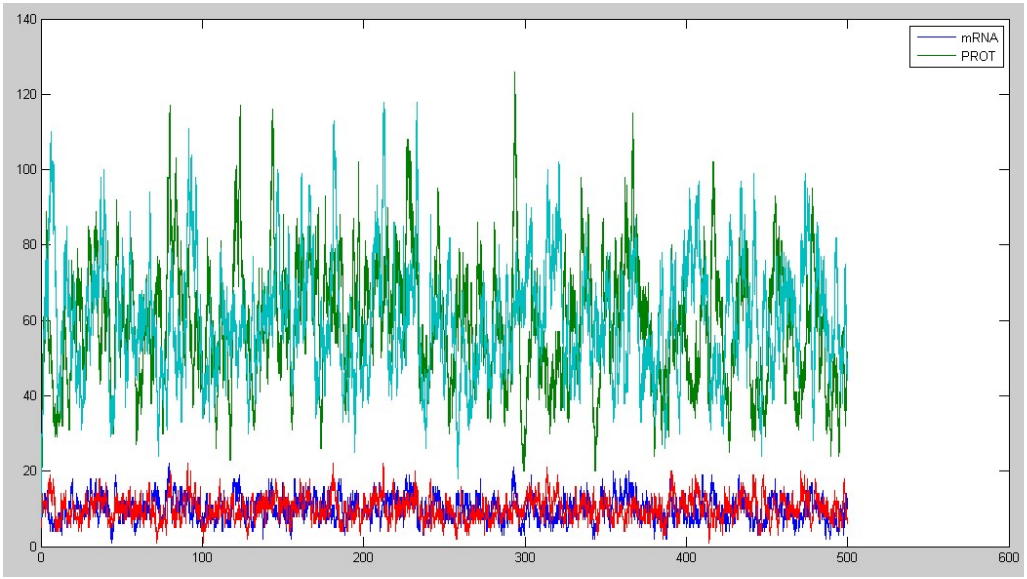
GILLESPI'S ALGORITHM:

```
function gillespi t=0; t_end=500;
km=10; kp=6; dm=1; dp=1; M=5; P=10;
z=1; t_array(z)=t; m_array(z)=M;
p_array(z)=P; while t<t_end
propensity=[km kp*M dm*M dp*P];
propsum=sum(propensity);
time_wait=-(1/propsum)*log(rand);
t=t+time_wait;
cuprob=cumsum(propensity)/propsum;
x=rand; if x>=0 && x<=cuprob(1)
M=M+1;
    elseif x>cuprob(1) && x<=cuprob(2)
P=P+1;
    elseif x>cuprob(2) && x<=cuprob(3)
        M=M-1;
else
P=P-1; end
z=z+1;
t_array(z)=t;
m_array(z)=M;
p_array(z)=P;
end
plot(t_array,m_array),hold all;
plot(t_array,p_array),legend('mRNA','PROT');
```

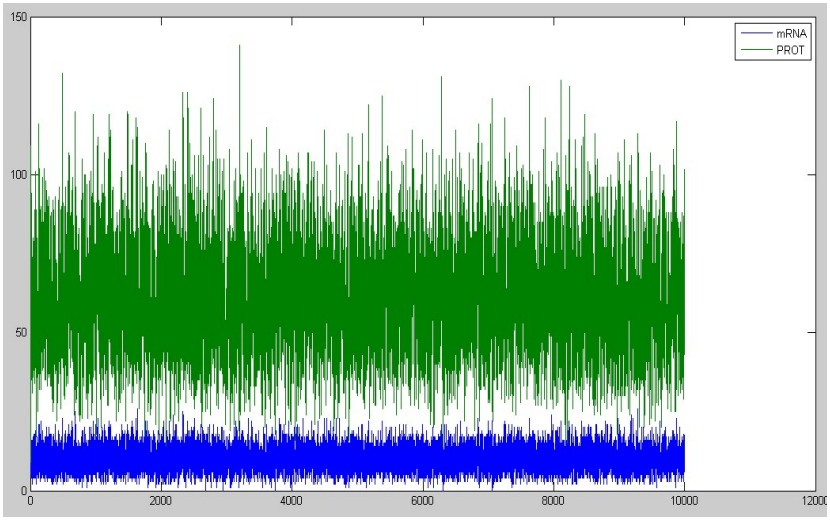
Plot:



Problem 7.8.24- Statistics of an Ensemble of Sample Paths

```
function ensemble
t=0;
t_end=10000;
km=10; kp=6;
dm=1; dp=1;
M=5; P=10;
z=1;
t_array(z)=t;
m_array(z)=M;
p_array(z)=P;
while t<t_end
propensity=[km kp*M dm*M dp*P];
propsum=sum(propensity); time_wait=-
(1/propsum)*log(rand); t=t+time_wait;
cuprob=cumsum(propensity)/propsum; x=rand;
if x>=0 && x<=cuprob(1) M=M+1;
elseif x>cuprob(1) && x<=cuprob(2)
P=P+1; elseif x>cuprob(2) &&
x<=cuprob(3)
M=M-1; else
P=P-1; end
z=z+1;
t_array(z)=t;
m_array(z)=M;
p_array(z)=P;
end
plot(t_array,m_array),hold all;
plot(t_array,p_array),legend('mRNA','PROT');
theoritical_m=std(m_array)/mean(m_array);
theoritical_p=std(p_array)/mean(p_array);
calculated_m=(dm/km)^(1/2);
calculated_p=(((km*dp)/(km/kp))^(1/2))*((1+(kp/(dp+dm))^(1/2)));
if theoritical_p == calculated_p & theoritical_m == calculated_m,
disp('In steady state') else
disp('Not in steady state')
end
```

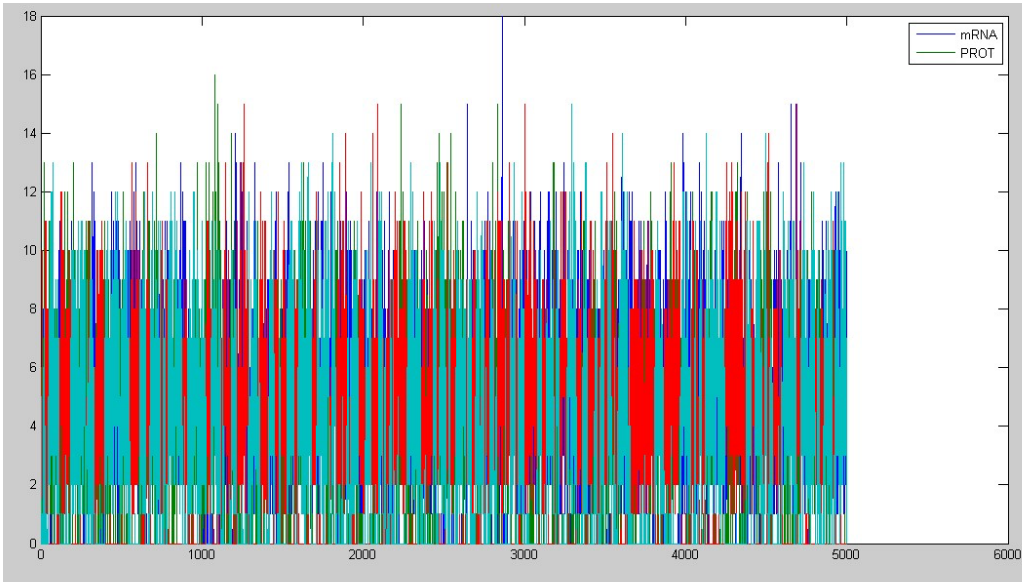
plot:



Problem 7.8.26-Noisy Toggle Switch

```
function noisyswitch t=0; t_end=5000; alpha = 5; delta = 1; beta = 4;
n1=0; n2=0; z=1; t_array(z)=t; n1_array(z)=n1; n2_array(z)=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;
cuprob = cumsum(propensitiesVec)/totalPropensity;
x=rand;
if x>=0 && x<=cuprob(1) n1=n1+1;
elseif x>cuprob(1) && x<=cuprob(2) n2=n2+1;
elseif x>cuprob(2) && x<=cuprob(3)
n1=n1-1; else n2=n2-1; end z=z+1;
t_array(z)=t; n1_array(z)=n1;
n2_array(z)=n2; end
plot(t_array,n1_array),hold all;
plot(t_array,n2_array),legend('mRNA','PROT');
```

Plot:



Problem 7.8.27-Statistics of an Ensemble of Sample Paths:

```
function occilation
t=0; t_end=500;
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; z=1;
t_array(z)=t;
nA_array(z)=nA;
nR_array(z)=nR;
nC_array(z)=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;
cuprob = cumsum(propensitiesVec)/totalPropensity;
x=rand;     if x>=0 && x<=cuprob(1)
nA=nA+betaA;     elseif x>cuprob(1) &&
x<=cuprob(2)          nR=nR+betaR;     elseif
x>cuprob(2) && x<=cuprob(3)          nA=nA-1;
elseif x>cuprob(3) && x<=cuprob(4)
nR=nR-1;     elseif x>cuprob(4) && x<=cuprob(5)
nA=nA-1;          nR=nR-1;          nC=nC+1;
else          nC=nC-1;          nR=nR+1;     end
z=z+1;     t_array(z)=t;     nA_array(z)=nA;
nR_array(z)=nR;     nC_array(z)=nC; end
plot(t_array,nA_array),hold all;
plot(t_array,nR_array),legend('NA','NC');
```

plot: