Problem 1

(A) The number n is the number of observations in a day. The number T is how many days we want to observe. μ is the risk premium for holding the asset for one year. σ is the volatility of the asset. The return of market index should be the weighted average of all individual assets annual return (252*μ) in the market. The market index volatility is the weighted average of the annualized σ of all assets in the market.

(B) The MATLAB code:

Listing 1. Simulation1.m

```
function [ X ] = Simulation1( n, T, mu, sigma   )

X=log(292.58)
delta=1/n
P=[exp(X)];
T_n=[0];
for t=0:1/n:T
    Z=normrnd(0,1,1);
    X=X+ mu*delta+sigma*sqrt(delta)*Z;
    S=exp(X);
    P=[P;S];
    T_n=[T_n;t];
end
plot(T_n,P)
print 2.jpg -djpeg -r600
end
```

The plot below shows the process of P, which starts from around 293 and moves up and down, driven by a standard Wiener process. Since the average risk premium is positive, the price moves upward in general.

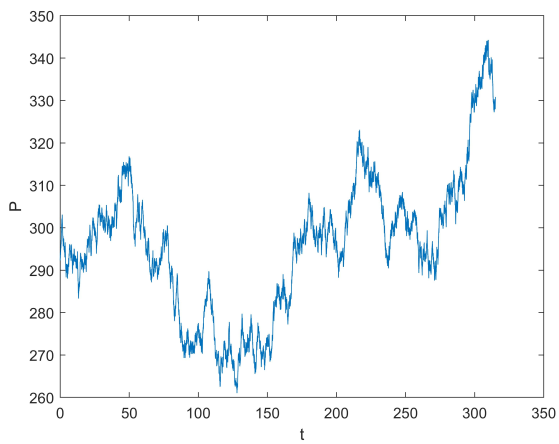


Figure 1. Price Process for 1.25*252 Days

Problem 2

(A) λ is the parameter representing the intensity of the jump of the process. σ_j is the variance of the size of each jump. σ_j is the variance of the size for each individual jump while σ is the variance of the entire process. The n in the denominator represents the number of observation each day, which transform the daily variance to each observation.

(B) The MATLAB code:

Listing 2. Simulation2.m

```
function [ J ] = Simulation1( n, T, lambda, sigma   )
S=0;
N=poissrnd(lambda*T);
U=unifrnd(0,T,[N,1]);
U=sort(U);
sigma_j=18*sqrt(sigma/n);
J=[0];
d=1;
T_n=[0];
for t=0:1/n:T
    for i=d:N
        if U(i) > t
            S=S;
            d=d;
            continue
        else
            Y=normrnd(0,sigma_j^2,1);
            S=S+Y;
            d=d+1;
        end
    end
    J=[J;S];
    T_n=[T_n;t];
end
plot(T_n,J)
N
T_n
print CompoundPoisson.jpg -djpeg -r600

end
```

The plot below shows the compound Poisson process starting from zero with intensity λ. Since the size of each jump has zero mean, we see the expectation of the process is around zero. The number of jump is 18, quite close to the theoretical average of the Poisson provess which is 18.75.

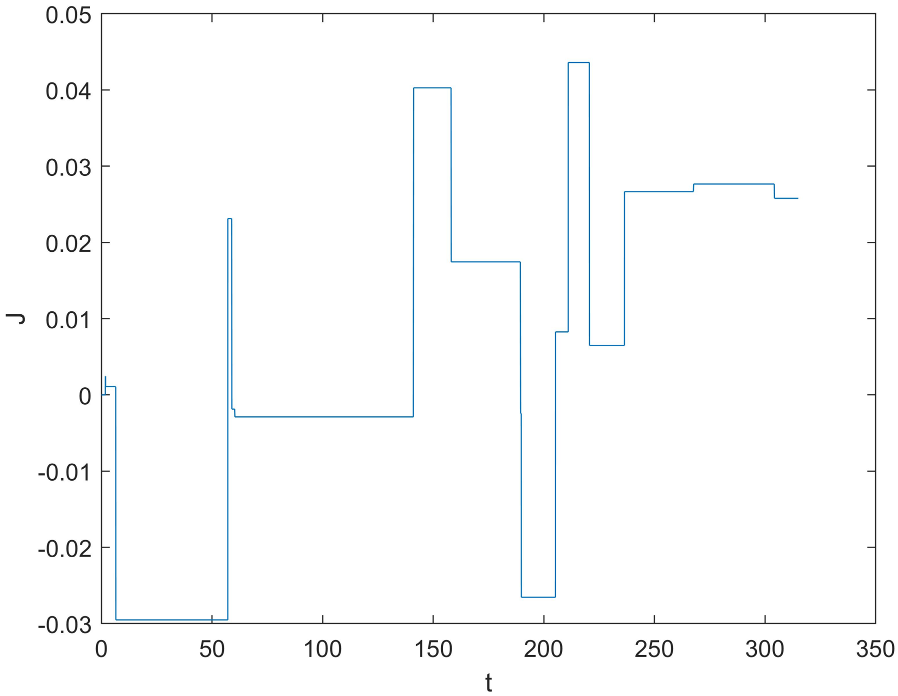


Figure 2. Compound Poisson Process for 1.25*252 Days

PROBLEM 3

(A) 1 and 4 are both correct in this context, since $X_t = \int_0^t \mu_s ds + \int_0^t \sqrt{c_s} dW_s + J_t$, which is equivalent to $X_t = \widehat{X}_t + J_t$ and it is also the same as $e^{X_t} = e^{\widehat{X}_t + J_t}$

(B) The MATLAB code:

Listing 3. Simulation3.m

```
function [ J ] = Simulation1( n, T, mu, lambda, sigma   )
X=log(292.58)
delta=1/n
P=[exp(X)];
S=0;
N=poissrnd(lambda*T);
U=unifrnd(0,n*T,[N,1]);
U=sort(U);
sigma_j=18*sqrt(sigma/n);
J=[0];
d=1;
T_n=[0];
for j=1:n*T
    r=j/n;
    if r<U(N)
        continue
    else
        break
    end
end
for t=1:1/n:r
    for i=d:N
        if U(i) > t
            S=S;
            d=d;
            Z=normrnd(0,1,1);
            X=X+ mu*delta+sigma*sqrt(delta)*Z;
            continue
        else
            Y=normrnd(0,sigma_j^2,1);
            S=S+Y;
            d=d+1;
            Z=normrnd(0,1,1);
            X=X+ mu*delta+sigma*sqrt(delta)*Z+S;
        end
    end
    a=exp(X);
    P=[P;a];
    T_n=[T_n;t];
end

for t=r:1/n:T
    Z=normrnd(0,1,1);
```

```
    X=X+ mu*delta+sigma*sqrt(delta)*Z;
    b=exp(X);
    P=[P;b];
    T_n=[T_n;t];
end
plot(T_n,P)
print JumpDiffusion.jpg -djpeg -r600
end
```

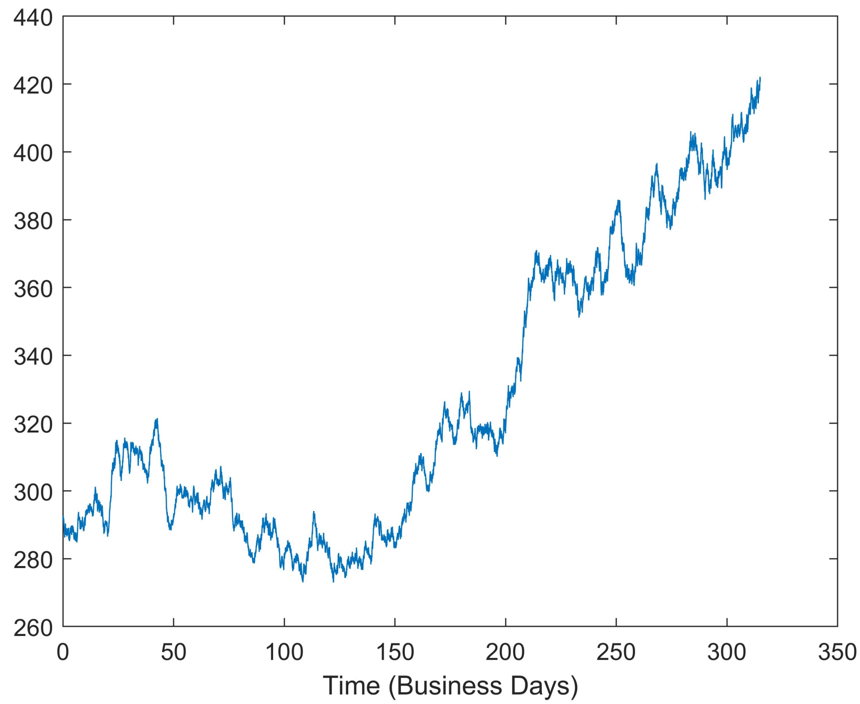


FIGURE 3. Jump Diffusion Process for 1.25*252 Days

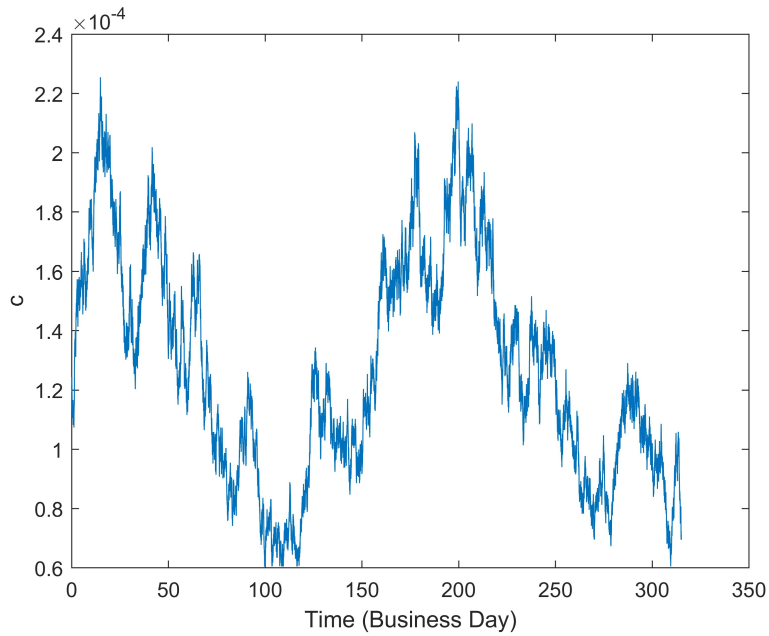
The figure simulates the jump-diffusion process. Since the risk premium is positive, we see the positive mean of the entire process. What is more, compared to the the figure in exercise 1, we see some large gaps, which are the 'jumps'.

PROBLEM 4

(a) μ_c is the average of the stochastic volatility. σ_c is the volatility of the stochastic volatility. ρ means that the expected 'extra' volatility at time $t+1$ is a portion of that at time t.

(b) The code is:

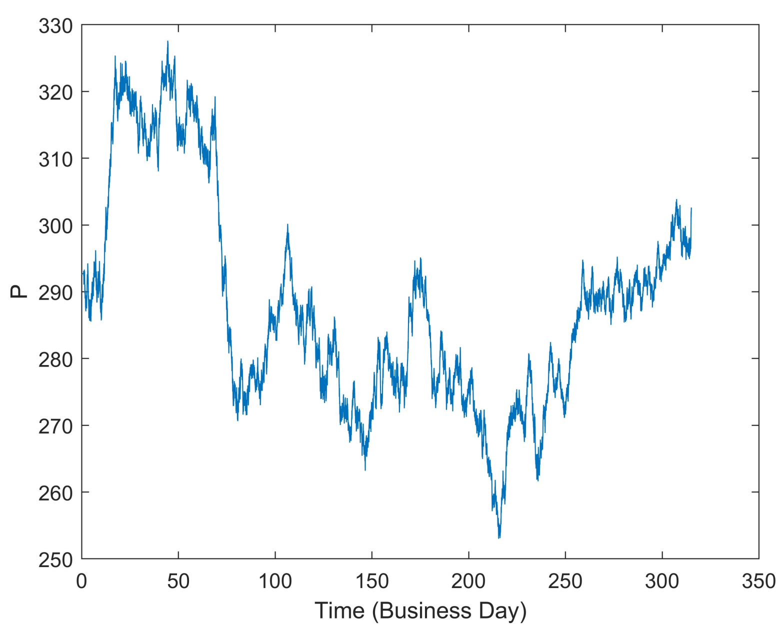
```
function [ C ] = Simulation4_1( ro,T, mu_c,n, sigma_c    )
n_e=20*n
delta_e=1/n_e
c=mu_c;
C=[c];
T_n=[0];
for t=0:1/n_e:T
    Z=normrnd(0,1,1);
    c=c+ro*(mu_c-c)*delta_e+sigma_c*sqrt(c*delta_e)*Z;
    if c< (mu_c/2)
        c=mu_c/2;
    else
        c=c;
    end
    C=[C;c];
    T_n=[T_n;t];
end
plot(T_n,C)
print C.jpg -djpeg -r600
end
```



The plot illustrates the stochastic volatility in the financial market. When it hits the bottom line at around 0.6, the volatility is approximated to $\mu_c/2$. The stochastic part is driven by a standard Wiener process Z_t .

(c) The code is:

```
function [ X ] = Simulation4_2( ro,T, mu_c,n, sigma_c     )
x=log(292.58)
n_e=20*n
delta_e=1/n_e
c=mu_c;
C=[c];
p=exp(x);
P=[p];
T_n=[0];
for t=1:1/n_e:T
    Z_1=normrnd(0,1,1);
    c=c+ro*(mu_c-c)*delta_e+sigma_c*sqrt(c*delta_e)*Z_1;
    if c< (mu_c/2)
        c=mu_c/2;
    else
        c=c;
    end
    Z_2=normrnd(0,1,1);
    x=x+sqrt(c*delta_e)*Z_2;
    p=exp(x);
    P=[P;p];
    T_n=[T_n;t];
end
plot(T_n,P)
print P.jpg -djpeg -r600
end
```



The price start from the given level and then change with stochastic volatility.

(d) The code is:

```
function [ X ] = Simulation4_2( ro,T, mu_c,n, sigma_c    )
x=log(292.58)
n_e=20*n
delta_e=1/n_e
c=mu_c;
C=[c];
p=exp(x);
P=[p];
X=[x];
T_n=[0];
for t=0:1/n_e:T
    Z_1=normrnd(0,1,1);
    c=c+ro*(mu_c-c)*delta_e+sigma_c*sqrt(c*delta_e)*Z_1;
    if c< (mu_c/2)
        c=mu_c/2;
    else
        c=c;
    end
    Z_2=normrnd(0,1,1);
    x=x+sqrt(c*delta_e)*Z_2;
    p=exp(x);
    X=[X;x];
    P=[P;p];
    T_n=[T_n;t];
end
R=[];
for j=2:n_e*T
    r=X(j)-X(j-1);
    R=[R;r];
end
plot(T_n(2:n_e*T),R)
print R.jpg -djpeg -r600
end
```

The figure of the log-return is on the next page. I cannot say the return volatility display a certain pattern. Sometimes it is positive and sometimes it is negative.

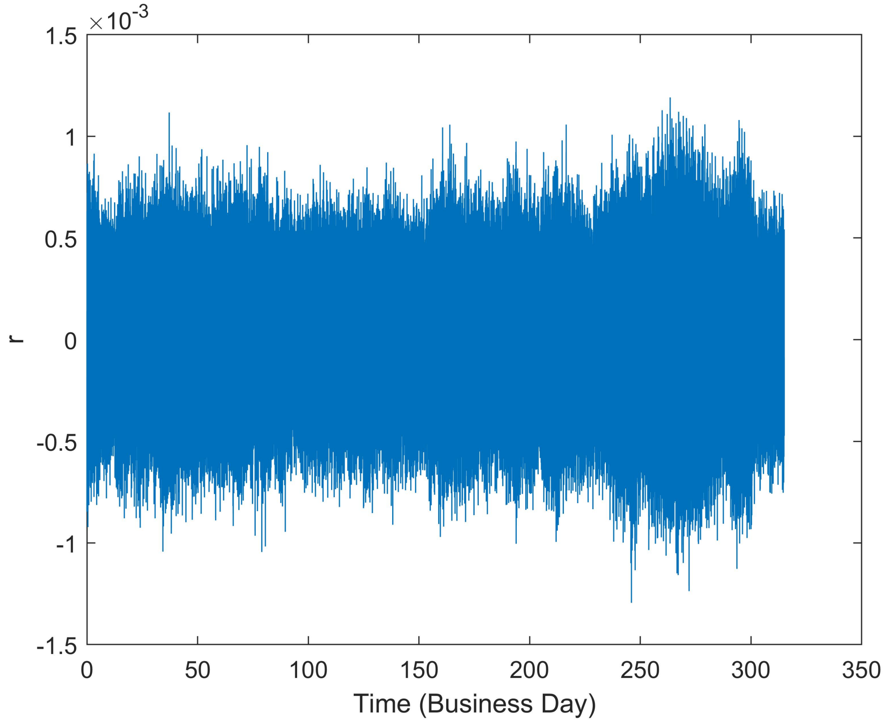


Figure 4. Log Return of the simulated process

(e) The code is

```
function [ X ] = Simulation4_2( ro,T, mu_c,n, sigma_c    )
x=log(292.58)
n_e=20*n
delta_e=1/n_e
c=mu_c;
C=[c];
p=exp(x);
P=[p];
X=[x];
T_n=[0];
for t=0:1/n_e:T
    Z_1=normrnd(0,1,1);
    c=c+ro*(mu_c-c)*delta_e+sigma_c*sqrt(c*delta_e)*Z_1;
    if c< (mu_c/2)
        c=mu_c/2;
    else
        c=c;
    end
    Z_2=normrnd(0,1,1);
    x=x+sqrt(c*delta_e)*Z_2;
    if t==fix(t)
        X=[X;x];
    else
        continue
    end
end
R=[];
for j=2:T
    r=X(j)-X(j-1);
    R=[R;r];
    T_n=[T_n;j]
end
plot(T_n(2:T),R)
print Rn.jpg -djpeg -r600
end
```

The volatility pattern does not show in this figure. The frequency is much lower than that of the former one. Typically, it becomes larger.

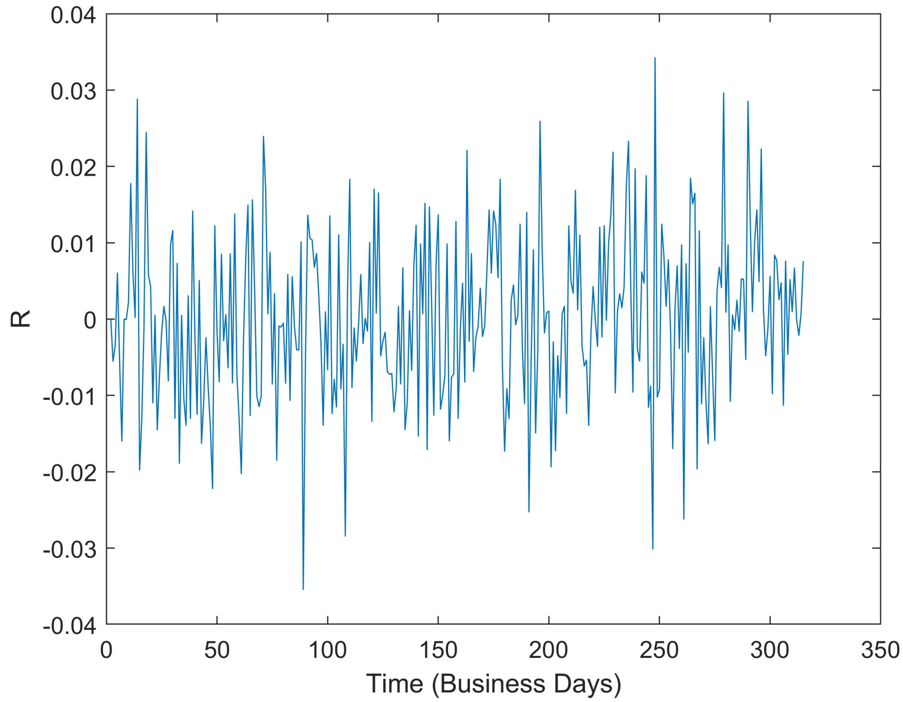


FIGURE 5. Log Return of the simulated process