PHYS/MATH/CPSC 220: Fall 2018 Midterm Project

Professor Justin Dressel

Due: October 26th, 2018, midnight. (2+ Week Duration)

In this midterm, you will make use of the computational tools that you have developed so far to perform computational research about an interesting mathematical sequence. Consider the sequence $(x_0, x_1, \ldots, x_N)$ that is defined by a real parameter $r \in [2.5, 4]$ and an initial real value $x_0 \in [0, 1]$ using the following recursive update map:

$$x_{n+1} = r\, x_n (1 - x_n), \quad n = 0, 1, 2, \ldots, N-1 \tag{1}$$

This sequence was historically defined to model populations of physical organisms that exhibit both growth and decay. It will be your task to explore the behavior of this sequence for different values of r and different initial conditions. You will find interesting asymptotic sequence behavior for large values of N. Use your computational skills to show your results and describe your findings.

Problem 0. [Code Structure (25pt)] Edit the `README.md` file in the midterm git repository to sign your name (5pt) and link it to Travis.ci in order to automate testing with the `nose` framework (5pt). (Your code should pass when implemented properly, making the icon turn green.) Place your python code in a properly commented and formatted module `midterm.py` (5pt), including proper doc-strings, and create a supplementary Jupyter notebook `Midterm.ipynb` (5pt) that imports your module and uses it to present your results in a clean and clear way. Describe the problem clearly in your notebook, then present your complete solution as a professional report (5pt). Done properly the notebook should contain almost no code (only simple function calls from the module) and should focus on a clear discussion of your results instead of the code itself.

Problem 1. [Sequence Implementation (30pt)] In your module `midterm.py`, implement a python class `MysterySequence` that subclasses the generic `object` class (5pt). Have its constructor method `__init__(self, r, x0 = 0.5, N = 100)` take three parameters—a float r, a float x_0 that defaults to 0.5, and an integer N that defaults to 100—and store them as the instance attributes `self.r`, `self.x0`, and `self.N`, respectively (5pt). Have the constructor also create an instance attribute `self.xs` that is initialized to the undefined value `None` (5pt). Create a method `evaluate(self)` that implements the update map in Eq. (1) and uses it to populate the attribute `self.xs` with the list of floats $[x_0, x_1, \ldots, x_N]$ (of length $N+1$) corresponding to the stored values of r, x_0, and N (10pt). Create the magic method `__call__(self)` so that it executes `self.evaluate()` if `self.xs` is still undefined, then returns `self.xs` (5pt).

Problem 2. [Plotting Implementation (12pt)] In your module `midterm.py`, implement a separate python class `SequencePlotter` that subclasses the `MysterySequence` class (4pt). In this class, do not change any of the existing methods of `MysterySequence` (including the constructor). Only add one method `plot(self)` that uses `matplotlib.pyplot.plot` to create a plot of the sequence `self.xs`. To do this, make the horizontal axis of the plot a list of integers $[0, 1, 2, \ldots, N]$ representing the sequence indices and the vertical axis of the plot `self.xs`. Label the horizontal axis "Iteration k" and the vertical axis "Population x_k" (2pt). Give the plot the title "Sequence Parameters : x0={}, r={}, N={}", where the placeholders {} should be replaced with the stored values of the parameters x_0, r, and N for the instance (2pt). (Hint: look up the `.format` method for string types.) Change the vertical plot range so that it ranges from $[0, 1]$. Keep the points connected by lines, but also plot marker dots (using `marker='.'`) on top of the lines (2pt). Make the lines black (using `color='k'`) and the dots red (using `color='r'`) (2pt).

Problem 3. [Scatterplot Implementation (13pt)] In your module `midterm.py`, implement a separate function `scatterplot()` that uses your `MysterySequence` class to create a large scatterplot figure (using, e.g., `matplotlib.pyplot.figure(1, figsize=(12,8))`) showing the asymptotic values of the sequence $(\ldots, x_N)$ vs. r, starting with initial condition $x_0 = 0.5$. That is, for each value r in the range $[2.9, 4]$ (with a small spacing of 0.001 between points), create a new sequence $(x_0, x_1, \ldots, x_{300})$ with $N = 300$, then discard the first 151 points to leave only the asymptotic tail $(x_{151}, x_{152}, \ldots, x_{300})$ consisting of 150 points (5pt). Using `matplotlib.pyplot.scatter` plot all these points as the vertical axis of a scatter plot, paired with the matched horizontal axis of 150 repeated r-points $(r, r, \ldots, r)$ (2pt). Make each point a dot with `marker='.'` and black with `color='k'`, and do not connect the points with line segments (2pt). Choose the visible horizontal domain for r to show $r \in [2.9, 4]$ and the visible vertical range for x to show $x \in [0, 1]$ (2pt). Done properly, after plotting the figure will show the asymptotic sequence values $\{x_n\}$ as a multi-valued function of r (2pt).

Problem 4. [Notebook Sequence Investigation (10pt)] In your notebook `Midterm.ipynb`, use your class `SequencePlotter` to plot the sequence for each of the specific r values: 2.5, 3.2, and 3.5. (3pt) For each value of r show two initial conditions: $x_0 = 0.5$ and $x_0 = 0.1$. Comment on the behavior for the distinct initial conditions. Similarly, create plots for the specific r values 3.5441, 3.5699, and 3.57, also with two initial conditions each, $x_0 = 0.5$ and $x_0 = 0.1$, and comment on the behavior in each case. (3pt) For each case, describe the qualitative behavior of the sequence for each value of r and support your conclusions by analyzing the quantitative behavior of the last 10 values of the sequences as well (i.e., by comparing the exact values contained in `self.xs[-10:]`) (2pt). What do you conclude about the behavior of the sequence as r increases from these observed cases? (2pt)

Problem 5. [Notebook Scatterplot and Conclusions (10pt)] Plot the scatterplot with `scatterplot()` to more clearly show the asymptotic behavior of the sequences as r increases (2pt). What do you conclude from this plot (2pt)? Discuss any surprising features that you observe in detail (2pt). Use quantitative analyses of specific sequence values to support your findings (2pt). Would you have guessed this behavior from the form of Eq. (1)? Speculate how you might approach further analyzing the behavior of this sequence if you were to go beyond this midterm and try to publish an academic paper about Eq. (1) (2pt).