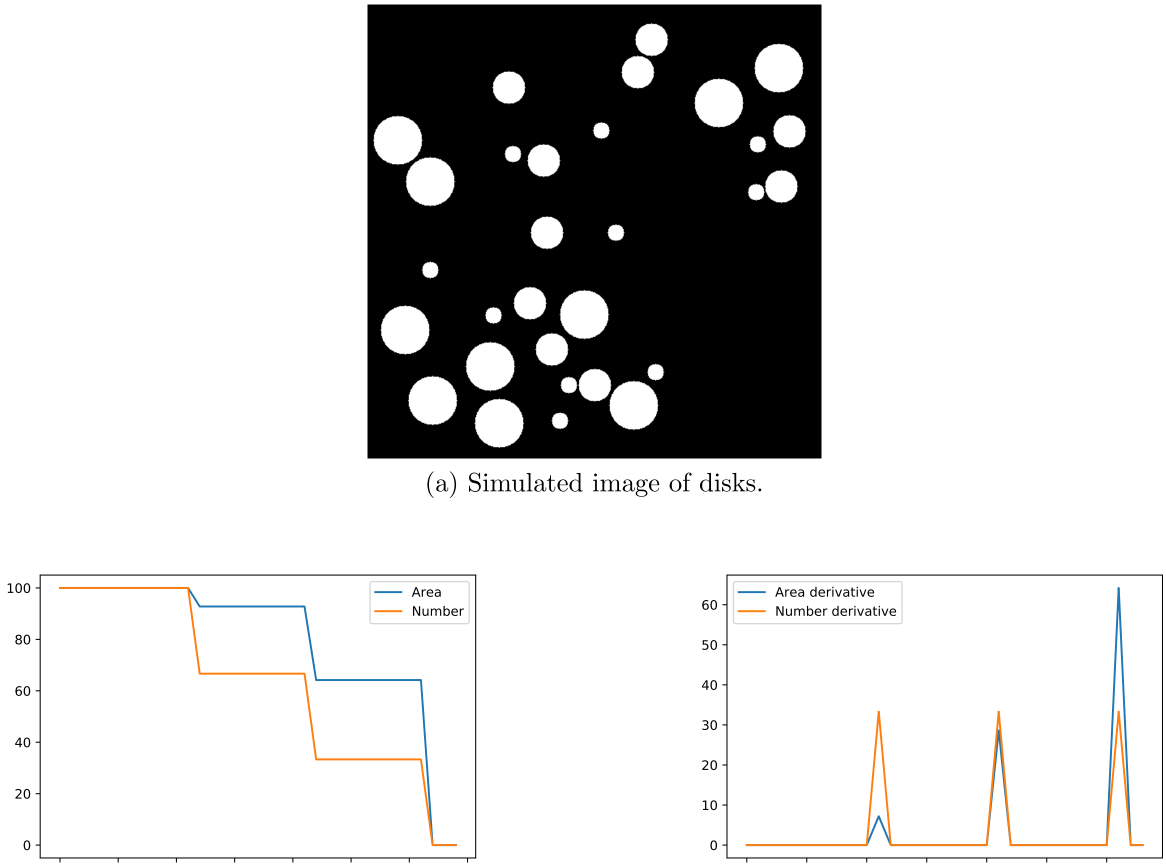
1 Python correction

Simulation The construction of an homothetic structuring element is necessary for computing a granulometry (through the function iterate_structure .

```python
def granulometry(BW, T=35):
    # total original area
    A = ndimage.measurements.sum(BW);

    # number of objects
    label, N = ndimage.measurements.label(BW);

    area=np.zeros((T,), dtype=np.float);
    number=np.zeros((T,), dtype=np.float);

    """
    Warning: the structuring elements must verify B(n) = B(n-1) o B(1).
    """
    se = ndimage.generate_binary_structure(2, 1);
    for i in np.arange(T):
        SE = ndimage.iterate_structure(se, i-1);
        m = ndimage.morphology.binary_erosion(BW, structure=SE);
        G = ndimage.morphology.binary_propagation(m, mask=BW);
        area[i]=100*ndimage.measurements.sum(G)/A
        label, n = ndimage.measurements.label(G);
        number[i] = 100*n/N; # beware of integer division

    plt.figure()
    plt.plot(area, label='Area')
    plt.plot(number, label='Number')
    plt.legend()
    #plt.savefig("granulo_poudre1.pdf");
    plt.show()

    plt.figure();
    plt.plot(-np.diff(area), label='Area derivative');
    plt.plot(-np.diff(number), label='Number derivative');
    plt.legend()
    #plt.savefig("granulo_poudre2.pdf");
    plt.show()
```

The results are shown in Fig. 1, generated by the following code:

```
## Granulometry of synthetic image
# read binary simulated image, normalize it
I = misc.imread("simulation.png")/255;
I = I[:,:,2]>.5;
plt.figure();
plt.imshow(I);
plt.show();

granulometry(I, 35);
```



(a) Simulated image of disks.

(b) Granulometry in number and area.

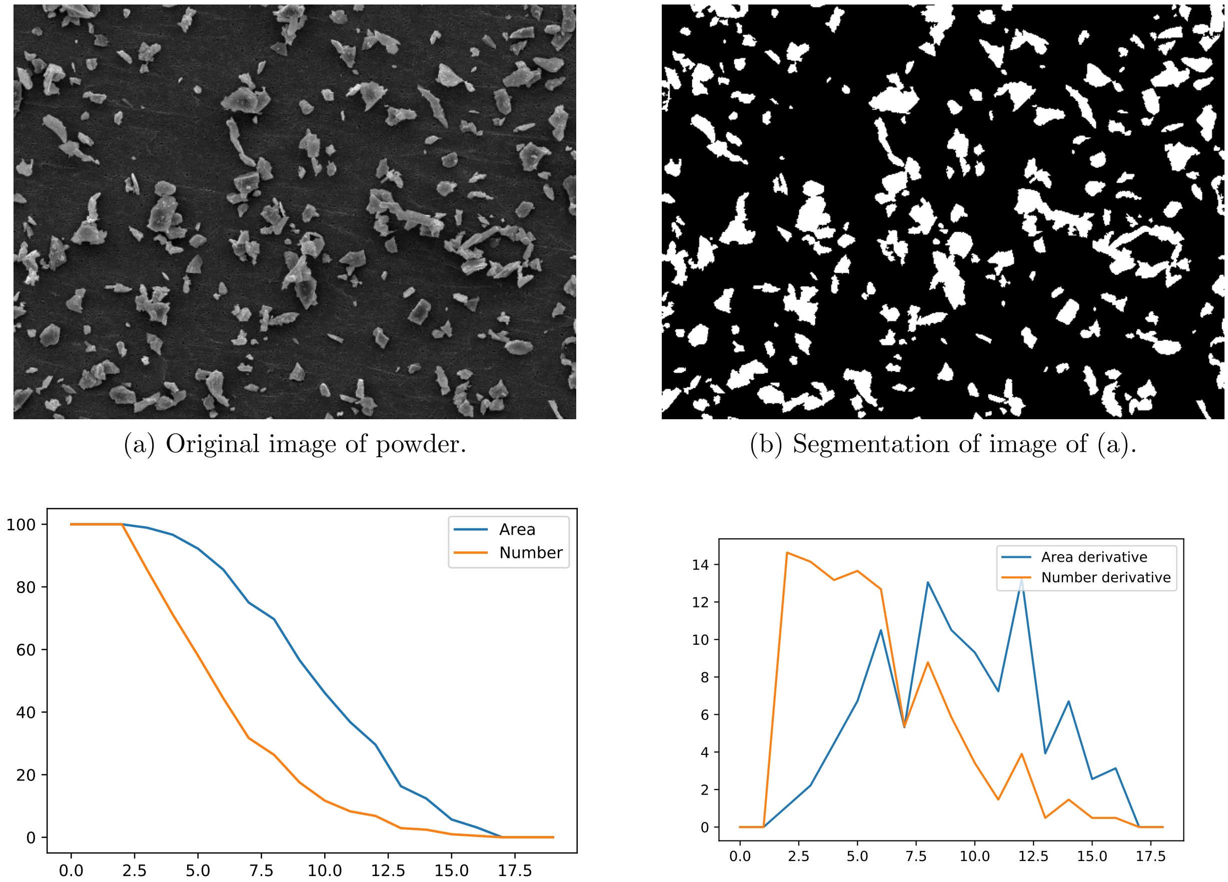
(c) Derivatives.

Figure 1: Granulometry on simulated image.

1.1 Powder image and segmentation

First of all, the image must be binarized, i.e. segmented. The following code is a proposition of segmentation, leading to the result presented in Fig. 2c.

```
## Granulometry of real image
I = misc.imread("poudre.bmp");

# segmentation
BW = I>74;
BW = ndimage.morphology.binary_fill_holes(BW);

# suppress small objects
se = ndimage.generate_binary_structure(2, 1);
m = ndimage.morphology.binary_opening(BW);
# opening by reconstruction
BW=ndimage.morphology.binary_propagation(m, mask=BW);

misc.imsave("segmentation.png", BW);
plt.imshow(BW)
plt.show()

granulometry(BW, 20);
```



(a) Original image of powder.

(b) Segmentation of image of (a).

(c) Granulometry.

(d) Derivative.

Figure 2: Results of granulometry analysis of original image from (a).