IAM 592 −Project

May 25, 2017

Question-1

Part-a

In this part, I generate 100 random numbers from exponential distribution with λ=0.001429. Quantile at 0.05 gives very smalle number 6.072019981934850e-05 and to address the question of whether distribution quantile (VaR) gives reasonable results by estimating the difference between it and mean, I subtract the mean of exponential distribution I obtained from 0.95 quantile. I get 0.0023. So, estimating the difference is reasonable but might not be if the difference would be 0.

Part-b

I follow the same procedure as in exponential distribution. Mean of the simulated data is 88.4334 and quantile at 0.05 is 48.3204. At 95 percentile, the difference between of the simulated data and the distribution is 129.1691. So, it is reasonable to estimate. In fact it is more reasonable than the exponential distribution.

Part-c and **Part-d**

In these parts, random numbers are generated from log normal and gamma distribution. In both cases, since difference between mean and the corresponding quantile is not zero, it can be concluded that it is reasonable to estimate the difference.

Matlab codes of all parts in the question-1 can be found below:

```
%% Question-1
```

```
%Exponential Distribution
rand_exp=exprnd(0.001429,100,1);
quan_exp=quantile(rand_exp, 0.05);
difference_exp=mean(rand_exp)-quantile(rand_exp,0.95);
%Normal Distribution
rand_norm=normrnd(90,25,[100,1]);
quan_norm=quantile(rand_norm,0.05);
difference_exp=mean(rand_norm)-quantile(rand_norm,0.95);
%Lognormal Distribution
rand_log=lognrnd(8,2,[100,1]);
quan_log=quantile(rand_log,0.05);
difference_log=mean(rand_log)-quantile(rand_log,0.95);
%Gamma Distribution
rand_gamma=gamrnd(25,5,[100,1]);
quan_gamma=quantile(rand_gamma,0.05);
difference_gamma=mean(rand_gamma)-quantile(rand_gamma,0.95);
```

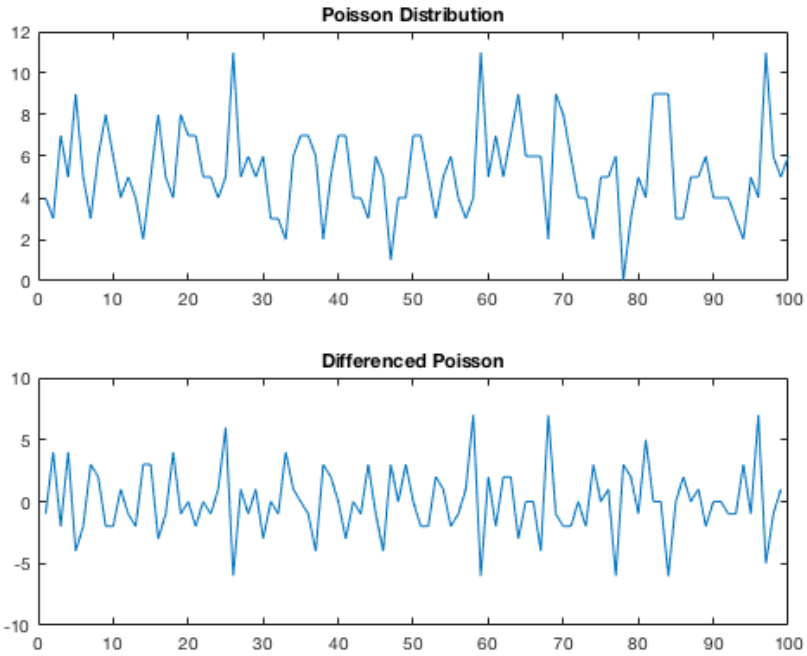
Question-2

Part-a Again using matlab, I simulate Poisson distribution with λ=5 and the matlab code is provided below. Plot of Poisson distribution shows that the data oscillate around 0 and 10 depending on our setting. As can be clearly seen in the differenced plot, it is stationary after taking the first difference.

```
function x = cspoirnd(lam,n)%lam =5;n = 100;
x = zeros(1,n);
j = 1;
while j <= n
flag = 1;
% initialize quantities
u = rand(1);
i = 0;
```

```
p = exp(-lam);
F = p;
while flag % generate the variate needed
if u <= F % then accept
x(j) = i;
flag = 0;
j = j+1;
else % move to next probability
p = lam*p/(i+1);
i = i+1;
F = F + p;
end
end

end
end
%% Plot
%Histogram
edges = 0:max(x);
f = histc(x,edges);
bar(edges,f/N,1,'w')
%lines
subplot(2,1,1)
plot(x)
title('Poisson Distribution')
diff_x=diff(x);
subplot(2,1,2)
plot(diff_x);% differenced plot
title('Differenced Poisson')
```



Part-b

In part-b of the question, I simulate a random walk process of the form. I set the initial value as 2 and get 100 variates. In the first graph. it can be observed that random walk fluctuates wildly. However, after taking the first differenced, it goes back and forth around 0 as in Poisson case.

$$y_t = y_{t-1} + \epsilon_t \tag{1}$$

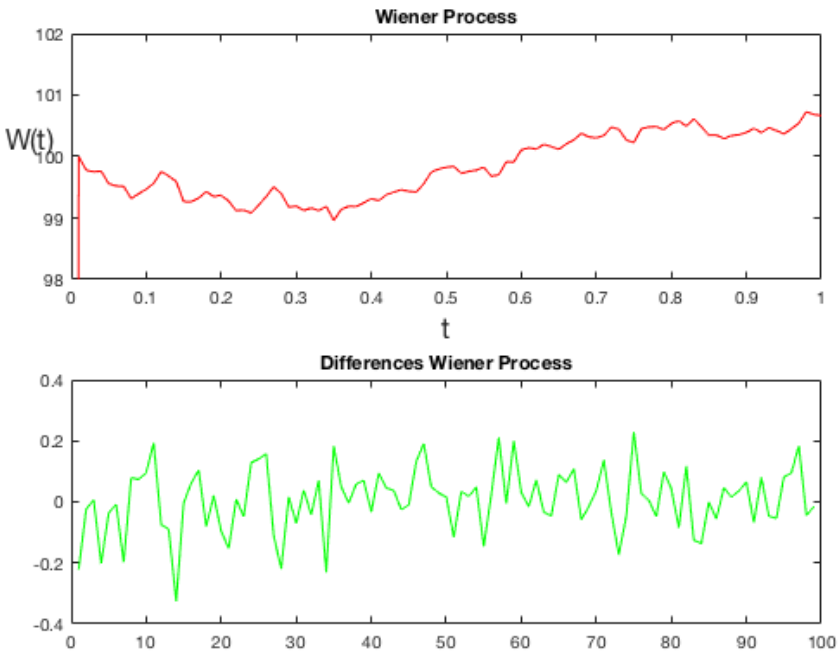
```
% Random Walk
e=randn(1,100);
i=1;
y=0*e;
size(y)
y(1)=2;
while(i<max(size(y)))
y(i+1)=y(i)+e(i+1);
i=i+1;
end
```

```
%% Plots
subplot(2,1,1)
t=[1:1:100];
plot(t,y);
title('Random Walk')
subplot(2,1,2)
diff_y=diff(y);
plot(diff_y);%Differenced plot
title('Differenced Random Walk')
```

Part-c

```
randn('state',100)          % set the state of randn
T = 1; N = 100; dt = T/N;
dW = zeros(1,N);            % preallocate arrays ...
W = zeros(1,N);             % for efficiency
```

```
dW(1) = sqrt(dt)*randn;        % first approximation outside the loop ...
W(1) = 100;                    % since W(0) = 0 is not allowed
for j = 2:N
dW(j) = sqrt(dt)*randn;    % general increment
W(j) = W(j-1) + dW(j);
end
%% Plots
subplot(2,1,1)
plot([0:dt:T],[0,W],'r-')% plot W against t
ylim([98 102])
xlabel('t','FontSize',16)
ylabel('W(t)','FontSize',16,'Rotation',0)
title('Wiener Process')
diff_W=diff(W);
subplot(2,1,2)
plot(diff_W,'g-')%Differenced plot
title('Differences Wiener Process')
```

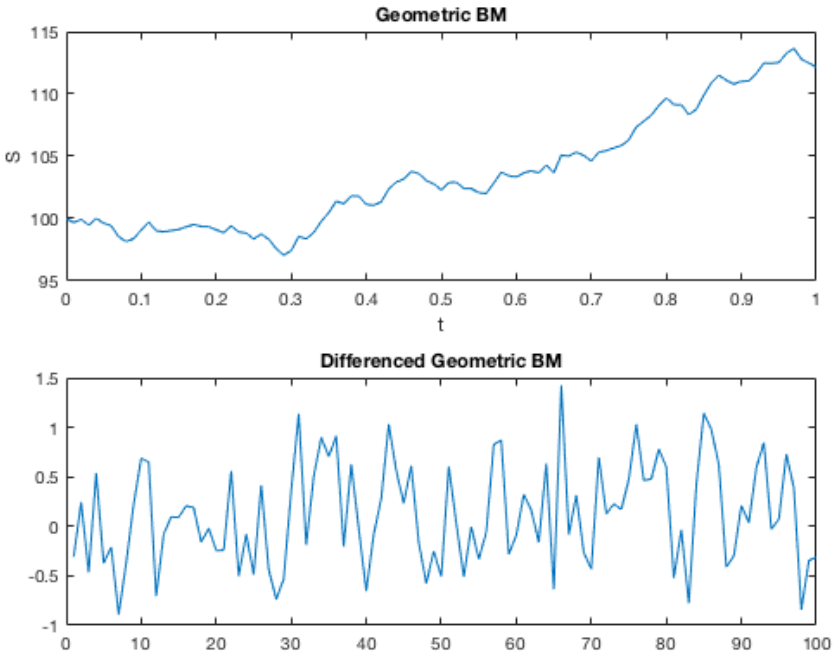


Part-d

This part belongs to the Geometric Brownian Motion. In order to simulate, I take initial value as 100, drift term as 0.1, volatility as 0.05. Again, I obtain 100 simulated data and since initial value is 100, all the simulated data is around 97 and 114. After taking the first difference, due partly to the small volatility, I take the stationary series.

```
%%Geometric BM
%T=1;M=100;dt=T/M;mu=0.1;sigma=0.05;S0=100;N=100;
S=zeros(1,M+1); %S(:,1)=S0*ones(1,N);
for j=1:M
dW=sqrt(dt)*randn;
S(1)=100;
a=mu*S(1,j); b=sigma*S(1,j);
S(1,j+1)=S(1,j)+a*dt+b*dW;
end

%% Plots
subplot(2,1,1)
plot([0:dt:T], S),
xlabel('t'), ylabel('S')
title('Geometric BM')
%Differenced
diff_S=diff(S);
subplot(2,1,2)
plot(diff_S),
title('Differenced Geometric BM')
```



European Call/Put Prices

Below can be found European Call and Put prices simulated by Geometric Brownian Motion and Black Scholes Equation.

```
function [C,P]=Call_Put(S,K,mu,sigma,tau,div) %;K=100;mu=0.1;sigma=0.05;tau=5,div=0
if nargin<6
div=0.0;
end
if tau>0
d1=(log(S/K)+(mu+0.5*sigma^2)*tau*ones(size(S)))/(sigma*sqrt(tau));
d2=d1-sigma*sqrt(tau);
N1=1/2*(1+erf(d1/sqrt(2)));
N2=1/2*(1+erf(d2/sqrt(2)));
C=exp(-div*tau)*S.*N1-K*exp(-mu*tau)*N2;
P=C+K*exp(-mu*tau)-exp(-div*tau)*S;
else
C=max(S-K,0);
P=max(K-S,0);
```

```
end
subplot(2,1,1)
plot(S,C),
xlabel('S'), ylabel('V');
subplot(2,1,2)
plot(S,P, 'r--')
xlabel('S'), ylabel('V');
```