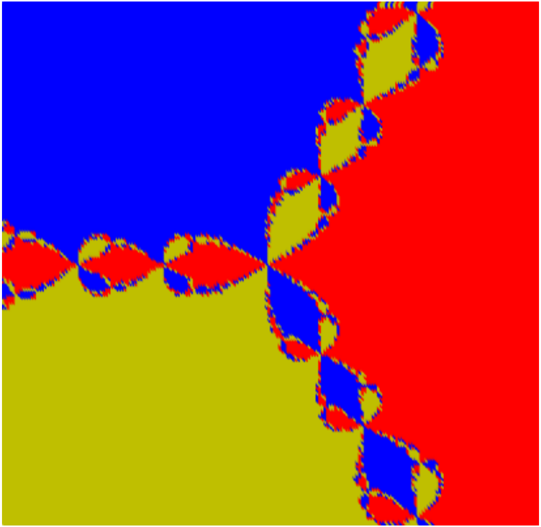
Project-based Programming in a Mathematics Course

Joe Barrera

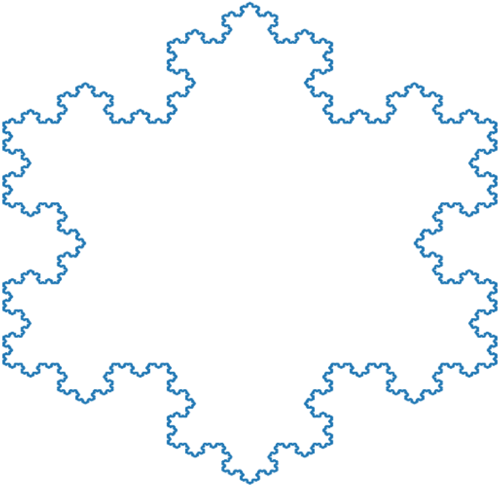
Converse University

August 16, 2023

Introduction

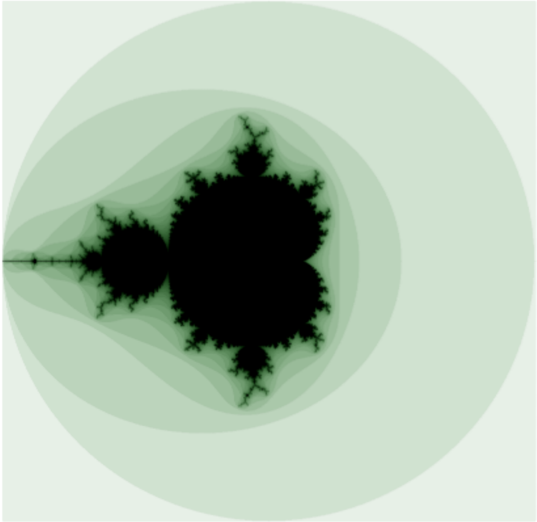


- Motivation
  - January terms 2020 & 2023
  - Learned Python over summer 2019
  - Started making fractals
- Course setup
  - 16 3-hour courses broken into two halves
  - Lecture portion
  - Programming portion



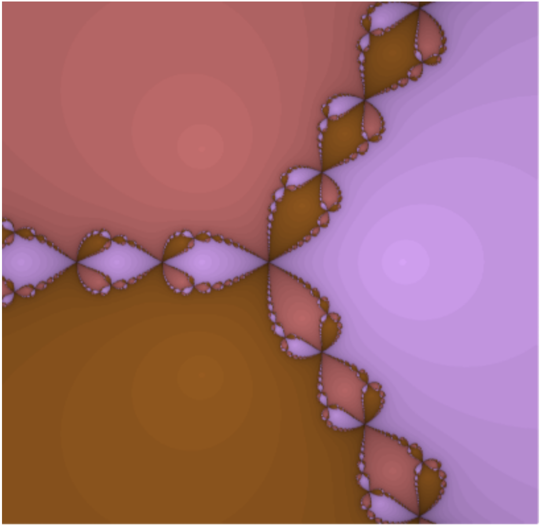
- Hybrid inquiry-based/group-based: examples in class, then group work at desks or board
- Notes prepared by instructor
    - Sourced from Cain's *Complex Analysis* [2], Spindler's JIBLM notes [3], and Brown and Churchill's *Complex Variables and Applications* [5]

Lecture portion



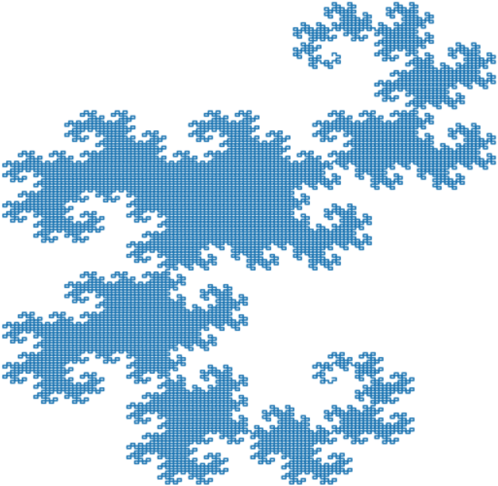
- Coverage
  - Limits
  - Triangle inequality
  - Differentiation & the Cauchy-Riemann equations
  - Elementary functions, including exponentials and logarithms
  - Curve parametrization, line integrals, and contour integrals
- Homework assigned every class with some presentations the next class period

Programming portion



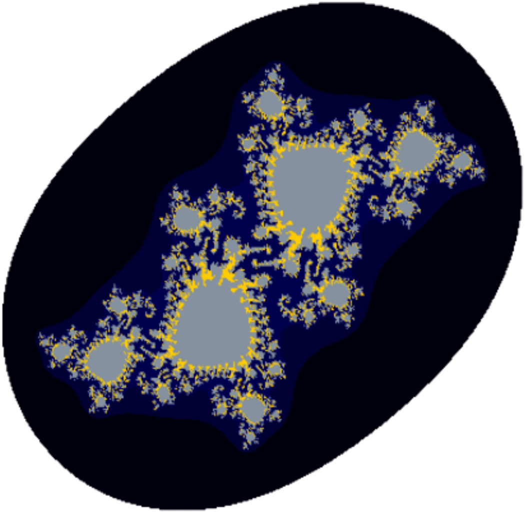
- CoCalc and Google Colaboratory for programming, LaTeX for written reflection
- Basics of Python during first 5–6 class periods
  - Sourced largely from *Automate the Boring Stuff with Python* [1]
- **Initial task:** pairs work at board to discover something about their project
  - More on this later

Stick to the plan



- The [4] pillars of IBL
  - Students deeply engaged in rich mathematical sense making.
  - Regular opportunities for students to collaborate with peers and instructors.
  - Instructor inquiry into student thinking.
  - Instructor focus on equity.

Equity in design

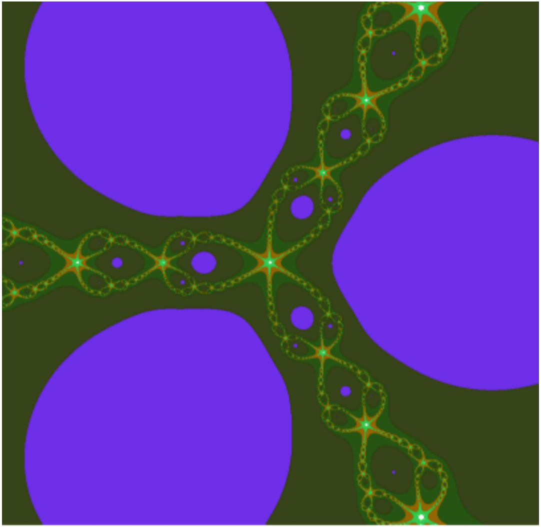


- How cheap can you go?
- Making sure everyone has equal access to the necessary technology
  - The first time I taught the course, I had students with very slow laptops.

Equity in design

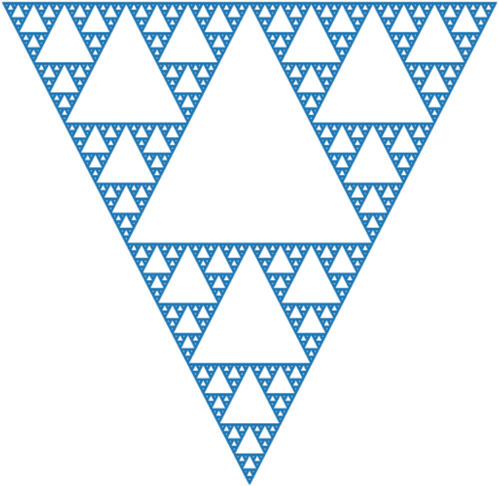
- How cheap can you go?
- Making sure everyone has equal access to the necessary technology
  - The first time I taught the course, I had students with very slow laptops.
- Lucky to have a room with large monitors that students could collaborate on

Instructor inquiry



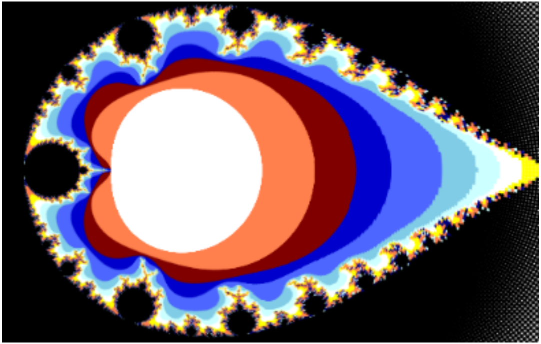
- In my opinion the hardest part for the instructor.
- Maintain students' motivation: "The first exercise is the hardest."
- Do the exercises beforehand!
    - Multiple ways to solve a problem.

Opportunities for collaboration



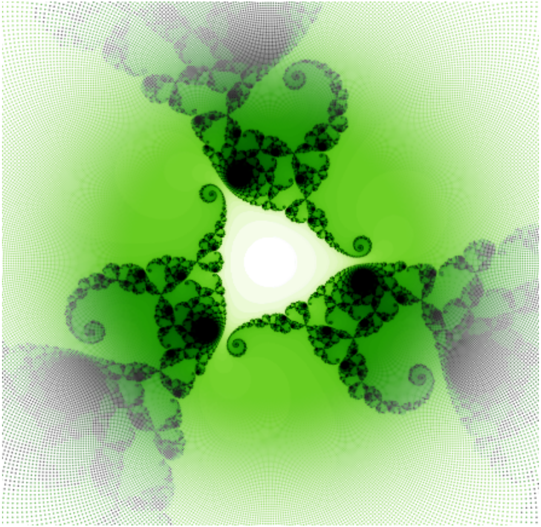
- Make sure programming duties are shared
  - In my class, students traded main programming duties and the other would takes notes for the journal.
- Opportunity for the instructor to get involved in the project
  - Ask for clarification; offer new suggestions
  - Encourage exploration beyond written exercises

Deep engagement



- Project should align with what's going on in class
  - Complex numbers, angles/exponential form, roots (of unity), etc.

Deep engagement

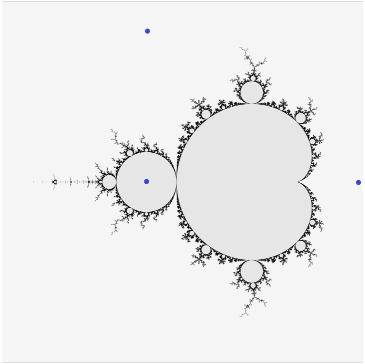


- Project should align with what's going on in class
  - Complex numbers, angles/exponential form, roots (of unity), etc.
- Initial tasks
  - Make students work with their hands without programming
  - Students need to believe that programming would be able to solve their problems more quickly and efficiently

Initial tasks

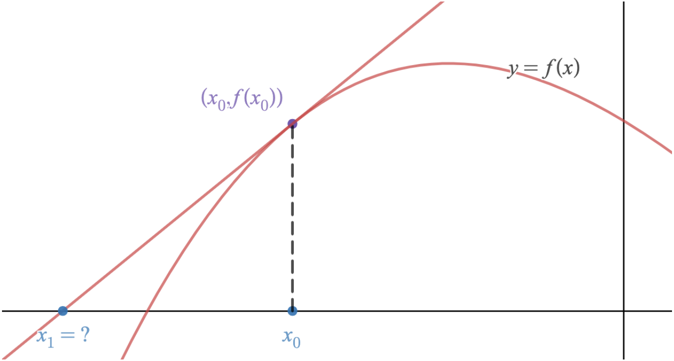
Mandelbrot set

- Initial task: iterate starting points in the plane to determine if they lie in the Mandelbrot set



- Exercise 1: generate an image of the Mandelbrot set with the assertion that a point is "in" if its first 100 iterates don't leave the disk of radius 2

- Initial task: derive Newton's method



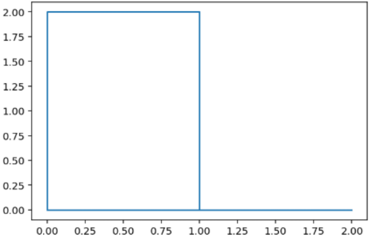
- Exercise 1: make the Newton fractal for $f(z) = z^3 - 1$, coloring the plane according to the *basins of attraction*

Initial tasks

L-systems

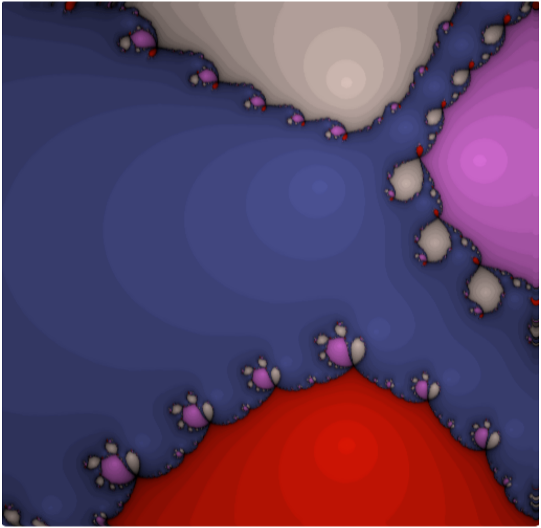
- Initial task: **Alphabet:** $F, +$; **Axiom:** F; **Rule(s):** $F \to F+$, $+ \to F$
  - Get to stage 5 and draw the result if F means "draw forward one unit" and $+$ means "change heading 90° counterclockwise"

$$F \to F+ \to F+F \to F+FF+ \to F+FF+F+F \to F+FF+F+FF+FF+$$



- Exercise 1: automate this process in Python (with more iterations you get a 1×1 grid in the first quadrant)

You can too!

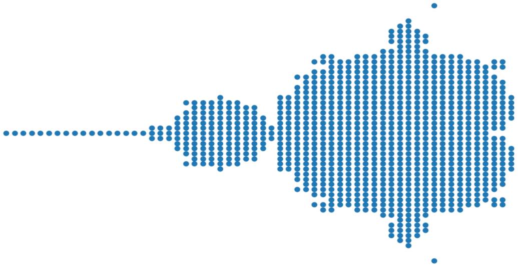


- What other classes lend themselves to programming modules?
  - Numerical analysis
  - Differential equations
  - Discrete math
  - Probability
  - Statistics
  - Applied mathematics
  - PICMath
  - Others?
- January term/May-mester: split time roughly 50/50
- Long semester: module in course

Lessons

- "Coverage vs. depth"
- Rewarding to see students discover for themselves
- Choice of programming language
  - Processing or JavaScript (p5.js) for animation and interactivity
  - Turtle graphics for L-systems

Acknowledgements



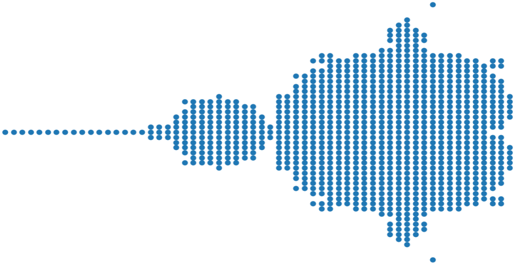
Thanks to:

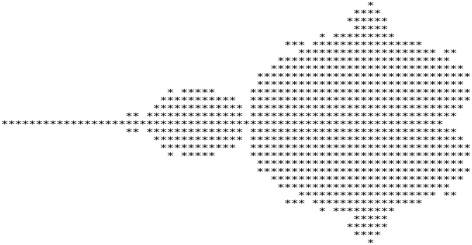
- IBLers everywhere
- Students of the class for letting me use their images
- You!

Contact:

joseph.barrera@converse.edu

Acknowledgements





Thanks to:

- IBLers everywhere
- Students of the class for letting me use their images
- You!

Contact:

joseph.barrera@converse.edu

References

[1] A. Sweigart. *Automate the Boring Stuff with Python, 2nd Edition*, No Starch Press, 2019.

[2] G. Cain. *Complex Analysis*, self-published, 1999, `https://cain.math.gatech.edu/winter99/complex.html`. Accessed 22 February 2023.

[3] R. Spindler. "A First Course in Undergraduate Complex Analysis." *Journal of Inquiry-Based Learning in Mathematics*, 2009, `http://jiblm.org/downloads/dlitem.php?id=72&category=jiblmjournal`. Accessed 22 February 2023.

[4] "The Pillars of Good Teaching." *The Academy of Inquiry Based Learning*, `http://www.inquirybasedlearning.org/`. Accessed 22 February 2023.

[5] J. W. Brown, R. V. Churchill. *Complex Variables and Applications, 8th Edition*, McGraw-Hill, 2008.