CSCI 4446 Spring 2019 PS9
Derek Wright

Note: I worked with Keegan McNamara on the assignment.

1.

I used r = 13.3 as my nonchaotic trajectory.

2.

Running our algorithms we get the following data tables for the largest Lyapunov exponents:

Chaotic: r = 45

ε	λ_1
30	1.7304222971761412
20	2.1919552458651994
10	1.991450798062021
5	1.961384376409598
2	1.9817019387035075
1	1.052244839284417

Looking at these results, I am mostly happy with how they came out. I know that the largest Lyapunov exponent for a chaotic trajectory must be positive, and I see that in the results. But I also see wide variability that is likely an artifact of the combination of my implementation of Wolf's algorithm as well as the many parameters passed in. Because of the range of consistency around 1.96 - 2.00, I would lean towards these values, i.e. $\lambda_1 = 1.98$. But of course I do not have a proof, as much of this analysis is by brail.

The non-chaotic trajectory is a bit tougher. The non-chaotic trajectory we chose was a simple one that converges to a fixed point from far away. This trajectory thus was not a good candidate for Wolf's algorithm in that the next set of points (x, z) will be **closer** together as they converge to the same point, instead of farther away until they reach the epsilon threshold we set. Still using Wolf's algorithm, we let the algorithm go to the end of the trajectory and give us a final L'/L ratio, which is less than 1, since we never grew outside the threshold range. This results in a negative log value, and thus a negative Lyapunov exponent. Since Wolf algorithm was still used, and the representation of the growth (or, in this case, contraction) is still valid, I use the following results as valid.

Non-Chaotic: r = 13.3

ε	λ_1
30	-0.895286232148458
20	-0.895286232148458
10	-0.872337320382135
5	-2.945380929829208
2	--*
1	--*

* Insufficient conditions for meaningful results

I know a non-chaotic trajectory should have no positive Lyapunov exponents, but zero and negative values are permissible. The former would indicate a limit cycle (points stay at the same distance), while a negative value would indicate an attracting fixed point. I know from Problem Set 5 that for r = 13.3, the trajectory indeed converges to a fixed point, so the results are consistent with that picture, though variability still plagues my implementation. **So even though Wolf's algorithm is only supposed to produce positive Lyapunov exponents because of the assumption that the nearest neighbor will spread out, if we generalize it to allow for contraction as well (which every fixed point attractor will do), there is no reason mathematically we can't have a negative λ.**

I noticed as epsilon got smaller, my algorithm had a harder time working smoothly, causing a lot more sampling and looking around for useable data. Therefore I would lean towards the first 3 values to be more useable for our purposes. I.e. λ_1 = -0.88

4a.

Using PS7, setting the runtime to t = 10 (i.e. 10,000 points), an eigenvalue calculator, and using the equation given in Problem 4 of the assignment, I get the following results:

I *should* get the same λ_1 value as in Problem 1 if these methods were both perfect, but alas they are not.

r	λ_1	λ_2	λ_3
45	1.075293719847029	-2.860867360553759	0.049753319651316
13.3	-0.405291968031407	-3.986249293827802	-1.010289034613098

These values (or, at least, the signs of these values) are in line with what I would expect from this process. Looking at the chaotic trajectory, we see one positive value, one negative,

and one fairly close to zero. These correspond to the actors creating chaos, bounding the attractor, and maintaining distance (sub-exponential growth) along the trajectory. Moreover, the largest Lyapunov exponent is of the same order as the one I calculated using Wolf's Algorithm, which is as good as could've hoped for using these disparate strategies. I am perfectly content focusing on order of magnitude and sign rather than paying too-close attention to the precise values.

Using PS7's variational equation method is somewhat suspect, since it is a linear approximation at its core, and we are running $t \to \infty$ (really only 10), which is plenty far enough for nonlinearities to show themselves and create nontrivial error in our results. Because of this, I would trust Wolf's algorithm more than the variational equation, and the Kantz algorithm more than either of the two used in this problem set.