Project 2

Abraham Jacob Reines

May 5, 2022

1 Project 4: Chaos in Newton's Method

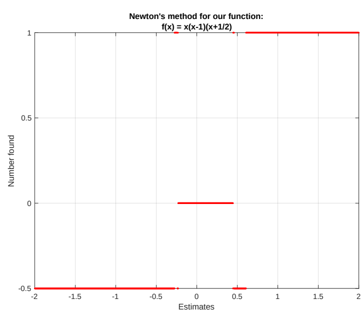
1.1 Applying Newton's Method for 'Part a'

The task is to apply Newton's Method to

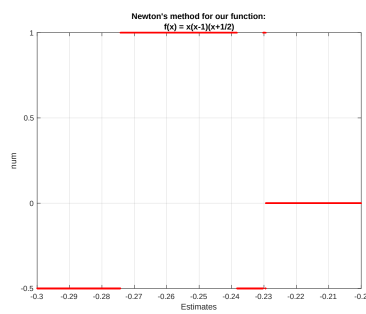
$$f(x) = x(x-1)(x+\frac{1}{2}) \tag{1}$$

for many initial guesses on (-2,2), and plot the zero it converges to as a function of the initial point. This is accomplished using 'AChaoticNewton.m' function and on lines 1 through 29 of 'Project2.m' algorithm. The figure below illustrates even as you make the interval smaller, the jumpiness of the solution looks similar. This means we have a self-similar, or chaotic behavior.

The solution procedure accesses the Newton function written to find roots. The algorithm plots the 'Number found' v. 'Estimates' for Equation (1).



(a)



(b)

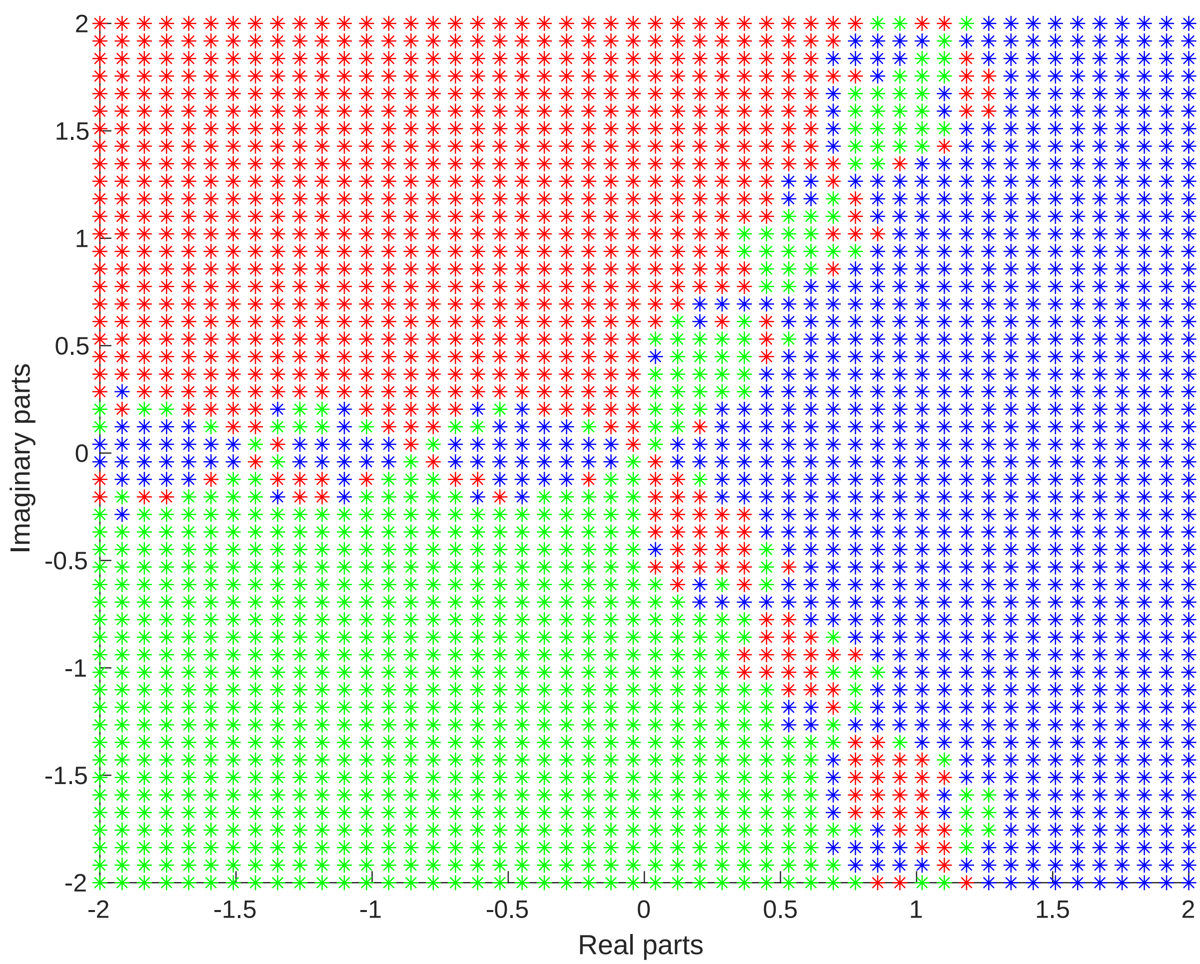
1.2 Applying Newton's Method for 'Part b'

The task is to apply Newton's Method to

$$f(x) = x^3 - 1 \tag{2}$$

with initial guess a selection of complex numbers with real and imaginary parts varying from -2 to 2. After accomplishing this in lines 31 to 57 of 'Project2.m' we plotted; the plot is colored based upon which of the three roots you converge to.

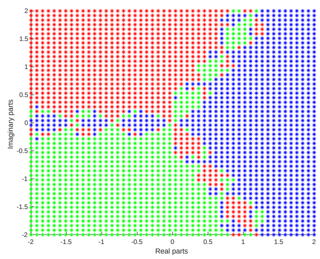
The solution procedure for this function requires a nested loop to compute the real and imaginary parts. We plot the 'real' v. 'imaginary' results in a colorful figure. See figure on next page...



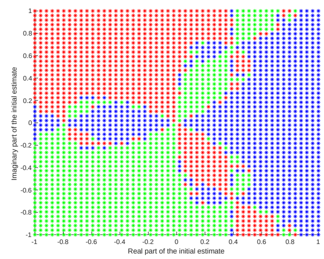
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
Imaginary parts
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
Real parts

1.3 Applying Newton's Method for 'Part c'

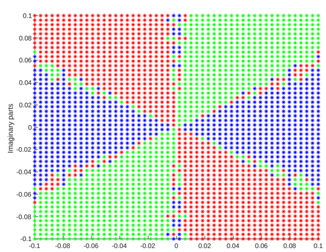
The task is to make a selection of pictures zooming in so the figure has more detail, and verify that the detail continues as you zoom in – it is a fractal. This is accomplished in lines 59 to 111 of 'Project2.m'.



(a)



(b)



(c)

2 Matlab Code

2.1 AChaoticNewton.m

```
function num = AChaoticNewton(f,x,df,total)
M = 100; h = 1;
err = @(xa,xr)abs(xa-xr);

if(nargin<3)
    df = @(x)(f(x+h)-f(x-h))/(2*h);
end

y(1) = f(x(1)); p(1) = df(x(1)); r(1) = x(1)-y(1)/p(1);
error(1) = Inf;
j = 2;

while (error(end)>total) && (j<=M)
    x(j) = r(j-1);    y(j) = f(x(j));
    p(j) = df(x(j));    r(j) = x(j)-y(j)/p(j);
    error(j) = err(x(j),r(j));
    j = j+1;
end

if (j>=M)
    fprintf("There is some dubious error in your code that
        will take you hours to find: HA!",M)
end
num = r(end);
end
```

2.2 Project2.m

```
clear all, close all, clc, format long, format compact

total = 10^-10; n = 203947;
fA = @(x)x.*(x-1).*(x + 1/2);
dfA = @(x)3*x.^2-x-0.5;
A = -2; B = 2; x = linspace(A,B,n);

for j = 1:n
    num(j) = AChaoticNewton(fA,x(j),dfA,total);
end

plot(x,num,'.','color','r')
xlabel('Estimates'), ylabel('Number found')
NewMeth = sprintf('Newton''s method for our function:\nf(x) = x(x-1)(x+1/2)');
title(NewMeth), grid on
saveas(gcf,'fig0.pdf')
ylim([-1,1.5]), xticks(linspace(A,B,23))

A = -0.3; B = -0.2; x = linspace(A,B,n);

for j = 1:n
    num(j) = AChaoticNewton(fA,x(j),dfA,total);
end

figure
plot(x,num,'.','color','r'), xlabel('Estimates'), ylabel('num')
NewMeth = sprintf('Newton''s method for our function:\nf(x) = x(x-1)(x+1/2)');
title(NewMeth), grid on
saveas(gcf,'fig1.pdf')
ylim([-1,1.5]), xlim([A,B])
xticks(linspace(A,B,21))

fB = @(x)x.^3-1; dfB = @(x)3*x.^2; n = 50; total = 1E-5;
```

```
r1=1; r2=(-1-sqrt(-3))/2; r3=(-1+sqrt(-3))/2;
A = -2; B = 2; x = linspace(A,B,n); y = linspace(A,B,n)*sqrt(-1);
q = ones(1,n*n); e = 1;

for i=1:n
    for j=1:n
        num(e) = AChaoticNewton(fB,x(i)+y(j),dfB,total);
        re(e) = x(i);
        im(e) = imag(y(j));
        if abs(num(e)-r2)<0.001
            q(e)=2;
        elseif (abs(num(e)-r3)<0.001)
            q(e)=3;
        end
        e = e+1;
    end
end

figure, hold on
plot(re(q==1),im(q==1),'*','color','B')
plot(re(q==2),im(q==2),'*','color','G')
plot(re(q==3),im(q==3),'*','color','r')
xlabel('Real parts ')
ylabel('Imaginary parts ')
saveas(gcf,'fig2.pdf')
NewMeth = sprintf('Newton''s method applied to\nf(x)=x^3-1');
title(NewMeth)

A = -1; B = 1; x = linspace(A,B,n); y = linspace(A,B,n)*sqrt(-1);
q = ones(1, n*n); e = 1;

for i=1:n
    for j=1:n
        num(e) = AChaoticNewton(fB,x(i)+y(j),dfB,total);
        re(e) = x(i);
```

```
            im(e) = imag(y(j));
            if (abs(num(e)-r2)<0.001)
                q(e)=2;
            elseif (abs(num(e)-r3)<0.001)
                q(e)=3;
            end
            e = e+1;
      end
end

figure, xlim([A,B]), ylim([A,B])
hold on
plot(re(q==1),im(q==1),'*','color','B')
plot(re(q==2),im(q==2),'*','color','g')
plot(re(q==3),im(q==3),'*','color','r')
xlabel('Real part of the initial estimate')
ylabel('Imaginary part of the initial estimate')
saveas(gcf,'fig3.pdf')
NewMeth = sprintf('Newton''s method for: \nf(x)=x^3-1');
title(NewMeth)

A = -0.1; B = 0.1; x = linspace(A,B,n); y = linspace(A,B,n)*
    sqrt(-1);
q = ones(1,n*n); e = 1;

for i=1:n
      for j=1:n
            num(e) = AChaoticNewton(fB,x(i)+y(j),dfB,total);
            re(e) = x(i);
            im(e) = imag(y(j));
            if (abs(num(e)-r2)<0.001)
                q(e)=2;
            elseif (abs(num(e)-r3)<0.001)
                q(e)=3;
            end
            e = e+1;
      end
end
```

```
figure, xlim([A,B]), ylim([A,B])
hold on
plot(re(q==1),im(q==1),'*','color','B')
plot(re(q==2),im(q==2),'*','color','g')
plot(re(q==3),im(q==3),'*','color','r')
xlabel('Real parts of the  estimates')
ylabel('Imaginary parts ')
saveas(gcf,'fig4.pdf')
NewMeth=sprintf('Newton''s method for:\nf(x)=x^3-1');
title(NewMeth)
```

3 Discussion of results and conclusions

3.1 Results

The results of this analysis is Equation (2) is what we define to be a 'fractal' - a complex geometric shape commonly exemplifying fractional dimensions.

3.2 Conclusions

These algorithms work properly: Project2.m uses the AChaoticsNewton function to find the roots of Equation (1) as well as Equation (2). The imaginary and real parts for Equation (2) form a fractal illustrated in Subsection 1.3.