Santa Fe Institute
2011 Complex Systems Summer School

Introduction to Nonlinear Dynamics
Lab 2

1 The Lorenz System

Use TISEAN's `lorenz` command to generate a 50,000-point trajectory of the Lorenz system with parameter values $R = 15$, $S = 16$, and $B = 4$:

```
lorenz -l50000 -R15 -S16 -B4 -x0 -o lorenzR15.dat
```

There are different naming conventions for these parameters; many people use a or σ instead of S. The `-o` option directs TISEAN to place its output in the filename that follows it. If you just type `-o` without a filename, TISEAN puts the output in `lorenz.dat`. The `-x0` option tells TISEAN not to discard any points from the beginning of the trajectory; we've chosen that here because we want you to see the transient.

Repeat this with $R = 45$ and plot both trajectories. If you're using gnuplot, try:

```
plot "lorenzR15.dat" using 1:2
```

which plots an $x - y$ projection of this 3D trajectory. To plot other projections, try `using 1:3` and `using 2:3` instead. You can plot in 3D with:

```
splot "lorenzR15.dat" with lines.
```

> **Note**: if the `lorenz` command does not work on your computer, you should use one of the lab computers to do this problem—or just grab the data files from the CSSS wiki.

2 Power Spectra

Use TISEAN's `spectrum` command to compute the power spectra of the trajectories from problem 1:

```
spectrum lorenzR15.dat -c# -o lorenzR15.spectrum
```

```
spectrum lorenzR45.dat -c# -o lorenzR45.spectrum
```

The `-c#` option tells TISEAN to read the whole file, not just the first column (which it will do by default).

Make a semilog plot of these power spectra. If you're using gnuplot, you can do this by typing `set logscale y` before typing the `plot` command.

Compare and contrast the two spectra; relate your observations back to what you know about the two state-space trajectories and their properties.

3 Delay-Coordinate Embedding and Lyapunov Exponents

Using the link at the top of the following webpage:

www.mpipks-dresden.mpg.de/~tisean/TISEAN_2.1/docs/tutorial/ex4.html

you can download a time-series data set called `amplitude.dat`. In this problem, you'll use TISEAN's `lyap_k` tool to embed this trajectory and compute its maximal Lyapunov exponent λ_1.

Begin by downloading this data. Next, read up on `lyap_k` on the TISEAN manual[1]. You'll notice lots of discussion of caveats regarding interpretation, data, and parameter values. *This is the main challenge in using nonlinear time-series analysis tools:* these algorithms are designed to work on a limited amount of noisy, finite-precision data. Doing so involves approximations to the full formal mathematics, each with an associated set of parameters: iteration limits, εs that specify scales, and something called the Theiler window that helps the algorithms avoid doubling back on themselves in bad ways, among other things. The Kantz & Schreiber book describes all of this in a lot more detail, and you should spend some serious time with it if you plan to use these tools in your work.

The first step in the analysis of scalar time-series data from a nonlinear dynamic(al) system is to embed it, and the first step in that process is to estimate the delay τ. One way to do this is to use TISEAN's `mutual` command. The second step is to estimate the embedding dimension m, which can be accomplished, for instance, with TISEAN's `false_nearest` command. After embedding the data with correct values for these parameters, one can compute the Lyapunov exponents, fractal dimensions,

[1] This is under "General Manual" on their website—www.mpipks-dresden.mpg.de/∼tisean—then "Lyapunov Exponents."

etc. for the system. Choosing correct parameter values for the TISEAN tools that perform these calculations, as mentioned above, can be a challenge; when you use any of them, you should **at the very least** explore values other than the defaults and see if your results change.

You'll duck all of those issues in this problem, though, and run `lyap_k` with parameter values that have been established to work well for this data set: $3 \leq m \leq 6$, a delay of 8, a Theiler window of 100, a minimum epsilon of 0.1, and 500 iterations:

```
lyap_k amplitude.dat -d8 -m3 -M6 -t100 -r.1 -s500 -o
```

Some versions of TISEAN have a slightly different syntax for specifying the minimum and maximum embedding dimensions:

```
lyap_k amplitude.dat -d8 -M3,6 -t100 -r.1 -s500  -o
```

The `-o` option tells TISEAN to put the output in a file with the same name as the data file, but with the suffix "lyap." You can re-route the output to another file if you want; just specify its name after the `-o`.

Plot the results. If you're using gnuplot, try

```
plot "amplitude.dat.lyap" with linespoints
```

Does your curve have a "scaling region"—a region where there's a clear diagonal line? The slope of this line, if it exists, is an estimate of the maximal Lyapunov exponent λ_1.

4 Fractal Dimension

The correlation dimension d_{corr} is one member of the (large) family of fractal dimensions, many of which are covered in Chapter 6 of Kantz & Schreiber.

Correlation dimension is defined in terms of the correlation integral, which can be approximated by the correlation sum. Informally, the correlation sum counts the number of pairs $(\vec{x}(i), \vec{x}(j))$ in a given set of vectors that are at most ε apart. Formally, the correlation sum is defined as:

$$C(\varepsilon) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \Theta(\varepsilon - ||\vec{x}(i) - \vec{x}(j)||), \quad \vec{x}(i) \in \mathbb{R}^m$$

where m is the embedding dimension, N is the number of points in the trajectory and $\Theta(x)$ is the Heaviside step function:

$$\Theta(x) = \begin{cases} 1 & : x > 0 \\ 0 & : x \leq 0 \end{cases}$$

As $N \to \infty$ we expect the correlation sum $C(\varepsilon)$ to scale like a power law, $C(\varepsilon) \propto \varepsilon^D$, where D is the correlation dimension defined by

$$d(N, \varepsilon) = \frac{\partial \ln C(\varepsilon, N)}{\partial \ln \varepsilon} \qquad\qquad D = \lim_{\varepsilon \to 0} \lim_{N \to \infty} d(N, \varepsilon)$$

That $N \to \infty$ requires an infinite amount of data, however, which is obviously not the case in practice. Therein lies the challenge of building and using algorithms like TISEAN's `d2`, which calculates the correlation sum of a trajectory.

Run the `d2` command on the amplitude dataset, embedded with $\tau = 8$ and a Theiler window of 100:

```
d2 amplitude.dat -d8 -t100 -o
```

This command will generate three files with the same name as the data set and different suffixes. We are interested in the ".c2" file. The first column in this file is ε and the second is an estimate of the correlation sum $C(\varepsilon)$ for that particular ε. The file will have several sections; these are the estimates for different embedding dimensions. The default range is $m = [1 - 10]$; you can specify otherwise with the `-m` and `-M` options to `d2` if you wish.

Make a log-log plot of ε versus $C(\varepsilon)$ and look for the scaling region in the plot. The slope of that region, if it exists, is an estimate of the correlation dimension of the data set.

Homework

- Change the TISEAN parameters (e.g. `d, m, M, t,` and `r`) for problems 3 and 4. Do your results change? A lot?
- Add some noise to your Lorenz data from problem 1 using TISEAN's `makenoise` command. Plot the trajectory and its power spectrum; compare the results to those from the noise-free data.
- Fit lines to the scaling regions of the plots in problems 3 and 4 and use them to compute numerical estimates of λ_1 and d_{corr}. You can do this with a straight-edge or with a regression package (e.g., Excel's `trendline`), as you wish.
- (for experts) Use `mutual` (or `corr`) and `false_nearest` to determine appropriate embedding parameters for the `amplitude.dat` data set.
- Use `delay` to embed the `amplitude.dat` data—either with the τ and m values prescribed in problem 3, or with the values that you determined yourself. Plot the results. Do you recognize this attractor?