Some graphical explorations of the Julia sets with python and pyreport

```
#!/usr/bin/env python
from scipy import *
from pylab import *
#from pylab import imshow
```

We start by defining a function J:

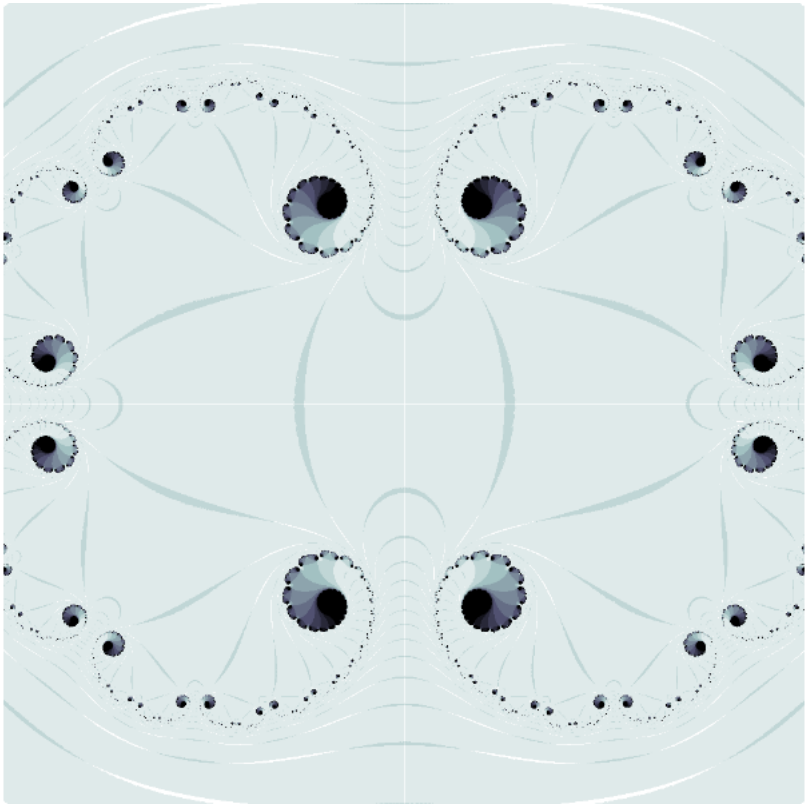
$$J_c : z \rightarrow z^2 + c$$

```
def J(c):
    return lambda z : z**2 + c

[x,y] = ogrid[ -1:1:0.002, -1:1:0.002 ]
z = x + y *1j
```

If we study the divergence of function J under repeated iteration depending on its inital conditions we get a very pretty graph

```
threshTime = zeros_like(z)
for i in range(40):
    z = J(0.285)(z)
    threshTime += z*conj(z) > 4
figure(0)
axes([0,0,1,1])
axis('off')
imshow(threshTime)
bone()
show()
```



We can also do that systematicaly for other values of c:

```
axes([0,0,1,1])
axis('off')
rcParams.update({'figure.figsize': [10.5,5]})
c_values = (0.285 + 0.013j, 0.45 - 0.1428j, -0.70176 -0.3842j,
    -0.835-0.2321j, -0.939 +0.167j, -0.986+0.87j)
```

```
36 for i,c in enumerate(c_values):
37     threshTime = zeros_like(z)
38     z = x + y *1j
39     for n in range(40):
40         z = J(c)(z)
41         threshTime += z*conj(z) > 4
42     subplot(2,3,i+1)
43     imshow(threshTime)
44     axis('off')
45 show()
```