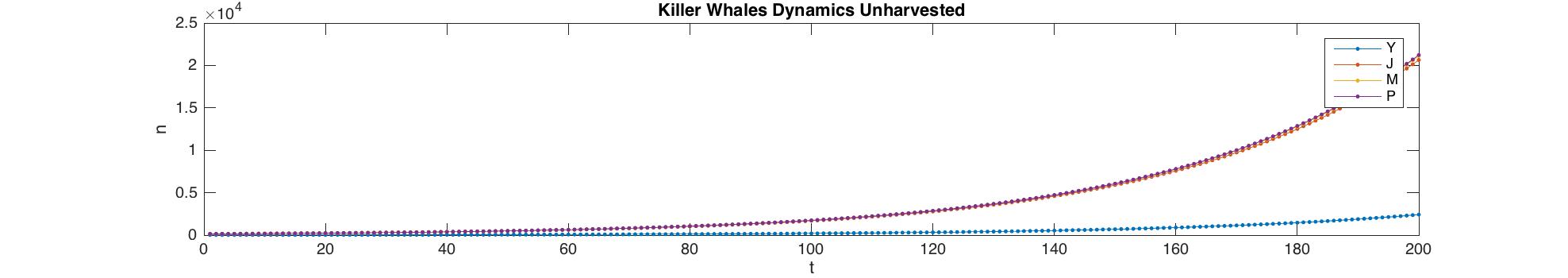
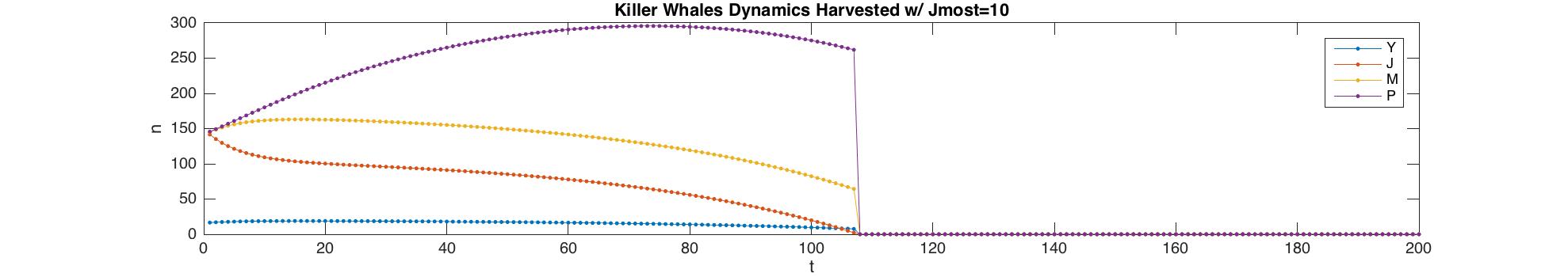
1. **Dwell Time Distribution Theory**
2. **Stimulating Markov Chains and dwell time**
3. **Stimulating Markov Chains and Neural Spiking**

From attached code behind, I calculated *lamda* = 1.0254, and *w* = [0.0663;0.5663;0.5793;0.5825].

Then I simulated the unharvested case in 200 steps (Figure 3).



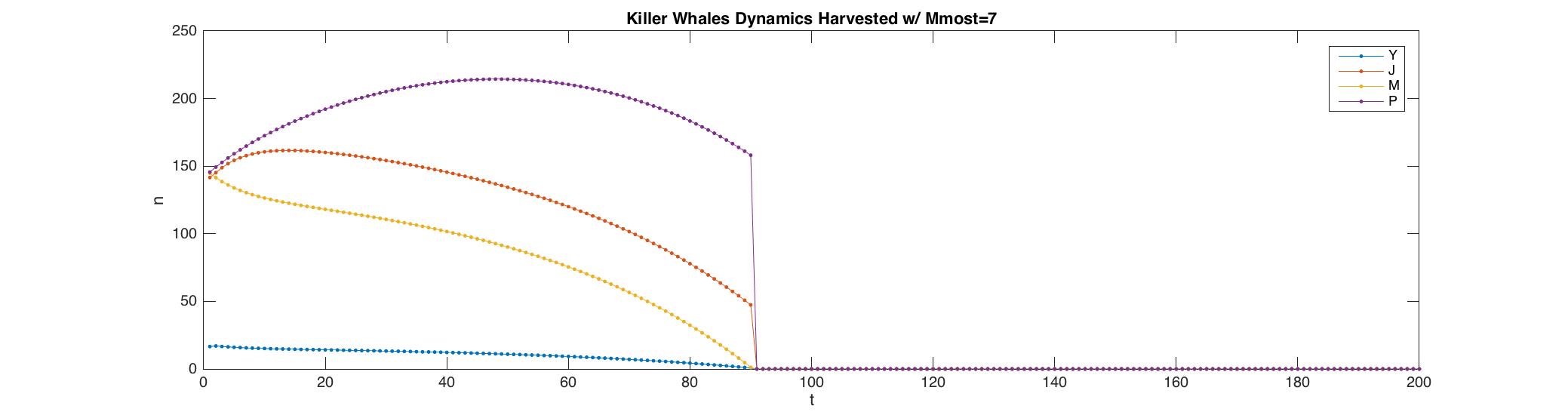
To simulated the harvesting on juveniles, I looped to harvest 1 more juveniles each step and break when it first extinct within 200 steps. The maximum juvenile to take without extinction is **9,** because if we go to 10, it goes to extinction (Figure 4).



To simulated the harvesting on reproductive adults,

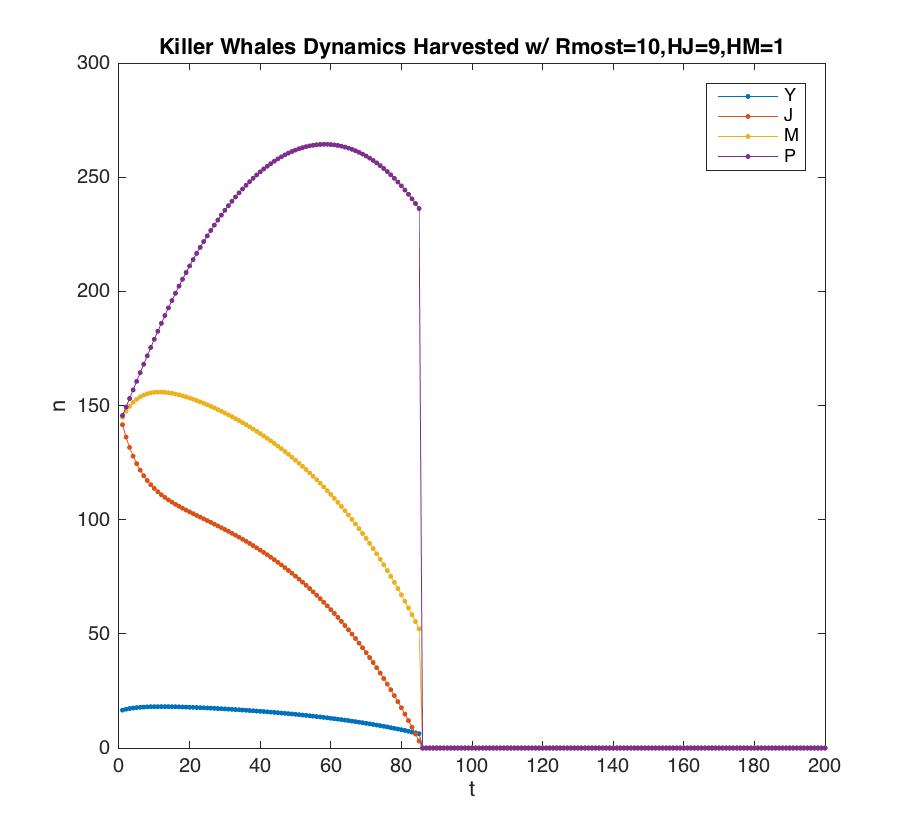
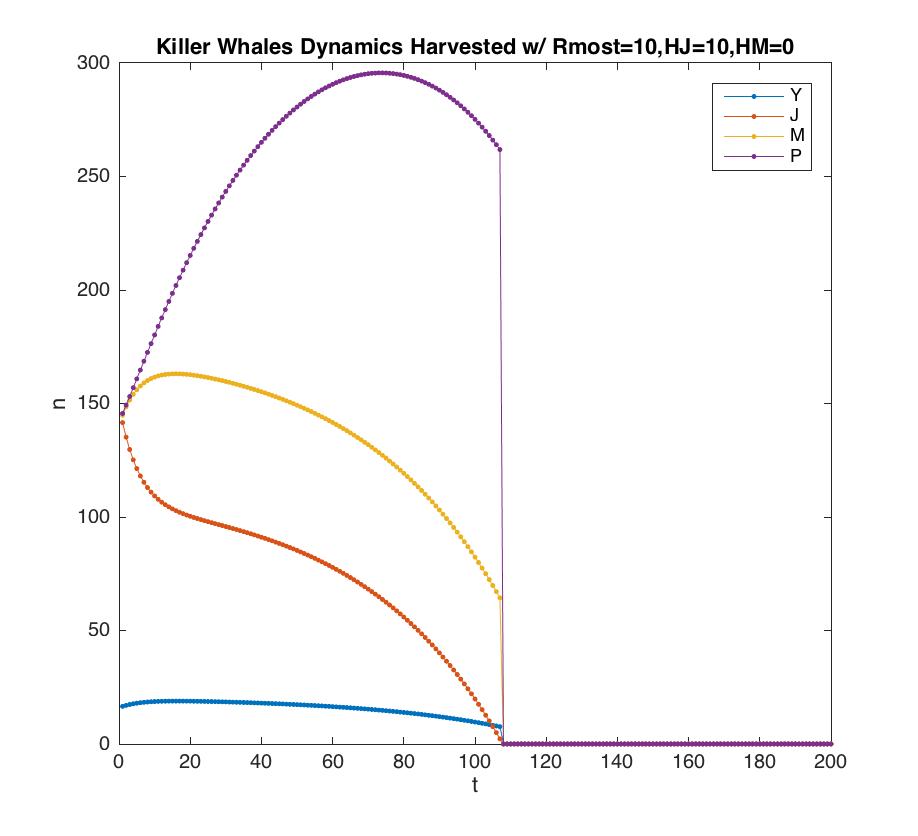
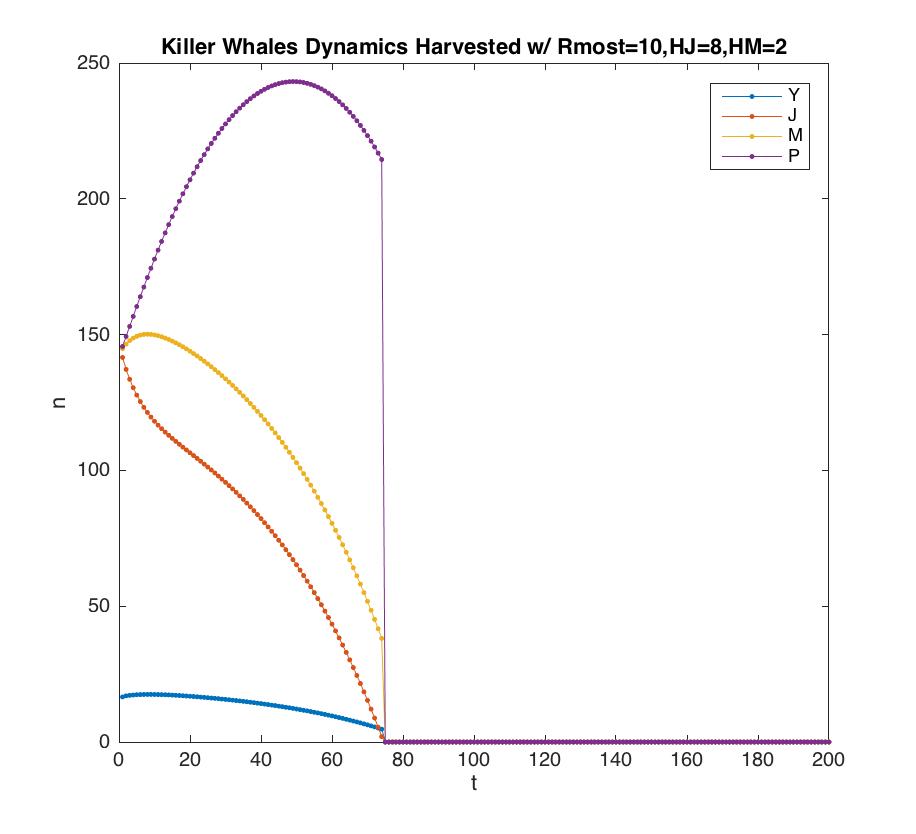
1) If we consider juvenile not as adult, only counting mature:

I looped to harvest 1 more juveniles until maximum and 1 more mature, each step and break when it first extinct within 200 steps. The maximum mature to take without extinction is **6,** because if we go to 7, it goes to extinction (Figure 5).



2) If we consider juvenile as adult as well, not just mature:

I looped to harvest 1 more juveniles until maximum and 1 more mature, each step and break when it first extinct within 200 steps. The maximum juvenile and mature to take without extinction is **9,** because if we go to 10, it goes to extinction (Figure 6, 7, 8).



% @Author: Baihan Lin

% @Date: Oct 2016

%% Question 3

% initiate projection matrix A.

A3 = [0 0.0043 0.1132 0;

0.9755 0.9111 0 0;

0 0.0736 0.9534 0

0 0 0.0452 0.9804]

N = 250;

% calculate dominant eigenvalue lamda

[V3, L3]=eig(A3);

L3 = diag(L3);

j3=find(abs(L3) == max(abs(L3)));

dominant\_eigenvalue=L3(j3) % 1.0254

w3 = V3(:,j3); %[0.0663;0.5663;0.5793;0.5825]

% simulation of unharvested case

n\_zero=w3\*N;

Tmax=200;

n\_vs\_t=zeros(4,Tmax);

n\_vs\_t(:,1)=n\_zero ;

for t=2:Tmax;

n\_vs\_t(:,t)=A3\*n\_vs\_t(:,t-1) ;

end

fig1 = figure;

set(gca,'FontSize',20);

plot(1:Tmax,n\_vs\_t','.-','MarkerSize',14);

xlabel('t','FontSize',20);

ylabel('n','FontSize',20);

title('Killer Whales Dynamics Unharvested');

legend('Y', 'J', 'M','P');

% simulation of harvested case

n\_zero=w3\*N;

Tmax=200;

n\_vs\_t\_harvested=zeros(4,Tmax);

n\_vs\_t\_breakJ=zeros(4,Tmax);

n\_vs\_t\_breakM=zeros(4,Tmax);

n\_vs\_t\_breakR=zeros(4,Tmax);

n\_vs\_t\_harvested(:,1)=n\_zero ;

n\_vs\_t\_breakJ(:,1)=n\_zero ;

n\_vs\_t\_breakM(:,1)=n\_zero ;

n\_vs\_t\_breakR(:,1)=n\_zero ;

Jmost = 0;

Mmost = 0;

RmostSet = [];

RmostJ = [];

RmostM = [];

% find the smallest J possible to get extinction.

% thus the maximum J not to extinct is Jmost - 1

for x = 0:1:100

h = [0;x;0;0];

for t=2:Tmax;

check = prod(((A3\*n\_vs\_t\_harvested(:,t-1)-h)>=0));

n\_vs\_t\_harvested(:,t)=check\*(A3\*n\_vs\_t\_harvested(:,t-1)-h);

end

if (prod((n\_vs\_t\_harvested(:,Tmax)==0))==1)

Jmost = x; %10, the maximum J not to extinct is Jmost - 1 = 9

n\_vs\_t\_breakJ(:,:)=n\_vs\_t\_harvested(:,:);

break;

end

end

figure;

set(gca,'FontSize',20);

plot(1:Tmax,n\_vs\_t\_breakJ','.-','MarkerSize',14);

xlabel('t','FontSize',20);

ylabel('n','FontSize',20);

title(strcat('Killer Whales Dynamics Harvested w/ Jmost=',num2str(Jmost)));

legend('Y', 'J', 'M','P');

% find the smallest M possible to get extinction.

% thus the maximum M not to extinct is Mmost - 1

for x = 0:1:100

h = [0;0;x;0];

for t=2:Tmax;

check = prod(((A3\*n\_vs\_t\_harvested(:,t-1)-h)>=0));

n\_vs\_t\_harvested(:,t)=check\*(A3\*n\_vs\_t\_harvested(:,t-1)-h);

end

if (prod((n\_vs\_t\_harvested(:,Tmax)==0))==1)

Mmost = x; %7, the maximum M not to extinct is Mmost - 1 = 6

n\_vs\_t\_breakM(:,:)=n\_vs\_t\_harvested(:,:);

break;

end

end

figure;

set(gca,'FontSize',20);

plot(1:Tmax,n\_vs\_t\_breakM','.-','MarkerSize',14);

xlabel('t','FontSize',20);

ylabel('n','FontSize',20);

title(strcat('Killer Whales Dynamics Harvested w/ Mmost=',num2str(Mmost)));

legend('Y', 'J', 'M','P');

% find the smallest reproductive adults possible to get extinction.

% thus the maximum R not to extinct is Rmost - 1

for x = 0:1:Jmost

for y = 0:1:100

h = [0;x;y;0];

for t=2:Tmax;

check = prod(((A3\*n\_vs\_t\_harvested(:,t-1)-h)>=0));

n\_vs\_t\_harvested(:,t)=check\*(A3\*n\_vs\_t\_harvested(:,t-1)-h);

end

if (prod((n\_vs\_t\_harvested(:,Tmax)==0))==1)

RmostSet = [RmostSet x+y];

RmostJ = [RmostJ x];

RmostM = [RmostM y];

break;

end

end

end

Rmost = max(RmostSet); % 10, maximum R not to extinct is Rmost - 1 = 9

JRmost = RmostJ(find(RmostSet == Rmost));

MRmost = RmostM(find(RmostSet == Rmost));

% plot each possible J and M breakpoints.

for k = 1:length(JRmost)

x = JRmost(k);

y = MRmost(k);

h = [0;x;y;0];

for t=2:Tmax;

check = prod(((A3\*n\_vs\_t\_breakR(:,t-1)-h)>=0));

n\_vs\_t\_breakR(:,t)=check\*(A3\*n\_vs\_t\_breakR(:,t-1)-h);

end

figure;

set(gca,'FontSize',20);

plot(1:Tmax,n\_vs\_t\_breakR','.-','MarkerSize',14);

xlabel('t','FontSize',20);

ylabel('n','FontSize',20);

title(strcat('Killer Whales Dynamics Harvested w/ Rmost=',num2str(Rmost),',HJ=',num2str(x),',HM=',num2str(y)));

legend('Y', 'J', 'M','P');

end

1. **MATLAB programming tools or tricks**
2. **clear all; close all; clc;**

I like to add this line to all my codes. It ensures no variables and figures get into the way of this round of analysis. I personally find this a good habit to make sure you have a clear workspace before the current programs run.

1. **diary('diary.txt'); diary on;**

I like to add this line to my code, this function diary save text of MATLAB session to a file. Diary function causes a copy of all subsequent command window input and most of the resulting command window output to be appended to the named file. I personally find this a good habit to keep a record of all the output. This is also very helpful when debugging.

1. **prompt = 'What is your folder?: '; path = input(prompt,'s');**

I like to use this line to create some interaction in my MATLAB session. This is very helpful for other people to run your code in a more general base. This facilitates coding collaboration.

1. **Project Warmup, E+G Ex. 1.1**
2. **Complete citation**

Yu, J., Xiao, J., Ren, X., Lao, K. and Xie, X.S., 2006. Probing gene expression in live cells, one protein molecule at a time. *Science*, 311(5767), pp.1600-1603.

1. **Purpose of the model**

This article tries to stochastically model the gene expression of a fusion protein of a fast-maturing yellow fluorescent protein (YFP) and a membrane-targeting peptide under a repressed condition. And they tried to use this model to explain the mechanism of the in vitro real-time production of single protein molecules in individual *Escherichia coli* cells.

1. **State variables**

Because it is in a single protein molecule level model, the state variables are the numbers of Tsr-Venus, *lac* repressor, *lac* operator, mRNA, ribosome molecules.

1. **Simplifying assumptions**

When they assigned the rate limiting step for protein production, they made the assumption that one mRNA molecule per gene expression burst, when this is not always true. They also derive the autocorrelation function based on the following three assumptions: (1) the transcription initiation events are temporally uncorrelated (i.e. a Poisson processes); (2) the number of protein molecules translated from one mRNA follows a geometric distribution; (3) there is one rate-limiting step in the post-translational assembly of the protein, and transcription/translation times are short.