论文实证部分

实现文献中采用的短期利率模型

代码中所指的公式1和2

$$\mathrm{dr_t} = \kappa_1(\mathrm{b_t} - \bar{\kappa}\lambda - \mathrm{r_t})\mathrm{dt} + \gamma\sqrt{\mathrm{r_t}}\mathrm{dW_{1t}} + (\mathrm{e}^{\mathrm{Z_t}} - 1)\mathrm{r_t}\mathrm{dN_t} \quad \text{公式1}$$

$$\mathrm{db_t} = \kappa_3(\theta - \mathrm{b_t})\mathrm{dt} + \eta_2\sqrt{\mathrm{b_t}}\mathrm{dW_2t} \quad \text{公式2}$$

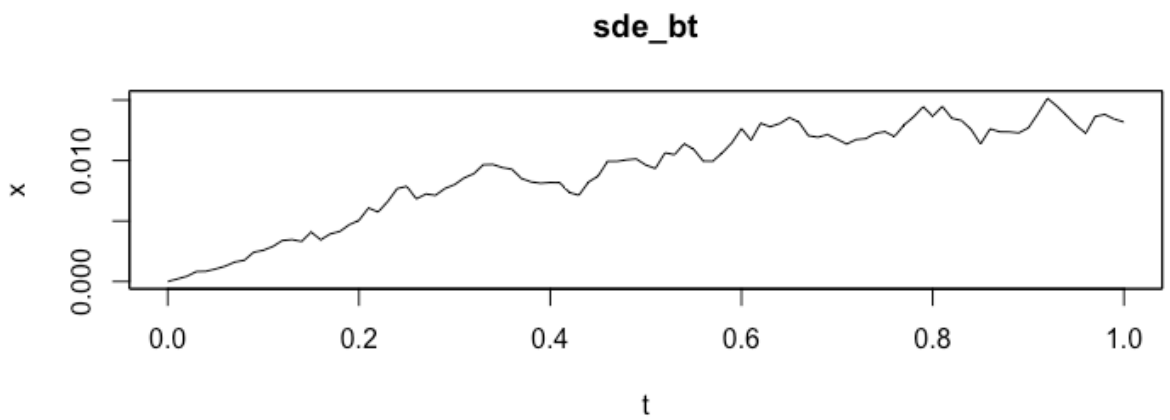
step1:实现公式（2），画出b_t

注：$\kappa_3 = 0.3319, \theta = 0.0624, \eta_2$=0.056

yuima代码如下

```
set.seed(2)
library(yuima)
mod1<-setModel(drift="0.3319*(0.0624-x)",diffusion="0.056*sqrt(x)")
b_t<-simulate(mod1)
plot(b_t,main="sde_bt")
```

图如下



自己写：

```
b_t <- rep(NaN, n)
b_t[1]<-0
for (i in 1:n){
  b_t[i+1]=b_t[i]+0.3319*(0.0624-b_t[i])*delt+0.056*sqrt(b_t[i])*sqrt(delt)*rnorm(1,0,1)
}
plot(b_t,type="l") ##图名吧b_t
```

图如下：

参考

> 2.3.1 Diffusion processes
>
> $$\mathrm{dX_t} = -3\mathrm{X_t dt} + \frac{1}{1+\mathrm{X_t^2}}\mathrm{dW_t}$$

```
library(yuima)
mod1<-setModel(drift="-3*x",diffusion="1/(1+x^2)")
X<-simulate(mod1)
plot(X)
```

Q：

sde和yuima画的图不一样

step2:实现公式（1）

$$\mathrm{dr_t} = \kappa_1(\mathrm{b_t} - \bar{\kappa}\lambda - \mathrm{r_t})\mathrm{dt} + \gamma\sqrt{\mathrm{r_t}}\mathrm{dW_{1t}} + (\mathrm{e^{Z_t}} - 1)\mathrm{r_t dN_t}$$

1. Z_t

$$Z_t \cong N(\mu_j, \sigma_j^2)$$
$$\mu_j = 0\,,\ \sigma_j = 0.0196$$

```
Z_t<-rnorm(N,0,0.0196)
```

1. $\bar{\kappa}$

$$\bar{\kappa} = E[e^{Z_t} - 1] = e^{(\mu_j + \frac{1}{2}\sigma_j^2)} - 1$$

```
kappa_bar<-exp(0+0.5*0.0196^2)-1
```

1. 跳跃

$$\mathrm{dr_t} = \kappa_1(\mathrm{b_t} - \bar{\kappa}\lambda - \mathrm{r_t})\mathrm{dt} + \gamma\sqrt{\mathrm{r_t}}\mathrm{dW_{1t}} + (\mathrm{e^{Z_t}} - 1)\mathrm{r_t dN_t}$$

$\kappa_1 = 0.68$,λ=10,γ=0.03

```
for (i in 1:n)
{
  jump<-rnorm(1,mean = 0,sd=1) ##跳跃的参数？暂且先如此
  x[i + 1] <- x[i] + 0.68*(b_t[i]-kappa_bar*10-x[i])*delt+0.03*sqrt(x[i])*sqrt(delt)*rnorm(1,0,1)+(exp(Z_t[i])-1)* x[i]*J[i] * jump
  jumps[i]<-J[i] * jump
}
```

```
#设定序列起点和终点以及跳跃强度
start = 0
end = 10
lambd <- 10 #根据文献是10

#分割区间数、漂移项、方差项
n = 1000
mu = 0.75
variance = 0.7

#窗宽
#h = 1000 ^ (-0.2)

#生成的正太随机数的均值与方差
meanofrn = 0
sdofrn = 1

#为了避免出现序列为负的情况，将序列起始值调整为10
startvalue = 10

#序列在起始点的值，即股票的初始值，还有参数lamdha的值
lambdaofN <- 20

#步长delt
delt <- (end - start) / n

r <- rnorm(n, mean = meanofrn, sd = sdofrn)
x <- rep(NaN, n)
xspot <- seq(start, end, length.out = n + 1)


#生成在区间内发生跳跃的次数
N <- as.numeric(rpois(1, lambda = lambdaofN))

#生成发生跳跃的点在区间内的坐标
jumpspot <- sort(runif(N, start, end))
plot(jumpspot)

#用cut函数生成区间按照总的区间数分隔后的区间序列interval
```

```
interval <- as.data.frame(cut(xspot, xspot)[-1])
colnames(interval) <- "interval"

#用cut函数找出发生跳跃点所在的区间，它的长度等于发生跳跃的次数N
jumpinterval <- as.data.frame(cut(jumpspot, xspot))
#colnames(jumpinterval) <- "jumpinterval"

#将发生跳跃的区间和对应的跳跃点合并
jumpintervalandspot <- cbind(jumpinterval, jumpspot)
colnames(jumpintervalandspot) <- c("interval", "jumpspot")

#用merge函数将数据框interval和jumpintervalandspot按照他们的共同变量interval合并，参数
all=T是为了保留所有的数据
#intervalandjump <-merge(interval,jumpintervalandspot,by.x = "interval",by.y =
"interval",all = T)
intervalandjump <-merge(interval,jumpintervalandspot,by= "interval",all = T)
#将intervalandjump第二列的缺失值用零代替
intervalandjump[, 2][is.na(intervalandjump[, 2])] <- 0

#生成的序列J表示在1的时候发生跳跃，在0时不跳跃，？ 1表示不跳跃，0表示跳跃
J <- ifelse(!intervalandjump[, 2] == 0, 1, 0)

#生成画板的默认参数配置opar,opar <- par(no.readonly = TRUE)用来更改当前变量环境
opar <- par(no.readonly = T)
# par(mfrow=c(2,1))
# par(mfrow=c(1,1))
plot(jumpspot)
plot(J)
plot(intervalandjump[, 2])

#初始化序列的起始值
x[1] <- startvalue

#以下是产生跳跃序列的代码
#jump为跳跃幅度，它设置为一个服从正太分布的随机变量
#分布的标准差为前一个股票价格的百分之十的绝对值的三分之一
#这样设定可以让99.74%的跳跃值都控制在振幅的百分之十以内，
#如果jump超过10%的振幅，就以百分之十代替

#变量jumps用来记录发生跳跃的区间以及跳跃值
jumps<-c()

for (i in 1:n)
{
  jump<-rnorm(1,mean = 0,sd=(abs(x[i]*0.1))/3)
  jump<-ifelse(abs(jump)>abs(x[i]*0.1),sign(jump)*abs(x[i]*0.1),jump)
  x[i + 1] <- x[i] + mu * delt + sqrt(variance) * 0.1 * r[i] + J[i] * jump
  jumps[i]<-J[i] * jump
}
```

```
plot(x)
plot(x, type = 'l',ylim = c(0,30))
plot(x, type = 'l')
```