Poisson Process Model for Coupled Logistic System

Table of Contents

Introduction

This program models the population dynamics of two species using poisson stochastic modelling. The growth profile are modelled using coupled logistic system Governing equations for growth and death rate: dx/dt = r1x(1-(x+ay)/k1) dy/dt = r2y(1-(y+bx)/k2) The two species modelled in this program represenst wild type (WT) MG1655 bacteria and MG1655 mutant(contains plasmids). r1,k1,k2 are found in the literature while r2 depends on several factors and is implemented at the begninning of the program

The program varies a,b in the coupled logistics eqautions as well as CRSIPR efficiency, which will change the initial WT and mutant population.

The program run the poisson stochstics model several times for each condition to obtain statistically significant data. For each condition, it calculates 5 output parameters: WT and mutant population at the end of smulation, WT population/total population(ratio) and standard deviation of WT and mutant population. The result is summarized in a table and exported to .xlsx file. The program can also generate 3-D graphs for any input and output variables

The input parameters are as follows: rconstWT (r1): growth rate constant for WT MG1655 KconstWT (k1): carrying capacity for WT MG1655 KconstMut (k2): carrying capacity for mutant plasmid_size: size of plasmid transformed into mutant (unit in kb) copy_number: nubmer of plasmid in each mutant nsteps: each step of the poisson simulation represents an hour, nsteps specifies how many step the model will run nruns: number of trials for each condition

The output variable of this program, output_data, is a cell matrix containing values of input and output parameters for each condition with a header in the first row. The columns are as follows: 1. r1, 2. r2, 3. k1, 4. k2, 5. a1, 6. a2, 7. p1, 8. p2, 9. avg_prey1 10. avg_prey2, 11. popratio, 12. stdev_prey1, 13. stdev_prey2

Published on August 15, 2014 Credit to Matthew Badali, Department of Physics, University of Toronto

default function call: coupled_logistics_poisson_simulation2(2.5076557,3.5e10,3.5e10,8,20,24,10)

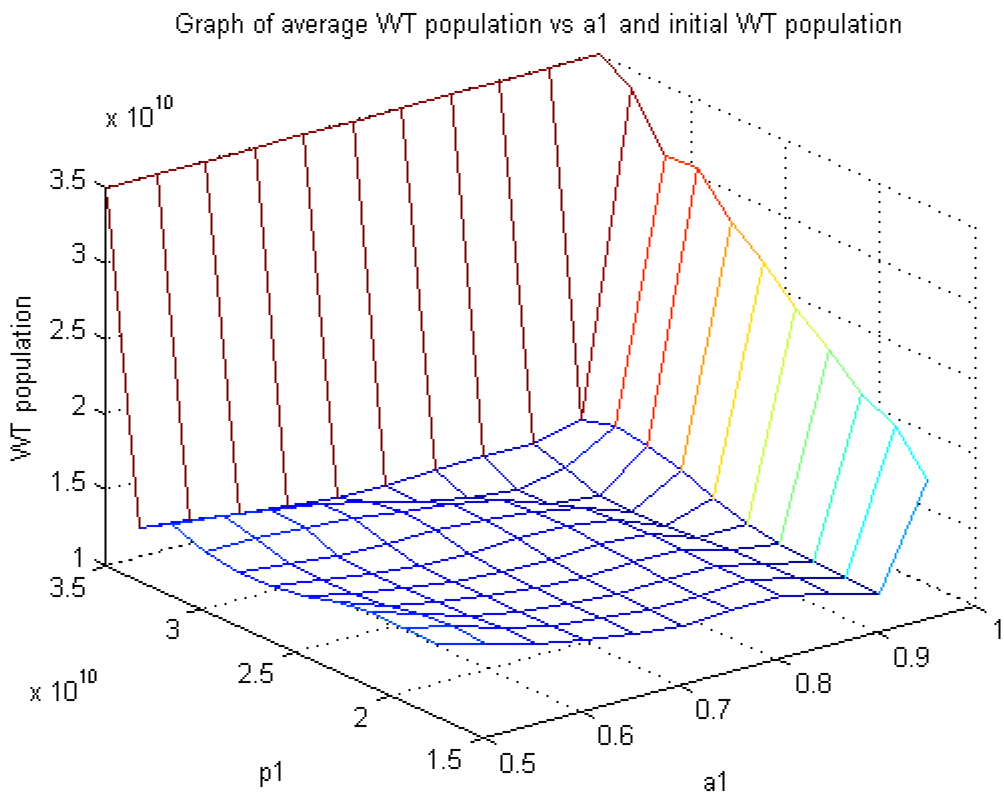
Implementation

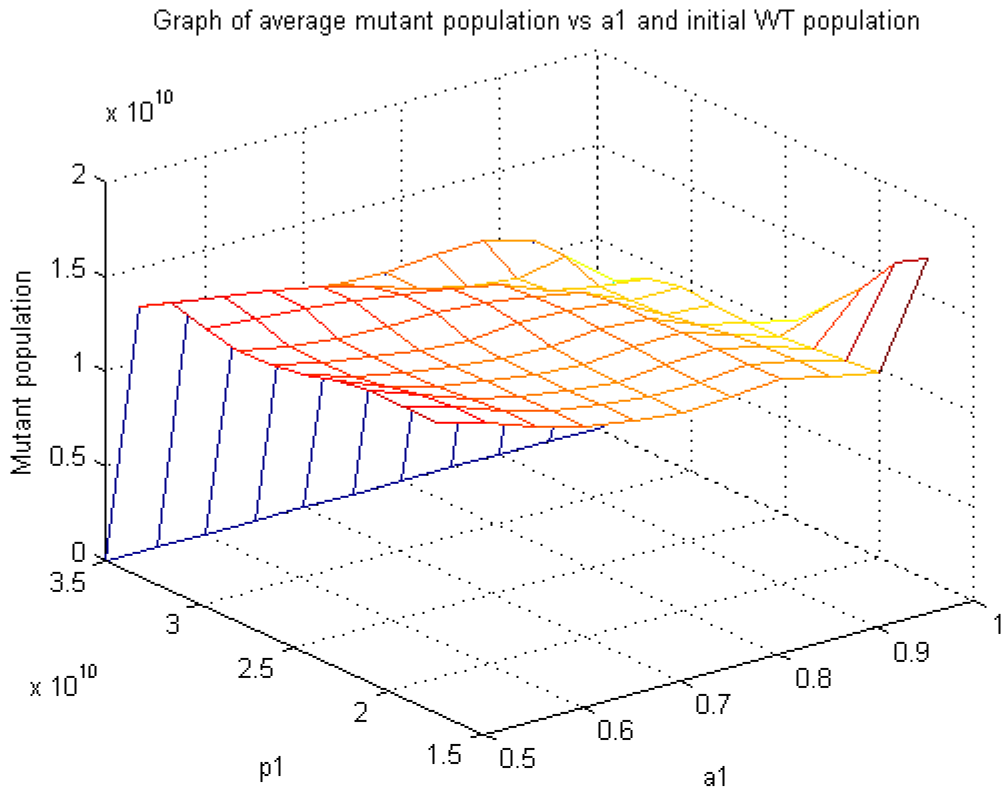
main function

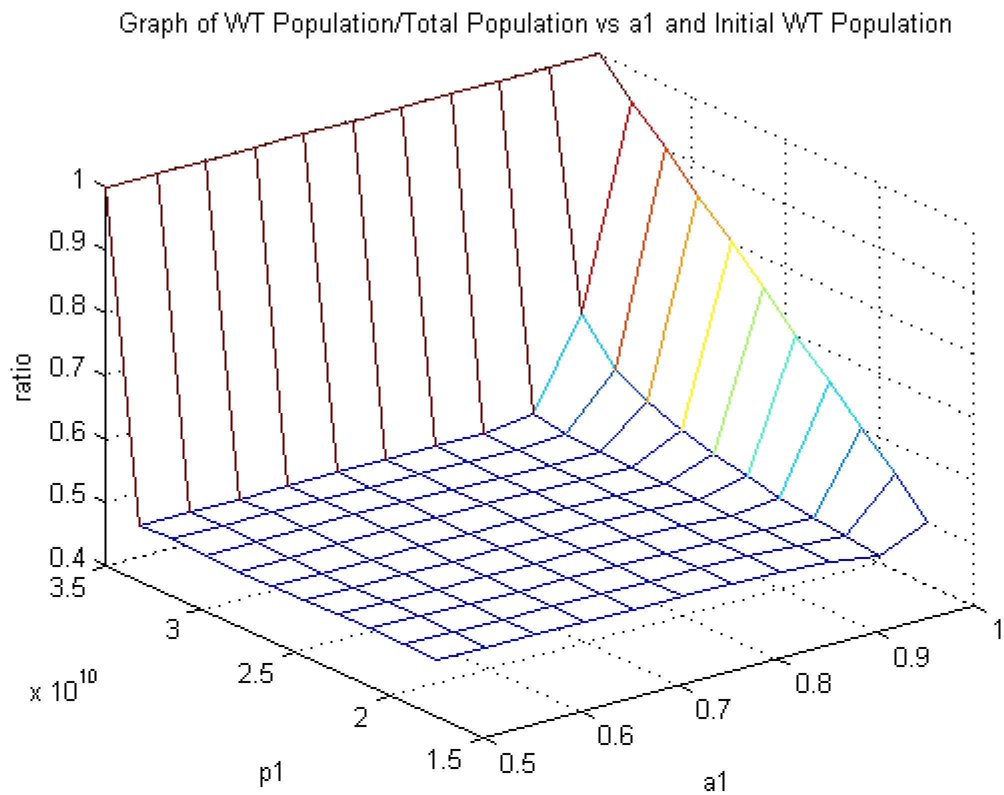
```
function output_data = coupled_logistics_poisson_simulation2(rconstWT,...
    KconstWT, KconstMut, plasmid_size, copy_number, nsteps, nruns)
```

```
%modify r2 based on r1 and total plasmid DNA
%slope is normalized by e^(Rmut-Rwt),Rwt is normalized to one
rconstMut = rconstWT + log(1-(7.2*10e-5)*plasmid_size*copy_number);

%header of the output file
output_data = {'r1','r2','k1','k2','a1','a2','p1','p2','avg_population1'...
    ,'avg_population2','ratio', 'std_pop1', 'std_pop2'};

%indicate range of conditions before running the simulation
interval_a = 0.05; %decrement by 0.01
upper_bound_a = 1;
lower_bound_a = 0.5; %a ranges from 1~0.9
total_points_a = round((1-lower_bound_a)/interval_a)+1;
interval_b = 0.05;
upper_bound_b = 1;
lower_bound_b = 0.5;
total_points_b = round((1-lower_bound_b)/interval_b)+1;

%vary a1, a2
for a = linspace(upper_bound_a,lower_bound_a, total_points_a)
    alpha = a;
    beta = alpha;

    %vary efficiency
    for b = linspace(upper_bound_b,lower_bound_b, total_points_b)
        efficiency = b;
        nWT0 = KconstMut*efficiency;
        nMut0 = KconstMut*(1-efficiency); %initial population n1, n2
        t0 = 0;

        %run simulation
        [etime_average, extinction_prob, lived, prey1, prey2] = ...
            manyruns(nruns, t0,nWT0,nMut0);

        %Summarize results
        [avg_prey1, avg_prey2, popratio, stdev_prey1, stdev_prey2]...
            = popcal(nruns, prey1, prey2);

        %Moved the next line to be inside the second for loop because
        %initial population depends on efficiency, which is varied in this
        %loop
        %Add rows to existing data after each loop
        output_data = [output_data; {rconstWT, rconstMut, KconstWT,...
            KconstMut, alpha, beta, nWT0, nMut0, avg_prey1, avg_prey2,...
            popratio, stdev_prey1, stdev_prey2}];
    end
end


%Output do excel
filename = 'testdata2.xlsx';
xlswrite(filename,output_data);
```

```
%Plot graphs
%1. r1, 2. r2, 3. K1, 4. K2, 5. a1, 6. a2, 7. p1, 8. p2, 9. avg_prey1
%10. avg_prey2, 11. popratio, 12. stdev_prey1, 13. stdev_prey2

%function definition: plotgraph(x,y,z,x_label, y_label, z_label)
plotgraph(5,7,9, 'a1', 'p1', 'WT population')
title('Graph of average WT population vs a1 and initial WT population')
%Graph of Average Mutant Population vs a1 and Initial WT Population
plotgraph(5,7,10, 'a1', 'p1', 'Mutant population')
title('Graph of average mutant population vs a1 and initial WT population')
%Graph of WT Population/Total Population vs a1 and Initial WT Population
plotgraph(5,7,11,'a1', 'p1', 'ratio')
title...
('Graph of WT Population/Total Population vs a1 and Initial WT Population')
```




Graph of average mutant population vs a1 and initial WT population
x 10^10
2
1.5
1
0.5
0
Mutant population
3.5
3
2.5
2
1.5
x 10^10
p1
0.5
0.6
0.7
0.8
0.9
1
a1


Graph of WT Population/Total Population vs a1 and Initial WT Population
1
0.9
0.8
0.7
0.6
0.5
0.4
ratio
3.5
3
2.5
2
1.5
x 10^10
p1
0.5
0.6
0.7
0.8
0.9
1
a1

Sub functions

```
%runs simulation for "nruns" times given initial populations
function [etime_average, extinction_prob, lived, prey1, prey2] =...
        manyruns(nruns, t0,nWT0,nMut0)
    extinction = zeros(1,2);
    extinction_prob = zeros(1,2);
    etime = zeros(1,2);
    etime_average = zeros(1,2);
    lived=0;
    time=zeros(1,nruns);
    prey1=zeros(1,nruns);
    prey2=zeros(1,nruns);

    for j  = 1:nruns
        [temp_nWTtemp,temp_nMut,temp_time] = runVerhulst2(nWT0,nMut0,t0);
        prey1(j) = temp_nWTtemp;
        prey2(j) = temp_nMut;
        time(j) = temp_time;
        %if first population dies, then add 1 to extinction event,
        %then add up all the extinction time for all extinction
        if temp_nWTtemp == 0
            extinction(1) = extinction(1) + 1;
            etime(1) = etime(1) + temp_time;
        %same as previous one except for another species
        elseif temp_nMut==0
            extinction(2) = extinction(2) + 1;
            etime(2)= etime(2) + temp_time;
        %no population died within specified runs (steps)
        else
            lived = lived + 1;
        end
    end

    for i = 1:length(extinction) % i is number of species
        if extinction(i)~= 0
            %average extinction time only for runs that
            %encoutered extinction
            etime_average(i) = etime(i)/extinction(i);
            % # of times one species becomes extinct
            extinction_prob(i) = extinction(i)/nruns;
        end
    end
end

%runs one step of the Poisson algorithm
function [nWT,nMut,time] = runVerhulst2(nWT0,nMut0,t0)

    nWT = nWT0;
    nMut = nMut0;
    time = t0;
    %defining instananeous growth rates
    while time<nsteps && nWT>0 && nMut>0
```

```
            rateWTGrowth = rconstWT*nWT;
            rateWTDeath = nWT*rconstWT*(nWT + alpha*nMut)/KconstWT;
            rateMutGrowth = rconstMut*nMut;
            rateMutDeath = rconstMut*nMut*(beta*nWT + nMut)/KconstMut;

            WTgrowth = poissrnd(rateWTGrowth);
            WTdeath = poissrnd(rateWTDeath);
            Mutgrowth = poissrnd(rateMutGrowth);
            Mutdeath = poissrnd(rateMutDeath);

            nWT = nWT + WTgrowth - WTdeath;
            nMut = nMut + Mutgrowth - Mutdeath;
            time = time + 1;
        end

    end

    %summarize results for all trials for a particular condition
    function [avg_prey1, avg_prey2, popratio, stdev_prey1, stdev_prey2] =...
            popcal(nruns, prey1, prey2)

        stdev_prey1 = std(prey1,1);
        stdev_prey2 = std(prey2,1);
        avg_prey1 = sum(prey1)/nruns;
        avg_prey2 = sum(prey2)/nruns;
        totalPop = avg_prey1+avg_prey2;
        popratio = avg_prey1/totalPop;

    end

    %plot 3-D graphs, x,y are indep variables (usually input variables)
    %and z is the dependent variable (usually output variables)
    function plotgraph(x,y,z,x_label, y_label, z_label)
        %turn output_data from cell array to a matrix for plotting purposes
        data = cell2mat(output_data(2:size(output_data,1),:));
        %5 denotes a1, 7 denotes p1 and 11 denotes popratio. See
        %introduction for references
        x = reshape(data(:,x),length(data(:,x))/total_points_a,total_points_a);
        y = reshape(data(:,y),length(data(:,y))/total_points_a,total_points_a);
        z = reshape(data(:,z),length(data(:,z))/total_points_a,total_points_a);
        %eg, if the array is 333222111 then it is reshaped into 333
        %222
        %111 in order to create a meshgrid for 3-D plotting

        %plot graph
        figure
        mesh(x,y,z)
        xlabel(x_label);
        ylabel(y_label);
        zlabel(z_label);

    end

    end
```

Published with MATLAB® R2013a