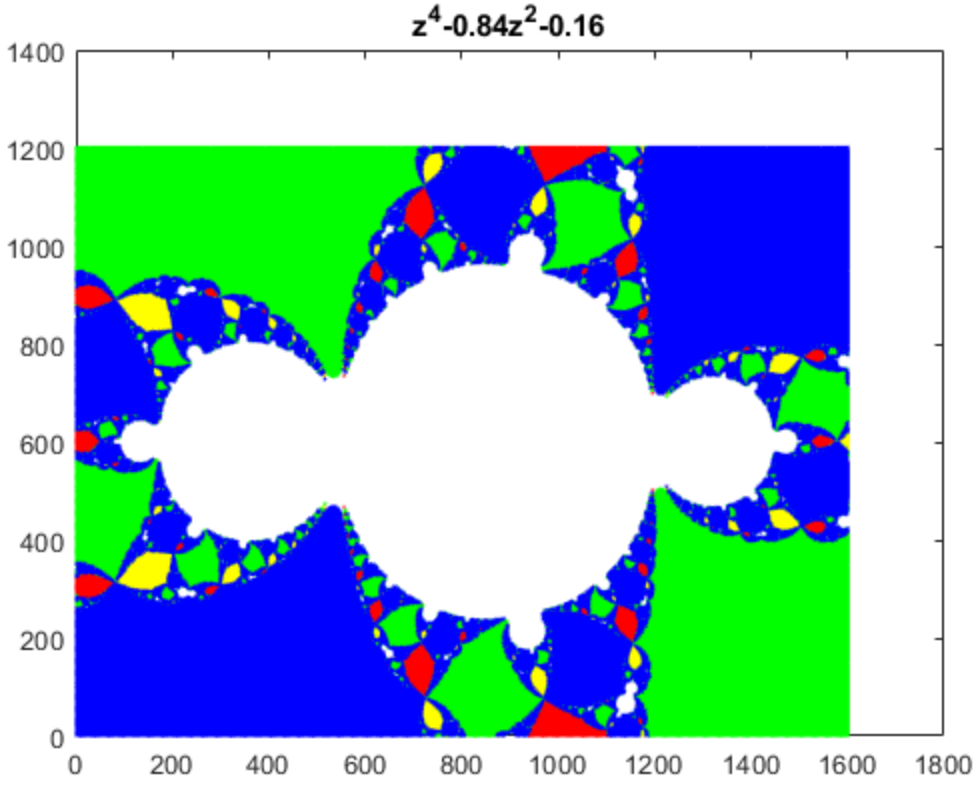
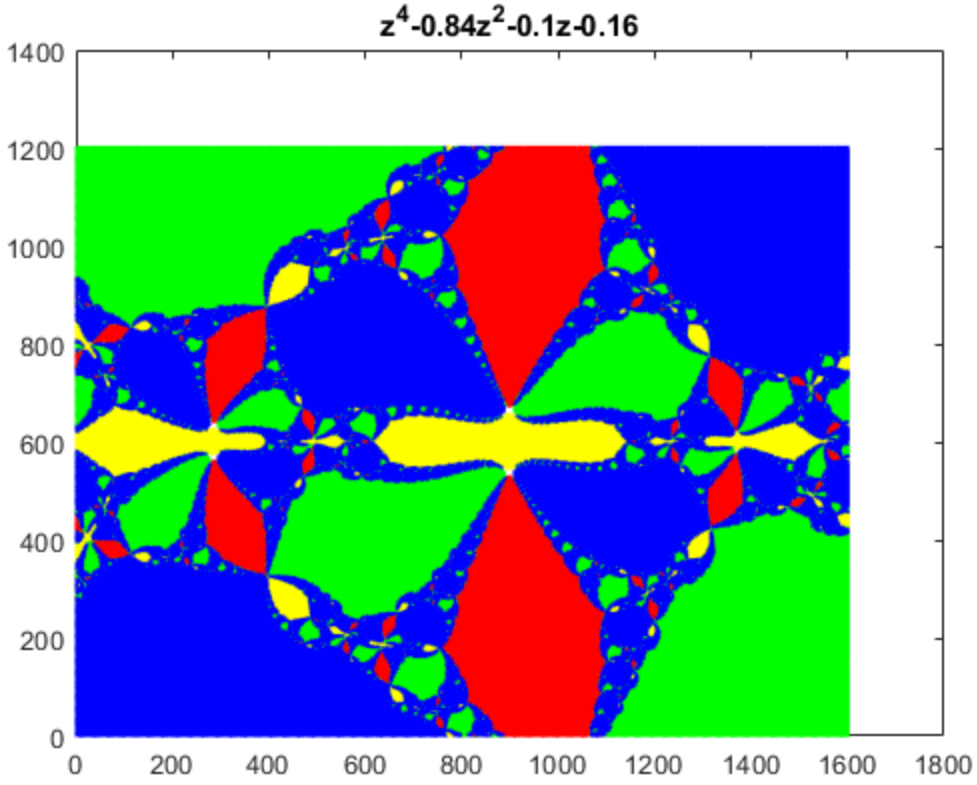
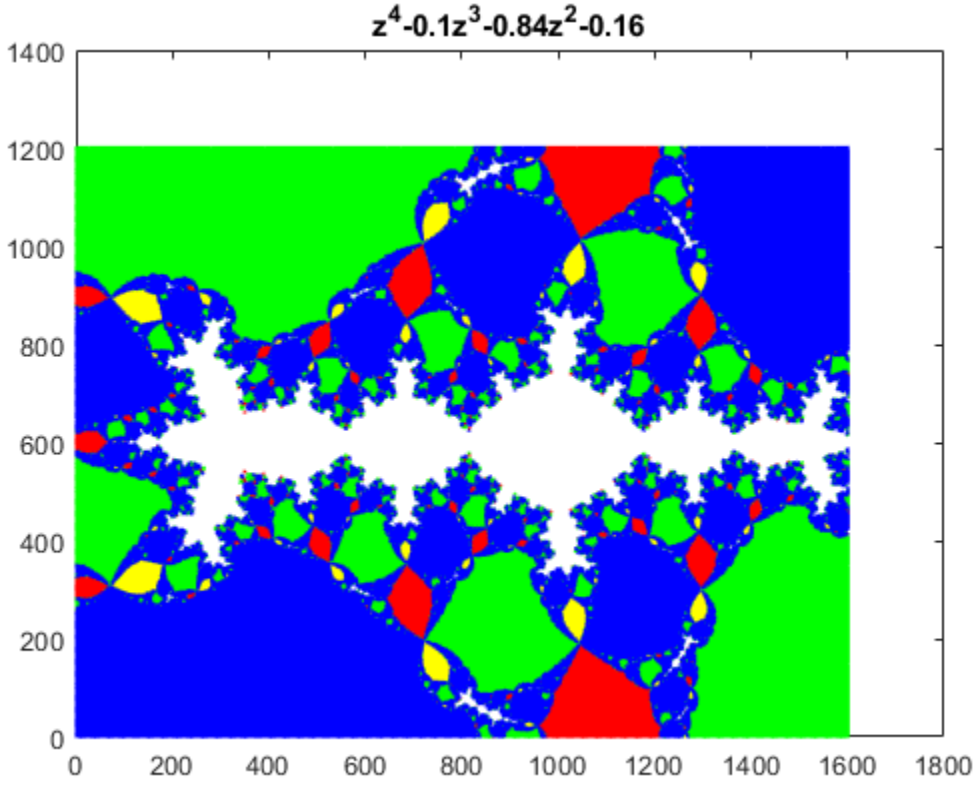
```
% Pierce Zhang, CMOR220, Fall 2023, Project 3: Newton Fractals
% newton_fractals.m
% Plot Newton basins and wastelands for four polynomials of degree 4.
% Last modified: 27 September 2023

% Driver function
% Uses qnewt to produce four plots for each of four quartic functions
function newton_fractals
% Parameters of Newton's method
xt = [0.15,0.00025,0.55];
yt = [-0.15,0.00025,0.15];
maxiter = 20;

% Four plots as specified
qnewt([1,0,-0.84,0,-0.16],xt,yt,maxiter);
qnewt([1 0 -0.84 -0.1 -0.16],xt,yt,maxiter);
qnewt([1 -0.1 -0.84 0 -0.16],xt,yt,maxiter);
qnewt([1 -0.1i -0.84 0 -0.16],xt,yt,maxiter);
end

% Inputs: q, a vector of coefficients of a polynomial
% Outputs: dq, a vector of coefficients of the analytical derivative of q
function [dq] = myownpolyder(q)
% Evaluates the analytical derivative of a polynomial
% Derivative function should have one less term
dq = zeros(1,length(q) - 1);
for i = 1:length(dq)
    % Where the coefficient is q(i) and the power is length(q) - i
    dq(i) = q(i) * (length(q) - i);
end
end

% Inputs:
% coeff, the coefficient of a monomial term
% img, a boolean indicating whether the coefficient is imaginary
% power, the power of z in the monomial term
% Outputs: term, a string containing the formatted monomial term
function [term] = qterm(coeff, img, power)
term = "";
% Sign handler
if (coeff < 0)
    term = term + '-';
elseif (coeff > 0)
    term = term + '+';
else
    return
end
% Coefficient handler
if (abs(coeff) ~= 1)
    term = term + abs(coeff);
elseif (power == 0 && img == false)
    term = term + abs(coeff);
```

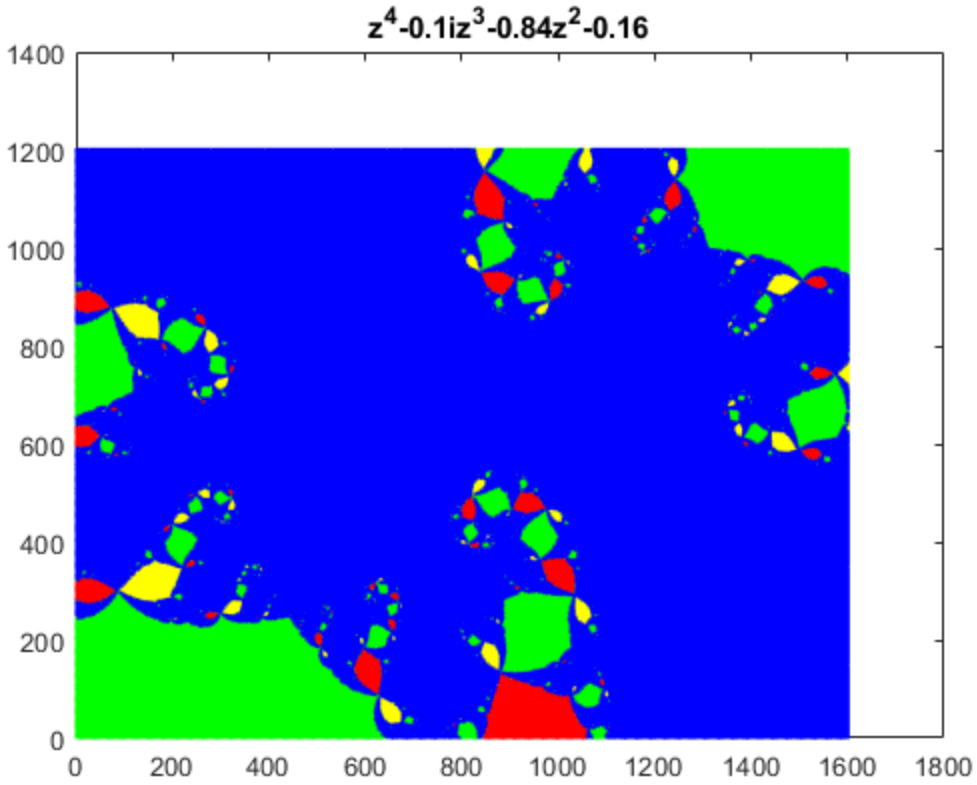
```
end
%Imaginary unit handler
if (img)
    term = term + 'i';
end
% Power handler
if (power == 1)
    term = term + 'z';
elseif (power > 1)
    term = term + 'z^' + power;
end
end

% Inputs: q, a  vector of coefficients for a polynomial
% Outputs: name, a string which represents a polynomial
function [name] = qlab(q)
% Produces a title for a given polynomial as a coefficient vector
name = "";
for i=1:length(q)
    coeff = q(i);
    power = length(q) - i;
    % Skip over all 0 coefficient terms
    if (coeff ~= 0)
        name = name + qterm(real(coeff), false, power);
        name = name + qterm(imag(coeff), true, power);
    end
end
% Remove leading plus sign.
if (name ~= "" && extract(name,1) == '+')
    name = eraseBetween(name, 1, 1);
end
end

% Inputs:
% q, a vector of up to 5 complex coefficients of a polynomial
% xt, a vector of 3 x grid values, xlo, xinc, and xhi
% yt, a vector of 3 y grid values, ylo, yinc, and yhi
% maxiter, the maximum number of iterations
% Outputs: none (produces plots)
function qnewt(q, xt, yt, maxiter)
% Plots the Newton's basins and wasteland for a given function
% 1. Generate initial guesses using meshgrid,xt,yt
% 2. Apply Newton's method
% 3. Assign the colors by using find command
% 4. Apply qlab to assign the titles for the plots

% Parameters of iteration
dq = myownpolyder(q);
x = xt(1):xt(2):xt(3);
y = yt(1):yt(2):yt(3);

% Initialize grid to store results of Newton's method iteration
[X,Y] = meshgrid(x,y);
Z = X + 1i*Y;
```

```
for k=1:maxiter
    % Newton's method on all grid values simultaneously
    Z = Z - polyval(q,Z)./polyval(dq,Z);
end

% Plot
figure()
r = roots(q); % find roots for comparison
colors = ['r','y','g','b'];
for i = 1:length(r)
    [h,k] = find(abs(Z - r(i))<0.001);
    plot(k,h,'.','color',colors(i),'MarkerSize',1);
    hold on
end
title(qlab(q));
end
```




$z^4-0.84z^2-0.1z-0.16$
1400
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
1400
1600
1800


$z^4-0.1z^3-0.84z^2-0.16$
1400
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
1400
1600
1800



Published with MATLAB® R2023a