Problem Set 4: Simulating the Spread of Disease and Bacteria Population

Handed out: Monday November 13th, 2017
Due: 11:59pm on Wednesday, November 22nd, 2017
Checkoff Due: 11:59pm on Monday, Dec 4th, 2017

Background: Bacteria, Antibiotics, and Computational Models

Bacteria are single-celled organisms that reproduce asexually. Some bacteria cause diseases, some are harmless, and some are beneficial. Bad bacteria can cause infections such as strep throat and tuberculosis. Bacterial infections are treated with antibiotics targeted to kill only the bad bacterial cells.

Bacteria that cause infections can resist and develop a resilience to antibiotics in two ways: (1) naturally and (2) via inappropriate use of antibiotics. Thus, populations of bacteria can undergo substantial evolutionary changes within a single patient over the course of treatment.

In this problem set, we will make use of simulations to explore the effect of introducing antibiotics to the bacteria population and determine how best to address these treatment challenges within a simplified model. We will implement a simplified stochastic model of bacteria population dynamics within a person.

10% - Problem 1: Implementing a Simple Simulation (No Antibiotic Treatment)

We start by modeling the growth of the bacterial population inside a patient not taking antibiotics.

Implement the `SimpleBacteria` and `Patient` classes, according to the behavior described in the docstrings in the provided code.

`SimpleBacteria` class

Maintains the state of a single bacteria. You will implement the following methods according to the docstring spec:

+ `__init__`
+ `is_killed`
+ `reproduce`

Hint: Use `random.random()` for generating random numbers between 0 and 1 to ensure that your results are consistent with ours.

`Patient` class

Maintains the state of a bacterial population associated with a patient. You will implement the following methods according to the docstring spec:

- `__init__`
- `get_total_pop`
- `update`

10% - Problem 2: Running and Analyzing a Simple Simulation (No Antibiotic Treatment)

In this part you will understand the behavior of a group of bacteria cells as time passes before introducing any antibiotic through a simulation.

Implement the function, `simulation_without_antibiotic` according to the docstring.

Feel free to use the provided debugging code to test your simulation. Make sure to comment it out when you are finished. Note: Your code may occasionally fail some test cases a small fraction of the time. It is expected from stochastic simulation.

20% - Problem 3: Calculating and Plotting a Confidence Interval

Using data collected in the simulation from Problem 2 as our sample, we want to construct and plot the 95% confidence interval of an estimate of the average bacteria population at each certain time step (zero-indexed).

Implement the functions `calc_pop_avg, calc_pop_std, calc_95_ci,` and `plot_simulation_without_antibiotic` according to the behavior described in the docstrings in the provided code. You can test your code using the tests we provided in ps4_tests.py

Given data $x_i \; for \; i = 1, 2, \ldots, n$, some formulas you might find useful are:

mean $$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

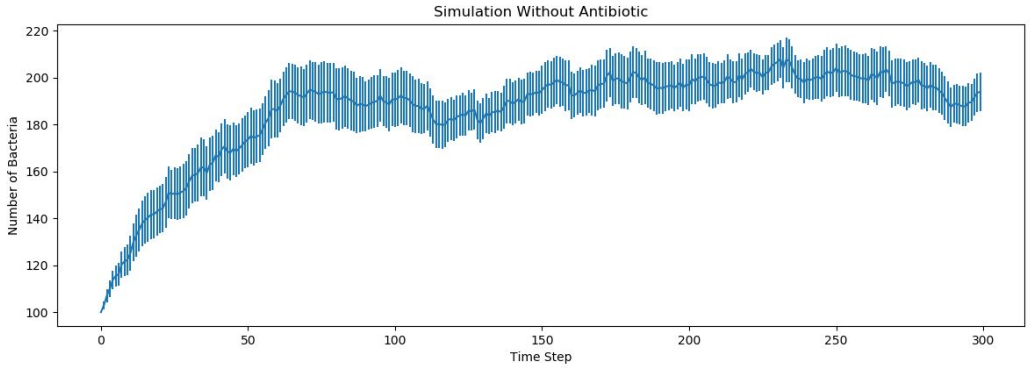
standard deviation $\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$

standard error of the mean $SEM = \frac{\sigma}{\sqrt{n}}$

95% confidence interval $\bar{x} \pm 1.96(SEM)$

Hint: Uncomment the code at the bottom of `ps4.py` to test your plotting function
Be prepared to demonstrate and discuss your plotting function during the checkoff.

You should aim for your graph to look similar to the following, It may take a bit of time to generate depending on your computer:



10% - Problem 4: Implementing a Simulation with an Antibiotic

In this problem, we consider the effects of both administering an antibiotic to the patient and the ability of bacteria cell to inherit or mutate genetic traits that provide antibiotic resistance. As the bacteria population reproduces, mutations will occur in the bacteria offspring. Some bacteria cells gain favorable mutations like antibiotic resistance. Note that bacteria in lower population densities mutate at a faster rate.

`ResistantBacteria` class

Implement the `ResistantBacteria` class, a subclass of SimpleBacteria, according to the docstrings The methods you need to implement are:

+ `__init__`
+ `get_resistant`
+ `is_killed`
+ `reproduce`

`TreatedPatient` class

Implement the `TreatedPatient` class, a subclass of `Patient`, according to the docstrings. The methods you need to implement are:

+ `__init__`
+ `set_on_antibiotic`
+ `get_resistant_pop`
+ `update`

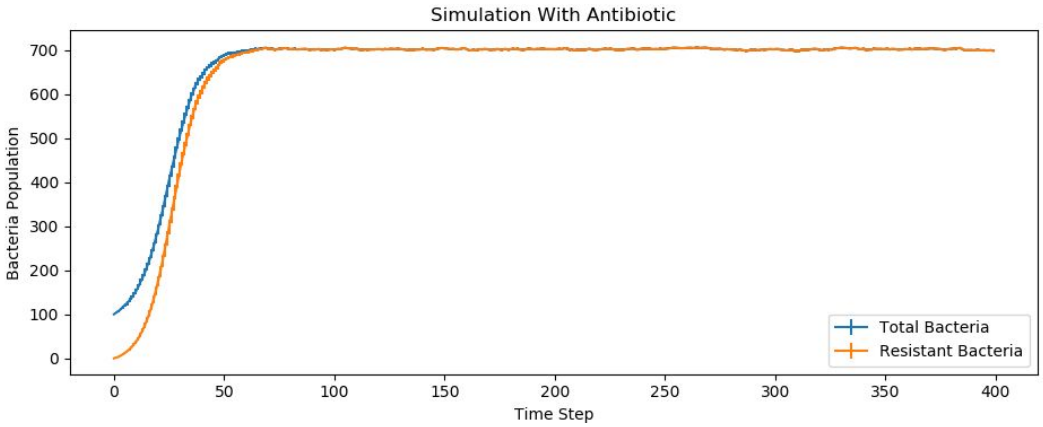
20% - Problem 5: Running and Analyzing a Simulation with an Antibiotic

In this problem, we will use the implementation you filled in for Problem 4 to run a simulation. The simulation explores the effects of resistant bacteria and antibiotic treatments.

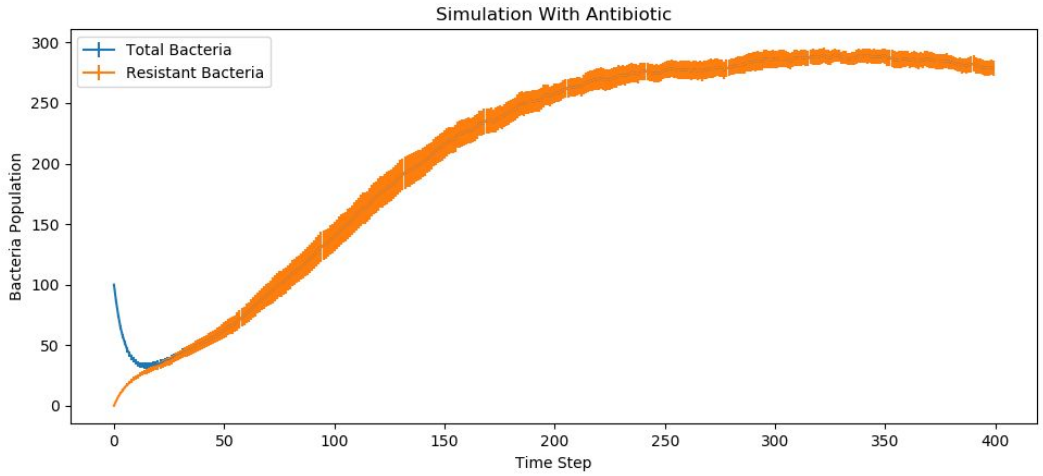
Implement the function `simulation_with_antibiotic` according to the behavior described in the docstrings in the provided code. Be prepared to demonstrate and discuss your plotting function during checkoff.

Using the provided examples, your graphs should look something like the ones below, where the two curves eventually converge (the actual drop off might look less drastic in your implementation):

EX1:



EX2:



30% - Checkoff

Be prepared to discuss your code and plots during checkoff.

Hand-In Procedure

1. Save

Save your solutions as **`ps4.py`**.

2. Time and Collaboration Info

At the start of each file, in a comment, write down the number of hours (roughly) you spent on the problems in that part, and the names of the people you collaborated with. For example:

```
# Problem Set 4
# Name:
# Collaborators (Discussion):
# Time:
# Difficult Sections
#
 ... your code goes here ...
```

3. Sanity checks

- After you are done with the problem set, do sanity checks. Run the code and make sure it can be run without errors. You should never submit code that immediately generates an error when run!

4. Submit

Upload all your files to the 6.00 staff site before the deadline