1. $$\frac{(1-i\sqrt{3})^{6/5}}{(1-i)^{12/5}} = \left(\frac{2e^{-\frac{1}{3}\pi i}}{\left(\sqrt{2}\,e^{-1/4\,\pi i}\right)^2}\right)^{6/5}$$

$$= \left(\frac{2e^{-1/3\,\pi i}}{2e^{-\pi/2\, i}}\right)^{6/5}$$

$$= \left(1\,e^{\pi/6\, i}\right)^{6/5}$$

$$= e^{\pi/5\, i}$$

$$= \cos(\pi/5) + i\sin(\pi/5)$$

$$= 0.809 + i\,0.588$$

2 a) $\cos(e^z)$

$$e^z = e^x e^{iy}$$

$$= e^x(\cos y + i\sin y)$$

$$\cos(e^z) = \cos(e^x(\cos y + i\sin y))$$

$$= \cos(e^x\cos y)\cos(e^x i\sin y) - \sin(e^x\cos y)\sin(e^x i\sin y)$$

$$= \cos(e^x\cos y)\cosh(e^x\sin y) - i\sin(e^x\cos y)\sinh(e^x\sin y)$$

Real part $\cos(e^x\cos y)\cosh(e^x\sin y)$

Imaginary part $-\sin(e^x\cos y)\sinh(e^x\sin y)$

2b)

$$\cos(z) = \cos(x+iy)$$
$$= \cos x \cos iy - \sin x \sin iy$$
$$= \cos x \cosh y - i \sin x \sinh y$$

$$e^{\cos x \cosh y - i \sin x \sinh y}$$
$$= e^{\cos x \cosh y} e^{-i \sin x \sinh y}$$
$$= e^{\cos x \cosh y}(\cos(-\sin x \sinh y) + i \sin(-\sin x \sinh y))$$
$$= e^{\cos x \cosh y}(\cos(\sin x \sinh y) - i \sin(\sin x \sinh y))$$

Real part $e^{\cos x \cosh y}\cos(\sin x \sinh y)$

Imaginary part $-e^{\cos x \cosh y}\sin(\sin x \sinh y)$

3a) $\sum_{k=0}^{n-1} w_k = 0$, $w_k = e^{2\pi i k/n}$

$= w_0 + w_1 + w_2 + \dots w_n$

$= 1 + e^{\frac{2\pi i}{n}} + e^{\frac{4\pi i}{n}} + e^{6\pi i/n} + e^{8\pi i/n} + \dots + e^{\frac{2(n-1)\pi i}{n}}$

$= 1 + \left(e^{\frac{2\pi i}{n}}\right)^1 + \left(e^{\frac{2\pi i}{n}}\right)^2 + \left(e^{\frac{2\pi i}{n}}\right)^3 + \dots + \left(e^{\frac{2\pi i}{n}}\right)^{(n-1)}$

$\alpha = w_1^0 + w_1^1 + w_1^2 + w_1^3 + \dots + w_1^{(n-1)}$

$w_1\alpha = w_1 + w_1^2 + w_1^3 + w_1^4 + \dots + w_1^n$

$\alpha - w_1\alpha = w_1^0 - w_1^n$

$\alpha(1-w_1) = w_1^0 - w_1^n$

$\alpha = \frac{w_1^0 - w_1^n}{1-w_1}$

$\alpha = \frac{1 - w_1^n}{1-w_1}$

$w_1 \neq 0$

$w_k = e^{2\pi i k/n} \neq 0$

$\Rightarrow w_1 \neq 0$

$\alpha = \frac{1 - \left(e^{\frac{2\pi i}{n}}\right)^n}{1 - e^{\frac{2\pi i}{n}}}$

$\alpha = \frac{1 - 1^n}{1 - w_1}$

$\alpha = 0$ $\therefore \sum_{k=0}^{n-1} w_k = 0$

3 b) $\prod_{k=0}^{n-1} w_k = \prod_{k=0}^{n-1} e^{\frac{2\pi i k}{n}}$

$= e^{\frac{2\pi i}{n}\left(0+1+2+\cdots+(n-1)\right)}$

$0+1+2+\cdots+(n-1) = \sum_{k=0}^{n-1} k$

$2\sum_{k=0}^{n-1} k = 1+2+\cdots+(n-2)+(n-1)+(n-1)+(n-2)+\cdots+2+1$

$= n(n-1)$

$\Rightarrow \sum_{k=0}^{n-1} k = \frac{n(n-1)}{2}$

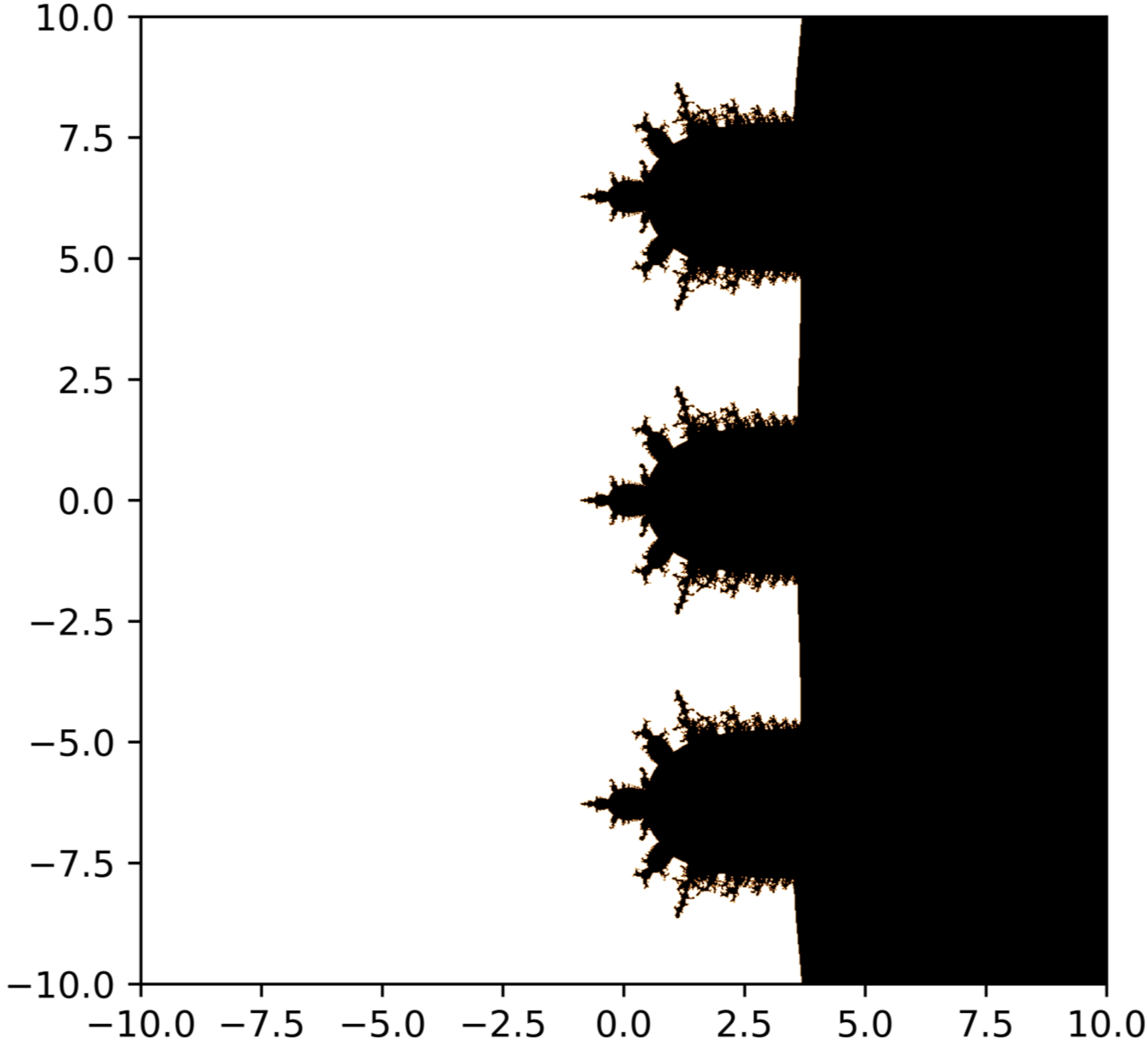
$\prod_{k=0}^{n-1} w_k = e^{\frac{2\pi i}{n}\left(\frac{n(n-1)}{2}\right)}$

$= e^{i\pi(n-1)}$

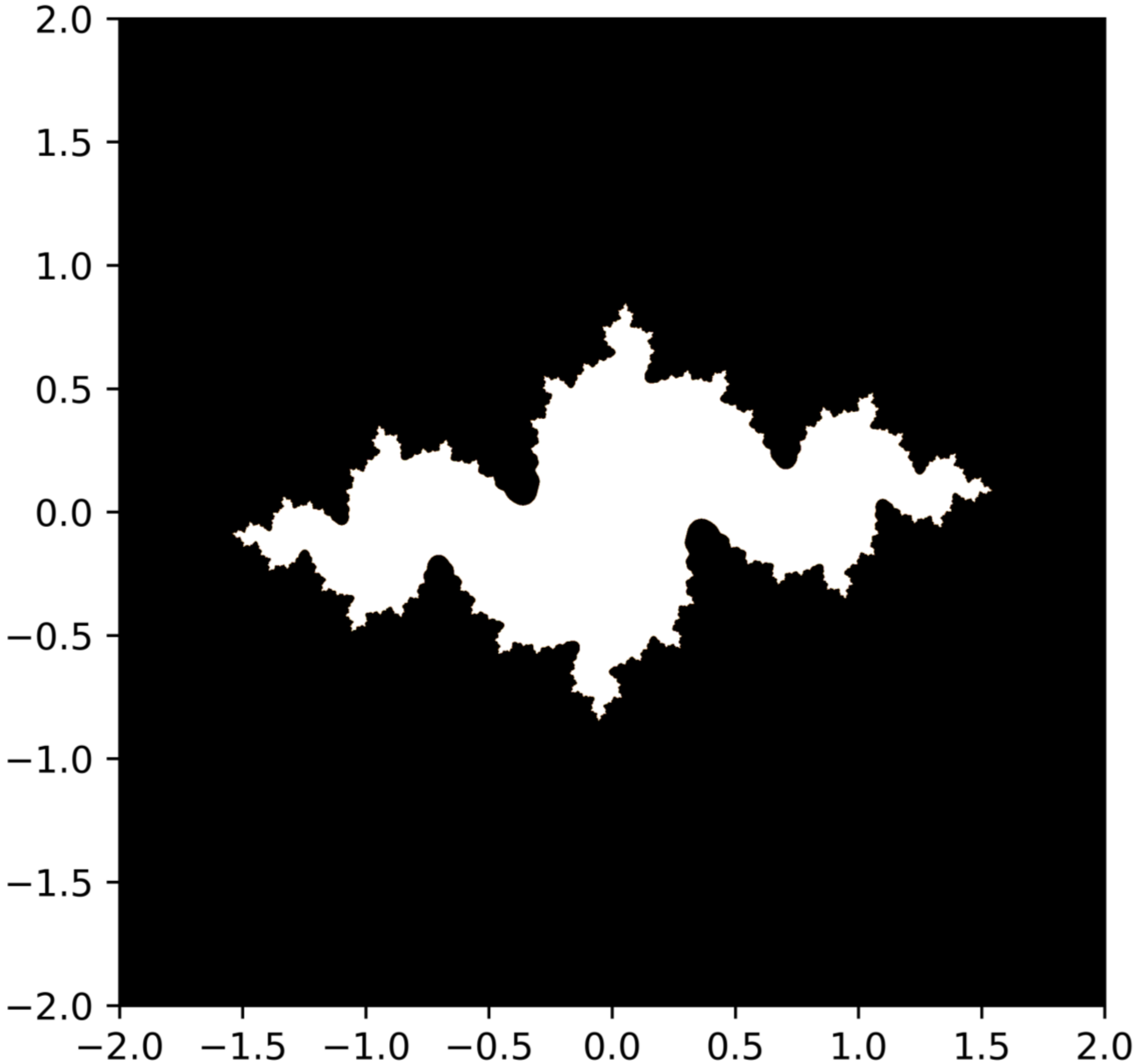
$= \left(e^{i\pi}\right)^{(n-1)}$

$= (-1)^{(n-1)}$

$\therefore \prod_{k=0}^{n-1} w_k = (-1)^{(n-1)}$



10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

```
# mandelbrot.py
import numpy as np
import matplotlib.pyplot as plt


def mandelbrotEsq(a, b, n, thresh):
    xn = 0
    yn = 0
    for i in range(n):
        x = np.exp(xn)*np.cos(yn)+a
        y = np.exp(xn)*np.sin(yn)+b
        if np.linalg.norm([x, y], 2) > thresh:
            return False
        xn = x
        yn = y
    return True


n = 1000
# generate a grid of points
a = np.linspace(-10, 10, n)
b = np.linspace(-10, 10, n)
# check if the point is in the mandelbrot set and if it is plot it on
the graph
# 10 iterations of the function are done and it is removed from the
set if the
# magnitude of the point is greater than 100
sucsess = np.zeros((n, n))
for i in range(n):
    for j in range(n):
        sucsess[j][i] = mandelbrotEsq(a[i], b[j], 10, 100)
# plot the points with a histogram and a color map legend
plt.imshow(sucsess, cmap='afmhot', extent=[-10, 10, -10, 10])
plt.xlim(-10, 10)
plt.ylim(-10, 10)
plt.savefig('mandelbrot.png', dpi=800)
```

```
# julia.py
import numpy as np
import matplotlib.pyplot as plt


def julia(x, y, a, b, n, thresh):
    for i in range(n):
        #zn = zn**2 + c
        xn = x**2 - y**2 + a
        yn = 2*x*y + b
        if np.linalg.norm([xn, yn], 2) > thresh:
            return False
        x = xn
        y = yn
    return True


a = -0.83
b = 0.18

n = 2000
x = np.linspace(-2, 2, n)
y = np.linspace(-2, 2, n)
sucsess = np.zeros((n, n))
# check if the point is in the julia set and if it is plot it on the
graph
# 15 iterations of the function are done and it is removed from the
set if the
# magnitude of the point is greater than 1000
for i in range(n):
    for j in range(n):
        sucsess[j][i] = julia(x[i], y[j], a, b, 15, 1000)
# plot the points with a histogram
plt.imshow(sucsess, cmap='afmhot', extent=[-2, 2, -2, 2])
plt.xlim(-2, 2)
plt.ylim(-2, 2)
plt.savefig('julia.png', dpi=800)
```