1 Fractal dimension regression

```
import numpy as np
import numpy.linalg as linalg
import matplotlib.pyplot as plt
from PIL import Image, ImageFilter, ImageOps
from scipy import interpolate
from src.intensity_entropy import *
from src.kernels import *
plt.rcParams['image.cmap'] = 'inferno'
```

```
img = ImageOps.grayscale(Image.open('test.jpg'))
scale = max(np.shape(img))
data = np.array(img)
img
```

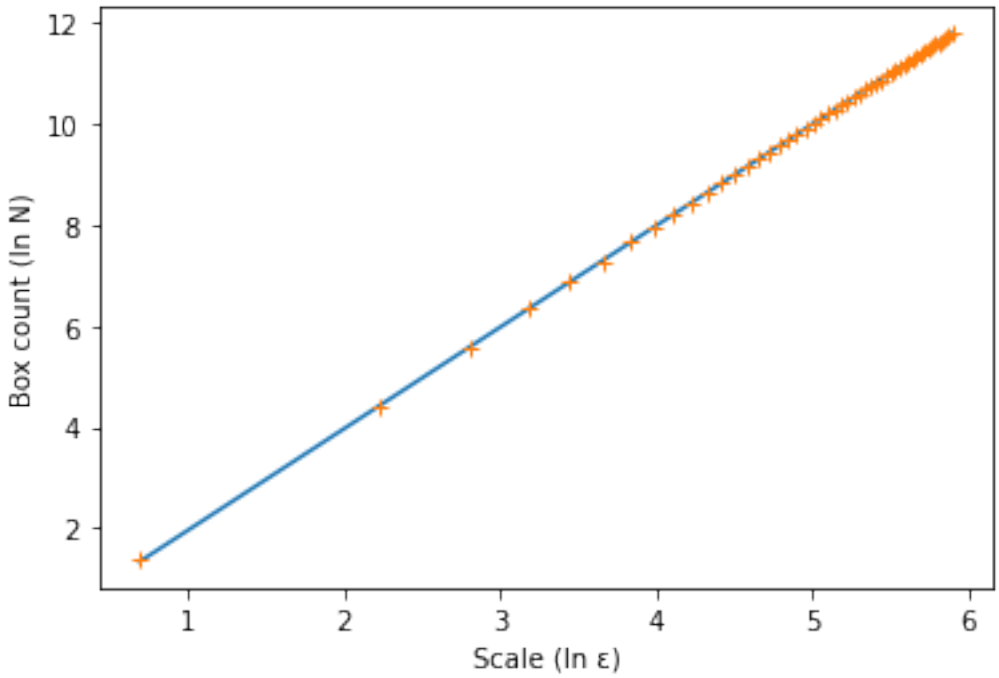
1.1 Box-counting dimension

```
def boxdim(data):
    εs = np.linspace(2, min(np.shape(data)))
    boxes = [np.log(np.sum(mapblocks(
        ε, ε, lambda x: 1 if np.any(x) else 0, data))) for ε in εs]
```

```
    logεs = np.log(εs)
    endεs = logεs[[0, -1]]
    dimfit = np.polyfit(np.log(εs), boxes, 1) # [slope, intercept]
    plt.plot(endεs, dimfit[0]*endεs + dimfit[1])
    plt.plot(logεs, boxes, '+')
    plt.xlabel('Scale (ln ε)')
    plt.ylabel('Box count (ln N)')
    return dimfit[0]
```

```
boxdim(data)
```

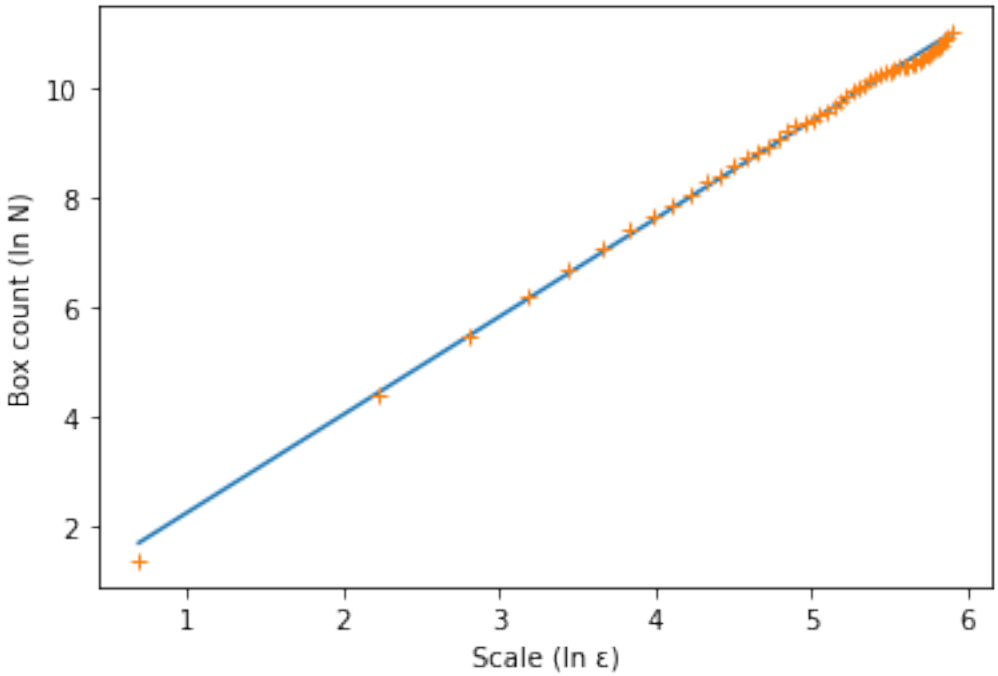
```
2.0087040269581435
```



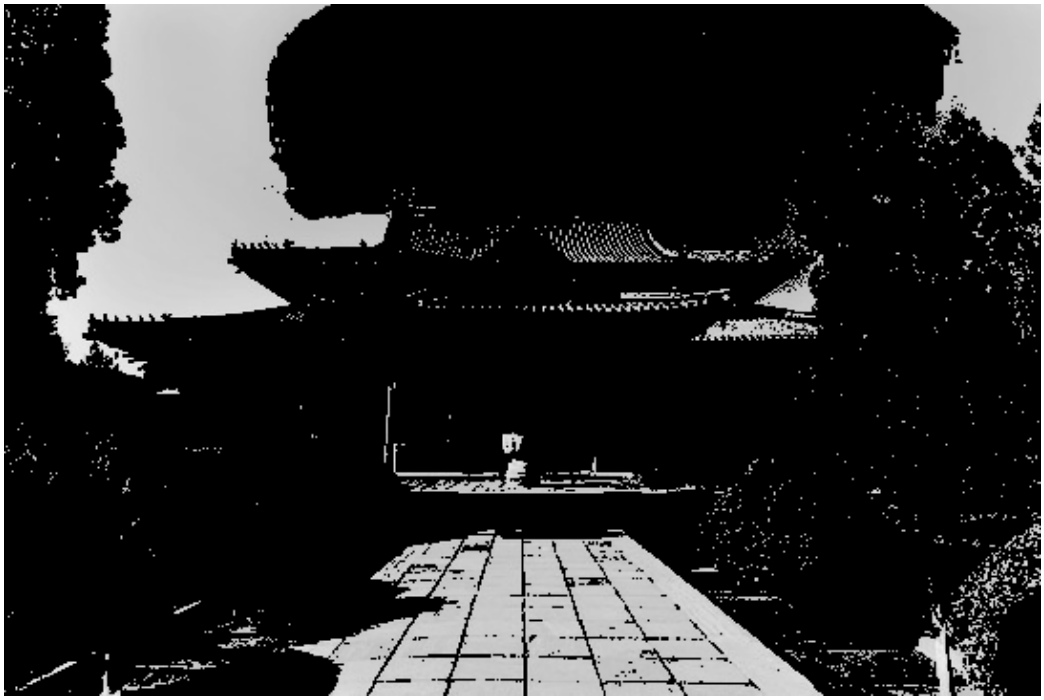

```
sky = data.copy()
sky[sky < 128+32] = 0
Image.fromarray(sky)
```

```
boxdim(sky)
```

```
1.7877778191348215
```

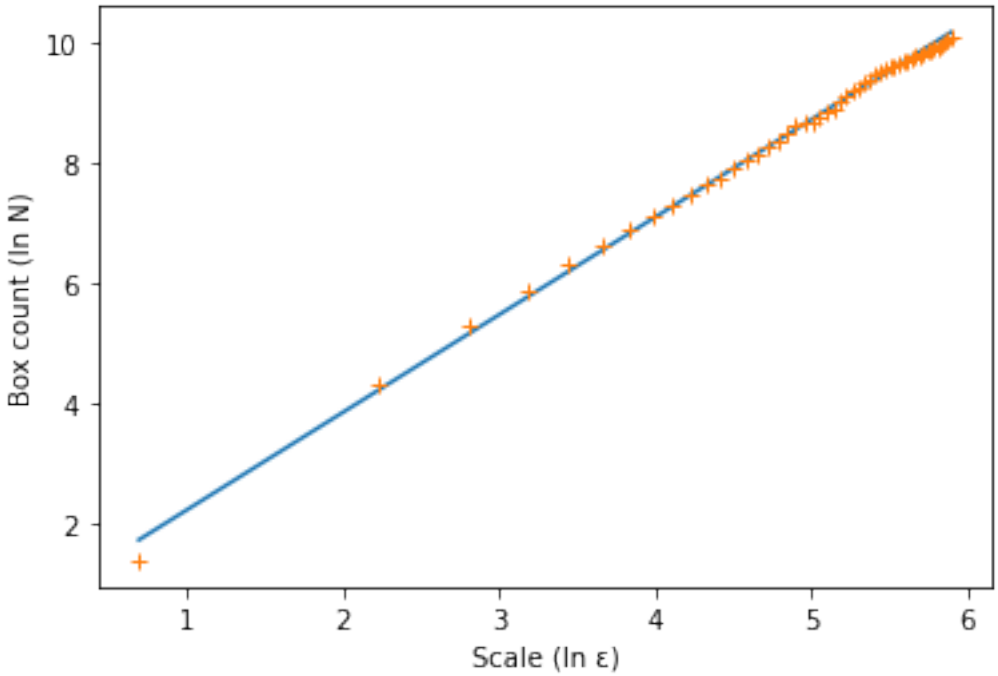



```
nosky = data.copy()
nosky[nosky < 128+64] = 0
Image.fromarray(nosky)
```



```
boxdim(nosky)
```

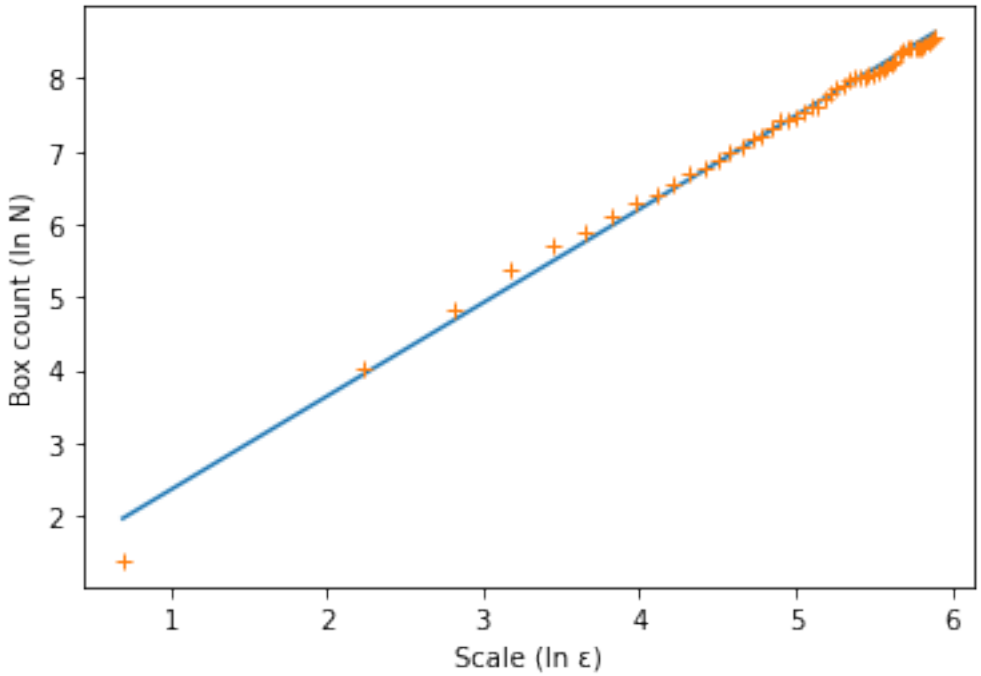
```
1.6214794967487127
```




```
dots = data.copy()
dots[nosky < 128+64+16] = 0
Image.fromarray(dots)
```

```
boxdim(dots)
```

```
1.2821025677557252
```

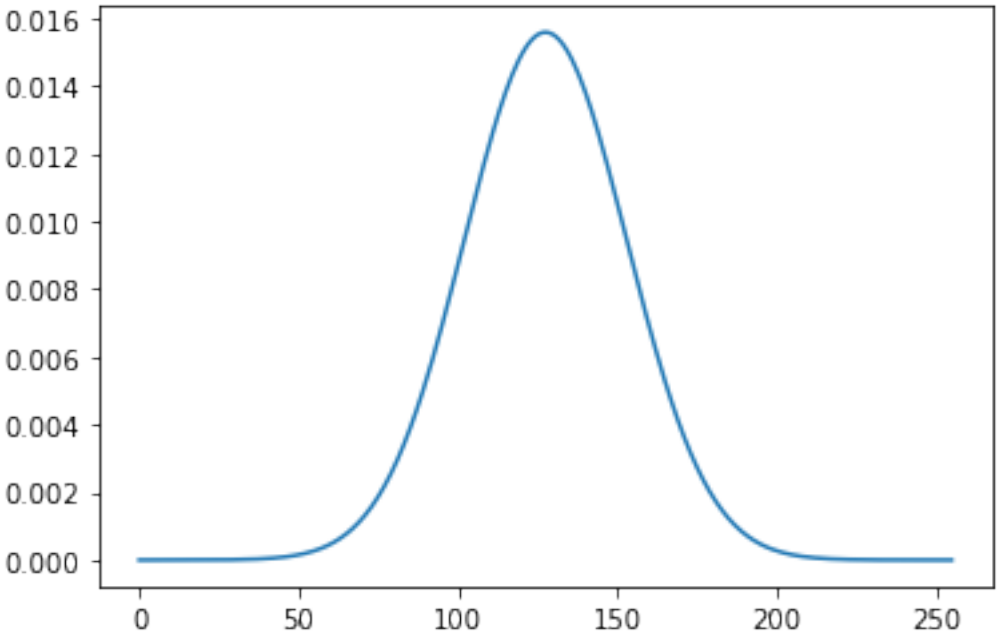


1.2 Information dimension

Figured out problem: discretized samples must still be normalized. I.e. integrate over ε-segment to produce value. See Wikipedia information dimension.

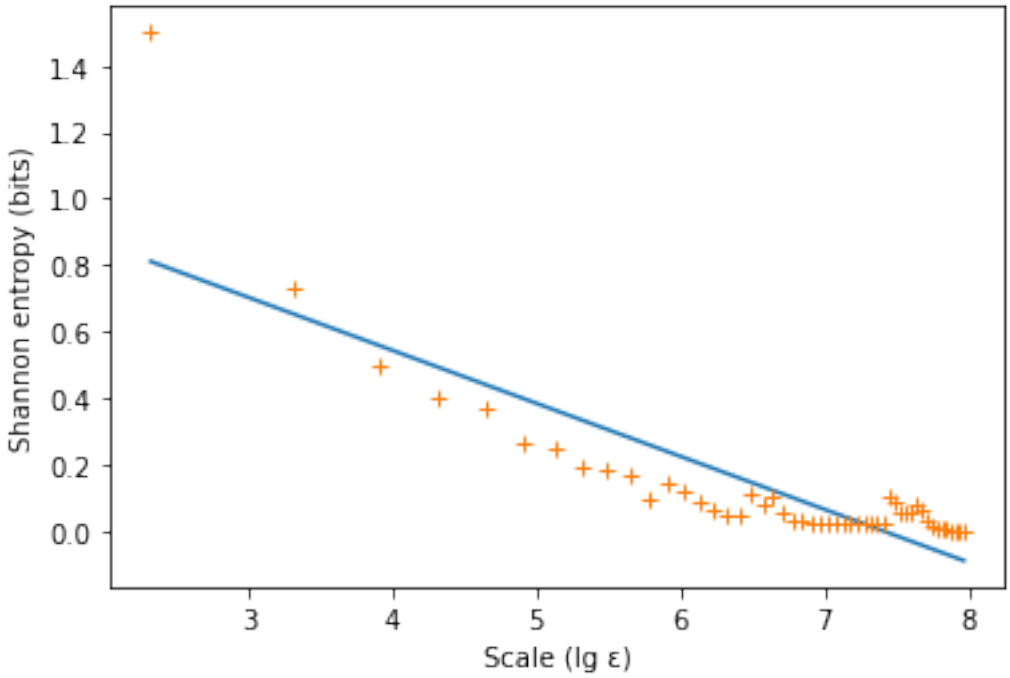
```
def infodim(dist):
    spl = interpolate.splrep(range(len(dist)), dist, s=0) # s=2e-5
    εs = np.arange(5, len(dist) - 1, 5)
    logεs = np.log2(εs)
    endεs = logεs[[0, -1]]
    entropies = [shannon_entropy(
        interpolate.splev(np.arange(0, len(dist), ε), spl)) for ε in εs]
    dimfit = np.polyfit(np.log2(εs), entropies, 1) # [slope, intercept]
    plt.plot(endεs, dimfit[0]*endεs + dimfit[1])
    plt.plot(logεs, entropies, '+')
    plt.xlabel('Scale (lg ε)')
    plt.ylabel('Shannon entropy (bits)')
    return dimfit
```

```
dist = (10 / 256) * np.exp(-np.linspace(-5, 5, 256)**2 / 2) / np.sqrt(2*np.pi)
plt.plot(dist);
```

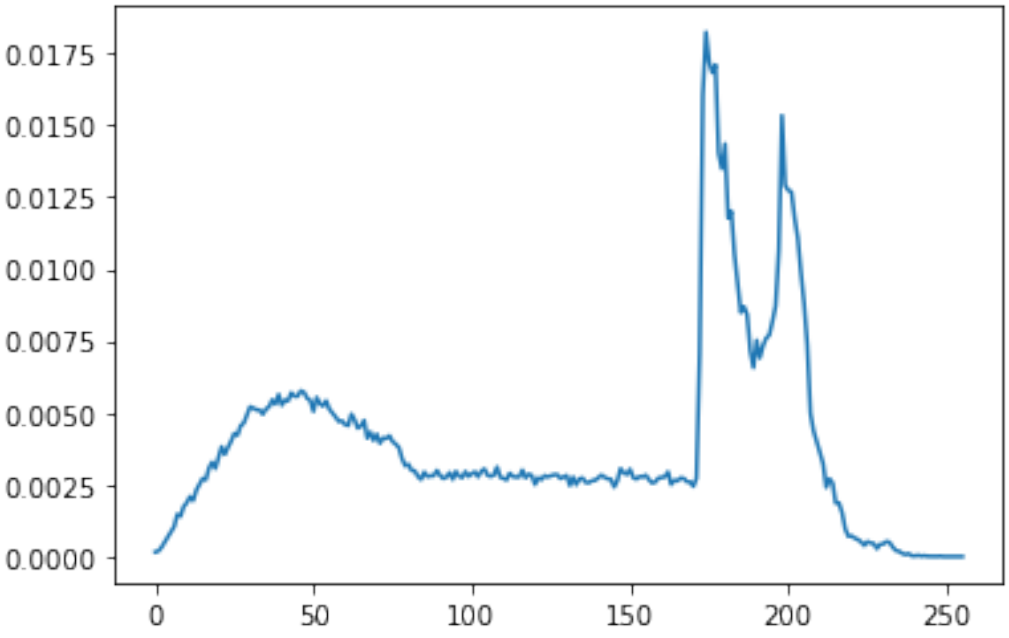



```
infodim(dist)
```

```
array([-0.1592692 ,  1.18147379])
```




```
dist = intensity_distribution(img)
plt.plot(dist);
```


0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250