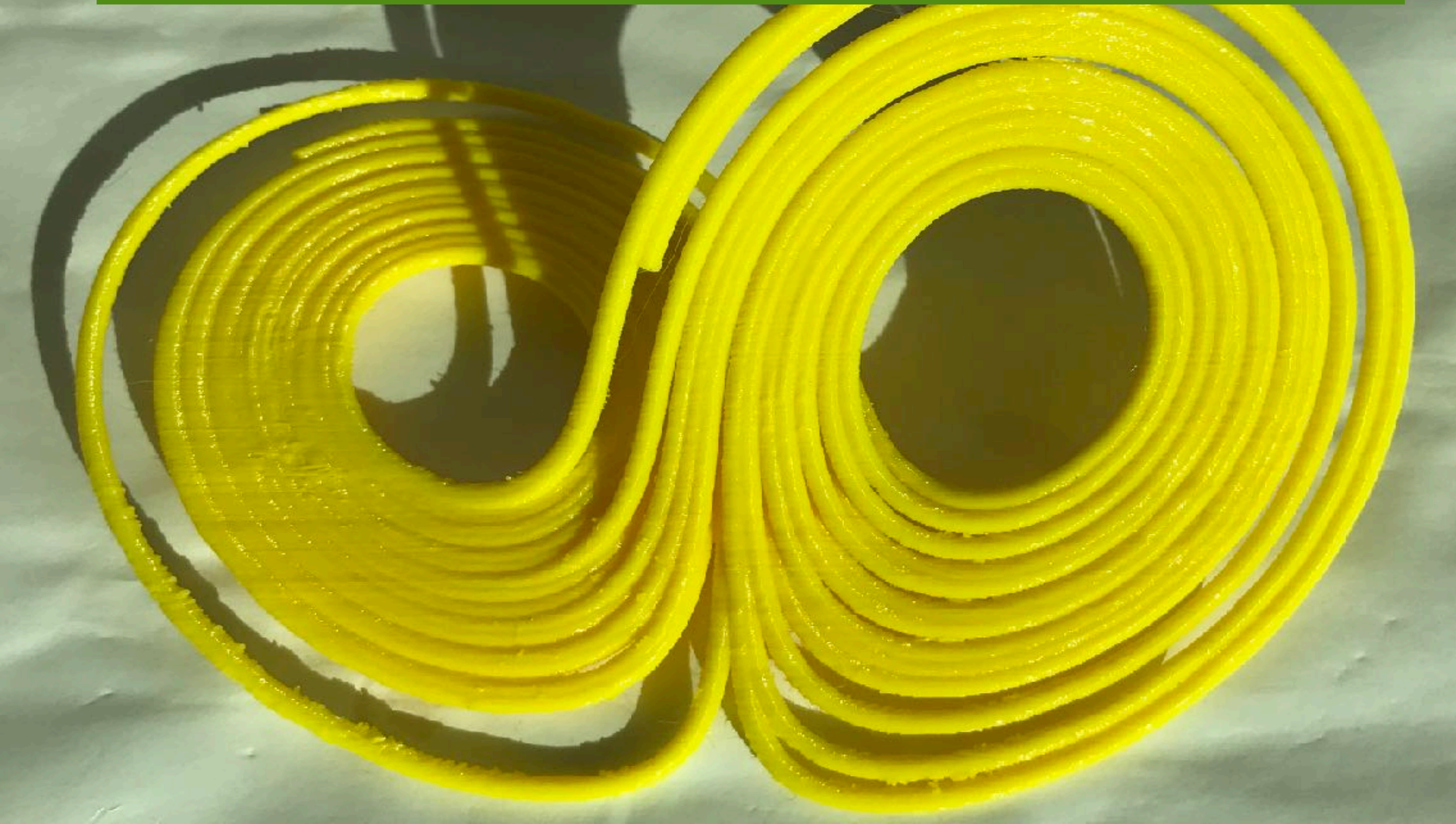
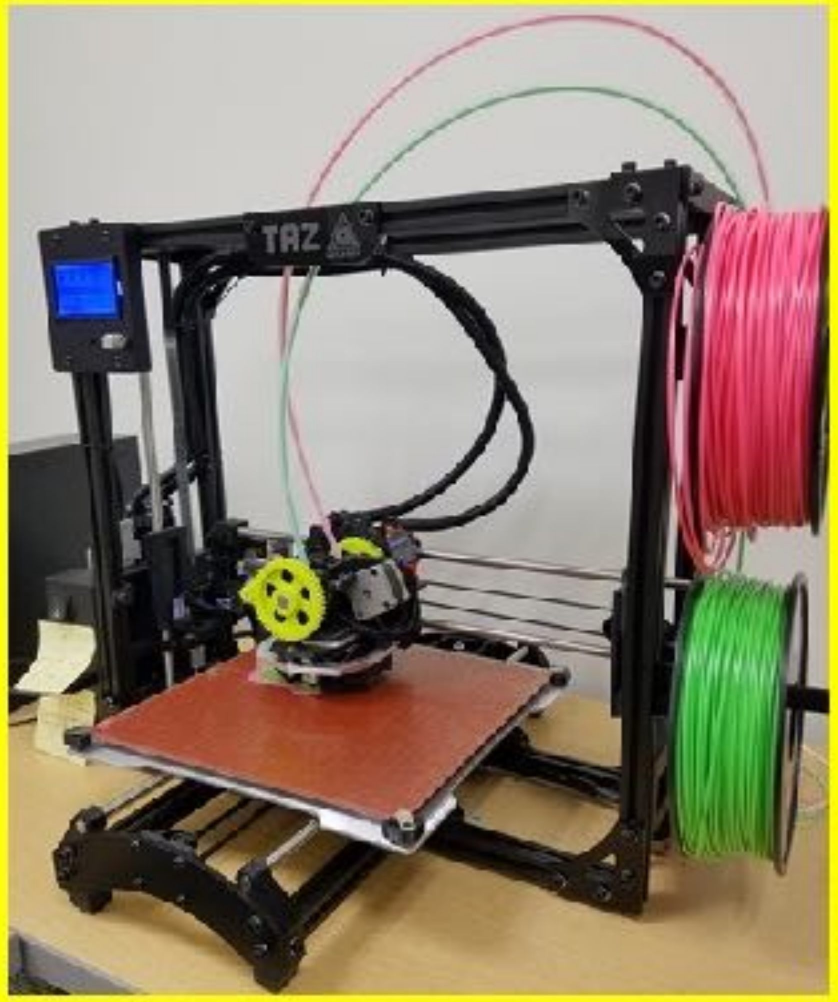
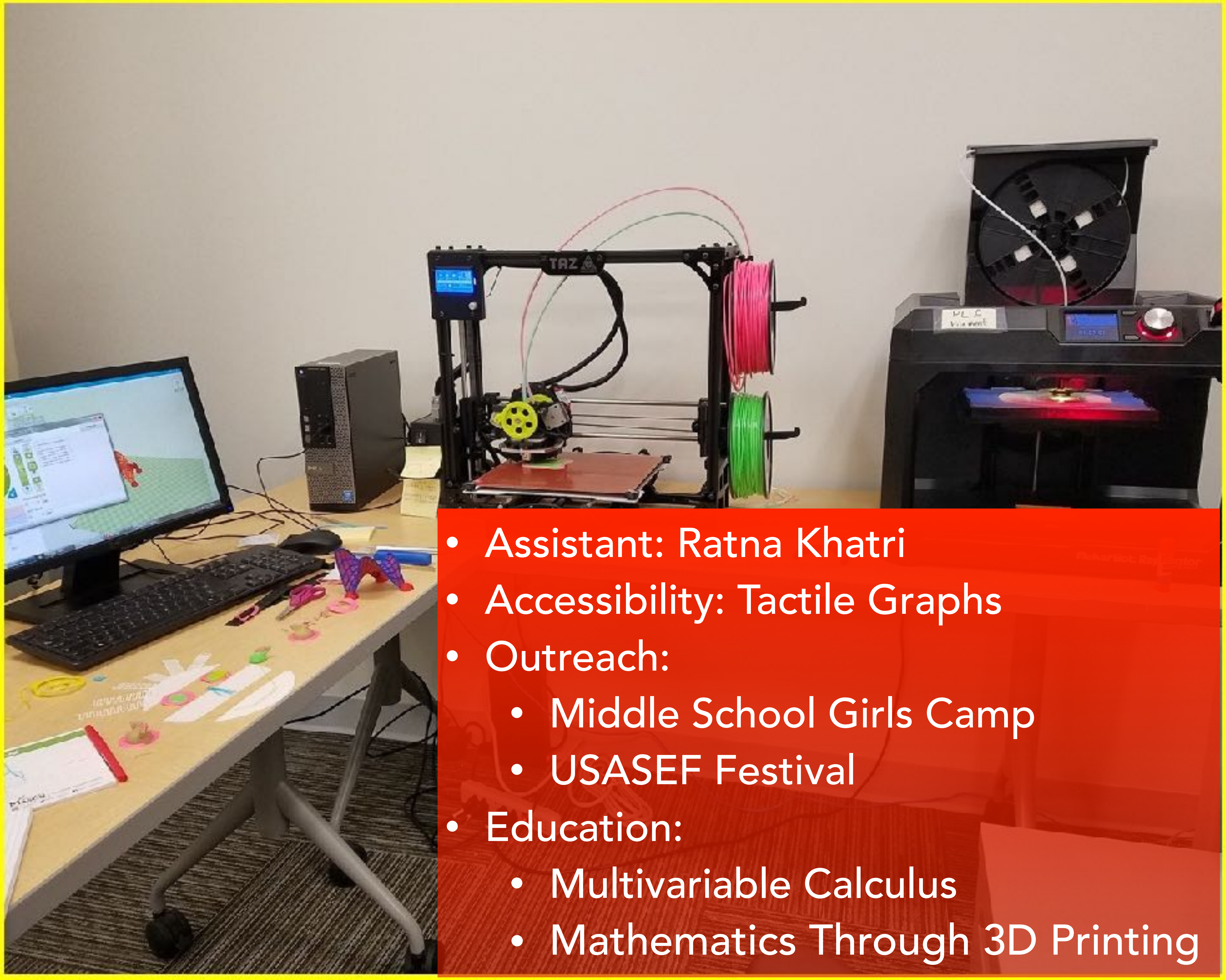
EVELYN SANDER, GEORGE MASON UNIVERSITY

PRINTING DYNAMICAL SYSTEMS AND CHAOS







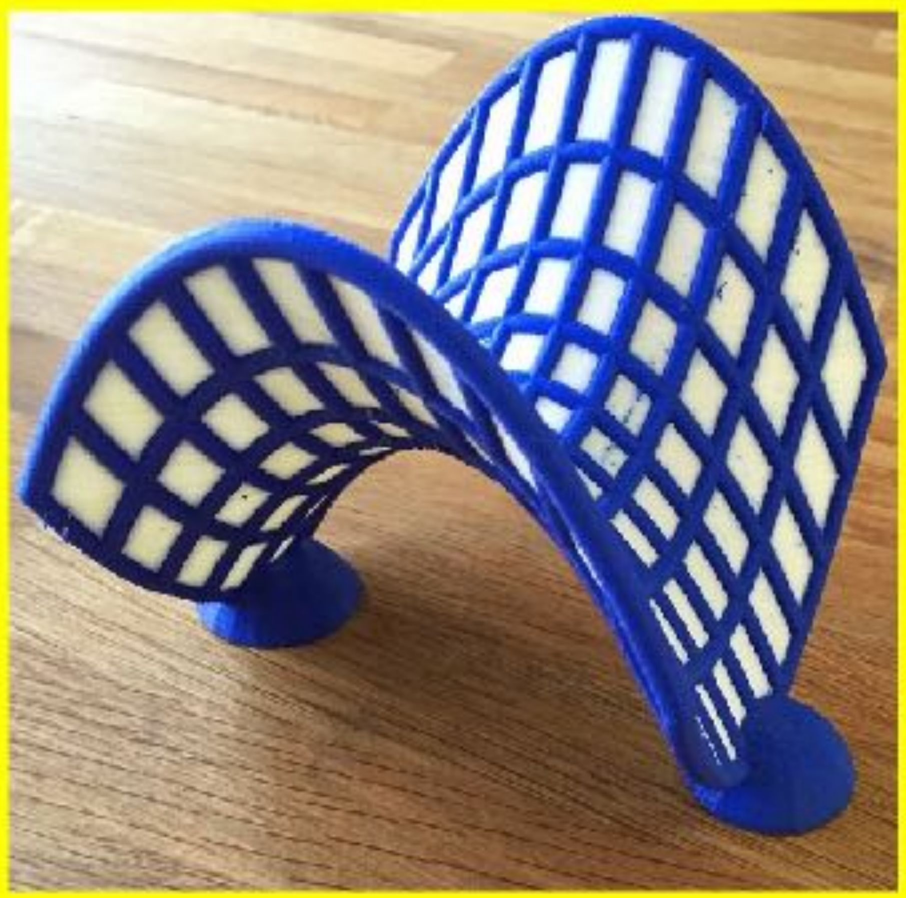
- Assistant: Ratna Khatri
- Accessibility: Tactile Graphs
- Outreach:
  - Middle School Girls Camp
  - USASEF Festival
- Education:
  - Multivariable Calculus
  - Mathematics Through 3D Printing

HTTP://GMUMATHMAKER.BLOGSPOT.COM

GMU MATH MAKERLAB

5 THINGS DESIGNED AND PRINTED BY STUDENTS, 1 THAT WAS NOT.

GOAL: BEYOND TRADITIONAL GEOMETRY



CONTENTS OF TALK

2. ITERATED FUNCTION SYSTEMS

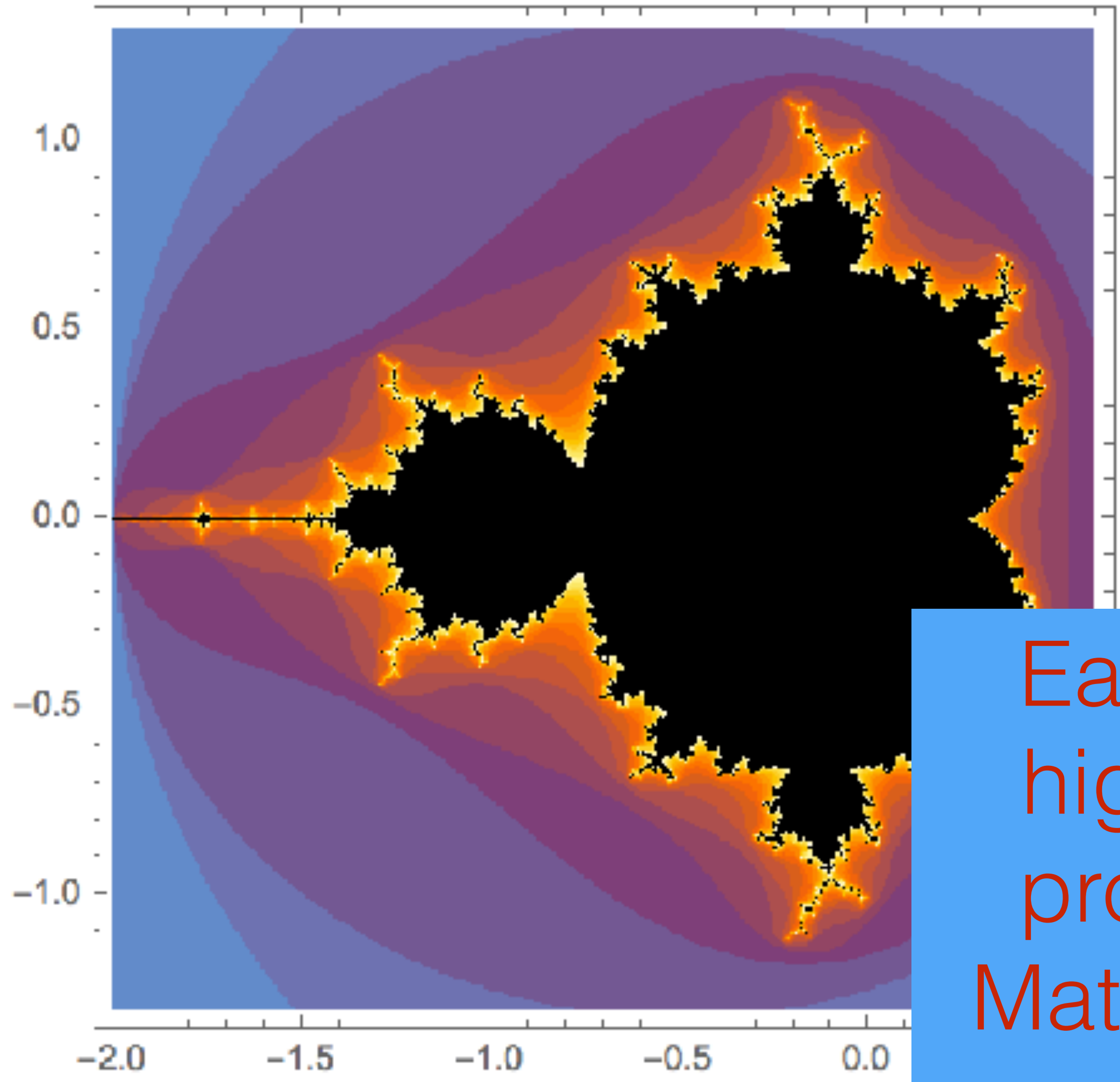
3. CHAOTIC ATTRACTORS

4. EVOLUTION EQUATIONS

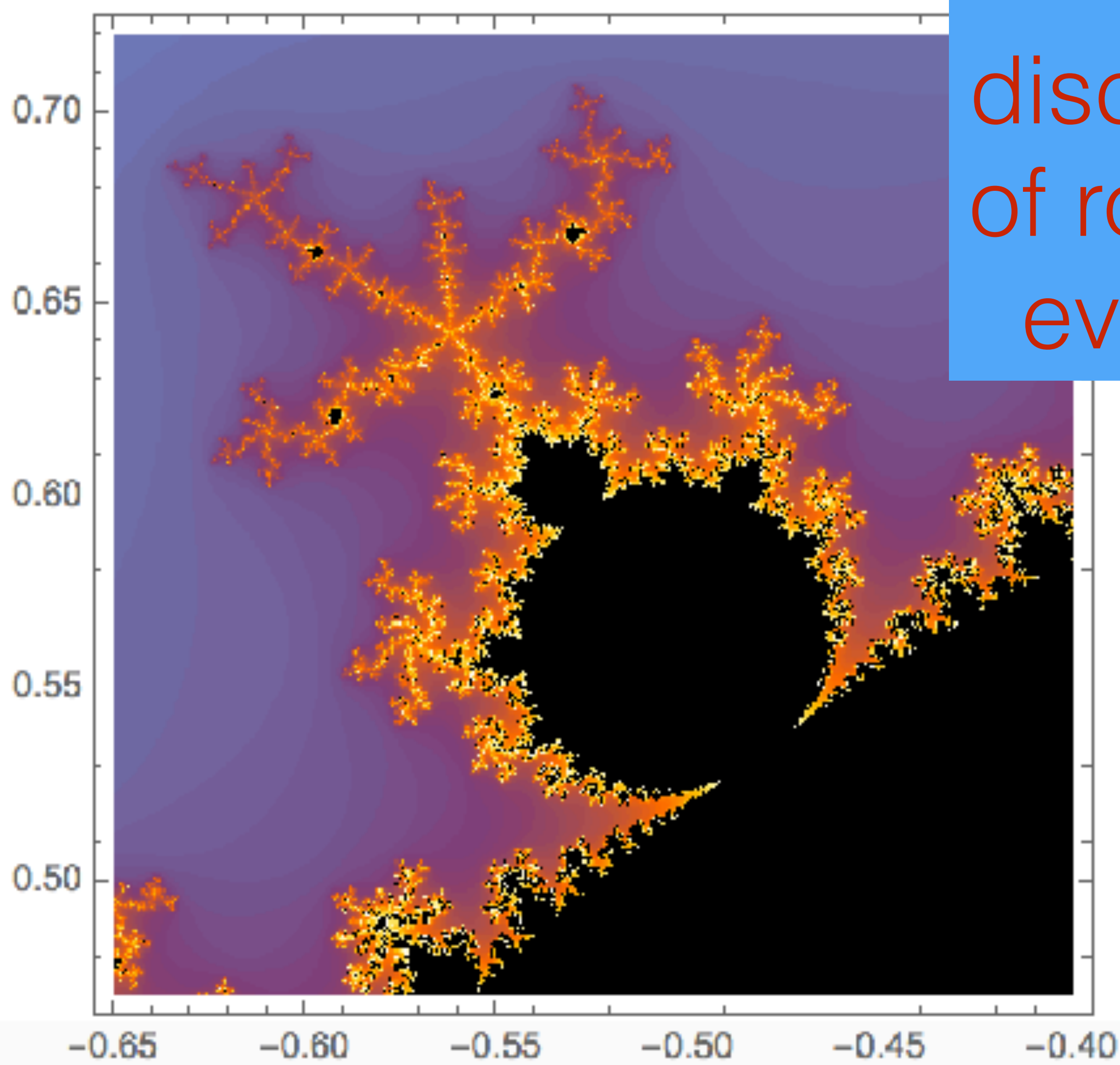
1. MANDELBROT AND JULIA SETS

1.MANDELBROT AND JULIA SETS

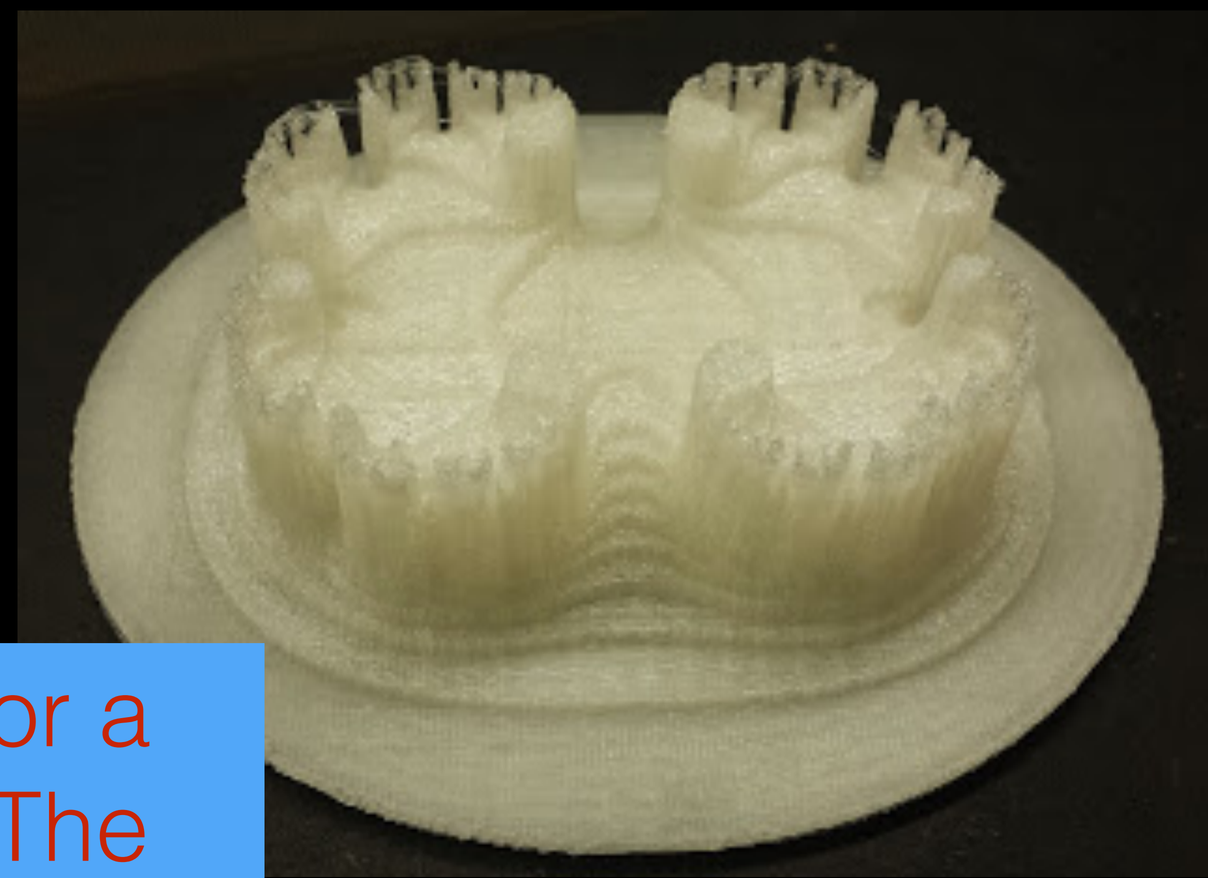
`MandelbrotSetPlot[MaxIterations → 20]`



`MandelbrotSetPlot[{-0.65 + 0.47 I,`



Easy to define even for a high school student. The process is automated by Mathematica. A good topic when starting complex numbers. Leads to a discussion of iteration. Plenty of room to add sophistication even at the graduate level.



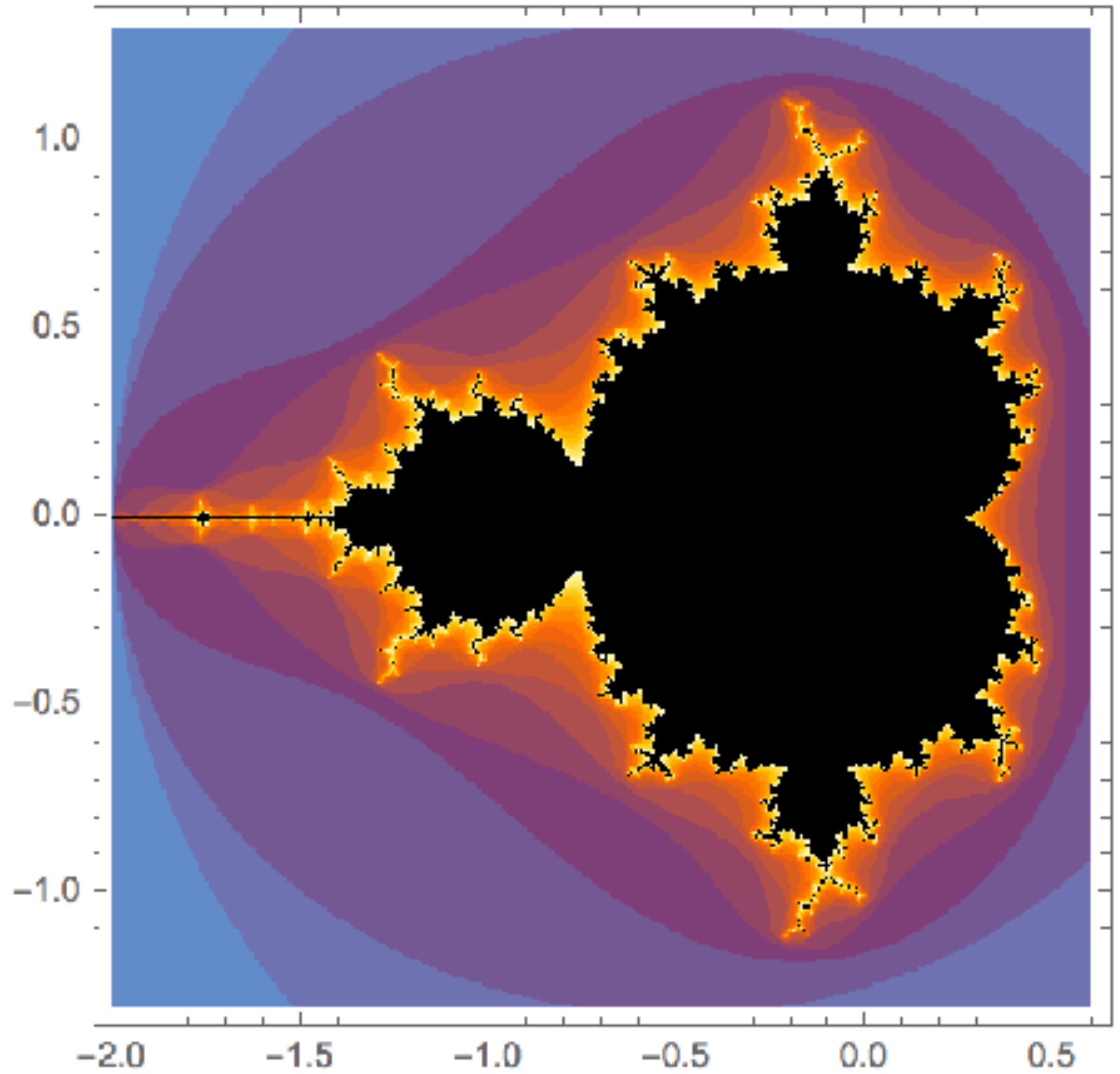
plex numbers c such
stays bounded under
$f(z) = z^2 + c$

Mathematica allows for the creation of STL files:

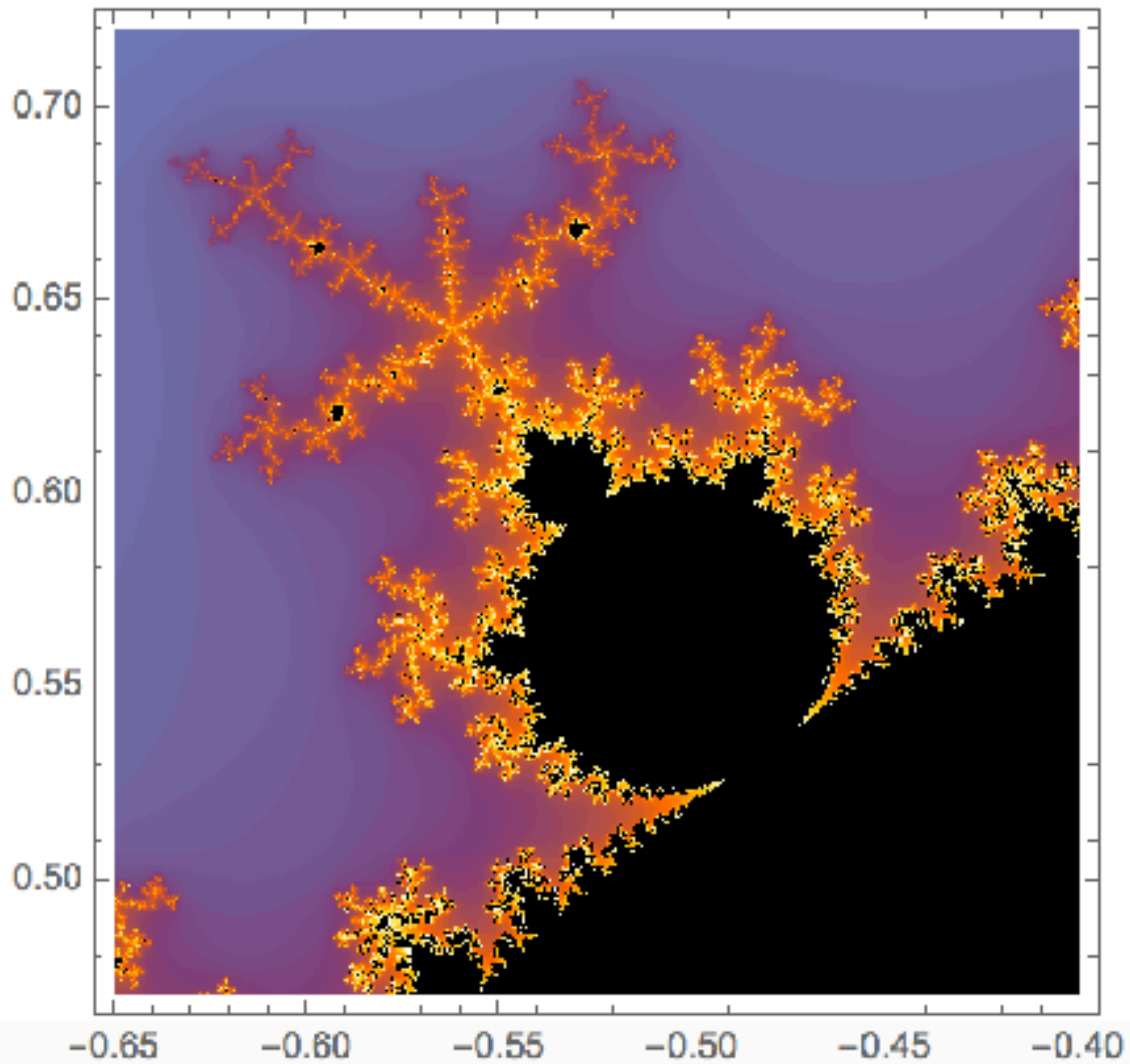
```
a = ParametricPlot3D[…]
Export["mymesh.stl", a]
```

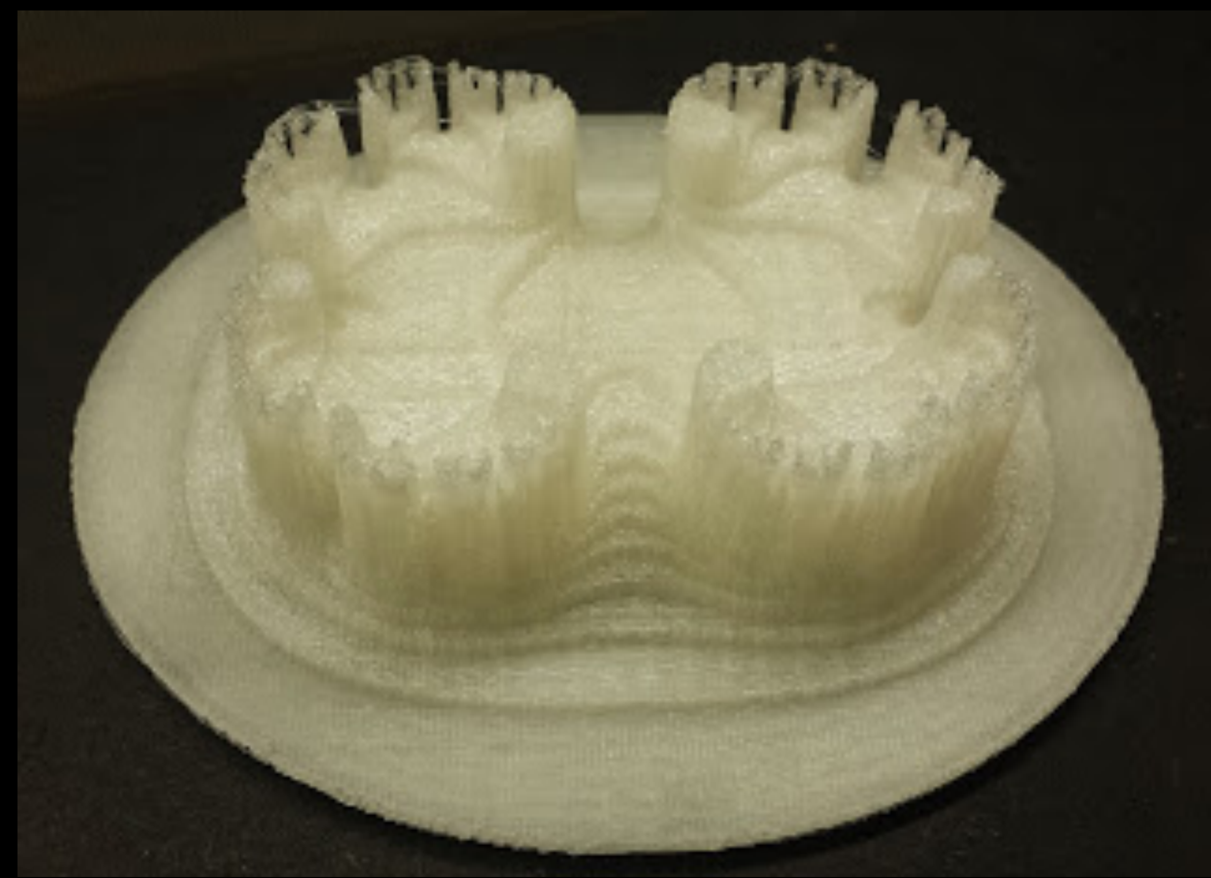
1.MANDELBROT AND JULIA SETS

```
MandelbrotSetPlot[MaxIterations → 20]
```




```
MandelbrotSetPlot[{-0.65 + 0.47 I, -0.4 + 0.72 I}, MaxIterations → 100]
```





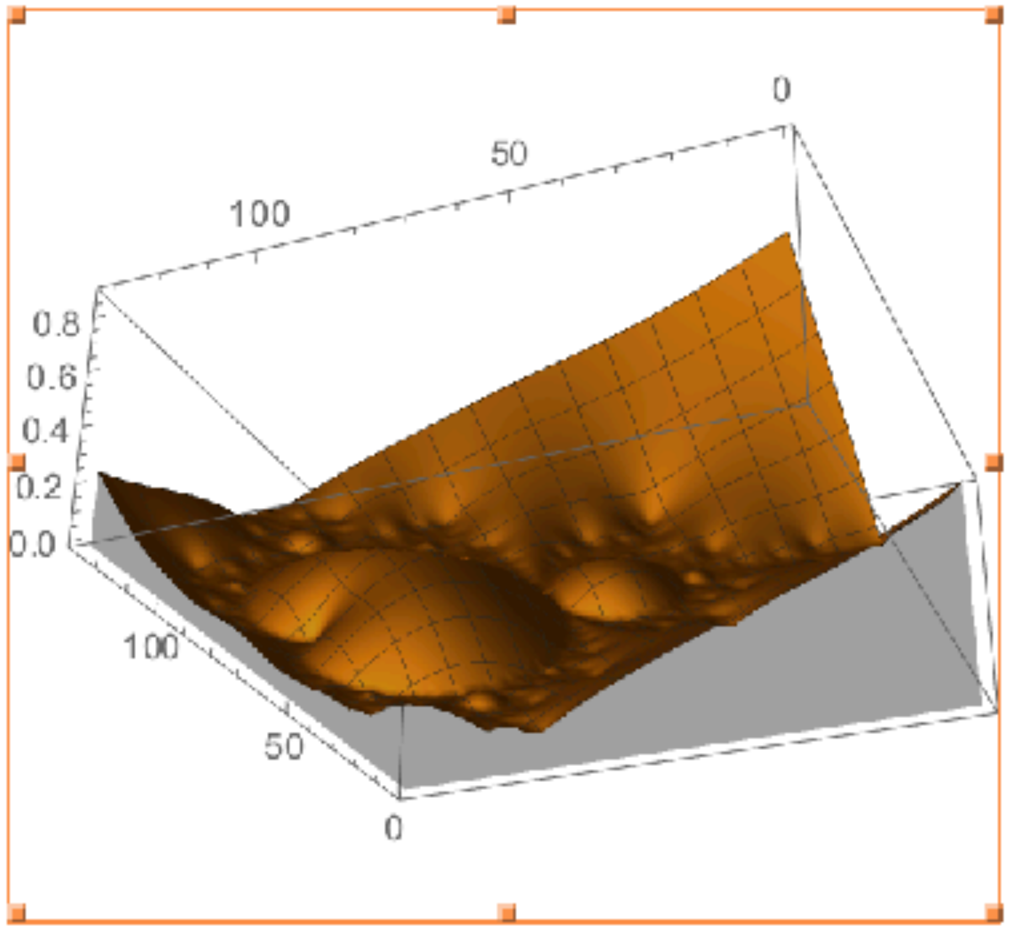
Complex numbers c such that 0 stays bounded under $f(z) = z^2 + c$

Mathematica allows for the creation of STL files:

a = ParametricPlot3D[…]
Export["mymesh.stl", a]

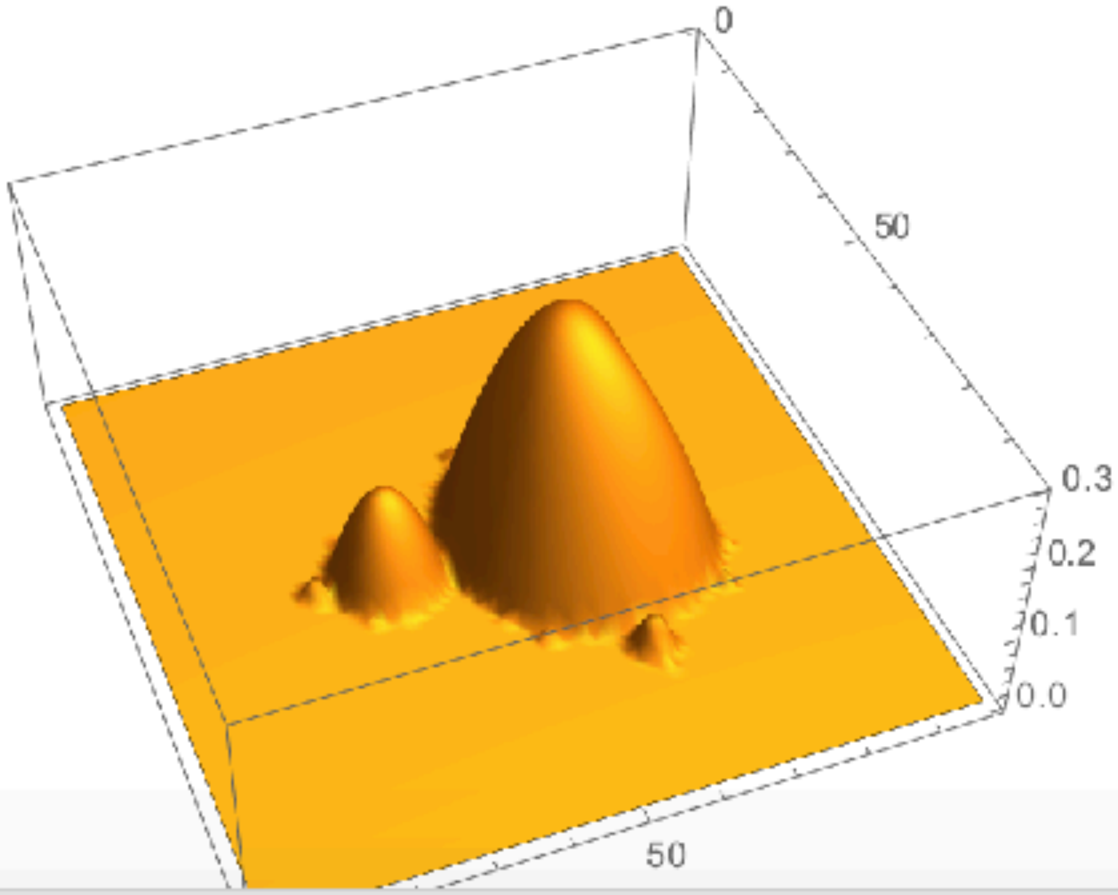
1.MANDELBROT AND JULIA SETS

```
data = Table[Max[MandelbrotSetDistance[#, "Interior"], MandelbrotSetDistance[#] / 4] &[x + I * y],
   {y, -1.5, 1, .02}, {x, -2, .6, .02}];
ListPlot3D[data, Filling -> Bottom]
```



The distance to the edge of the Mandelbrot set

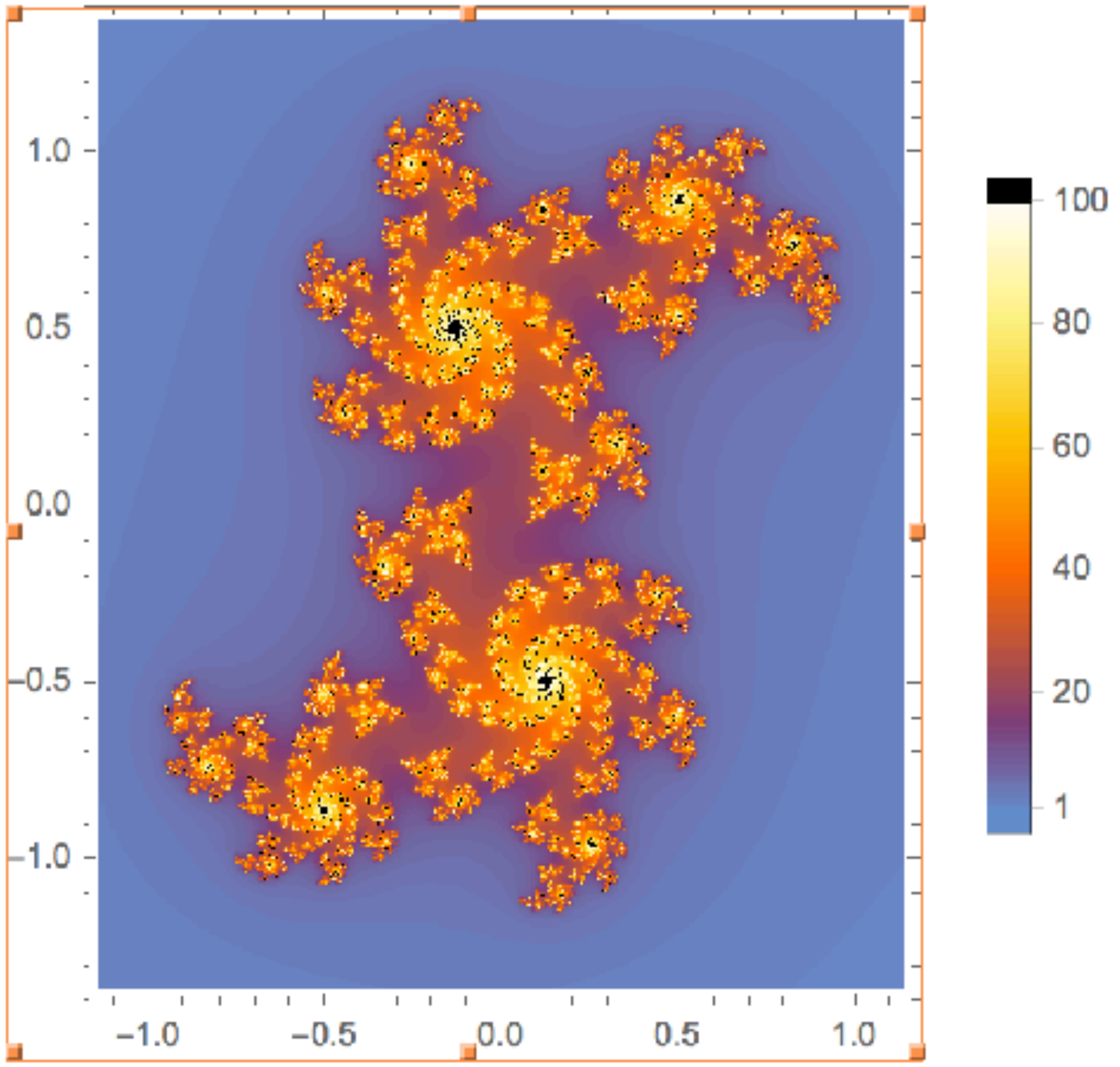
```
data = Table[MandelbrotSetDistance[x + I * y, "Interior", MaxIterations → 40], {x, -2., 1., .031},
   {y, -1.5, 1.5, .031}];
ListPlot3D[data, Mesh → None, PlotRange → Full, Filling -> Bottom]
```



The distance to the exterior of Mandelbrot set

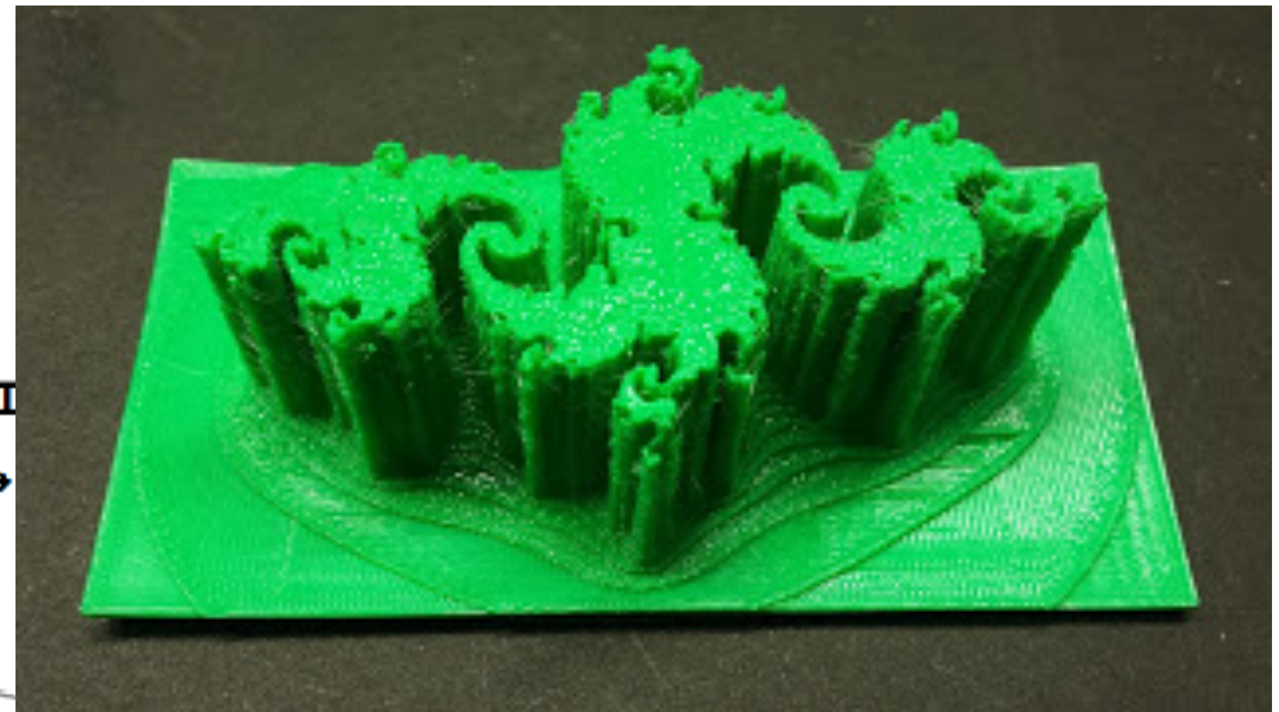
1.MANDELBROT AND JULIA SETS

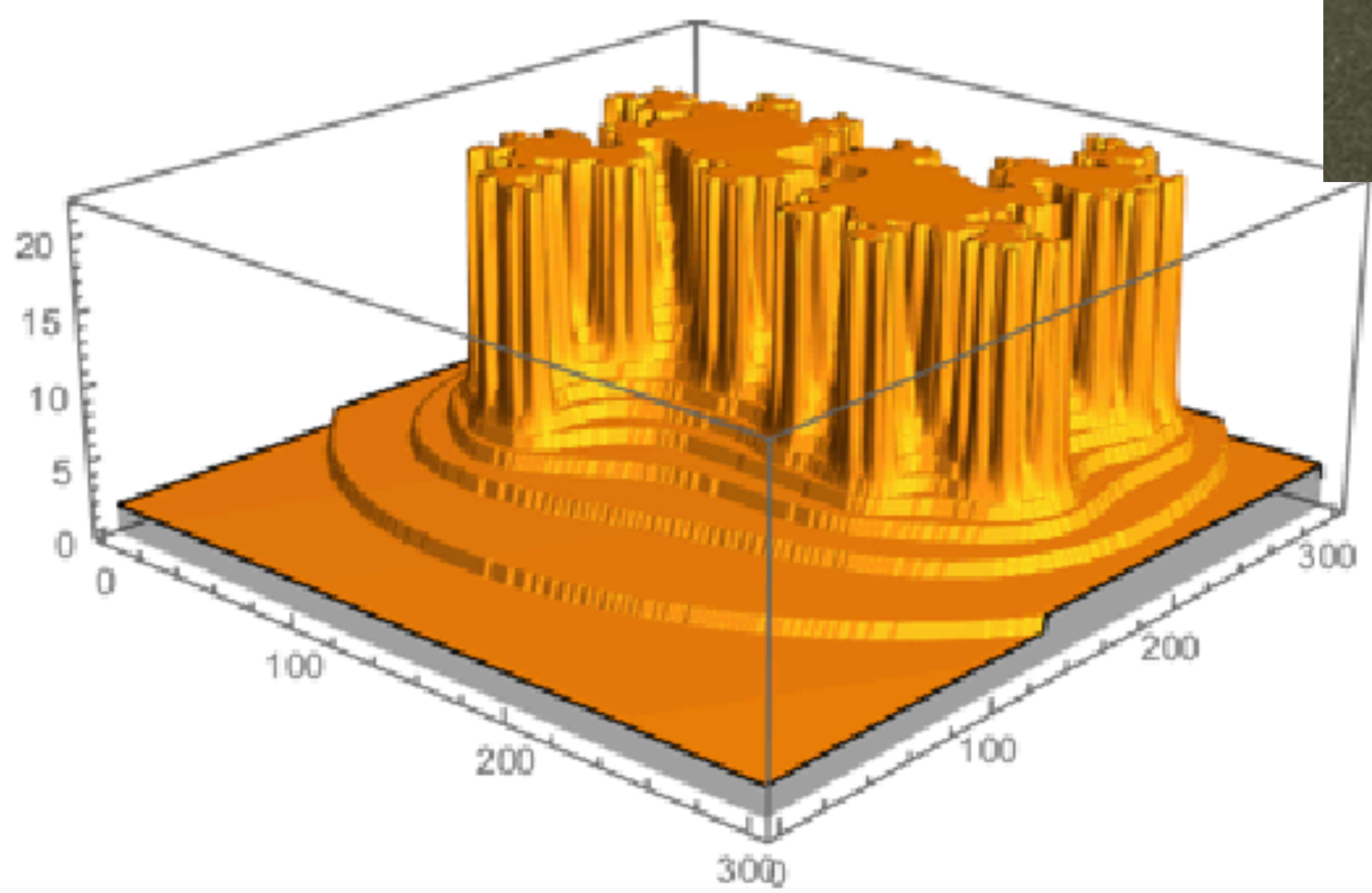
```
JuliaSetPlot[0.365 - 0.37 i, PlotLegends → Automatic]
```



Filled Julia set: set of points z that stays bounded under $f(z) = z^2 + c$

```
ListPlot3D[
 JuliaSetIterationCount[0.365 - 0.37 i, Table[x + I
   MaxIterations → 20] + 1, Mesh → False, Filling →
```

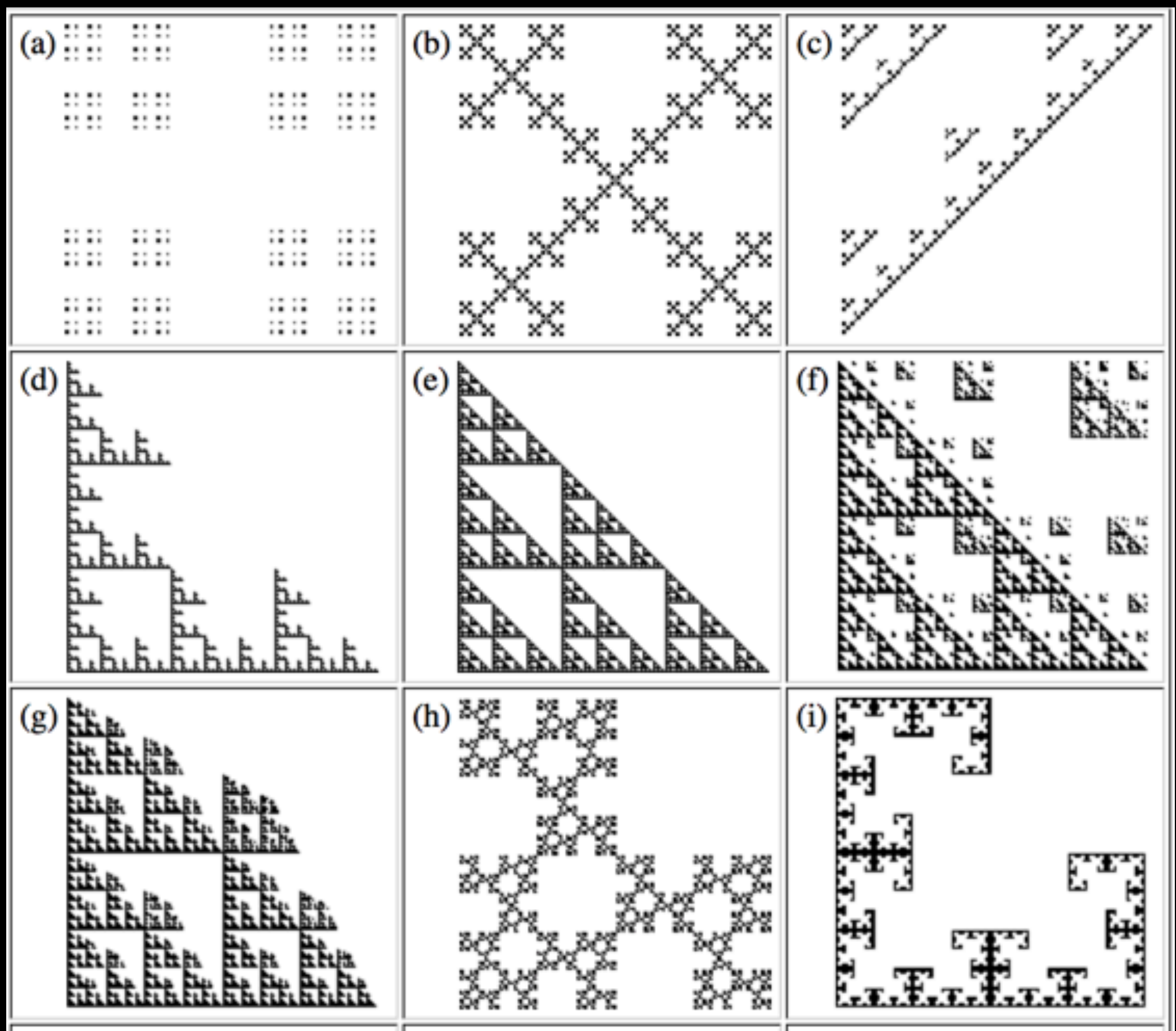


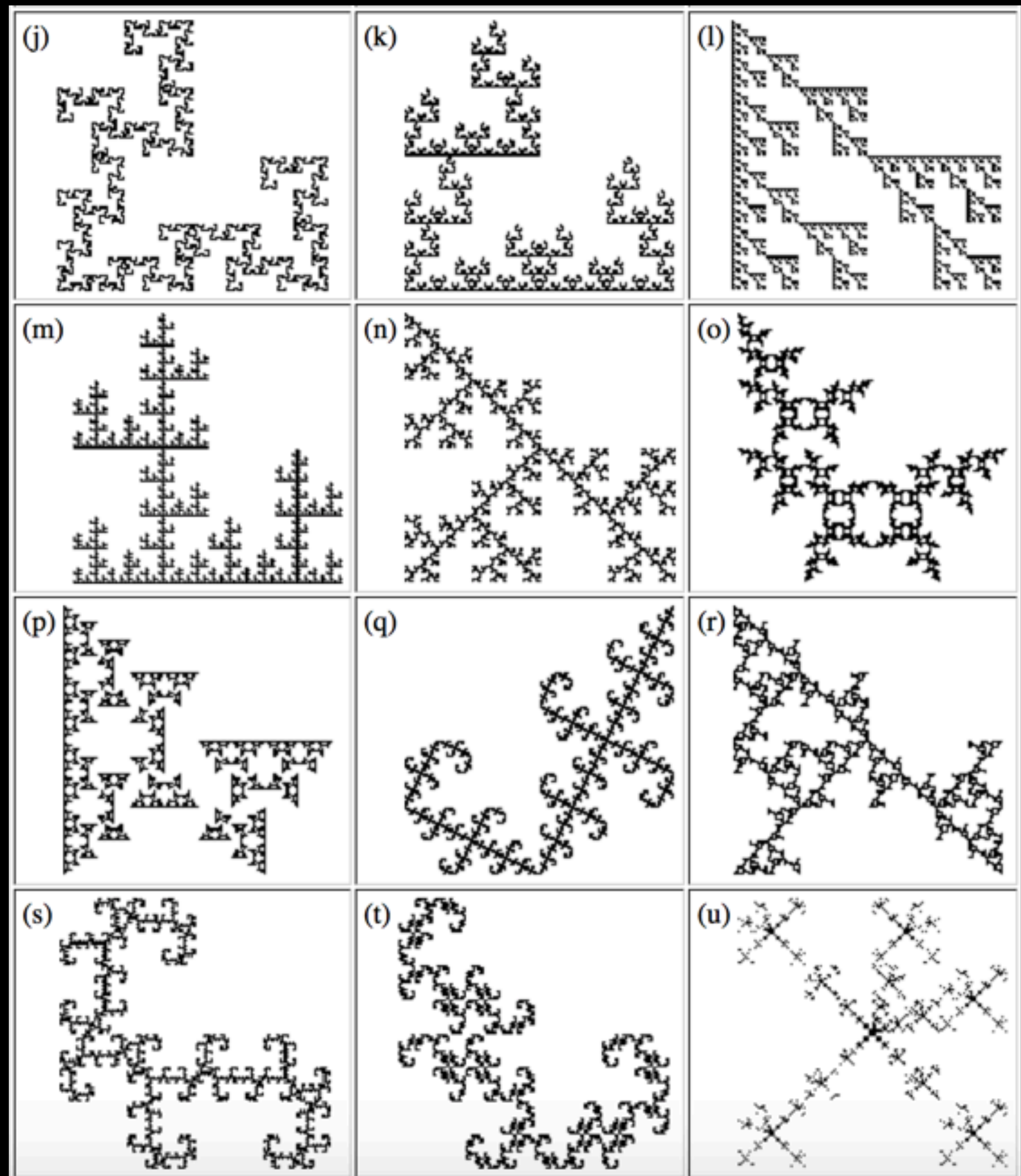


Number of iterates to divergence

2. ITERATED FUNCTION SYSTEMS

A type of fractal
Examples from Fractal Worlds @ Yale
Leads to discussion of transformations and dimension - even with high school geometry students. At the college level, can learn to compute fractal dimension and other dynamical concepts.





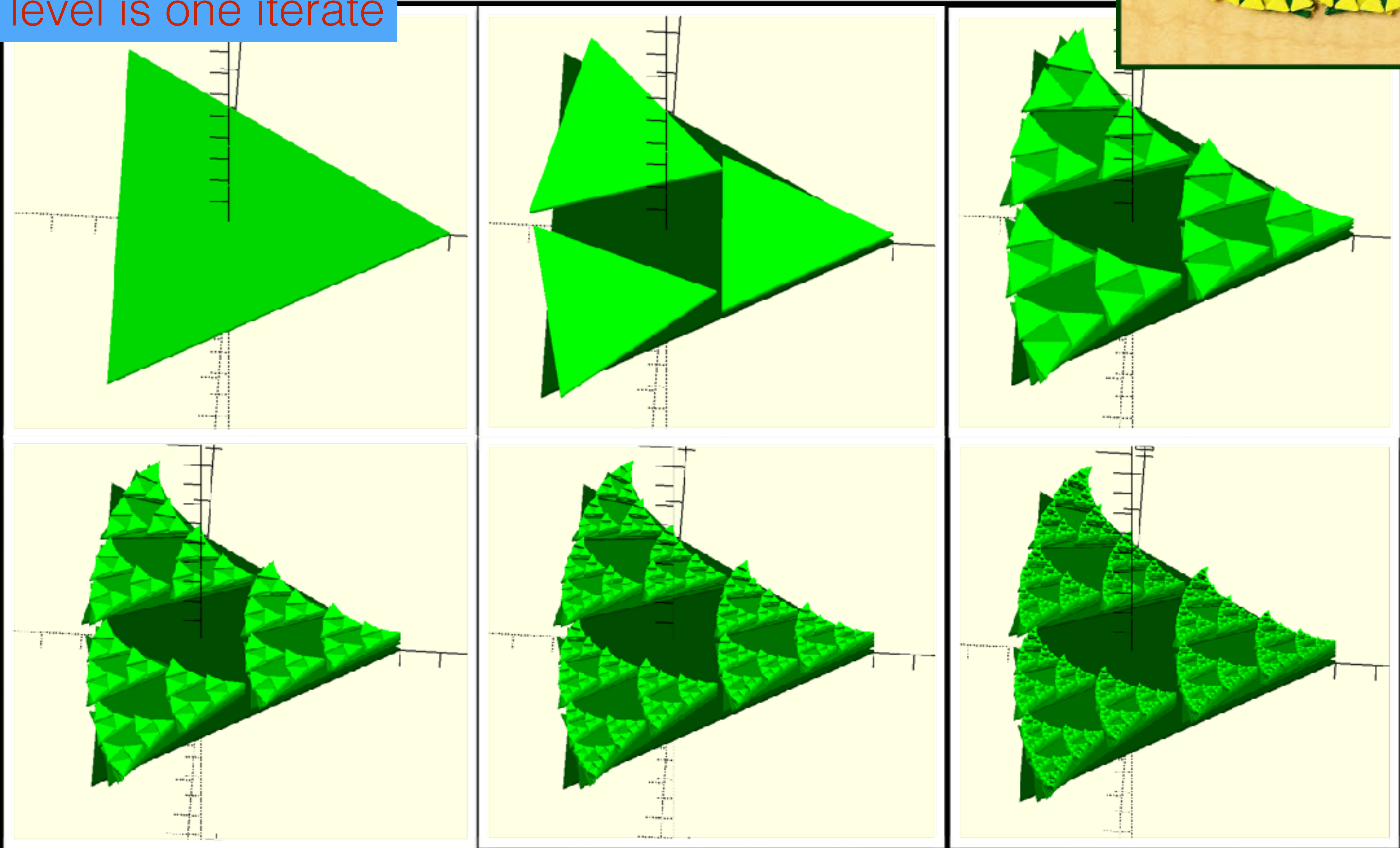
2. ITERATED FUNCTION SYSTEMS

```
// We create fractals! Modify the numbers below to change the final look.

// Here we make the size and shape
levels = 0; // number of levels for th
    you will WAIT!
height  = 1.0; // height of the first layer
len = 50; // length of the first segment
si = 3; // number of sides to the polygon of the starting shape
```

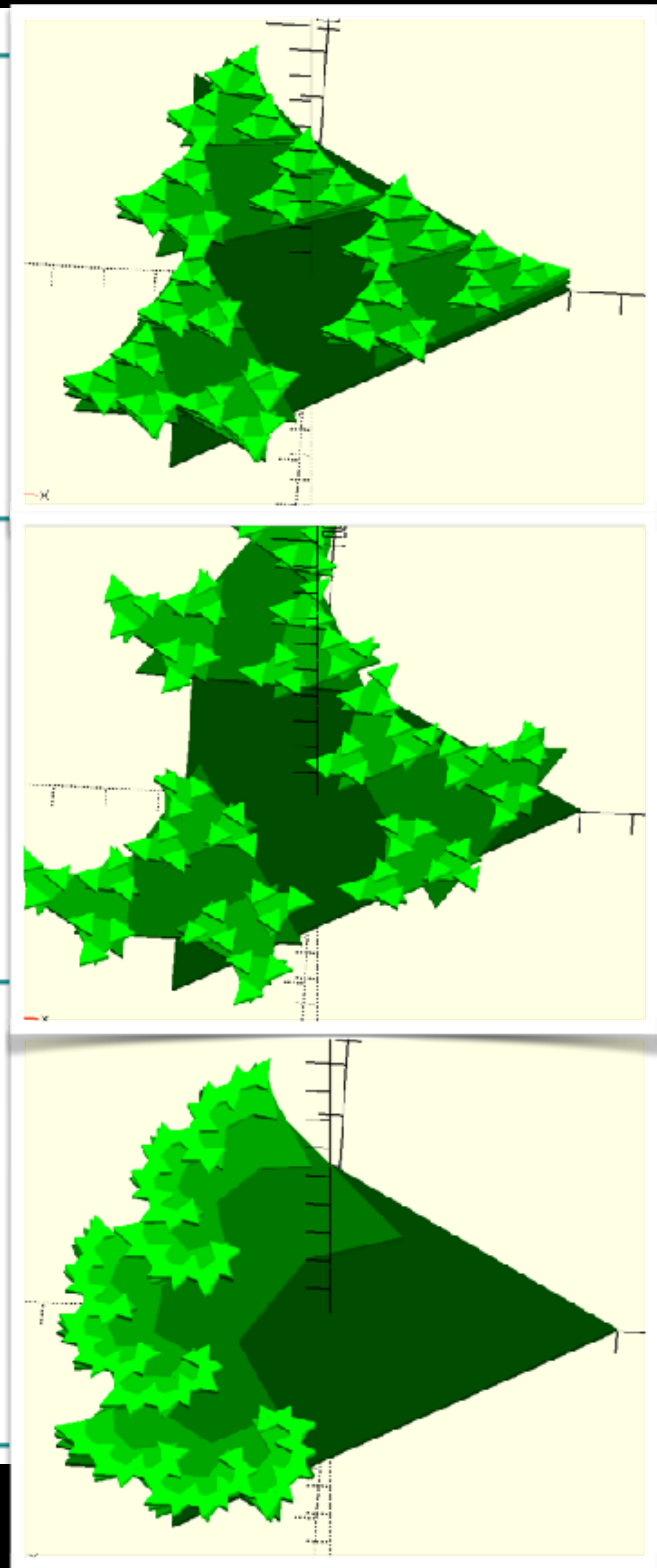
Generates a fractal using OpenSCAD

Each level is one iterate



2. ITERATED FUNCTION SYSTEMS

```
//------------------------------------------------------------
// The first transformation
scale1x = 0.5; // Scaling in direction x
scale1y = 0.5; // Scaling in direction y
theta1  = 11;  // Angle of rotation in degrees
trans1x = -0.25*len; // Translation in direction x
trans1y = -0.4*len; // Translation in direction y
//------------------------------------------------------------
// The second transformation
scale2x = 0.5; // Scaling in direction x
scale2y = 0.45; // Scaling in direction y
theta2 =  3; // Angle of rotation in degrees
trans2x = 0.5*len; // Translation in direction x
trans2y = 0*len; // Translation in direction y
//------------------------------------------------------------
// The third transformation
scale3x = 0.5; // Scaling in direction x
scale3y = 0.53; // Scaling in direction y
theta3 =  -14;   // Angle of rotation in degrees
trans3x = -0.25*len; // Translation in direction x
trans3y = 0.43*len; // Translation in direction y
//------------------------------------------------------------
```



parameters control transformation

3. CHAOTIC ATTRACTORS

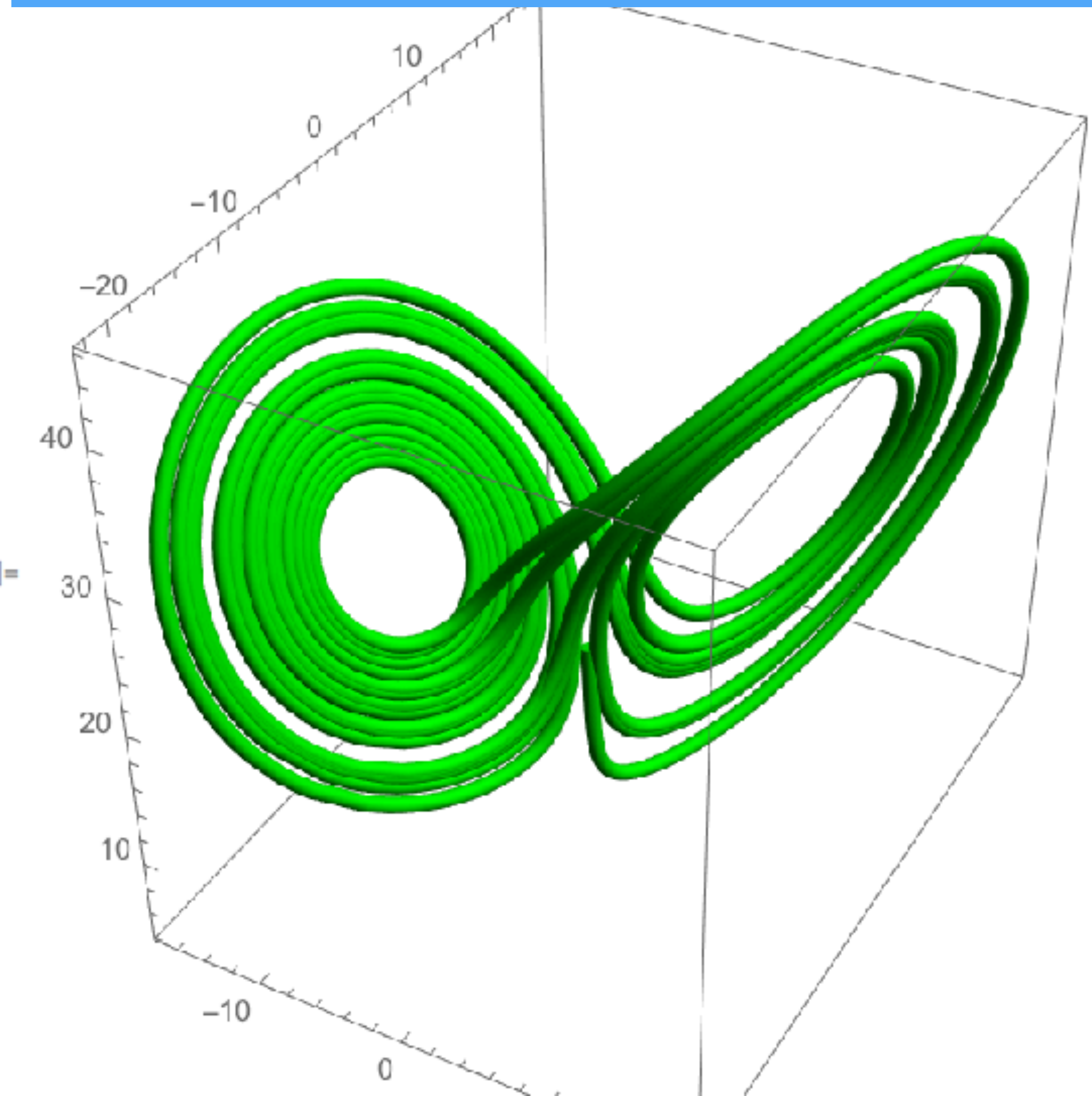
In[44]:=
```
rho = 28; sig = 10; beta = 8/3;
TubeThickness = 0.52;
Tmax = 50;
xinit = 11.13261;
yinit = 17.5124;
zinit = 21.0398;
```

In[48]:=
```
q = NDSolve[{x'[t] == (sig * (y[t] - x[t])), y'[t] == (x[t] * (rho - z[t]) - y[t]), z'[t] == (x[t] * y[t] - beta * z[t]),
    x[0] == xinit, y[0] == yinit, z[0] == zinit}, {x, y, z}, {t, 0, Tmax}, MaxSteps -> Infinity];
```

In[49]:=

This topic involves understanding of solutions to differential equations. It is a good lecture demonstration for an first ODE class, but further discussion would require learning about chaos and its properties.

Out[50]=



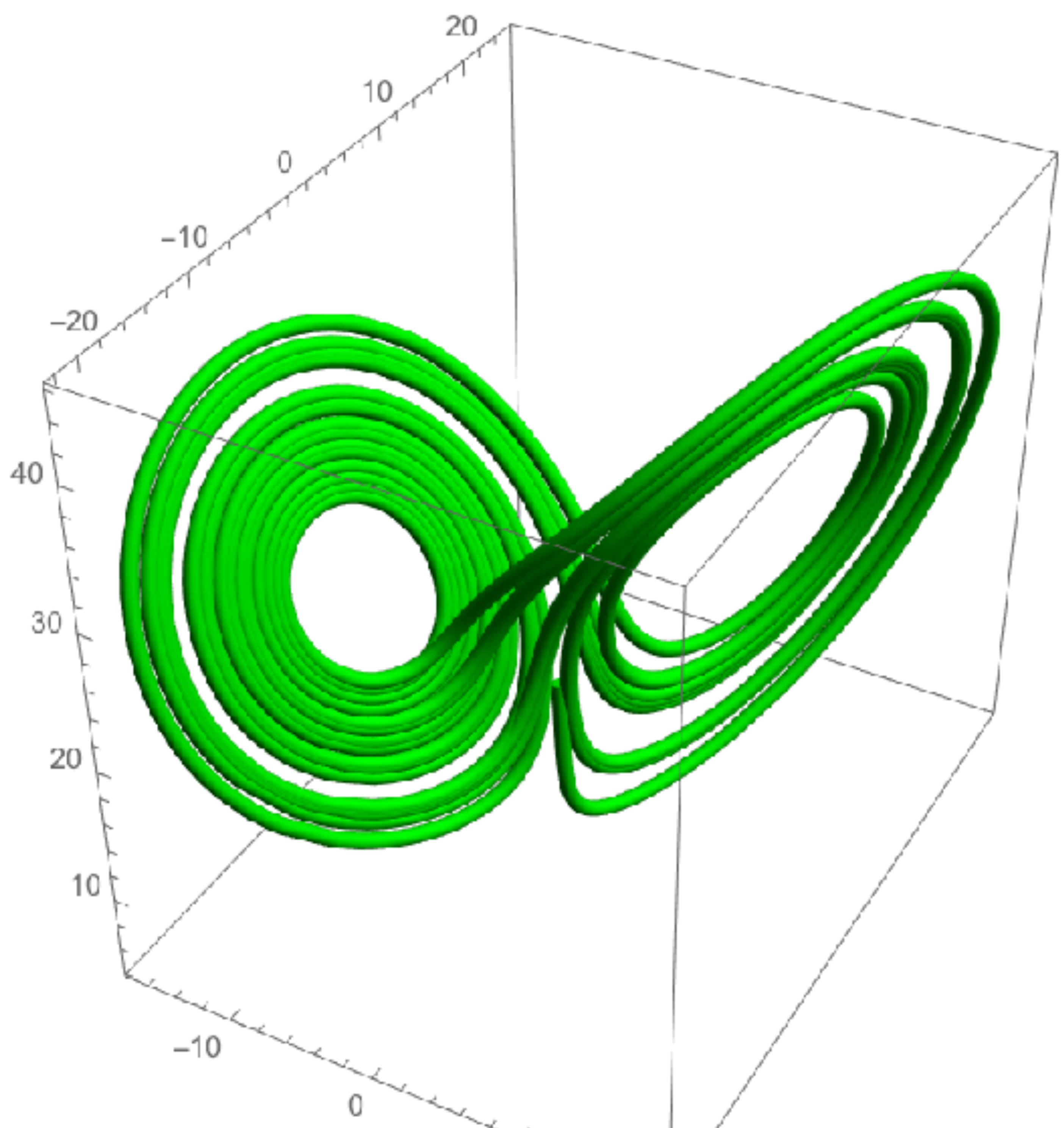
3. CHAOTIC ATTRACTORS

```
In[44]:= rho = 28; sig = 10; beta = 8 / 3;
         TubeThickness = 0.52;
         Tmax = 50;
         xinit = 11.13261;
         yinit = 17.5124;
         zinit = 21.0398;

In[48]:= q = NDSolve[{x'[t] == (sig * (y[t] - x[t])), y'[t] == (x[t] * (rho - z[t]) - y[t]), z'[t] = (x[t] * y[t] - beta * z[t]),
              x[0] == xinit, y[0] == yinit, z[0] == zinit}, {x, y, z}, {t, 0, Tmax}, MaxSteps -> Infinity];

In[49]:= fig1 = ParametricPlot3D[Evaluate[{x[t], y[t], z[t]} /. q], {t, 5, Tmax / 2},
           PlotStyle → {Tube[TubeThickness, PlotPoints → Tmax], Green}, PlotRange → All];
         Show[fig1]
```

Out[50]=

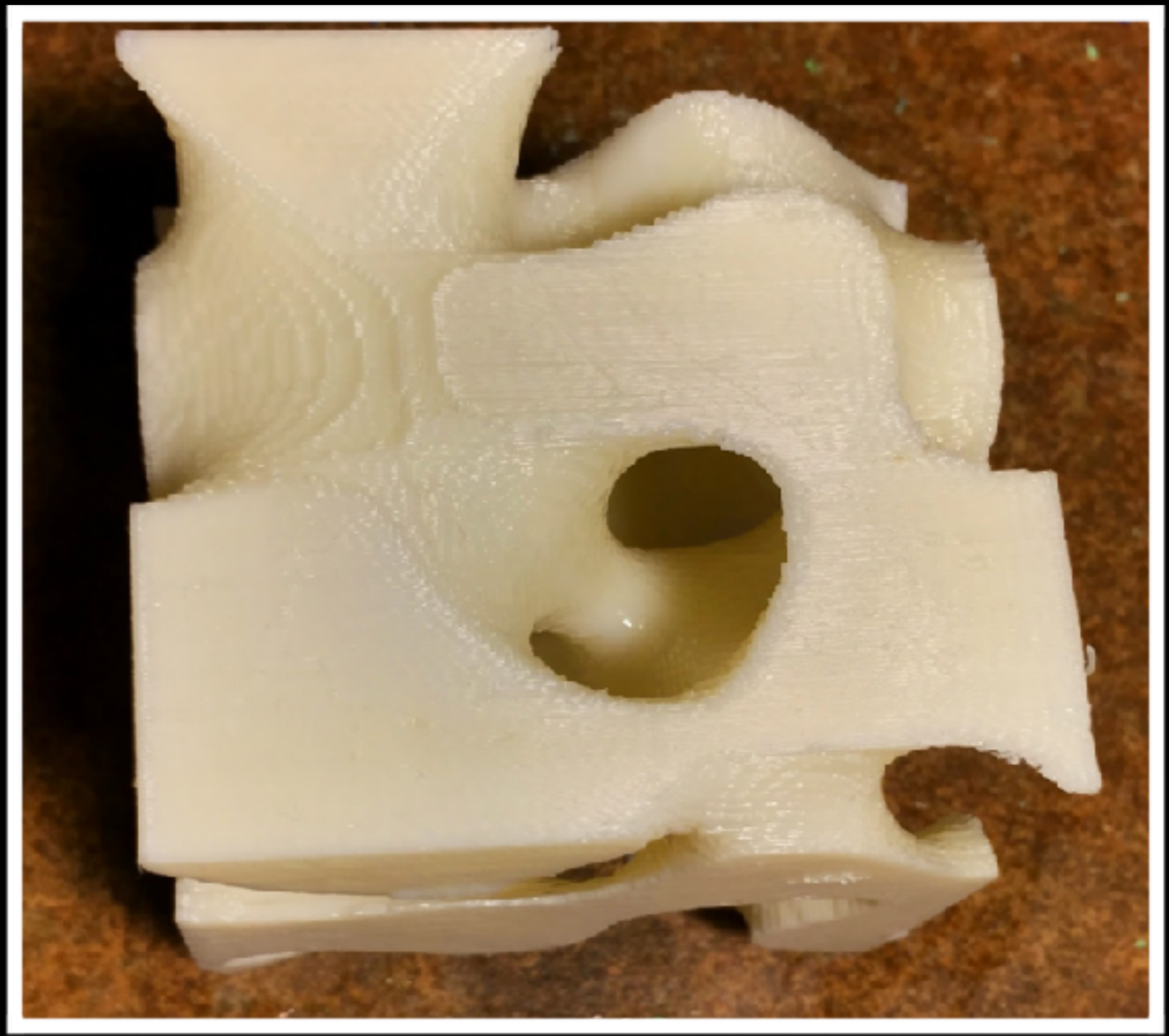


Mathematica NDSolve solves a differential equation. We put a Tube around the solution curve to make it 3D.

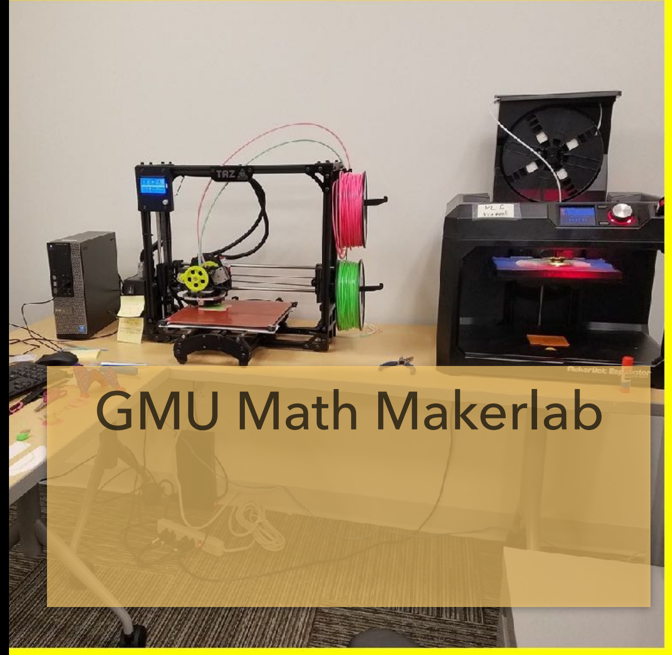
Number of iterates to divergence

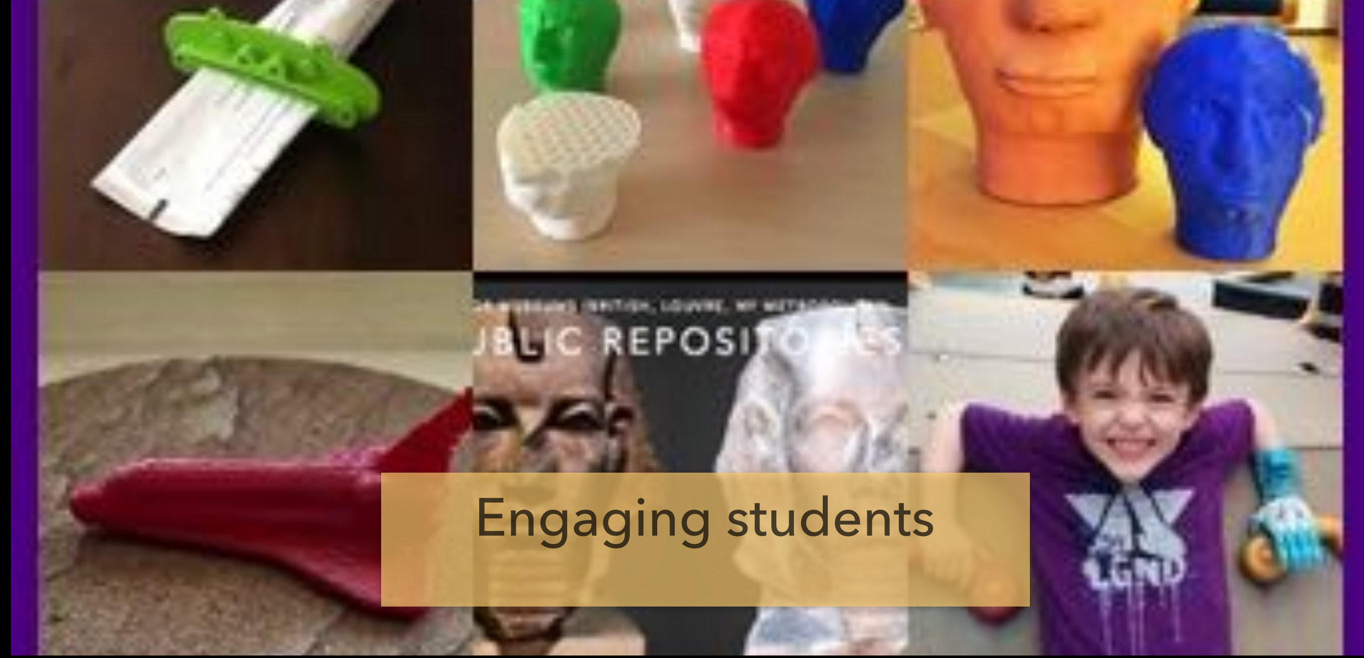
4. EVOLUTION EQUATIONS

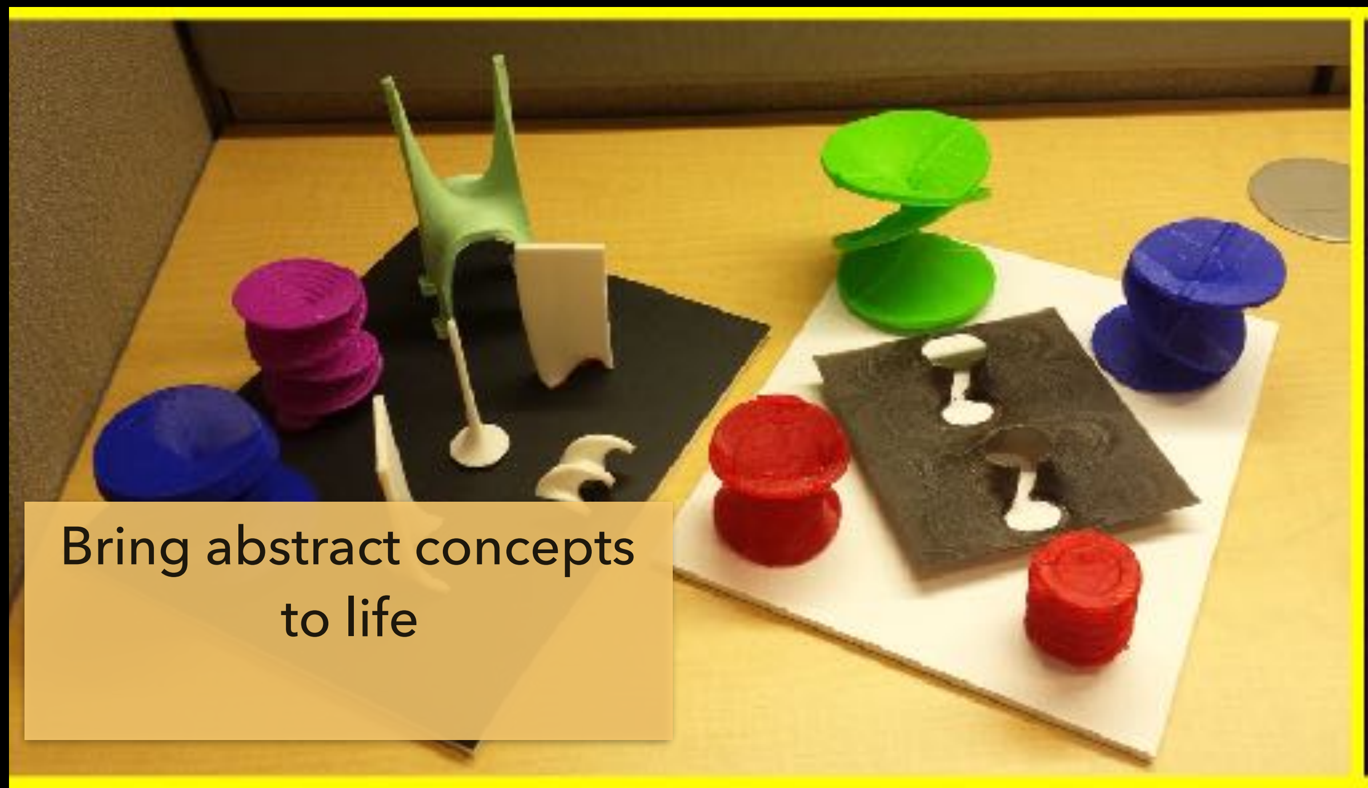
Computation beyond the capabilities of a standard interpreted code language. Generated by solving a partial differential equation using C++ code. The data is a set of points in a solid set. Create a mesh from data using ParaView. Printed with dissolvable filament.

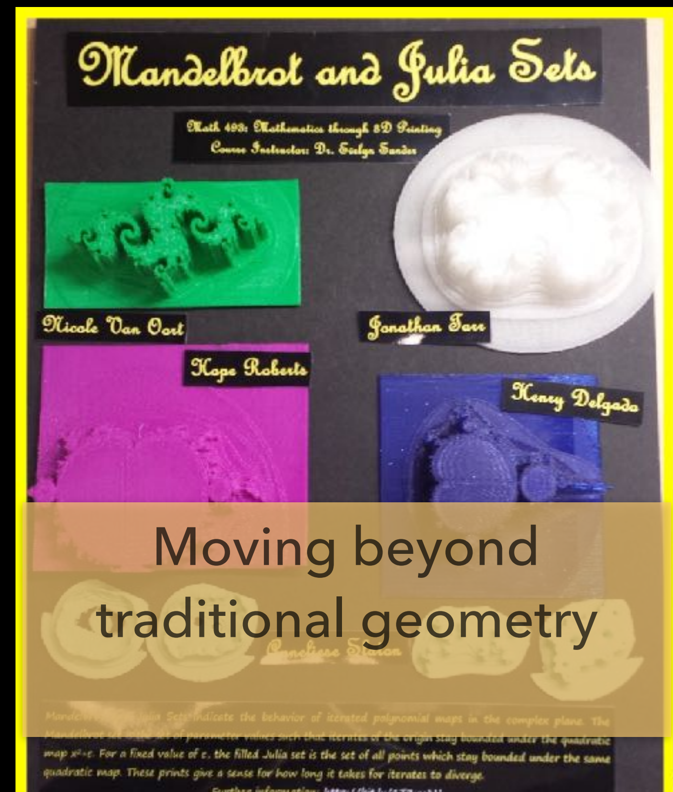


Spinodal decomposition for the Cahn-Hilliard equation.



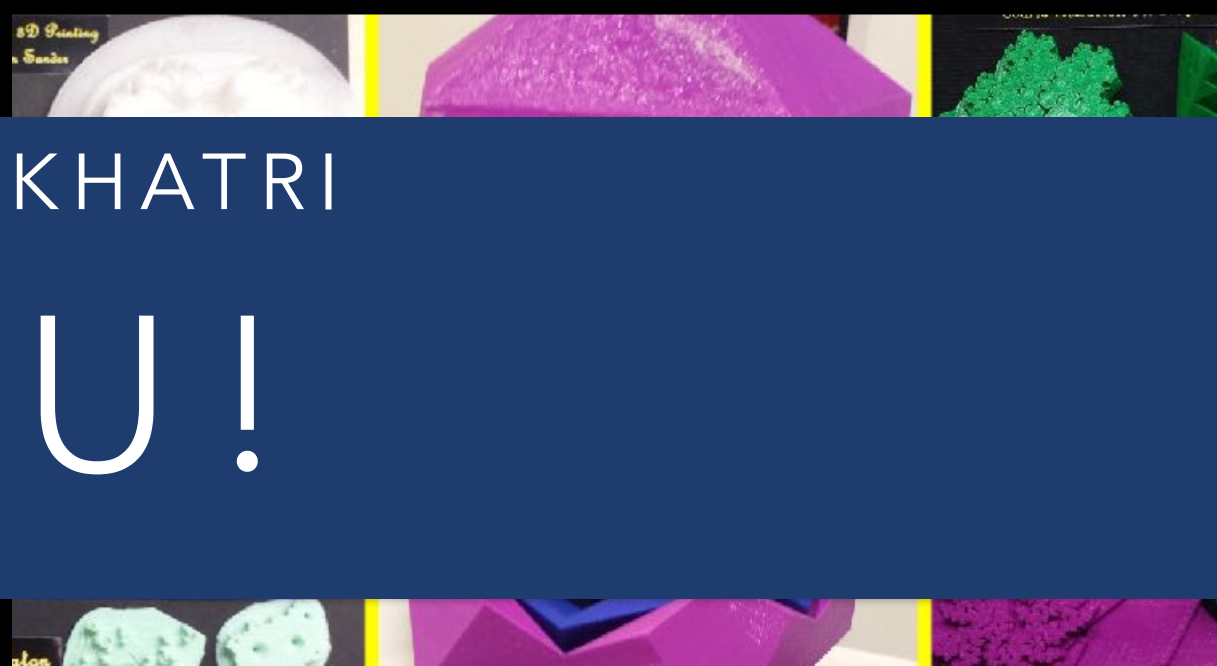






Dynamical systems and chaos

Evelyn Sander esander@gmu.edu
GMU Math Makerlab
http://gmumathmaker.blogspot.com



SPECIAL THANKS TO RATNA KHATRI

THANK YOU!