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 scipy import integrate
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
```

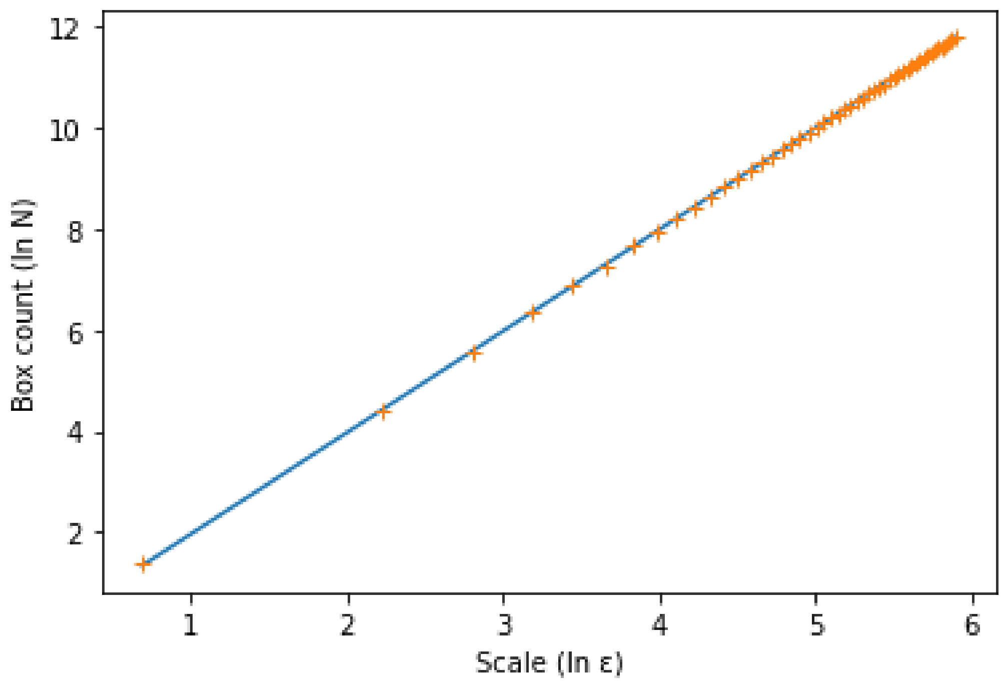
1.1 Box-counting dimension

```
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)
```

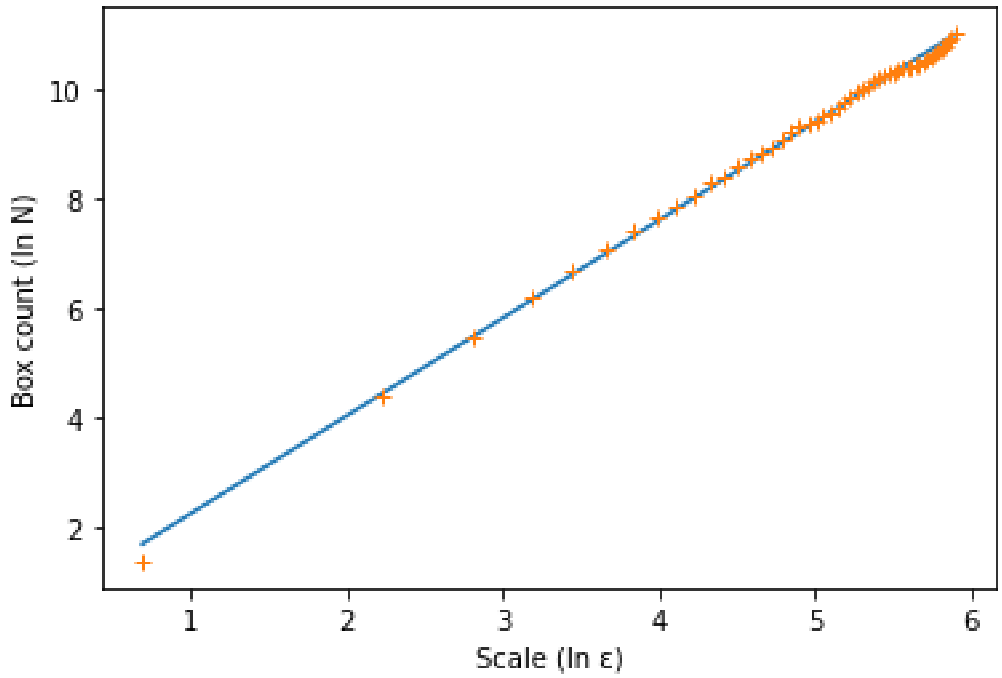
```
2.0087040269581435
```



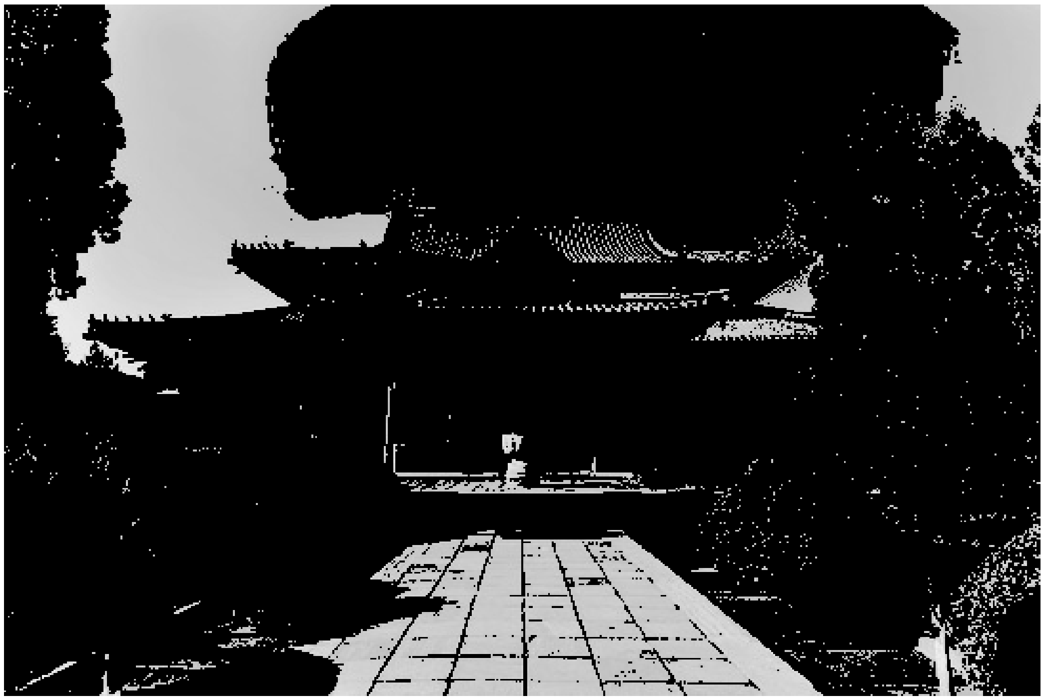

```
sky = data.copy()
sky[sky < 128+32] = 0
Image.fromarray(sky)
```

```
boxdim(sky)
```

```
1.7877778191348215
```

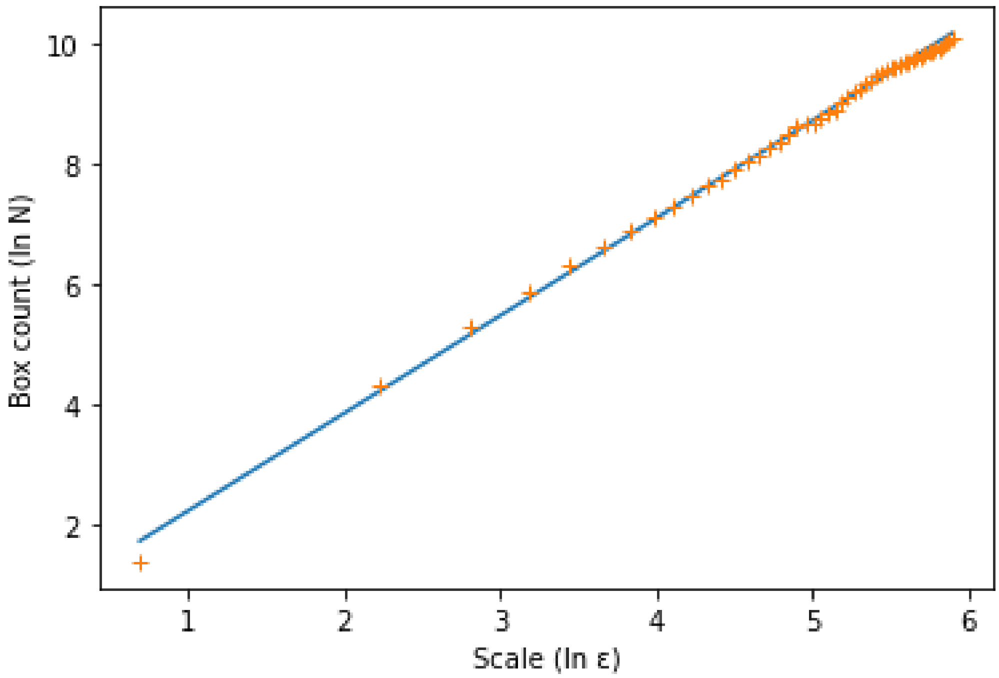



```
nosky = data.copy()
nosky[nosky < 128+64] = 0
Image.fromarray(nosky)
```



```
boxdim(nosky)
```

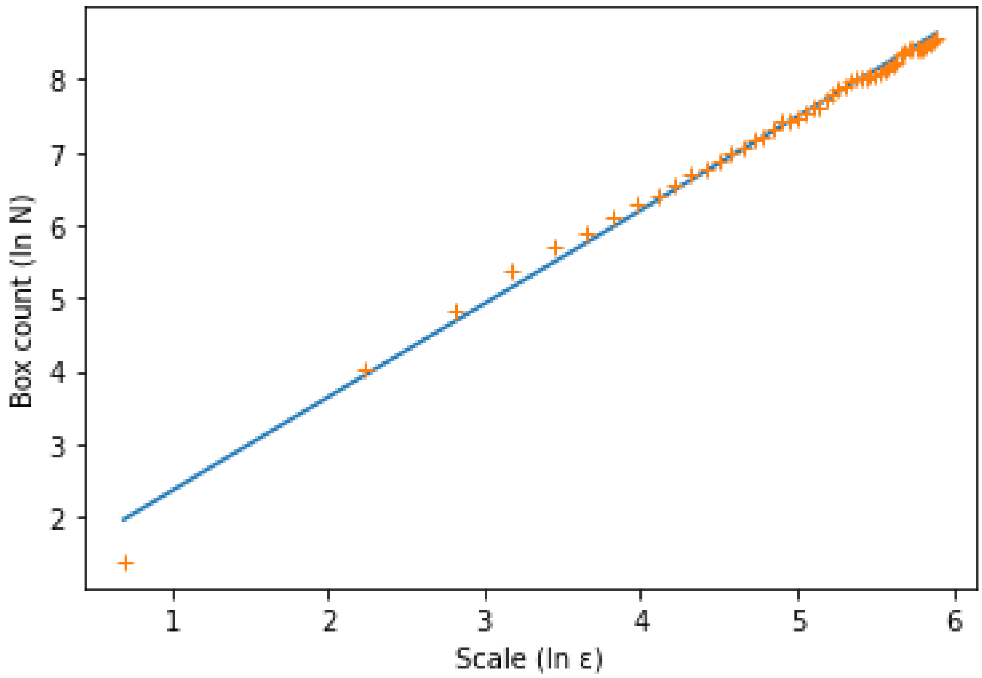
1.6214794967487127




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

```
boxdim(dots)
```

```
1.2821025677557252
```



1.2 Information dimension

```
def discretize(f, a, b, ε, N=20):
    return [integrate.simps(f(np.linspace(c - ε/2, c + ε/2, N)), dx=ε / (N - 1))
            for c in np.arange(a + ε/2, b, ε)]
```

```
def infodim(dist, s=1e-5):
    l = len(dist)
    spl = interpolate.splrep(range(l), dist, s=s)
    f = lambda x: interpolate.splev(x, spl)

    εs = l / np.linspace(10, l)
    logεs = -np.log2(εs)
    endεs = logεs[[0, -1]]
    entropies = [shannon_entropy(discretize(f, 0, l, ε)) for ε in εs]
    dimfit, cov = np.polyfit(logεs, entropies, 1, cov='unscaled')

    plt.plot(endεs, dimfit[0]*endεs + dimfit[1])
    plt.plot(logεs, entropies, '+')
    plt.xlabel('Scale (lg ε)')
    plt.ylabel('Shannon entropy (bits)')
```

```
    return dimfit[0], cov[0,0]
```

The Gaussian distribution

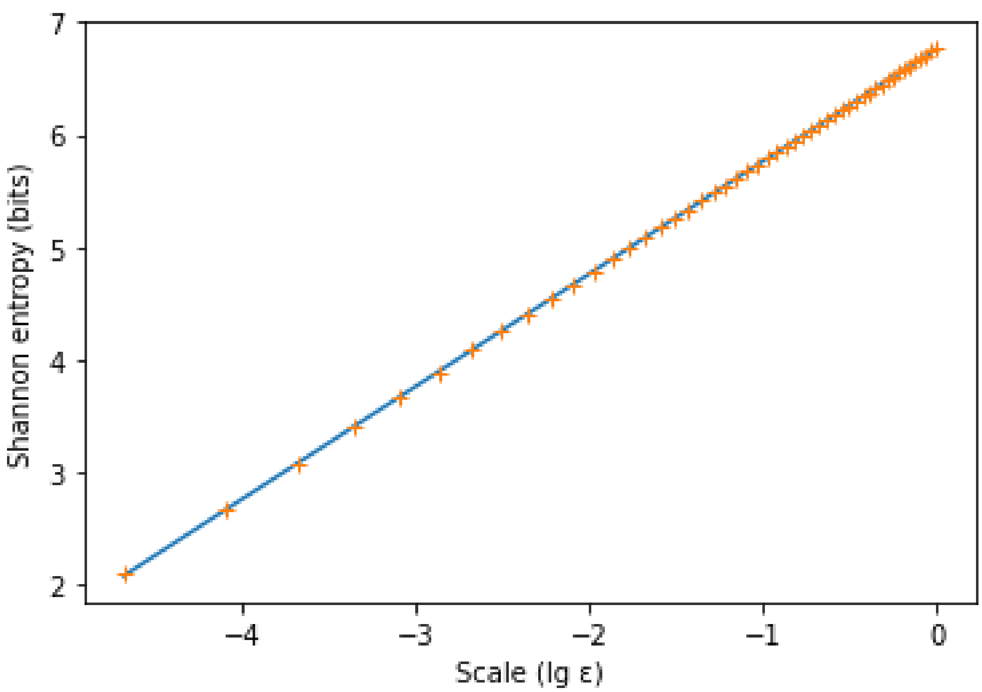
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

is continuous, so its information dimension is 1.

```
def gaussian(μ, σ, x):
    return np.exp(-(x - μ)**2 / (2*σ**2)) / (σ*np.sqrt(2*np.pi))
```

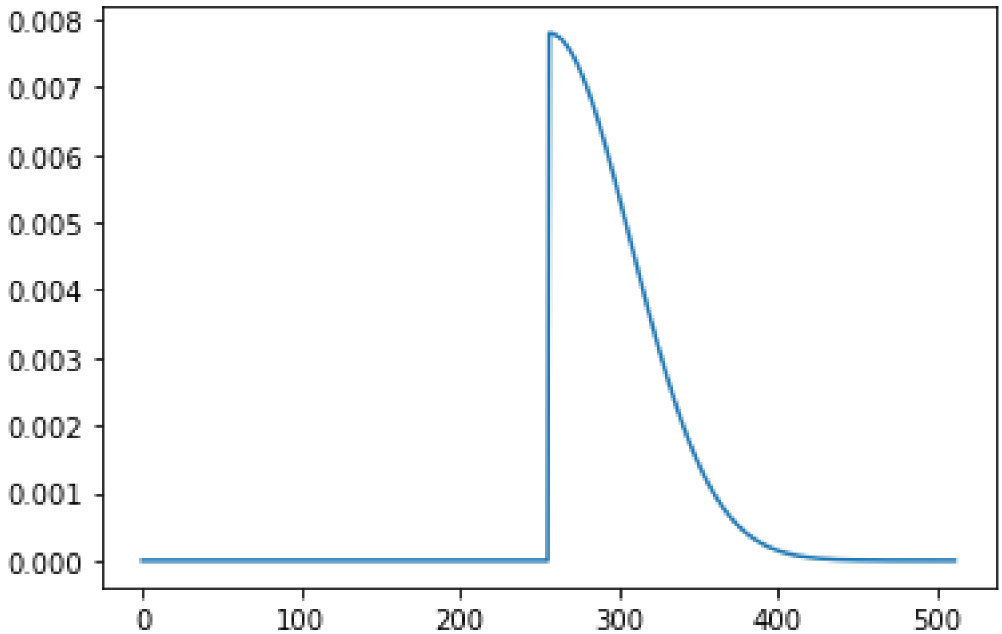
```
infodim((10/256) * gaussian(0, 1, np.linspace(-5, 5, 256)))
```

```
(1.0014184290221988, 0.015707497682893923)
```



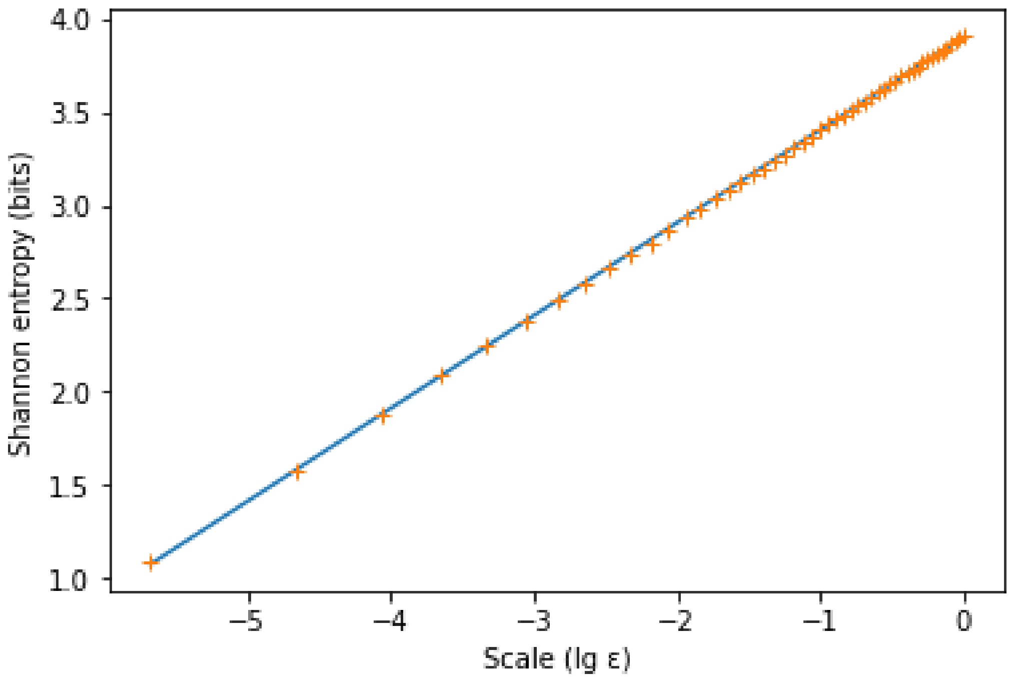
The rectified Gaussian distribution $g(x) = \Theta(x)f(x)+\delta(x)/2$ is half-continuous, so its information dimension is 1/2.

```
dist = np.concatenate([[0]*256, (5/256)*gaussian(0, 1, np.linspace(0, 5, 256))])
plt.plot(dist);
```



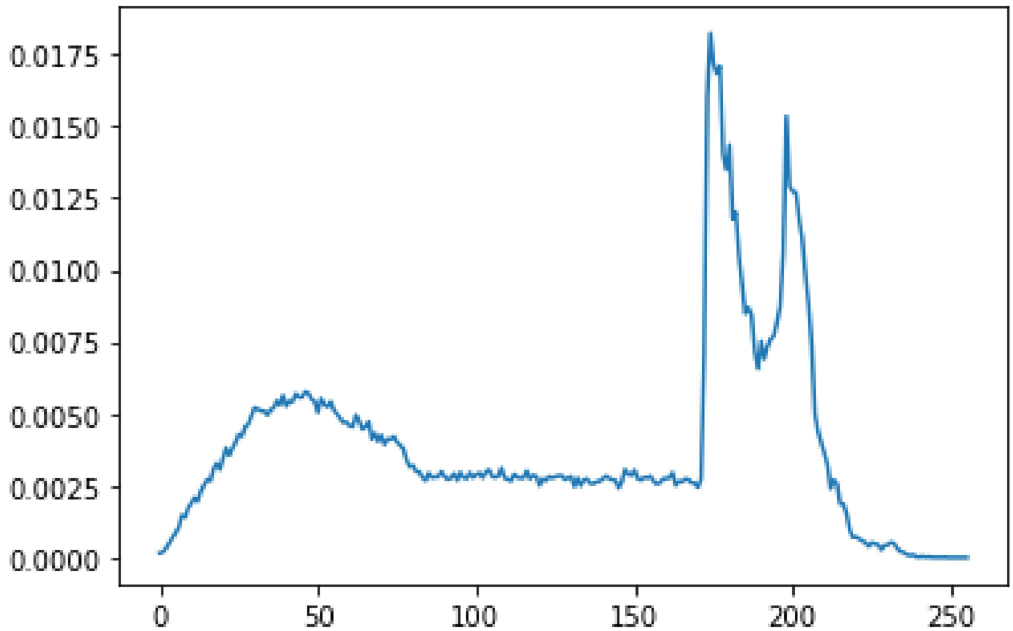

```
infodim(dist)
```

```
(0.4979088715795226, 0.012313889825394398)
```



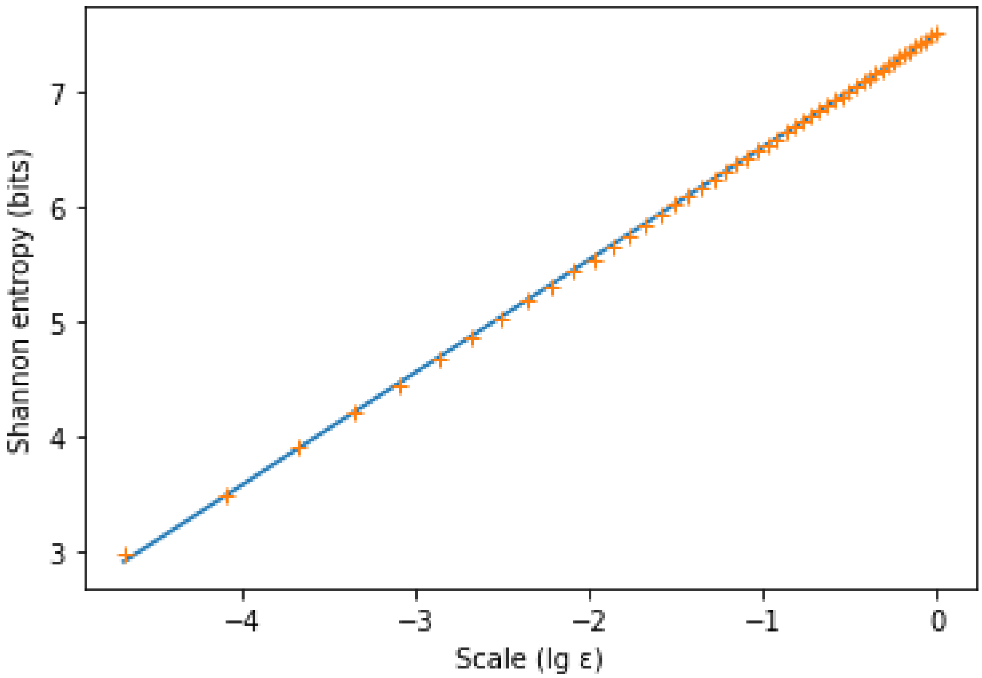
Now that we've validated `infodim`, what does it say about the intensity distribution of an image?

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




```
infodim(dist)
```

```
(0.98265843047326, 0.015707497682893923)
```


Shannon entropy (bits)
Scale (lg ε)
7
6
5
4
3
−4
−3
−2
−1
0