Exercise 1

(A) Interprets of parameters:
n: The number of steps per unit time;
T: The total observe time;
μ: The risk premium (mean) of holding asset during a time unit (day);
σ: The standard deviation of asset price during a time unit (day).

Assuming the trading time for a year is T, then we have:

$$E(X_T^n) = T * E(X_1^n) = T\mu \tag{1}$$

$$sd(X_T^n) = \sqrt{Var(X_T^n)} = \sqrt{T * Var(X_1^n)} = \sqrt{T}\sigma \tag{2}$$

(B) In order to stimulate the continuous time process of asset price P, we need to decompose the continuous time into small time interval (step):

$$X_i = X_{i-1} + \mu\Delta_n + \sigma\sqrt{\Delta_n}Z_i \tag{3}$$

then we have:

$$X_i - X_{i-1} = \mu\Delta_n + \sigma\sqrt{\Delta_n}Z_i \tag{4}$$

$$X_n^T = \sum_{i=1}^{nT/\Delta_n} (\mu\Delta_n + \sigma\sqrt{\Delta_n}Z_i) \tag{5}$$

Here Z_i are identical independent normal distribution, that is $Z_i \sim \mathcal{N}(0,1)$ for i = 1,2 $\cdots$, nT.

To construct X_n^T:

(i) Construct a series number from the standard normal distribution;

(ii) Adding each component in equation(5);

(iii) Converting the log-price to prices.

Here is the plot of time series prices:

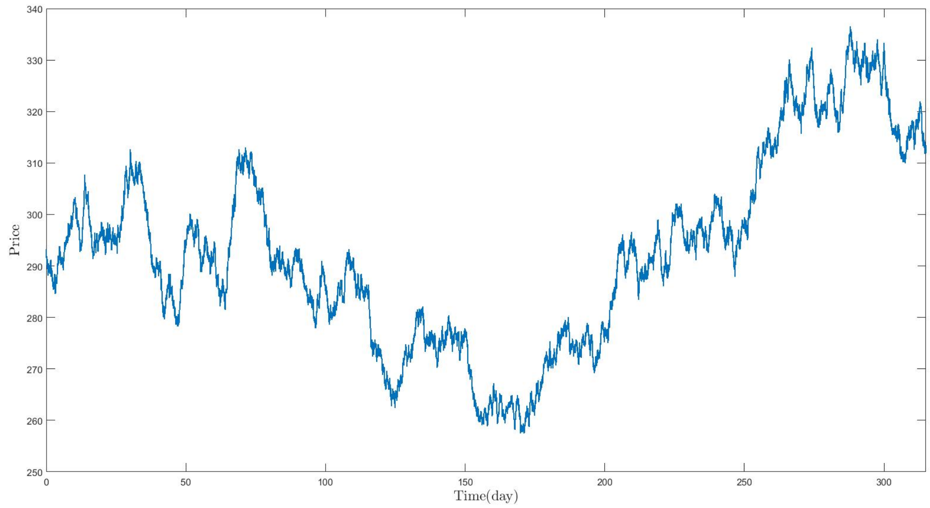

Figure 1: Time Series Prices (Gaussian Diffusion with Constant Coefficients)

From the figure, we can see the price range is (200,340) and the price fluctuates a lot.

The **MATLAB** code:

```
function [x] = GD_c(T,n,c,u,x0)
% simulate a Gaussian diffusion model with CONSTANT coefficient
% T is total observe time
% x is the array of log—price of asset between time 0~T
% x0 is the initial value of xt(at t=0)
% n is the number of observations in every unit of time
% c is the standard variance of log—price
% u is the mean of asset's log—price
    x=[];
    x(1)=x0; % initial value of x(at t=0)
    delta_t=1/n; % delta_t is the time gap between every observation
    for i=1:T*n % observation number from 1 to T*n
        z=normrnd(0,1); % construct random variable from standard normal distribution
        x(i+1)=x(i)+u*delta_t+sqrt(c*delta_t)*z; % calculate x(i+1) by iteration
    end
end
```

```
% ———EXCERCIZE #1(GAUSSIAN DIFFUSION MODEL WITH CONSTENT COEFFICIENTS)——
% SET PARAMETERS
n=80;
T=1.25*252;
u=0.03872/100;
c=0.011^2;
x0=log(292.58);

% PART B(STIMULATE MODEL)
X1=GD_c(T,n,c,u,x0) % calculate value of Xt
P1=exp(X1); % convert log-price to price
t=0:1/n:T; % construct observation time series
f1=figure('name','Time Series of Prices(GAUSSIAN DIFFUSION MODEL WITH
    CONSTENT COEFFICIENTS)');
plot(t,P1);% plot time series of prices
xlabel('Time(day)');
ylabel('Price');
xlim([0,T]);
% ————————————————END OF EXCERCIZE #1——————————————————
```

Exercise 2

(A) Interprets of parameters:
λ: The density of Poisson distribution during a time unit(per day);
σ_j: The standard deviation of step jump.

Here σ is the standard deviation of daily jump, while σ_j is the standard variance of jump of every step. Since every day is divided into n steps, we need to convert daily standard deviation to step standard deviation by divided $\sqrt{n}$.

(B) To construct compound Poisson process J_i:

(i) Produce a Poisson distribution with density λT to product the jump number N;

(ii) Produce jump time by producing N random number from Unif $\sim (0, T)$;

(iii) Produce jump size by producing N random number from $\mathcal{N}(0, 1)$;

Here is the plot of time series prices:

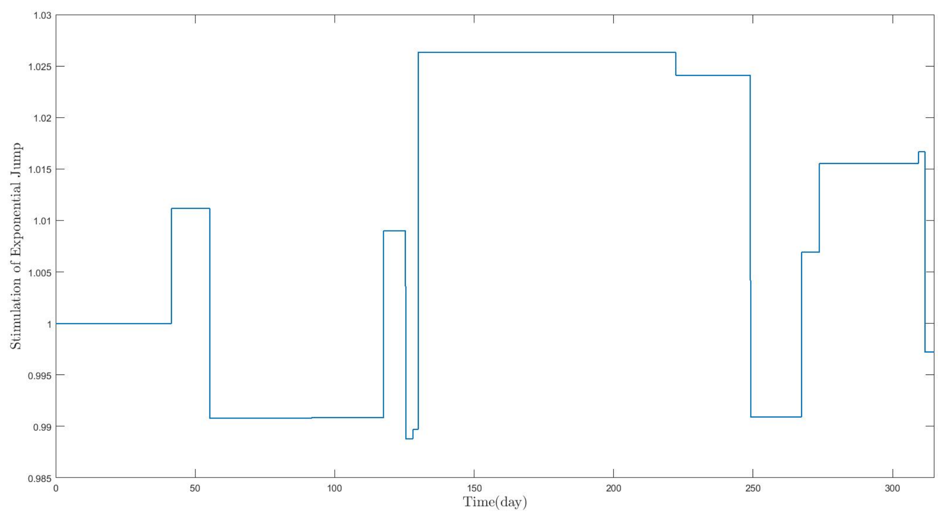

Figure 2: Time Series of Exponential Jump

From the figure, we can see there are total 12 times of jump during our observation (from time=0 to 315). The (exponential) jump size range from 0.85 to 1.03. Among these 12 jumps, 7 of them are upside jumps, while 5 of them are downside jumps.

The **MATLAB** code:

```
function [j] =jump(lamda,T,n,sigma)
% simulate a Compound Poisson Process
% T is total observe time
% j is the array of accumulated jump at a specifit time from 0 to T
% n is the number of observations in every unit of time
% sigma is the standard variance of jump
  j=[];
  j(1)=0; % the initial value of jump (at t=0)
  N=random('poisson',lamda*T); % the number of total jumps follow a Poisson
     distribution with density lamda*T
  t=random('uniform',0,T,N,1); % the time of jumps follow a uniform
     distribution from (0,T)
  y=random('normal',0,(18*sigma)/sqrt(n),N,1);% the size of jumps follow
     normal distribution with mean=0,sd=(18*sigma)/sqrt(n)
  for i=1:T*n
      k=0;
      for h=1:N % this loop aims at calculate the jumps which happens during
          time period ((i-1)/n,i/n)
         p=y(h)*indic((i-1)/n,i/n,t(h));
         k=k+p;
      end
      j(i+1)=j(i)+k;% accumulate jumps from time 0 to (i+1)/n
  end
end
```

```
% ----------------EXCERSIZE #2(COMPOUND POISSON PROCESS)------------------
% SET PARAMETERS
n=80;
T=1.25*252;
sigma=0.011;
lamda=15/252;

% PART B(STIMULATE MODEL)
J1=jump(lamda,T,n,sigma); % calculate value of Jt
e_J1=exp(J1);% make exponential for log-jump
t=0:1/n:T; % construct observation time series
f2=figure('name','Time Series of  Exponential Jump(COMPOUND POISSON PROCESS)'
    );
plot(t,e_J1);% plot of the simulated compound Poisson process
xlabel('Time(day)');
ylabel('Stimulation of Exponential Jump');
xlim([0,T]);
%------------------------END OF EXCERCIZE #2------------------------------
```

Exercise 3

(A) Both expression 1 and 4 are correct. Expression 1 is the log-price (with log-jump) of asset, while expression 4 is the price (with jump) of asset.

(B) Here use expression 4 to calculate the price (with jump) of asset.Following is the plot of time series prices:

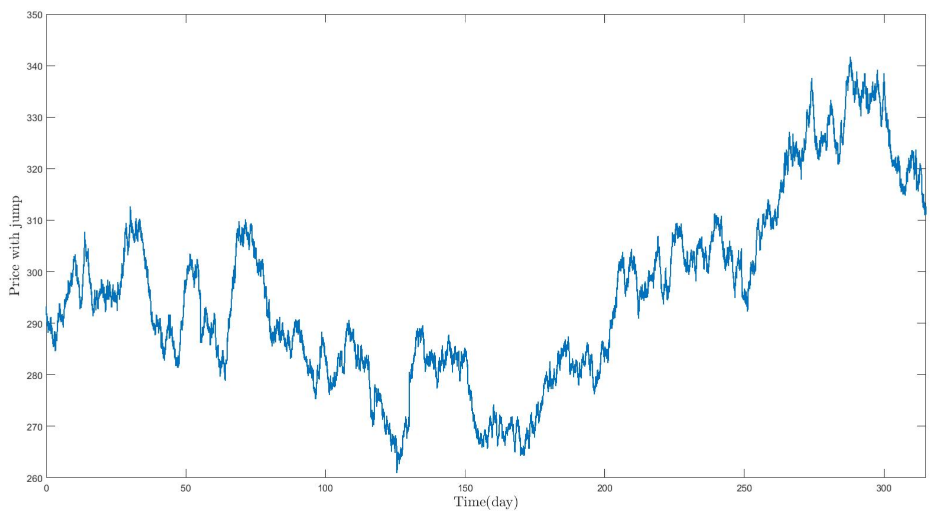

Figure 3: Time Series of Price with Jump

Compare to figure in Exercise 1, we can see the price range doesn't change a lot, but the fluctuation of price has increased.

The **MATLAB** code:

```
% ———EXCERCIZE #3(JUMP DIFFUSION MODEL WITH CONSTENT COEFFICIENTS)————
% SET PARAMETERS
n=80;
T=1.25*252;
u=0.03872/100;
c=0.011^2;
x0=log(292.58);
sigma=0.011;
lamda=15/252;

% PART B(STIMULATE MODEL)
X1=GD_c(T,n,c,u,x0); % calculate value of Xt
J1=jump(lamda,T,n,sigma); % calculate value of Jt
```

```

t=0:1/n:T; % construct observation time series
X2_with_jump=X1+J1; %calculate log-price with jump
P2_with_jump=exp(X2_with_jump);%convert log-price to price
f3=figure('name','Time Series of Price with Jump(JUMP DIFFUSION MODEL WITH
    CONSTENT COEFFICIENTS)');
plot(t,P2_with_jump);% plot of time series price
xlabel('Time(day)');
ylabel('Price with jump');
xlim([0,T]);
%------------------------END OF EXCERCIZE #3------------------------------
```

Exercise 4

(A) Interprets of parameters:
ρ: The rate of convergence;
μ_c: The mean of step price volatility.
σ_c: The volatility of step price volatility.

(B) To construct stochastic variance process c_j:

(i) Construct a series number from the standard normal distribution;

(ii) Compute each c_j by iteration;

Here is the plot of time series prices:

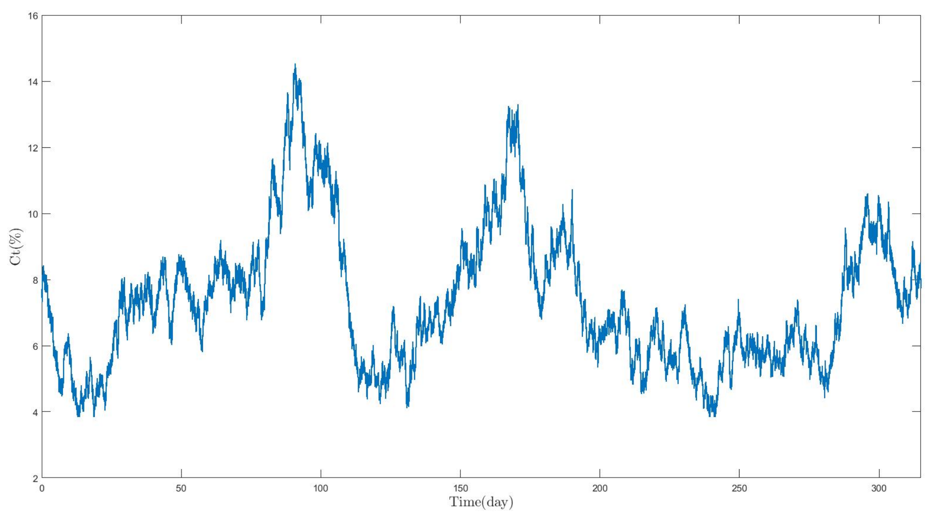

Figure 4: Stimulation of Annual Ct

From the figure, we can see at time around 40, 100, 180 and 300, there are large jump of Ct.

(C) To construct high frequency log-price X_j:

(i) Construct a series number from the standard normal distribution;

(ii) Use c_j to compute each X_j by iteration;

Here is the plot of time series prices:

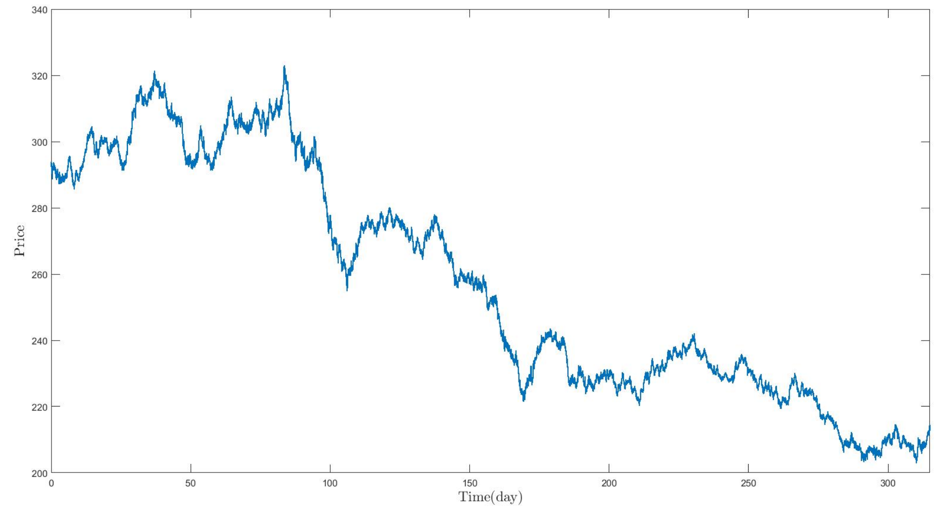

Figure 5: Stimulated of Time Series Price Xt (Use High Frequency Samples

From the figure, those we can see at time around 40, 100 and 180 and 300, Xt has large jump, too.

(D) Here is the plot of time series log-return:

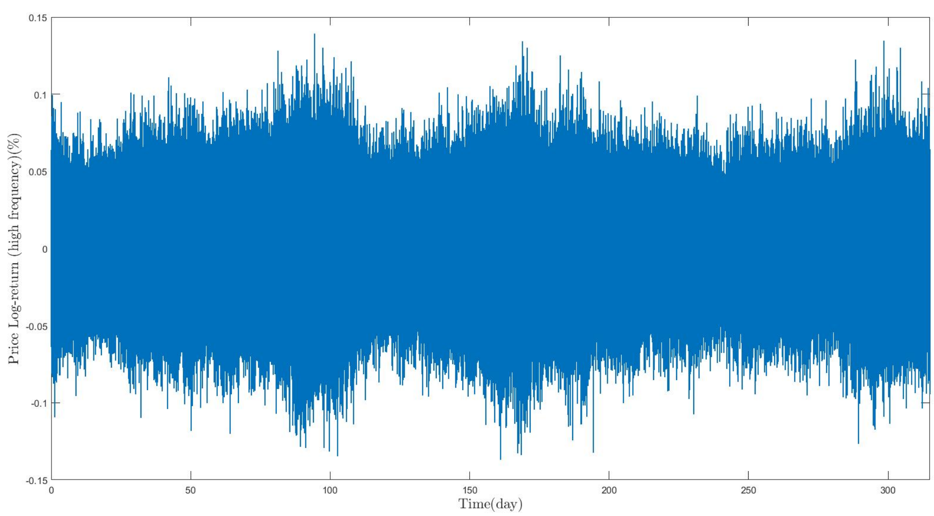

Figure 6: Time Series Log-return (Use High Frequency Samples)

From the figure, those we can see at time around 40, 100 and 180 and 300, where Xt has large jump, the price log-return fluctuates a lot , so well as the log-return in their neighbor time interval.We can find the pattern of volatility clustering.

(E) Here is the plot of time series log-return with coarser frequency sample:

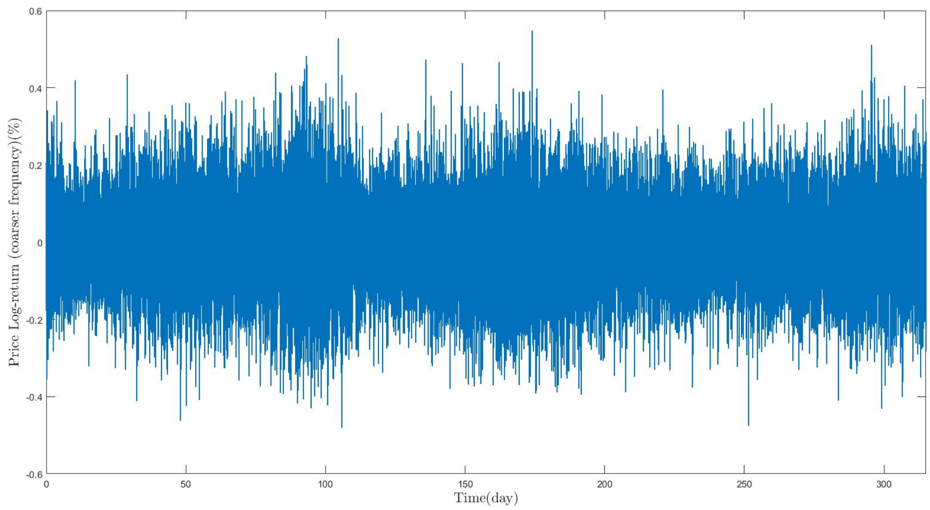

Figure 7: Time Series Log-return (Use Coarser Frequency Samples)

From the figure we can find the pattern of volatility clustering is more obvious than the case using high frequency sample.

(F) Here are the plots of stochastic volatility with different rate of convergence ρ:

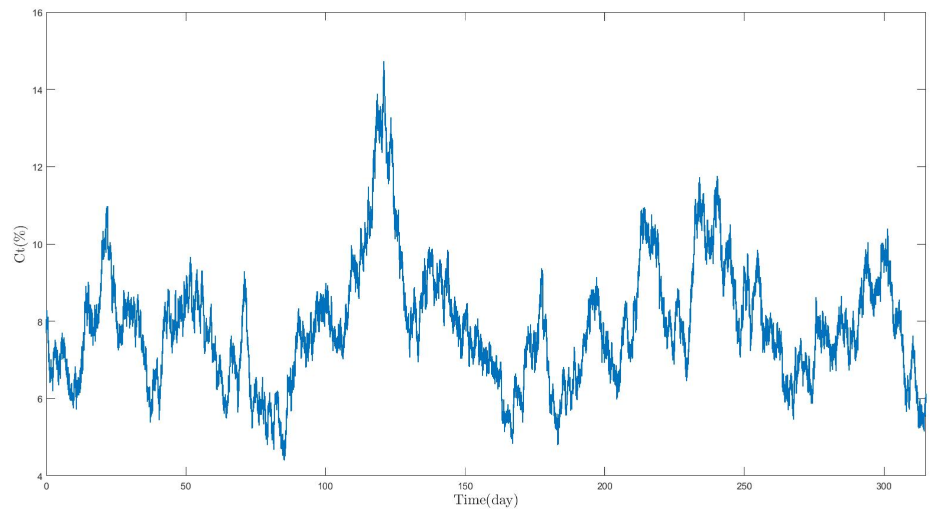

Figure 8: Stochastic Volatility Ct with $\rho = 0.03$

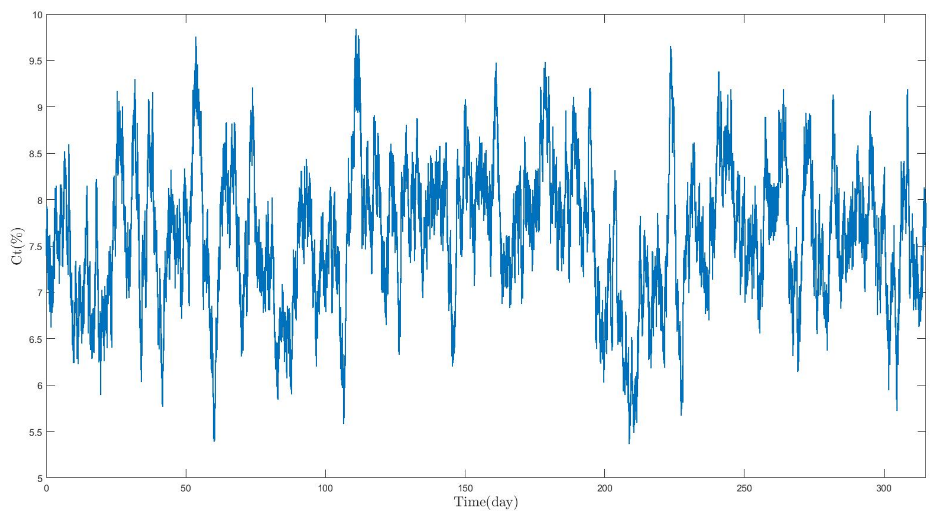

Figure 9: Stochastic Volatility Ct with $\rho = 0.1$

From the figure, we can find as the rate of convergence ρ increases, the pattern that price stochastic volatility Ct fluctuates around its mean becomes more obvious, that is, the price stochastic volatility Ct converges at a faster speed.

(G) Here are the plots of stochastic volatility with different volatility σ_c:

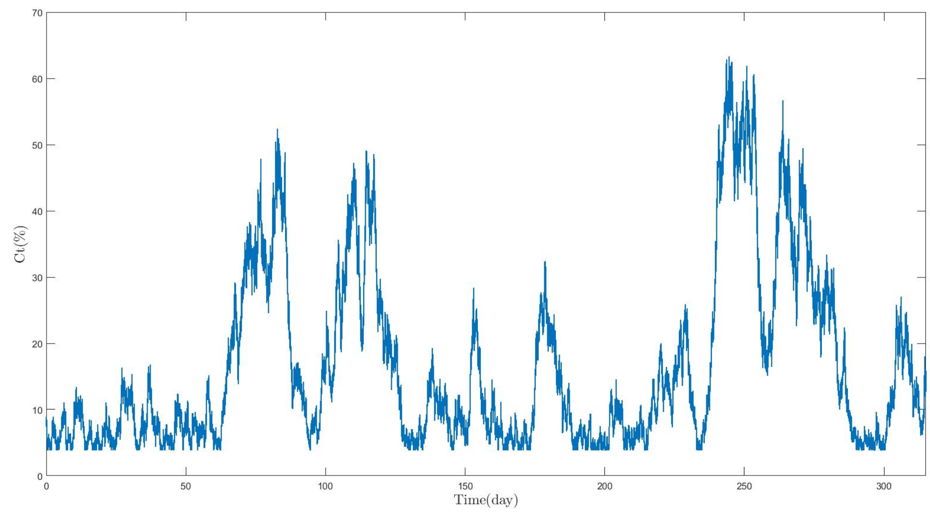

Figure 10: Stochastic Volatility Ct with $\sigma_c = 0.005$

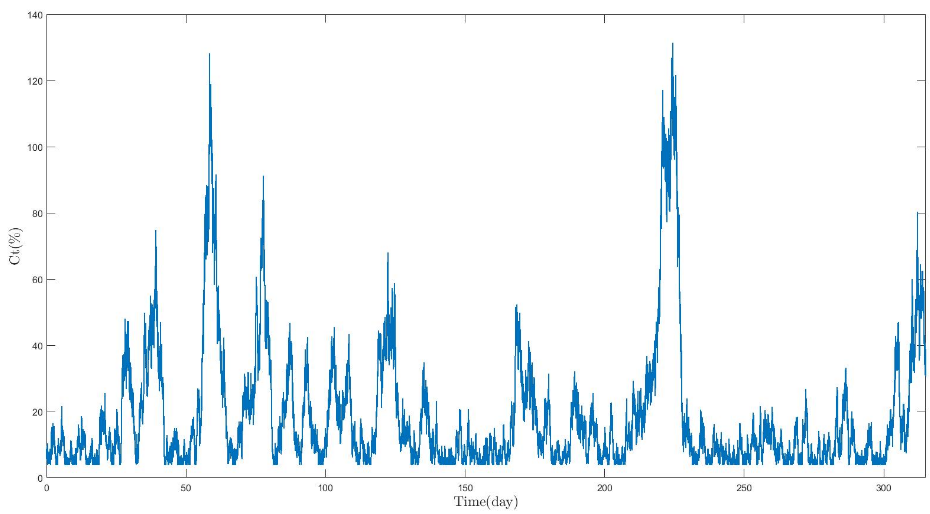

Figure 11: Stochastic Volatility with $\sigma_c = 0.01$

From the figure, we can find as the stochastic volatility with different volatility σ increases from 0.005 to 0.01, the range of stochastic volatility Ct has doubled from $(\frac{\mu_c}{2}, 70)$ to $(\frac{\mu_c}{2}, 140)$, the volatility of Ct increases.

The **MATLAB** code:

```
function [c] = JD_sv_ct(ne,T,l,u_c,sigma_c,c0)
% simulate stochastic variance process {ct} in a Jump—diffusion model
```

```
% c is the array of ct between t vary from 0~T
% T is total observe time
% l is the correlation parameter
% ne is the number of observations in every unit of time
% sigma is the standard variance of {ct}
% u is the mean of {ct}
c=[];
c(1)=c0; % initial value of ct(at t=0)
delta_e=1/ne; % delta_e is the time gap between every observation
 for j=1:ne*T %observation number from 1 to ne*T
     r=normrnd(0,1); %construct random variable from standard normal
         distribution
     c(j+1)=max(c(j)+l*(u_c-c(j))*delta_e+sigma_c*sqrt(c(j)*delta_e)*r,u_c/2)
         ;  % calculate c(i+1) by iteration
 end
end
```

```
function [x] =JD_sv_xt(ne,T,C,x0)
% simulate high-frequence log-price {xt} in a Jump-diffusion model with
    stochastic volatility
% x is the array of xt between t vary from 0~T
% x0 is the initial value of xt(at t=0)
% c is the array of ct between t vary from 0~T
% T is total observe time
% l is the correlation parameter
% ne is the number of observations in every unit of time
% sigma is the standard variance of {ct}
% u is the mean of {ct}
delta_e=1/ne;  % delta_e is the time gap between every observation
x=[];
x(1)=x0; % initial value of xt(at t=0)
 for i=1:ne*T %observation number from 1 to ne*T
    r=normrnd(0,1);%construct random variable from standard normal
        distribution
    x(i+1)=x(i)+sqrt(C(i)*delta_e)*r;  % calculate c(i+1) by iteration
 end
end
```

```
% ------EXCERSIZE #4(JUMP DIFFUSION MODEL WITH STOCHASTIC VOLATILITY)------
% SET PARAMETERS
n=80;
T=1.25*252;
ne=n*20;
l=0.03;
u_c=0.011^2;
```

```
sigma_c=0.001;
c0=u_c;
x0=log(292.58);

% PART B(STIMULATE Ct MODEL)
Ct=JD_sv_ct(ne,T,l,u_c,sigma_c,c0);%calculate Ct
te=0:1/ne:T; % construct observation time series
f4_b=figure('name','Stimulation of Annual Ct');
plot(te,Ct*sqrt(252*ne)*100);% plot of time series Ct(annual percentage)
xlabel('Time(day)');
ylabel('Ct(\%)');
xlim([0,T]);

% PART C(STIMULATE Xt MODEL)
X3=JD_sv_xt(ne,T,Ct,x0);%calculate Xt
P3=exp(X3); %convert log-price to price
f4_c=figure('name','Stimulation of Price Xt');
plot(te,P3);% plot of time series price
xlabel('Time(day)');
ylabel('Price');
xlim([0,T]);

% PART D(STIMULATE LOG-RETURN)
R1=diff(X3);%calculate log_return
D1=[0,R1];
f4_d=figure('name','Time Series Log_return With High Frequence Sample');
plot(te,D1*100);% plot of time series percentage log_return
xlabel('Time(day)');
ylabel('Price Log-return (high frequency)(\%)');
xlim([0,T]);

% PART E(STIMULATE Xt AT A COARSER FREQUENCY)
t=0:1/n:T; % construct coarser observation time series
X4=X3(1:20:end);% choose coarser frequency observation
R2=diff(X4);%calculate log_return
D2=[0,R2];
f4_e=figure('name','Time Series Log_return  of Coarser Frequency Sample');
plot(t,D2*100);% plot of time series annual log_return of coarser frequency
    observations
xlabel('Time(day)');
ylabel('Price Log-return (coarser frequency)(\%)');
xlim([0,T]);

% PART F(INCREASE THE RATE OF CONVERGENCE)
```

```
% increase the rate convergence from 0.03 to 0.1
l2=0.1;
Ct_2=JD_sv_ct(ne,T,l2,u_c,sigma_c,c0);%calculate Ct
te=0:1/ne:T; % construct observation time series
figure('name','Stimulation of Annual Ct(with rate of convergence = 0.1');
plot(te,Ct_2*sqrt(252*ne)*100);% plot of time series Ct(annual percentage)
xlabel('Time(day)');
ylabel('Ct(\%)');
xlim([0,T]);
% increase the rate convergence from 0.03 to 0.5
l3=0.5;
Ct_3=JD_sv_ct(ne,T,l3,u_c,sigma_c,c0);%calculate Ct
te=0:1/ne:T; % construct observation time series
figure('name','Stimulation of Annual Ct(with rate of convergence = 0.5');
plot(te,Ct_3*sqrt(252*ne)*100);% plot of time series Ct(annual percentage)
xlabel('Time(day)');
ylabel('Ct(\%)');
xlim([0,T]);


% PART G(INCREASE THE RATE OF CONVERGENCE)
% increase the volatility of stochastic volatility from 0.001 to 0.005
sigma_c1=0.005;
Ct_4=JD_sv_ct(ne,T,l,u_c,sigma_c1,c0);%calculate Ct
te=0:1/ne:T; % construct observation time series
figure('name','Stimulation of Annual Ct(with volatility = 0.005');
plot(te,Ct_4*sqrt(252*ne)*100);% plot of time series Ct(annual percentage)
xlabel('Time(day)');
ylabel('Ct(\%)');
xlim([0,T]);
% increase the rate convergence from 0.03 to 0.5
sigma_c2=0.01;
Ct_5=JD_sv_ct(ne,T,l,u_c,sigma_c2,c0);%calculate Ct
te=0:1/ne:T; % construct observation time series
figure('name','Stimulation of Annual Ct(with rate of convergence = 0.01');
plot(te,Ct_5*sqrt(252*ne)*100);% plot of time series Ct(annual percentage)
xlabel('Time(day)');
ylabel('Ct(\%)');
xlim([0,T]);
```