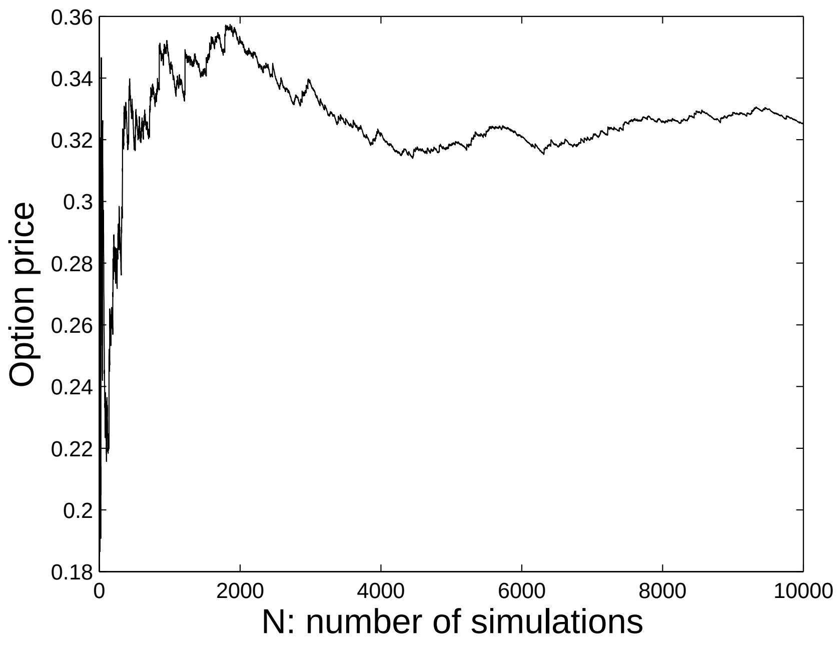

Figure 12: The dependence of the arithmetic average of the discounted payoff of the Asian call option on the number of simulations. The maximum number of simulations is 10000. $\sigma = 1$ and $r = 0.5$.

```
VT(m)=max(0,S(N+1)-S_av(m))
```

with

```
VT(m)=max(0,S_max(m)-S(N+1)).
```

The rest stays the same.

Figure 13 and 14 give the graphs plotting from the lookback option. They all are very similar to the Asian option. All the price values are non-negative and the option price also turns out to be convergent.

Now, let us see how well the simulation coincides with the exact solution. Firstly, we need to plot the exact option price against the initial asset price using formulae (50) and (51). The option price is the value of $v(t, x, y)$ in (51) at $t = 0$. At the initiation, the maximum value equals to the initial value of the price process. Therefore, we want to know $v(0, x_0, x_0)$ where x_0 is the initial price of the underlying asset. Since $\tau = T - t$, $t = 0$ and T is assumed to be 1, then $\tau = 1$. What's more, x/y in (51) would become 1. In such case, (50) becomes constant and (51) will be a straight line depending on x_0. Use codes:

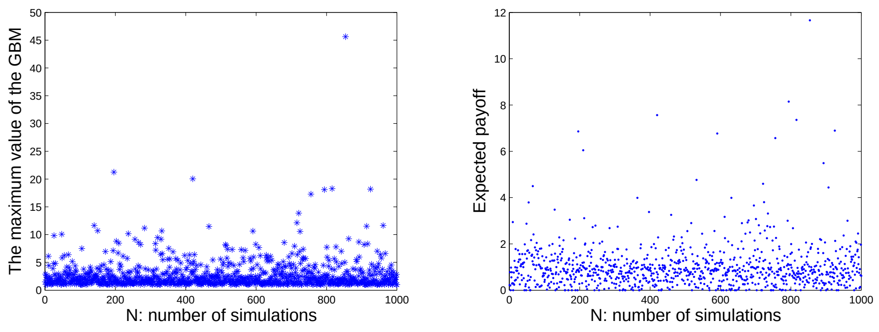

Figure 13: The left graph is the maximum value of the 200-point GBM at each simulation in total of 1000 simulations; The graph to the right is the expected payoff at each simulation in total of 1000 simulations of the lookback call option.

```
sigm=1; tau=1; r=0.5;

x=0:0.1:1; %the initial asset price takes value from 0 to 1 and
            increases by 0.1
dp=1/(sigm*sqrt(tau))*(r+0.5*sigm^2)*tau; %expression of delta plus
dm=1/(sigm*sqrt(tau))*(r-0.5*sigm^2)*tau; %expression of delta minus

% the expression of option price i.e. (51) at initiation
V0=(1+sigm^2/2/r)*x*cdf('norm',dp,0,1)+exp(-r*tau)*x*cdf('norm',-dm,0,1)
   -(sigm^2/2/r)*exp(-r*tau)*x*cdf('norm',-dm,0,1)-x;
% plot the option price against the initial value
figure;
plot(x,V0)
```

Here, we make the initial asset price take a value from 0 to 1 and increase by 0.1. Besides, assume $\sigma = 1$ and $r = 0.5$. As what we expected before, the graph is a straight line as in figure 15:

Now, we plot the same graph with the numerical simulation (see Appendix A for the Matlab code) and put it together with the exact solution to see the difference. As figure 16 shows, there is a distinguished difference between the green and the blue lines which tells that the simulation is not very accurate. There are three ways in order to fix this problem: (1) Narrow down the length of time subintervals to get more points on a price path so that the discreteness is reduced (use N to denote

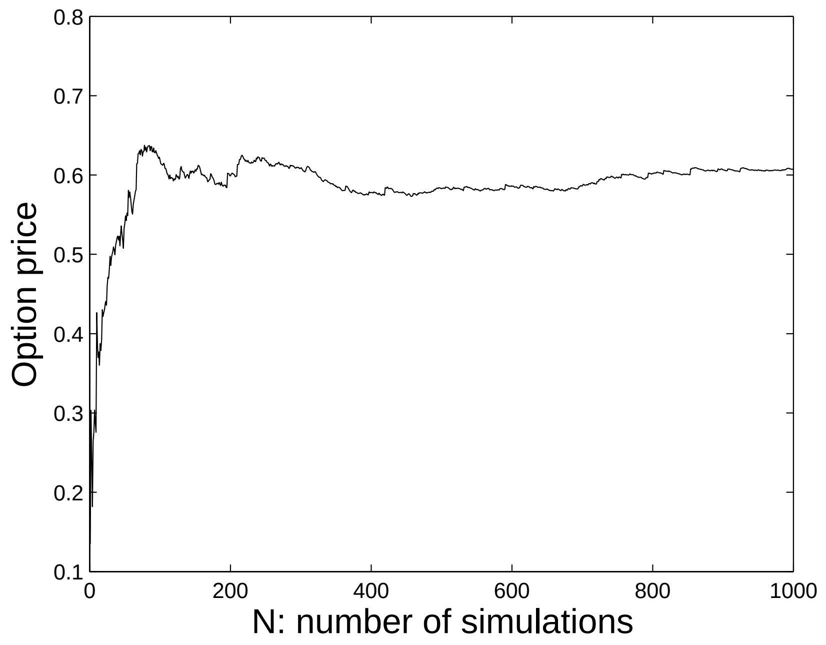

Figure 14: The dependence of the maximum value of the discounted payoff of the lookback option on the number of simulations. The maximum number of simulations is 1000. $\sigma = 1$ and $r = 0.5$.

the number of time intervals); (2) Increase the number of simulations to get more information to plot the simulated path of option price (use M to denote the number of simulations). (3) Divide the interval of initial asset price into more subintervals so that more details will be added to the graph (use dx to denote the increment of initial asset price). Or these three ways could be done at the same time and the accuracy of the simulated price will be improved very promptly.

We can see that in figure 17, before changing the increment of initial asset price, the simulated option price path is a straight-line-like graph. After changing it, the simulated path shows a fluctuation. Figure 17 also shows that all the three method make the simulated option price closer to the exact price. But still not yet coincide. Therefore, we try three ways at the same time to get figure 18. As it shows in the graph, the accuracy of the simulation improved. This may be a good simulation of the option price.

From all the graphs we have, we can see straightforwardly that the green line is always under the blue line, i.e. the numerical solution is always smaller than the exact solution. This makes sense because the process is continuous but it is only

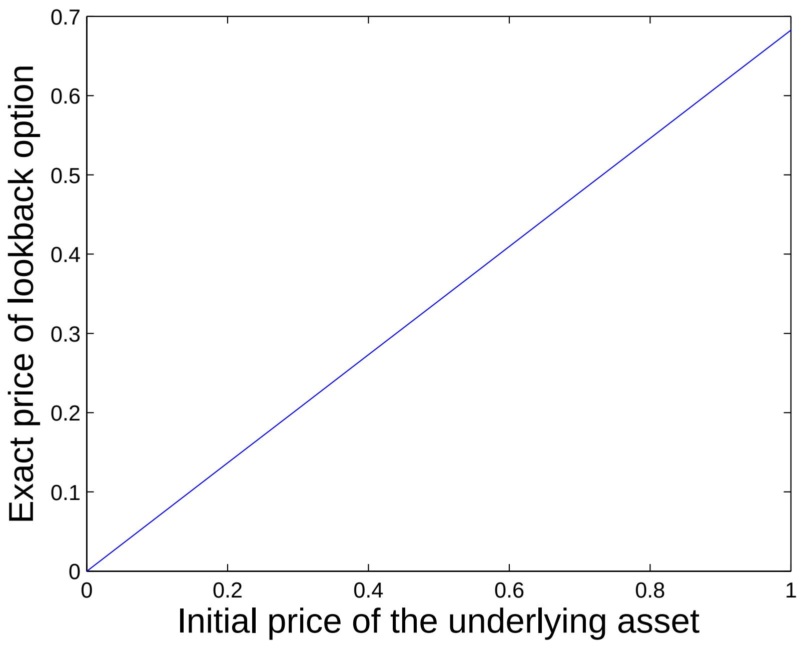

Figure 15: The exact option price plotting against initial asset price taking values from 0 to 1 and increasing by 0.1.

simulated at certain points, and the point with the maximum value may not be sampled. Hence, the simulated maximum value is no bigger than the exact maximum. This also explains why when we increased the number of time intervals the numerical solution became closer to the exact solution. Because there are more points being simulated along the path, then the numerical maximum becomes closer to the real maximum. When the number of simulations were increased, the error also becomes smaller since more paths have been simulated and the average value of all the paths becomes closer to the real mean value. The observation we found in figure 17 is that oscillation appears when the increment of initial option price get decreased. The reason is obvious, more points get plotted in the graph, so more details are shown. The simulated path of option price is supposed to show the randomness since it comes from a geometric Brownian motion which is a random process.

6 Processes with Jumps

Although the majority of models used in derivatives pricing assume that the underlying assets have continuous sample paths, many studies have found evidence of the importance of jumps in prices and have advocated the inclusion of jumps in pricing

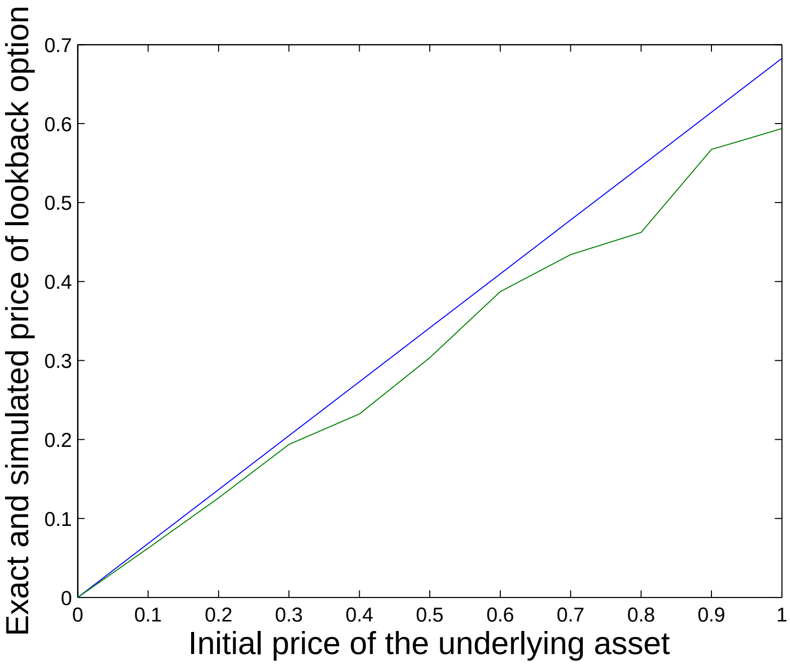

Figure 16: Green line: the simulated option price plotting against initial asset price taking values from 0 to 1 and increasing by 0.1; The number of time intervals is 200 and number of simulations is 1000. Blue line: the exact option price.

models. Compared with a normal distribution, the logarithm of a price process with jumps has a high peak and heavy tails, features typical of market data and this is called leptokurtotic [2]. In this section, we will discuss a simple model with jumps - jump-diffusion Model.

6.1 A Jump-Diffusion Model

Merton [8] firstly introduced and analyzed the models with both jump and diffusion terms for the pricing of derivative securities. The jumps are interpreted by him as shocks that affecting an individual company but not the market as a whole.
The jump-diffusion model can be specified through the SDE

$$\frac{dS(t)}{S(t-)} = \mu dt + \sigma dB(t) + dJ(t) \tag{52}$$

where μ and σ are constants, B is a sBM and J is a process independent of B with piecewise constant sample paths. In particular, J is given by

$$J(t) = \sum_{j-1}^{N(t)} (Y_j - 1) \tag{53}$$

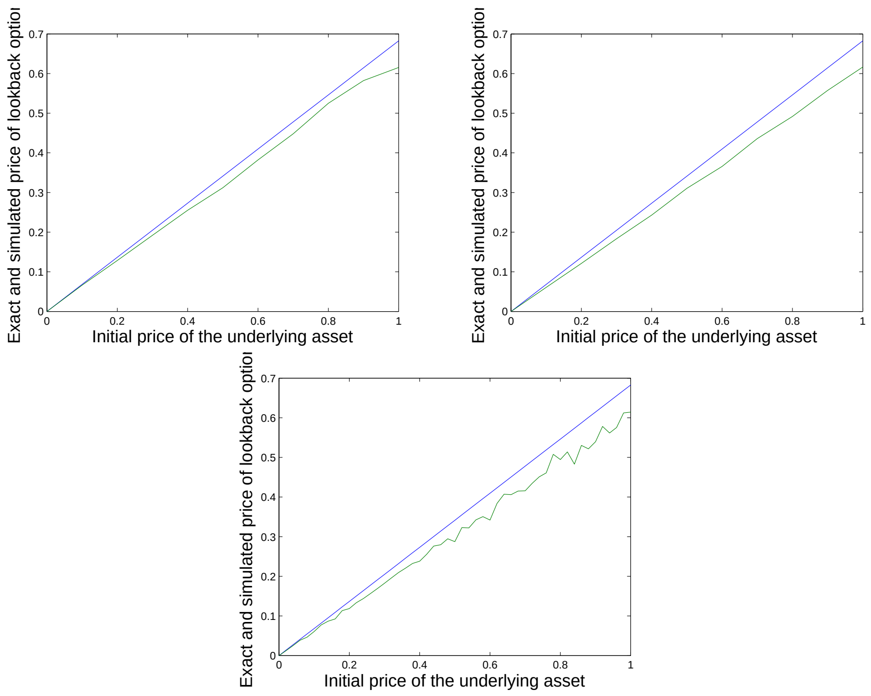

Figure 17: The first graph is with the condition: N=1000, M=1000, dx=0.1; The second graph is with the condition: N=200, M=10000, dx=0.1; The third graph is with the condition: N=200, M=1000, dx=0.02

where $Y_1, Y_2, \ldots$ are random variables and $N(t)$ is a counting process. This means that there are random arrival times $0 < \tau_1 < \tau_2 < \cdots$ and $N(t) = \sup\{n : \tau_n \le t\}$ counts the number of jumps in $[0, t]$. The symbol $dJ(t)$ in (52) stands for the jump in J at time t. The size of this jump is $Y_j - 1$ if $t = \tau_j$ and 0 if t does not coincide with any of the τ_j [5].

If there is a jump happening at time t, we need to specify whether the notation $S(t)$ means the values of S just before the jump or just after the jump. We follow the assumption that the processes are continuous from the right, so

$$S(t) = \lim_{u \downarrow t} S(u)$$

includes the effect of any jump at t. Denote the value just before a potential jump (from the left) by $S(t-)$, which is the limit

$$S(t-) = \lim_{u \uparrow t} S(u).$$

If we write (52) as

$$dS(t) = \mu S(t-)dt + \sigma S(t-)dB(t) + S(t)dJ(t), \tag{54}$$

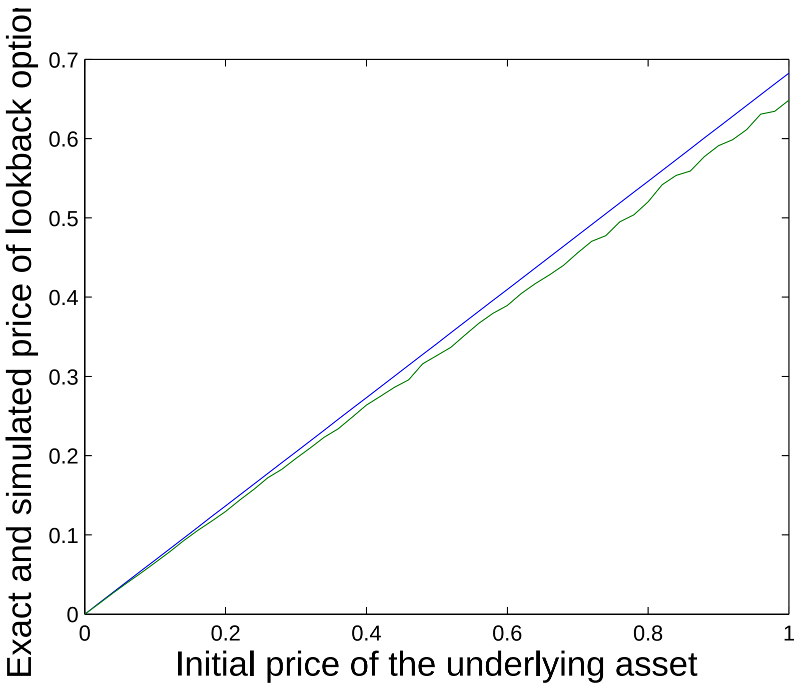

Figure 18: Green line is with the condition: N=10000, M=10000, dx=0.02; Blue line: the exact option price.

we see that the increment $dS(t)$ in S at t depends on the value of S just before a potential jump at t and not on the value just after the jump. This is as it should be. The jump in S at time t is $S(t) - S(t-)$. This is 0 unless J jumps at t, which is to say unless $t = \tau_j$ for some j. The jump in S at τ_j is

$$S(\tau_j) - S(\tau_j-) = S(\tau_j-)[J(\tau_j) - J(\tau_j-)] = S(\tau_j-)(Y_j - 1), \tag{55}$$

therefore, $S(\tau_j) = S(\tau_j-)Y_j$.

From this we can see that the Y_j are the ratios of the asset price before and after a jump - the jumps are multiplicative. This also explains why we wrote $Y_j - 1$ rather than Y_j in (53).

If we restrict Y_j to be positive random variables, then $S(t)$ can never be negative. In such a case,

$$\log S(\tau_j) = \log S(\tau_j-) + \log Y_j, \tag{56}$$

so the jumps are additive in the logarithm of the price. The solution of (52) is given by

$$S(t) = S(0)e^{(\mu - \frac{1}{2}\sigma^2)t + \sigma B(t)} \prod_{j=1}^{N(t)} Y_j, \tag{57}$$

which evidently generalizes the corresponding solution for geometric Brownian motion.

Now, we introduce the distribution to the jump process $J(t)$. Consider the simplest model which takes $N(t)$ to be a Poisson process with rate λ. This makes the inter-arrival times $\tau_{j+1} - \tau_j$ independent with a common exponential distribution,

$$P(\tau_{j+1} - \tau_j \leq t) = 1 - e^{-\lambda t}, t \geq 0. \tag{58}$$

We further assume that the Y_j are i.i.d. and independent of N and W. Under these assumptions, J is called a *compound Poisson process*.

If $Y_j \sim LN(a, b^2)$, i.e. are lognormally distributed so that $\log Y_j \sim N(a, b^2)$, then for any fixed n,

$$\prod_{j=1}^{n} Y_j \sim LN(an, b^2 n). \tag{59}$$

It follows that, conditional on $N(t) = n$, $S(t)$ has the distribution of

$$\begin{aligned} S(0)e^{(\mu - \frac{1}{2}\sigma^2)t + \sigma B(t)} \prod_{j=1}^{n} Y_j &\sim S(0) \cdot LN((\mu - \frac{1}{2}\sigma^2)t, \sigma^2 t) \cdot LN(an, b^2 n) \\ &= LN(\log S(0) + (\mu - \frac{1}{2}\sigma^2)t + an, \sigma^2 t + b^2 n). \end{aligned} \tag{60}$$

The drift parameter in (52) is that $\mu = r - \lambda m$, where r is the risk-free rate and $m = \mathbb{E}[Y_j] - 1$ (derivation please see [2], Sec. 3.5). Thus, (52) can be rewritten as

$$\frac{dS(t)}{S(t-)} = r dt + \sigma dB(t) + [dJ(t) - \lambda m dt], \tag{61}$$

the last two terms on the right are martingales and the net growth rate in $S(t)$ is indeed r.

6.2 Simulating at Fixed Dates

In this method we simulate the process at a fixed set of dates $0 = t_0 < t_1 < \cdots < t_n$ without explicitly distinguishing the effects of the jump and diffusion terms. We continue to assume that N is a Poisson process, that $Y_1, Y_2, \ldots$ are i.i.d., and that N, W, and $Y_1, Y_2, \ldots$ are mutually independent. Here, we do not assume that the Y_j are lognormally distributed since that will constitute a special case and we generate a more general case here.

Generalize (57) into

$$S(t_{i+1}) = S(t_i)e^{(\mu-\frac{1}{2}\sigma^2)(t_{i+1}-t_i)+\sigma[B(t_{i+1})-B(t_i)]} \prod_{j=N(t_i)+1}^{N(t_{i+1})} Y_j, \tag{62}$$

with the usual convention that the product over j is equal to 1 if $N(t_{i+1}) = N(t_i)$. We now write this into a more preferable form. This is for computational convenience. Let $X(t) = \log S(t)$, then

$$X(t_{i+1}) = X(t_i) + (\mu - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma[B(t_{i+1}) - B(t_i)] + \sum_{j=N(t_i)+1}^{N(t_{i+1})} \log Y_j. \tag{63}$$

this recursion replaces products with sums and we can exponentiate simulated values of the $X(t_i)$ to produce samples of the $S(t_i)$.

The algorithm of this method, is that:
1. generate standard normal random variable $Z \sim N(0,1)$;
2. generate $N \sim Poisson(\lambda(t_{i+1} - t_i))$, if $N = 0$, then $M = 0$ and go to step 4;
3. generate $\log Y_1, \ldots, \log Y_N$ from their common distribution, where $Y_1, ..., Y_N$ are random variables, and set $M = \log Y_1 + \cdots + \log Y_N$;
4. use the recursion

$$X(t_{i+1}) = X(t_i) + (\mu - \frac{1}{2}\sigma^2)(t_{i+1} - t_i) + \sigma\sqrt{t_{i+1} - t_i}Z + M. \tag{64}$$

This method relies on two properties of the Poisson process: the increment $N(t_{i+1}) - N(t_i)$ has a Poisson distribution with mean $\lambda(t_{i+1} - t_i)$, and it is independent of increments of N over $[0, t_i]$.

Depending on different distributions of Y_j, the third step in the above algorithm can be simplified into different expressions. For example, if the Y_j have the lognormal distribution $LN(a, b^2)$, then step 3 becomes:
3'. generate $Z_2 \sim N(0,1)$; set $M = aN + b\sqrt{N}Z_2$.

Now we see the implementation of this method in Matlab. Assume the Y_j are lognormally distributed. Set $a = 0$ and $b = 2$, i.e. $\log Y_j$ have mean value 0 and standard variation of 2. Besides, set the Poisson coefficient λ be 2. To start with the program, we also need the previous assumptions: total time interval $T = 5$, number of time subintervals $n = 500$, drift parameter $\mu = 0.5$, volatility coefficient $\sigma = 1$, and the initial value $X(0) = 1$.

```
% set all the coefficients and initial value
n=500; % number of subintervals
T=5; dt=T/n; mu=0.5; sigma=1;
lambda=2;  a=0; b=2;
X(1)=1;
% define every time point
for i=1:n+1
    t(i)=(i-1)*dt;
end
% main loop of simulating X(t)
for i=1:n
    N=poissrnd(lambda*dt); % generate Poisson random variable with
                             coefficient lambda*dt
    M=0; %the situation when N=0
    %the situation when N is not 0
    if N~=0
        M=a*N+b*sqrt(N)*randn; %M has lognormal distribution
    end
    %recursion of X
    X(i+1)=X(i)+(mu-0.5*sigma^2)*dt+sigma*sqrt(dt)*randn+M;
end
% to plot X against t
figure
plot(t,X,'-k')
xlabel('t','FontSize',16)
ylabel('X(t)','FontSize',16,'Rotation',90)
% to plot S against t
figure
plot(t,exp(X),'-k')
xlabel('t','FontSize',16)
ylabel('S(t)','FontSize',16,'Rotation',90)
```

From figure 19, we can see some obvious jumps in both of the graphs. The left graph shows some negative values, since it is the graph of $X(t)$ that is determined

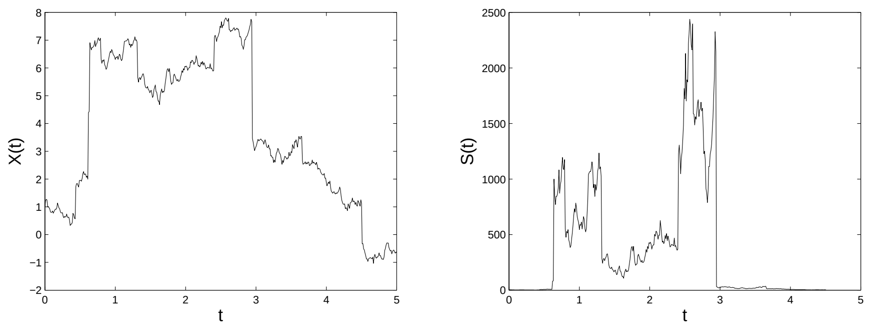

Figure 19: The graphs of $X(t)$ and $S(t)$ simulating at fixed dates when $a = 0, b = 2$ and $\lambda = 2$.

by (64). However the values of right graph (graph of $S(t)$) keeps non-negative in that $S(t) = \exp(X(t))$ and this is the real price path. By adjusting the values of a, b and λ. It has been found from the outcomes that the value of a mainly controls the sign of the jumps since a give the mean value of Y_j (the ratios of asset price before and after the jumps). If a takes value at 0, then there is half of the possibility that the jumps are going downwards and half of the possibility that the jumps are going upwards; if a is positive, then the jumps are more likely to go up than down, vise versa. b controls the size of the jumps. The bigger the b is, the bigger the amplitudes of the jumps are. Last, the number of jumps in a certain time period depends on how big λ is, and this is also a positive relationship.

6.3 Simulating Jump Times

The previous method does not identify the times at which $S(t)$ jumps. It only generates the total number of jumps in each time interval $(t_i, t_{i+1}]$, using the fact that the number of jumps has a Poisson distribution. However in this section, we simulate the jump times $\tau_1, \tau_2, ...$ explicitly.

$S(t)$ will behave like an ordinary geometric Brownian motion from one jump time to the next since we have assumed that B and J in (57) are independent of each other. Therefore, it follows that, conditional on the times $\tau_1, \tau_2, ...$ of the jumps,

$$S(\tau_{j+1}-) = S(\tau_j) e^{(\mu - \frac{1}{2}\sigma^2)(\tau_{j+1} - \tau_j) + \sigma[B(\tau_{j+1}) - B(\tau_j)]}, \tag{65}$$

and

$$S(\tau_{j+1}) = S(\tau_{j+1}-) Y_{j+1}. \tag{66}$$

Simulate $X(t)$ like in last method and we get

$$X(\tau_{j+1}) = X(\tau_j) + (\mu - \frac{1}{2}\sigma^2)(\tau_{j+1} - \tau_j) + \sigma[B(\tau_{j+1}) - B(\tau_j)] + \log Y_{j+1}. \quad (67)$$

Now, the algorithm has the following steps:

1. generate R_{j+1} from the exponential distribution with mean $\frac{1}{\lambda}$;
2. generate $Z_{j+1} \sim N(0, 1)$;
3. generate $\log Y_{j+1}$;
4. set $\tau_{j+1} = \tau_j + R_{j+1}$ and use recursion

$$X(\tau_{j+1}-) = X(\tau_j) + (\mu - \frac{1}{2}\sigma^2)R_{j+1} + \sigma\sqrt{R_{j+1}}Z_{j+1}, \quad (68)$$

$$X(\tau_{j+1}) = X(\tau_{j+1}-) + \log Y_{j+1}. \quad (69)$$

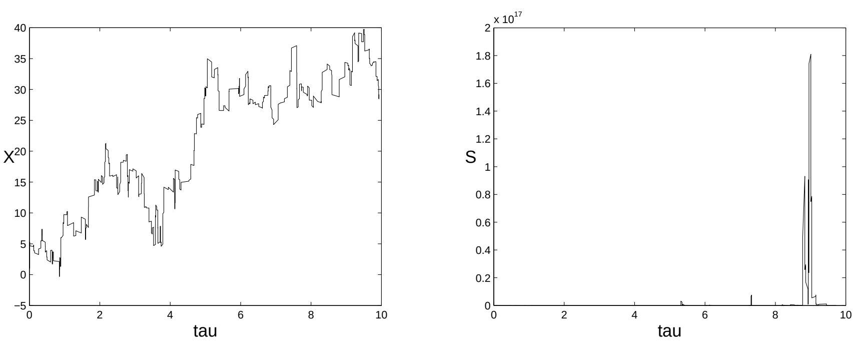

Figure 20: The graphs of $X(t)$ and $S(t)$ simulating jump times when $a = 0, b = 1$ and $\lambda = 20$.

Here again, we assume the Y_j are lognormally distributed. The program is similar with simulating at fixed dates, but is independent from the total time interval and also the time increments. Therefore, the figures are plotted with price process against jump times τ. (See Appendix B for the Matlab code.)

This algorithm gives the results as in figure 20. We see jumps everywhere in the $X - \tau$ graph. There is a high peak in the right graph of figure 20 since the right graph shows the exponentiation of the left one, therefore small increment in the left graph could lead to really large value to the real price path. The growth rate of τ (the jump time) is controlled by the exponential random variable R. R is determined by the exponential distribution coefficient λ. The larger λ is, the slower the τ increases. In other words, λ controls the expected value of the time between two consecutive jumps. So the larger λ is, the shorter time between the jumps is and we then expect that the final time will be smaller.

The two methods we have discussed above can be combined together. For example, suppose we fix a date t in advance that we would like to include among the simulated dates. Suppose it happens that $\tau_j < t < \tau_{j+1}$ (i.e., $N(t) = N(t-) = j$). Then

$$S(t) = S(\tau_j) e^{(\mu - \frac{1}{2}\sigma^2)(t-\tau_j) + \sigma[B(t) - B(\tau_j)]}, \tag{70}$$

and

$$S(\tau_{j+1}) = S(t) e^{(\mu - \frac{1}{2}\sigma^2)(\tau_{j+1} - t) + \sigma[B(\tau_{j+1}) - B(t)]} Y_{j+1}. \tag{71}$$

Both approaches to simulating the basic jump-diffusion process (52) simulating the number of jumps in fixed subintervals and simulating the times at which jumps occur can be useful at least as approximations in simulating more general jump-diffusion models.

7 Conclusion

In this report, three kinds of stochastic processes have been simulated - the ordinary Brownian motion, geometric Brownian motion and processes with jumps. These processes are very common ones in financial engineering. They are used to describe the prices in various of financial derivatives. All these simulations were implemented to the discrete case.

We started with sampling the paths of ordinary Brownian motion. There were three constructions being discussed. The first and most straightforward one is called random walk construction. It just simply used the distribution of Brownian motion and n standard normal random variables to sample the value point by point in the time order. The Brownian bridge construction breaks this time order. It always generates the midpoint between the start and end point. Therefore, it gives us greater control over the coarse structure of the simulated Brownian path. The construction is based on the conditional distribution between the point that is going be to sampled and the points that have already been sampled, and uses Markov property to simplify this conditional distribution between two points. The algorithm becomes more complicated since it uses the joint distribution and refers to the order of generating midpoints. The last construction introduced was principal components construction. It takes advantage of the property of the covariance matrix by using its eigenvalues and eigenvectors. It is based on the idea that most variabilities of Brownian motion can be explained by the first few largest eigenvalues. i.e. most information of a Brownian path can be established on the first maybe 4 or 5 largest eigenvalues. This construction produces a way of using less random variables to simulate a path, whereas in the first two methods, n points depend on n random variables.

Then we talked about geometric Brownian motion which is a more broadly used process in the financial market. Since it has the property of non-negativity that just reflects the property of the assets price. In such case, it is widely applied to describe the price path of financial derivatives like stock price, option price, etc. Geometric Brownian motion is just the exponentiation of ordinary Brownian motion. Therefore, it is lognormally distributed. The methods of constructing the paths of geometric Brownian motion are the same with the ordinary Brownian motion. Our interests stay on option pricing by geometric Brownian motion. The option pricing stands on the base of risk-neutral measure. We listed two examples - the Asian option and the lookback option. They are differentiated by the payoffs. In order to price the option, we took the average value over a large amount of paths and discounted the averaged final payoff to the initiation. After implementing the Asian option by Matlab, we found that the option price converges which is expected. However, the Asian option does not have an analytical solution which we could compare our numerical solution with. Then we came to the lookback option which has an exact solution, therefore is testable. The way of programming the lookback option is very similar to the Asian option with the exception of the payoff. In the figures, we plotted the path of the option price against the number of simulations. Comparing this with the exact solution, the result is fairly good. The numerical option price is convergent and converges to the exact solution. The graphs of option price plotting against the initial option price were also given. They showed that the error between numerical and exact option price becomes bigger as the initial value increases. But we can prove the accuracy by adjusting the parameters in the program, like increasing the number of simulations, reducing the length of time increment, etc.

The last section briefly introduced the processes with jumps. This is a more intuitive view of financial markets, with allowance for larger moves in asset prices caused by sudden world events. A typical jump process is the jump-diffusion model. Being different from the previous processes, there is an additional term added to the expression, called the jump term. The values just before and after the jump are different but multiplicative, which means there exists a jump ratio between the two values. The way of simulating such models can be divided into two approaches. One is simulating at fixed dates, another is simulating jump times. The first one does not distinguish the effects of the jump and diffusion terms. The second one simulates the jump times explicitly. Both of them got the result of random processes with jumps in them like the figures showed.

In the future work, we can continue the discussion of processes with jumps, extending it to the pure-jump process. This report just gives a framework on the topics that are being discussed. Every topic could be extended to the continuous and multi-dimensional cases.

References

[1] Wikipedia (2014): *http://en.wikipedia.org/wiki*

[2] Glasserman P. (2003): *Monte Carlo Method in Financial Engineering*, Chap.2, Chap.3

[3] Lorinczi J. (2014): *Lecture Notes to Stochastic Calculus and Theory of Pricing*

[4] Karatzas I., Shreve S. (1991): *Brownian Motion and Stochastic Calculus*

[5] Zhao H.Z. (2013): *Lecture Note of Introduction to Measure Theory and Martingales*

[6] Shreve S. (2004): *Stochastic Calculus in Finance II*

[7] Feng C.R. (2013): *Lecture Notes of Stochastic Models in Finance*

[8] Merton, R.C. (1976): *Option pricing when underlying stock returns are discontinuous* Journal of Financial Economics 3:125-144.

[9] Cont R., Tankov P. (2003): *Financial Modelling with Jump Processes*

A Appendix

```
randn('state',12) % Set initial state for repeatability;
M=1000; % number of simulations
N=100; % number of subintervals
T=1; dt=T/N;
mu=0.5; % drift parameter
r=mu; % drift parameter is the risk-free rate
sigma=1; % volatility parameter
% define time points
for i=1:N+1
    t(i)=(i-1)*dt;
end
% define an array to VOM - the option price
VOM=[];
x=0:0.1:1;
% main loop of getting the option price
for x0=0:0.1:1;
    S(1)=x0; % give initial value to GBM
    % calculate the expected payoff every time of simulation
    for m=1:M
        % simulate the paths for GBM
        for i=1:N
            S(i+1)=S(i)*exp((mu-0.5*sigma^2)*(t(i+1)-t(i))
                   +sigma*sqrt(t(i+1)-t(i))*randn);
        end
        S_max(m)=max(S);
        VT(m)=max(0,S_max(m)-S(N+1)); % expected payoff
    end
    % calculate the averaged discounted payoff
    VOM=[VOM exp(-r*T)*sum(VT(1:M))/M];
end
% plot the option price against initial value
figure
plot(x,VOM)
```

B Appendix

```
randn('state',12)% Set initial state for repeatability;
% give values to all the parameters
n=200; % number of subintervals
mu=0.5; sigma=1;
lambda=20;
a=0; b=2;
tau=0; X=1;
temp=[0]; % this is for saving jump times
% main loop for sampling the jump path
for i=1:n
    %generate exponential distributed random variables
    R=exprnd(1/lambda);
    %generate the jump which is lognormally distributed
    M=a+b*randn;
    %generate jump times
    tau=tau+R;
    %define a time array for saving two values at jump times
    temp=[temp, tau, tau];
    %the value just before the jumps
    X_temp=X(end)+(mu-0.5*sigma^2)*R+sigma*sqrt(R)*randn;
    %define the process with values just before and after the jumps
    X=[X, X_temp, X_temp+M];
end
%plot the figures
figure
plot(temp,X,'-k')
xlabel('tau','FontSize',16)
ylabel('X','FontSize',16,'Rotation',0)

figure
plot(temp,exp(X),'-k')
xlabel('t','FontSize',16)
ylabel('S','FontSize',16,'Rotation',0)
```