Julia Sets

Table of Contents

B1

In this part we define functions which:

1. Create a matrix of evenly spaced complex numbers in a grid. (makegrid)

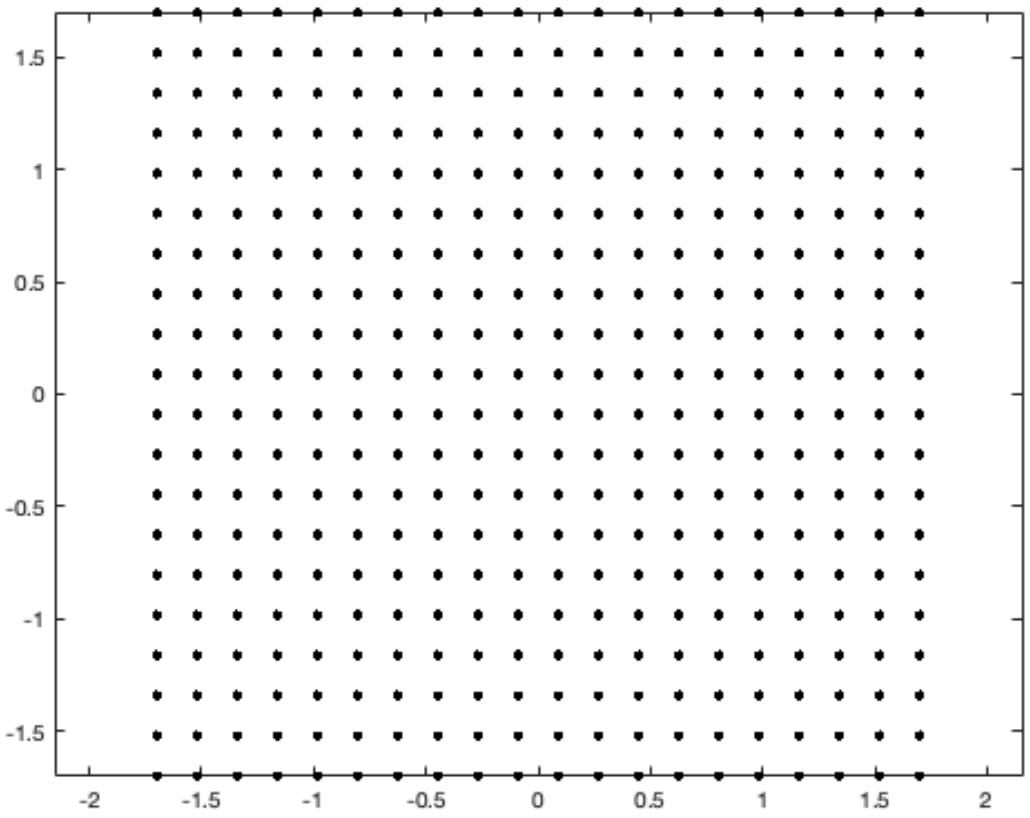
2. Apply an iteration formula 10 times to the matrix's entries. (tensteps)

3. Add a yellow pixel in the complex plane for each complex number who's magnitude is less than 2. (plotW)

Together, these functions create a shape in the complex plane which represents the complex numbers that remain bounded under the iteration formula.

First, we test the makegrid function by plotting a 20x20 grid of black dots.

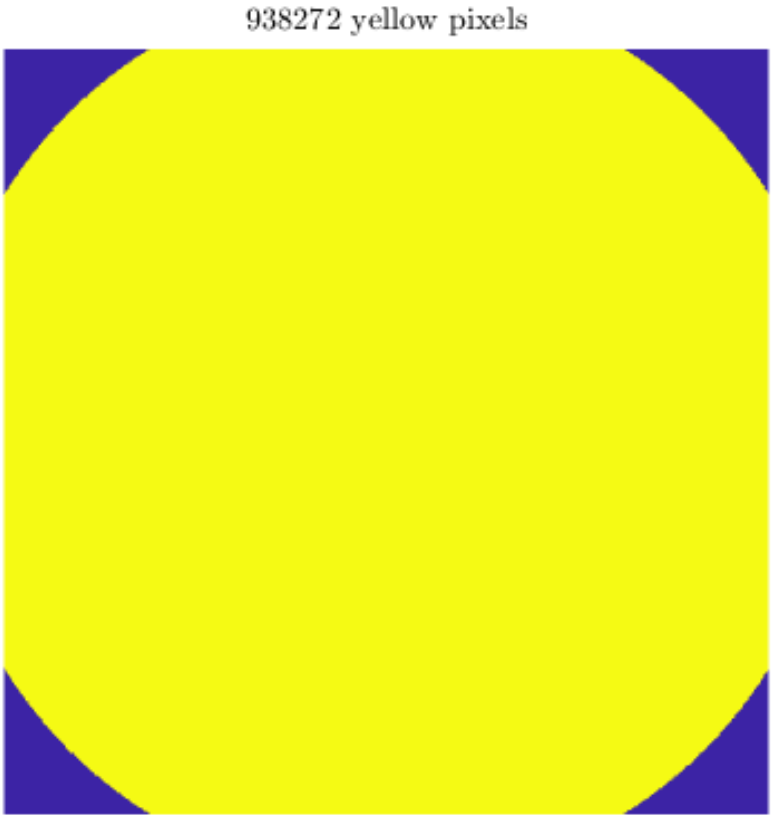
```
% Display the makegrid function.
type makegrid.m

% Test makegrid
figure(1)
plot(makegrid(20), '.k', 'markersize', 10), axis equal


% This function returns an npts x npts matrix of complex numbers x + iy
% in the square -1.7 <= x,y <= 1.7.

function Z = makegrid(npts)
    s = linspace(-1.7, 1.7, npts); % vector of evenly spaced numbers
    [X, Y] = meshgrid(s,s); % mesh to get a set of coordinates
    Z = X + 1i*Y; % produce the matrix of complex numbers
end
```



Now test the plotW function with an evenly spaced grid of complex numbers. A circle is produced as complex numbers with $|w| < 2$ are coloured.

```
% Display the plotW function.
type plotW.m

NUM_PTS = 1000; % size of pixel grid

% Test plotW
figure(2)
plotW(makegrid(NUM_PTS))


% This function takes in a matrix of complex numbers, and adds a yellow
% pixel in the complex plane for each complex number which has |z| < 2.

function plotW(W)
    % Sets each element of W to 0, 1
    % depending on if its magnitude is less than 2.
    pixels = double(abs(W)<2);

    % The pixels corresponding to entries of 1 are coloured yellow.
    pcolor(pixels), shading flat
```

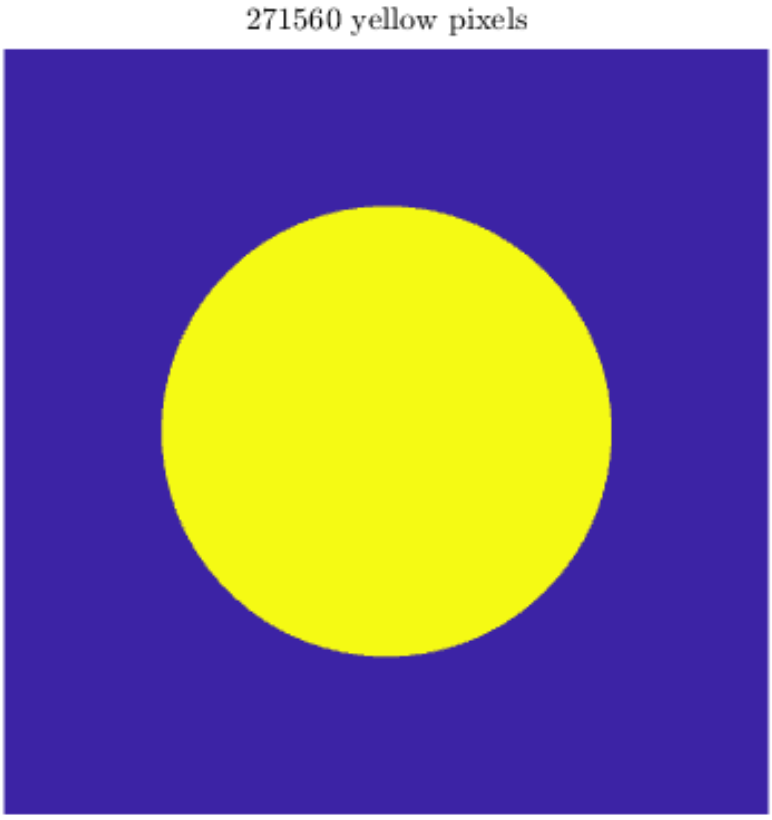
```
    axis square off

    % Count the number of pixels satisfy |z| < 2 and add a title
    num_pixels = sum(pixels, 'all'); % count the number of 1's
    num_pixels_str = int2str(num_pixels); % convert to string
    title(strcat(num_pixels_str, ' yellow pixels'),'Interpreter','latex')
    set(gca, 'fontsize', 13)
end
```



We test the tensteps function, with c = 0. In this case the iteration formula is $w := w^2$. Complex numbers with $|w| > 1$ will 'blow up' under this iteration formula, and points with $|w| \leq 1$ will stay bounded. Hence the pixels shaded yellow, are within the circle $|w| \leq 1$.

```
% Display the tensteps function
type tensteps.m

% Test tensteps
figure(3)
plotW(tensteps(makegrid(NUM_PTS),0))


% This function applies the iteration w := w^2 + c to each complex number
% of the matrix W, 10 times.
```

```
function W = tensteps(W, c)
    for i = 1:10
        W = W.^2 + c;
    end
end
```



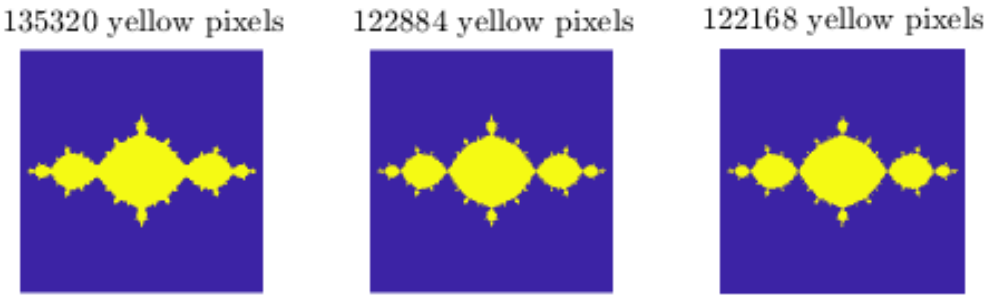
B2

In this part we define the threeplots function which plots the complex numbers which remain bounded after 10, 20, 30 iterations of the formula $w := w^2 + c$.

We test the threeplots function with c = -1, and a very detailed fractal is produced. As the number of iterations increases from 10, 20, 30 the number of pixels in the plot decreases, and the fractal produced becomes more clearly defined. This is because complex numbers are no longer coloured yellow when their magnitude is greater than 2 and after each iteration, some more points will have crossed this threshold and will be excluded from the plot, as some complex numbers will be slow to 'blow up' under the formula.

```
% Display the threeplots function
type threeplots.m

% Test threeplots
```

```
figure(4)
threeplots(-1)


% This function creates three sub-plots on the same
% figure, of the complex numbers which remain bounded,
% after 10, 20, 30 iterations of w := w^2 + c, for a given c

function threeplots(c)
    W = makegrid(1000); % make the grid

    for j = 1:3
        subplot(1,3,j) % switch sub-plot

        W = tensteps(W, c); % apply the iteration formula
        plotW(W) % plot the bounded points
    end
end
```

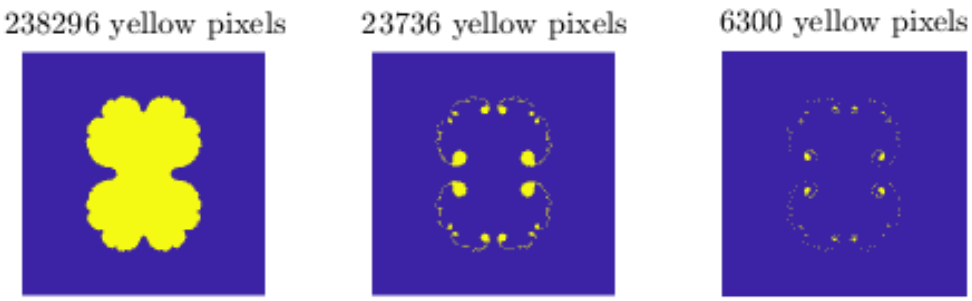


B3

In this part we produce threeplots for c = 0.3.

The first fractal plotted differs greatly from the second and third. Between 10-20 iterations the number of pixels is greatly reduced and the fractal becomes very thin. For this value of c, the plots differ very profoundly, with the biggest difference being between plot 1 & 2. It's clear that the complex numbers from the body of the first fractal all cross $|w| < 2$ between 10 - 20 iterations. This fractal takes longer to form a defined shape, than in B2. This could be because the value of c used (0.3) is closer the 0 than in B2 (-1), so complex numbers are slower to 'blow up' under the iteration formula.

```
figure(5)
threeplots(0.3)
```

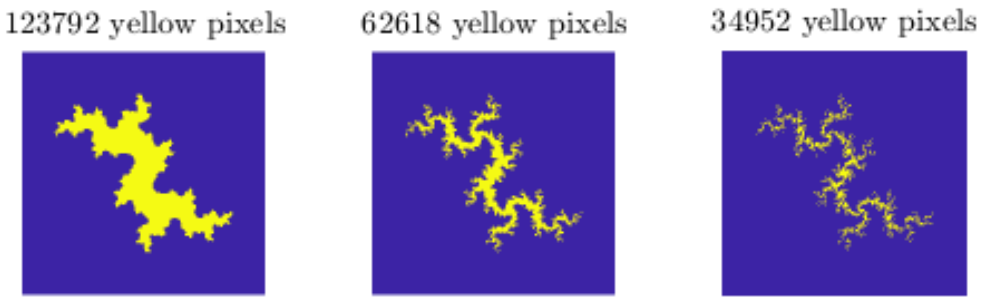


B4

In this part we produce threeplots for two values of c

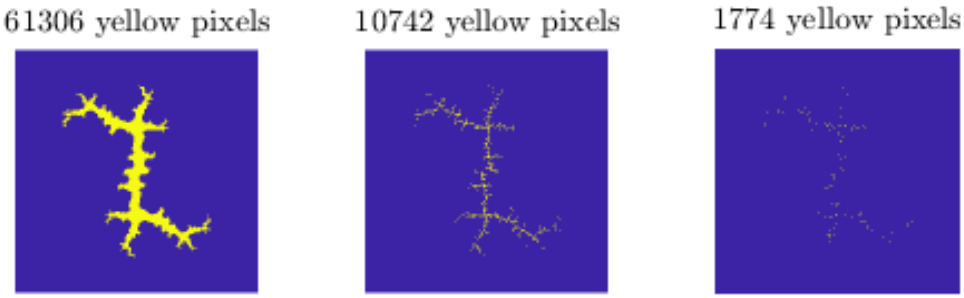
For this value of c we get a very detailed fractal. As the number of iterations increases, the fractal becomes more clearly defined, however unlike in B3, there are no significant differences between any two plots.

```
figure(6)
threeplots(0.75*1i)
```



For this value of c we get a very thin and detailed fractal. Similarly to the previous one, there aren't significant differences between the three plots, the fractal just becomes more clearly defined. After 30 iterations the fractal is very thin and hard to make out. If we took a denser grid of complex numbers, there would be more pixels and the fractal would be better defined.

```
figure(7)
threeplots(0.4+0.6*1i)
```



Published with MATLAB® R2021b