Dynamical Systems Theory in Machine Learning & Data Science

lecturers: Daniel Durstewitz
tutors: Christoph Hemmer, Alena Brändle, **Lukas Eisenmann**, Florian Hess
WS2024/25

Exercise 7

To be uploaded before the exercise group on December 11, 2024

1 Lorenz Map

Consider the Lorenz system given by the following system of ODEs:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = x(\rho - z) - y$$
$$\dot{z} = xy - \beta z$$

Consider its chaotic regime by choosing $\sigma = 10, \beta = \frac{8}{3}$ and $\rho = 28$.

1. Choose a reasonable initial condition and draw a trajectory of the system. Visualize 3D state space. The trajectory should be long enough such that you can identify the butterfly-shaped chaotic attractor.

2. Edward Lorenz used consecutive maxima of the z coordinate to quantify the chaotic nature of the system. Plot the time trace of the z coordinate of the trajectory drawn before. Find all consecutive maxima of the z coordinate, $\{z_n\}$, and plot the first-order return plot of the resulting series of peaks. Overlay the bisectrix. Confirm for yourself by inspection of the return plot, that $\forall n : \ |F'(z_n)| > 1$.
Hint: To find the peaks of the signal, you can use Peaks.jl in Julia or scipy.signal.find_peaks in Python.

3. Now, given the Lorenz map $z_{n+1} = F(z_n)$ and assuming $|F'(z_n)| > 1 \ \forall n$, show that *all* closed orbits of F are unstable.
Hint: Consider the fate of a small perturbation ϵ_n at step n under repeated application of F, and recall, that for a k-cycle we need to have $z_{n+k} = z_n$.

2 Strange Attractors, Fractal Dimensions and Chaos

For this exercise, make sure you have a running installation of Julia. The simplest way to install Julia is by installing Juliaup. We will make ourself familiar with the phenomenal DynamicalSystems.jl library. The library exports several submodules like ChaosTool and FractalDimensions which we will use here as well. To install ther required package, check out the Github Page.* If you want to run

*If you have problems to get Julia and the library running on your machine, contact Lukas.Eisenmann@zi-mannheim.de.

Julia in VS Code you can follow this Tutorial. You can also run Julia in a JupyterNotebook.

Visit `https://www.dynamicmath.xyz/strange-attractors/` and pick any chaotic attractor under "Interactive Simulations" you feel attracted to! If you click on one of the system, you will be lead to an interactive session demonstrating the dynamics of the attractor. In this exercise, you will determine its correlation dimension and empirically estimate its maximum lyapunov exponent.

1. Define a **ContinousDynamicalSystem** that implements the system equation of your system of choice.[†] The library provides an Example for the Lorenz model. You do **not** need to implement the Jacobian!

2. Draw a trajectory from your system and plot the attractor in state space. Make sure that you implemented the system correctly by visually comparing the attractor to the one shown on the web page.

3. In the lecture, you learnt that the correlation dimension d_{corr} can be used to compute the fractal dimension of the attractor. Compute the correlation sum $C(\epsilon)$ for a suitable range of small radii ϵ and plot ϵ against $C(\epsilon)$ in log-log fashion. You should see a linear relationship.
   Hint: Use the function **correlationsum**.

4. Determine d_{corr} by fitting a line to the linear region of the log-log plot. Is the attractor you picked indeed "strange"?
   Hint: Use **linear_region** to fit the line.

5. There are strange attractors that do not display chaos (i.e. fractal dimension but no sensitive dependency on initial conditions). Determine the maximum Lyapunov exponent of the system running into the strange attractor. Does the attractor indeed display deterministic chaos?
   Hint: Use the **lyapunov** function.

[†] Do it yourself, even if the library already provides the system.