Bifurcation Diagram for the Logistic Map

My Solutions ›

Compute the bifurcation diagram for the logistic map. The logistic map is given by

$$x_{n+1} = \mu x_n(1 - x_n),$$

and the bifurcation diagram illustrates the behavior of the iterates of the map as a function of the parameter Here is a resonable outline:

Loop 1 Start at $\mu = 2.4$ and finish at $\mu = 4$.

Set $x = x_0$

Loop 2 Iterate logistic map a fixed number of times (transient)

Compute x

Loop 2 (end)

Loop 3 Iterate logistic map a fixed number of times (data)

Compute and save x

Loop 3 (end)

Loop 1 (end)

Warning #1: The array x_data will be graded. To pass the assessment, the indices and values in your x_da means you are required to start your map iteration at x=x_0 and then iterate the map exactly n_trans times. another n_data times. The values of x you obtain from these second n_data iterations are to be stored in x match mine, the grader will not pass you.

Warning #2: You will need to write your iteration in the form x=mu*x*(1-x). Roundoff errors and the butterfly systems will prevent you from passing the assessment if you write the equation in a different form, such as x results!

Script

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```
mu_min=2.4; mu_max=4; %range of mu values
n_mu=500; %number of mu pixels
n_x=400; %number of x pixels
mu_edges=linspace(mu_min,mu_max,n_mu+1); %edges of mu pixels
mu=(mu_edges(1:n_mu)+mu_edges(2:n_mu+1))/2; %values of mu on which to perform compu
x_edges=linspace(0,1,n_x+1); %edges of x pixels

n_trans=20000; %transient iterations
n_data=10000;  %number of x values per mu value

x_data=zeros(n_data,n_mu); %x-data used to construct figure

x_0=0.5; %initial condition

% WRITE THE COMPUTATIONAL ENGINE OF THE CODE.
% USE THE ALREADY DEFINED PARAMETERS AND VARIABLES: n_mu, mu, x_0, n_trans, n_data.
% YOUR FINAL RESULT WILL BE THE VARIABLE x_data, and this variable will be assessed
for i = 1:size(mu,2)
    x = x_0;
    for j = 1:n_trans
        x = mu(i)*x*(1-x);
    end
    for j = 1:n_data
        x = mu(i)*x*(1-x);
        x_data(j,i) = x;
    end
end
```

▶ Run Script

Assessment: All Tests Passed

Submit

✔ **Test x_data variable**

Output