Research Outline

Ryan Greenup & James Guerra

August 24, 2020

Contents

```
code /home/ryan/Dropbox/Studies/QuantProject/Current/Python-Quant/ & disown
```

Here's what I gatthered from the week 3 slides

§ 1 Give a brief Sketch of the project

¶ 1.1 Topic / Context

We are interested in the theory of problem solving, but in particular the different approaches that can be taken to attacking a problem.

Essentially this boils down to looking at how a computer scientist and mathematician attack a problem, although originally I thought there was no difference, after seeing the odd way Roozbeh attacks problems I see there is a big difference.

¶ 1.2 Motivation

¶ 1.3 Basic Ideas

- Look at FOSS CAS Systems
  - Python (Sympy)
  - Julia
    - Sympy integration
    - symEngine
    - Reduce.jl
    - Symata.jl
- Maybe look at interactive sessions:
  - Like Jupyter
  - Hydrogen
  - TeXmacs
  - org-mode?

After getting an overview of SymPy let's look at problems that are interesting (chaos, morphogenesis and order from disarray etc.)

¶ 1.4 Where are the Mathematics

- Trying to look at the algorithms underlying functions in Python/Sympy and other Computer algebra tools such as Maxima, Maple, Mathematica, Sage, GAP and Xcas/Giac, Yacas, Symata.jl, Reduce.jl, SymEngine.jl
  - For Example Recursive Relations
- Look at solving some problems related to chaos theory maybe
  - Mandelbrot and Julia Sets
- Look at solving some problems related to Fourier Transforms maybe

AVOID DETAILS, JUST SKETCH THE PROJECT OUT.

¶ 1.5 Don't Forget we need a talk

1.5.1 Slides In Org Mode

- Without Beamer
- With Beamer

§ 2 What we're looking for

- Would a reader know what the project is about?
- Would a reader become interested in the upcoming report?
- Is it brief but well prepared?
- Are the major parts or phases sketched out

§ 3 Download RevealJS

So first do `M-x package-install ox-reveal` then do `M-x load-library` and then look for `ox-reveal`

Download Reveal.js and put it in the directory as `./reveal.js`, you can do that with something like this:

```
# cd /home/ryan/Dropbox/Studies/2020Spring/QuantProject/Current/Python-Quant/Outline/
wget https://github.com/hakimel/reveal.js/archive/master.tar.gz
tar -xzvf master.tar.gz && rm master.tar.gz
mv reveal.js-master reveal.js
```

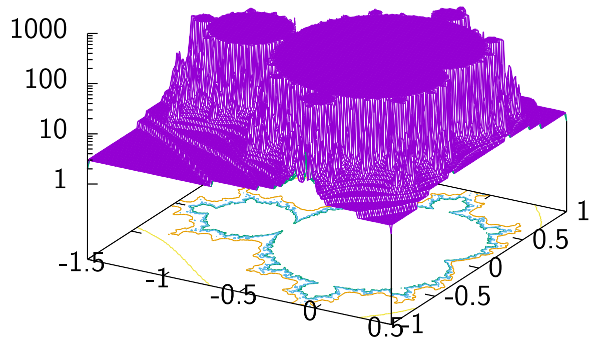
Then just do `C-c e e R R` to export with RevealJS as opposed to PHP you won't need a fancy server, just open it in the browser.

§ 4 GNU Plot

limit of recursion is 250

```
complex(x,y) = x*{1,0}+y*{0,1}
mandel(x,y,z,n) = (abs(z)>2.0 || n>=200) ? \
                      n : mandel(x,y,z*z+complex(x,y),n+1)

set xrange [-1.5:0.5]
set yrange [-1:1]
set logscale z
set isosample 200
set hidden3d
set contour
splot mandel(x,y,{0,0},0) notitle
```

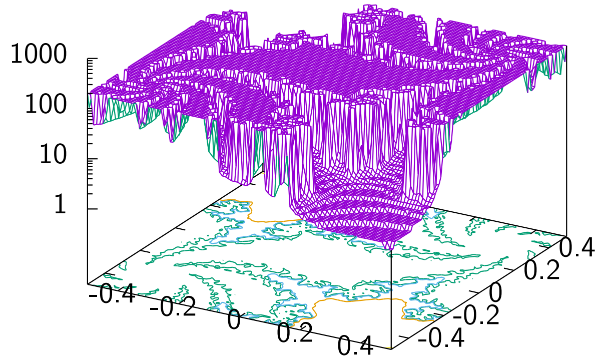


reference for image

,#+begin$_{\text{src}}$ gnuplot

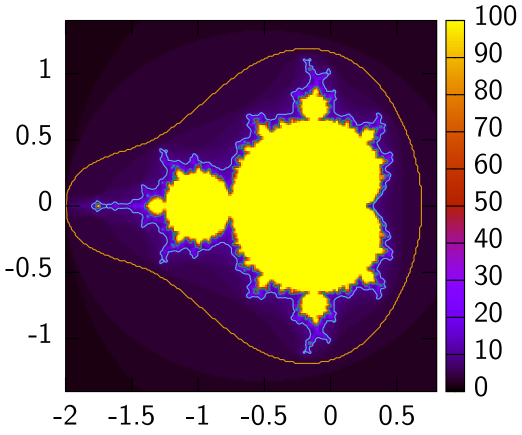
```
complex(x,y) = x*{1,0}+y*{0,1}
mandel(x,y,z,n) = (abs(z)>2.0 || n>=200) ? \
                      n : mandel(x,y,z*z+complex(x,y),n+1)

set xrange [-0.5:0.5]
set yrange [-0.5:0.5]
set logscale z
set isosample 100
set hidden3d
set contour
a= -0.37
b= -0.612
splot mandel(a,b,complex(x,y),0) notitle
```



reference

```
rmax = 2
nmax = 100
complex (x, y) = x * {1, 0} + y * {0, 1}
mandelbrot (z, z0, n) = n == nmax || abs (z) > rmax ? n : mandelbrot (z ** 2 + z0, z0, n + 1)
set samples 200
set isosamples 200
set pm3d map
set size square
splot [-2 : .8] [-1.4 : 1.4] mandelbrot (complex (0, 0), complex (x, y), 0) notitle
```


1
0.5
0
-0.5
-1
-2
-1.5
-1
-0.5
0
0.5
100
90
80
70
60
50
40
30
20
10
0