Assignment 8

YUYU ZHANHG

10/30/2020

PART 1 For O-U process, the following stochastic process works

$$dX_t = \theta(\mu - x_t)dt + \sigma W_t$$
$$\theta > 0 \ \mu \ (Mean \ value) \ \sigma > 0$$

are all parameters

$$W_t$$

are Brownian Motion.
We now want to show how this happen:

The first order differential equation:

$$\frac{dy}{dx} + p(x)y = Q(x)$$

Integration in both side:

$$\int_0^t dX(u) = -\int_0^t \theta X(u)du + \int_0^t \theta\mu du + \int_0^t \sigma dW(u)$$

both sides multiplied by $e^{\theta u}$:

$$\int_0^t e^{\theta u}dX(u) = -\int_0^t e^{\theta u}\theta X(u)du + \int_0^t e^{\theta u}\theta\mu du + \int_0^t e^{\theta u}\sigma dW(u)$$

LHS equal to:

$$\begin{aligned} LHS &= e^{\theta u}X(u) \mid_{u=0}^{t} - \int_0^t e^{\theta u}\theta X(u)du \\ &= e^{\theta u}X(t) - X(0) - \int_0^t e^{\theta u}\theta X(u)du \end{aligned}$$

And similarly, RHS equal to:

$$\begin{aligned} RHS &= -\int_0^t e^{\theta u}\theta X(u)du + \int_0^t \mu de^{\theta u} + \int_0^t e^{\theta u}\sigma dW(u) \\ &= -\int_0^t e^{\theta u}\theta X(u)du + \mu(e^{\theta u} - 1) + \int_0^t e^{\theta u}\sigma dW(u) \end{aligned}$$

Then: LHS = RHS we get:

$$e^{\theta u}X(t) - X(0) - \int_0^t e^{\theta u}\theta X(u)du = -\int_0^t e^{\theta u}\theta X(u)du + \mu(e^{\theta u} - 1) + \int_0^t e^{\theta u}\sigma dW(u)$$

$$e^{\theta u}X(t) - X(0) = \mu(e^{\theta u} - 1) + \int_0^t e^{\theta u}\sigma dW(u)$$

$$\Rightarrow X(t) = e^{-\theta u}(X(0) - \mu) + \mu + \int_0^t e^{-\theta(t-u)}\sigma dW(u)$$

Then , if we just assume that: time start from t to $t + \Delta$, the equation will become:

$$X(t + \Delta) = e^{-\theta\Delta}X(t) + \mu - \mu(e^{-\theta\Delta}) + \frac{\sigma}{\sqrt{2\theta}}(\sqrt{1 - e^{-2\theta\Delta}}Z)$$

Which is equivalent to the textbook:

$$X(t + \Delta) = e^{-\alpha\Delta}X(t) + b(1 - (e^{-\alpha\Delta})) + \frac{\sigma}{\sqrt{2\alpha}}(\sqrt{1 - e^{-2\alpha\Delta}}Z)$$

where Z N(0,1).

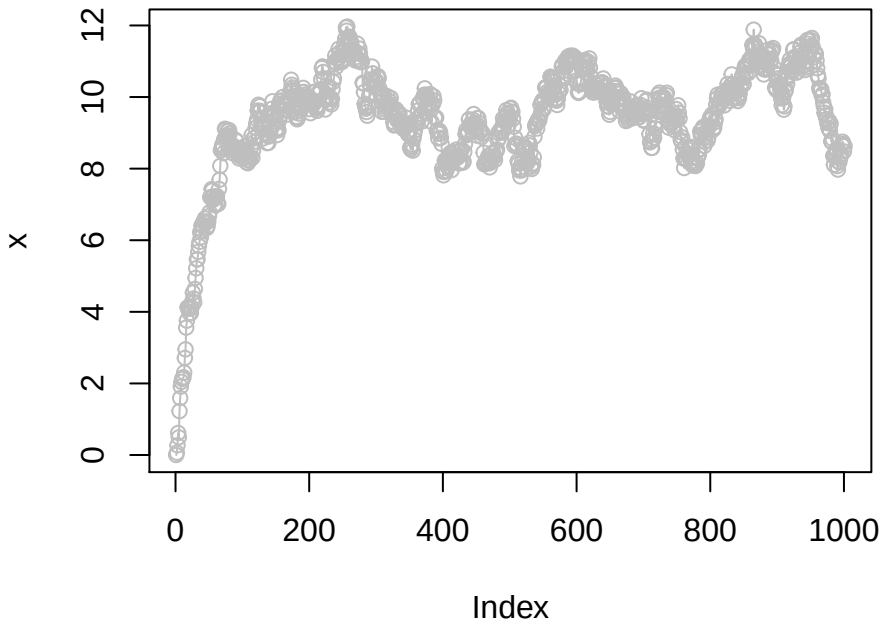
PART 2 Use the transition distribution from the last part to implement a random walk construction for the process on time interval[0,T]

after confirm the parameter, we see how the price change with O-U process

```
sigma <- 1.  # Standard deviation.
mu <- 10.  # Mean.
tau <- 0.05  # Time constant.
dt<- .001  # Time step.
T <- 1  # Total time.
n <- (T / dt)  # Number of time steps.
t=seq(0,1,by=dt)  # Vector of times.
sigma_bis = sigma * sqrt(2./tau)
sqrtdt = sqrt(dt)

x=vector()
x[1]=0
for(i in 1:1000){
 NW <-x[i]+dt*(-(x[i]-mu)/tau)+sigma_bis*sqrtdt*rnorm(1,mean=0, sd=1)
  i<-i+1
  x<- append(x,NW)
}


plot(x,type = "o", col = "grey")
```

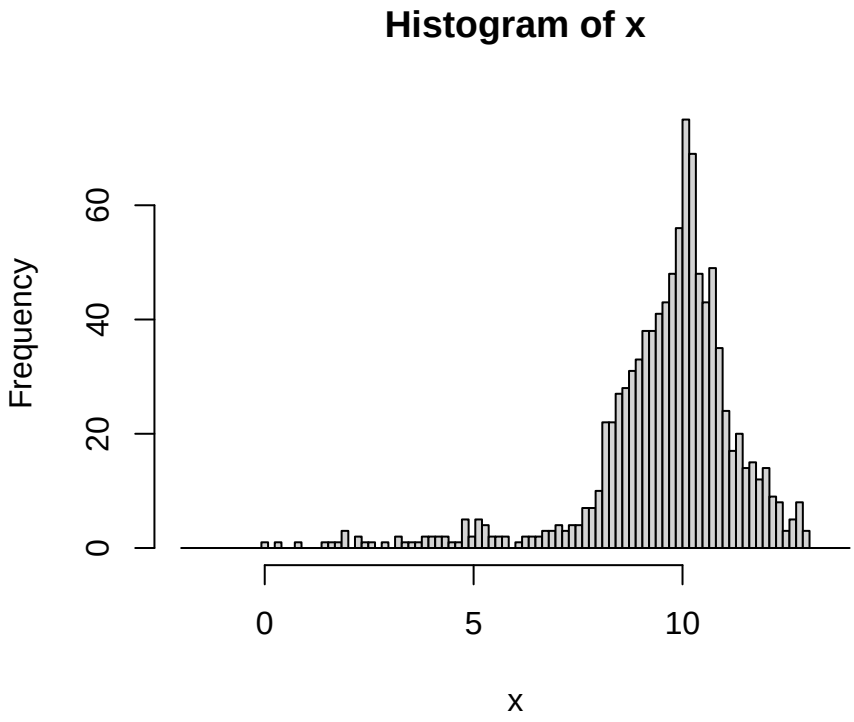


Next we will make simulation

```
sigma <- 1.  # Standard deviation.
mu <- 10.  # Mean.
tau <- 0.05  # Time constant.
dt<- .001  # Time step.
T <- 1  # Total time.
n <- (T / dt)  # Number of time steps.
t=seq(0,1,by=dt)  # Vector of times.
sigma_bis = sigma * sqrt(2./tau)
sqrtdt = sqrt(dt)


ntrials = 10000
X = vector()
# We create bins for the histograms.
bins = seq(from = -2.,to = 14.,  by = 0.16)

x=vector()
x[1]=0
for(i in 1:1000){
 NW <-x[i]+dt*(-(x[i]-mu)/tau)+sigma_bis*sqrtdt*rnorm(1,mean=0, sd=1)
  i<-i+1
  x<- append(x,NW)
}
hist(x,
   breaks = bins)
```



Part 3 to simulation:

```
r <- -1 # growth / decay

sigma <- 0.01 # relative

b <- 3 # initial


M <- 100 # number of steps

T <- 1 # maximum time
h <- T/M # time step
t <- seq( length = M +1 , from =0 , by= h ) # t is the vector


X <- array (0 , c( M +1) ) # place to store


N <- 30*( M +1) # number of steps

p <- 0.5
S <- array (0 , c( N +1) )
rw <- cumsum ( 2 *( runif ( N ) <= p ) -1 )
S [2:( N +1) ] <- rw

 WcaretN <- function ( z ) {
```

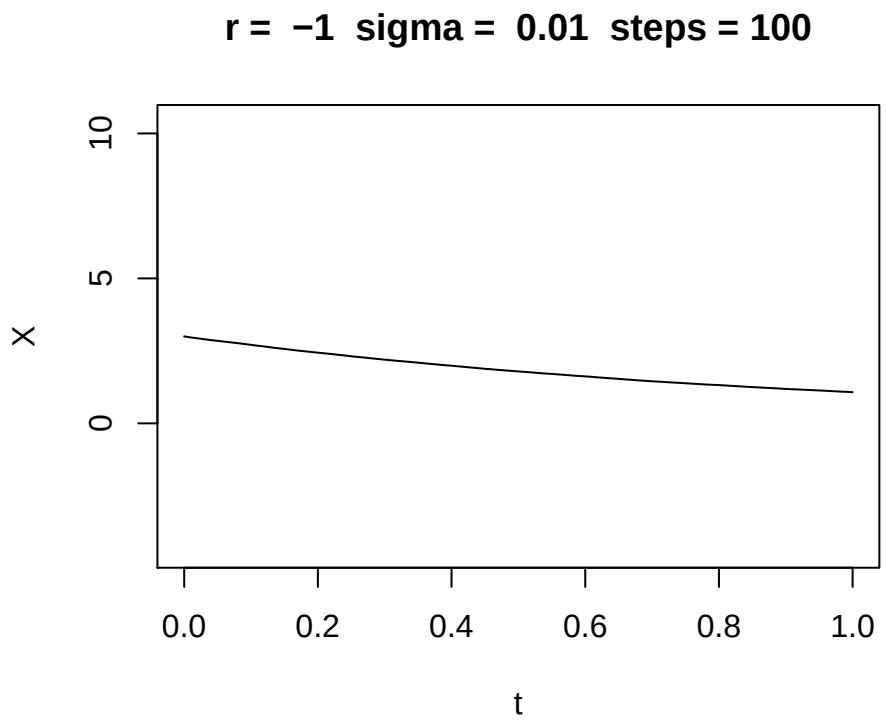
```
 Delta <- T/N

 prior = floor ( z/ Delta ) + 1
 subsequent = ceiling ( z/ Delta ) + 1

retval <- sqrt ( Delta )*( S [ prior ] + (( z/ Delta +1) -
prior )*( S [ subsequent ] - S [ prior ]) )
 }

 X [1] <- b
 for ( i in 1: M ) {
 X [ i +1] <- X [ i ]+ r*X [ i ]*h + sigma *X [ i ]*( WcaretN (t[ i ]+ h ) -
WcaretN (t[ i ]) )
 }

 plot (t,X ,"l", xlim =c(0 , T ) , ylim =c( X [1] - exp (abs ( r )*T +1) ,
X [1]+ exp(abs( r )*T +1) ) )
 title( main = paste ("r = ", r , " sigma = ", sigma , " steps =",
M ) )
```



To show the value of r(1)

```
library(INLA)
```

```
## Loading required package: Matrix
## Loading required package: sp
## Warning: package 'sp' was built under R version 4.0.3
## Loading required package: parallel
```

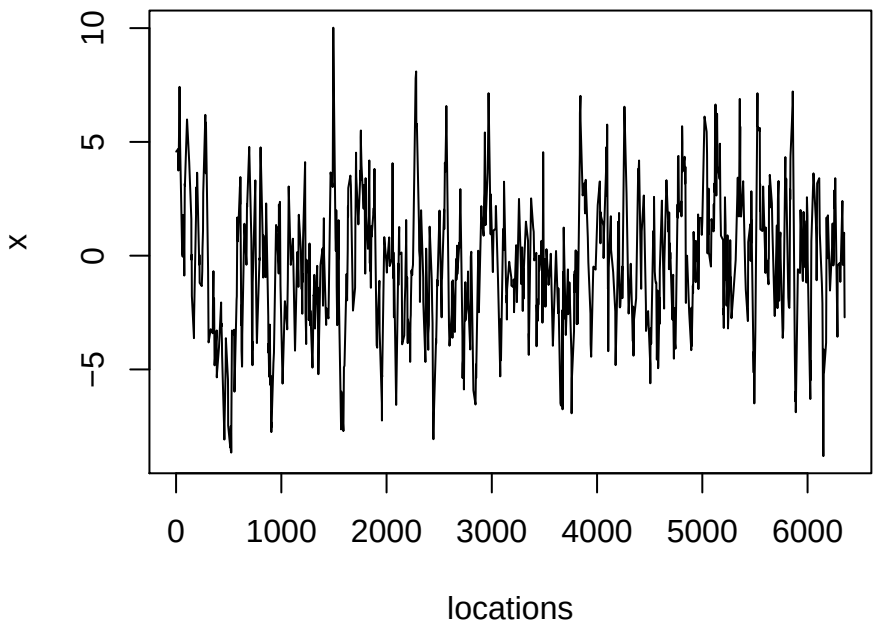
```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 4.0.3
```

```
## This is INLA_20.03.17 built 2020-10-31 15:12:30 UTC.
## See www.r-inla.org/contact-us for how to get help.
```

```
## simulate an OU-process and estimate its parameters back.
phi = -log(0.95)
sigma = 1
marg.prec = 2*phi/sigma^2
n = 1000
locations = cumsum(sample(c(1, 2, 5, 20),n, replace=TRUE))

## do it sequentially and slow (for clarity)
x = numeric(n)
x[1] = rnorm(1, mean=0, sd = sqrt(1/marg.prec))
for(i in 2:n) {
delta = locations[i] - locations[i-1]
x[i] = x[i-1] * exp(-phi * delta) +
rnorm(1, mean=0, sd = sqrt(1/marg.prec * (1-exp(-2*phi*delta))))
}
## observe it with a little noise
y = 1 + x + rnorm(n, sd= 0.01)
plot(locations, x, type="l")
```

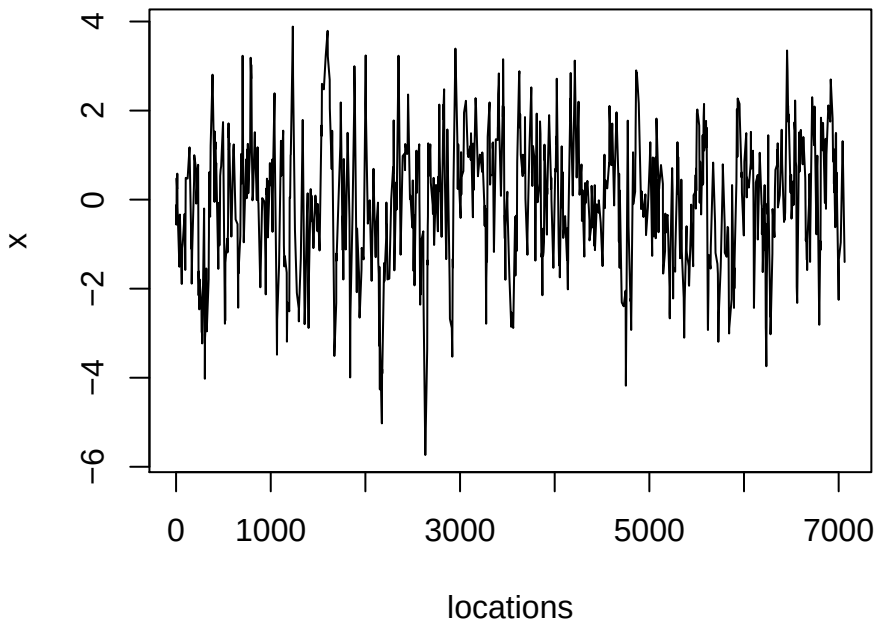



```
formula = y ~ 1 + f(locations, model="ou", values=locations)
r = inla(formula, data = data.frame(y, locations))
summary(r)
```

```
##
## Call:
##    "inla(formula = formula, data = data.frame(y, locations))"
## Time used:
##     Pre = 1.07, Running = 1.1, Post = 0.248, Total = 2.41
## Fixed effects:
##             mean    sd 0.025quant 0.5quant 0.975quant mode kld
## (Intercept) 0.69 0.243      0.211     0.69      1.169 0.69   0
##
## Random effects:
##   Name     Model
##     locations Ornstein-Uhlenbeck model
##
## Model hyperparameters:
##                                           mean       sd 0.025quant 0.5quant
## Precision for the Gaussian observations 1.97e+04 1.89e+04   1215.527 1.41e+04
## Precision for locations                 1.04e-01 9.00e-03      0.087 1.04e-01
## Phi for locations                       5.40e-02 5.00e-03      0.045 5.40e-02
##                                         0.975quant     mode
## Precision for the Gaussian observations   6.97e+04 3239.486
## Precision for locations                   1.21e-01    0.104
## Phi for locations                         6.50e-02    0.054
##
## Expected number of effective parameters(stdev):  999.90(0.139)
## Number of equivalent replicates : 1.00
##
## Marginal log-Likelihood:  -1935.42
```

```
library(INLA)
## simulate an OU-process and estimate its parameters back.
phi = -log(0.95)
sigma = 0.5
marg.prec = 2*phi/sigma^2
n = 1000
locations = cumsum(sample(c(1, 2, 5, 20),n, replace=TRUE))

## do it sequentially and slow (for clarity)
x = numeric(n)
x[1] = rnorm(1, mean=0, sd = sqrt(1/marg.prec))
for(i in 2:n) {
delta = locations[i] - locations[i-1]
x[i] = x[i-1] * exp(-phi * delta) +
rnorm(1, mean=0, sd = sqrt(1/marg.prec * (1-exp(-2*phi*delta))))
}
## observe it with a little noise
y = 1 + x + rnorm(n, sd= 0.01)
plot(locations, x, type="l")
```




```
formula = y ~ 1 + f(locations, model="ou", values=locations)
r = inla(formula, data = data.frame(y, locations))
summary(r)
```

```
##
## Call:
##    "inla(formula = formula, data = data.frame(y, locations))"
## Time used:
##     Pre = 0.911, Running = 1.51, Post = 0.253, Total = 2.68
## Fixed effects:
##              mean   sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 0.961 0.11      0.746    0.961      1.177 0.961   0
##
## Random effects:
##   Name     Model
##     locations Ornstein-Uhlenbeck model
##
## Model hyperparameters:
##                                           mean       sd 0.025quant 0.5quant
## Precision for the Gaussian observations 1.86e+04 1.84e+04   1273.255 1.32e+04
## Precision for locations                 4.33e-01 3.40e-02      0.370 4.32e-01
## Phi for locations                       5.90e-02 5.00e-03      0.049 5.90e-02
##                                         0.975quant     mode
## Precision for the Gaussian observations   6.71e+04 3483.133
## Precision for locations                   5.03e-01    0.429
## Phi for locations                         7.00e-02    0.058
##
## Expected number of effective parameters(stdev):  999.64(0.499)
```

```
## Number of equivalent replicates : 1.00
##
## Marginal log-Likelihood:  -1277.14
```

```
plot(density(x),ylim=c(0,0.2))
```