UNIVERSIDADE FEDERAL DO PARANÁ
LICENCIATURA EM MATEMÁTICA

CMM102 - Tópicos de Matemática 2: Fractais
Professora Elizabeth Wegner Karas
Trabalho em grupos

Curitiba, novembro de 2019.

Sumário

Capítulo 1

Conjuntos de Julia e Mandelbrot

Amanda Cristina Foetsch
Bianca Elena Wiltuschnig
Marcel Thadeu de Abreu e Souza
Matheus Daniel Galvão de Melo

1.1 Contextualização histórica

1879 - Cayley trabalhou em um precursor dos fractais modernos.

Os formatos descobertos pela primeira onda de teóricos dos fractais incluíam curvas enrugadas e "curvas monstro", que tinham anteriormente sido desconsideradas por serem consideradas exemplos patológicos de curvas. O estudo dos fractais, que eram muito irregulares para serem descritos usando geometria euclidiana padrão, foi guardado e esquecido por um tempo.

1904 - Von Koch cria sua curva do floco de neve.

A curva do floco de neve, ou floco de neve de Koch, foi desenvolvida pelo matemático sueco Niels von Koch (1870 - 1924). O floco de neve de Koch é um exemplo de um fractal, um dos primeiros a serem definidos.

1915 - Sierpinski cria o triângulo de Sierpinski.

O triângulo de Sierpinski foi descrito pela primeira vez pelo matemático polonês Waclaw Sierpinski (1882 - 1969), na forma de uma curva matematicamente definida em vez de uma forma geométrica, também um exemplo de fractal.

1918 - Hausdorff intruduz o conceito de dimensão fracionária.

O modo pelo qual Felix Hausdorff examinou a dimensão tinha a ver com escala. Se uma linha é ampliada por um fator de 3, ela é três vezes mais comprida do que era antes. Como $3 = 3^1$ uma linha é dita ter dimensão 1. Se um quadrado sólido é ampliado por um fator de 3, sua área é 9 vezes o seu valor anterior de 3^2, então a dimensão é 2. Contudo, podemos não ter a mesma base, e assim a dimensão de D resulta em um número fracionário. Um modelo famoso de geometria em dimensão fracionária é o floco de neve de Koch, por exemplo, se a unidade básica da curva de Koch for ampliada em uma escala de 3, ela se torna 4 vezes mais longa do que era antes, assim D é tal que $4 = 3^D$, ou seja, $D = 1,262$. Por outro lado, por ser uma curva, o floco de neve de Koch teria dimensão 1. Com os fractais frequentemente ocorre que a dimensão de Hausdorff é maior que a dimensão comum. Posteriormente, Mandelbrot usa a dimensão de Hausdorff para caracterizar a definição de um fractal como sendo um conjunto de pontos cujo valor de D não é um número inteiro, dimensões fracionárias se tornaram a principal propriedade dos fractais.

1919 - Julia e Fatou investigam estruturas fractais no plano complexo.

Ambos os matemáticos trabalharam com estruturas parecidas com os fractais em planos complexos de forma independente nos anos que se seguiram à Primeira Guerra Mundial. As curvas por eles estudadas não eram chamadas de fractais ainda e elas não possuíam forma definida, devida à ausência de equipamento tecnológico que possibilitasse a visualização de seus formatos.

1975 - Mandelbrot introduz o termo fractal.

É apanas na segunda metade do século XX que acontece a invenção e o desenvolvimento dos computadores, tornando possível a visualização dos fractais que eram apenas teoria até então. Os computadores tornaram possíveis a produção de figuras a partir de dados, mostrando a beleza dos fractais e tornando mais acessível o estudo de suas propriedades. O matemático polonês Benoît Mandelbrot (1924 - 2010) juntou os exemplos anteriores de fractais, deu a eles o nome oficial de fractal e definiu suas condições, dentre elas que fractais geralmente têm uma dimensão de Hausdorff diferente de sua dimensão topológica real. Mandelbrot explorou também a presença dessas estruturas fractais tanto no mundo natural, quanto nos sistemas artificiais como a economia, e determinou que eles são um modelo muito comum, encontrados mais frequentemente do que simples estruturas da geometria euclidiana. Enquanto que a geometria euclidiana trata de suavidade, o que raramente se encontra na natureza, os fractais podem expressar a qualidade rude do universo real. Há muitas formas quase fractais na natureza, incluindo flocos de neve, árvores, galáxias, ramificações dos vasos sanguíneos, nuvens, montanhas, o formato dos litorais, etc. Assim, o termo fractal foi usado também para descrever qualquer forma que tenha uma estrutura intrincada, independentemente do quanto a ampliemos. Examinando a história, Mandelbrot descobriu que matemáticos como Henri Poincaré e Arthur Cayley tiveram breves lampejos da ideia dos fractais cem anos antes dele, porém eles não tinham o poder computacional para investigar as questões mais a fundo. Os trabalhos de Gaston Julia e Pierre Fatou também foram ressuscitados com a popularização do estudo dessas estruturas tão interessantes. Em 1979, Mandelbrot descobre e descreve matematicamente o conjunto de Mandelbrot, um exemplo de fractal muito conhecido nos dias de hoje.

1.2 Conjunto de Julia

Consideremos uma função do tipo $f_c(z) = z^2 + c$, em $\mathbb{C}$. O Conjunto de Julia, que depende do $c = p + qi$ da função, é o conjunto fechado que é fronteira tanto da bacia de confinamento quanto da bacia de atração do infinito, e também compreende o fecho do conjunto dos pontos periódicos repulsores da função f_c. O Conjunto de Julia é um conjunto instável pois qualquer vizinhança de um ponto desse conjunto contém pontos com órbitas de comportamento diferente. Esse conjunto também é por si só um exemplo de fractal, pois ele assume a forma intricada de um fractal na maioria dos casos (exceto para $c = 0$ e $c = -2$).

1.3 Conjunto de Mandelbrot

Consideramos uma família de funções $f_c(z) = z^2 + c$, que tem parâmetro $c = p + qi$, e calculamos as órbitas da origem para cada valor de c. Assim, o conjunto dos pontos c para os quais as órbitas da origem de f_c são confinadas é o Conjunto de Mandelbrot, que podemos expressar na notação de conjunto como $M = \{c \in \mathbb{C}; |f_c^n(0)| \leq 2\}$. Há uma correspondência morfológica entre cada ponto c do Conjunto de Mandelbrot com o Conjunto de Julia de f_c, o formato do Conjunto de Julia com o c correspondente muda de acordo com a região na qual c está no Conjunto de Mandelbrot. Além disso, ontos que pertencem ao conjunto de Mandelbrot geram Conjuntos de Julia conexos e pontos no exterior do Conjunto de Mandelbrot geram Conjuntos de Julia desconexos (poeiras de Cantor).

1.4 Geogebra

O Geogebra é uma ferramenta prática para ser usada em sala de aula. É um software de matemática que reuni Geometria, Álgebra, Cálculo, Probabilidade e Estatística. Vamos ver agora uma maneira de plotar o conjunto de Mandelbrot e de Julia. Utilizamos o Geogebra Clássico 5 versão 5.0.564.0-d (22 October 2019) disponível para download no site *www.geogebra.org* .

1.4.1 Conjunto de Mandelbrot

Vamos ver agora uma forma de saber quais do valores da Janela Gráfica do Geogebra pertecem ou não ao Conjunto de Mandelbrot, de forma aproximada. Primeiro, no Geogebra, vamos exibir a planilha de cálculo do Geogebra. Devemos clicar no menu Exibir e escolher a opção Planilha, como na Figura **??**.

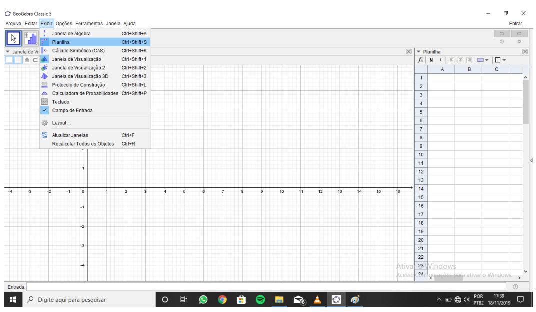


Figura 1.1: Planilha

Na Entrada vamos digitar um número complexo qualquer, com parte imaginária e real diferente de zero, apenas para poder obter um valor numérico (diferente de zero) em ambas as coordenadas. Escolhemos o número $1 + i$. Ao digitar o mesmo número na primeira célula $A1$ (digitamos $= 1 + i$), obtemos o valor na tela. Se arrastarmos o valor na Janela Gráfica, o número da célula muda. O número da célula $A1$ corresponde a constante c da equação $f_c(x) = x^2 + c$. Vamos calcular a órbita do 0. Temos que

$$A1 = f_c(0) = 0^2 + c = c$$

$$A2 = f_c^2(0) = f_c(f_c(0)) = f_c(c) = c^2 + c = (A1)^2 + A1$$

$$A3 = f_c^3(0) = f_c(f_c^2(0)) = f_c(c^2+c) = (c^2+c)^2 + c = (A2)^2 + A1$$
$$A4 = f_c^4(0) = f_c(f_c^3(0)) = f_c((c^2+c)^2+c) = (c^2+c)^2 + c^2 + c = (A3)^2 + A1$$
$$\vdots$$
$$A25 = f_c^{25}(0) = (A24)^2 + A1$$

Na célula $A2$, então, vamos digitar "$= A2 + A\$1$". O símbolo \$ tem a função de deixar o valor da célula $A1$ para cálculos em outras células. Obtemos os outros elementos da órbita do zero copiando o conteúdo de $A2$ nas células $A3$ até $A25$. Quando o número da célula é muito grande, aparece o simbolo $?+?i$ na célula correspondente.

Queremos saber apenas se o número $A1$ pertence ao conjunto de Mandelbrot, os demais podemos selecionar, clicar com o botão direito e clicar na opção Esconder Objeto.

Para saber se o número pertence ou não no conjunto de Mandelbrot, vamos usar uma função $test(x) = (x == x)$, que vai testar se a variável é o valor de fato ou apenas uma aproximação do real valor. Vamos testar a última célula ($A25$), e, caso o teste seja verdadeiro, o valor $A1$ pode estar contido no conjunto de Mandelbrot, caso seja falso, o número não esta contido no conjunto. Colocamos a função $test(x) = (x == x)$ na entrada, como na Figura **??**.

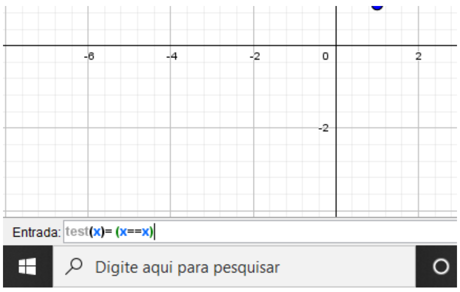


Figura 1.2: Função $test(x)$

Portanto

- Quando o ponto vai para o infinito, vamos colorir de vermelho;
- Quando o ponto fica no conjunto, vamos colorir de preto.

Para fazer isso, clicamos com o botão direito na primeira célula, $A1$, e clicamos em Propriedades, abrindo a janela da Figura **??** abaixo e clicamos em Avançado.

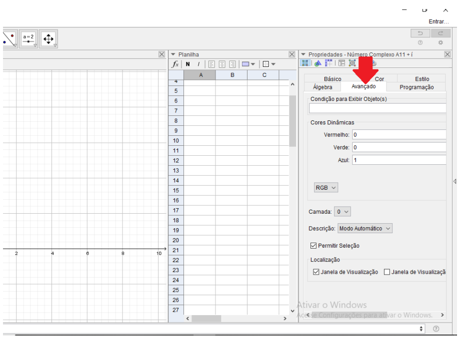


Figura 1.3: Propriedades

Abrindo a janela, obtemos uma página como da Figura ??. Como referido, na janela “Vermelho” colocamos a ordem “Se[test(A25), 0, 1]”, e nas janelas “Verde” e “Azul”, colocamos 0. Isso significa que, caso seja verdadeiro o teste em $A25$, o número da célula $A1$ terá cor [0, 0, 0] (preto), e caso seja falso o teste em $A25$, o número da célula $A1$ terá cor [1, 0, 0] (vermelho).

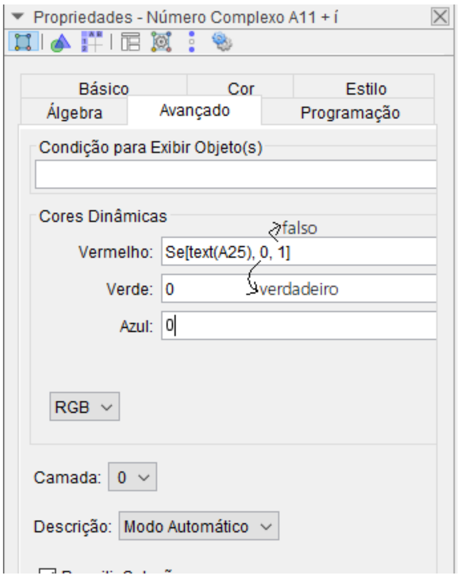


Figura 1.4: Avançado

Com o botão direito, clicamos no número da célula $A1$ visível na Janela Gráfica e selecionamos a opção Habilitar Rastro. Selecionando a opção do Mouse na barra inicial, clicamos no valor de $A1$ e arrastamos, obtendo a Figura ??.

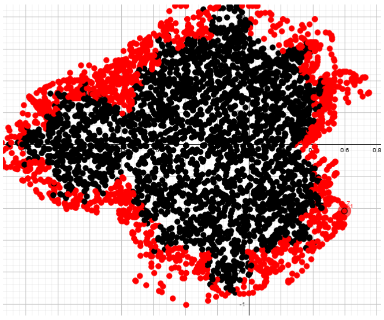
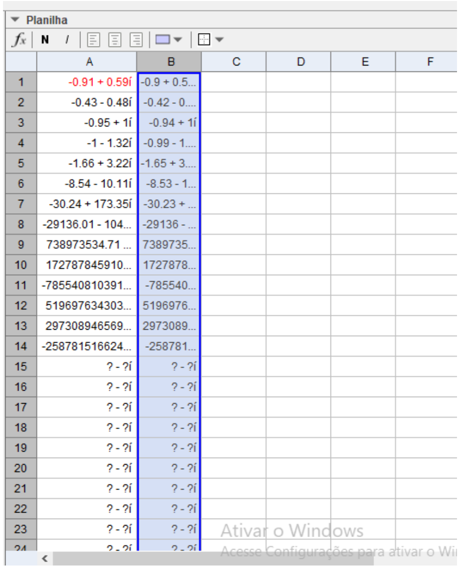


Figura 1.5: Avançado

Seria legal ter um pincel maior, não? Para isso, clicamos na célula $B1$, e colocamos nela “$= A1 + 0.01$”. Selecionamos as células $A2$ até $A25$ e colamos na célula $B2$, obtendo a planilha como na Figura ??.

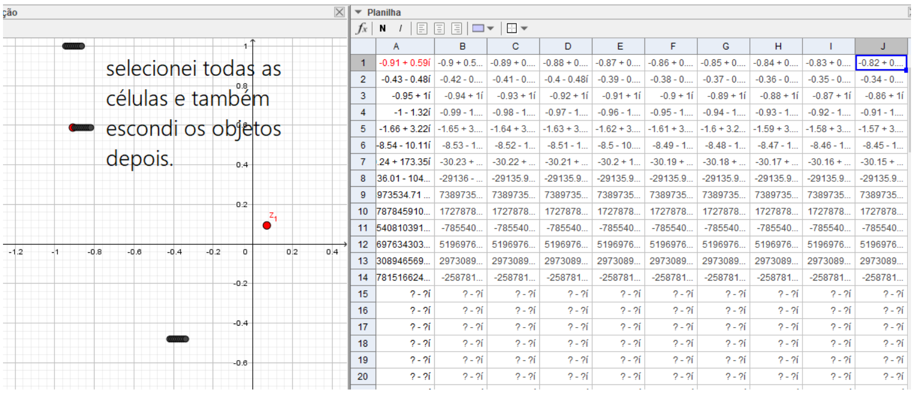


Planilha

	A	B	C	D	E	F
1	-0.91 + 0.59í	-0.9 + 0.5...				
2	-0.43 - 0.48í	-0.42 - 0....				
3	-0.95 + 1í	-0.94 + 1í				
4	-1 - 1.32í	-0.99 - 1....				
5	-1.66 + 3.22í	-1.65 + 3....				
6	-8.54 - 10.11í	-8.53 - 1...				
7	-30.24 + 173.35í	-30.23 + ...				
8	-29136.01 - 104...	-29136 - ...				
9	738973534.71 ...	7389735...				
10	172787845910...	1727878...				
11	-785540810391...	-785540...				
12	519697634303...	5196976...				
13	297308946569...	2973089...				
14	-258781516624...	-258781...				
15	? - ?í	? - ?í				
16	? - ?í	? - ?í				
17	? - ?í	? - ?í				
18	? - ?í	? - ?í				
19	? - ?í	? - ?í				
20	? - ?í	? - ?í				
21	? - ?í	? - ?í				
22	? - ?í	? - ?í				
23	? - ?í	? - ?í				

Figura 1.6: Planilha B

Agora, copiamos a coluna B toda e colamos nas colunas seguintes, até a coluna J, como na Figura **??**.



Planilha

	A	B	C	D	E	F	G	H	I	J
1	-0.91 + 0.59í	-0.9 + 0.5...	-0.89 + 0....	-0.88 + 0....	-0.87 + 0....	-0.86 + 0....	-0.85 + 0....	-0.84 + 0....	-0.83 + 0....	-0.82 + 0....
2	-0.43 - 0.48í	-0.42 - 0....	-0.41 - 0....	-0.4 - 0.48í	-0.39 - 0....	-0.38 - 0....	-0.37 - 0....	-0.36 - 0....	-0.35 - 0....	-0.34 - 0....
3	-0.95 + 1í	-0.94 + 1í	-0.93 + 1í	-0.92 + 1í	-0.91 + 1í	-0.9 + 1í	-0.89 + 1í	-0.88 + 1í	-0.87 + 1í	-0.86 + 1í
4	-1 - 1.32í	-0.99 - 1....	-0.98 - 1....	-0.97 - 1....	-0.96 - 1....	-0.95 - 1....	-0.94 - 1....	-0.93 - 1....	-0.92 - 1....	-0.91 - 1....
5	-1.66 + 3.22í	-1.65 + 3....	-1.64 + 3....	-1.63 + 3....	-1.62 + 3....	-1.61 + 3....	-1.6 + 3.2...	-1.59 + 3....	-1.58 + 3....	-1.57 + 3....
6	-8.54 - 10.11í	-8.53 - 1...	-8.52 - 1...	-8.51 - 1...	-8.5 - 10....	-8.49 - 1...	-8.48 - 1...	-8.47 - 1...	-8.46 - 1...	-8.45 - 1...
7	.24 + 173.35í	-30.23 + ...	-30.22 + ...	-30.21 + ...	-30.2 + 1...	-30.19 + ...	-30.18 + ...	-30.17 + ...	-30.16 + ...	-30.15 + ...
8	36.01 - 104...	-29136 - ...	-29135.9...	-29135.9...	-29135.9...	-29135.9...	-29135.9...	-29135.9...	-29135.9...	-29135.9...
9	973534.71 ...	7389735...	7389735...	7389735...	7389735...	7389735...	7389735...	7389735...	7389735...	7389735...
10	787845910...	1727878...	1727878...	1727878...	1727878...	1727878...	1727878...	1727878...	1727878...	1727878...
11	540810391...	-785540...	-785540...	-785540...	-785540...	-785540...	-785540...	-785540...	-785540...	-785540...
12	697634303...	5196976...	5196976...	5196976...	5196976...	5196976...	5196976...	5196976...	5196976...	5196976...
13	308946569...	2973089...	2973089...	2973089...	2973089...	2973089...	2973089...	2973089...	2973089...	2973089...
14	781516624...	-258781...	-258781...	-258781...	-258781...	-258781...	-258781...	-258781...	-258781...	-258781...
15	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í
16	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í
17	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í
18	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í
19	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í
20	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í	? - ?í

Figura 1.7: Planilha B

Deixamos apenas a linha A com os objetos visíveis, os demais selecionamos e escondemos os objetos. Selecionamos a linha A e colocamos os comandos de cores, igual fizemos na Figura **??**, lembrando que na célula $B1$ colocamos na parte "Vermelho" "Se[test(B25), 0, 1]", os demais colocamos 0, na coluna $C1$ colocamos na parte "Vermelho" "Se[test(C25), 0, 1]", os demais colocamos 0, e assim sucessivamente.

Selecionando a opção do Mouse na barra inicial, clicamos no valor de $A1$ na Janela Gráfica e arrastamos ele por toda parte onde sabemos que existe o conjunto de Mandelbrot, ou seja, para parte real entre $|x| = 2$ e parte imaginária $|y| = 1$. Obtemos a Figura **??**.

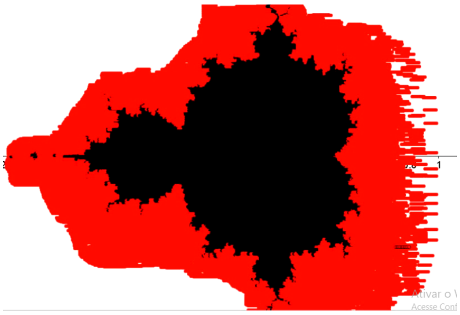


Figura 1.8: Conjunto de Mandelbrot

Caso queira obter maior precisão, pode-se aumentar o número de iterações para mais de 25. Um pincel maior também pode ser feito, apenas aumentando o número de colunas da planilha.

1.4.2 Conjunto de Julia

O primeiro passo é digitar na entrada o ponto c escolhido da função geradora do Conjunto de Julia $f_c(z) = z^2 + c$, conforme a Figura ??.

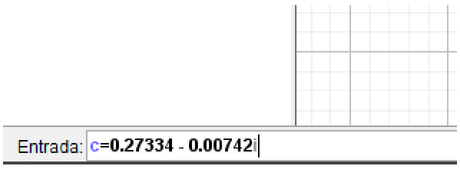


Figura 1.9: Escolha do ponto c

Para o exemplo que faremos aqui, tomaremos $c = 0,27334 - 0,00742i$, no entanto, a escolha para o ponto c é livre, pode ser qualquer ponto no plano complexo.

Depois de tomado c, é indicado que o oculte, isto pode ser feito indo na janela de álgebra e desmarcando a opção "Exibir Objeto", conforme na Figura ??.

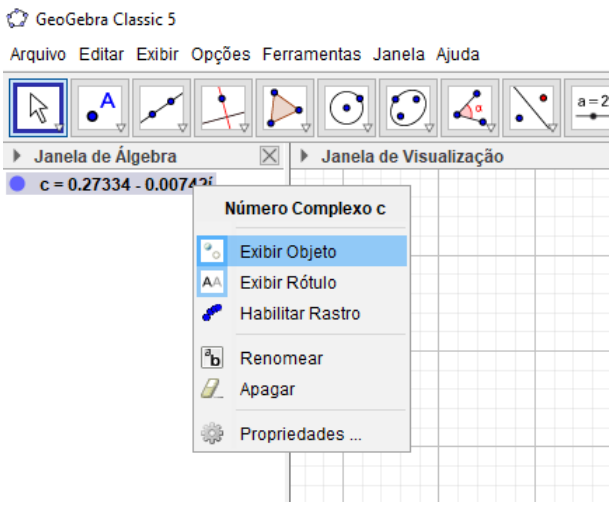


Figura 1.10: Escolha do ponto c

Feito isto, agora para o próximo passo devemos abrir a planilha. Para isso, clicamos em Exibir e marcamos a opção Planilha, conforme a Figura ??.

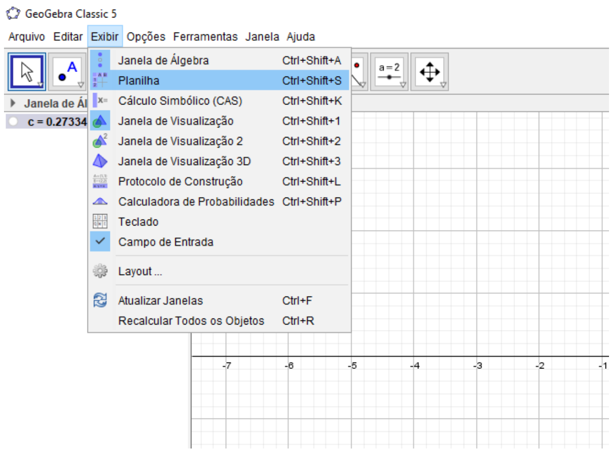


Figura 1.11: Abrindo a planilha

Agora, na coluna A linha 1, digitaremos um ponto do plano complexo qualquer que possa ser visualizado na Janela Gráfica, por exemplo, $1 + i$, conforme a Figura ??.

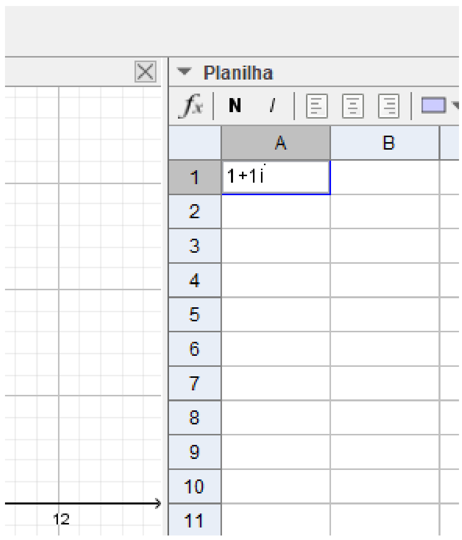


Figura 1.12: Digitando A1

Depois disso, calcularemos os pontos da órbita do número na célula $A1$, calcularemos os primeiros 25 elementos da órbita, no entanto, caso deseje obter maior precisão, pode-se aumentar o número de iterações para mais de 25.

Os elementos das próximas células serão justamente os próximos elementos da órbita do ponto da célula $A1$, do seguinte modo:

$$A1$$

$$A2 = f_c(A1) + c = (A1)^2 + c$$

$$A3 = f_c^2(A1) = f_c(f_c(A1)) = f_c(A2) = A2^2 + C$$

$$A4 = f_c^3(A1) = f_c(f_c^2(A1)) = f_c(f_c^2(A1)) = f_c(A2) = A^3 + c$$

$$\vdots$$

$$A25 = f_c^{25}(A1) = (A24)^2 + c$$

Portanto, na célula $A2$, digitaremos $A1^2 + c$, conforme a Figura ??.

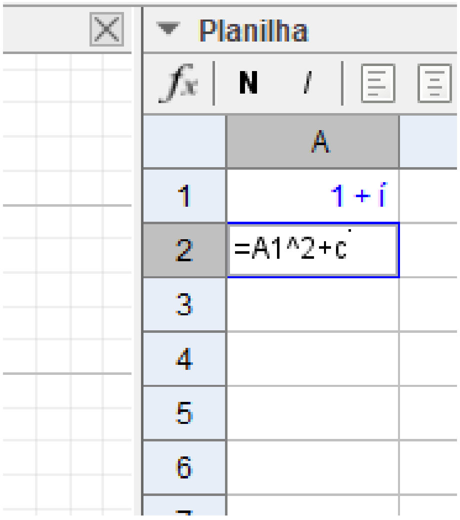


Figura 1.13: Digitando A2

Para as próximas células da Coluna A, podemos preenchê-las clicando com o botão direito na célula $A2$ e em seguida "Copiar". Após isso, para preenchermos as próximas células automaticamente, selecionaremos todas as próximas células até a 25 e clicaremos em "Colar", conforme a Figura ??.

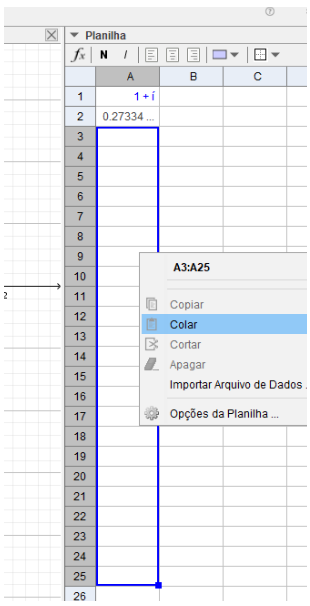


Figura 1.14: Coluna A

Os valores das próximas células serão, portanto, o valor da célula anterior ao quadrado somado com c, e serão as órbitas do ponto A1 em relação à $f_c(z)$. E a partir disso conseguimos desenhar o conjunto de Julia. Um ponto pertencerá ao conjunto de Julia se ele não tender ao infinito conforme a função $f_c(z) + c$ é iterada.

Como estamos trabalhando com 25 iterações, diremos que se na 25ª iteração o ponto tiver valor absoluto maior que 2, ele diverge e portanto pintaremos de vermelho, caso contrário, o ponto está no conjunto de Julia e pintaremos de preto. Com apenas 25 iterações não será um desenho muito preciso do conjunto de Julia, mas podemos aumentar o número de iterações simplesmente aumentando o número de células na coluna A. Para pintarmos deste modo, devemos habilitar o traço dinâmico no ponto $A1$. Para tal, clicamos com o botão direito na célula $A1$ e clicamos em propriedades, conforme a Figura ??.

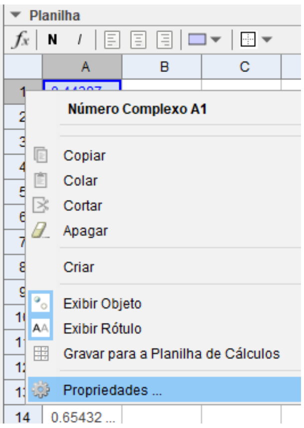


Figura 1.15: Propriedades

Em seguida, vamos em Avançado, e colocaremos o comando que diz para pintar de vermelho se o módulo de $A25$ for maior que 2, e pintar de preto caso contrário. Como estamos no sistema de cores RGB, deixando 0 nas outras cores e 1 em vermelho, pintaremos de vermelho, se deixarmos 0 em todas as cores, pintaremos de preto. Disto, segue o comando que pode ser escrito com: Se[abs(A25)<=2, 0, 1], conforme a Figura ??.

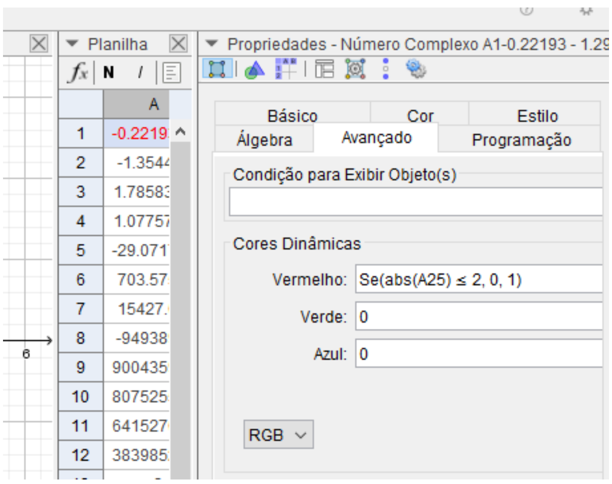


Figura 1.16: Avançado

Após isso, podemos habilitar o traço do ponto $A1$. Conforme $A1$ varia pelo plano, teremos uma aproximação do Conjunto de Julia. Para maior precisão, é necessário aumentar o número de iterações.

Para pintarmos o Conjunto de Julia, precisaremos ficar arrastando o ponto $A1$ para que seja calculado a órbita de diferentes pontos $A1$, e assim avalie quais pontos pertencem ao conjunto de Julia e quais não. No entanto, para facilitar o trabalho, podemos aumentar o tamanho do pincel de modo inteiramente análogo ao que foi feito com o conjunto de Mandelbrot. Na célula $B1$, digitamos $A1 + 0.01$, depois selecionamos todas as demais células da coluna A e colamos embaixo de $B1$, conforme a Figura **??**.

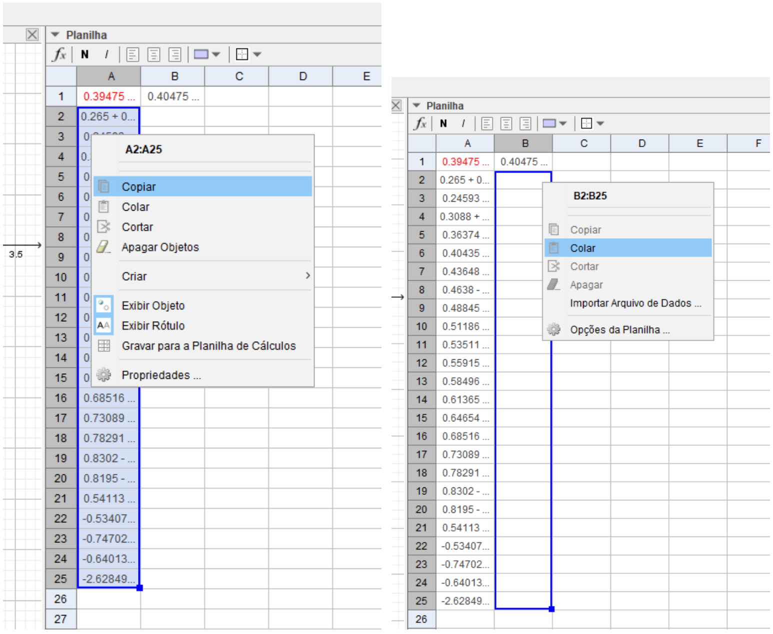


Figura 1.17: Planilha

Para aumentarmos o pincel, repetimos o mesmo processo, fazendo $C1 = B1+0.01$ e repetindo o processo de copiar as demais células de B nas outras células de C abaixo de $C1$. Depois podemos estender ainda mais o pincel para outras colunas repetindo o mesmo processo.

Segue o Conjunto de Julia esboçado pelo Geogebra para o ponto $c = 0,27334 - 0,00742i$ na Figura **??**.

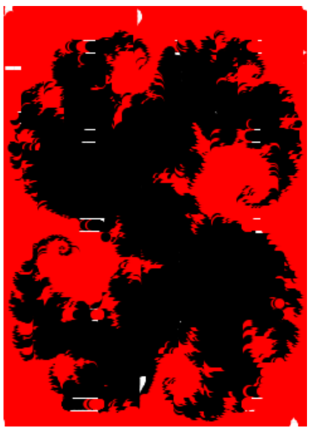


Figura 1.18: Conjunto de Julia de $c = 0,27334 - 0,00742i$

O mesmo fractal, mas feito com mais iterações e precisão pode ser visto na Figura ??.

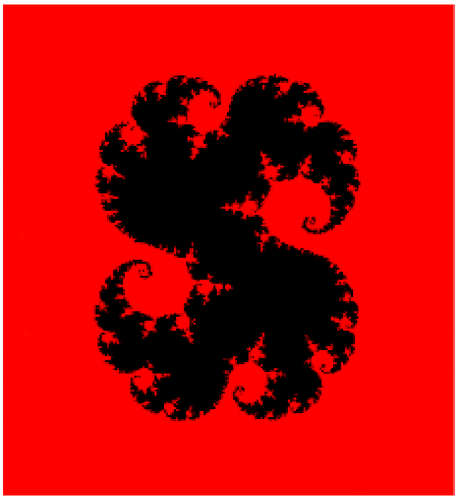


Figura 1.19: Conjunto de Julia de $c = 0,27334 - 0,00742i$ com mais iterações

Para calcularmos outros Conjuntos de Julia, basta trocar o valor do c inicial.

1.5 Explorador de Fractais

O Explorador de Fractais é um software criado utilizando o motor gráfico Unity, que tem o propósito de representar, em tempo real, os conjuntos de Julia e Mandelbrot da função

$$f : \mathbb{C} \to \mathbb{C}$$
$$f(z) = z^n + c.$$

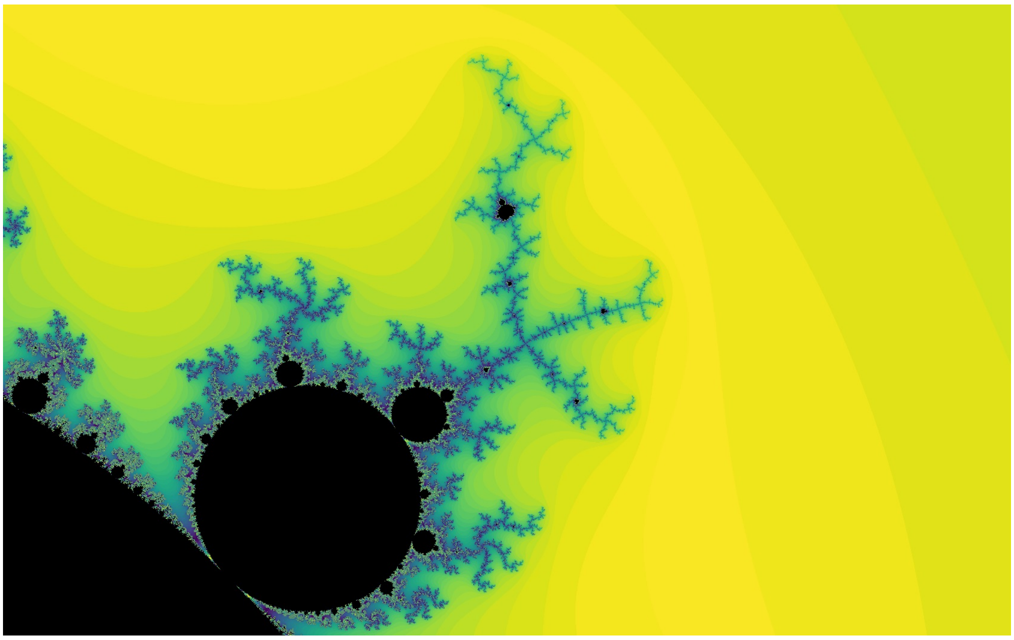


Figura 1.20: Conjunto de Mandelbrot gerado no Explorador de Fractais

1.5.1 Como utilizar o software?

O usuário pode alterar os seguintes valores para gerar os fractais:

- A potência de z, isto é, o valor de n em $f(z) = z^n + c$
- O valor de c em $f(z) = z^n + c$ (no caso do conjunto de Julia)
- O número máximo de iterações

Além disso, o usuário pode realizar os seguintes comandos para navegar pelos fractais:

- Mover
- Rotacionar
- Aproximar / afastar (zoom)
- Trocar a paleta de cores

1.5.2 Fractais gerados pelo software

Veja alguns fractais gerados pelo Explorador de Fractais.

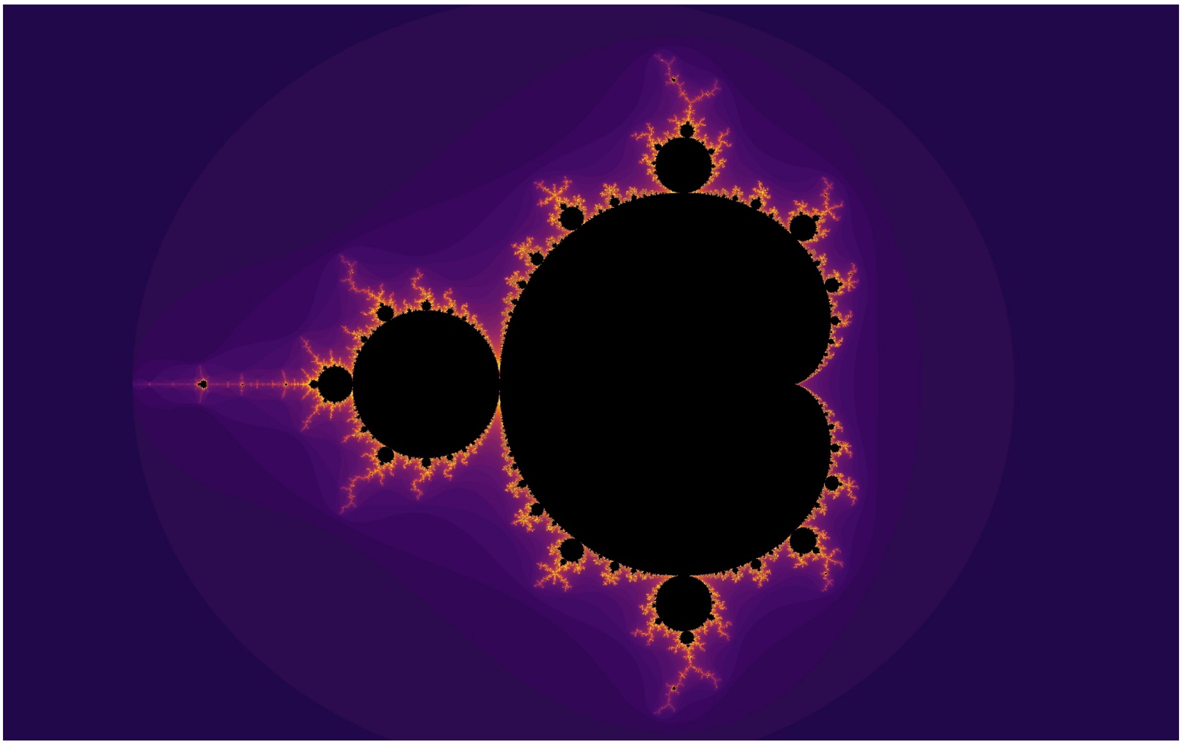


Figura 1.21: Conjunto de Mandelbrot de $f(z) = z^2 + c$

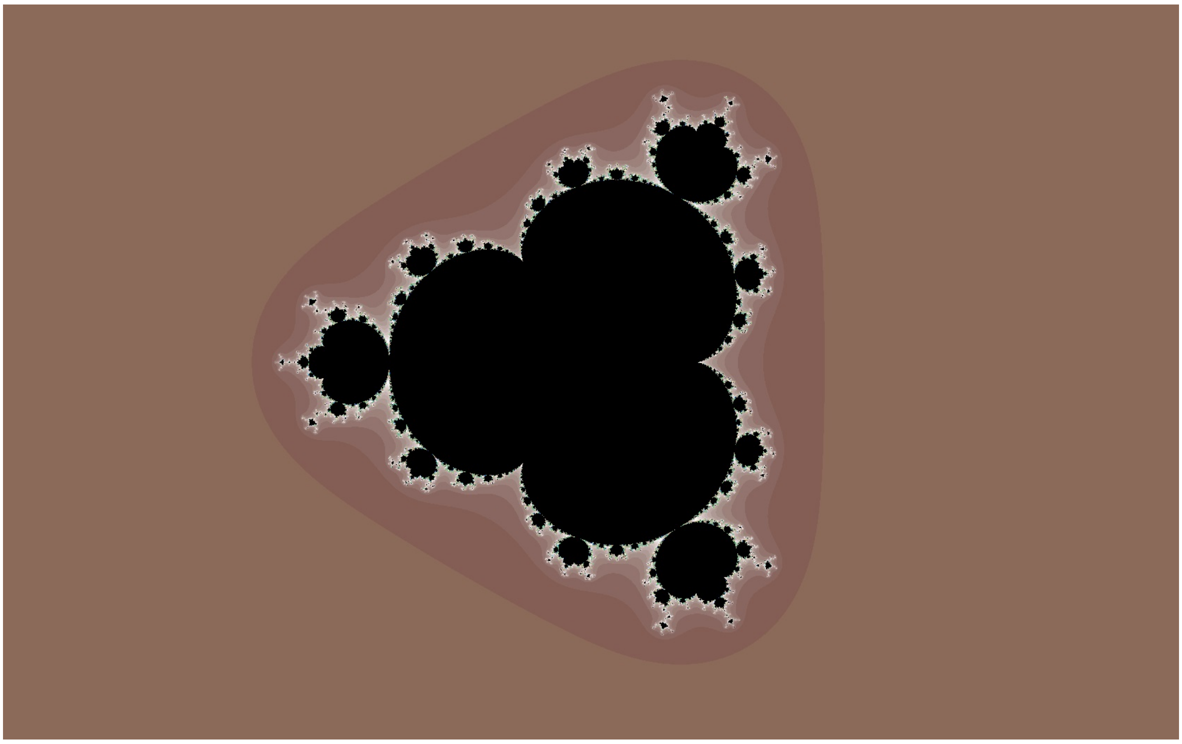


Figura 1.22: Conjunto de Mandelbrot de $f(z) = z^4 + c$

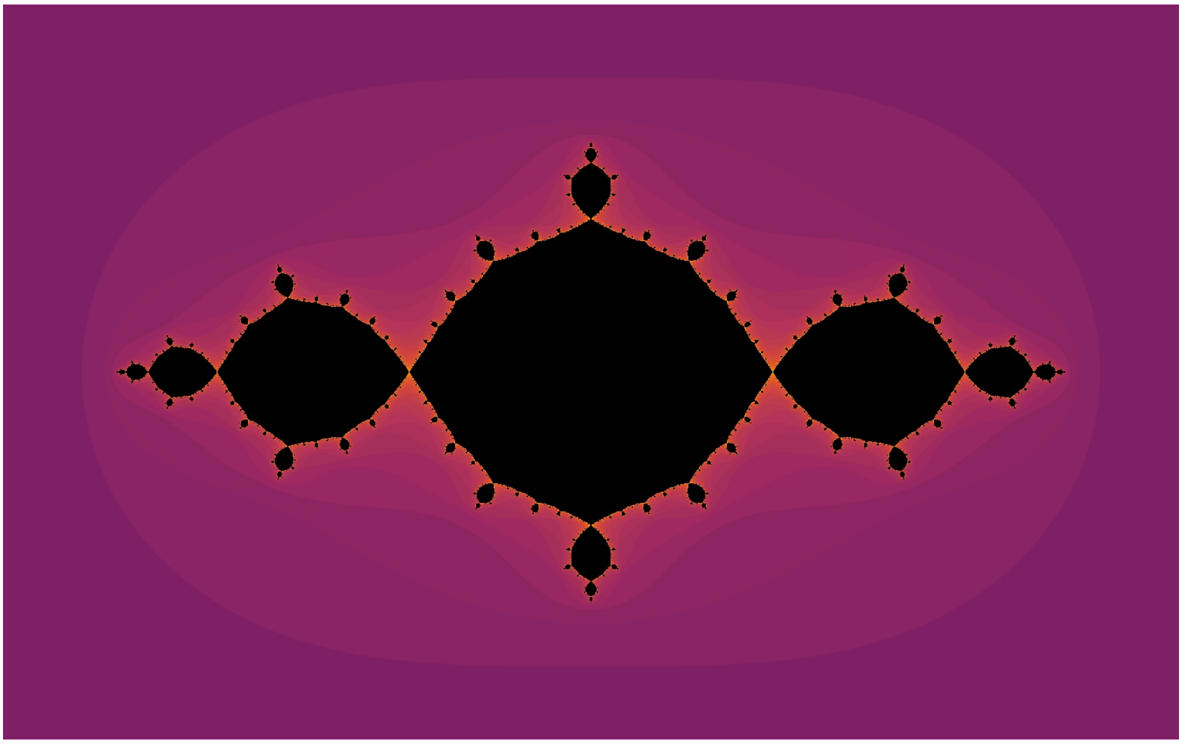


Figura 1.23: Conjunto de Julia de $f(z) = z^2 - 1$

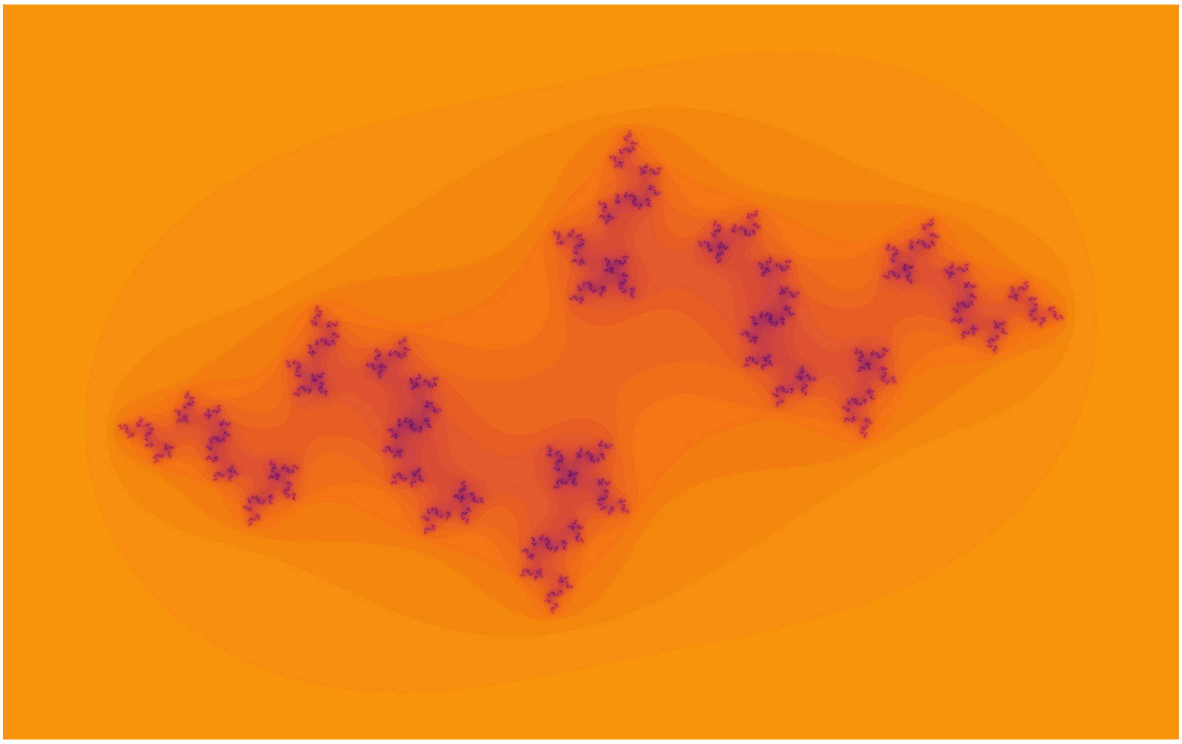


Figura 1.24: Conjunto de Julia de $f(z) = z^2 - 0,95 - 0,4i$

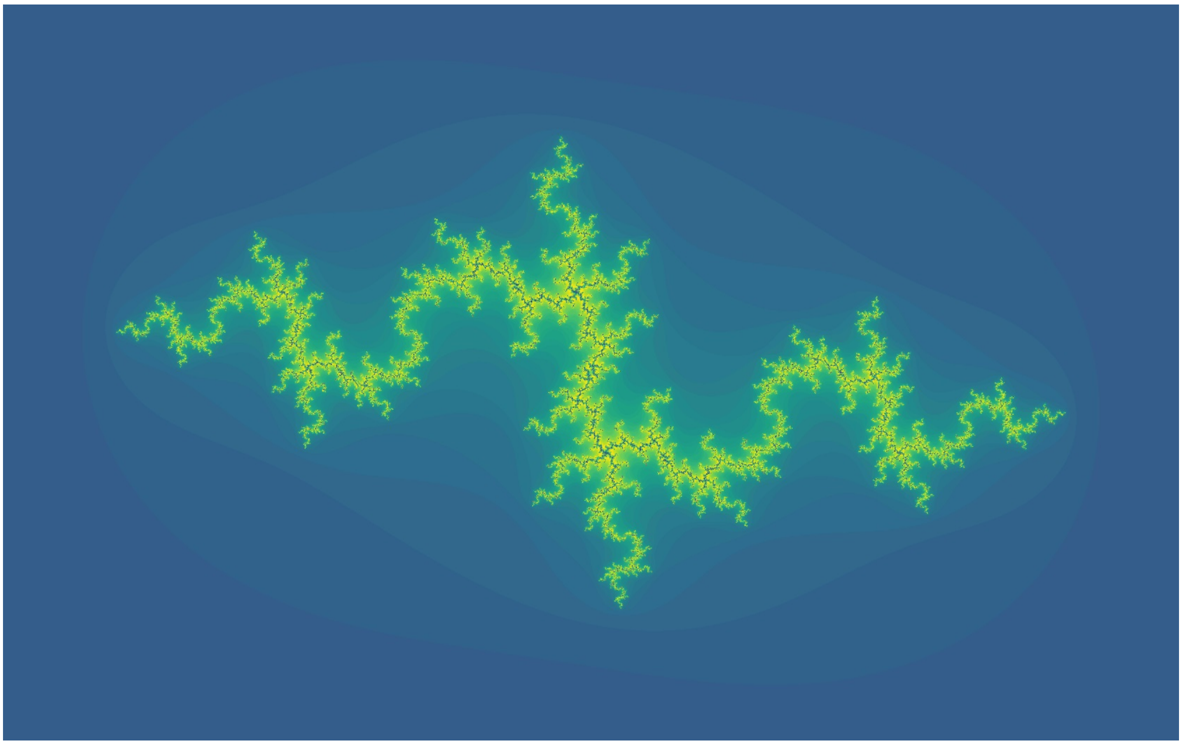


Figura 1.25: Conjunto de Julia de $f(z) = z^2 - 0,98 + 0,3i$

1.5.3 Por trás do software

A Unity é um motor de jogo que oferece aos usuários a capacidade de criar jogos e aplicações em tempo real. O Explorador de Fractais foi criado utilizando esse motor de jogo. Os scripts foram escritos na linguagem C# e os shaders em HLSL.

Veja a seguir os códigos-fonte dos shaders referentes aos conjuntos de Julia e Mandelbrot.

Conjunto de Julia

```
// Julia.shader
Shader "Explorer/Julia"
{
    Properties
    {
        // Propriedades do Shader
        _MainTex("Texture", 2D) = "white" {}
        _Color("Color Texture", 2D) = "white" {}
        _MaxIter("Max Iterations", Float) = 128
        _Power("Power", Int) = 2
        _Area("Area", Vector) = (0, 0, 4, 4)
        _Seed("Seed", Vector) = (0.0, 0.0, 0.0, 0.0)
        _Angle("Angle", range(-3.1415, 3.1415)) = 0
    }
    SubShader
    {
        Cull Off ZWrite Off ZTest Always

        Pass
        {
            CGPROGRAM
            #pragma vertex vert
            #pragma fragment frag

            #include "UnityCG.cginc"

            struct appdata
            {
                float4 vertex : POSITION;
                float2 uv : TEXCOORD0;
            };

            struct v2f
            {
                float2 uv : TEXCOORD0;
                float4 vertex : SV_POSITION;
            };

            v2f vert (appdata v)
            {
                v2f o;
                o.vertex = UnityObjectToClipPos(v.vertex);
                o.uv = v.uv;
                return o;
            }

            // Variáveis
```

```
            float4 _Area;
            float4 _Seed;
            sampler2D _MainTex;
            sampler2D _Color;
            float _MaxIter;
            float _Power;
            float _Angle;

            // Função para rotacionar fractal
            float2 rotate(float2 p, float2 pivot, float angle)
            {
                p -= pivot;
                p = float2(p.x * cos(angle) - p.y * sin(angle), p.x *
                    sin(angle) + p.y * cos(angle));
                p += pivot;

                return p;
            }

            // Função que gera o fractal
            fixed4 julia(float2 c, float2 z)
            {
                int iter;
                for (iter = 0; iter < _MaxIter; iter++) {
                    z = float2(pow((z.x * z.x + z.y * z.y), (_Power / 2)) *
                        cos(_Power * atan2(z.y, z.x)), pow((z.x * z.x + z.y *
                        z.y), (_Power / 2)) * sin(_Power * atan2(z.y, z.x))) + c;
                    if (length(z) > 2) break;
                }
                float4 color = 0;
                if (iter < _MaxIter) {
                    color = tex2D(_Color, float2( (0.01f * _MaxIter) * (
                        (float)iter / (float)_MaxIter ) + _Time.x, 0.0f));
                }
                return color;
            }

            fixed4 frag(v2f i) : SV_Target
            {
                float2 c = _Seed;
                float2 z = _Area.xy + (i.uv - 0.5)*_Area.zw;
                z = rotate(z, _Area.xy, _Angle);

                return julia(c, z);
            }
            ENDCG
        }
    }
}
```

Conjunto de Mandelbrot

```
// Mandelbrot.shader
Shader "Explorer/Mandelbrot"
{
    Properties
    {
        // Propriedades do Shader
        _MainTex("Texture", 2D) = "white" {}
        _Color("Color Texture", 2D) = "white" {}
        _MaxIter("Max Iterations", Float) = 128
        _Power("Power", Int) = 2
        _Area("Area", Vector) = (0, 0, 4, 4)
        _Angle("Angle", range(-3.1415, 3.1415)) = 0
    }
    SubShader
    {
        Cull Off ZWrite Off ZTest Always

        Pass
        {
            CGPROGRAM
            #pragma vertex vert
            #pragma fragment frag

            #include "UnityCG.cginc"

            struct appdata
            {
                float4 vertex : POSITION;
                float2 uv : TEXCOORD0;
            };

            struct v2f
            {
                float2 uv : TEXCOORD0;
                float4 vertex : SV_POSITION;
            };

            v2f vert (appdata v)
            {
                v2f o;
                o.vertex = UnityObjectToClipPos(v.vertex);
                o.uv = v.uv;
                return o;
            }

            // Variáveis
            float4 _Area;
            sampler2D _MainTex;
            sampler2D _Color;
            float _MaxIter;
```

```
            float _Power;
            float _Angle;

            // Função para rotacionar fractal
            float2 rotate(float2 p, float2 pivot, float angle)
            {
                p -= pivot;
                p = float2(p.x * cos(angle) - p.y * sin(angle), p.x *
                    sin(angle) + p.y * cos(angle));
                p += pivot;

                return p;
            }

            // Função que gera o fractal
            fixed4 mandelbrot(float2 c)
            {
                float2 z = c;
                int iter;
                for (iter = 0; iter < _MaxIter; iter++) {
                    z = float2(pow((z.x * z.x + z.y * z.y), (_Power / 2)) *
                        cos(_Power * atan2(z.y, z.x)), pow((z.x * z.x + z.y *
                        z.y), (_Power / 2)) * sin(_Power * atan2(z.y, z.x))) + c;
                    if (length(z) > 2) break;
                }
                float4 color = 0;
                if (iter < _MaxIter) {
                    color = tex2D(_Color, float2( (0.01f * _MaxIter) * (
                        (float)iter / (float)_MaxIter ) + _Time.x, 0.0f));
            }
                return color;
            }

            fixed4 frag(v2f i) : SV_Target
            {
                float2 c = _Area.xy + (i.uv - 0.5)*_Area.zw;
                c = rotate(c, _Area.xy, _Angle);

                return mandelbrot(c);
            }
            ENDCG
        }
    }
}
```