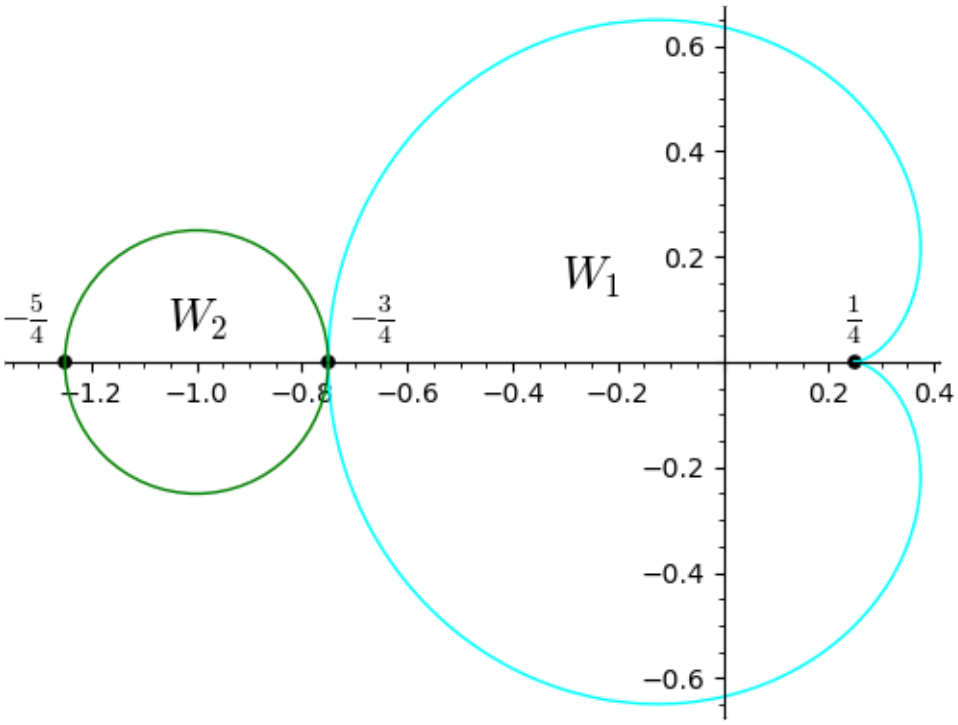
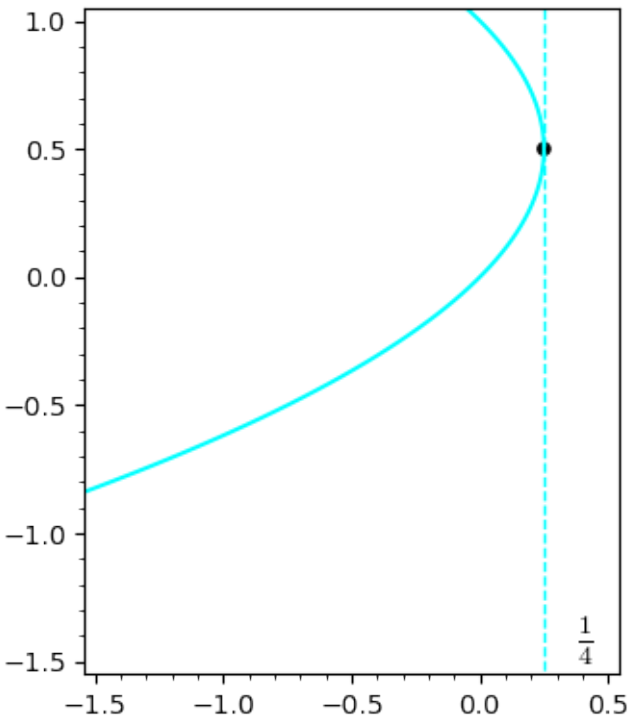
mandelbrot_intro_II

May 25, 2021

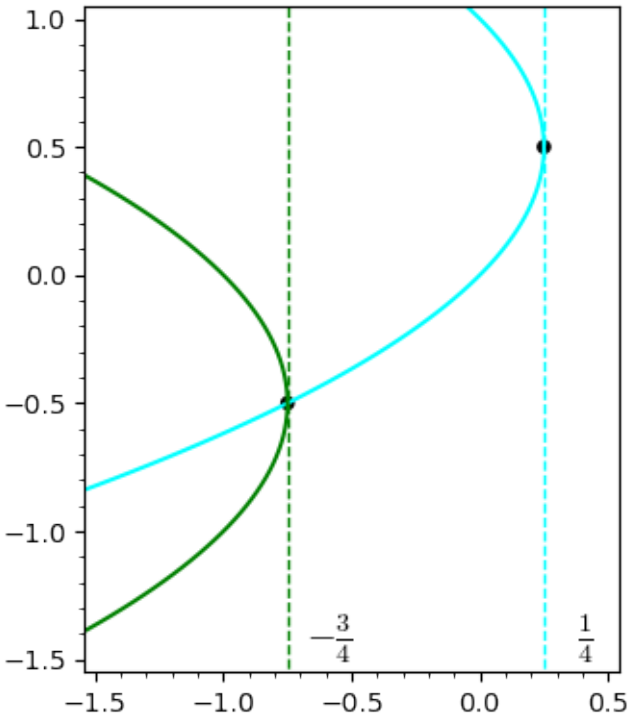
[7]:
```
# Hyperbolic components of periods 1 and 2
var('t, r')
# Parametrization of the hyperbolic component of period 1
cx = r*(1/2)*cos(2*pi*t) - r^2*(1/4)*cos(4*pi*t)
cy = r*(1/2)*sin(2*pi*t) - r^2*(1/4)*sin(4*pi*t)
W1 = parametric_plot((cx(r = 1), cy(r = 1)), (t, 0, 1), color = 'cyan')
W2 = circle((-1,0), 1/4, color = 'green')
pts = list_plot([(-5/4,0),(-3/4,0),(1/4,0)], color = 'black', size = 30)
eps = 0.08
label1 = text("$W_1$", (- 0.25, 2*eps), color = 'black', fontsize = 'xx-large')
label2 = text("$W_2$", (- 1, eps), color = 'black', fontsize = 'xx-large')
label3 = text("$\\frac{1}{4}$", (1/4, eps), color = 'black', fontsize =␣
 ↪'x-large')
label4 = text("$- \\frac{3}{4}$", (-3/4 + eps, eps), color = 'black', fontsize␣
 ↪= 'x-large')
label5 = text("$- \\frac{5}{4}$", (-5/4 - eps, eps), color = 'black', fontsize␣
 ↪= 'x-large')
labels = label1 + label2 + label3 + label4 + label5
list_plot([(-3/4,0),(1/4,0)], color = 'black', size = 30)
(W1 + W2 + pts + labels).show(aspect_ratio = 1)
```



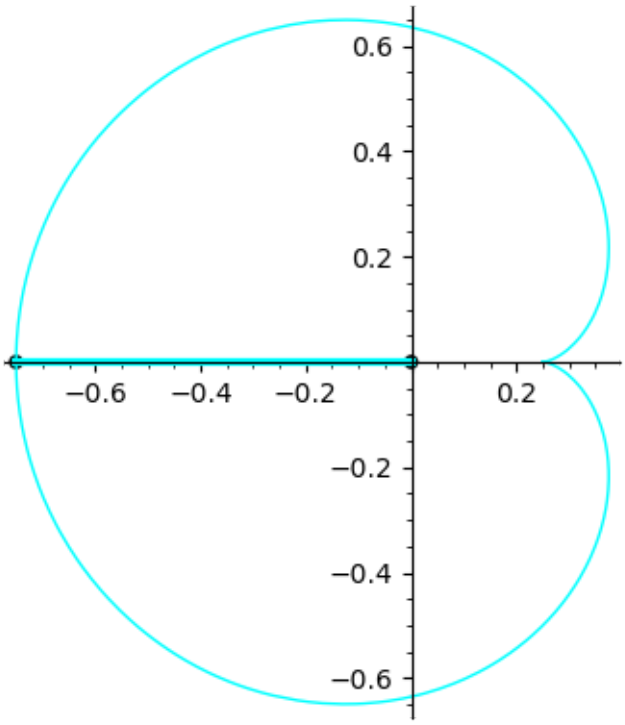
[8]:
```
# Real fixed points
var('x,c')
f = x^2 + c
per1 = implicit_plot(f - x == 0, (c, -2, 2), (x, -2, 2), color = 'cyan')
bif1 = list_plot([(1/4,-2),(1/4,2)], plotjoined = True, color = 'cyan',␣
 ↪linestyle = '--')
pt1 = list_plot([(1/4,1/2)], color = 'black', size = 30)
txt1 = text("$\\frac{1}{4}$", (1/4 + 2*eps, -1.5 + eps), color = 'black',␣
 ↪fontsize = "x-large")
doubling1 = per1 + bif1 + pt1 + txt1
(doubling1).show(xmin = -1.5, xmax = 0.5, ymin = -1.5, ymax = 1)
```



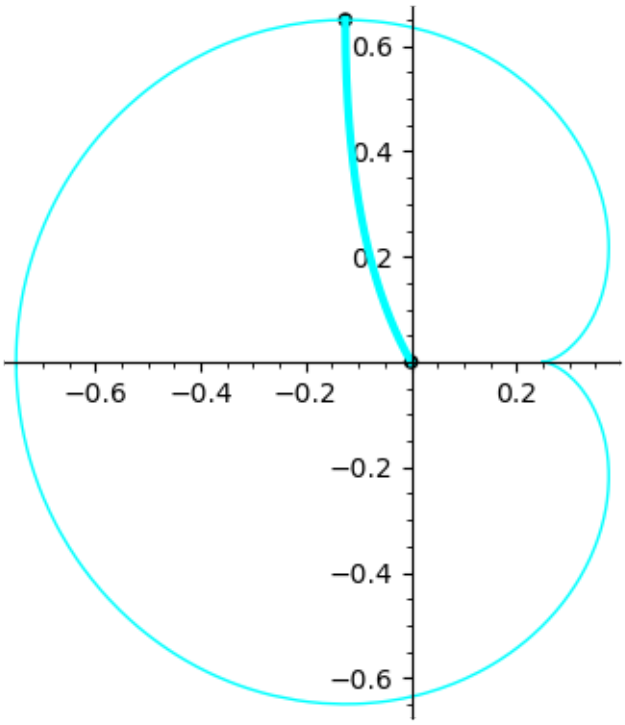
[9]:
```
# Real periodic points of period 2
g = (f(x = f) - x)/(f - x)
per2 = implicit_plot(g == 0, (c, -2, 2), (x, -2, 2), color = 'green')
bif2 = list_plot([(-3/4,-2),(-3/4,2)], plotjoined = True, color = 'green',␣
↪linestyle = '--')
pt2 = list_plot([(-3/4,-1/2)], color = 'black', size = 30)
txt2 = text("$-\\frac{3}{4}$", (-3/4 + 2*eps, -1.5 + eps), color = 'black',␣
↪fontsize = 'x-large')
doubling2 = per2 + bif2 + pt2 + txt2
(doubling1 + doubling2).show(xmin = -1.5, xmax = 0.5, ymin = -1.5, ymax = 1)
```



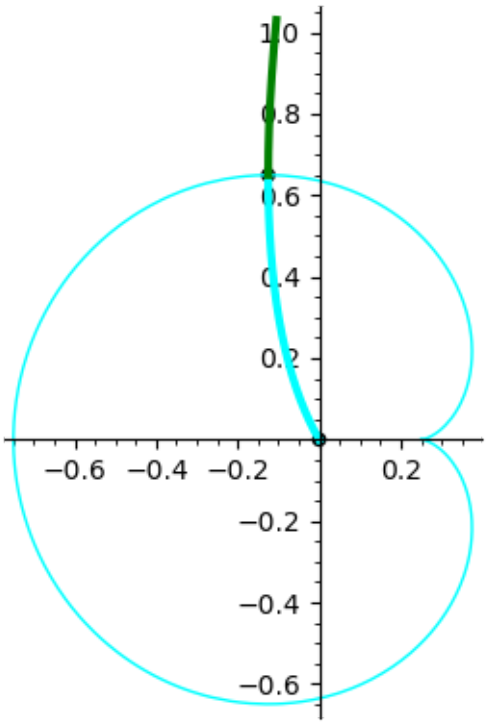
[12]:
```
# Internal ray of the main hyperbolic component
# Angle 1/2
varphi = 0.5
internal_ray = parametric_plot((cx(t = varphi),cy(t = varphi)), (r, 0, 1),␣
 ↪color = 'cyan', thickness = '3')
pvarphi = list_plot([(0,0),(cx(t = varphi, r = 1), cy(t = varphi, r = 1))],␣
 ↪size = 30, color = 'black')
(W1 + internal_ray + pvarphi).show(aspect_ratio = 1)
```



[13]:
```
# Internal ray of angle 1/3
varphi = (1/3).n()
internal_ray = parametric_plot((cx(t = varphi),cy(t = varphi)), (r, 0, 1),␣
 ↪color = 'cyan', thickness = '3')
pvarphi = list_plot([(0,0),(cx(t = varphi, r = 1), cy(t = varphi, r = 1))],␣
 ↪size = 30, color = 'black')
(W1 + internal_ray + pvarphi).show(aspect_ratio = 1)
```



[14]:
```
# Extended internal ray of angle 1/3
internal_ray_extended = parametric_plot((cx(t = varphi),cy(t = varphi)), (r, 1,␣
↪1.4), color = 'green', thickness = '3')
pvarphi_extended = list_plot([(0,0),(cx(t = varphi, r = 1), cy(t = varphi, r =␣
↪1))], size = 30, color = 'black')
(W1 + internal_ray + internal_ray_extended + pvarphi).show(aspect_ratio = 1)
```



[15]:
```
# Alpha fixed points near the period tripling bifurcation
w = polygen(CC)
N = 400
per1 = []
for i in range(N):
    ri = 0.8 + 0.4*i/N
    c = ((1/2)*ri*exp(2*pi*I*(1/3)) - (1/4)*ri^2*exp(4*pi*I*(1/3))).n()
    for ro in (w^2 - w + c).roots(ring = CC):
        if ro[0].imag() > 0:
            # print(ro[0])
            per1.append((ro[0].real(), ro[0].imag(), ri))
# list_plot(C, size = 1, aspect_ratio = 1)
per1_plot = list_plot(per1, size = 1, aspect_ratio = 1, color = 'cyan')
per1_plot.show()
```

Graphics3d Object

[16]:
```
M = 1000
per3 = []
for i in range(M):
    ri = (1 + 0.2*(i/M)^3).n()
    c = ((1/2)*ri*exp(2*pi*I*(1/3)) - (1/4)*ri^2*exp(4*pi*I*(1/3))).n()
    z = 0
```

```
    L = 10 + int(M^(2/3))
    for i in range(L):
        z = z^2 + c
        if i >= L - 3:
            per3.append((z.real(),z.imag(),ri))
per3_plot = list_plot(per3, size = 1, aspect_ratio = 1, color = 'green')
(per1_plot + per3_plot).show()
```

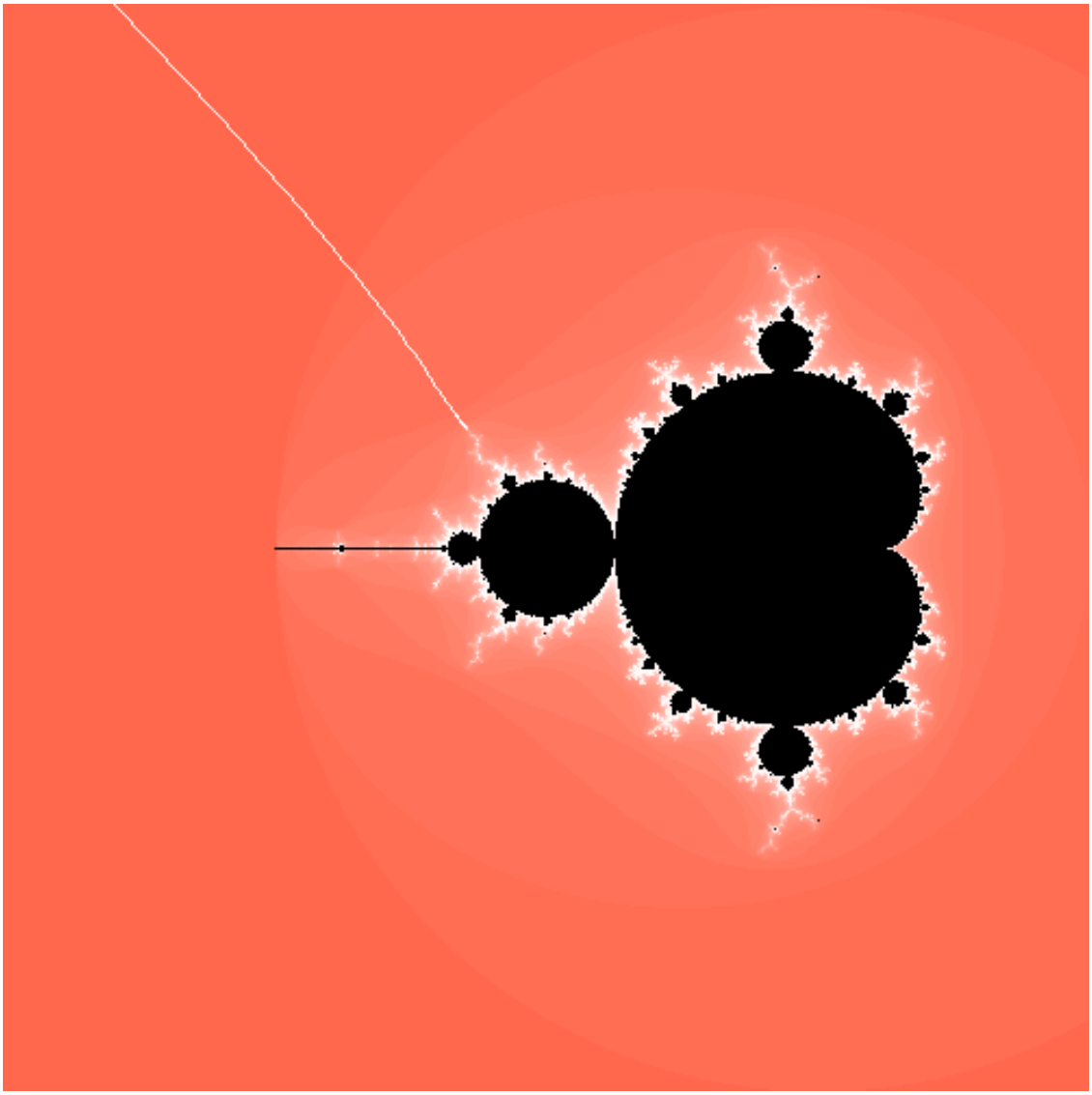
Graphics3d Object

[17]:
```
# Period doubling bifurcation: Attracting basins
# To create movie, use:
# ffmpeg -framerate 5 -i "bif2_julia%02d.png" -c:v libx264 -r 30 -pix_fmt␣
 ↪yuv420p bif2.mp4
R.<z> = CC['z']
N = 100
for i in range(N):
    ci = - i/N
    ji = julia_plot(z^2 + ci, mandelbrot = False)
    ind = str(i)
    if len(ind) == 1:
        ind = '0'+ind
    ji.save('movies/bif2_julia' + ind + '.png')
```

[23]:
```
# Period tripling bifurcation: Attracting basins
# To create movie, use:
# ffmpeg -framerate 5 -i "bif3_julia%02d.png" -c:v libx264 -r 30 -pix_fmt␣
 ↪yuv420p bif3.mp4
N = 100
for i in range(N):
    ri = 1.15*i/N
    ci = ((1/2)*ri*exp(2*pi*I*(1/3)) - (1/4)*ri^2*exp(4*pi*I*(1/3))).n()
    ji = julia_plot(z^2 + ci, mandelbrot = False)
    ind = str(i)
    if len(ind) == 1:
        ind = '0'+ind
    ji.save('movies/bif3_julia' + ind + '.png')
```

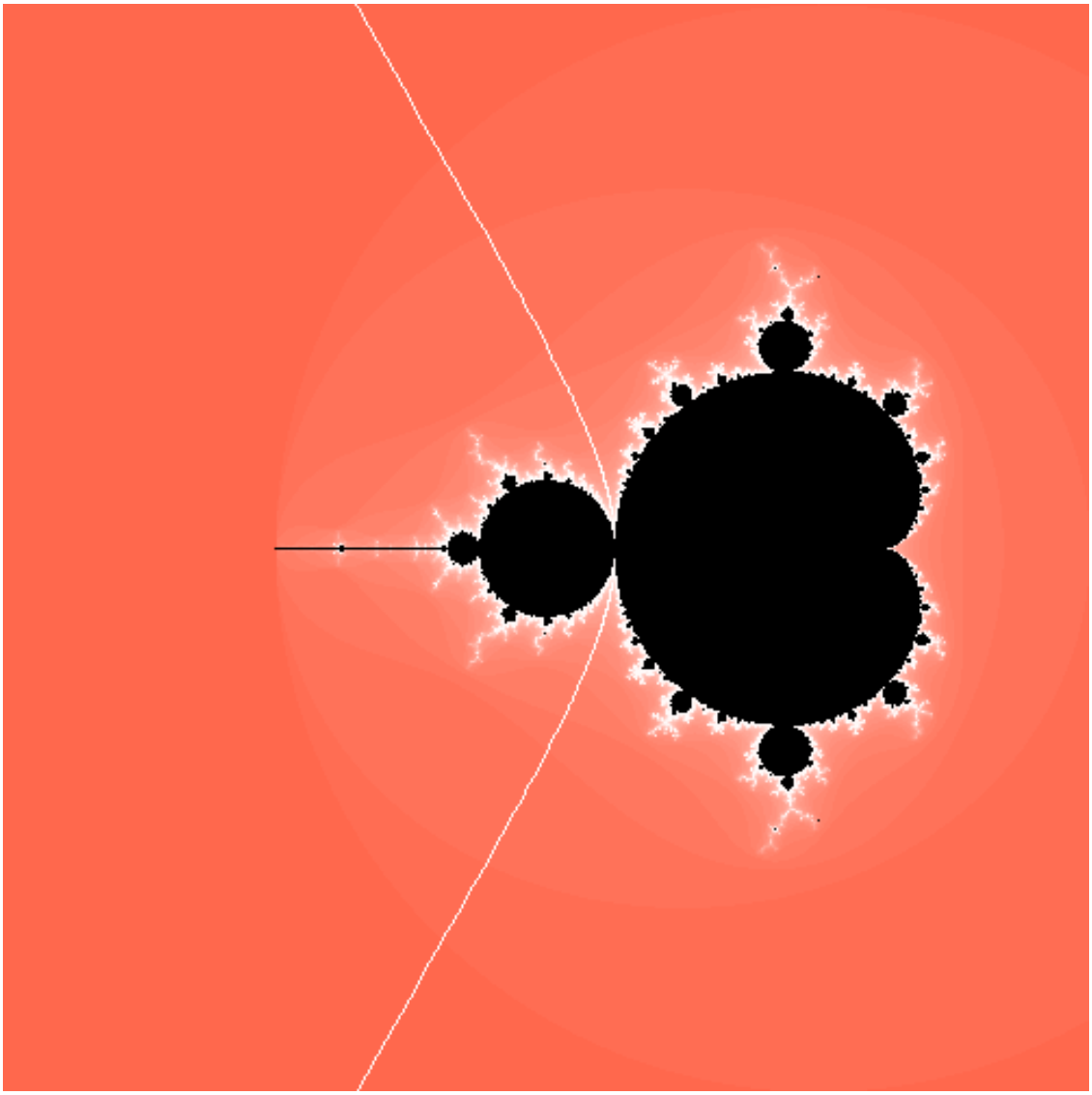
[124]:
```
external_ray(0.375)
```

[124]:



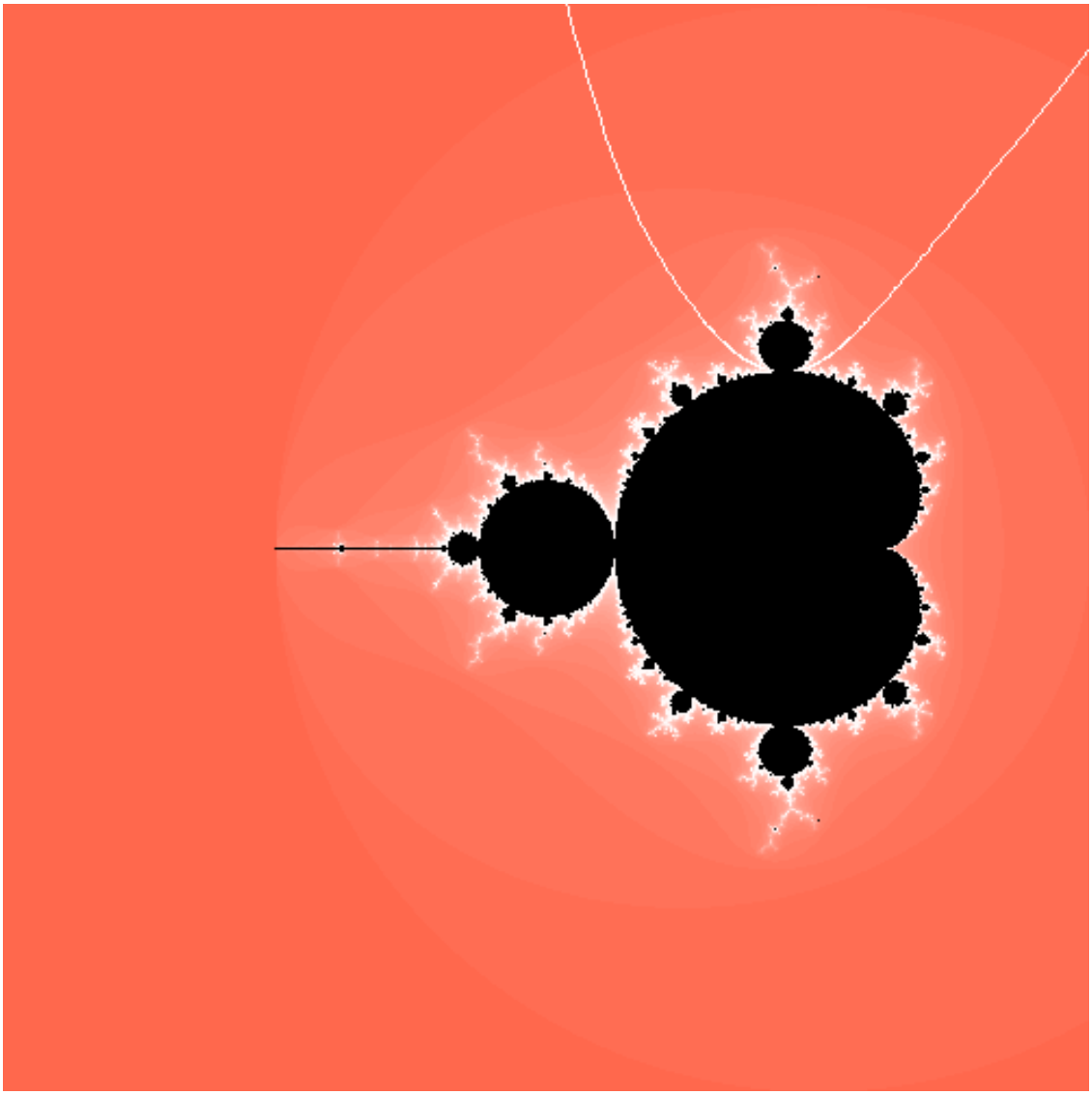
[117]:
```
external_ray([1/3, 2/3])
```

[117]:



[118]:
```
external_ray([1/7, 2/7])
```

[118]:



[5]:
```
# Straight ray
theta = 0.375
var('r')
straight_ray = parametric_plot((r*cos(2*pi*theta), r*sin(2*pi*theta)), (r, 1,␣
 ↪5), color = 'black')
straight_ray_plot = unit_circle + straight_ray
straight_ray_plot.axes(False)
s = 2.5
unit_circle = circle((0,0), 1, fill=True, edgecolor='black', facecolor='white',␣
 ↪transparent = True)
straight_ray_plot.show(xmin = -s, xmax = s, ymin = -s, ymax = s)
# To change background:
# 1. Create image with the " trasparent = True " option;
```

```
# 2. Create background with the right size and color:
# convert -size 500x500 xc:#0097b2 bg.png
# 3. Merge:
# convert bg.png circle.png -composite out.png
```