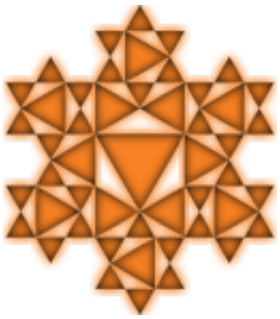


# Fractalyse 3.0

# User manual

Gilles Vuidel and Cécile Tannier

2022-01-17

# Contents


**I Generalities** 2

**1 Introduction** 3
- 1.1 About Fractalyse 3.0 . . . 3
- 1.2 Authors . . . 3
- 1.3 Terms of use . . . 3
- 1.4 System requirements . . . 3
- 1.5 How to install Fractalyse on your PC? . . . 3

**2 Input data** 4

**3 Fractal analysis methods** 5
- 3.1 Monofractal analysis . . . 5
  - 3.1.1 Box counting analysis . . . 6
  - 3.1.2 Dilation analysis . . . 6
  - 3.1.3 Correlation analysis . . . 7
  - 3.1.4 Radial analysis . . . 7
- 3.2 Multifractal analysis . . . 8
  - 3.2.1 Box counting multifractal analysis . . . 11
  - 3.2.2 Sandbox . . . 11
- 3.3 Network analysis . . . 11

**II Graphical user interface** 13

**4 Data** 14
- 4.1 Loading data: `File` menu . . . 14
  - 4.1.1 `Load vector data` menu . . . 14
  - 4.1.2 `Load raster data` menu . . . 15
  - 4.1.3 `Load network graph` menu . . . 16
  - 4.1.4 `Load estimation` menu . . . 17
  - 4.1.5 Other menus . . . 17
- 4.2 Data handling: `Tools` menu . . . 17
  - 4.2.1 `Rasterize` menu . . . 17
  - 4.2.2 `Binarize` menu . . . 18
  - 4.2.3 `Negative` menu . . . 18
  - 4.2.4 `Selection` menu . . . 18
  - 4.2.5 Contextual layer menu . . . 18

**5 Fractal analysis in practice** 19
- 5.1 Choice of a scale range for the analysis . . . 19
- 5.2 `Vector` menu . . . 20
  - 5.2.1 `Box counting` menu . . . 20
  - 5.2.2 `Dilation` menu . . . 21
  - 5.2.3 `Correlation` menu . . . 23
  - 5.2.4 `Radial` menu . . . 24
  - 5.2.5 `Batch` menu . . . 24
  - 5.2.6 `Multifractal` menu . . . 25

5.3 Raster menu . . . 25
5.3.1 Box counting menu . . . 25
5.3.2 Dilation menu . . . 25
5.3.3 Correlation menu . . . 26
5.3.4 Radial menu . . . 26
5.3.5 Radial all points menu . . . 27
5.3.6 Multifractal menu . . . 28
5.4 Network menu . . . 28
5.4.1 Choice of distance and mass counting . . . 29
5.4.2 Correlation menu . . . 29
5.4.3 Radial menu . . . 30
5.4.4 Service menu . . . 31
5.4.5 Backbone menu . . . 31
5.4.6 Multifractal menu . . . 31

**6 Estimation module** **32**
6.1 Monofractal . . . 32
6.2 Multifractal . . . 35
6.3 Save/reload estimation . . . 37
6.4 Estimation layer: scale range . . . 38

**7 Miscellaneous** **39**
7.1 Preferences menu . . . 39
7.1.1 Memory . . . 39
7.1.2 Processors . . . 39
7.2 Log window menu . . . 39

**III Command line interface** **40**

**8 Prerequisite** **41**
8.1 Launch Fractalyse in CLI mode . . . 41
8.2 Syntax . . . 41
8.2.1 Definition . . . 41
8.2.2 Character separator . . . 41
8.2.3 Optional parameter . . . 41
8.2.4 Command execution . . . 42

**9 Command reference** **43**
9.1 General command . . . 43
9.1.1 --help: display help . . . 43
9.2 Data manipulation commands . . . 43
9.2.1 --rasterize: convert vector to raster . . . 43
9.2.2 --binarize: convert grayscale raster to binary . . . 44
9.3 Monofractal analysis commands . . . 44
9.3.1 --boxcounting: box counting on vector data . . . 44
9.3.2 --rboxcounting: box counting on raster data . . . 44
9.3.3 --dilation: dilation on vector data . . . 45
9.3.4 --rdilation: dilation on raster data . . . 45
9.3.5 --correlation: correlation on vector data . . . 45
9.3.6 --rcorrelation: correlation on raster data . . . 45
9.3.7 SAMPLING: scale range definition . . . 45
9.3.8 Estimation parameter . . . 46
9.4 Other commands . . . 46
9.4.1 --clusters: counting clusters for vector layer . . . 46
9.5 Options . . . 46
9.5.1 -proc . . . 46
9.5.2 -mpi . . . 46

**10 Examples of commands** **47**
10.1 Data handling commands . . . . . 47
10.2 Fractal analysis commands . . . . . 47

**11 Performance tuning** **48**
11.1 Parallelism to speed up execution . . . . . 48
11.1.1 One computer: threads . . . . . 48
11.1.2 Computer cluster: mpi . . . . . 48
11.2 Memory management . . . . . 48

# Part I

# Generalities

# Chapter 1

# Introduction

## 1.1 About Fractalyse 3.0

Fractalyse is a software application for the fractal analysis of 2D patterns (sets of points). This version 3 of Fractalyse has been written in Java language, from which results the improvement of data management with GIS (Geographical Information Systems), graphical user interface and performance with parallelism. Fractalyse runs on any computer supporting Java Virtual Machine.

## 1.2 Authors

Fractalyse has been developed by Gilles Vuidel at ThéMA laboratory (CNRS – University of Franche-Comté, Besançon, France).

## 1.3 Terms of use

Fractalyse is distributed free-of-charge. Users must cite it (with the number of the version used) in their publications. The source code is available and licensed in GPLv3.

## 1.4 System requirements

Fractalyse 3.0 runs on any computer supporting Java 1.8 or later (PC under Linux, Windows, Mac, etc.). Please note that when dealing with very large data sets, the amount of RAM memory of the computer can limit the maximum image size that can be processed in a single run with Fractalyse. In addition, for some methods, processing power (CPU) will determine the speed of computing. For details, see section 7.1 and 11 below.

## 1.5 How to install Fractalyse on your PC?

Download and install Java 1.8+ from adoptium.net.
Download fractalyse-3.0.jar from sourcesup.renater.fr/www/fractalyse/ then launch it.

# Chapter 2

# Input data

Fractalyse can read vector and raster input data.

### Vector data

Fractalyse supports the vector formats *geopackage* (.gpkg), *geojson* (.geojson) and *shapefile* (.shp), and three types of geometries: points, lines, and polygons.

### Raster data

Fractalyse supports the raster formats TIFF and Ascii Grid.

Ascii grid is a text format. An image is described as a matrix preceded by a header of 5 lines:

```
ncols        5
nrows        5
xllcorner    0
yllcorner    0
cellsize     1
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
```

TIFF is a binary format commonly used for raster images. It can contain an extension named GeoTIFF, which adds spatial references to the image. Fractalyse supports TIFF images with or without GeoTIFF extension. It does not support RGB color images. Consequently, you must convert your color image into a gray scale image before analysing it with Fractalyse.

# Chapter 3

# Fractal analysis methods

Here we describe the basic principles of the fractal analysis methods integrated in Fractalyse.

## 3.1 Monofractal analysis

Fractalyse proposes four methods to estimate the (mono-)fractal dimension of a 2D object:

- box counting analysis
- dilation analysis
- correlation analysis
- radial analysis

In any case, estimating a fractal dimension involves:

- a counting process,
- an estimation process.

#### Counting process

The counting process operates step by step following an iterative logic. At each iteration step, the number of elements (polygons, pixels...) contained in a counting window is counted. From one step to the next, the size of the counting window is enlarged. Thus two elements vary according to the counting step (iteration step) (i):

- the number of counted elements ($N_i$),
- the size of the counting window ($r_i$).

The series of points ($r_i$, $N_i$) is plotted on a two-dimension graph. The Y-axis corresponds to the number of counted elements ($N_i$) and the X-axis corresponds to the size the counting window ($r_i$), which increases from step to step.

#### Estimation process

The series of points ($r_i$, $N_i$) forms a curve (named the empirical curve). This empirical curve is fitted with another one, the estimated curve. If the empirical curve follows a (mono-)fractal law, the estimated curve has the form of a power law (parabolic or hyperbolic):

$$N = r^D$$

or

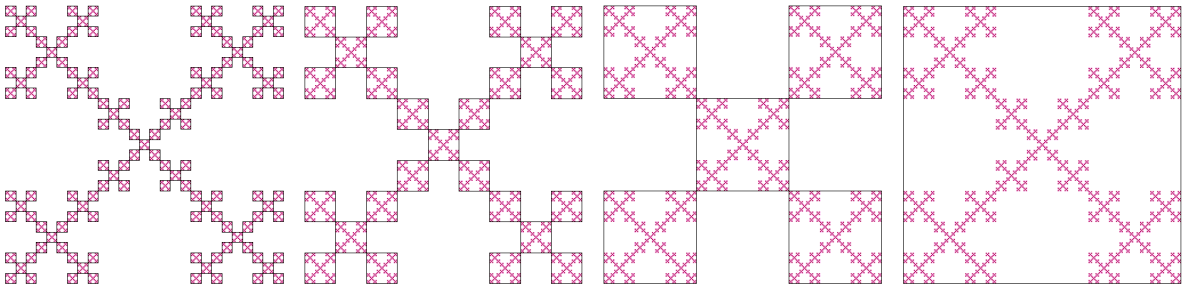
$$N = r^{-D}$$

where $D$ is the fractal dimension. Usually, the empirical curve is represented on a log-log plot. Thus the estimation is a linear regression and $D$ is the slope of the resulting estimated straight line.

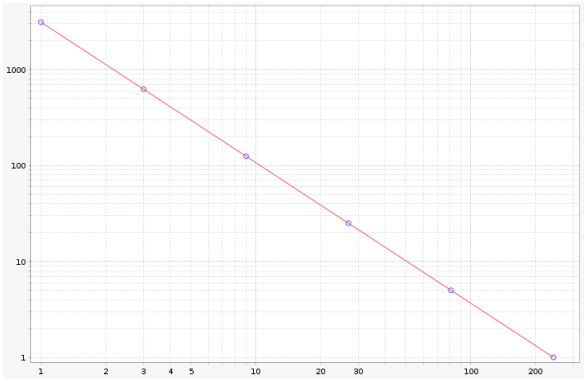
$$\log N = \log r^D = D \log r$$

### 3.1.1 Box counting analysis

Box counting analysis is the simplest method that can be used to estimate a fractal dimension. The pattern under analysis is covered by a quadratic grid. The size of the grid cells $r_i$ increases from one iteration step to the next. Considering each grid cell size $r_i$, the number of cells $N_i$ containing at least one point (black pixel in case of raster data; point, line or polygon in case of vector data) is counted.



$r : 9$ $N = 125$ $r : 27$ $N = 25$ $r : 81$ $N = 5$ $r : 243$ $N = 1$

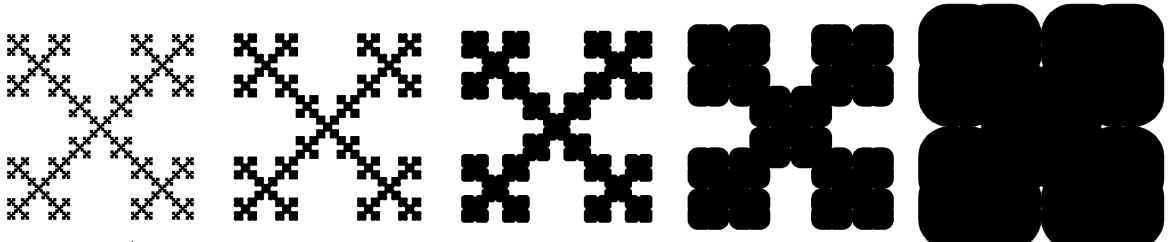



$D \sim 1.465$

### 3.1.2 Dilation analysis

Each point of the pattern (black pixel in case of raster data; polygon in case of vector data) is surrounded by a square (or a circle) of size $r_i$, the surface of which is considered to be completely occupied. These squares or circles are gradually enlarged, and the total area $A(r_i)$ covered at each stage $i$ of the dilation process is measured. As the squares are enlarged, any details smaller than $r_i$ are overlooked. More and more squares overlap and the total occupied area $A(r_i)$ for a particular value $r_i$ is lower than it would be if the same number of points that make up the original shape would have been surrounded individually. By dividing this total area by the area of a square ($r_i^2$) or a circle ($\pi \frac{r_i}{2}^2$), we obtain an approximation of the number of elements $N_i$ required to cover the whole pattern.

$$N_i \approx \frac{A(r_i)}{r_i^2} \text{ or } \frac{A(r_i)}{\pi \frac{r_i}{2}^2}$$

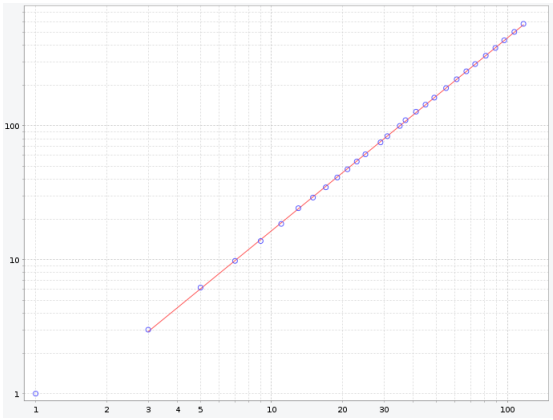




$D \sim 1.454$

### 3.1.3 Correlation analysis

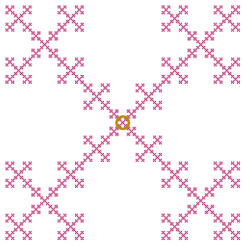
Each point of the pattern is surrounded by a small square (or circle) window. The number of points (black pixels in case of raster data) within each window is counted. Then the mean number of points per window size is calculated ($N_i$). The same operation is applied for windows of increasing sizes $r_i$. In principle, it is possible to choose any shape for the counting window, such as circle, hexagon, etc. Yet in Fractalyse, the counting window is a square for raster data, which helps us to avoid rounding errors, and a circle for vector data. Note: because the correlation analysis considers the simultaneous presence of two points at a certain distance, i.e. the mean distance between a pair of points, the correlation dimension is a second order fractal dimension.



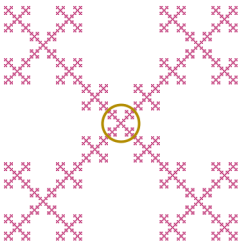

$D \sim 1.44$

### 3.1.4 Radial analysis

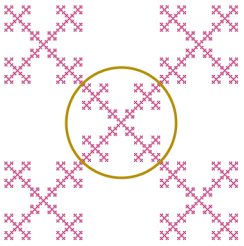
The radial analysis is identical to the correlation analysis but only one counting point is considered. The counting point is located anywhere in the pattern (most often at the centre of the pattern). A counting window is drawn around the counting point. Its size grows at each iteration step $r_i$. The number of points $N_i$ within the window is counted.
As for the correlation analysis, the shape of the counting window is a square for raster data and a circle for vector data.



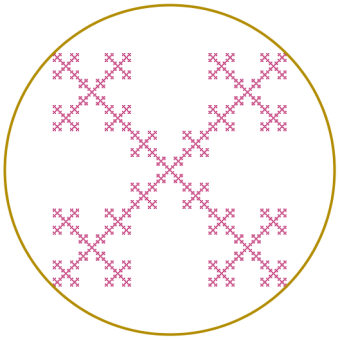
$r : 9\sqrt{2}\ N = 25$



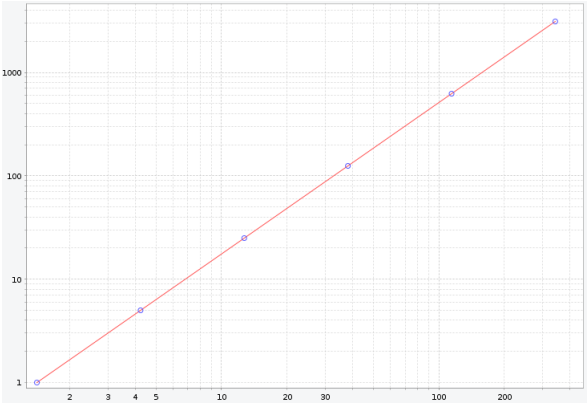
$r : 27\sqrt{2}\ N = 125$



$r : 81\sqrt{2}\ N = 625$



$r : 243\sqrt{2}\ N = 3125$




$D \sim 1.465$

## 3.2 Multifractal analysis

Whereas monofractal analyses are performed on binary images representing the presence or absence of a phenomenon (e.g. built-up surfaces), multifractal analyses also take into account the relative amount of the phenomenon in each place. The intensity of the phenomenon under consideration varies with the spatial aggregation level. Multifractal analysis can be thought of as an extension of monofractal analysis by the addition of information about the relative intensity in each place of the phenomenon under consideration. It integrates not just a series of nested spatial resolutions (i.e. analyzing the phenomenon using spatial units of nested sizes), as monofractal analysis does [Frankhauser(1998), Stanley et al.(1999)], but also a series of points of view about the quantity of information contained in each spatial unit [Sémécurbe et al.(2016)].

Multifractal analysis consists in analysing the spatial distribution of the measure $\mu_{i,r}$ for a series of scales $r$ and then to observe the result of the function when $r$ changes. Obviously, a single function is not enough to fully characterize the spatial distribution of the measure $\mu_{i,r}$. Alternatively, a group of functions, namely the structure functions $M_{r,q}$, are suitable for this:

$$M_{r,q} = \sum_i {\mu_{i,r}}^q$$

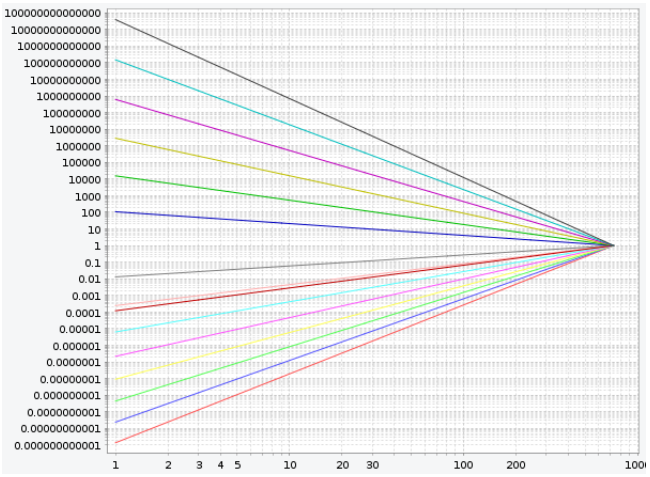



Figure 3.8: A series of structure functions $M_{r,q}$

Multifractal analysis requires us to set the assumption that the structure functions $M_{r,q}$ follow a scaling law when the values of $r$ are small:

$$M_{r,q} \approx r^{\tau_q}$$

$\tau_q$ is a scaling exponent. It can be calculated using the following formula:

$$\tau_q = \lim_{r \to 0} \frac{\log M_{r,q}}{-\log r}$$

Of course, it is impossible to reduce $r$ toward 0 indefinitely. Consequently, the estimation of $\tau_q$ is given by the slope of the curve of $\log M_{r,q}$ as a function of $\log r$.

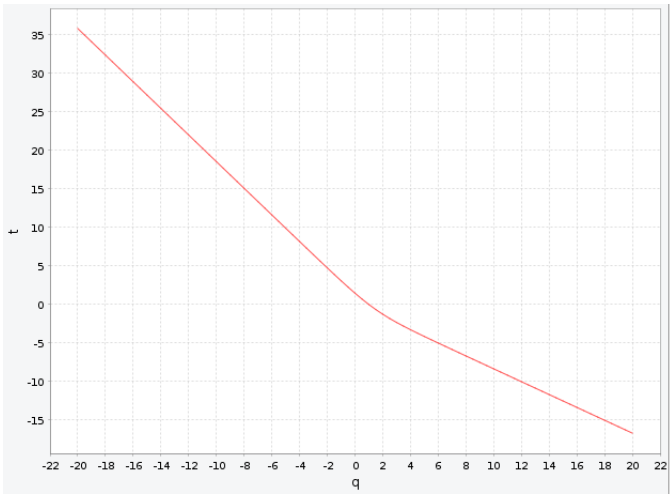



Figure 3.9: $\tau_q$

On the basis of $\tau_q$, it is possible to define the generalized dimensions $D_q$:

$$D_q = \frac{1}{1-q}\tau_q \text{ for } q \neq 1$$

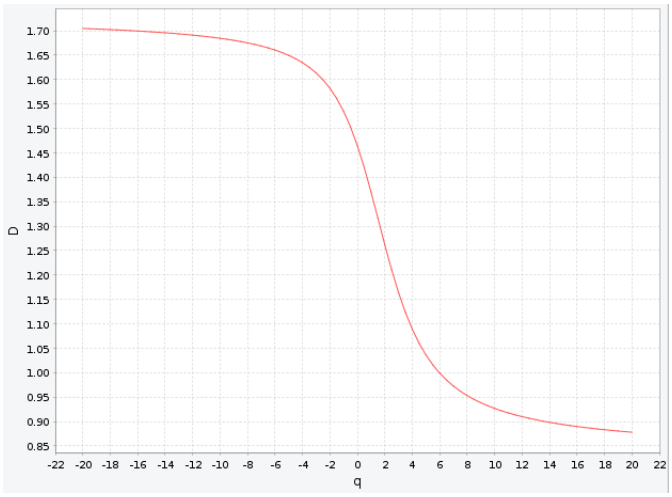



Figure 3.10: Dimension spectrum $D_q$

### Dimension spectrum

$D_q$ with respect to $q$ is called the dimension spectrum. The coefficient $q$ acts as a filter when calculating the generalized dimensions. For $q \to +\infty$, only the cells in which high measures are concentrated are taken into account. Conversely, for $q \to -\infty$ generalized dimensions describe the spatial distribution of the elements in which low measures are concentrated. For $q = 1$, all cells have the same importance whatever their measure. For $q = 0$, the generalized dimension does not take into account the spatial variation of the measure; only the fact that the cells are populated or not is taken into account. Because of this sort of filtering operated by $q$, generalized dimensions $D_q$ allow a characterization of the spatial distribution of population that is both multiscale (via the scale $r$) and multi-viewpoint (via the exponent $q$) [Sémécurbe et al.(2016)].

### Singularity spectrum $f(\alpha_q)$ (also called multifractal spectrum)

The multifractal formalism enables the determination of the singularity spectrum through a Legendre transform of the scaling exponent $\tau_q$ [Jaffard et al.(2007)]. $f(\alpha_0)$ gives the maximum value of the singularity spectrum. It indicates the strength of the irregularities of the measure. Complementarily, the range of the spectrum informs us about the multifractality of the spatial distribution of the measure (more or less strong, even non-existent in the case of a monofractal distribution) [Wendt et al.(2007)].

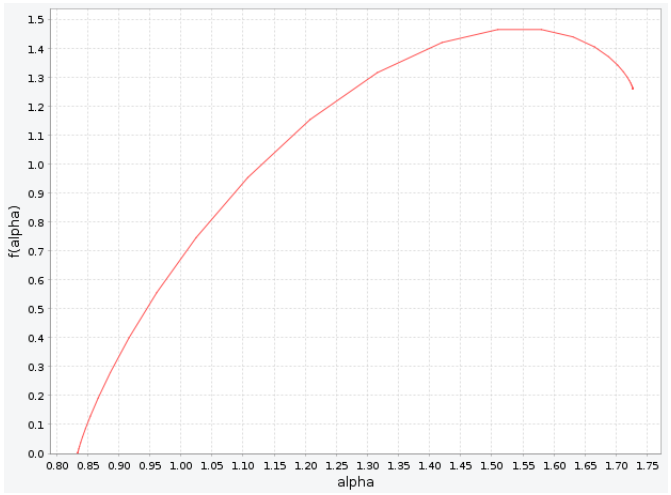



Figure 3.11: Singularity spectrum $f(\alpha_q)$

Case studies in spatial analysis are highly dependent on the size of the spatial units used for the analysis. This has been expressed by [Openshaw(1983)] as the Modifiable Areal Unit Problem (MAUP): whatever the phenomenon under consideration, it is impossible to identify a single spatial partition that would be most appropriate for analyzing it. Interestingly, both spectrums $D_q$ and $f(\alpha_q)$ enlighten the MAUP [Sémécurbe et al.(2016)].

- The absence of MAUP means that the spatial distribution of population is almost uniform. In this case, the dimension spectrum is a horizontal line with the y-intercept equal to 2 and the singularity spectrum is the point of coordinates $(2, 2)$.
- When the measure is monofractal then the singularity spectrum $f(\alpha_q)$ is only a point of coordinates $\alpha_q = f(\alpha_q) = D_0$. In this case, the MAUP only results from the spatial distribution of the populated places. Basically, the study area contains identical buildings all having the same number of inhabitants and being spatially distributed in a monofractal manner. In this case, local differences in population density come only from the spatial distribution of the buildings.
- A singularity spectrum having a range of $\alpha_q$ values indicates a multifractal behavior. In this case, the spatial footprint of the buildings varies (e.g. small single-family houses, multi-family houses, high and large collective buildings) as well as their number of inhabitants: the MAUP results from both.

### 3.2.1 Box counting multifractal analysis

The counting method follows the same principle as the monofractal box counting method (3.1.1) except that $\mu_{i,r}$ measures here the proportion of elements in the grid cell $i$ of size $r$ and not just the fact that the grid cell is occupied or not.

For $q = 1$ $D_q$ is:

$$D_1 = \lim_{r \to 0} \frac{\log \sum_i \mu_{i,r} \log \mu_{i,r}}{\log r}$$

When $q = 0$, the function $M_{r,q}$ corresponds to the mono-fractal box counting function, i.e. the number of occupied cells is simply counted whatever the proportion of elements within each grid cell.

### 3.2.2 Sandbox

The counting method follows the same principle as the monofractal correlation method (3.1.3).
Since the sandbox dimension is a second order fractal dimension, $q$ exponent in $M_{r,q}$ is shifted:

$$M_{r,q} = \frac{1}{n} \sum_{i=1}^{n} {\mu_{i,r}}^{q-1} = \langle {\mu_{i,r}}^{q-1} \rangle$$

When $q = 2$ the function $M_{r,q}$ is equivalent to the mono-fractal correlation dimension.

For $q = 1$ $D_q$ is:

$$D_1 = \lim_{r \to 0} \frac{\langle \log \mu_{i,r} \rangle}{\log r}$$

## 3.3 Network analysis

Correlation (3.1.3), radial (3.1.4) and sandbox (3.2.2) analyses have been adapted to deal with network data. In this case, the size $r_i$ measured is not a euclidean distance but a network distance (topological distance, metric length or any cumulative cost). The counted elements $N_i$ can be the edges or the nodes of the graph as chosen by the user.

# Bibliography

[Frankhauser(1998)] Pierre Frankhauser, 1998. The fractal approach. a new tool for the spatial analysis of urban agglomerations. *Population*, 10(1) 205–240.

[Jaffard et al.(2007)] Stéphane Jaffard, Bruno Lashermes, and Patrice Abry, 2007. Wavelet leaders in multifractal analysis. In Tao Qian, MangI Vai, and Yuesheng Xu, editors, *Wavelet Analysis and Applications*, Applied and Numerical Harmonic Analysis, pages 201–246. Birkhäuser Basel.

[Openshaw(1983)] Stan Openshaw, 1983. The modifiable areal unit problem. *Geo Books*, 38.

[Stanley et al.(1999)] HE Stanley, LA Nunes Amaral, AL Goldberger, S Havlin, P Ch Ivanov, and C-K Peng, 1999. Statistical physics and physiology: monofractal and multifractal approaches. *Physica A: Statistical Mechanics and its Applications*, 270(1) 309–324.

[Sémécurbe et al.(2016)] François Sémécurbe, Cécile Tannier, and Stéphane G. Roux, 2016. Spatial distribution of human population in france: Exploring the modifiable areal unit problem using multifractal analysis. *Geographical Analysis*, 48(3) 292–313.

[Wendt et al.(2007)] Herwig Wendt, Patrice Abry, G. Roux, Stéphane, and Stéphane Jaffard, 2007. Analyse multifractale d'image : l'apport des coefficients dominants. In *GRETSI 2007*. Université de technologie de Troyes (UTT) ; Université de Reims Champagne-Ardennes (URCA), GRETSI, Troyes, France.

# Part II

# Graphical user interface

# Chapter 4

# Data

## 4.1 Loading data: `File` menu

### 4.1.1 `Load vector data` menu

Fractalyse can read three types of vector data from Geographic information systems (GIS): Geopackage (.gpkg), GeoJSON (.geojson) and Shapefile (.shp).
Three types of geometries are supported: points, lines, and polygons.

When a shapefile is loaded with the menu `Load vector data`, a new layer is created and displayed in the GUI (graphical user interface).

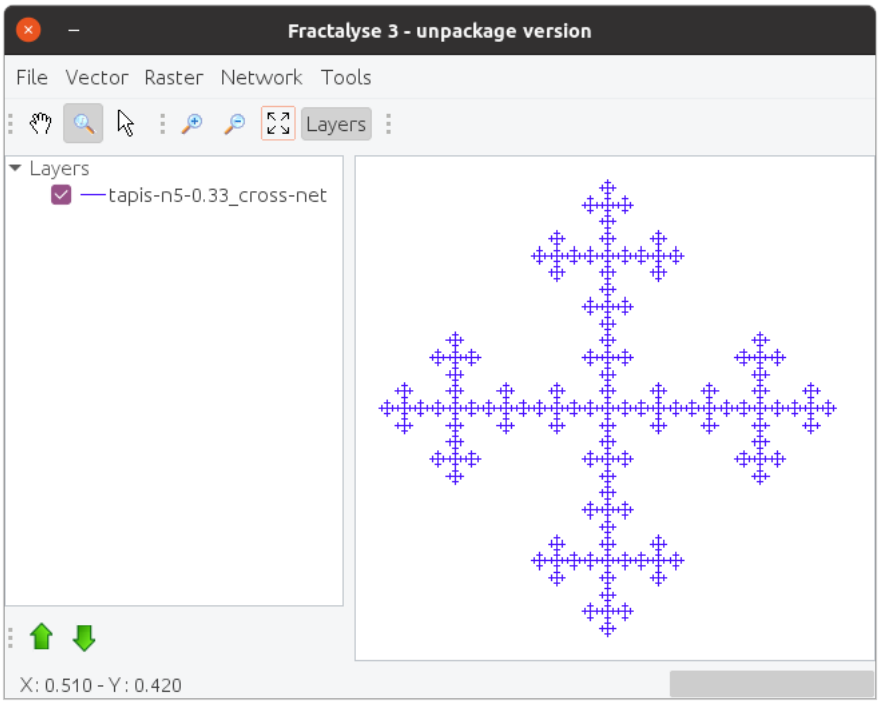



Figure 4.1: Vector data loaded in Fractalyse

#### Network vector data

If the vector layer contains only simple linear geometries, a graph representation of the network is automatically created. This allows the user to perform network analysis in addition to vector analysis.

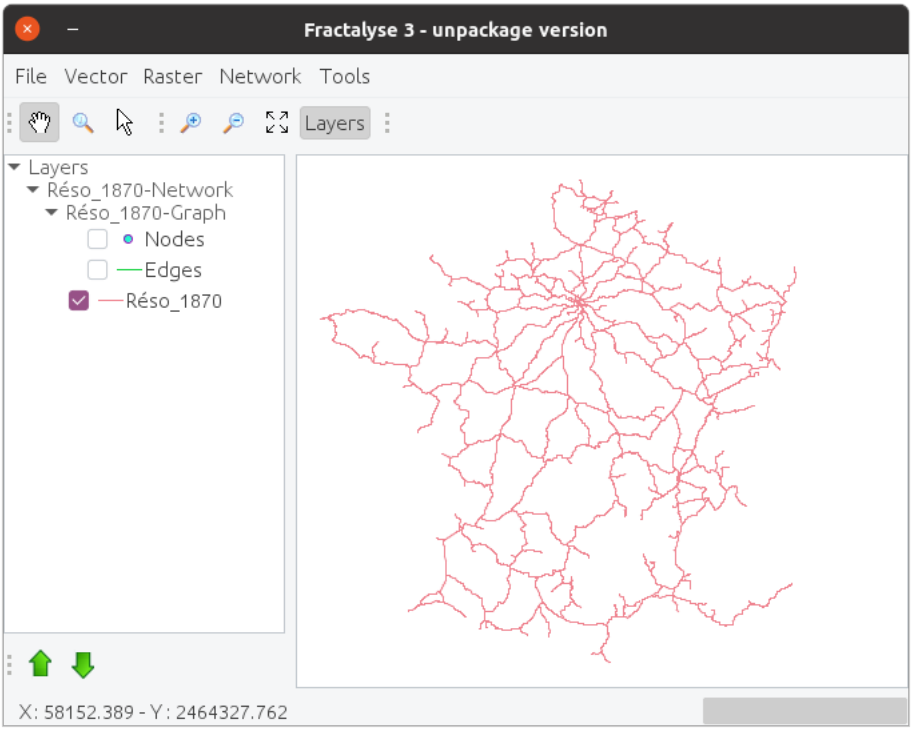



Figure 4.2: Vector network data loaded in Fractalyse

### 4.1.2 `Load raster data` menu

Fractalyse can read two types of raster data: TIFF and Ascii Grid.

When the input data file is loaded with the menu `Load raster data` menu, a new layer is created and the image is displayed. If the image contains only 0 and 1 values, Fractalyse classifies the layer as a binary layer and sets white color for 0 and black color for 1. Otherwise, it sets a gray scale color ramp for the image. Color images (RGB) are not supported.

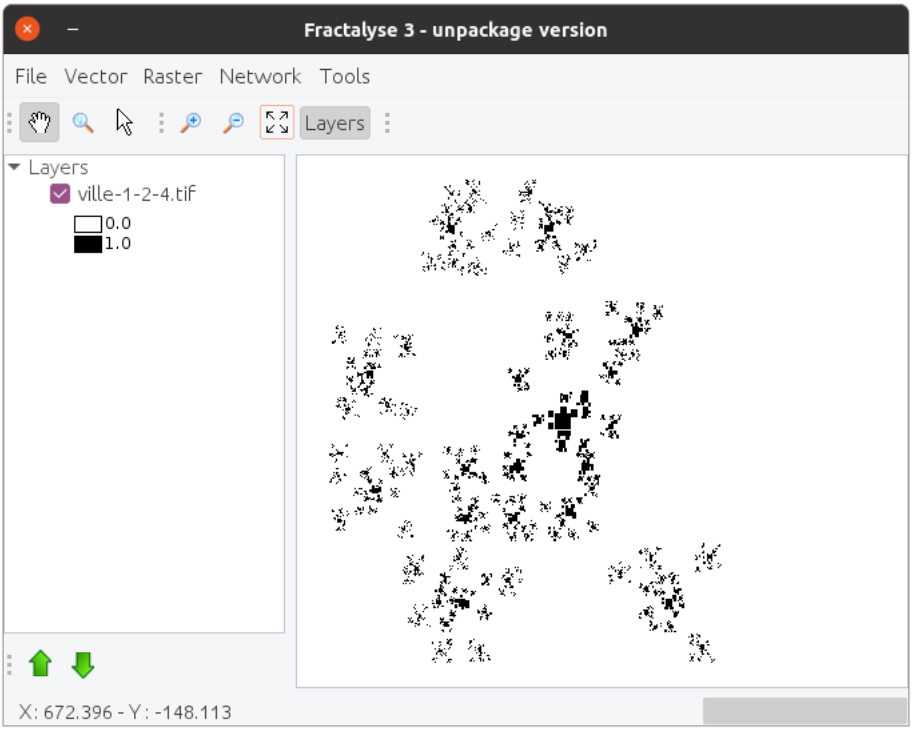



Figure 4.3: Raster binary data loaded in Fractalyse


Figure 4.4: Raster gray scale layer of data loaded in Fractalyse

### 4.1.3 `Load network graph` menu

If your network data is defined as a point layer and a table of links, you can load your network from the menu `Load network graph`.

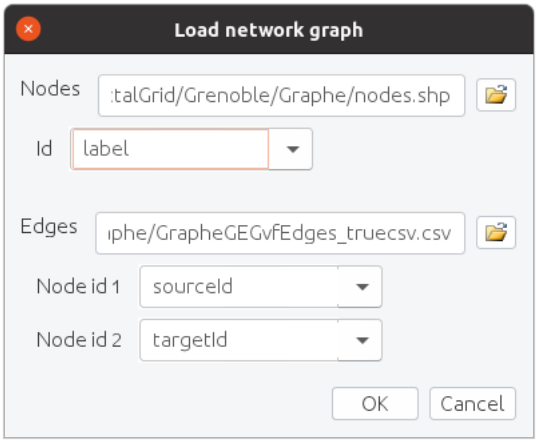



Figure 4.5: Dialog box that opens with the `Load network graph` menu

### 4.1.4 `Load estimation` menu

A previously saved estimation can be reloaded from this menu entry. For more details, see 6.3.

### 4.1.5 Other menus

Other `File` menu entries are explained in Miscellaneous chapter.

## 4.2 Data handling: `Tools` menu

In case of vector analyses, some types of data are not supported (for example, lines or polygons can not be used for a correlation analysis). When raster data are used, most analyses require binary input data. Fractalyse integrates functions in the `Tools` menu that enable the conversion of vector data into raster data (`Rasterize` menu) and the conversion of gray scale raster data into binary data (`Binarize` menu).

### 4.2.1 `Rasterize` menu

The `Rasterize` menu converts vector data into raster data. Any vector layer can be rasterized. If `Field` is not filled in, the result is a binary raster layer and a gray scale raster otherwise. If the vector layer contains polygons, the rasterization mode may be changed with `Poly mode` option.

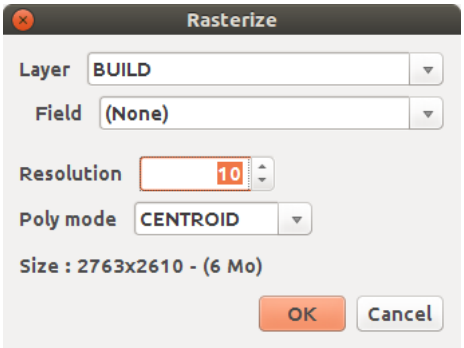



Figure 4.6: Rasterization of a vector layer

### 4.2.2 `Binarize` menu

The `Binarize` menu converts gray scale raster to black and white raster.

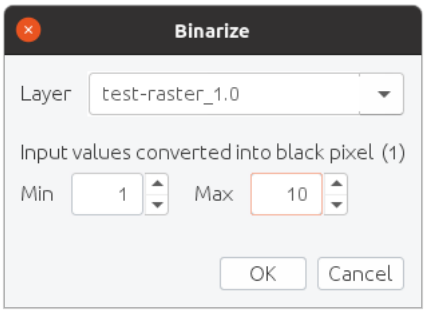



Figure 4.7: Binarization of a gray scale layer

### 4.2.3 `Negative` menu

The `Negative` menu reverses binary raster data (i.e. black and white images). The user has to right click on the layer and select the `Negative` menu. Black pixels become white and white pixels become black.

### 4.2.4 `Selection` menu

Selection menu allows to select a sub-part of a layer to analyse it separately. After selecting the region of interest by a rectangle, click on `Create` button to create a new layer containing only the selected region.

### 4.2.5 Contextual layer menu

By right-clicking on a layer, user can access a contextual menu allowing to :

- Remove: remove the current layer from Fractalyse
- Style: change the drawing style of the layer
- Export: save the current layer to a new file

# Chapter 5

# Fractal analysis in practice

Fractalyse offers three analysis menus, one for each type of input data: Vector, Raster, Network.

## 5.1 Choice of a scale range for the analysis

All fractal analysis methods require to define the range of scales that will be considered for the analysis. This sequence is given in a panel contained in each analysis frame.

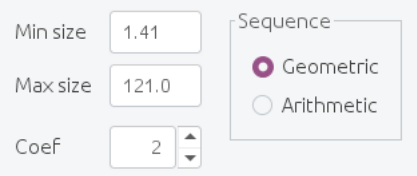



Figure 5.1: Choice of a scale range

The scale range is comprised between `Min size` and `Max size`. By default, the sequence of scales is geometric, which means that each scale value is multiplied by the value `Coef` (starting from `Min size`). When the sequence chosen is arithmetic, the value `Coef` is added to each scale value. For geometric sequence, `Coef` must be strictly greater than 1, for arithmetic sequence `Coef` must be strictly greater than 0.

`Min size` and `Max size` values are by default automatically estimated from the input data. For raster data, `Min size` is the pixel size. For vector data, `Min size` is the size of the smallest element (polygon or line segment). The value of `Max size` is set by default as the half of the minimum extent of the data layer.

For raster data, the scales have to be rounded to the pixel unit. This may result in slight deviations from the defined sequence.

## 5.2 `Vector` menu

Fractal analysis for vector data.

### 5.2.1 `Box counting` menu

The method is described in 3.1.1. The user first needs to set the vector layer under consideration and defines the scale range for the analysis. She/he also has to set the value of two specific parameters: `Number of grid positions` and `View boxes`. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

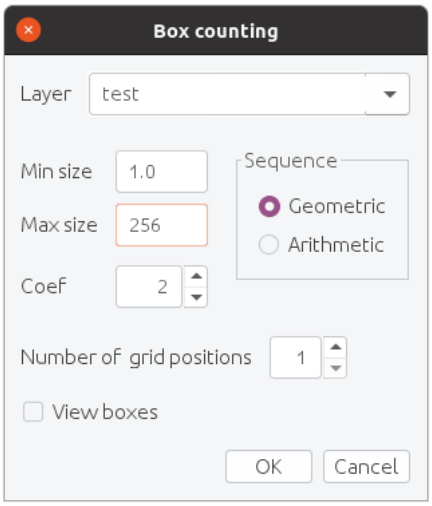



Figure 5.2: `Box counting` dialog box

`Number of grid positions`

At each iteration step, a chosen `Number of grid positions` are tested and the number of elements $N_i$ is counted for each grid position. The appropriate grid position corresponds to the lowest value of $N_i$. Note that, by the way, the position of the grid may vary from one iteration step to the next.

`View boxes`

If you check the `View boxes` option, all the boxes containing data are saved at each iteration step and can be displayed on the map; one layer for each scale. Note that this option is memory intensive since the boxes kept in memory can be numerous.

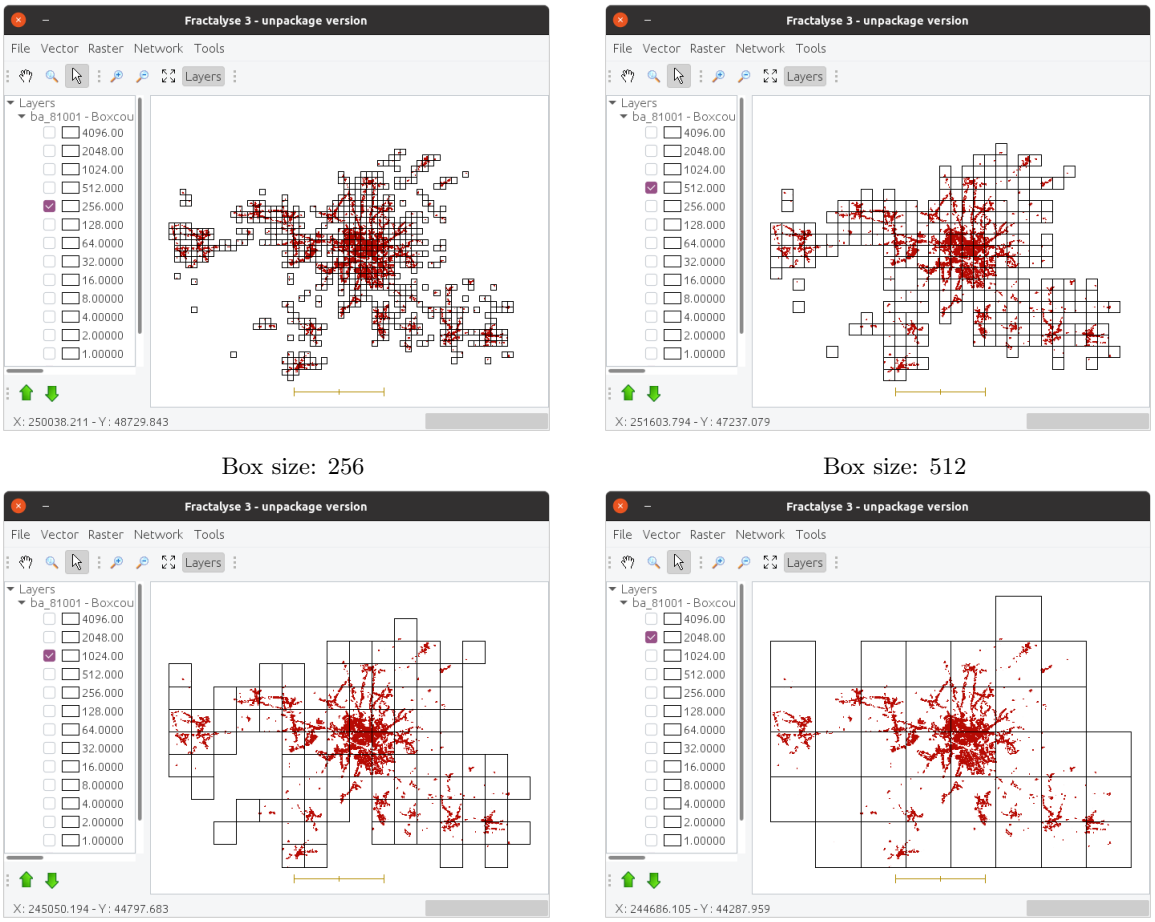



Figure 5.3: Box counting - View boxes

### 5.2.2 `Dilation` menu

The method is described in 3.1.2. The user first needs to set the vector layer under consideration and defines the scale range for the analysis. She/he also has to set the value of two specific parameters: `Stop at one cluster` and `View dilated patterns`. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

In the estimation window, the curve of the number of clusters at each dilation size can also be displayed in the `Other curve` panel.

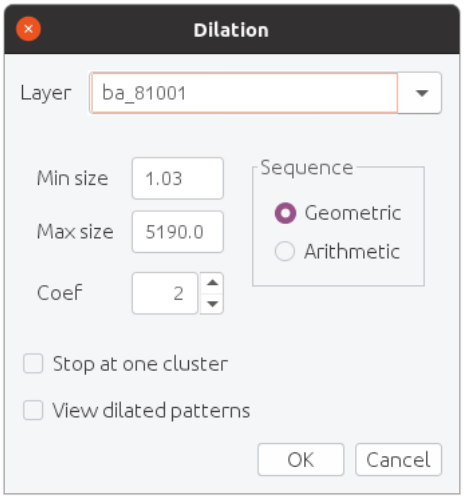


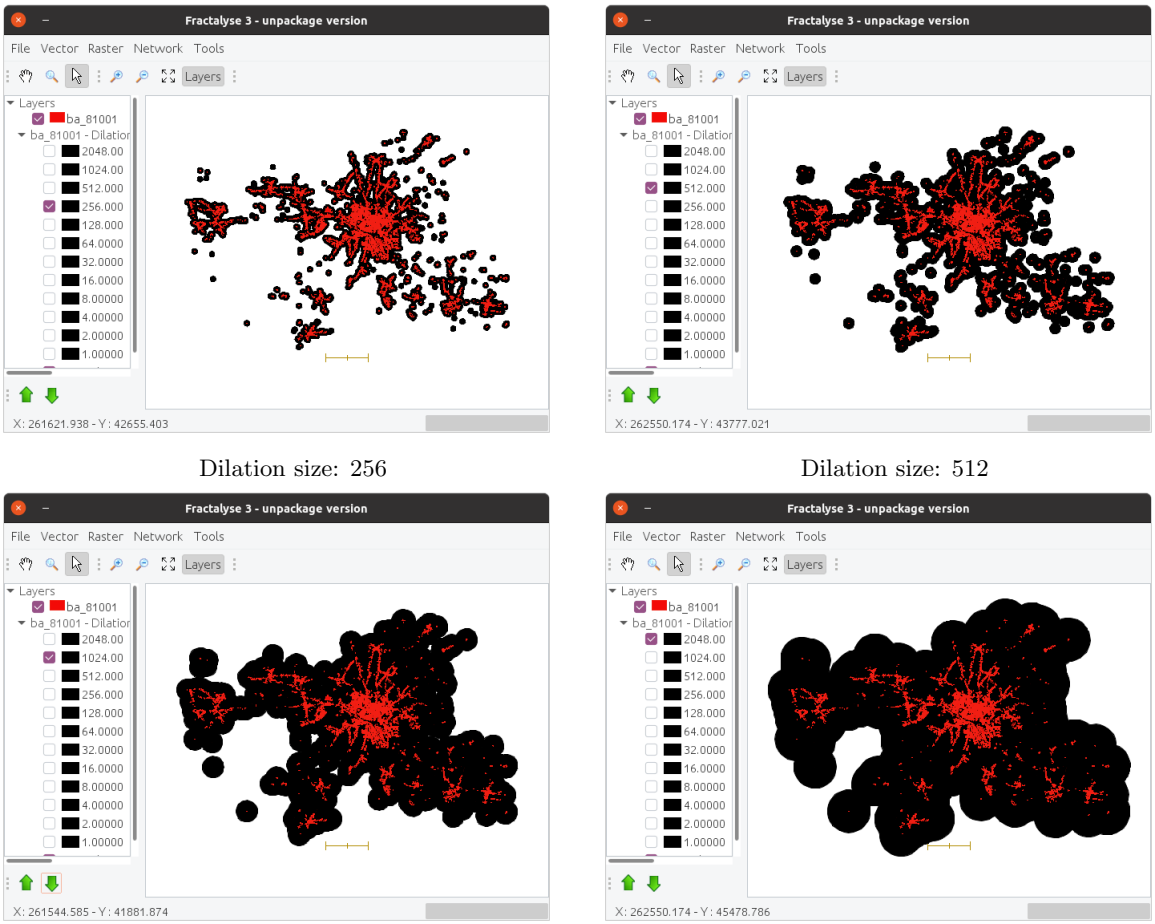

Figure 5.4: Dilation dialog box

`Stop at one cluster`

If the user selects the **`Stop at one cluster`** option, the dilation stops when the dilated pattern is made up of a single polygon. Look at the pattern dilation size 2048 on figure 5.5.

`View dilated patterns`

If the user selects the **`View dilated patterns`** option, the dilated geometries are saved at each iteration step and can be displayed on the map; one layer for each scale. Note that this option can require a large amount of memory since all dilated geometries are kept in memory.




Dilation size: 256

Dilation size: 512

Dilation size: 1024

Dilation size: 2048

Figure 5.5: Dilation - View dilated patterns

### 5.2.3 `Correlation` menu

The method is described in 3.1.3. Only point data can be analyzed with the vector correlation. The analysis of lines or polygons requires first to rasterize (see 4.2.1) the data, and then to analyse them with the raster correlation (5.3.3).
The user first needs to set the vector layer under consideration and defines the scale range for the analysis. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

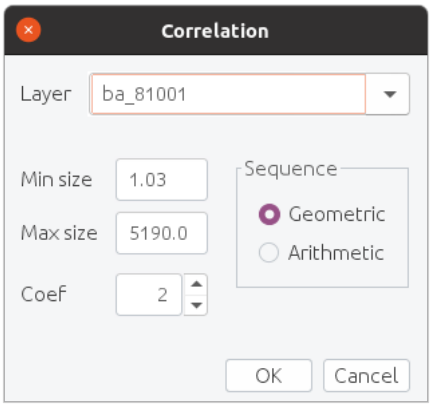



Figure 5.6: Correlation dialog box

### 5.2.4 `Radial` menu

The method is described in 3.1.4. The user first needs to set the vector layer under consideration and defines the scale range for the analysis. She/he also has to set the value of one specific parameter: `Centre` that corresponds to the starting point of the counting process. The coordinates can be entered directly or set by clicking on the map.
At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range on the map is added.
The counting unit depends on geometry type of the vector layer:

- puntal: number of points
- lineal: length
- polygonal: area

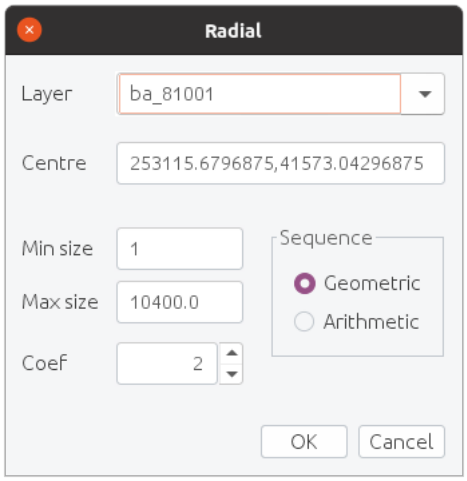



Figure 5.7: Radial dialog box

### 5.2.5 `Batch` menu

Vector methods can be executed in batch mode on each subarea of a vector layer. The subareas can be defined by a regular grid or another polygon layer.

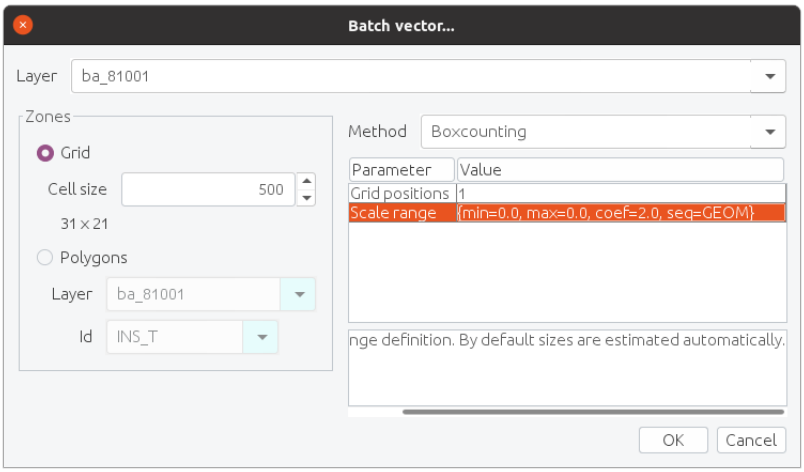



Figure 5.8: Batch dialog box

Each resulting polygon contains the fractal dimension in its attributes and also the confidence interval and the r2.

### 5.2.6 `Multifractal` menu

The principles of multifractal analysis are described in 3.2. Fractalyse implements 2 multifractal analysis: boxcounting and sandbox.

#### `Box counting` menu

The method is described in 3.2.1. The settings are exactly the same as the monofractal boxcounting analysis (see 5.2.1).
At the end of the computation, the estimated fractal dimension spectrum is displayed in the `Multi-fractal estimation` frame (see 6.2) and a new layer displaying the scale range at the bottom of the map is added.

#### `Sandbox` menu

The method is described in 3.2.2. The settings are exactly the same as the correlation analysis (see 5.2.3).
At the end of the computation, the estimated fractal dimension spectrum is displayed in the `Multi-fractal estimation` frame (see 6.2) and a new layer displaying the scale range at the bottom of the map is added.

## 5.3 `Raster` menu

Fractal analysis for raster data. Most analysis only work with binary raster except radial and multi-fractal sandbox which support binary and grayscale raster layers.

### 5.3.1 `Box counting` menu

The method is described in 3.1.1. The user first needs to set the binary raster layer under consideration and defines the scale range for the analysis. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

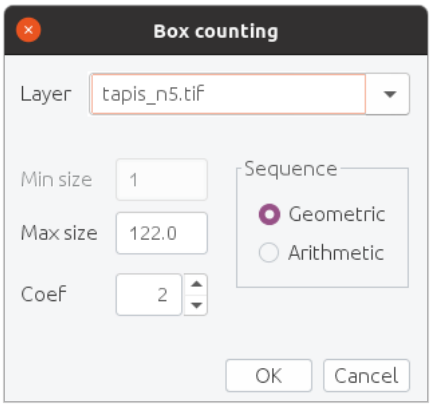



Figure 5.9: Box counting dialog box

### 5.3.2 `Dilation` menu

The method is described in 3.1.2. The user first needs to set the binary raster layer under consideration and defines the scale range for the analysis. At the end of the computation, the estimated fractal

dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

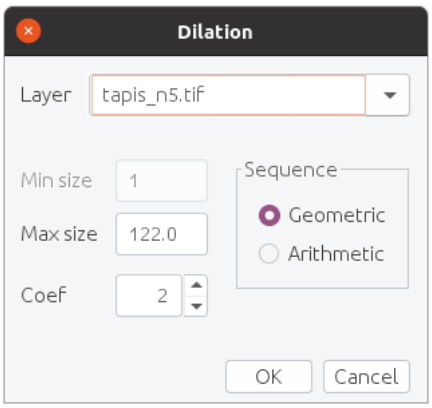



Figure 5.10: Dilation dialog box

### 5.3.3 `Correlation` menu

The method is described in 3.1.3. The user first needs to set the binary raster layer under consideration and defines the scale range for the analysis. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range at the bottom of the map is added.

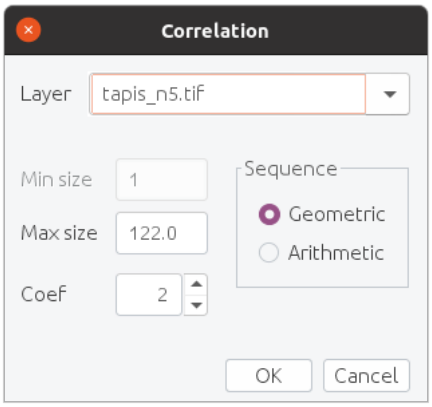



Figure 5.11: Correlation dialog box

### 5.3.4 `Radial` menu

The method is described in 3.1.4. The user first needs to set the raster layer (binary or grayscale) under consideration and defines the scale range for the analysis. She/he also has to set the value of one specific parameter: `Centre` that corresponds to the starting point of the counting process. The coordinates can be entered directly or set by clicking on the map.
At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` frame (see 6.1) and a new layer displaying the scale range on the map is added.

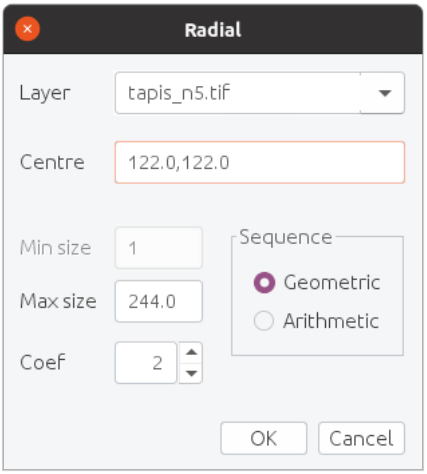



Figure 5.12: Radial dialog box

### 5.3.5 `Radial all points` menu

This analysis applies radial method centered on each black pixel of the raster and estimates for each, the fractal dimension. The result is a raster where each black pixel contains the local fractal dimension. Additional rasters contain the $r^2$ quality of estimation and the confidence interval if `Compute confidence interval` is checked. The default estimation model is $ax^d$ and can be changed. The sampling sequence is arithmetic from size 1 to `Max size`. The max size can be estimated automatically by checking `Auto threshold`. In this case a minimal max size must be given. The max size is estimated by finding the main inflexion point in the estimated curve between `min` and `Max size` parameters. If there is no inflexion point `Max size` is used. With this option, an additional raster is added containing the estimated max size.

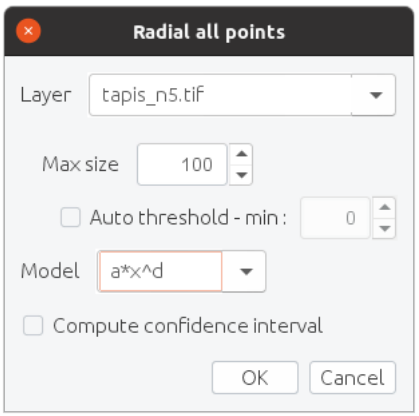



Figure 5.13: Radial all points dialog box

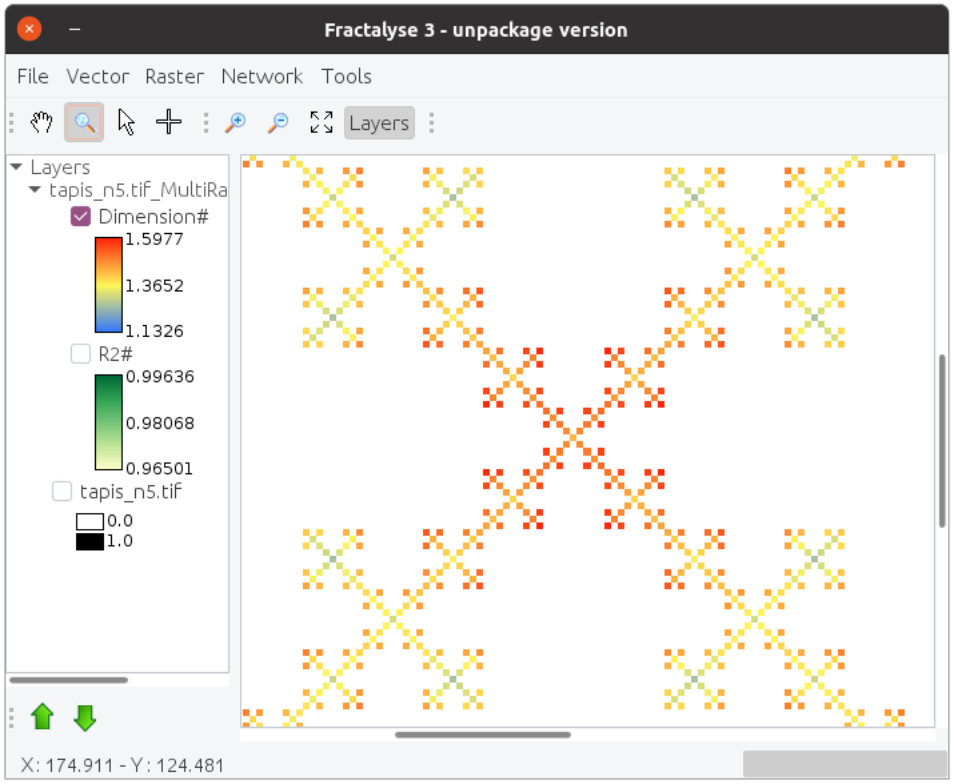



Figure 5.14: Radial all points result

### 5.3.6 `Multifractal` menu

The principles of multifractal analysis are described in 3.2. Fractalyse implements 2 multifractal analysis: boxcounting and sandbox.

#### `Box counting` menu

The method is described in 3.2.1. The settings are exactly the same as the monofractal boxcounting analysis (see 5.3.1).
At the end of the computation, the estimated fractal dimension spectrum is displayed in the `Multi-fractal estimation` frame (see 6.2) and a new layer displaying the scale range at the bottom of the map is added.

#### `Sandbox` menu

The method is described in 3.2.2. The settings are exactly the same as the correlation analysis (see 5.3.3).
At the end of the computation, the estimated fractal dimension spectrum is displayed in the `Multi-fractal estimation` frame (see 6.2) and a new layer displaying the scale range at the bottom of the map is added.

## 5.4 `Network` menu

This menu contains fractal analysis for network data. For each method, before defining the sampling as previous methods, user have to define the distance type and counting mass data to be used.

### 5.4.1 Choice of distance and mass counting

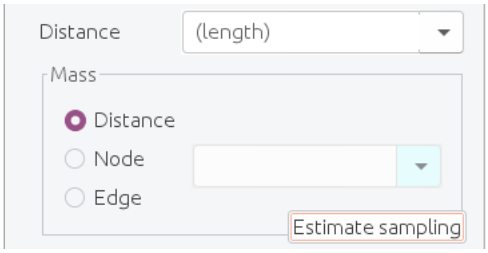



Figure 5.15: Network panel

First the distance measure must be set. It can be of 3 types:

- (topology): topological distance ie. number of edges
- (length): length in spatial units of the edges
- layer attribute: the edge distance corresponds to its attribute value

After, the counted mass must be set. This mass corresponds to:

- Distance: the total length of the network in the selected distance unit.
- Node attribute: the sum of the selected node attribute.
- Edge attribute: the sum of the selected edge attribute.

After setting these 2 parameters, the sampling can be set manually or automatically by clicking on `Estimate sampling`.

### 5.4.2 `Correlation` menu

The method is described in 3.1.3. The user first needs to set the network layer under consideration, defines the distance type and the counted mass, and defines the scale range for the analysis. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` dialog (see 6.1).

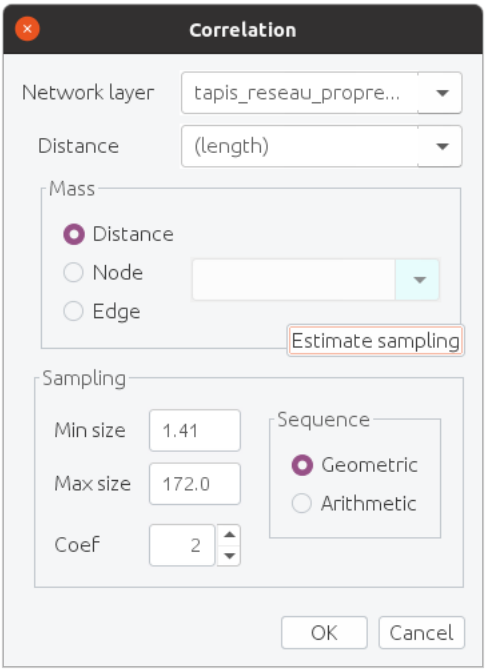



Figure 5.16: Network correlation dialog box

### 5.4.3 `Radial` menu

The method is described in 3.1.4. The user first needs to set the network layer under consideration, defines the distance type and the counted mass, and defines the scale range for the analysis. She/he also has to set the value of one specific parameter: `Starting point` that corresponds to the starting point of the counting process. The coordinates can be entered directly or set by clicking on the map. The starting point will be automatically snapped to the nearest node of the network. At the end of the computation, the estimated fractal dimension is displayed in the `Estimation` dialog (see 6.1).

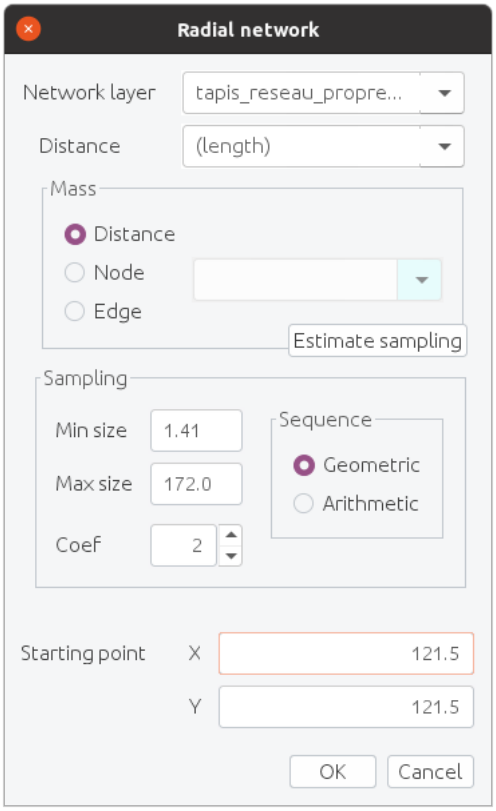



Figure 5.17: Network radial dialog box

### 5.4.4 `Service` menu

Not documented yet

### 5.4.5 `Backbone` menu

Not documented yet

### 5.4.6 `Multifractal` menu

The principles of multifractal analysis are described in 3.2. Fractalyse implements one multifractal analysis for network: sandbox.

#### `Sandbox` menu

The method is described in 3.2.2. The settings are exactly the same as the correlation analysis (see 5.4.2). At the end of the computation, the estimated fractal dimension spectrum is displayed in the `Multi-fractal estimation` dialog (see 6.2).

# Chapter 6

# Estimation module

After the counting process, the fractal dimension(s) can be estimated with this module. There is 2 main types of estimation depending of the counting method: monofractal and multifractal.

## 6.1 Monofractal

The principles of the monofractal estimation is described in 3.1. The empirical curve coming from the monofractal method is estimated by an OLS (ordinary least square) regression. The estimated D coefficient corresponds to the fractal dimension. There is 2 types of regression in Fractalyse:

- LOG: the curve is logged to use a linear model $\log y = D \log x + b$. This type is used by default with geometric sequence.
- DIRECT: the curve is estimated with a power law function of type $y = ax^D$. This type is used by default with arithmetic sequence.

It is strongly recommended to not change the `Type` and the `Model`.

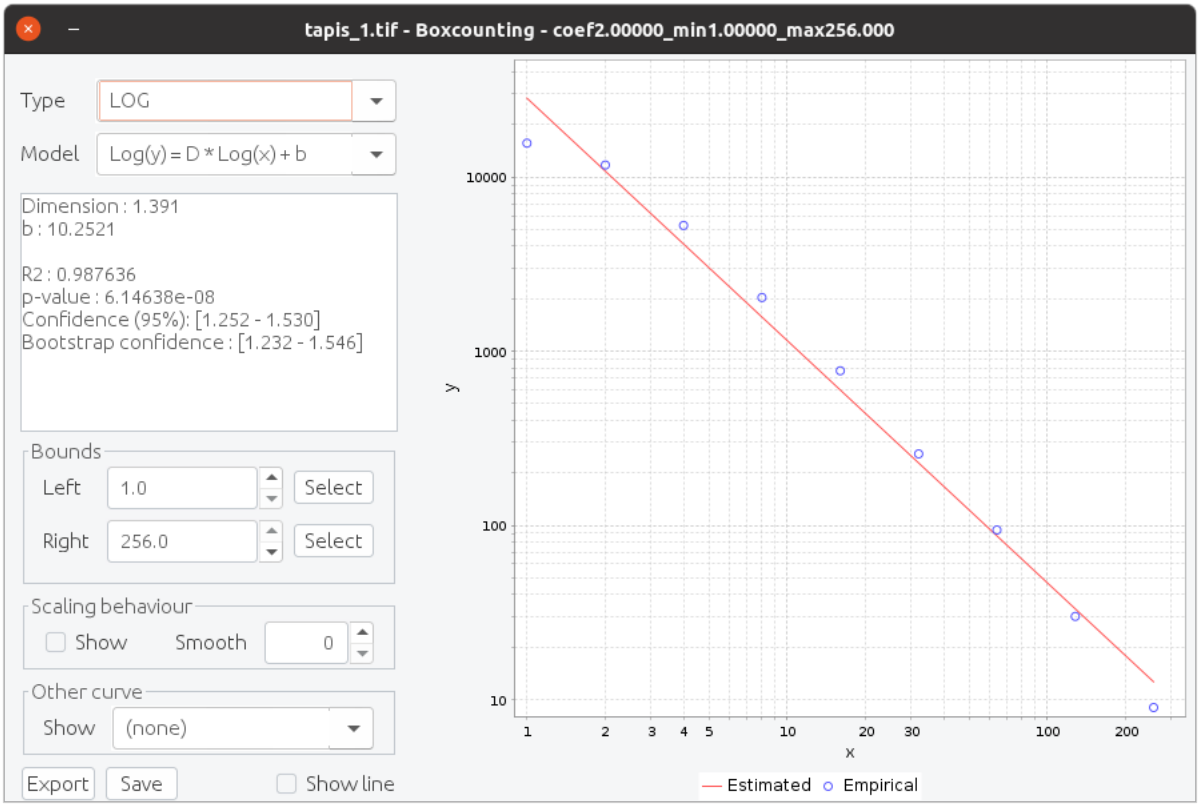



Figure 6.1: Estimation window

The graph part shows the empirical curve as little circles and the estimated curve (regression model) as red line.

### Numerical results

- Dimension: the estimated fractal dimension (D coefficient of the model)
- a,b,c: other coefficients depending of the chosen model
- R2: the determination coefficient of the regression $r^2$
- p-value: the significance of the regression, must be lower than 0.001
- Confidence (95%): the confidence interval of the fractal dimension at 95%. Only computed for linear regression.
- Bootstrap confidence: the confidence interval of the fractal dimension at 95% estimated by bootstrap. Computed for all models.

### Bounds

By default the regression is calculated on all points of the empirical curve. User can remove some points at the start (left) or end (right) of the curve. The regression is automatically updated and also the scale range on the map.

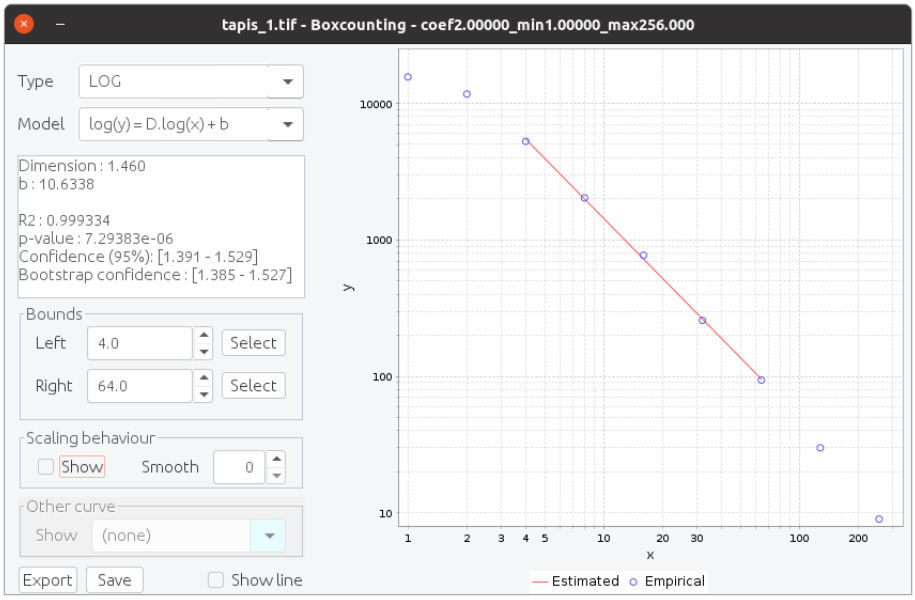



Figure 6.2: Estimation with user defined bounds

### Scaling behaviour

The scaling behaviour corresponds to the first derivative of the logged empirical curve. It shows the local estimates of the fractal dimension.

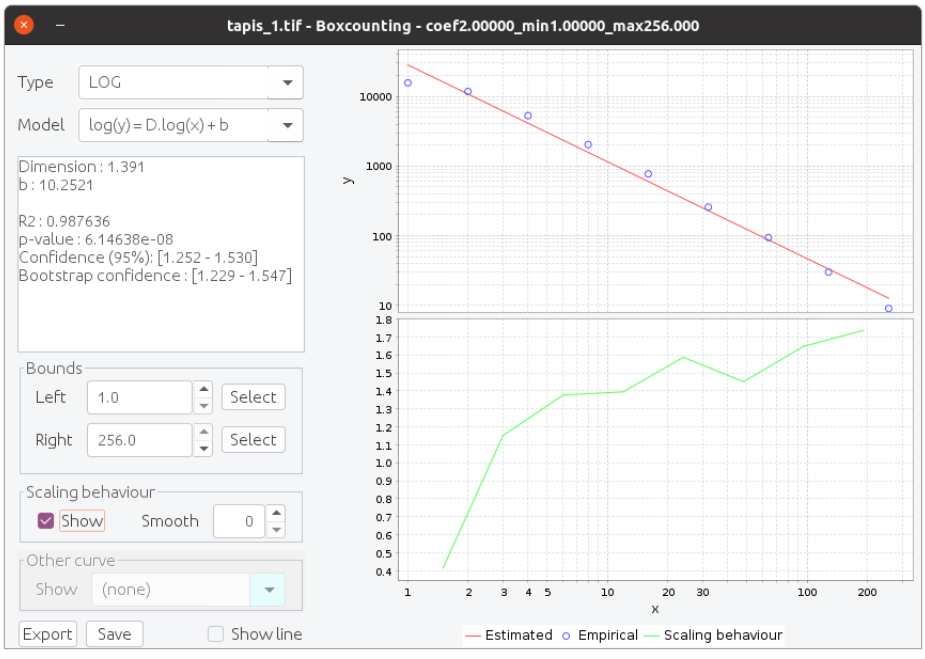



Figure 6.3: Scaling behaviour

### Other curve

For some methods, there is additional curve. For the moment only dilation method produce additional curve: the number of clusters for each dilation size.

The `Export` button allows to export all numerical results in a text file or export the graphical curves in SVG format.

The `Save` button allows to save the estimation in XML format to reload it after. See Save/reload estimation.

## 6.2 Multifractal

The principles of the multifractal estimation is described in 3.2. The logged empirical curves ($M_q$) coming from the multifractal method are estimated by a linear OLS (ordinary least square) regression. The estimated slopes correspond to the $\tau_q$ curve.

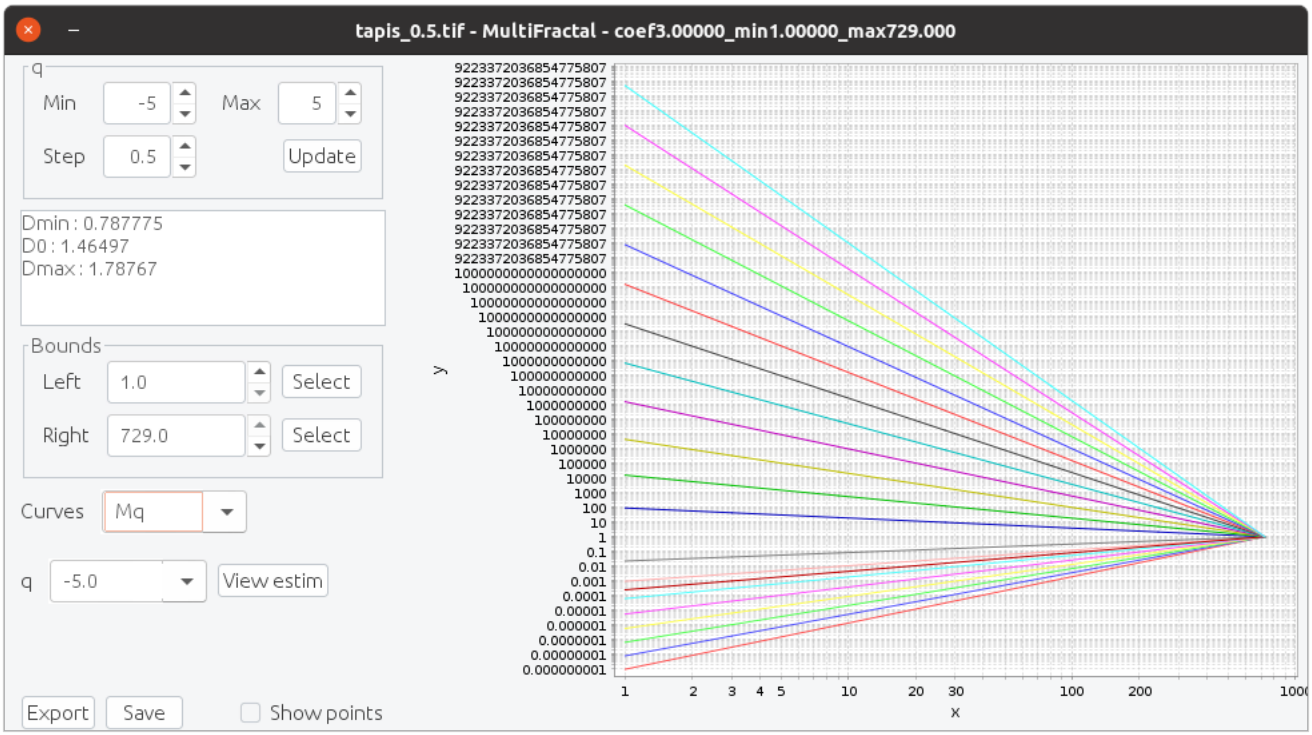



Figure 6.4: Multifractal estimation window - $M_q$

### q set

The set of $q$ can be changed with min, max and increment step. User have to click on `Update` button after, to update all curves ($M_q$, $\tau_q$, ...) and numerical results.

### Numerical results

The numerical results consist of the min/max fractal dimension spectrum $D_q$ and $D_0$ corresponding to the classical monofractal dimension.

### Bounds

As monofractal estimation, the regression on $M_q$ curves can be only computed on a subset of scale sizes.

### Curves

By default, the graph shows the $M_q$ curves. User can displayed the other curves : $\tau_q$, $D_q$ and $f(\alpha_q)$ by selecting it in the dropdown list.

### View estim

Since there is many linear regressions (one for each $M_q$), they are not showed directly in the frame. Estimation of one $M_q$ curve can be shown by selecting the wanted $q$ and click on `View estim` button. A monofractal estimation window is shown with the estimation of the selected $M_q$ curve.

The `Export` button allows to export all numerical results in a text file or export the current graphical curves in SVG format.

The `Save` button allows to save the estimation in XML format to reload it after. See Save/reload estimation. Be careful, the q-set cannot be changed after reloading it.

### Multifractal curves

The $\tau_q$ curve must be monotonic decreasing.

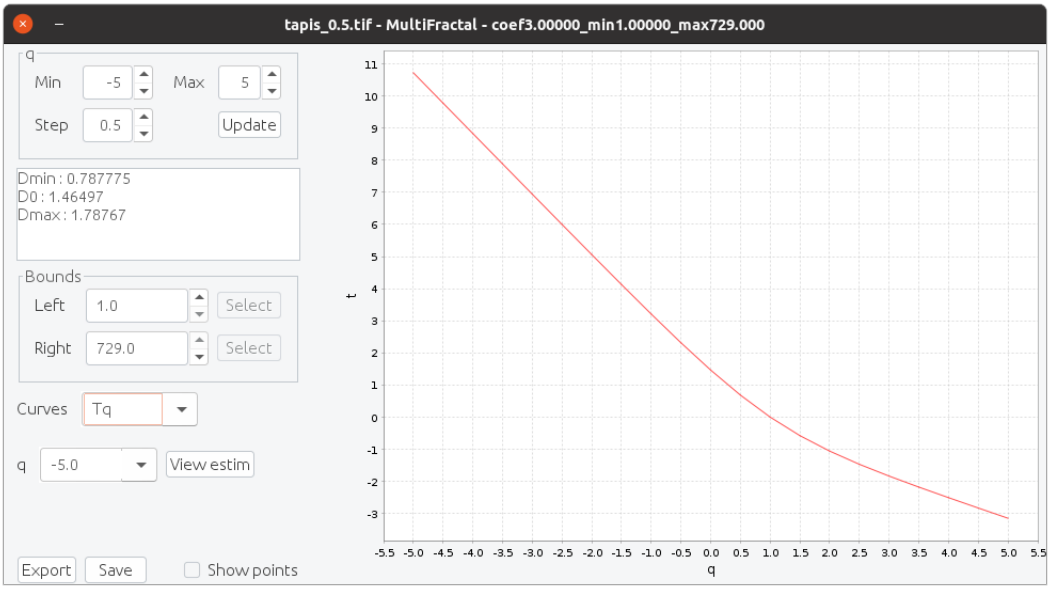



Figure 6.5: Multifractal estimation window - $\tau_q$

The dimension spectrum $D_q$ must be monotonic decreasing. If not, the results cannot be used.

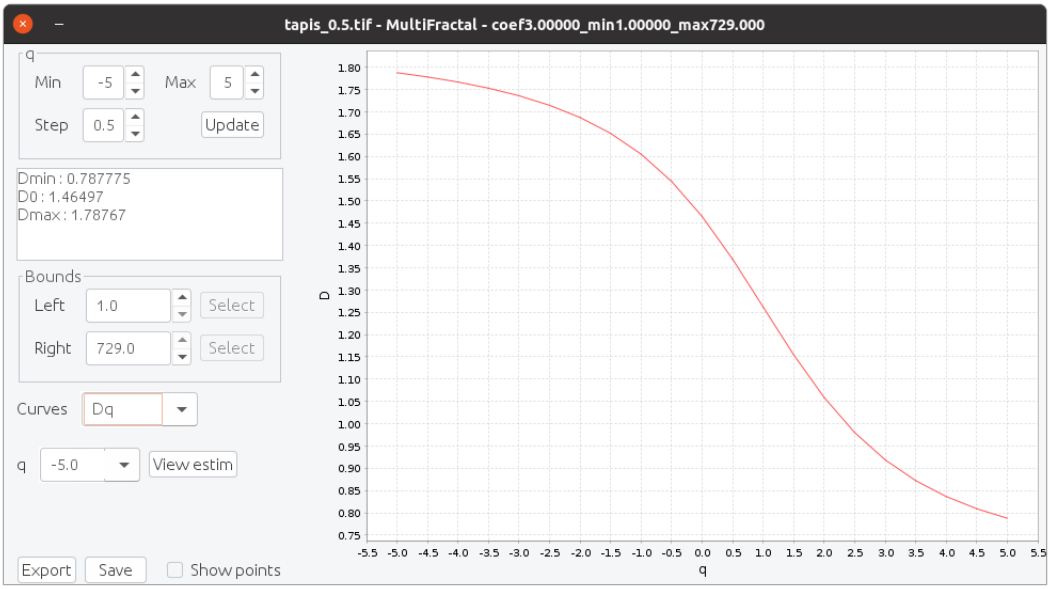



Figure 6.6: Multifractal estimation window - $D_q$

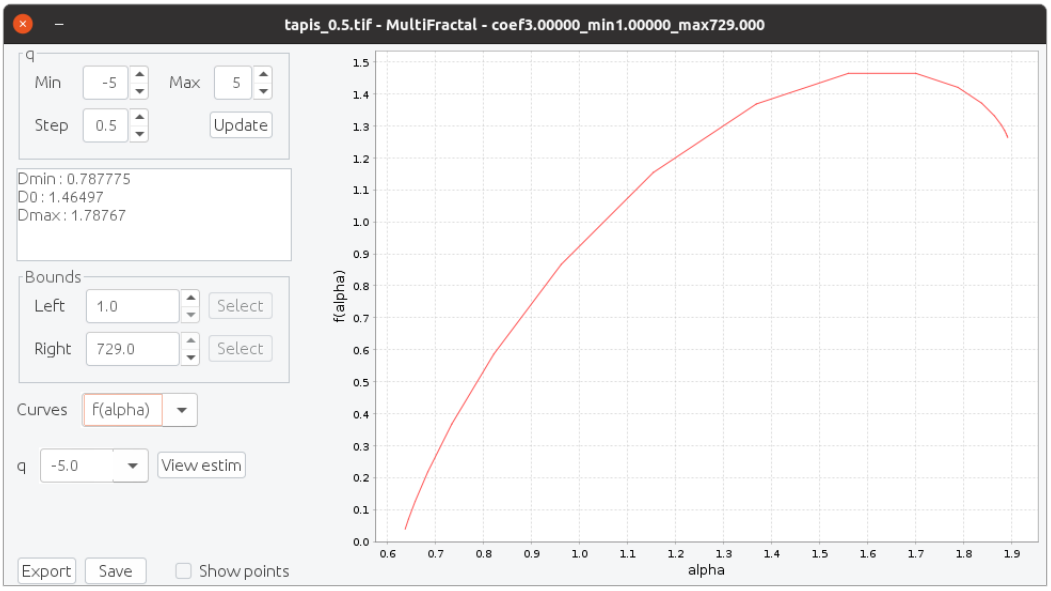



Figure 6.7: Multifractal estimation window - $f(\alpha_q)$

## 6.3 Save/reload estimation

Estimations can be saved and reloaded later. To save an estimation, use the `Save` button on the estimation frame. The estimation is stored in XML format. If you have already closed the estimation frame you can reopen it from the layer panel of the main frame. To reload an estimation, use the menu `File - Load estimation`. After selecting the XML file, the estimation window will appear with the saved estimation.

For multifractal estimation, the q-range cannot be changed after reloading it.

## 6.4 Estimation layer: scale range

For each estimation frame, a new layer showing the scale range on the map is created. This scale range is updated when user changes the bounds of the estimation.

The contextual menu of this layer allows to:

- Remove: remove this estimation and the scale range layer
- Estimation: reload the estimation frame if you have already closed it
- Save: save the estimation in a XML file

# Chapter 7

# Miscellaneous

## 7.1 `Preferences` menu

### 7.1.1 Memory

The memory used by Fractalyse plays an important role. If there is not enough RAM, computation will be slower or may fail (OutOfMemoryError or GC Overhead message). This panel can be used to adjust the memory allocated to Fractalyse. If you have a 32-bit version of Java, Fractalyse will be limited to about 2 Go (2000 Mo) of memory. If your computer has more than 2 Go of RAM memory, it is highly recommended you install the 64-bit version of Java to use the available memory beyond 2 Go. After changing the amount of memory, you have to restart Fractalyse.

### 7.1.2 Processors

Most of Fractalyse calculations are parallelized. This means that Fractalyse can use several cores/processors to speed up computations, a quad-core processor being theoretically four times faster than a single-core processor. By default Fractalyse sets the number of cores to the number of computer cores minus 1. After changing the parameter, you have to restart Fractalyse.

## 7.2 `Log window` menu

This window show some details when error occurred. These messages are useful for debugging purpose.

# Part III

# Command line interface

# Chapter 8

# Prerequisite

Fractalyse can be used in command line interface (CLI). It is useful for executing Fractalyse on a distant computer without a graphical interface, or batching some processes that are not available in the graphical user interface (GUI).

## 8.1 Launch Fractalyse in CLI mode

First you have to open a terminal window. Then, go to the directory of the Fractalyse program with *cd* command. Finally, type the following command to display the Fractalyse help screen:

```
java -jar fractalyse-3.0.jar --help
```

Result

```
Usage :
java -jar fractalyse.jar [-proc n]
...
...
```

You're ready to use Fractalyse in CLI mode !

## 8.2 Syntax

### 8.2.1 Definition

Commands always start with a double dash (`--`).
A global option starts with only one dash (`-`).
A parameter does not have a dash.

### 8.2.2 Character separator

Blank spaces are used to separate commands and parameters. You cannot have a name containing blank spaces.

### 8.2.3 Optional parameter

Parameters enclosed in brackets are optional. Therefore, parameters not in brackets are mandatory.

### 8.2.4 Command execution

Fractalyse can execute only one command.

```
java -jar fractalyse-3.0.jar ...
```

# Chapter 9

# Command reference

## 9.1 General command

### 9.1.1 `--help`: display help

Command :

```
java -jar fractalyse-3.0.jar --help
```

Result :

```
Usage :
java -jar fractalyse.jar [-mpi | -proc n] COMMAND
COMMAND:
        --rasterize [neg] res=val file_1.shp [... file_n.shp]
        --binarize min=val max=val file_1.tif [... file_n.tif]
        --boxcounting [gliding=val] SAMPLING [estim=log|direct] file_1.shp [... file_n.shp]
        --rboxcounting SAMPLING [estim=log|direct] file_1.tif [... file_n.tif]
        --dilation SAMPLING [estim=log|direct] file_1.shp [... file_n.shp]
        --rdilation SAMPLING [estim=log|direct] file_1.tif [... file_n.tif]
        --correlation SAMPLING [estim=log|direct] file_1.shp [... file_n.shp]
        --rcorrelation [border=val] SAMPLING [estim=log|direct] file_1.tif [... file_n.tif]
        --clusters [buf=radius]file_1.shp [... file_n.shp]
SAMPLING:
        [coef=val] [min=val] [max=val] [seq=arith|geom]
```

## 9.2 Data manipulation commands

### 9.2.1 `--rasterize`: convert vector to raster

`--rasterize [neg] res=`*`val file_1.shp [... file_n.shp]`*

This command converts vector files into raster tif file. Shapefile and Geopackage vector layer can be rasterized. The result is a binary raster layer. The new files have the same name. Only the file extension is changed.

#### Parameters

- `res=val`: the pixel resolution of the resulting raster
- `neg`: if set pixel values are reverse, black pixels become white and white pixels become black.

### 9.2.2 `--binarize`: convert grayscale raster to binary

`--binarize min=`*val* `max=`*val* *file_1.tif* `[...` *file_n.tif*`]`

This command converts gray scale raster to black and white raster. The new files name are suffixed with `-bin`.

#### Parameters

- `min=val`: the minimal value for a pixel to become black
- `max=val`: the maximal value for a pixel to become black.

## 9.3 Monofractal analysis commands

Each fractal method can be applied to one or more spatial data files. For each file, the counting process and the fractal dimension estimation are calculated and saved in a text file with the same name. Another text file named with the method and its parameters is also created containing the summary of all estimations.

The parameters are globally the same for all methods :

- method: the methods name prefix with r for raster data
- optional parameters if any
- sampling: the scale sampling (optional)
- estimation type: logarithmic or direct (optional)
- one or more files corresponding to vector or raster layers

### 9.3.1 `--boxcounting`: box counting on vector data

`--boxcounting [gliding=`*val*`] SAMPLING [estim=log|direct]` *file_1.shp* `[...` *file_n.shp*`]`

This command apply the vector boxcounting method to one or more vector layers. The method is described in 3.1.1. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

#### Optional parameter

- `gliding=val`: number of gliding grid tested. For more details see `Number of grid positions` in section 3.1.1.

For each vector file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.2 `--rboxcounting`: box counting on raster data

`--rboxcounting SAMPLING [estim=log|direct]` *file_1.tif* `[...` *file_n.tif*`]`

This command apply the raster boxcounting method to one or more raster layers. The method is described in 3.1.1. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

For each raster file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.3 `--dilation`: dilation on vector data

`--dilation SAMPLING [estim=log|direct] file_1.shp [... file_n.shp]`

This command apply the vector dilation method to one or more vector layers. The method is described in 3.1.2. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

For each vector file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.4 `--rdilation`: dilation on raster data

`--rdilation SAMPLING [estim=log|direct] file_1.tif [... file_n.tif]`

This command apply the raster dilation method to one or more raster layers. The method is described in 3.1.2. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

For each raster file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.5 `--correlation`: correlation on vector data

`--correlation SAMPLING [estim=log|direct] file_1.shp [... file_n.shp]`

This command apply the vector correlation method to one or more point vector layers. The method is described in 3.1.3. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

For each vector file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.6 `--rcorrelation`: correlation on raster data

`--rcorrelation [border=val] SAMPLING [estim=log|direct] file_1.tif [... file_n.tif]`

This command apply the raster correlation method to one or more raster layers. The method is described in 3.1.3. The sampling configuration is described in 9.3.7 and the estimation type in 9.3.8.

#### Optional parameter

- `border=val`: exclude pixels as counting centre too near from the raster border to avoid border effect. The value given corresponds to the distance from the raster border in pixel. To avoid completely border effect, user have to give the same distance in maxsize sampling parameter with respect to spatial unit.

For each raster file a text file with the same name is created containing the detail of each estimation. One text file is also created containing the summary of all fractal estimations.

### 9.3.7 `SAMPLING`: scale range definition

`[coef=val] [min=val] [max=val] [seq=arith|geom]`

All fractal analysis methods require to define the range of scales that will be considered for the analysis. If not set the sampling is automatically estimated from the data layer.

**Optional parameters**

- `seq=arith|geom`: arithmetic or geometric sequence. By default the sequence sampling is set to geometric.
- `coef=val`: coefficient for increasing scales from `min` to `max`. By default $coef = 2$ for geometric sequence and $coef = 1$ for arithmetic sequence.
- `min=val`: the minimal scale size
- `max=val`: the maximal scale size

The scale range is comprised between `min` and `max`. By default, the sequence of scales is geometric, which means that each scale value is multiplied by the value `coef` (starting from `min`). When the sequence chosen is arithmetic, the value `coef` is added to each scale value. For geometric sequence, `coef` must be strictly greater than 1, for arithmetic sequence `coef` must be strictly greater than 0.

The parameters `min` and `max` are by default automatically estimated from the input data. For raster data, `min` is the pixel size. For vector data, `min` is the size of the smallest element (polygon or line segment). The `max` parameter is set by default as the half of the minimum extent of the data layer. If a fractal method is applied on several data layer the sampling range (min and max size) may differ for each layer.

For raster data, the scales have to be rounded to the pixel unit. This may result in slight deviations from the defined sequence.

### 9.3.8 Estimation parameter

`[estim=log|direct]`

By default, the estimation type (log or direct) is set depending on the sequence sampling: for geometric sequence the estimation is logarithmic and for arithmetic sequence the direct type is used. User can change this behaviour by adding `estim` parameter. For more details on estimation method, see 6.1.

## 9.4 Other commands

### 9.4.1 `--clusters`: counting clusters for vector layer

`--clusters [buf=radius]` *`file_1.shp`* `[...` *`file_n.shp`*`]`

## 9.5 Options

### 9.5.1 `-proc`

Defines the number of processors (or cores) used by Fractalyse. By default, CLI mode uses the value defined in the preferences window. See the Performance tuning section for more details.

### 9.5.2 `-mpi`

This option is used to execute commands on several computers in the MPI environment. See the parallelism section of Performance tuning for more details.

# Chapter 10

# Examples of commands

## 10.1 Data handling commands

**–rasterize**

The following command rasterizes all vector geopackage files contained in the current directory. For each .gpkg file, it creates a new raster .tif file with a resolution of 5 meters. Each new file has the same name with the suffix: `_res5.tif`.

```
java -jar fractalyse-3.0.jar --rasterize res=5 *.gpkg
```

**–binarize**

The following command binarizes all tif images contained in the current directory. For each .tif file, it creates a new file in which each pixel having a value between 1 and 10 is set to 1 and to 0 otherwise. Each new file has a name with the suffix: `_bin1-10.tif`.

```
java -jar fractalyse-3.0.jar --binarize min=1 max=10 *.tif
```

## 10.2 Fractal analysis commands

The following command calculates the fractal dimension with box counting method for each raster previously binarized.

```
java -jar fractalyse-3.0.jar --rboxcounting *_bin*.tif
```

The following command calculates the fractal dimension with box counting method for each raster previously binarized and the scale sampling range is set between 10 and 1000. The sequence is left unchanged (geometric) and also the coefficient (coef = 2).

```
java -jar fractalyse-3.0.jar --rboxcounting min=10 max=1000 *_bin*.tif
```

The same principle can be applied for all monofractal methods.

# Chapter 11

# Performance tuning

## 11.1 Parallelism to speed up execution

### 11.1.1 One computer: threads

If your computer has more than one core (most of them), you can take advantage of parallelism. Most Fractalyse commands are parallelized. You can speed up command execution by defining the number of cores (or processors) used by Fractalyse with the option `-proc` after the project command:

```
java -jar fractalyse-3.0.jar -proc 8 ...
```

By default, CLI mode uses the number of processors defined in the `Preferences` window of the GUI.

### 11.1.2 Computer cluster: mpi

Fractalyse can be run on computer clusters wich support Java for OpenMPI.

```
mpirun java -jar fractalyse-3.0.jar -mpi  ...
```

Only some commands can be used in mpi environments: –boxcounting and –rcorrelation

## 11.2 Memory management

In CLI mode, the memory configuration defined in the `Preferences` window cannot be used. By default, the amount of memory available for Fractalyse is system dependent. It can vary from 128 Mb to several Gb. In most cases, Fractalyse runs normally. But if a large image is analysed, some commands would be slow or even crash due to memory limitation. If Fractalyse execution ends up with the message `OutOfMemoryError`, `GC overhead` or `Java Heap space`, you need to increase the memory allocated to Fractalyse.

To define manually the maximum amount of memory allocated to Fractalyse, use Java option `-Xmx` :

```
java -Xmx2g -jar fractalyse-3.0.jar ... # 2Gb allocated
java -Xmx1500m -jar fractalyse-3.0.jar ... # 1500 Mb -> 1.5Gb allocated
```

If you cannot allocate more than 1Gb or 1.5G although your computer has more memory available, you have probably a 32-bit version of Java, which is limited to less than 2Gb of memory. Check your Java version:

```
java -version
```

If it is a 32-bit version, install a 64-bit Java version to handle all your computer memory.