

PHY307, Science and Computers I

Lab #4, September 12, 2002 (due by end of lab period)

Cultivating Chaos

Summary of what you will do:

The goal is to for you to remind yourself about how we visualized dynamics in the previous lab, to explore lists and graphing in Python, and to apply this to chaos. Again, your report for this lab will not be on this handout, but in a document that you create.

I will start with a brief motivation for the logistic map – you will want to take notes on that. Remember, chaos is not in itself random, but it can *amplify* randomness or small errors. It leads to complex behavior.

MORE MAPS

These first few steps should be familiar:

1. Log in to your workstation using your CMS/SUnix name & password.
2. Open up some editor or word processor to start your report. Put in a title, your name, and date. As you write the report, make it clear what each section is. At the end of the lab, print your report out, staple it together, and hand it in.
3. Get IDLE for VPython going by going through the Menu: Start ? SU Academic Departments ? Physics.
4. Open a program file editor by “New” under the “File” menu and save your (currently empty) work to your I: or K: drive as “map1.py” or a similar name.
5. Enter the following program (or download from the web page, under codes):

```
from visual import *
scene.autoscale = 0

# create two cones (assigned to conea and coneb) that point
# "up" on the screen (positive y - by setting axis)
# and have narrow bases, slightly separated in position, pointed
# oppositely (base to base)
conea = cone(radius=0.1, color=color.green, x=0.5, axis=(0,1,0))
coneb = cone(radius=0.1, color=color.red, x=0.51, axis=(0,-1,0))

# repeatedly apply reflection map, 3 times a second, until program
# stopped.
while 1:
    rate(3)
    conea.x = -conea.x
    coneb.x = -coneb.x
```

6. Run this program – note that the cones start together and stay together. There is no “sensitivity to initial conditions” for this map and these starting points. (That is, there is no apparent butterfly effect. This can be *proven*, in fact:
 - a. Proof of no butterfly effect for the map $x \mapsto -x$. Consider the *iterated map*, which you get by doing the map twice. This takes an original x to $-x$ and then back to x . So if two points start together, with coordinates x and y , with $|x-y|$ small, then after two applications of the map, they are back to their starting points and the difference in positions is still small – the same as it originally was.
7. Such proofs are “rare”. There are classes of maps which can be shown to be non-chaotic, but the general case is tough to study mathematically. So computers have proven very important in exploring when the butterfly effect exists. This usually does *not* provide a *proof* of chaos or orderly motion, but can be very strong evidence.
8. Now let's look at some more interesting maps. Save your program to a new file, `map2.py` and modify the code: instead of simple reflection, use the map $x \mapsto a*x*(1-x)$. (This map is called the logistic map.) To do this,
 - a. Introduce a new variable `a` at the beginning of your program, and assign it a value of 2.5.
 - b. Change the map for both cones to the new map.
9. Run the program. Record your results in some detail.
 - a. Do the cones separate, If so, do they approach each other later?
 - b. Do you believe that you see a butterfly effect?
 - c. Make the initial separation much smaller. How does this affect what you see?
 - d. Include your codes (*both* with the initial separation of 0.01 and the much smaller separation.)
 - e. Summarize your results.
1. Repeat steps 9a-9e for $a=2.5$, $a=3.5$ and $a=3.8$. Compare what you see for these different values of the parameter **a**.