**Лабораторна робота №3**

*Моделювання випадкових процесів*

Нехай - ймовірнісний простір. Функція  називається випадковим процесом, якщо при кожному фіксованому ,  є випадковою величиною, визначеною на ймовірнісному просторі .

Нехай випадковий процес приймає значення в множині *Х*, тобто .

, де -σ-алгебра підмножин з *Х*, - називається фазовим простором випадкового процесу .

Найчастіше роль параметру  - це час.

При фіксованому ,  є лиш функцією часу і її називають реалізацією чи траєкторією випадкового процесу.

1. **Моделювання вінерового процесу**.

Вінеровий процес  є математичною моделлю броунівського руху (неперервного хаотичного руху мікроскопічної частинки в рідкому середовищі)

При моделюванні вінерового процесу слід використати наступний факт:

причому, випадкові величини і незалежні, і .

Отже, зручно скористатись наступною формулою

Де – нормально розподілена випадкова величина з середнім о і дисперсією .

Розглянемо інтервал і розіб’ємо його на під інтервали однакової довжини:

Перепишемо формулу (1) в наступному вигляді

Заважимо, що в лабораторній роботі №1 вже вивчалось питання моделювання випадкової величини .

На графіку представлені траєкторії вінерового процесу на інтервалі з кроком .



З графіку також видно, що функція  обмежує можливі значення вінерового процессу.

1. **Моделювання динаміки вартості акцій**

Розглянемо модель вартості акції, що описується стохастичним диференціальним рівнянням

З початковою умовою , –сталі (коефіцієнт волатильності та коефіцієнт дрейфу).

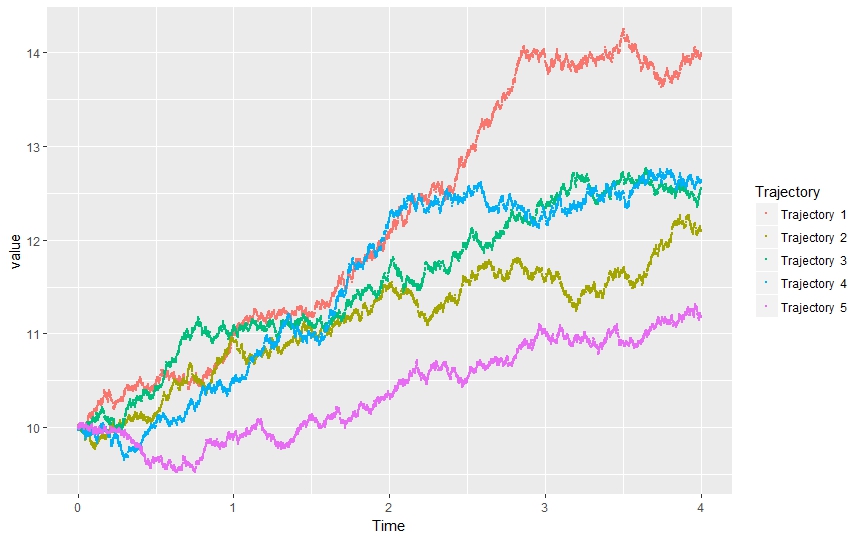
Розвязком даного рівняння є випадковий процес

S. (2)

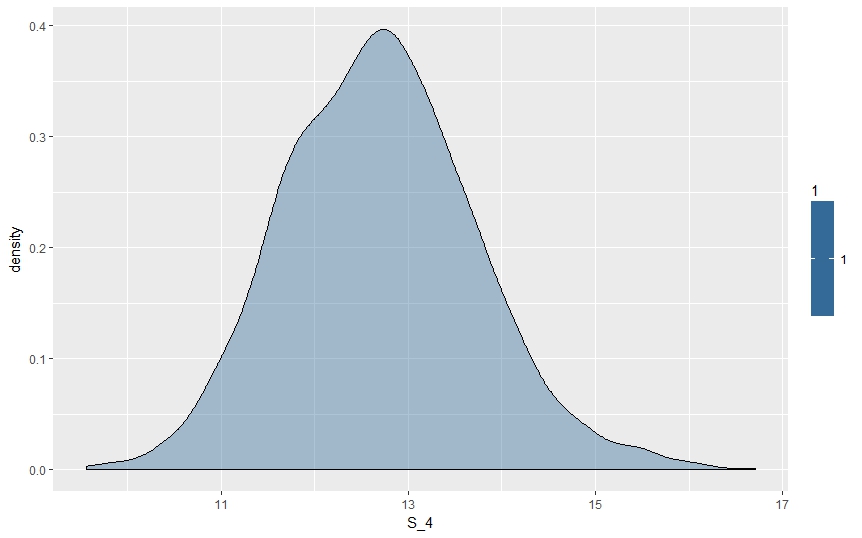
З формулою (2) ви вже працювали в лабораторній роботі №1 (завдання 4).

Можна подати розвязок (2) за допомогою наступної рекурентної формули

описане вище.



Розглянемо розподіл випадкової величини S(4), побудований за траєкторіями



1. **Моделювання Пуассонового процесу**

Нехай подія А може відбуватись в будь-який випадковий момент часу, нехай -це кількість появ події за проміжок часу довжиною . Процес - це однорідний процес з незалежними приростами, причому



Такий процес  називається процесом Пуассона.

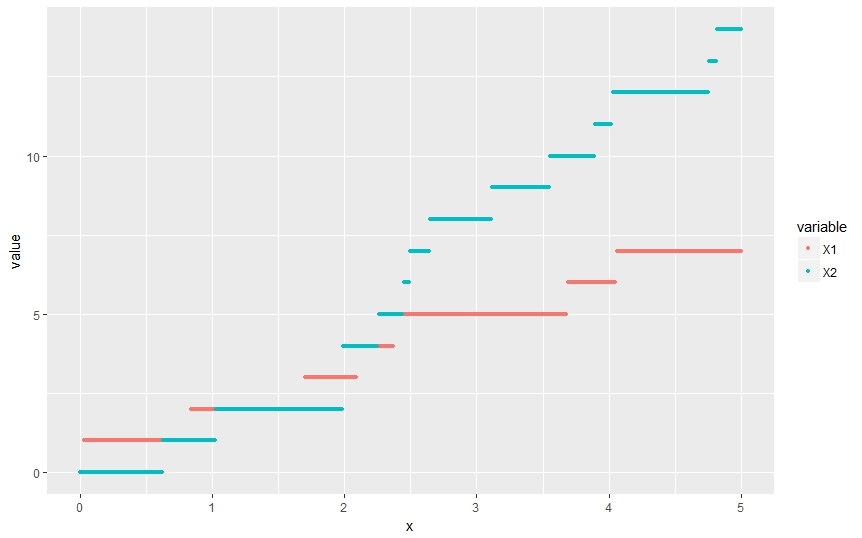
Для процесу Пауссона має місце наступний результат



Початкове значення пуассонового процесу дорівнює 0, а далі у випадкові моменти часу його значення збільшуватиметься щоразу на 1.

При моделюванні реалізацій пуассонового процесу слід використати той факт, що інтервали між моментами зміни значень випадкового процесу:  є незалежними експоненціально розподіленими випадковими величинами з параметром λ.

Результат моделювання можна розглянути на графіку нижче



**Завдання до лабораторної роботи**

1. Побудувати 10 траєкторій вінерового процесу на одному графіку і функцію .
2. Змоделювати ціну вартості акціїї (побудувати декілька траєкторій процесу (2)), використовуючи ті ж значення сталих, як і в прикладі, розглянутому вище. Побудувати розподіл випадкової величини S(4).
3. Побудувати 10 траєкторій процесу пуассона з параметром λ=вашому порядковому номеру.

**Завдання на додаткові бали**

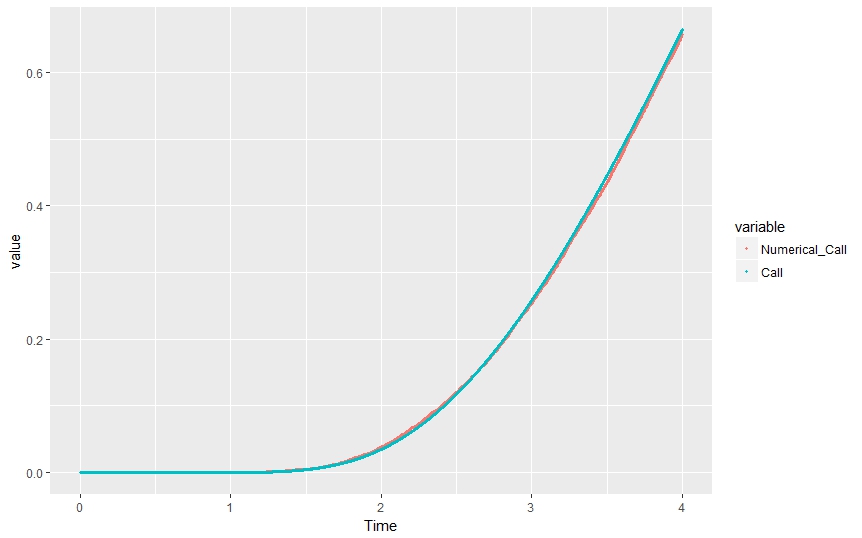
Consider call option, price for which are define from next formula:

where strike price for option, – expiry date, – value of stock price in time . Using this formula and value of the from 2., calculate value of the option prise for . Compare result with standard formula

where

– commulative distribution function for standard normal distribution:

Using numerical (with 2000 trajectories) and exact formula for calculation price, compare vales of price of Call option. In result you nmust to get next figure:



Code in R:

######## 1

It = 5000;

L = 9;

mu = 5000;

U = seq(10,150000,100)

Prob\_ban = c()

B = c()

for (i in 1:It){

N = rpois(1,L)

Sample = rexp(N,1/mu)

B[i] = sum(Sample)

#print(Sample)

}

for (j in 1: length(U)){

u = U[j]

Prob\_ban[j] = sum(B>u)/It

}

Prob\_ban

plot(U,Prob\_ban, col = 'blue', xlab = 'Initial capital', ylab = 'Probability bankr.', type = 'l')

abline(0,0)

######## 1

set.seed(2)

Time = 3000

h=0.4

Iter = 8

RNORM = rnorm(floor(Iter\*Time/h))\*sqrt(h)

M = matrix(RNORM, nrow = Iter)

for (i in 1:(dim(M)[1])){

M[i, ] = cumsum(M[i,])

}

DATA = data.frame(seq(h,Time,h), t(M))

names(DATA)[1] = 'Time'

DATA\_long <- melt(DATA, id="Time")

ggplot(data=DATA\_long, aes(x=Time, y=value, colour=variable)) +

geom\_point(size = 0.1)

LL2 <- function(x){

n = length(x)

for (i in 1:n){

y[i] = sqrt(2\*x[i]\*log(log(max(2.8, x[i]))))

}

return(y)

}

y = LL2(seq(h, Time, h))

z = -y

DATA = data.frame(seq(h,Time,h), t(M), z, y)

N = dim(DATA)[1]

names(DATA)[1] = 'Time'

DATA\_long <- melt(DATA, id="Time")

DATA\_long$variable = as.character(DATA\_long$variable)

for (i in 1:Iter){

DATA\_long$variable[N\*(i-1)+seq(1,N)] = rep(paste('Trajectory ', as.character(i)), N)

}

DATA\_long$variable[N\*(Iter)+seq(1,N)] = rep('-sqrt(2\*t\*log(log(t)))', N)

DATA\_long$variable[N\*(Iter+1)+seq(1,N)] = rep('sqrt(2t\*\*log(log(t)))', N)

DATA\_long$variable = as.factor(DATA\_long$variable)

ggplot(data=DATA\_long, aes(x=Time, y=value, colour=variable)) +

geom\_line()

## 2

S0 = 10

r = 0.06

sigma = 0.04

Time = 4

h=0.001

Iter = 2000

RNORM = rnorm(floor(Iter\*Time/h))\*sqrt(h)

M = matrix(RNORM, nrow = Iter)

for (i in 1:(dim(M)[1])){

M[i, ] = cumsum(M[i,])

}

H = t(matrix(rep(seq(h,Time,h),Iter), ncol = Iter))

S = S0\*exp((r-sigma^2/2)\*H+sigma\*M)

DATA = data.frame(seq(h,Time,h), t(S[1:5,]))

names(DATA)[1] = 'Time'

N = dim(DATA)[1]

DATA\_long <- melt(DATA, id="Time")

names(DATA\_long)[2] = 'Trajectory'

DATA\_long$Trajectory = as.character(DATA\_long$Trajectory)

for (i in 1:5){

DATA\_long$Trajectory[N\*(i-1)+seq(1,N)] = rep(paste('Trajectory ', as.character(i)), N)

}

DATA\_long$Trajectory = as.factor(DATA\_long$Trajectory)

ggplot(data=DATA\_long, aes(x=Time, y=value, colour=Trajectory)) +

geom\_point(size = 0.1)

Send = data.frame(S[,Time/h])

names(Send) = 'S\_4'

a <- ggplot(Send, aes(x = S\_4))

a + geom\_area(stat = "bin", bins = 50, alpha=0.6)

a + geom\_density(aes(fill = 1), alpha=0.4)

## 4

K = 12

Time = 5

h=0.001

Iter = 200

RNORM = rnorm(floor(Iter\*Time/h))\*sqrt(h)

M = matrix(RNORM, nrow = Iter)

for (i in 1:(dim(M)[1])){

M[i, ] = cumsum(M[i,])

}

H = t(matrix(rep(seq(h,Time,h),Iter), ncol = Iter))

S = S0\*exp((r-sigma^2/2)\*H+sigma\*M)

u = pmax(Send$S\_4-K,0)

u[is.na(u)]=0

Call = exp(-r\*Time)\*mean(u)

d1 = log(S0/K+(r+sigma^2/2)\*Time)/(sigma\*sqrt(Time))

d2 = d1 - sigma\*sqrt(Time)

Call1 = S0\*pnorm(d1) - K\*exp(-r\*Time)\*pnorm(d2)

## plot call price

U = dim(S)

TT = seq(h,Time,h)

CALL\_N = c()

CALL = c()

for (i in 1:U[2]){

u = pmax(S[,i]-K,0)

u[is.na(u)]=0

CALL\_N[i] = exp(-r\*TT[i])\*mean(u)

d1 = log(S0/K+(r+sigma^2/2)\*TT[i])/(sigma\*sqrt(TT[i]))

d2 = d1 - sigma\*sqrt(TT[i])

CALL[i] = S0\*pnorm(d1) - K\*exp(-r\*TT[i])\*pnorm(d2)

}

DATA = data.frame(TT, CALL\_N,CALL)

names(DATA) = c('Time', 'Numerical\_Call', 'Call')

DATA\_long <- melt(DATA, id="Time")

ggplot(data=DATA\_long, aes(x=Time, y=value, colour=variable)) +

geom\_point(size = 1)

## 3

lambda = 2

Iter = 2;

set.seed(7)

x = seq(0, 5, 0.01)

M = matrix(NA, nrow = length(x), ncol = Iter)

for (j in 1:Iter){

tau = cumsum(rexp(30, lambda))

tau

n = length(tau)

y = c()

for (i in 1: length(x)){

y[i] = sum(tau<x[i])

}

M[,j] = y

}

DATA = data.frame(x, M)

DATA\_long <- melt(DATA, id="x")

ggplot(data=DATA\_long, aes(x=x, y=value, colour=variable)) +

geom\_point(size = 1)