# Stephanie Mecham

EPID 634

# **Lab #2 - Population dynamics**

# **Introduction to Simulink.**

The following section provides an overview of the basic syntax of Simulink. Simulink is a software package for modeling simulating, and analyzing dynamical systems. It provides a graphical interface for building models as block diagrams, using click-and-drag mouse operations. With this interface, you can draw the models just as you would with pencil and paper.

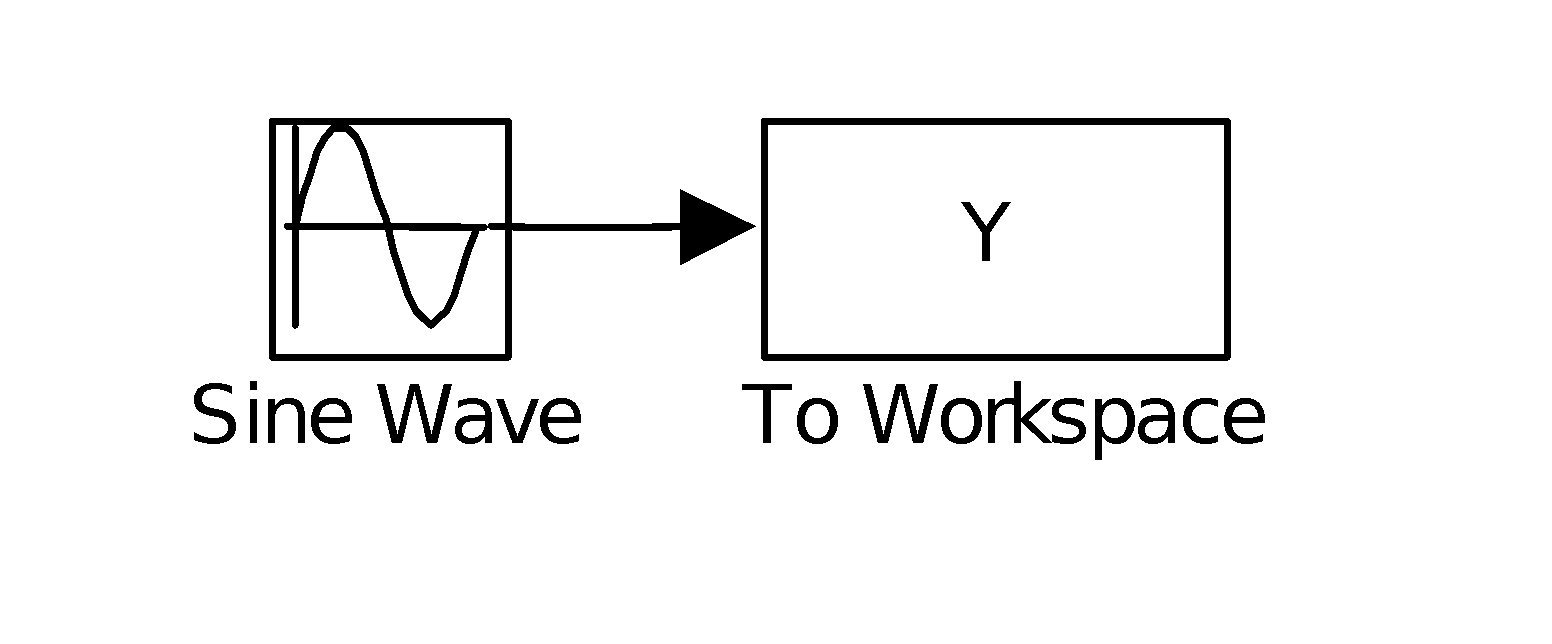
First download the file 'Epid634.mdl' to your local directory. To startup simulink, type 'Epid634' in the MATLAB command window. A simulink library window will open. This library contains all the blocks that you will need to build models for this class. To access the complete collection of functional blocks offered by MATLAB type ‘simulink’.

Models are built within windows. You can start building a new model by selecting File/New/Model from the library window.

## ***1. Building a Model using Simulink***

First let's explore the different block structures and how then interconnect. If you examine the block icons, you see an angle bracket on the right of the Sine Wave block and one on the left of the Workspace block. The > symbol pointing out of a block is an *output port*; if the symbol points to a block it is an *input port*. A signal travels out of an output port and into an input port of another block through a connecting line. When the blocks are connected, the port symbols disappear.

* Click-and-drag the sine wave and Workspace blocks from the library to the untitled window and connect the blocks. To do this, use the mouse to position the pointer over the output port on the right side of the Sine Wave block. Notice that the cursor shape changes to cross hairs. Hold down the mouse button and move the cursor to the input port of the Workspace block. Notice that the line is dashed while the mouse button is down and that the cursor shape changes to double-lined cross hairs as it approaches the Workspace block. Now release the mouse button. You can get rid of any block or connecting line by clicking on it and then typing the delete button. Connect the two blocks to look like the following:



To run this simulation

1. Double click on the Workspace block and change the format to array (the default is structure).
2. Double click on the Sine Wave block. This will open up a window allowing you to adjust the parameters of the sine wave generator. Set the frequency to 0.1 and then click on the 'apply' and then 'OK' buttons.
3. Under the simulation menu select 'Parameters' and change the 'stop time' to 100. Then to run the simulation select 'Start'. The solver should be set to ‘discrete’ and Step Size = 2.
4. Type plot(tout,Y)

**Note: Make sure that all work space blocks are outputting in “array” vs. “structure” format**

The Figure window will appear. Selecting 'parameters' under the simulation menu can access a few parameters that you should be aware of when running simulations. The most important parameters are the following:

start time this value will usually be zero

stop time this will vary depending on how long you want the simulation to run.

solver options this option is most important when simulating differential equations. We will leave this at the default setting.

max step When simulating a continuous function such as a sine wave, MATLAB calculates the value of the function at discrete intervals. The *step size* is the size of that interval. The time variable is incremented by this step size before the specific equation being solved is recalculated. The increment will never be greater than the *max step* size. When using a fixed interval the step size and the max step are always equal. Sometimes it is more efficient to vary the interval based on the property of the function. In general, the larger the step size the faster the simulation but the less smooth the trajectory

The next two tasks illustrate the importance of this last feature when using periodic signals such as sine waves.

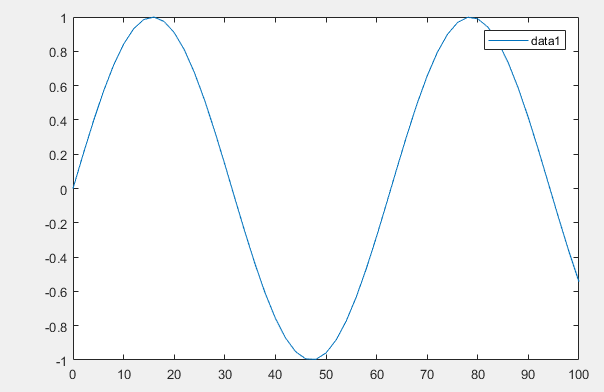
Task1: What is the frequency and period of our sine wave?

**Frequency is 0.1/2pi**

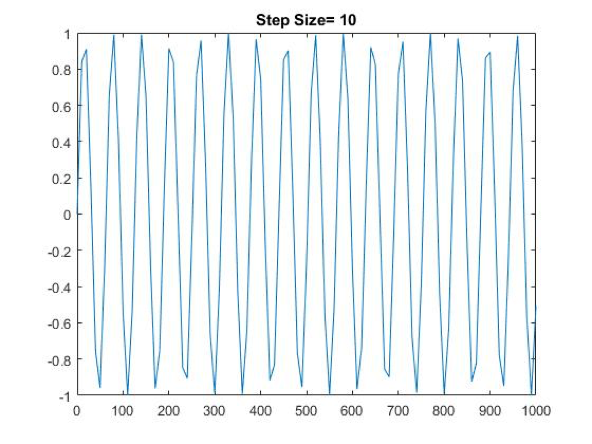
**Period is 2pi/0.1**

Task2: Try increasing the maximum step size until the sine wave looses it smoothness. Use 1, 6, 10, 30, and 70. You will need to change the 'stop time' to 1000 (under the 'simulation'/'parameter menu). What is happening to the period of the sine wave? What is a good rule-of-thumb for setting the maximum time step when using a periodic signal?

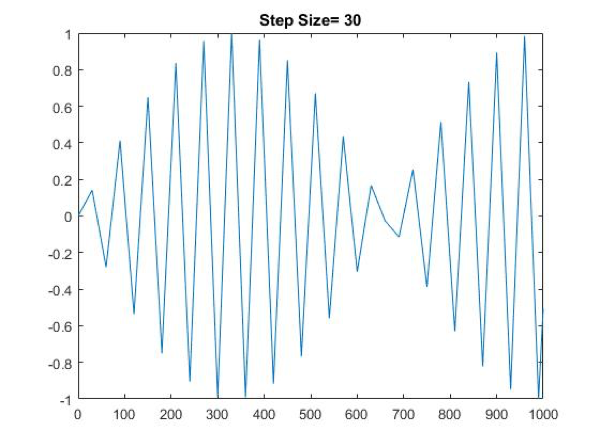
Step size=1



Step size = 10

****

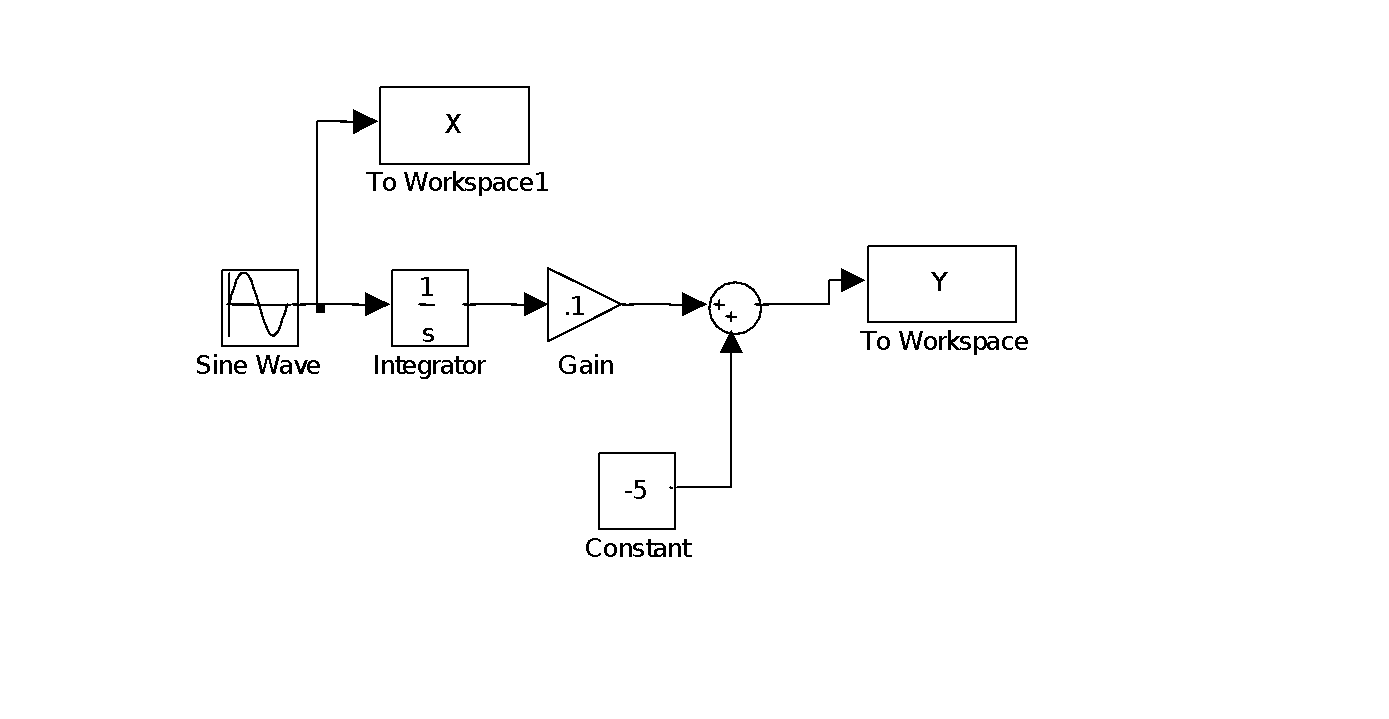
Step size=30

****

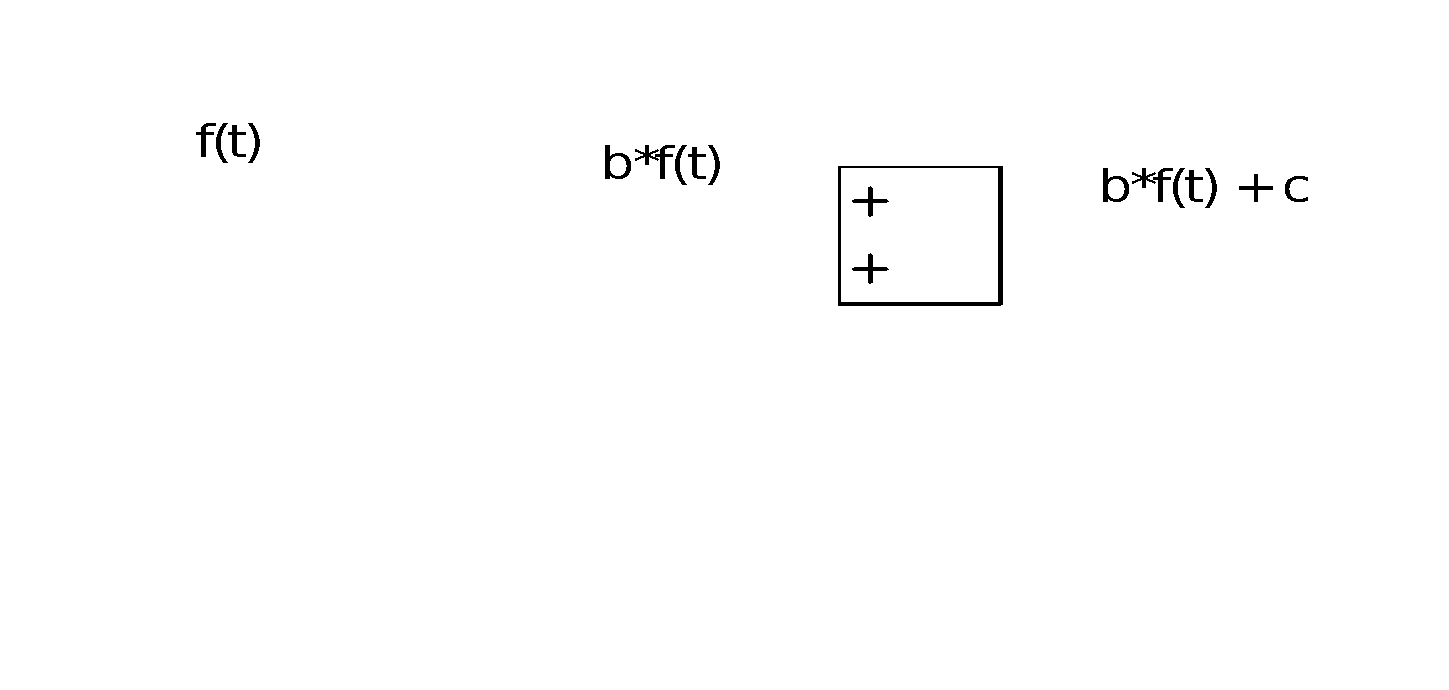
**As you increase the step size, you decrease the period and increase the frequency of the sine wave. A good rule of thumb is that the step size has to be less than ¼ of the period.**

* Now lets combine a few more of these function blocks to see how they work. This next model integrates a sine wave, scales it by 0.1, and adds the constant, -5. Again it is plotted against time. Build this model and run a simulation with the frequency equal to 0.1 and the amplitude equal to 5 (set max step size back to ‘auto’ and stop time to 100).

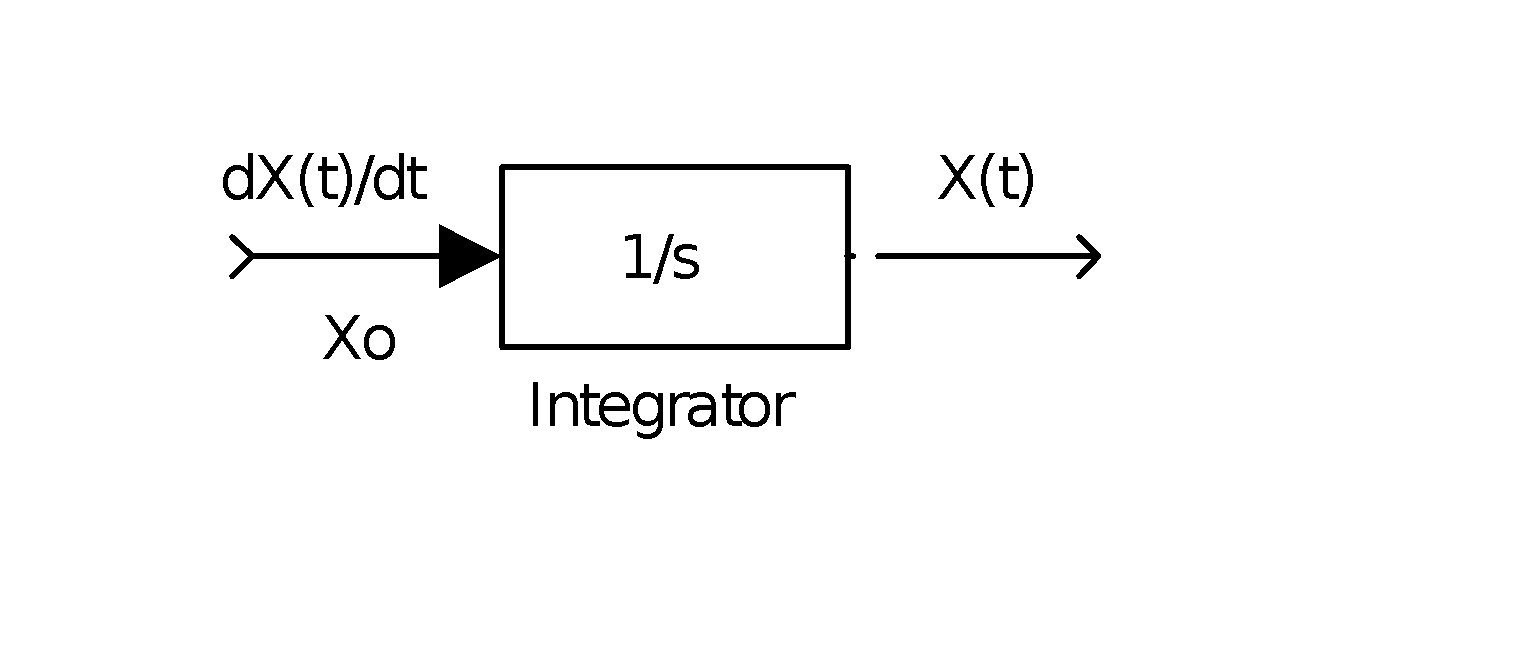
**Hint: Make sure all workspace blocks are outputting in an ‘array’ vs. a ‘structure’ format**



Note that we are using a few new functional blocks. The following diagrams the functional attributes of these blocks



The other block that needs some explanation is the integrator block



This block integrates a function from an initial to final time (defined as the start and stop time in the parameter window within the simulation menu), using the initial condition, Xo (set by double clicking on the integrator block).

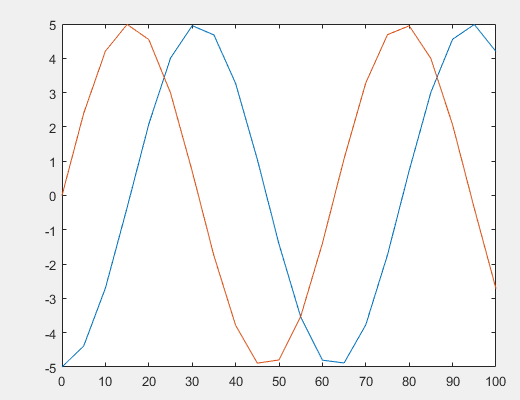
Task3: Run the simulation and plot the input sine wave, X, and the output, Y

plot(tout,Y,’--‘);

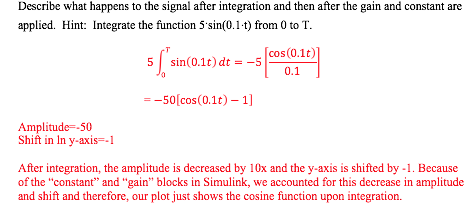
hold on;

plot(tout,X);

Describe what happens to the signal after integration and then after the gain and constant are applied. Hint: Integrate the function 5.sin(0.1.t) from 0 to T.



Graphically, we can see the waves are shifted. Adding the constant changes the amplitude.



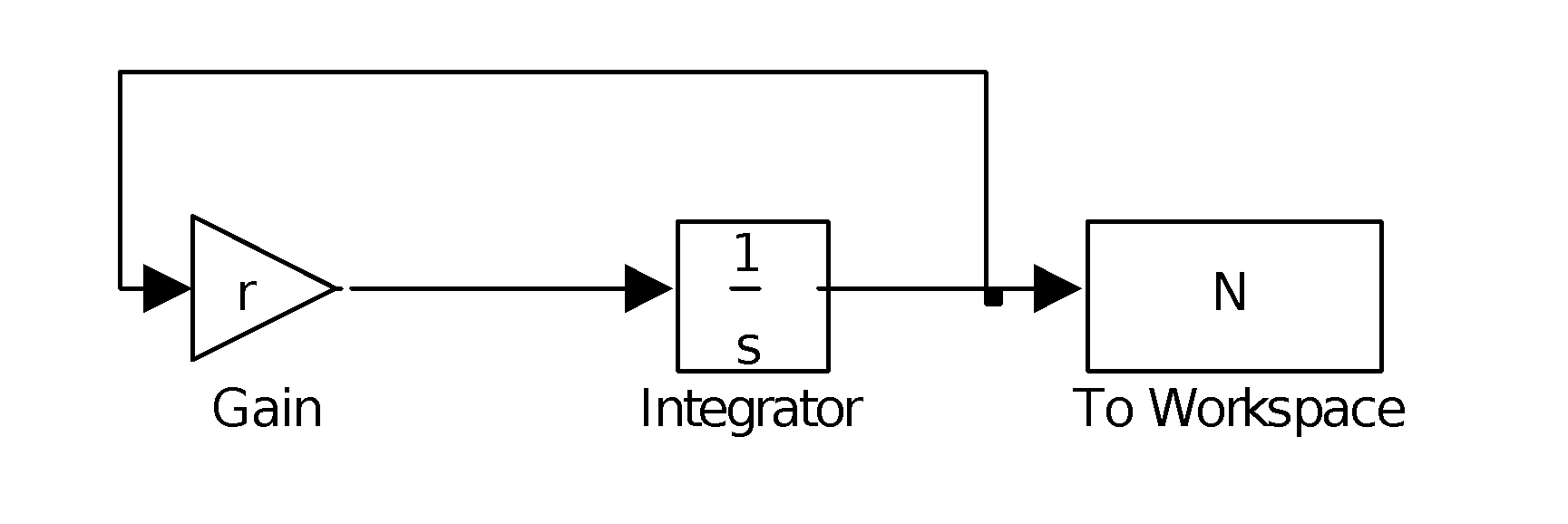
Results of integration shows that the y axis shifts by -1 and the amplitude decreases to -50.

## ***Differential equations models using Simulink***

### Exponential growth model

Now we are ready to develop differential equation models. We will start with an exponential (Malthusian) growth model. The equation for this model is:

In Simulink differential equations are represented graphically using the integrator block. If the output of the integrator block is N then the input is *dN/dt*. Therefore the input and the output are related to each other by a factor of r. This can be shown graphically as follows



Task4: Build and simulate this model, and then plot the output *N* vs *t*.

Note:

1. The gain value is r. This value can be defined in the workspace (set r=2).
2. To simulate this model you first need to provide an initial condition, by double clicking on the integrator block and providing an initial condition.
3. Set the ‘stop time’ = 10.

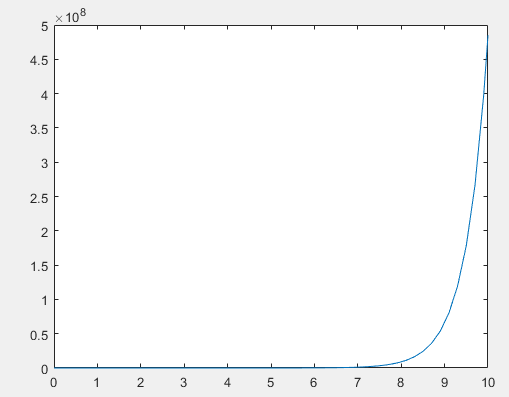
Calculate the per capita change in population level at different time points.

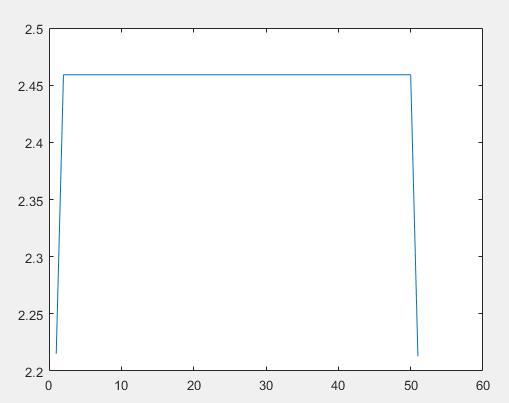
You can do this graphically by either: 1. using marks on the figure (type help 'grid' for details) and adjusting the axes to zoom in on a small region or; 2. using the values *N* and *t* in the workspace. For this second method, you can use a similar technique as when you calculated the doubling times in Lab 1. In this case you want to calculate Δ*N*/Δ*t* / *N*.

(Hint: If you have a time variable that is a row vector containing an unknown number of elements, then you can use the *size* function to calculate the vector length. Therefore, tout( 2:size(tout,1) ) - tout( 1:size(tout,1)-1 ) would result in a vector of Δ*t* values.)

Using the second method you will get a vector of estimates for ‘*r*’ at different time points.

By examining the ‘*r*’ values in the resultant vector, what general property of exponential growth have you shown? Why might your resultant vector not output the exact value of r? How would you increase the accuracy of your estimate?



****

**Code:**

**deltaT=tout( 2:size(tout,1) ) - tout( 1:size(tout,1)-1 )**

**deltaN=N( 2:size(tout,1) ) - N( 1:size(tout,1)-1 )**

**dNdT=deltaN./deltaT**

**N\_agree= N(1:size(N,1)-1)**

**r=dNdT./N\_agree**

**plot(tout,N)**

**plot(r)**

**output:**

**2.2151**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.4591**

**2.2129**

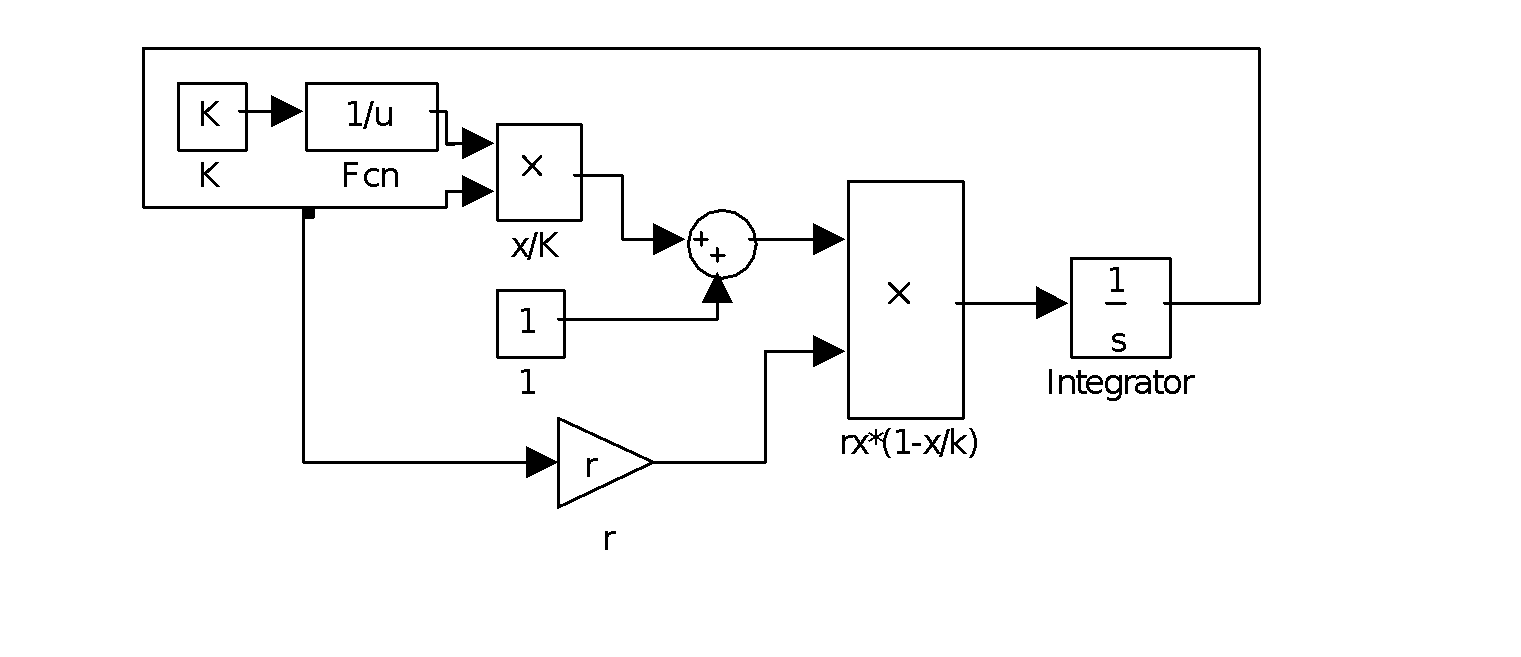
We have shown that the derivative of an exponential growth rate will be constant. Our output/graph is not entirely accurate at the extreme low and extreme high ends, and this is likely due to the code only being a close approximation of the derivative (i.e. calculates the instantaneous rate of change, which will be less accurate at the earliest and latest time steps). We can improve the accuracy of this calculation by decreasing the step size.

### Logistic growth model

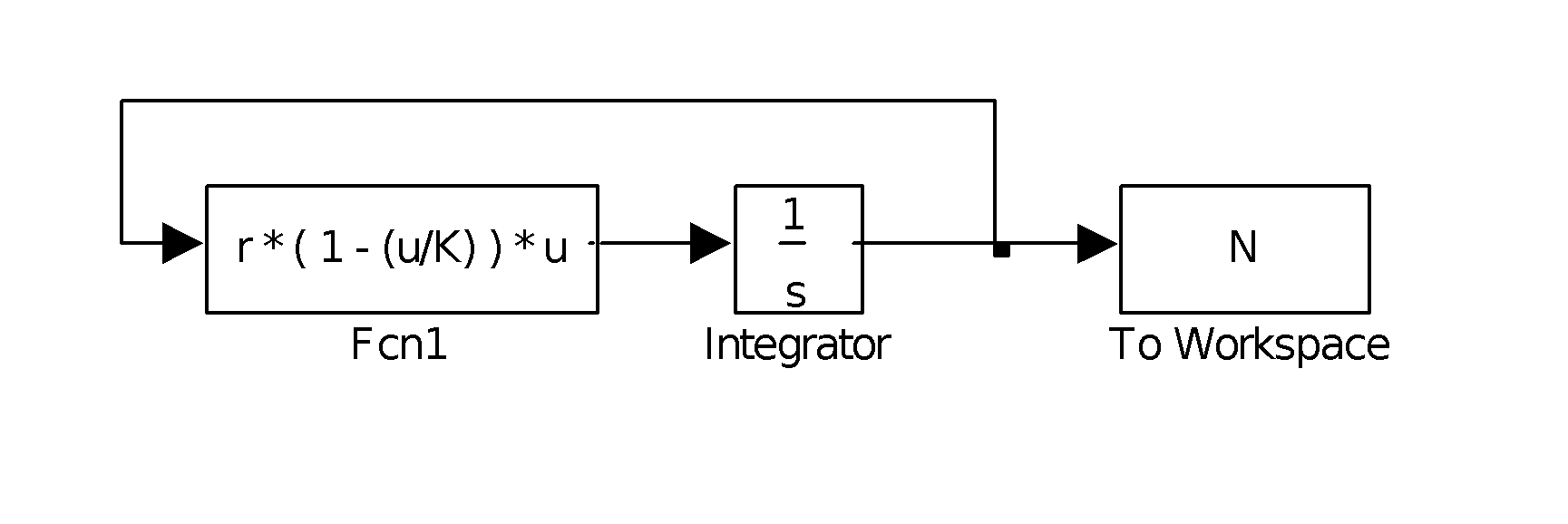
The logistic model, as presented in lecture, incorporates regulation in a population model by assuming that the *per capita* growth rate is a decreasing function of the current population level. As you recall, the equation is as follows:



In the same way we built the Simulink model for exponential growth, we arrive at the following diagram for the logistic model.



The one new block used in this model is the Fcn block or the ‘General Expression’ block. You can double click on the block in the library to get an idea of how it works. In this case it takes the inverse of the input value. To implement this block you just need to click-and-drag the Fcn block to your window, double click on the block, and type 1/u for the expression. As you can see these diagrams can get complicated very quickly.

Another easier approach is to use the Fcn block to describe the complete expression on the right side of the differential equation. Then the model will look like the following:

#### where Fcn1 contains two parameters, r and K, and a variable, u, which represents the input (in this case it represents the variable N).

Task 5: Construct this model (or use logistic2.mdl) and by trial and error find estimates for r and K that will fit this model to the human population data in file humpop.txt. This data is U.S. census data from 1790 to 1910.

Hint: Based on the *humpop* dataset, define the initial condition in the integration box and the initial and final times in the dialog box under the simulation/parameters menu. For efficiency try the following, where No is set as the initial condition in the integration block, and K, r, No are set in the workspace (make sure you use capital K as that is how the variable is defined in the Simulink model.

hold on;

No = 3929214;

sim(‘logistic2’,[1790, 1910]);

plot(humpop(:,1), humpop(:,2), ‘\*’); plot(tout, N);

grid on;

hold off;

What does this model predict for the current population (1990-2000)? When will the population level saturate and at what level?

hold on;

No = 3929214;

sim('logistic2',[1790, 2050]);

plot(humpop(:,1), humpop(:,2), '\*'); plot(tout, N);

grid on;

hold off;

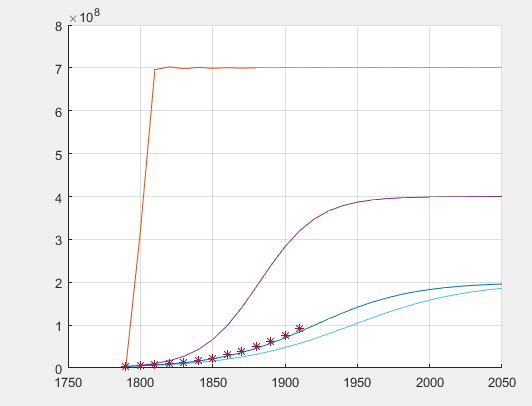
Tries:

r=0.5 & K = 7,000,000,000

r=0.05 & K = 4,000,000,000

r= 0.025 & K = 2,000,000,000

r= 0.030 & K = 2,000,000,000



At around the year 2000, the population was approximately 1.8x108. Population saturation occurs at approximately the year 2050, at a population level of 2x108.

Pearl and Reed proposed this model in 1920 as a departure from the empirical methods of curve fitting. Their approach was to develop a model based on the following biological hypotheses: that the rate of population increase in a limited area at any instant of time is proportional to a) the magnitude of the population existing at that instant and b) the still unutilized potentialities of population support existing in the limited area.

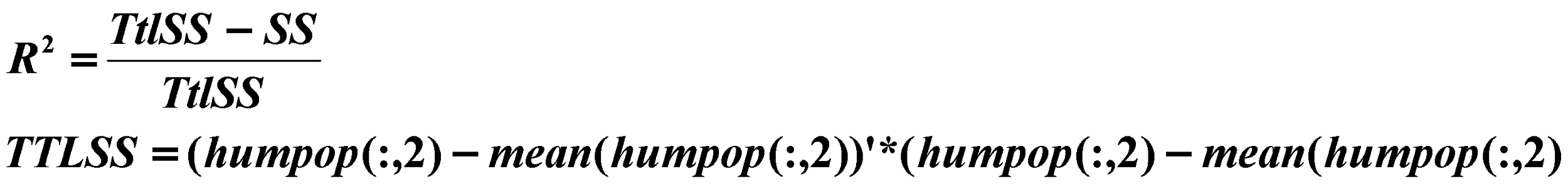
The current U.S. population is approximately 250 million. Given the task of predicting 80 to 100 years into the future, based on population data up to 1920, small errors in the data may produce large changes in projections.

One way to assess this sensitivity of the model is to calculate the change in sum-of-squares for various changes in the parameter values. A technical difficulty in the sum-of-squares calculation is that the time values from the model do not coincide with the time values of the data. To address this issue MATLAB provides an interpolation function, ‘interp1’. Interpolating the human population data is achieved using the following syntax:

N1 = interp1(tout, N, humpop(:,1))

The variables tout and N are the simulated time and population values respectively, humpop(:,1) are the time points from the data, and N1 are the interpolated population values. The variable N1 contain predicted population values (model generated) that correspond to the time points reflected in the data ( humpop(:,1) ). Thus, we create population levels from our simulation output that directly corresponds with the data.

Task 6: Use this interpolation function to calculate the sum-of-squares, and fine-tune your estimates of *r* and *K*. Then calculate the percent change in the *R2* for various changes in the parameter estimates, where:



Discuss how sensitive R2 is to the parameters *K* and *r*. You can you the script PearlReed.m for this task.

Sum-of-squares: .947

Attempts at fine-tuning:

Changing r’s

r = 0.027 & K = 2 x 108 (R2 =0.8523)

r= 0.029 & K= 2 x 108 (R2 = 0.9590)

r= 0.030 & K= 2 x 108 (R2 = 0.9885)

r= 0.031 & K = 2 x 108 (R2=0.9996)

r=0.032 & K = 2 x 108 (R2=0.9907)

Changing K’s:

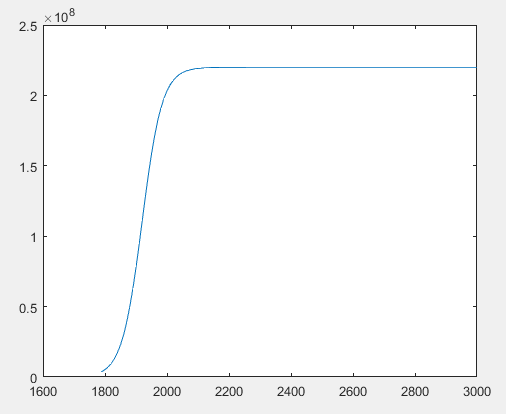
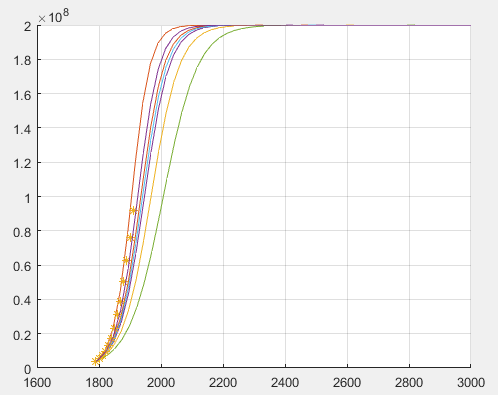
r= 0.031 & K = 1.9 x 108 (R2=0.9972)

r= 0.031 & K = 2 x 108 (R2=0.9996)

r= 0.031 & K = 2.1 x 108 (R2=0.9996)

r= 0.031 & K = 2.2 x 108 (R2=0.9989)

Highlighted options above appear to be the best fit.



Percent change from r = 0.029 to 0.030, holding K constant at 2x108 = 3.076% change

Percent change from r = 0.030 to 0.031, holding K constant at 2x108 = 1.123% change

Percent change from K = 1.9 x 108 to 2 x 108, holding r constant at 0.031 = 0.24% change

Percent change from K = 2.0 x 108 to 2.1 x 108, holding r constant at 0.031 = undetectable change

As seen from the percent changes and graphs above, R2 is far more susceptible to changes due to r than to K. A change in r at the scale of 0.001 resulted in ~1-3% changes in population, whereas increasing K by 10000000 resulted in miniscule changes to the output (undetectable to four digits/less than one percent).

Leach re-evaluates the use of the logistic curve for human population projection by assessing the fit of data from 1790 to successive census dates, starting with 1870 and ending with 1970.

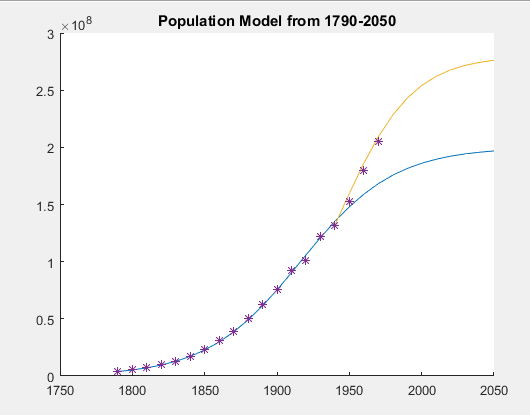
Task 7: Use your logistic model to recreate Fig. 3 from Leach (1981), assuming that one logistic model governs population growth from 1790 to 1940 and another logistic model governs growth from 1940 to 2050. Note: that *c* = *r*, *Ns* = *K*, and *No* = the initial population level; the different logistic models differ by their parameter values r and K; and the data only exist from 1790 to 1970..

Hint: 1. Simulate from 1790 - 1940 using one parameter set, and from 1940 to 2050 using a second parameter set).

2. For 1790-1910 use humpop, and for 1920 - 1970 census data use [101 122, 132, 153, 180, 205] x 106

For this problem you do not need to calculate the sum-of-squares, just find a reasonable fit.

What happened in the 1940’s?



hold on;

No=3929214;

r= .031;

K=200000000;

sim('logistic2',[1790 2050]);

plot(tout,N);

plot(merged(:,1),merged(:,2), '\*');

title("Population Model from 1790-2050");

grid;

No=132000000;

r= .040;

K=280000000;

sim('logistic2',[1940 2050]);

plot(tout,N);

plot(merged(:,1),merged(:,2), '\*');

title("Population Model from 1790-2050");

grid;

hold off;

We notice a resurgence in population around the 1940s. This is likely a representation of the baby boom generation that started growing rapidly after World War II ended in 1945.

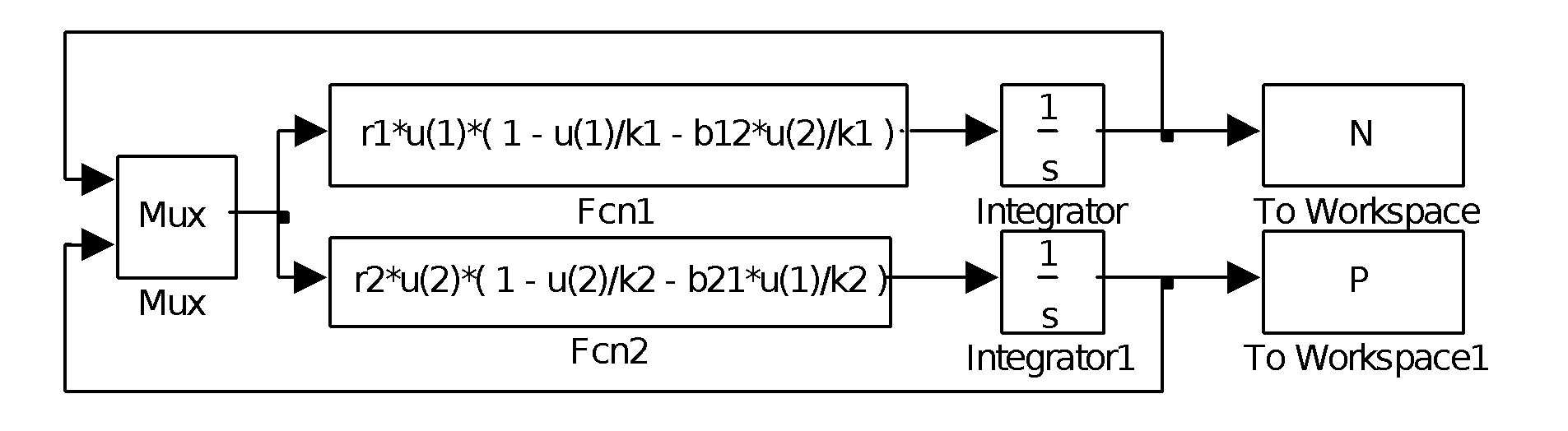
**The following section is optional**

## ***3. Mixed populations: Intraspecific and Interspecific competition***

Assuming that the growth of independent populations could be expressed by a logistic function, Gausse in 1932, developed a modeling framework to explain the population dynamics of a mixed population of two species competing for shared resources. The equations of the model were as follows

Note that if *aN* and *aP* are zero the equations simplify to two independent logistic equations.

Extend your simulink model of one population to two interacting populations (or download intercomp.mdl). The following is a sketch of the model diagram.



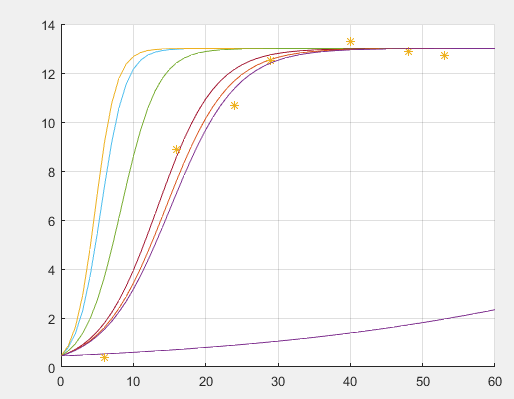
You will notice the addition of a block that we have not used yet, the MUX block. This block combines several input lines into a single vector. In this case, the inputs N and P are combined and are accessed in the function block using u(1) and u(2) respectively.

Task 8: Analyze Gause's experiments on yeast populations.

1. For the two yeast species *Saccharomyes* and *Schizosaccharomyces*, estimate the growth rates and carrying capacities when cultured in a single species medium, assuming an initial condition of 0.45. The data are in files sacc\_g.txt, schi\_g.txt. Again, to calculate the sum-of-squares use the following interpolation function

N1 = interp1(tout,N,sacc\_g(:,1))

Then as before ss = (N1 - sacc\_g(:,2))\* (N1 - sacc\_g(:,2))'



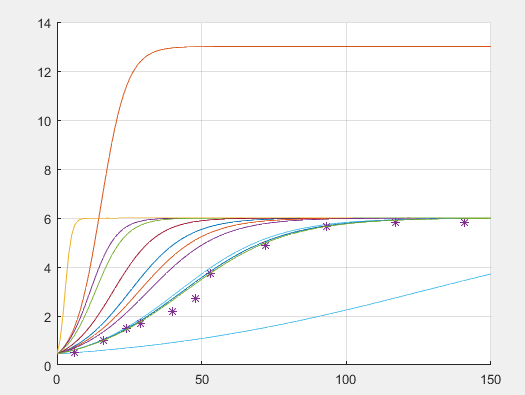
Saccharomyes

Growth rate ~ 0.24

Carrying capacity ~ 13

R2 = .9666

Sum of Squares = 4.226



Schizosaccharomyces:

Growth rate ~ .057

Carry capacity~ 6

R2 = .9845

Sum of squares = 0.6398

Using population data from experiments of two-species interaction, estimate the competition coefficiencts, *aN* and *aP*. The data are in the file mixed\_g.txt. Data in the file mixed\_g.txt are from an experiment in which the two yeast species were cultured in the same medium(the second column represents the *Saccharomye*s population and the third column represents the *Schizosaccharomyces* population).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time | Sacc Alone | Sacc Mixed | Schi Alone | Schi Mixed |
| 6 | 0.37 | .375 | .5 | .29 |
| 16 | 8.87 | 3.99 | 1 | .98 |
| 24 | 10.66 | 4.69 | 1.5 | 1.47 |
| 29 | 12.5 | 6.15 | 1.7 | 1.46 |
| 40 | 13.27 | 6.35 | 2.2 | 1.58 |
| 48 | 12.87 | 7.27 | 2.73 | 1.71 |
| 53 | 12.7 | 8.3 | 3.75 | 1.84 |

Used excel to find the mean of Σ(mixed-alone) for each species.

*aN: 4.87*

*aP: 0.58*

What are the basic assumptions that go into extending the one species logistic growth model to Gause's two species competition model?

-Assume growth is not only a function of intraspecific competition, but also a function of interspecific competition

-Assume the presence of each population leads to a depression of the other’s growth rate

-Assume that if either of the coefficients of competition =0 or one of the species goes extinct, the other species exhibit logistic growth

* In Gause’s Yeast experiments, both species coexist. This is not always the case. Often one species will drive the other species extinct. In his competition experiments with four ciliate protozoa (three *Paramecium* species and the species *Blepharisma* sp.) Vandemeer estimated the following single species population parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| **Species** | **r** | **K** | **No** |
| *P. aurelia* | 1.05 | 671 | 2.5 |
| *P. caudatum* | 1.07 | 366 | 5.0 |
| *P. bursaria* | 0.47 | 230 | 5.0 |
| *Blepharisma* sp. | 0.91 | 194 | 3.0 |

and the following competition coefficients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *j*th species | | | |
| *i*th species | PA | PC | PB | BL |
| PA | 1.00 | 1.75 | -2.00 | -0.65 |
| PC | 0.30 | 1.00 | 0.50 | 0.60 |
| PB | 0.50 | 0.85 | 1.00 | 0.50 |
| BL | 0.25 | 0.60 | -0.50 | 1.00 |

Task 9: Use your two-species model to predict the outcome of each pairwise combination; i.e., predict whether species one will persist and species two will not, species two will persist and species one will not, or both species will persist.

|  |  |  |
| --- | --- | --- |
|  | Sacc | Schi |
| Sacc | 1.00 | 4.87 |
| Schi | 0.58 | 1.00 |

Because 4.87 > 0.58, we can conclude that species one will persist and species two will not.

