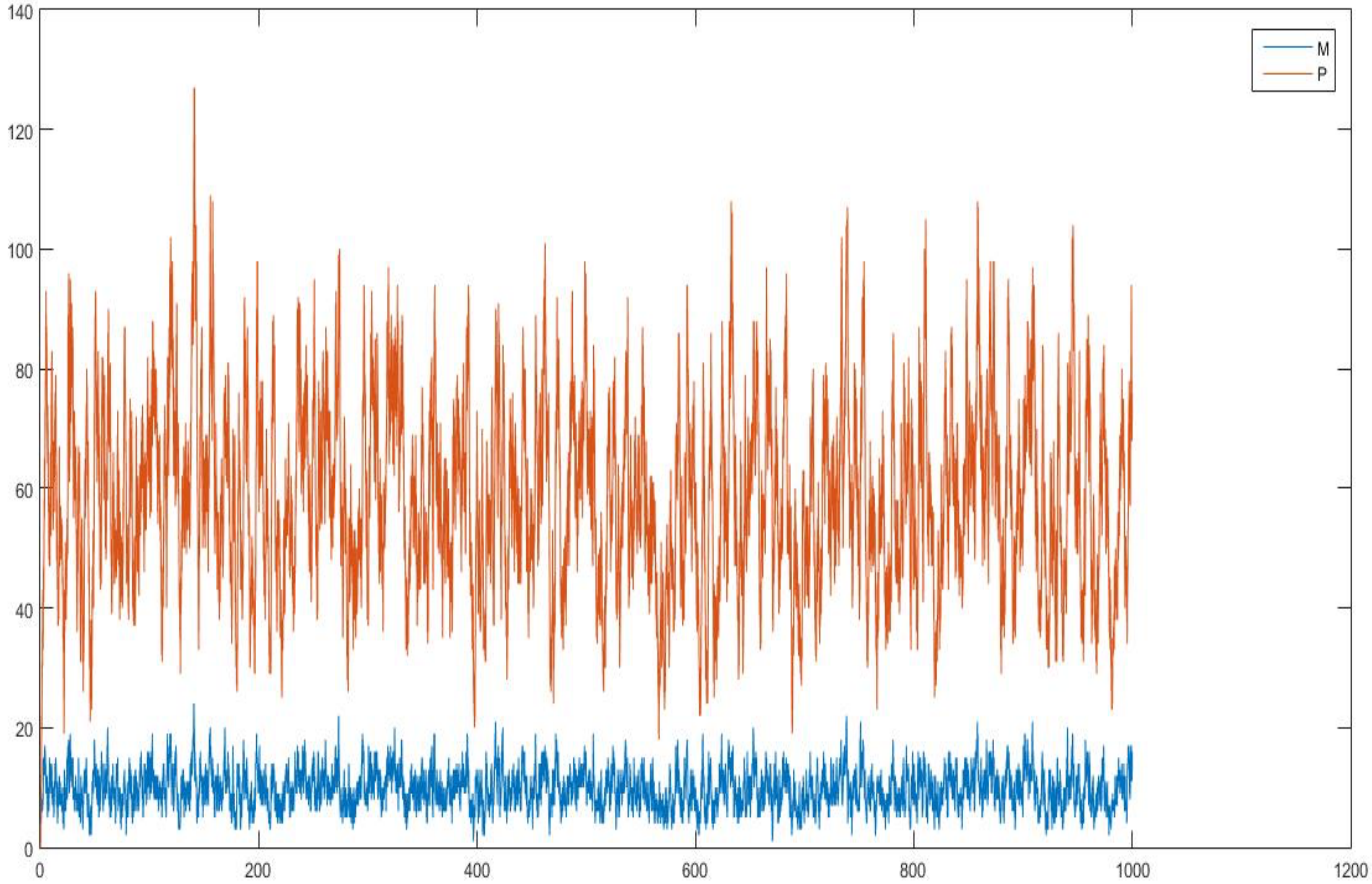
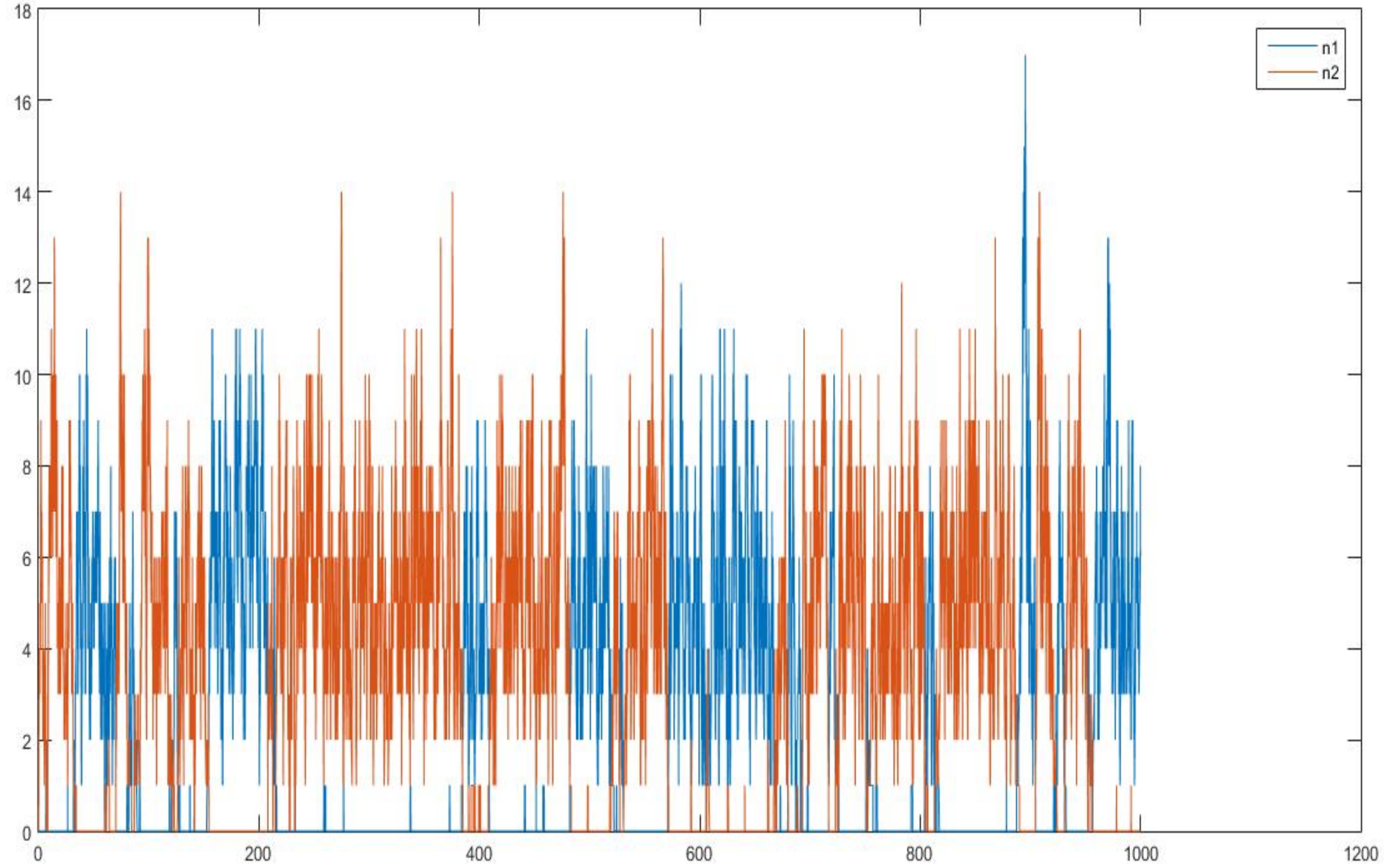
Q1. Statistics of an Ensemble of Sample Paths. (Mathematical Modeling in Systems Biology: An Introduction Brian Ingalls: 7.8.24* Exercise)

```
function gillespie
% R1: null -> M    ; propensity = km
% R2: null -> P    ; propensity = kp * M
% R3: M    -> null ; propensity = dm * M
% R4: P    -> null ; propensity = dp * P
t=0;
t_end=1000;
km=10;
kp=6;
dm=1;
dp=1;
% starting condition
M=0;
P=0;
j=1;
t_array(j)=t;
m_array(j)=M;
p_array(j)=P;
while t<t_end
    propensity=[km kp*M dm*M dp*P];
    totalPropensity=sum(propensity);
    time_wait=-(1/totalPropensity)*log(rand);
    t=t+time_wait;
    cumulativePropensity=cumsum(propensity)/totalPropensity;
    k=rand;
    if k>=0 && k<=cumulativePropensity(1)
        M=M+1;
    elseif k>cumulativePropensity(1) &&
k<=cumulativePropensity(2)
        P=P+1;
    elseif k>cumulativePropensity(2) &&
k<=cumulativePropensity(3)
        M=M-1;
    else
        P=P-1;
    end
    j=j+1;
    t_array(j)=t;
    m_array(j)=M;
    p_array(j)=P;
end
plot(t_array,m_array),hold all;
plot(t_array,p_array),legend('M','P');
```


140
120
100
80
60
40
20
0
0
200
400
600
800
1000
1200
M
P

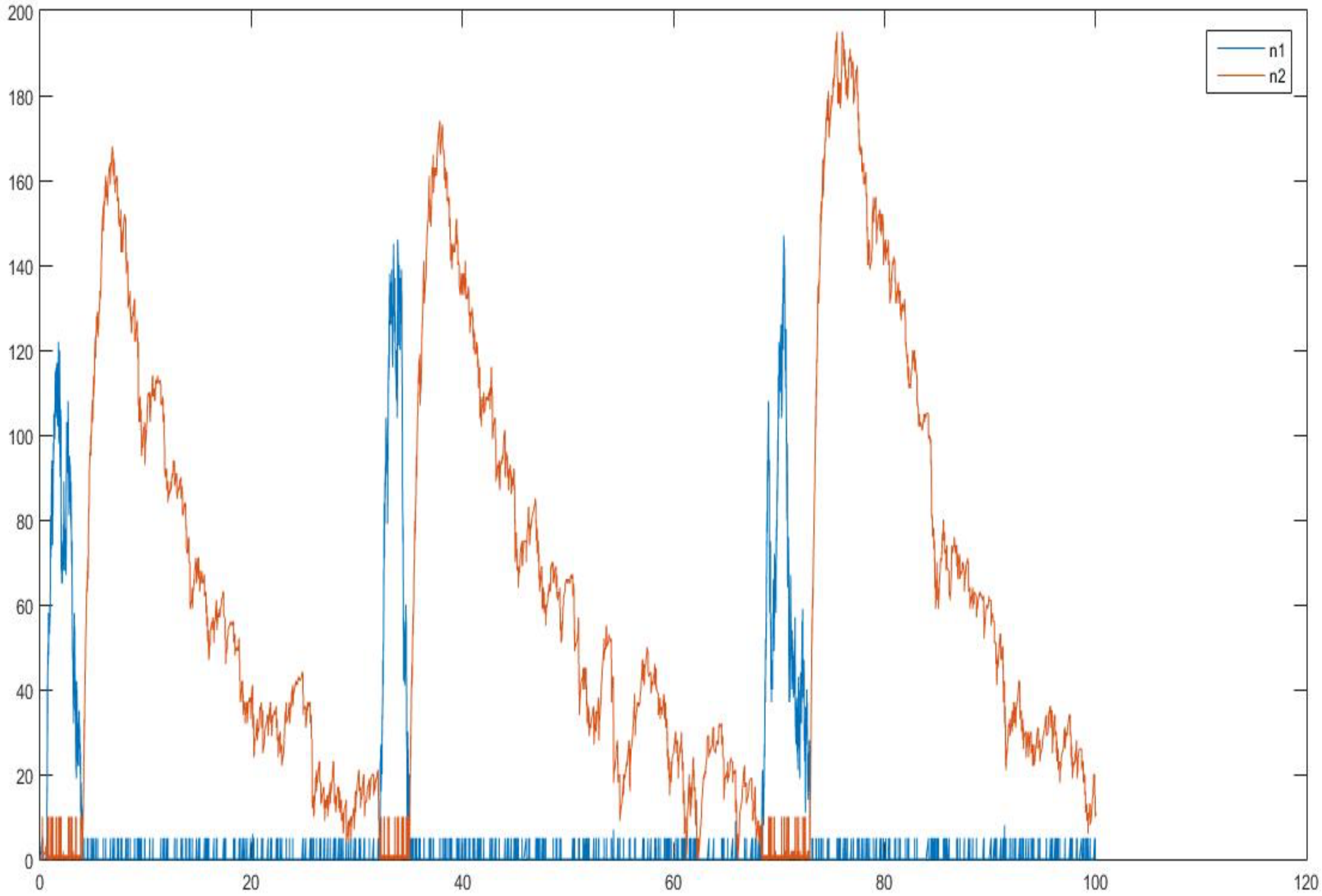
Q2. 7.8.26* Noisy Toggle Switch

```
function Noisy_Toggle
% R1(synthesis): null -> p1 ; propensity = alpha/(1+n2^beta)
% R2(synthesis): null -> P2 ; propensity = alpha/(1+n1^beta)
% R3(decay): p1 -> null ; propensity = delta*n1
% R4(decay): p2 -> null ; propensity = delta * n2
t=0;
t_end=1000;
alpha = 5; %Run simulations of the stochastic system for α = 5,
50, 500, and 5000
delta = 1;
beta = 4;
% starting condition
n1=0;
n2=0;
j=1;
t_array(j)=t;
n1_array(j)=n1;
n2_array(j)=n2;
while t<t_end
propensitiesVec = [alpha/(1+n2^beta) alpha/(1+n1^beta) delta*n1
delta*n2];
totalPropensity = sum(propensitiesVec);
waitTime = - log( rand(1,1) ) / totalPropensity;
t=t+waitTime;
   cumulativePropensity =
cumsum(propensitiesVec)/totalPropensity;
   k=rand;
    if k>=0 && k<=cumulativePropensity(1)
        n1=n1+1;
    elseif k>cumulativePropensity(1) &&
k<=cumulativePropensity(2)
        n2=n2+1;
    elseif k>cumulativePropensity(2) &&
k<=cumulativePropensity(3)
        n1=n1-1;
    else
        n2=n2-1;
    end
    j=j+1;
    t_array(j)=t;
    n1_array(j)=n1;
    n2_array(j)=n2;
end
plot(t_array,n1_array),hold all;
plot(t_array,n2_array),legend('n1','n2');
```


18
16
14
12
10
8
6
4
2
0
0
200
400
600
800
1000
1200
n1
n2

Q3. 7.8.27* Noise-Induced Oscillations

```
function Noise_Induced_Oscillations
% R1(activator synthesis): null -> bA*A ; propensity =  ((gammaA/betaA)*((alpha0+(nA/kA))/(1+(nA/kA))))
% R2(repressor synthesis): null -> bR*R ; propensity = (gammaR/betaR)*((nA/kR)/(1+(nA/kR)))
% R3(activator decay): A -> null ; propensity = deltaA*nA
% R4(repressor decay): R -> null ; propensity = detaR*nR
% R5(association):A+R -> C ; propensity=kC*nA*nR
% R6(dissociation and decay) C -> R; propensity=deltaA*nC
%nA=activator;nR=repressor;nC=activator_repressor complex
t=0;
t_end=100;
gammaA = 250;
betaA = 5;
kA = 0.5;
alpha0 = 0.1;
deltaA = 1;
gammaR = 50;
betaR =10;
kR = 1;
kC = 200;
detaR = 0.1;
nA=0;
nR=0;
nC=0;
j=1;
t_array(j)=t;
nA_array(j)=nA;
nR_array(j)=nR;
nC_array(j)=nC;
while t<t_end
p1=((gammaA/betaA)*((alpha0+(nA/kA))/(1+(nA/kA))));
p2=(gammaR/betaR)*((nA/kR)/(1+(nA/kR)));
p3=deltaA*nA;
p4=detaR*nR;
p5=kC*nA*nR;
p6=deltaA*nC;
propensitiesVec = [p1 p2 p3 p4 p5 p6];
totalPropensity = sum(propensitiesVec);
waitTime = - log( rand(1,1) ) / totalPropensity;
t=t+waitTime;
cumulativePropensity = cumsum(propensitiesVec)/totalPropensity;
   k=rand;
    if k>=0 && k<=cumulativePropensity(1)
        nA=nA+betaA;
    elseif k>cumulativePropensity(1) && k<=cumulativePropensity(2)
        nR=nR+betaR;
    elseif k>cumulativePropensity(2) && k<=cumulativePropensity(3)
        nA=nA-1;
    elseif k>cumulativePropensity(3) && k<=cumulativePropensity(4)
        nR=nR-1;
    elseif k>cumulativePropensity(4) && k<=cumulativePropensity(5)
        nA=nA-1;
        nR=nR-1;
        nC=nC+1;
    else
        nC=nC-1;
        nR=nR+1;
    end
    j=j+1;
    t_array(j)=t;
    nA_array(j)=nA;
    nR_array(j)=nR;
    nC_array(j)=nC;
end
plot(t_array,nA_array),hold all;
plot(t_array,nR_array),legend('n1','n2');
```


200
180
160
140
120
100
80
60
40
20
0
0
20
40
60
80
100
120
n1
n2