SOFTWARE FOR VISUALIZING AND ANALYZING DYNAMICAL SYSTEMS IN $\mathbb{R}^2$

Alec McGlasson, Daniel Madison, Gavin Menning, and Adam Case (advisor)
Drake University
alec.mcglasson@drake.edu, daniel.madison@drake.edu, gavin.menning@drake.edu, adam.case@drake.edu

ABSTRACT

Dynamical systems are of great interest to mathematicians and before the advent of computers, it was extremely difficult to create a visual representation of them. Now, with the aid of computer software, we can draw and analyze these systems on a virtual plane. The purpose of the software that we developed and describe in this paper is to provide researchers with a tool to perform these tasks. Our software (called DSGraph) is currently able to draw individual dynamical systems and their trajectories with user defined specifications. We are currently working on implementing functionality to compute the compressibility of individual trajectories and plot probabilistic dynamical systems. In the future we have plans to implement features that will let the user draw in spaces other than $\mathbb{R}^2$ and draw Markov chains.

INTRODUCTION

A dynamical system in $\mathbb{R}^2$ is defined by a function $F: \mathbb{R}^2 \to \mathbb{R}^2$. An *n-trajectory* of F, given an initial condition (X_0, Y_0), is a sequence $(X_0, Y_0), (X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$ such that

$$F(X_0, Y_0) = (X_1, Y_1), F(X_1, Y_1) = (X_2, Y_2), \ldots, F(X_{n-1}, Y_{n-1}) = (X_n, Y_n).$$

Mathematicians are often interested in graphing these trajectories to provide a visual representation of a dynamical system. For example, the *Hénon attractor*, discovered by Michel Hénon [4] is a dynamical system $T: \mathbb{R}^2 \to \mathbb{R}^2$ defined by

$$T(x, y) = (1 + y - 1.4x^2, 0.3x).$$

By graphing trajectories of T, given a particular initial condition, we obtain the image found in Figure 1 (generated in DSGraph).

Drawing visual representations of dynamical systems by hand is typically a very challenging task. Researchers often use computer graphing software to automate this process. This paper describes software, developed by the authors in C#, that can be used to define functions $F: \mathbb{R}^2 \to \mathbb{R}^2$ and generate their trajectories on a 2-D plane. This software, *DSGraph*, uses the library mXParser to parse the definitions of functions specified by the user. This software also makes use of ScottPlot, a library used to create 2D graphs, to visualize trajectories of a dynamical system. Current and future work will be discussed, including the development of analytical tools that will aid researchers in this field.

THE SOFTWARE

DSGraph was programmed in C# using the Microsoft Visual Studio .Net Development Framework. Visual Studio's GUI customization features helped to expedite the development of DSGraph's user interface.

The software uses mXParser [2], a math library used to parse and calculate user-defined functions in $\mathbb{R}^2$, and ScottPlot [3], a library that allows for the visualization of trajectories of dynamical systems.

ScottPlot and mXParser provide the core functionality of DSGraph. The mXparser library gives us the ability to define and calculate functions in a way that is intuitive to mathematicians and those who are not familiar with programming. ScottPlot gives the user the ability to plot multiple functions on an interactive 2D plane. The integration of these libraries significantly reduced the development time of DSGraph.

MXparser's functionality is essential to the back end of DSGraph because it can create user-defined functions by parsing a string that is entered into a text box located in the user interface. After these functions are defined, they can be calculated with any input by calling a built-in function. This library also provides the user with common trigonometric, unary, and binary functions, which adds to DSGraph's versatility.

ScottPlot's functionality is essential to the front end of DSGraph because it is the main viewing panel. In ScottPlot, the points of a plot are defined by populating two arrays, where the first array contains the x-values and the second array contains the y-values. The trajectories of a dynamical system are plotted using a PlottableScatter object, which is a basic scatter plot. ScottPlot allows the user to update a trajectory by changing the values in the arrays related to it. Another feature of ScottPlot is its user control functionality, which allows the user to drag the plot around, zoom, define a viewing window by either highlighting a region by middle-clicking and dragging or going into ScottPlot's settings window.

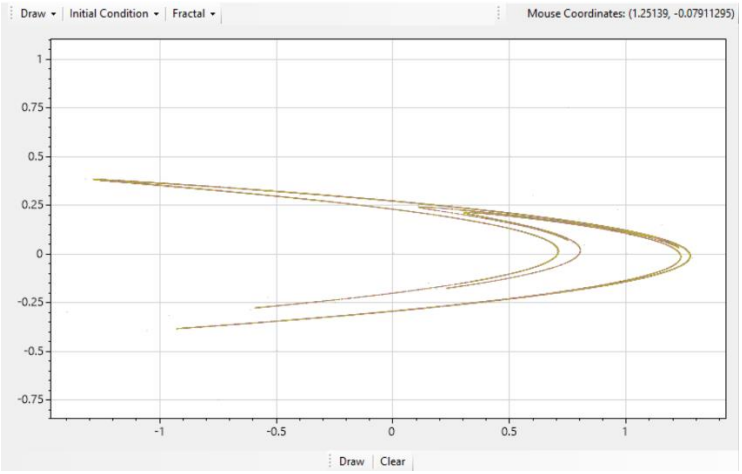

Figure 1: The Hénon attractor generated in DSGraph

CURRENT WORK

Trajectory compression. We are currently working on functionality that will let DSGraph perform compression of individual trajectories using the Noemax DotNetCompression library. DotNetCompression is a real-time compression library in C# that provides compression algorithms such as LZF4, DEFLATE/ZLIB/GZIP, and LZMA/BZIP2 [5]. DSGraph will compress trajectories by converting their respective arrays into strings with a given precision. Compressibility of trajectories can vary based on the precision of the points in the trajectory string. The precision will be specified in the user interface.

Being able to approximate the compressibility of individual trajectories may yield insights into the orbit complexities and topological entropies of dynamical systems that exist within computable metric spaces [1]. DSGraph will be able to compress the trajectory strings in real-time and display the compressibility in terms of string length, which will be viewed in the GUI of DSGraph.

Random Dynamical Systems. Another feature that is currently in progress is a method for drawing random dynamical systems. A random dynamical system is a dynamical system where there are multiple functions that can be applied at each point along the trajectory, and each of these functions has a set probability of being chosen. When drawing the dynamical system, the function used at each point is chosen at random based on the function's given probability. This makes it so two random dynamical systems with the same inputs may result in different drawings.

The plan for implementing this feature is to design a random dynamical systems option which, when selected, will allow for multiple functions to be input along with a place to input the probability for each function to be chosen. Then during the drawing process, a random number will be chosen at each point which will be used to choose which function will be utilized for the specific iteration of the dynamical system.

FUTURE WORK

DSGraph is still being developed and is currently in its alpha stage. The software currently supports a variety of tools for drawing the trajectories of dynamical systems in $\mathbb{R}^2$. However, future versions will include additional tools to aid researchers in the analysis of these systems.

Drawing in spaces other than $\mathbb{R}^2$. Currently, DSGraph only produces visual representations of dynamical systems in $\mathbb{R}^2$. Providing researchers with the ability to define and draw dynamical systems in $\mathbb{R}^3$ would be valuable to researchers who study 3D dynamical systems. In addition, allowing users to visualize dynamical systems in the complex plane would be of interest to mathematicians who study the Mandelbrot Set or Julia sets.

Markov Chain. A possible future feature to be implemented would be a process to draw Markov chains. Markov chains are similar to the currently in-progress random dynamical systems, except each possible function's probability of being chosen is dependent on the previously chosen function.

CONCLUSION

DSGraph is a tool that will be invaluable to researchers that are doing work with dynamical systems. It uses preexisting C# packages to allow researchers to draw and analyze user-defined dynamical systems. It can currently draw individual dynamical systems and trajectories and we are working on implementing functionality that will let DSGraph draw probabilistic dynamical systems and calculate the compressibility of individual trajectories, which may provide insight into orbital complexities and topological entropies. We also plan to develop features that will let the user draw in spaces other than $\mathbb{R}^2$ and draw Markov chains.

REFERENCES

[1] Galatolo, S., Hoyrup, M., Rojas, C., Effective symbolic dynamics, random points, statistical behavior, complexity and entropy, Information and Computation, 208, (1), 23-41, 2010.

[2] Gromada, M., mXparser, 2017, http://mathparser.org/, retrieved February 2020.

[3] Harden, S., ScottPlot, 2020, https://github.com/swharden/ScottPlot, retrieved February 2020.

[4] Hénon, M., A Two-dimensional Mapping with a Strange Attractor, *Communications in Mathematical Physics*, 50, (1), 69-77, 1976.

[5] Noemax Technologies, DotNetCompression, 2017, https://www.noemax.com/dotnetcompression/, retrieved March 2021.