EE194/BIO196: Modeling Biological Systems
HW 6: Bacterial chemotaxis

Overview

We've discussed in class how we can model bacterial chemotaxis at a very high level. We've also looked at a simple Python program to implement the model and to animate the results. The code for that program is available for you to look at and use.

In this homework, we will take that code a bit further and turn it into an object-oriented version. You will write an EColi class in Python. Essentially, you will be taking the code from class and writing the infrastructure for classes around it.

Details

You can start with two files. Chemo_highlevel.py is the non-object-oriented version that is given to you. Chemo_highlevel_classes_swarm.py is the object-oriented version. Some of it is already written, but much of it is not. You will fill in the missing portions by taking code snippets from chemo_highlevel.py, modifying them as needed, and placing them into chemo_highlevel_classes.py.

Of course, the homework is not just a simple matter of cutting and pasting. You will that you must also modify some of the code (e.g., to refer to instance variables with the "*self.*" notation), and you must reorganize some of the code to make it fit an object-oriented framework.

In a very object-oriented view of the world, only EColi member functions are allowed to access its instance variables. Python does not enforce this guideline; however, for this homework you should follow it. One consequence of this is that essentially *every* piece of code that accesses an instance variable will now be an EColi member function.

As part of this homework, you'll see the use of the function *random.uniform*(-1,1). This is another random-number function, much like *random.randrange*() and *random.gauss*(). It creates random numbers; in this case, between -1 and 1. Unlike randrange(), it returns floating-point numbers, and so may return -.9, .75, etc.

The other part of the homework is a bit of animated razzle-dazzle. Now that we've built an EColi class, it is easy to instantiate as many bacteria as we want. You will note that the main *chemotaxis*() function instantiates an entire swarm of bacteria. As part of your *Ecoli.__init__*() function, all of them should enter together at the lower left as before; however, afterwards each should move independently in its own random direction.

Most of the functions in chemo_highlevel.py were covered in class. The animation-related code was not. You will not need to touch this code. However, we've tried to comment the code well enough that, if you like, you can reverse engineer it as much as needed without an unreasonable effort.

Discussion:

- No discussion questions need to be turned in for this assignment.

Potential final project

You could extend this assignment into a final project if you like. The first part is to learn more about biology. We've discussed a very high-level model for bacterial chemotaxis. Bacterial chemotaxis is one of the most well-studied systems in bacteria, and is in fact, quite well understood at this point in time (at least in E.Coli). Your first task is to learn more about how it actually works, and report on it (either orally or in writing). A summary can be found at https://docs.google.com/document/d/11hg93nO_2HGFGEwJ6zVT66t0Vrm5dOvoGf310xVo3W4/edit?usp=sharing . The summary has pointers to numerous other papers and reviews. Much of the material in it comes from Chapter 7 of Uri Alon's *An Introduction to Systems Biology: Design Principles of Biological Circuits* (on reserve at Tisch).

With that done, your next task is to use our chemical-reactions framework to instantiate the actual chemotaxis reactions (or a reasonable approximation) and show one E.coli bacteria moving towards sugar.

Logistics:

- Use any lab PC system to write your code. You may use your own laptop if you prefer.
- The due date for this assignment is on the class calendar
- Submit your project at https://www.ece.tufts.edu/ee/194MSO/provide.cgi, which is also accessible from the course web page. You should turn in chemo_highlevel_classes.py. If desired, also turn in a file for one of the extra-credit problems.