Stefanie Chaves dos Santos

Fractal Analysis of Forest Patches

Brazil

May 15th, 2023

Stefanie Chaves dos Santos

Fractal Analysis of Forest Patches

Master's thesis presented to Universidade do Estado da Bahia, Biosystems Modeling and Simulation Course, as a partial requirement for obtaining the master's degree in Biosystems Modeling and Simulation.

Universidade do Estado da Bahia - UNEB

Postgraduate Program in Modeling and Simulation of Biosystems

Advisor: Antônio Teófilo Ataíde do Nascimento

Co-advisors: Maria Dolores Ribeiro Orge and Mara Rojane Barros de Matos

Brazil

May 15th, 2023

Abstract

Since its creation, fractal geometry has been widely used in the study of many natural objects, as forestry patches. Due to the spread of the perimeter-area dimensions as a landscape metric, a way to utilize it in the planning of recuperation of fragmented landscapes was sought in order to minimize the edge effects. The method was named patch expansion and consists in the delimitation of the recovery areas. Additionally, an app based on Python language was developed for the application of the method in the analysis of forestry landscapes' images. A study of the effect of the perimeter-area dimension's proportionality constant in the dimension's fluctuation with patch's size variation was also developed.

Key-words: Perimeter-area dimension; landscape ecology; fractal geometry; forestry recovery.

List of figures

List of tables

Summary

Introduction

By have been created to describe patterns too irregular to be adequately described by euclidean geometry (FALCONER, 2014), fractal geometry become an area of big interest to researchers interested in describe natural phenomena (BARNSLEY et al., 1988), especially in the hope to solve scaling problems (TURNER; GARDNER, 2015).

A fractal's dimension is one of its more basic features (ROSENBERG, 2020), being able to be calculated in many ways, and provides diverse information about the object being studied. It's needed to highlight that there are many shapes that, despite having the same perimeter-area dimension, present completely different shapes (ALLAIN; CLOITRE, 1991).

One of the main applications of fractal geometry in landscape ecology is the use of the perimeter-area dimension as a measure of patch's complexity and anthropic action (LOPEZ; FROHN, 2017), in the study of landscape patterns (TURNER; GARDNER, 2015) and in the design of urban and rural sustainable landscapes (YU et al., 2019).

Currently, one of the main problems faced by tropical forests are fragmentation and edge effect, both caused by the decrease of the patches' area (BROADBENT et al., 2008; MURCIA, 1995). Because of this, the question of how the fractal dimension could be used in the project of ecological recuperation of fragmented landscapes in order to minimize the edge effect was brought up.

This thesis was developed with the objective of define a method, named patch expansion, in order to recuperate forest patches by obtaining landscape patches with the desired fractal dimension, exploring the hypothesis that the perimeter-area dimension could be applied in the method's development due to its easy calculus and already spread application was a landscape metric.

In order to do that it was needed to study the applications of fractal geometry in landscape ecology, analyze the meaning of the perimeter-area dimension and the role of the proportionality constant of the perimeter-area dimension's equation, and the development of a method to obtain any polygon with the desired perimeter-area dimension starting from another polygon with any dimension.

Additionally, in order to apply this method to landscape patches in GIS (Geographic Information System), a program was developed in Python language to analyze the images and delimitate the expansion points.

In Chapter 1 of this thesis a revision of literature of fractal geometry and its applications in landscape ecology with focus in the perimeter-area dimension. Chapter 2 presents a description of the methodology applied in this thesis.

The method to obtain a polygon with the desired perimeter-area dimension from any polygon through the addition of a triangle is developed in Chapter 3. The method is initially developed in a square, then generalized to any polygon, additionally a requirement to the possibility of expansion is defined.

The method of expansion presented in Chapter 3 is further developed in Chapter 4, substituting the addition of a triangle for the addition of an isosceles trapezoid. Both methods are then compared and the advantages and difficulties of the new method are discussed.

The analysis of the proportionality constant of the perimeter-area dimension's equation is then made in Chapter 5, which also brings a method to the definition of the proportionality constant of any polygon. Possibilities of application of this notion are then presented in the chapter that also shows an application in the analysis of seeds of *Zeyheria tuberculosa* (Vell.) Bureau ex Verl.

Chapter 6 presents a description of the program developed and shows the function created to expand the polygons. The chapter also brings an application of the program in the analysis of images of the forestry landscapes patches of Basin of Catu River that spreads through the cities of Alagoinhas, Aramari, Catu and Pojuca, at Bahia (VAZ; MATOS, 2014).

The articles resulting from this research and proposal to future research are presented in Chapter 7. Then, Chapter 8 brings the conclusion of the developed research.

1 State of the Art

1.1 Fractal Geometry

The term fractal was created by the french mathematician Benoit B. Mandelbrot that defined it by any set with Hausdroff-Besicovitch fractal dimension (denoted by D_H) bigger than its topological dimension, asserting that any set D_H assumed non-integer values is a fractal (MANDELBROT, 1982).

However, Falconer points out that this definition "proved to be unsatisfactory in that it excluded a number of sets that clearly ought to be regarded as fractals" (FALCONER, 2014, p. xxvii). With the objective of solving this situation, the author proposed, not an exact definition, but a group of common characteristics to most fractals, although not all fractals present all of those characteristics.

So, we can think about fractals as a set that usually presents fine structure, that is, detailing in small scales; is too irregular to the described by traditional geometrical or analytical language; is simply defined, generally recursive; presents some form of self-similarity and possessed its fractal dimension bigger than its topological.

1.1.1 Self-similarity and self-affinity

Mandelbrot (1982) pointed out that most fractals described in his work were invariant through some scale transformation and named the fractals invariant through ordinary geometric similarities as self-similarities. The self-similar and self-affine are both composed by infinity copies of themselves, that is, they are formed by the same pattern that repeats in different length scales, however while in self-similar objects this characteristic is seen in any direction, in self-affine objects this characteristic doesn't happen in all directions (ROSENBERG, 2020; SEURONT, 2010).

1.1.2 Fractal Dimension

One of the central points of fractal dimension is the concept of dimension since when we analyze a set's dimension we can obtain information about its complexity and irregularity, its geometrical properties and the volume it occupies (FALCONER, 2014).

Fraser highlights the importance of interaction between the dimension of a set and measures in dimension theory. The author considers that

> "A natural approach to dimension theory is to quantify how large a set is at a given scale by considering optimal covers by balls whose diameter

> is related to the scale. [...] Understanding how to find covers of a set at small scales underpins much of dimension theory" (FRASER, 2020, p. 5)

Currently there are several ways to calculate the dimension. Each one of these methods leads to different values and describes different aspects. With the "hope that complex systems could be explained using a relatively low number of parameters" (SEURONT, 2010, p. 1) researchers seek to understand the relationships between the different dimensions and discover how different environments and families of examples affect their behavior (FALCONER, 2014).

1.1.2.1 Hausdorff Dimension

Hausdorff dimension is the oldest and most important of the fractal dimension and, although it can be defined to any set, its computational estimation is often hard (FALCONER, 2014). It's determined through the s-dimension Hausdorff measure, $\mathcal{H}^s(F)$. To almost any $s \geq 0$, $\mathcal{H}^s(F)$ is equal to ∞ ou 0. However, there is a critical value of s in which $\mathcal{H}^s(F)$ goes from ∞ to 0. This value of s is called Hausdorff dimension of F, $D_{\mathcal{H}}(F)$.

1.1.2.2 Box-counting Dimension

Being the most used fractal dimension, Falconer comments that the box-counting dimension "has a simple intuitive formulation and is one of the most widely used dimensions. [...] and its popularity is largely due to its relative ease of mathematical calculation and empirical estimation." (FALCONER, 2014, p. 27). Given a set $F \subset \mathbb{R}^2$ the author defines the box-counting dimension, $D_{\mathcal{B}}$, of F by the ratio of the logarithm of the number of $\delta-$cover of the set by the size of the covering, for very small coverings.

1.1.3 Lacunarity

Although dimensions provide valuable information about the geometry of objects, the fact that objects of completely different shapes share the same fractal dimension makes it necessary to have other methods of analyzing fractal objects (BARNSLEY et al., 1988).

Like several concepts related to fractal geometry, lacunarity doesn't have a formal definition (HALLEY et al., 2004), however it can be thought of as a "way of characterizing the spatial configuration of points or other components of a spatial pattern, such as patches or pixels" (NEWMAN et al., 2019, p. 10) or as a measure of the uniformity of a set (ROSENBERG, 2020).

Barnsley et al. proclaim that the concept of lacunarity was created by Mandelbrot as "an initial step toward quantifying texture" (BARNSLEY et al., 1988, p. 67-68). MANDELBROT states that, although the notion of texture is elusive, fractal geometry can be thought of as a tool for its implicit study, distinguishing lacunarity as one of its aspects.

Following the example of the fractal dimension, the methods for calculating lacunarity vary as much as its definition. However, the gliding-box algorithm by Allain e Cloitre (1991) has become the most used (HALLEY et al., 2004).

1.2 Fractal Geometry Applied to Landscape Ecology

Sugihara e May claim that "fractal models describe the geometry of a wide variety of natural objects" (SUGIHARA; MAY, 1990, p. 79). The authors comment that the study of the shape and measure of real world objects used to be made through simplifications to elements of Euclidean geometry, but it was later recognized that the irregularity present in such objects placed them under the study area of fractal geometry.

Although patterns in nature aren't ideal fractals, they are even less like objects in Euclidean geometry (HALLEY et al., 2004) and they are similar enough that it's possible to use approximations to treat them as fractals (FALCONER, 2014).

Milne (1988) declares that landscapes are better described as fractals given the non-euclidean nature of their density and perimeter-area ratio. Metzger defines landscape as a "heterogeneous mosaic formed by interactive unities, this heterogeneity existing for at least one factor, according to an observer and in a given observation scale" (METZGER, 2001, p. 4, my translation). Thus, "landscape ecology emphasizes the interaction between spatial pattern and ecological process" (TURNER; GARDNER, 2015, p. 2).

1.2.1 Scales

Wu highlights that since spatial heterogeneity is present at all scales, it's necessary not only to be able to quantify it, but also to measure its variation at different scales, making it feasible to "understand how landscapes affect, and are affected by, biophysical and socioeconomic activities" (WU, 2004, p. 125). Turner highlights the influence of the scale on the results of the studies because "processes and parameters important at one scale may not be as important or predictive at another scale" (TURNER, 1989, p. 174).

The studies regarding scale effects performed by ecology, remote sensing and geography using landscape metrics are employed to "shed new light on the problems of scale effects in pattern analysis as well as the multiscaled nature of spatial heterogeneity" (WU, 2004, p. 126).

For Montello (2015), scale, whose concept can be confusing, is one of the fundamental problems of any science that studies phenomena distributed over the earth's surface. The author differentiates the concepts of analysis and phenomenon scale as the "size of the units in which phenomena are measured and the size of the units into which measurements are aggregated for data analysis and mapping." (MONTELLO, 2015, p. 1) and "the size at which geographic structures exist and over which geographic processes operate in the world" (MONTELLO, 2015, p. 2) respectively.

Gustafson (2019) points out that, despite several attempts at resolution, the problems of measurement at different scales remain one of the biggest problems faced by landscape ecologists. Halley et al. point out that even with the various problems of establishing an ideal scale, it is not possible to disregard the effect of scales since "most processes of interest to ecologists [...] cannot even be defined without reference to a specific scale or range of scales" (HALLEY et al., 2004, p. 264).

Lopez e Frohn (2017) define the fractal dimension (together with dominance and contagion) as one of the three fundamental metrics for the study of landscape patterns. It is possible to think of a fractal as a set whose appearance remains the same at different scales (ROSENBERG, 2020). Although natural patterns maintain self-similarity only on a finite number of scales, for Mandelbrot (1982) the fact that fractals serve as reasonable approximations of natural patterns is already a reason for admiration.

Halley et al. (2004) considers fractals as the first step to deal with scale problems, but warns that before a fractal characteristic can be announced, it is necessary to define a power law relationship between scale and occupancy that must be maintained across a significant range of scales.

1.2.2 Power Laws

The idea of measurement at different scales is inherent to most definitions of fractal geometry, in which several power laws relate measurement and scale and the exponents of these laws define different fractal dimensions which determine the connection between the space occupied by an object and its size (ROSENBERG, 2020; SEURONT, 2010).

Halley et al. (2004) states that power laws were already being applied in ecology even before fractal geometry and that their intimate relationship with fractals increased the interest in studies related to fractal geometry. The power law exponents are generally different from unity in biological cases resulting in graphs in the form of curves when applied to linear axes, but it is possible to make these graphs linear by using logarithmic axes (SEURONT, 2010).

1.3 Fractal Metrics of Landscape

Fractal metrics are used in the study of landscapes to quantify the complexity of the patch's shape, estimate the degree of anthropic action (LOPEZ; FROHN, 2017), relate the patch's patterns to "potential processes associated with the complexity, and determine the nature of landscape fragmentation" (YU et al., 2019, p. 186).

There are many methods for the calculus of fractal dimension, many are applicable to landscape ecology. Milne (1988) discusses the complexity of landscapes and the need to develop methods for quantifying the relationships between different landscape features and identifying appropriate length scales for studies of landscape patterns. In his work, he also presented different methods for calculating the fractal dimension in landscape analysis.

1.3.1 Perimeter-Area Dimension

The most widely used of these methods is the perimeter-area dimension, originally applied by Lovejoy (1982) in the calculation of the fractal dimension of clouds, but it was introduced in landscape ecology by Krummel et al. (1987).

Notice that, although widely applied in landscape ecology, perimeter-area dimensions are not true dimensions in the mathematical sense (LOPEZ; FROHN, 2017), since they are not related to the concept of measurement highlighted by Fraser (2020).

The perimeter-area method has received several variations over time, but they are mainly used in GIS (graphic information system) applications (HALLEY et al., 2004; YU et al., 2019). In landscape ecology they are used as a measure of the complexity of forest patches edges in relation to the area they occupy (SEURONT, 2010).

Let P, A and D_F be the perimeter, area and fractal dimension of the forest patch, respectively. So the three are related in a way that (KRUMMEL et al., 1987):

$$P \propto \sqrt{A^{D_F}} \tag{1.1}$$

where $\propto$ indicates proportionality. Thus, there is a constant of proportionality k that makes true:

$$P = kA^{\frac{D_F}{2}} \tag{1.2}$$

Applying logarithms to both sides of the equation and expanding we get:

$$D_F = 2\frac{\ln\left(\frac{P}{k}\right)}{\ln A} \tag{1.3}$$

The data of the areas and perimeters patches measurement are plotted in a $\log - \log$ scatter plot and linear regression is then applied. Since the Equation (1.2) has two unknowns, the fractal dimension, which is a dimensionless measure, is then calculated through the slope of the regression line.

To test the theory that small and large patches have different dimensions, Krummel et al. (1987) apply linear regression separately to small and large patch data. The result found was that the smaller patches had a smaller fractal dimension compared to the larger patches.

It is agreed that D_F varies between 1 and 2 according to the complexity of the patches (KENKEL; WALKER, 1996; TURNER; RUSCHER, 1988; VRANKEN et al., 2015). While simple shapes like circles have $D_F = 1$, in more complex shapes, whose perimeter tends to fill the plane, $D_F \rightarrow 2$ (KRUMMEL et al., 1987; TURNER, 1990). Several studies managed to relate the fractal dimension of patches with anthropic action: forest areas have higher fractal dimensions due to their complex shape, while agricultural areas have lower dimensions due to their simpler format (LOPEZ; FROHN, 2017).

Although the perimeter-area dimension is more popular among landscape ecologists (GUSTAFSON, 2019), several problems have been presented in relation to its application, especially in relation to the use of linear regression (HALLEY et al., 2004; LOPEZ; FROHN, 2017; OLSEN; RAMSEY; WINN, 1993) whose estimation is impaired as the number of patches that make up the landscape decreases.

1.3.1.1 Constant k to square grids

Aiming to improve the perimeter-area dimension, making its application possible in landscapes composed even of a single patch, Olsen, Ramsey e Winn (1993) developed in their work a method that eliminated the need to apply linear regression by defining the constant of proportionality, k.

From the Equation (1.2) and taking into account the square format of the grid, the authors used the perimeter-area ratio of the square, in which

$$P = 4A^{\frac{1}{2}} \tag{1.4}$$

to define the constant as $k = 4$. Thus the dimension of a single patch was defined as:

$$D_F = 2\frac{\ln\left(\frac{P}{4}\right)}{\ln\left(A\right)} \tag{1.5}$$

Tripathi et al. states that “when $k = 4$, D_F can be thought of as the amount by which the perimeter of a given object deviates from that of a perfect square of equal area” (TRIPATHI et al., 2015, p. 1421).

The same idea can be expanded since Ricotta et al. (1997) and Chen (2020) expanded the study by calculating the constant for geometric figures other than the square. The authors differ since, while Ricotta et al. (1997) brings this differentiation due to the variation in the format of the GIS grid, Chen (2020) argues that the constant should be varied according to the format and number of growth directions of the studied system, confirming the idea of the fractal dimension as a metric of human action (O’NEILL et al., 1988).

1.3.1.2 Constant k to different shapes of grid

Ricotta et al. (1997) and Chen (2020) generalize the method of Olsen, Ramsey e Winn (1993), by calculating the constant of proportionality based on other regular polygons, the rectangle, the circle and even Koch's snowflake.

Since the area and perimeter of a regular polygon with n sides of length l are respectively calculated through the equations

$$A = \frac{nl^2}{4\tan\left(\frac{\pi}{n}\right)} \tag{1.6}$$

$$P = nl \tag{1.7}$$

equating the Equations (1.6) and (1.7) we then obtain

$$\frac{P}{n} = \sqrt{\frac{4\tan\left(\frac{\pi}{n}\right)A}{n}}$$

$$P = 2\sqrt{n\tan\left(\frac{\pi}{n}\right)}A^{\frac{1}{2}} \tag{1.8}$$

which fits the format of the Equation (1.2). Consequently, the perimeter-area proportionality constant, k, of a regular polygon with n sides is given by

$$k = 2\sqrt{n\tan\left(\frac{\pi}{n}\right)} \tag{1.9}$$

Tabela 1 – Approximate values of the constant of proportionality, k, for some regular polygons calculated through Equation (1.9).

Nº of sides	Constant
3	4,55901
4	4
5	3,81194
6	3,72242
7	3,67207

Therefore, we can rewrite Equation (1.3) as

$$D_F = 2\frac{\ln\left(\frac{P}{2\sqrt{n\tan\left(\frac{\pi}{n}\right)}}\right)}{\ln A} \tag{1.10}$$

Ricotta et al. (1997) defines the constant as $k = 2\sqrt{r}\left(1 + \frac{1}{r}\right)$ for a rectangle whose ratio between the long side and the short side is r, while Chen (2020) defines the circle constant as $k = 2\sqrt{\pi}$ and Koch's snowflake constant as $k = \frac{12}{\sqrt{6\sin\left(\frac{\pi}{3}\right)}}$.

2 Methodology

2.1 Methods

This thesis proposes to develop a planning method for the recovery of forest fragments, named patches expansion, in order to minimize edge effects. Since forestry patches recovery consists of the delimitation of an area for planting native vegetation or protection of this vegetation so that it expands naturally, the expansion of fragments is then a method to delineate the sites for the recovery area.

Patch expansion is then theoretically presented in Chapters 3 and 4, with the development of the expansion equations, while Chapter 6 brings an application of the method in images of real landscapes.

Based on perimeter-area dimension, the expansion method utilizes the area, perimeter and side length data of forestry patches to define potential sites of recovery in the shape of triangle (Chapter 3) or isosceles trapezoid (Chapter 4). We also search to define a method to verify if a patch admits expansion and which sides can be expanded.

In order for this study to be carried out in GIS (Geographic Information System), a program was developed (Chapter 6) to the patch analysis and definition, when possible, of the potential recuperation sites. The chosen analysis scale was in the point of view of a drainage basin.

Additionally, Chapter 5 develops a method of the analysis of the shape of nature objects based on the study of the proportionality constant of the perimeter-area dimension, in addition to a proposal to the expansion with the modified constant and an application of the method in the seeds analysis.

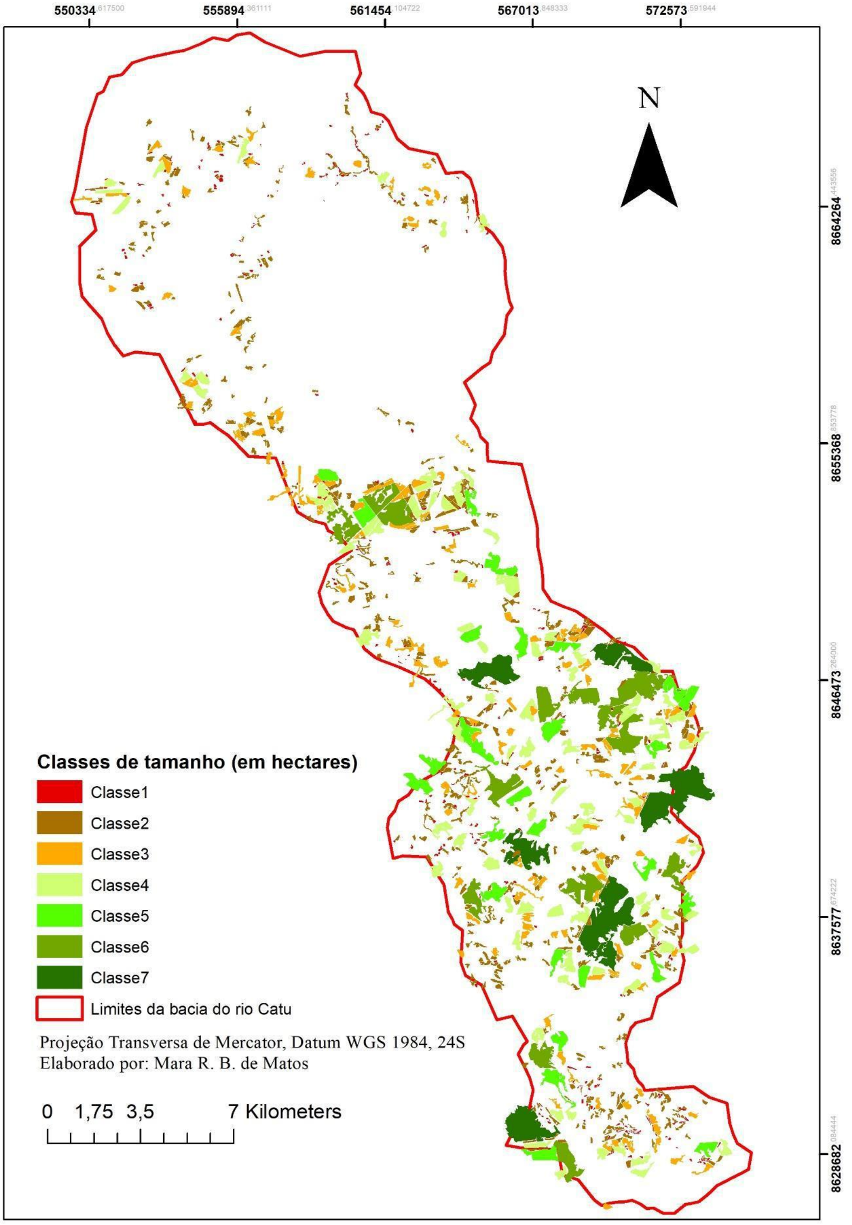
2.2 Materials

The FracFor program, just like the images of polygons and graphics featured, were developed in Python language, versions 3.9 and 3.10, with the support of Visual Studio Code.

The landscape patches used to the analysis in the program FracFor were chosen from a image of forestry patches, delineated through vectorization from a Landsat image, of the Catu River Basin in Bahia (Figure 1) provided by Prof. Dr. Mara Rojane Barros de Matos.

The analysis of seeds utilized photos of 33 samples of seeds of *Zeyheria tuberculosa*

Figura 1 – Map of forest patches in the Catu River Basin.


550334
555894
561454
567013
572573
8664264
8655368
8646473
8637577
8628682
N
Classes de tamanho (em hectares)
Classe1
Classe2
Classe3
Classe4
Classe5
Classe6
Classe7
Limites da bacia do rio Catu
Projeção Transversa de Mercator, Datum WGS 1984, 24S
Elaborado por: Mara R. B. de Matos
0 1,75 3,5 7 Kilometers

Fonte: Mara Rojane Barros de Matos

(Vell.) Bureau ex Verl. The samples were provided by the herbarium of the Campus II of Universidade do Estado da Bahia (UNEB), with the aid of Prof. Dr. Gracineide Selma Santos de Almeida. The photos were taken from a smartphone camera of 48 Mpx, with the help of a phone stand to guarantee the evenness of the distances of the images.

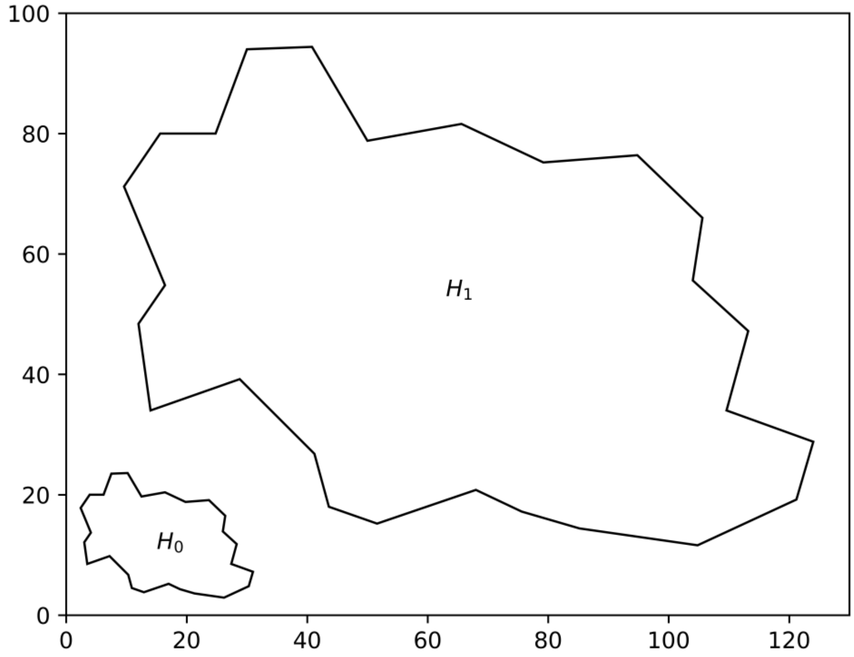
3 Triangle Expansion

Since, when observed as remote sensing images, the landscape patches are actually polygons with a very large number of sides, the expansion method was initially developed in polygons. The purpose of polygon expansion is to plan the recovery of forest patches, so that they can have the desired perimeter-area dimension.

3.1 Fractal dimension as a function of size

Krummel et al. (1987) theorized that the fractal dimension, according to Equation (1.3) varies according to the patch's size. Since the fractal dimension of a single patch can be calculated through Equation (1.5), it's possible to verify this hypothesis. In order to do that we create a irregular polygon, H_0, e apply homothety, a geometrical transformation that preserves shape and angle, but change distance, to obtain of H_0 another polygon H_1, similar to H_0 (Figure 2).

Figura 2 – An homothety of ratio 4 was applied to the polygon H_0 to obtain the polygon H_1.



Analyzing Table 2 we can observe that the fractal dimension of H_1 is less than of H_0. However, given the irregularity of H_0, it's interesting to analyze how a regular polygon's fractal dimension would behave with variation in the side length. Notice that dimension is dimensionless, while area and perimeter are in a general measure of unit.

Tabela 2 – Data of area (A), perimeter (P) and perimeter-area dimension (D_F), calculated through Equation (1.5), of polygons H_0 e H_1 in Figure 2.

	A (u.m.)	P (u.m.)	D_F
Polygon H_0	87,92204	363,715	1,04816
Polygon H_1	351,68816	5819,44	1,03275

Since the perimeter and area of a regular polygon are given by Equations (1.6) and (1.7), then the fractal dimension as a function of size is

$$D(l) = 2\frac{\ln\left(\dfrac{nl}{4}\right)}{\ln\left(\dfrac{nl^2}{4\tan\left(\frac{\pi}{n}\right)}\right)} \tag{3.1}$$

where P, A, l and n are respectively the polygon's perimeter, area, side length and number of sides and $D(l)$ is the fractal dimension.

Note that in graphs of Figures 3a and 3b, when $l \to \sqrt{\dfrac{4\tan\left(\frac{\pi}{n}\right)}{n}}$, that is, when the polygon's area tends to 1, the one-sided limits of $D(l)$ are ∞ e $-\infty$, and, when $l \to \infty$, $D(l) \to 1$. While the graph of Figure 3c, that is, the square, $D(l)$ is constant equal to 1. That supports the idea of Tripathi et al. (2015) that the dimension, as Equation (1.5), can be thought of as a measure of the variation of the perimeter-area ratios of a patch when compared to a square.

However, figures quite different from a regular polygon might present a fractal dimension, calculated through Equation (1.5), equal to 1, as seen in Figure 4. Thus, there are irregular polygons with the same perimeter-area ratio as a square.

With the objective of obtain polygons with fractal dimension equal to 1, as given by Equation (1.5), we developed a method to, starting from any polygon, F, obtain another polygon, F', with fractal dimension equal to 1.

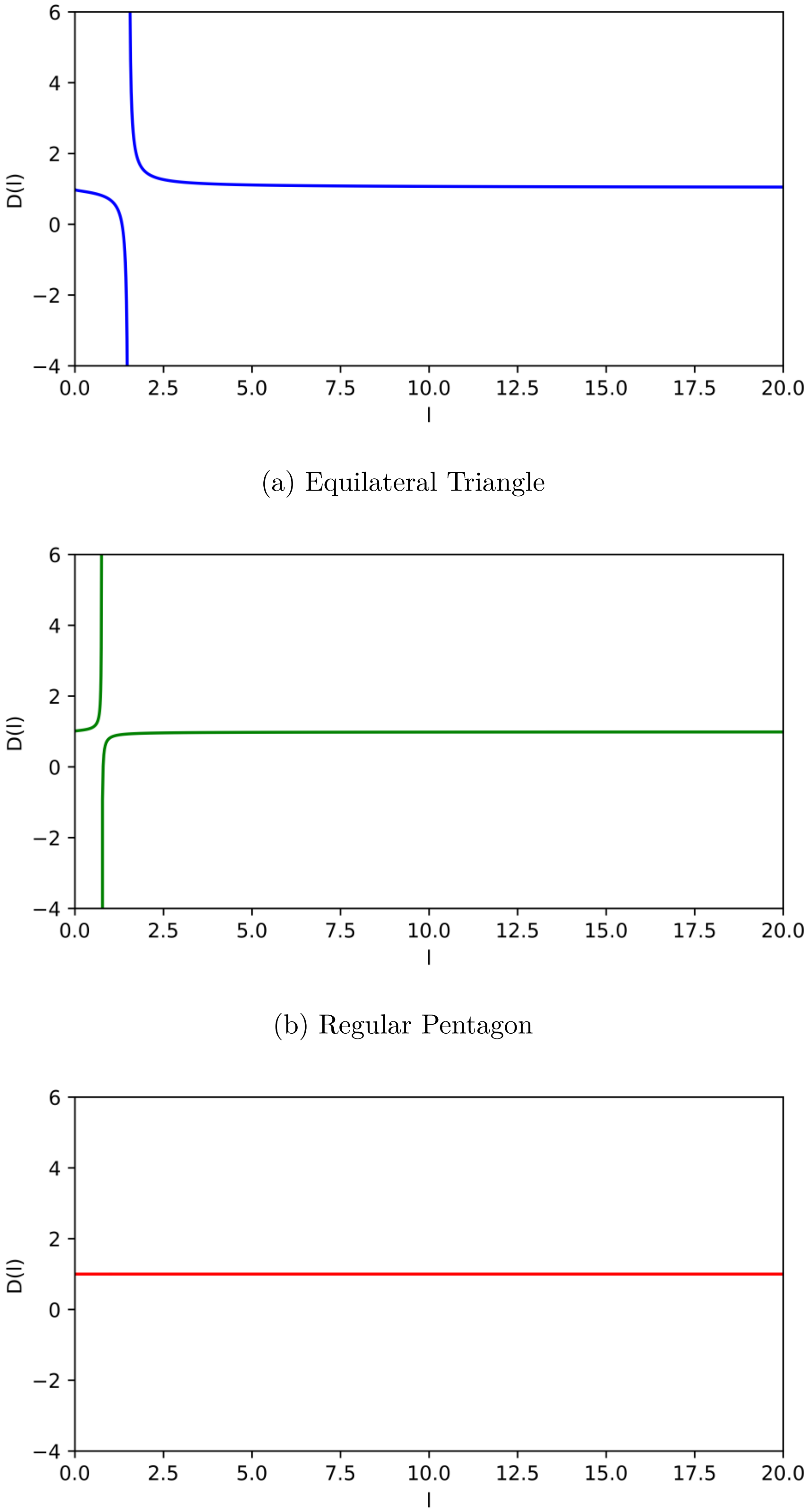
3.2 Expansion starting from a square

Initially we are going to define the creation of a pentagon with a fractal dimension equal to 1 starting from a square. That is, given a square of consecutive vertices R, S, T and U, from the elimination of the side RS, we determine the locus of points V such that the pentagon of vertices R, V , S, T and U satisfies the equality

$$P_{RVSTU} = 4\sqrt{A_{RVSTU}}$$

where P_{RVSTU} and A_{RVSTU} are respectively the pentagon $RVSTU$'s perimeter and area.

Figura 3 – Fractal dimension graph, $D(l)$, of different regular polygons as a function of their side length, l.



(a) Equilateral Triangle

(b) Regular Pentagon

(c) Square

Figura 4 – Irregular decagon with fractal dimension equal to 1.

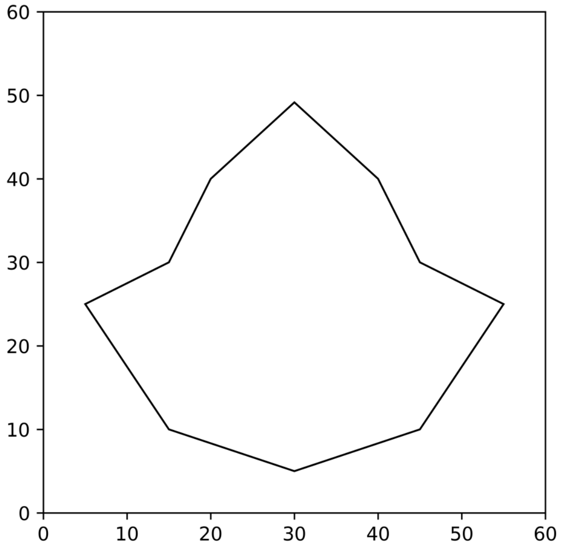
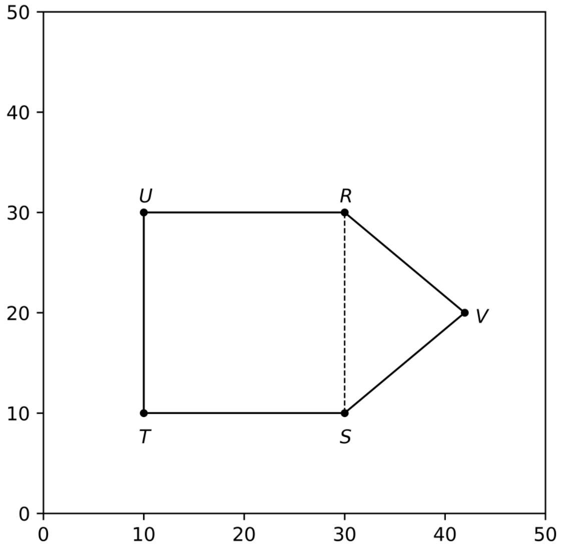


Figura 5 – Irregular pentagon $RVSTU$ with fractal dimension equal to 1 created by expanding the square $RSTU$.



Let l, c_1 and c_2 be respectively the length of sides RS, RV and VS, then $P_{RVSTU} = 2l + c_1 + c_2$. If $R = (x_R, y_R)$, $S = (x_S, y_S)$ and $V = (x, y)$, then:

$$P_{RVSTU} = 3l + \sqrt{(x - x_R)^2 + (y - y_R)^2} + \sqrt{(x - x_S)^2 + (y - y_S)^2} \tag{3.2}$$

On the other hand, A_{RVSTU} is the sum of the areas of square $RSTU$ and the triangle of vertices R, V and S. That is, $A_{RVSTU} = A_{RSTU} + A_{RVS}$. By analytic geometry we know that

$$A_{RVS} = \frac{1}{2}\begin{vmatrix} x_R & y_R & 1 \\ x_S & y_S & 1 \\ x & y & 1 \end{vmatrix} = \frac{(x_R y_S + x_S y + y_R x) - (x_R y + y_R x_S + y_S x)}{2}$$

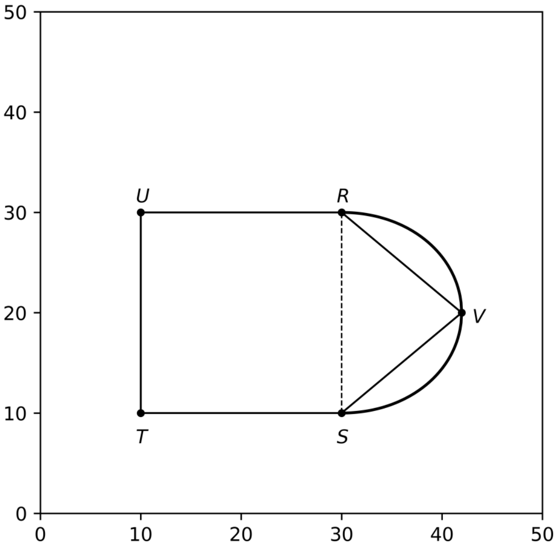
Therefore,

$$A_{RVSTU} = l^2 + \frac{x_R y_S + x_S y + y_R x - x_R y - y_R x_S - y_S x}{2} \tag{3.3}$$

Considering Equations (3.2) and (3.3), the points $V = (x, y)$, that define the pentagon $RVSTU$, whose fractal dimension is equal to 1, satisfy

$$3l + \sqrt{(x - x_R)^2 + (y - y_R)^2} + \sqrt{(x - x_S)^2 + (y - y_S)^2} = \\ 4\sqrt{l^2 + \frac{x_R y_S + x_S y + y_R x - x_R y - y_R x_S - y_S x}{2}} \tag{3.4}$$

Figura 6 – Locus of points V satisfying the Equation (3.4) for the side RS of the square $RSTU$.



It is important to highlight that the expansion does not depend on the position of the square in relation to the coordinate axes (Figure 7), since the expansion equation depends only on the polygon's area and perimeter and the length of the expansion side. However, to simplify the calculations we will usually assume that the expansion side is on the *oy* axis. Since it is always possible to move any polygon to that position through rotation and translation, both isometric transformations and therefore do not influence the fractal dimension, this simplification does not affect the generality of the method.

Besides that, the expansion curve is not a semicircle of radius $l/2$ centered in the medium point of the expansion side since, given the points $R = (0, 0)$, $S = (l, 0)$, $T = (l, -l)$ and $U = (0, -l)$, the point $V = (l/2, l/2)$ does not belong to the expansion curve of side RS of square $RSTU$, since the pentagon $RVSTU$'s perimeter and area are respectively equal to $(3 + \sqrt{2})l$ and $\frac{5l^2}{4}$, thus the dimension of $RVSTU$ is

$$D_{RVSTU} = 2\frac{\ln\left(\frac{(3+\sqrt{2})l}{4}\right)}{\ln\left(\frac{5l^2}{4}\right)}$$

Figura 7 – Locus that satisfies the Equation (3.4) in a square whose sides are not parallel to the coordinate axes.

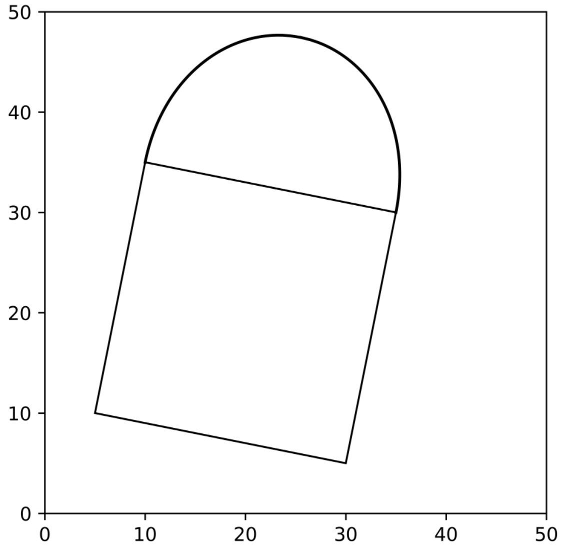
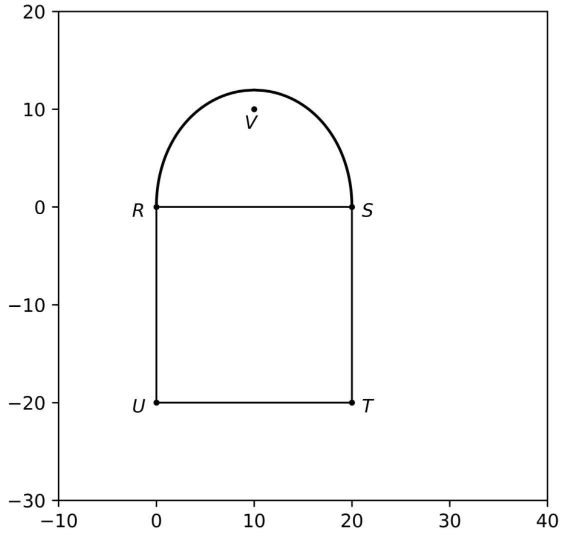


Figura 8 – Given the square $RSTU$, the point $V = (10, 10)$ does not belong to the expansion curve.



by using the logarithmic properties we can rewrite the equation as

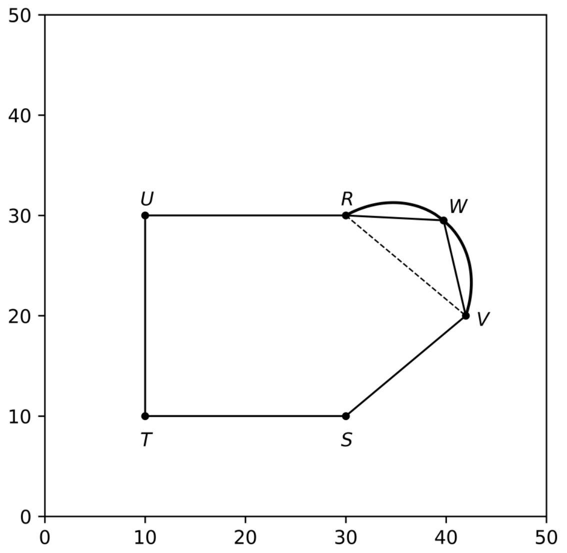
$$D_{RVSTU} = \frac{2\ln((3+\sqrt{2})l) - \ln(4)}{2\ln(\sqrt{5}l) - \ln(4)}$$

since $3+\sqrt{2} > \sqrt{5}$, that implies that $D_{RVSTU} > 1$.

3.3 Expansion of any polygon

Let's generalize the fractal expansion process to any polygon. Given a polygon F with n side defined by the successive vertices $V_1, V_2, \cdots, V_n$. Without loss of generality, we

Figura 9 – Expansion of pentagon $RVSTU$ of Figure 5 into the hexagon $RWVSTU$ with fractal dimension equal to 1.



can make the expansion of F starting from the side V_1V_n. Let $V = (x, y)$ be the vertice added to the polygon F from the side V_1V_n. Then, we can obtain a polygon F' with $n+1$ sides whose successive vertices are $V_1, V_2, \cdots, V_n, V$.

The perimeter and area of F' are respectively given by

$$P(F') = P(F) - d + d_1 + d_2$$

$$A(F') = A(F) + A(T(x, y))$$

where $d = d(V_1, V_n)$, $d_1 = d(V_n, V)$, $d_2 = d(V, V_1)$ and $T(x, y)$ is the triangle of vertices V_1,V_n and V. If $V_1 = (x_1, y_1)$ and $V_n = (x_n, y_n)$, then the fractal expansion curve is given by equation

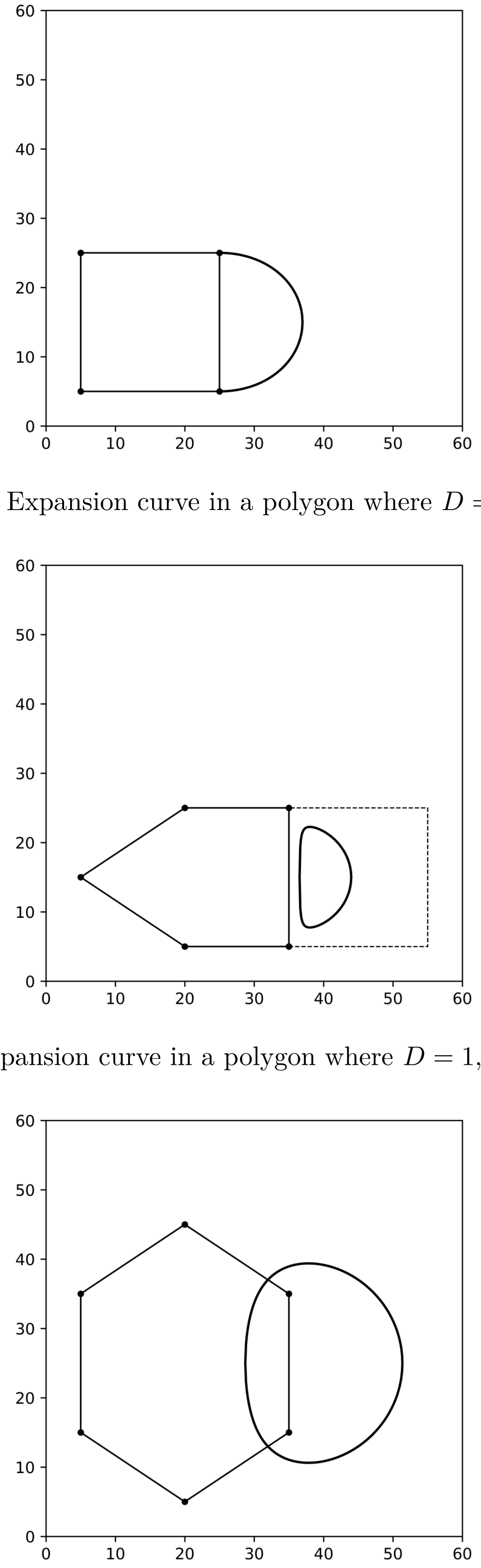
$$P(F) - d + \sqrt{(x - x_1)^2 + (y - y_1)^2} + \sqrt{(x - x_n)^2 + (y - y_n)^2} = 4\sqrt{A(F) + \frac{x_1y_n + x_ny + y_1x - x_1y - y_1x_n - y_nx}{2}} \tag{3.5}$$

3.4 Existence condition of the Expansion Curve

It's necessary for us to analyze the existence condition of the curve given by Equation (3.5). In order to simplify the calculus let's consider that the expansion side is over the oy axis. Note that this is not a restriction, since both, rotation and translation, are isometric transformations, and, thus, do not affect the fractal dimension.

Then, given a polygon F with successive vertices $V_1, V_2, \cdots, V_n$, let's expand the side V_1V_n, where $V_1 = (0, 0)$ and $V_n = (0, l)$. The length of V_1V_n is then l. Equation (3.5)

Figura 10 – The different possibilities for the expansion curve depending on the fractal dimension of the polygon.



(a) Expansion curve in a polygon where $D = 1$.

(b) Expansion curve in a polygon where $D = 1,00461$.

(c) Expansion curve in a polygon where $D = 0,98001$.

can be then rewritten as

$$P(F) - l + \sqrt{x^2 + y^2} + \sqrt{x^2 + (y-l)^2} = 4\sqrt{A(F) + \frac{lx}{2}} \tag{3.6}$$

Depending on the polygon's fractal dimension, the expansion curve presents different configurations. If $D = 1$, the side of expansion overlaps a section of the curve 10a). When $D > 1$, the expansion curve does not intersect the polygon and is limited by a square with side length l (Figure 10b). If $D > 1$, then a section of the curve is inside the polygon (Figure 10c).

In every case, the expansion curve intersects the axis $y = \frac{l}{2}$. By fixating $y = \frac{l}{2}$ in Equation (3.6) we obtain

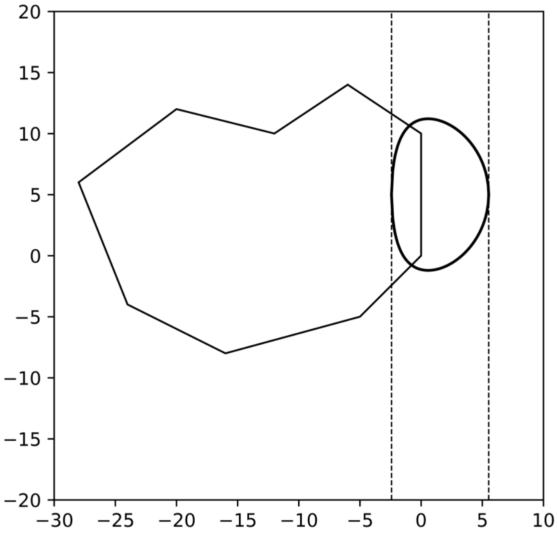
$$P(F) - l + \sqrt{x^2 + \left(\frac{l}{2}\right)^2} + \sqrt{x^2 + \left(\frac{l}{2} - l\right)^2} = 4\sqrt{A(F) + \frac{lx}{2}}$$

$$P(F) - l + \sqrt{x^2 + \frac{l^2}{4}} + \sqrt{x^2 + \frac{l^2}{4}} = 4\sqrt{A(F) + \frac{lx}{2}}$$

$$P(F) - l + 2\sqrt{x^2 + \frac{l^2}{4}} - 4\sqrt{A(F) + \frac{lx}{2}} = 0 \tag{3.7}$$

So, in order to define the curve's existence condition, it's enough to study under what conditions Equation (3.7) has real solutions.

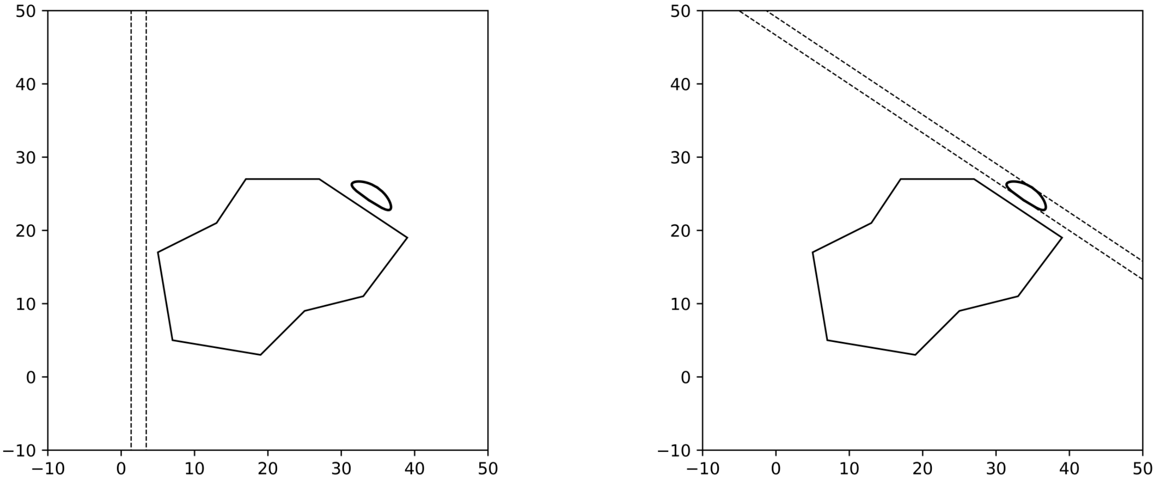
Figura 11 – The expansion curve is delimited by the lines $x = -2,42022$ and $x = 5,52158$, both solutions of the Equation (3.7) for the described polygon.



Therefore, given a side V_1Vn of the polygon F, the expansion curve exists if and only if $\exists x \in \mathbb{R}^+$ that satisfies Equation (3.7). It is important to point out that the equation is not restrained by the cases in which the expansion side overlaps the oy axis, however, if that's the case, and if x_1, x_2 are real solutions of Equation (3.7), then the lines $x = x_1$ and $x = x_2$ bound the expansion curve.

Nevertheless, through the inverse rotation and translation processes that would overlap the expansion side to the *oy* axis, it's possible to obtain two lines that bound the expansion curve of any side (Figure 12).

Figura 12 – Since the polygon expansion side does not coincide with the *oy* axis, the lines provided by the roots of the Equation (3.7) do not delimit the expansion curve, however, through rotation and translation inverse to those necessary to take the expansion side to the *oy* axis it is possible to make the lines delimit the expansion curve to a non-parallel side to the *oy* axis.



Furthermore, given that the expansion side length is one of the unknowns of Equation (3.7), in the same polygon there might be both expandable and non-expandable sides.

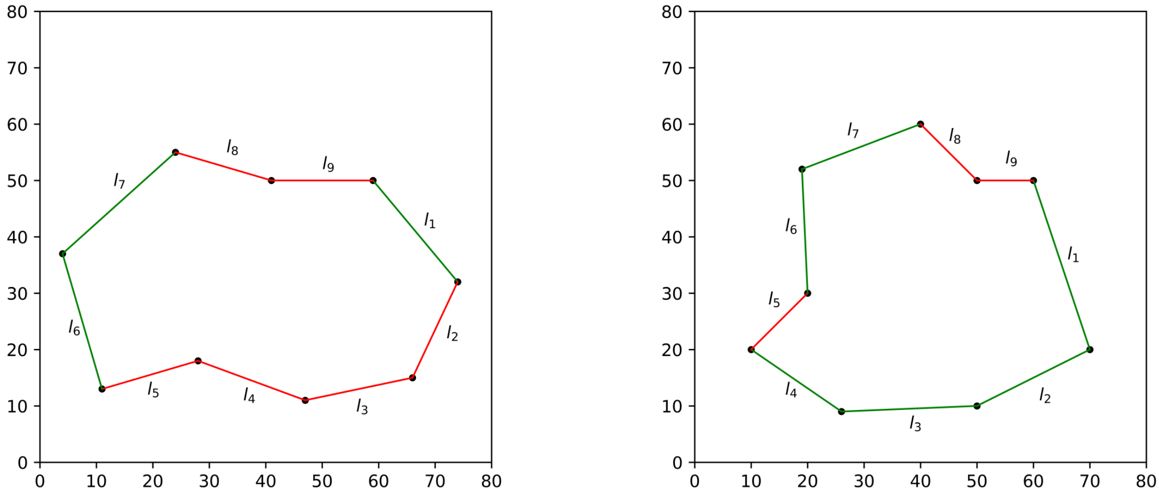
With the objective to test this theory two irregular polygons were created (Figure 13) and Equation (3.7) was applied in the values of length of each side in order to test which side could be expanded. By observing Tables 3 and 4 it's possible to notice that the sides that do not admit expansion are always the sorter.

In each of the examples, by fixing perimeter and area in Equation (3.7) and calculating the side length as a function of x, we can observe that there is a lower limit to the side length in order to the real solutions of equation to exist (Figure 14).

Since there is no upper limit to the length of a side in order for it to be expandable, the best way to test if a polygon can be expanded is to calculate Equation (3.7) to the longest side.

Let l_m be the length of the longest side of a polygon F, then F admit expansion if

Figura 13 – According to the results obtained with the application of the test given by Equation (3.7), the sides highlighted in green admit expansion, while the sides that do not admit expansion were highlighted in red.



(a) Polygon 1

(b) Polygon 2

Tabela 3 – The values of side length and roots of Equation (3.7) to each of the sided of the polygon in Figure 13a.

	Length (u.m.)	Roots
l_1	23,43075	$\{1{,}39593;\ 4{,}68283\}$
l_2	18,78829	$\varnothing$
l_3	19,41649	$\varnothing$
l_4	20,24846	$\varnothing$
l_5	17,72005	$\varnothing$
l_6	25,0	$\{1{,}22052;\ 5{,}75959\}$
l_7	26,90725	$\{1{,}07686;\ 7{,}09556\}$
l_8	17,72005	$\varnothing$
l_9	18,0	$\varnothing$

Tabela 4 – The values of side length and roots of Equation (3.7) to each of the sided of the polygon in Figure 13b.

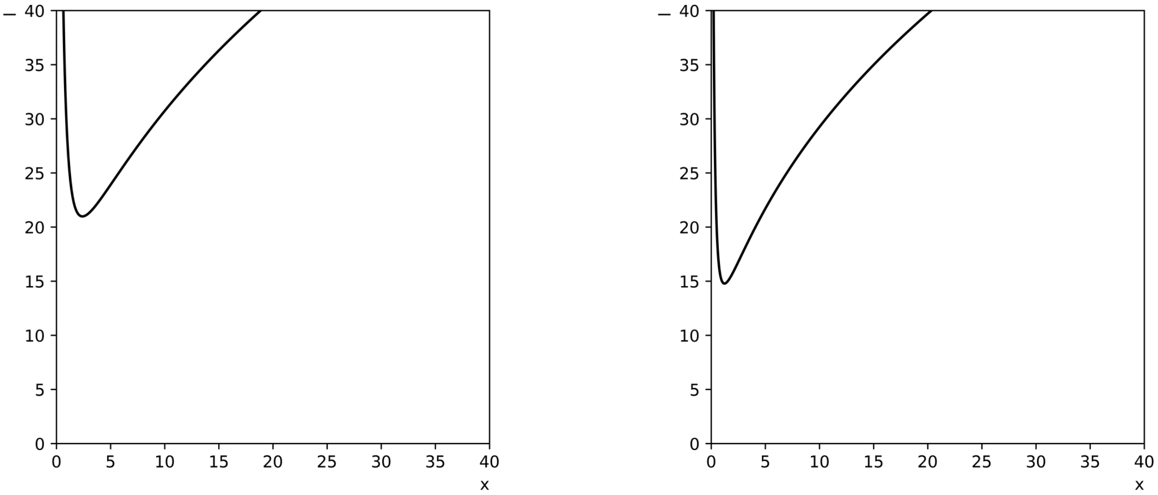
	Length (u.m.)	Roots
l_1	31,62278	$\{0{,}29241;\ 11{,}94003\}$
l_2	22,36068	$\{0{,}43704;\ 5{,}37903\}$
l_3	24,02082	$\{0{,}39976;\ 6{,}36857\}$
l_4	19,41649	$\{0{,}53104;\ 3{,}79283\}$
l_5	14,14214	$\varnothing$
l_6	22,02272	$\{0{,}44571;\ 5{,}18647\}$
l_7	22,47221	$\{0{,}43428;\ 5{,}44322\}$
l_8	14,14214	$\varnothing$
l_9	10,0	$\varnothing$

and only if the equation

$$P(F) - l_m + 2\sqrt{x^2 + \frac{l_m^2}{4}} - 4\sqrt{A(F) + \frac{l_m x}{2}} = 0 \tag{3.8}$$

has real solutions.

Figura 14 – Curves obtained by fixing the perimeter and area in the Equation (3.7) according to the values obtained in the polygons of the Figure 13.



(a) Graph of the polygon in Figure 13a, where $P = 187,23133$ u.m. and $A = 2178,5$ u.m.

(b) Graph of the polygon in Figure 13b, where $P = 180,19996$ u.m. and $A = 2025,0$ u.m.

Notice that the existence text is only needed for polygons whose perimeter-area dimension is greater than 1, since polygons with dimension less to 1 can always be expanded. To prove that, let's suppose that there is a polygon F with area A and perimeter P such that $l = 0$ satisfies Equation (3.7). Equation (3.7) can then be rewritten as

$$P + 2\sqrt{x^2} - 4\sqrt{A} = 0$$

so

$$|x| = \frac{4\sqrt{A} - P}{2}$$

thus $4\sqrt{A} \geq P$. If $P \geq 1$ and $A > 1$, then $2\frac{\ln\left(\frac{P}{4}\right)}{\ln(A)} < 1$. That is, F has a perimeter-area dimension equal to 1.

3.5 Application at landscape ecology

Although a patch with a fractal dimension close to or equal to 1 is initially seen as a bad thing since this is an indication of anthropic action (KRUMMEL et al., 1987;

LOPEZ; FROHN, 2017), for patches already affected by human action, expansion can be beneficial.

Since expanded polygons maintain the same perimeter-area ratio as the square, landscape patches recovered through expansion would also maintain this perimeter-area ratio, which would decrease the edge effect.

The Subsection 6.4.1 illustrates, using the program, the application of the landscape patch expansion method in the analysis of patches from the Catu River basin.

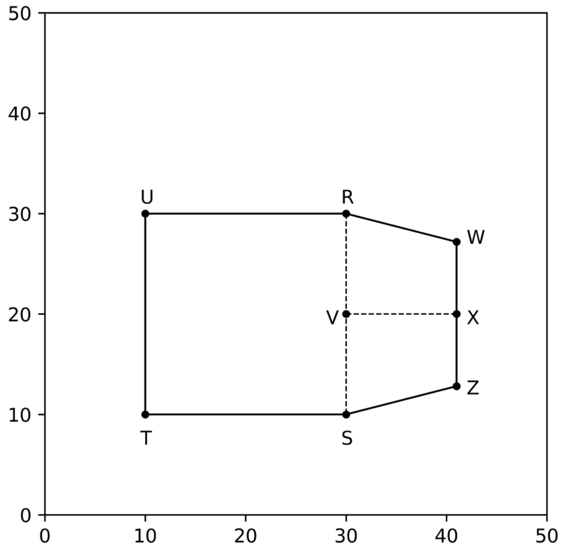
4 Trapezoidal Expansion

In Chapter 3 we defined a method for the recovery of forestry patches, so that they had the desired perimeter-area ratio, from the addition of a triangle. However, it's possible to develop the expansion of a polygon through the addition of other polygons. In this chapter, we develop the expansion of any polygon F to polygon F', with a fractal dimension equal to 1, through the addition of an isosceles trapezoid. We also explain the benefits of the development of expansion methods based on figures other than the triangle.

4.1 Starting the expansion from a square

Given a square with consecutive vertices R, S, T and U, we aim to define two points W and Z such that the hexagon $RWZSTU$ has fractal dimension, as per Equation (1.5), equal to 1 and the quadrilateral $RWZS$ is a isosceles trapezoid (Figure 15).

Figura 15 – Hexagon $RWZSTU$ with fractal dimension equal to 1 created through the trapezoidal expansion of the square $RSTU$.



In order to facilitate our calculus, let's suppose that the side RS is parallel to axis oy, notice that this is not a restriction, since, as said in Section 3.4, the fractal dimension is rotation invariant. In those conditions, if $R = (x_R, y_R)$ and $S = (x_S, y_S)$, then $x_R = x_S = x_{RS}$, therefore the points W and Z can the described as

$$W = \left(x_{RS} + i, \frac{y_R + y_S}{2} + j\right)$$

$$Z = \left(x_{RS} + i, \frac{y_R + y_S}{2} - j\right)$$

where $i, j \in \mathbb{R}^+$.

The length of the sides RS, RW, WZ and ZS are respectively given by

$$d(R,S) = |y_S - y_R|$$
$$d(R,W) = \sqrt{i^2 + \left(\frac{y_S - y_R}{2} + j\right)^2}$$
$$d(W,Z) = 2j$$
$$d(Z,S) = \sqrt{i^2 + \left(\frac{y_S - y_R}{2} + j\right)^2}$$

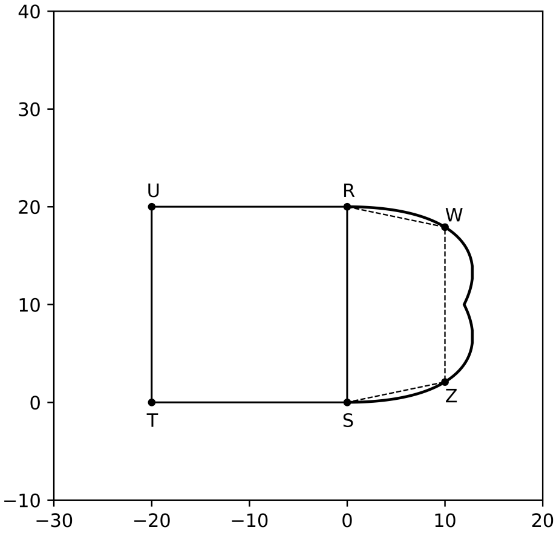
Then, the perimeter and area of hexagon $RWZSTU$ are respectively

$$P_{RWZSTU} = 3|y_S - y_R| + 2j + 2\sqrt{i^2 + \left(\frac{y_S - y_R}{2} + j\right)^2}$$
$$A_{RWZSTU} = |y_S - y_R|^2 \frac{(|y_S - y_R| + 2j)\, i}{2}$$

In order to $RWZSTU$ have dimension, D_{RWZSTU}, equal to 1 we need then that

$$3|y_S - y_R| + 2j + 2\sqrt{i^2 + \left(\frac{y_S - y_R}{2} + j\right)^2} = 4\sqrt{|y_S - y_R|^2 + \frac{(|y_S - y_R| + 2j)\, i}{2}} \tag{4.1}$$

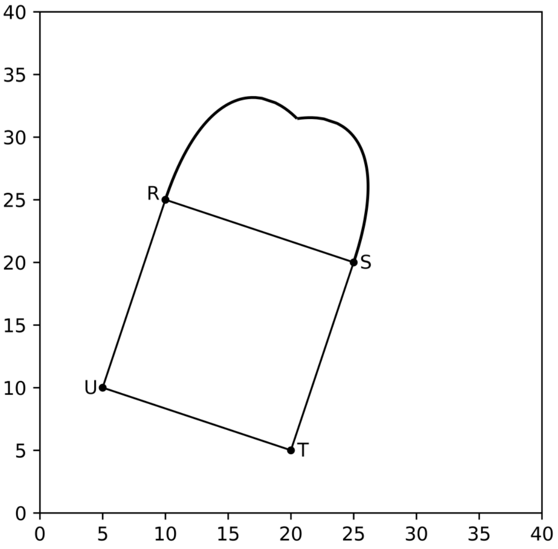
Figura 16 – Locus of points W and Z such that $RWZSTU$ is a hexagon with fractal dimension equal to 1, according to Equation (1.5), and the quadrilateral $RWZS$ is an isosceles trapezoid, for a square $RSTU$ whose side RS is parallel to the axis oy and S coincides with the origin.



We can further simplify our calculus, without compromising the generality of the method, by translating the side RS, already rotated, in order to have $R = (0, l)$ and $S = (0, 0)$. In those conditions

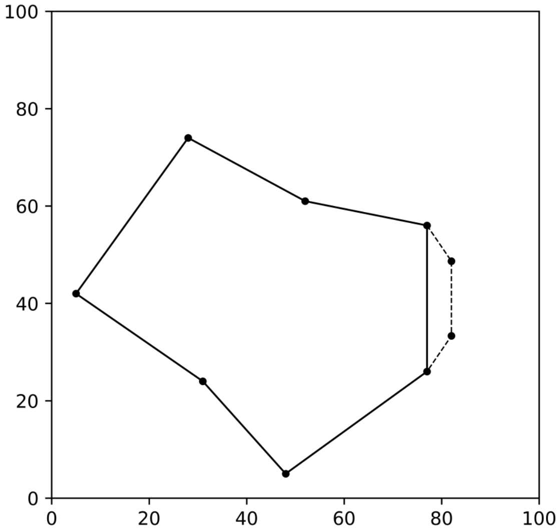
$$3l + 2j + 2\sqrt{i^2 + \left(j - \frac{l}{2}\right)^2} = 4\sqrt{l^2 + \frac{(l + 2j)i}{2}} \tag{4.2}$$

Figura 17 – Expansion curve from square $RSTU$ in which RS not parallel to axis oy.



4.2 Expansion of any polygon

Figura 18 – Expansion of a heptagon to an enneagon with fractal dimension equal to 1.



To generalize the trapezoidal expansion process to any polygon, define a polygon F with n consecutive vertices $V_1, V_2, \cdots, V_n$. If the side V_1V_n is parallel to axis oy and V_n, V_{n+1} and V_{n+2} are the vertices added to F by trapezoidal expansion forming a polygon F', then the perimeter and area of F' are respectively given by

$$P(F') = P(F) - d(V_1, V_n) + d(V_1, V_{n+1}) + d(V_{n+1}, V_{n+2}) + d(V_{n+2}, V_n)$$

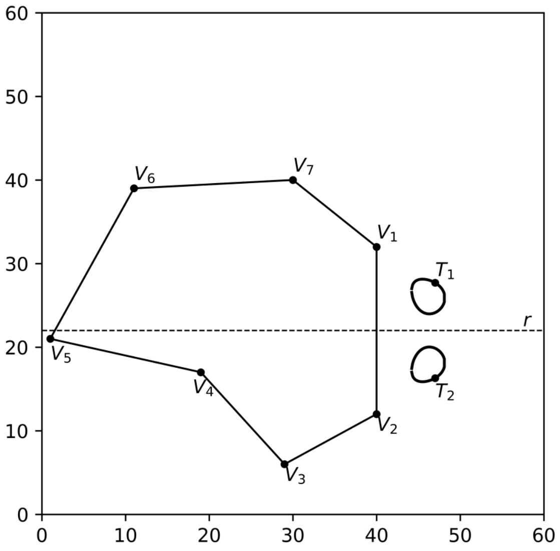
$$A(F') = A(F) + A(T)$$

where T is the isosceles trapezoid of vertices V_1, V_{n+1}, V_{n+2} e V_n. Then, the fractal stability curve is given by the equation

$$P(F) - |y_n - y_1| + 2j + 2\sqrt{i^2 + \left(\frac{y_n - y_1}{2} + j\right)^2} = 4\sqrt{A(F) + \frac{(|y_n - y_1| + 2j)\, i}{2}} \quad (4.3)$$

It is necessary to point out that the trapezoidal expansion curve is symmetric in relation to the orthogonal line to the expansion side and that passes through the midpoint. Each side of the curve provides one of the vertices of the trapezoid.

Figura 19 – The trapezoidal expansion curve of the side V_1V_2 of the heptagon $V_1V_2V_3V_4V_5V_6V_7$ is symmetric with respect to the line r which is orthogonal to the expansion side and passes through its midpoint.



An existence condition for the trapezoidal expansion curve was not found, however, j is always less than the length of the expansion side.

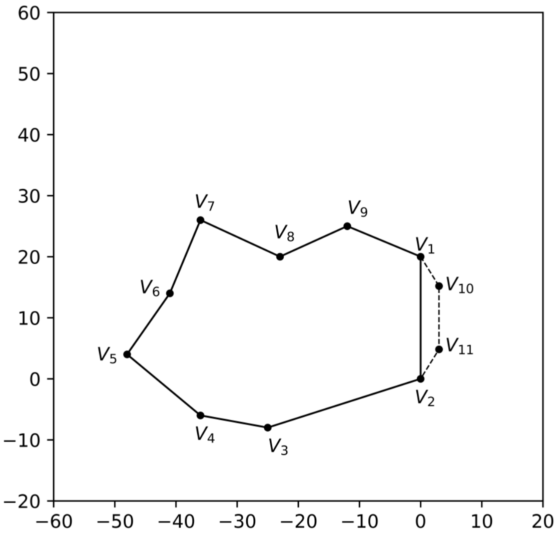
4.3 Triangle expansion × trapezoidal expansion

Note that, to $j = 0$, $V_{n+1} = V_{n+2}$ and Equation (4.3) provides an isosceles triangle, that is, Equation (4.3) becomes equal to Equation (3.6) to $y = \frac{l}{2}$. Therefore, we conclude that whenever triangle expansion is possible, so is trapezoidal expansion.

The advantage of the trapezoidal expansion over the triangle expansion is that there are cases where it is not possible to expand a polygon by triangle, but the trapezoidal expansion can be done (Figure 20).

These additional expansion possibilities provide a range of new fragments that can be recovered.

Figura 20 – The enneagon E of consecutive vertices V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8 and V_9 has perimeter and area respectively equal to $137,65707138873026$ and $1168,5$. According to Equation (3.7), its side V_1V_2 with length 20 cannot be expanded, however through pentagon expansion we get two points V_{10} and V_{11} that expand E into an undecagon with fractal dimension equal to 1.



If only the trapezoidal expansion curve exists, then the sides of the expansion curve do not touch, since an isosceles triangle expansion would not make the polygon's dimension equal to 1 (Figure 20).

5 Geometric interpretation of the fractal dimension's proportionality

5.1 Proportionality constant of a regular polygon

Figures 3a and 3b show that the graph of the fractal dimension, given by Equation (1.5), of an equilateral triangle and a regular pentagon as a function of their side length are not constant. However, Figure 3c shows that the fractal dimension, given by Equation (1.5), of a square is always equal to 1 regardless of the side length.

Figure 21 shows that, when the proportionality constant of Equation (1.3), the graph of the fractal dimension of a regular pentagon as a function of its side length behaves differently.

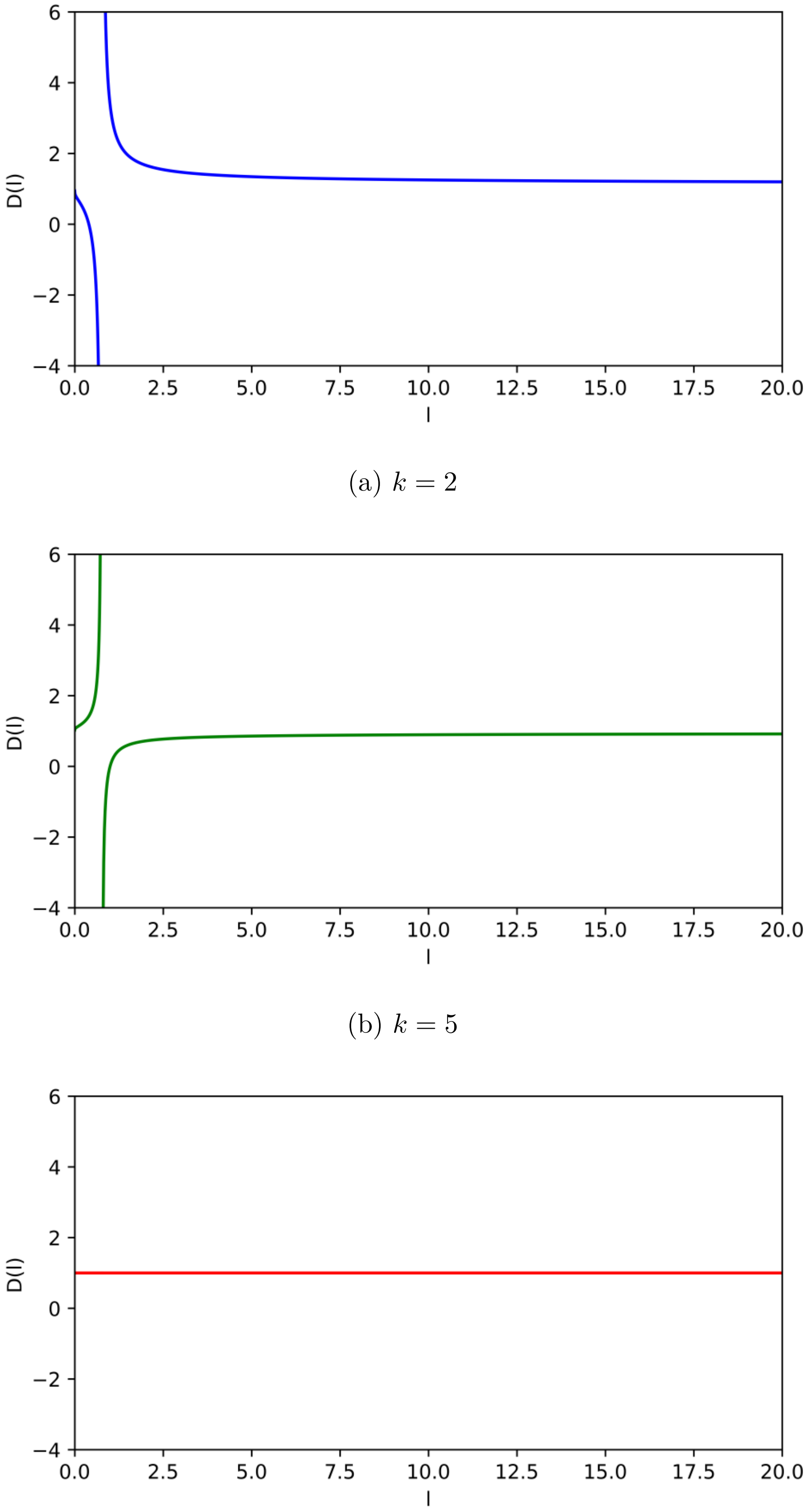
To k equal to $2\sqrt{5\tan\left(\frac{\pi}{5}\right)}$ which is the value of the proportionality constant of a regular pentagon by Equation (1.9), the graph is constant (Figure 21c), yet to values of k different from the defined by Equation (1.9) , the graph show varied behaviors as k becomes more of less than $2\sqrt{5\tan\left(\frac{\pi}{5}\right)}$.

The same happens to other regular polygons if their fractal dimensions are calculated with their respective proportionality constants, that is, by using Equation (1.9). Since a regular polygon of n sides has proportionality constant, according to Equation (1.9), equal to $2\sqrt{n\tan\left(\frac{\pi}{n}\right)}$, the value of their fractal dimension F, as a function of its side length is then given by

$$D(l) = 2\frac{\ln\left(\frac{P}{2\sqrt{n\tan\left(\frac{\pi}{n}\right)}}\right)}{\ln A} = 2\frac{\ln\left(\frac{nl}{2\sqrt{n\tan\left(\frac{\pi}{n}\right)}}\right)}{\ln\left(\frac{nl^2}{4\tan\left(\frac{\pi}{n}\right)}\right)} = \frac{\ln\left(\frac{nl^2}{4\tan\left(\frac{\pi}{n}\right)}\right)}{\ln\left(\frac{nl^2}{4\tan\left(\frac{\pi}{n}\right)}\right)} = 1$$

Therefore, it is possible to attach a geometric meaning to the constant of proportionality, which contradicts Chen that affirms that "in the framework of fractal geometry, the proportionality coefficient bears little information" (CHEN, 2020, p. 12).

Figura 21 – Graphs of the fractal dimension, $D(l)$, of a regular pentagon as a function of its side length l for different values of k.



(a) $k = 2$

(b) $k = 5$

(c) $k = 2\sqrt{5\tan\left(\frac{\pi}{5}\right)} \approx 3,81194$

5.2 Proportionality constant of any polygon

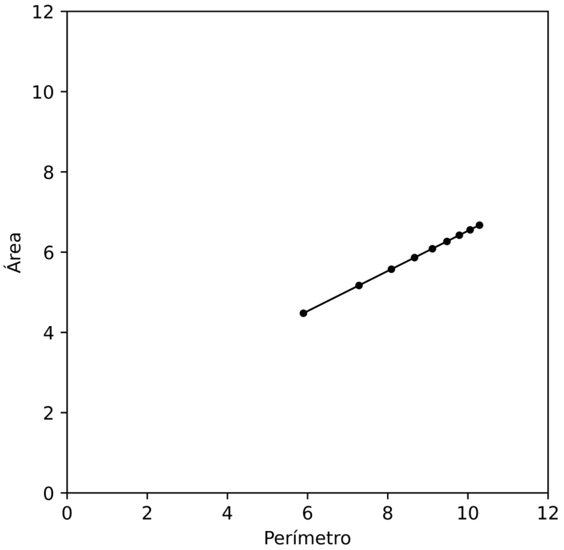
Note that, given any polygon F, if the fractal dimension of F, according to Equation (1.5) is equal to 1, then F has the same fractal dimension that the square:

$$D(F) = 1 \Rightarrow 2 \cdot \frac{\ln\left(\frac{P(F)}{4}\right)}{\ln\left(A(F)\right)} = 1 \Rightarrow P(F) = 4\sqrt{A(F)}$$

It's possible then to ascertain the proportionality constant associated with any polygon.

One of the methods to determine the constant is with the use of linear regression on the values of the perimeter's logarithm by the area's logarithm of different polygons obtained through homothety, however, instead of the slope of the line, which gives us the fractal dimension, the constant is obtained through the value of intercept y.

Figura 22 – Regression line of the logarithm of the perimeter against the logarithm of the area of the homotheties of the polygon seen in Figure 2.



By applying successive homotheties to the polygon in Figure 2 and applying linear regression on the data, we obtain the line $0,5x + 1,52827$ seen in Figure 2. Thus, the proportionality constant of the polygon is $e^{1,52827} \approx 4,61019$. Note that the fractal dimension is twice the slope of the regression line, therefore equal to 1.

However, the exponential function has a high rate of change, so the number of decimal places chosen can make a big difference in determining the perimeter-area proportionality constant. Thus, a different method for determining the constant is derived from Equation (1.2), rewritten as

$$k = \frac{P}{\sqrt{A}} \tag{5.1}$$

Tabela 5 – The values of homothety ratio (R_H), perimeter (P) area (A) used at the graph in Figure 22.

R_H	P (u.m.)	A (u.m.)
1	87,92204	363,715
2	175,84408	1454,86
3	263,76612	3273,435
4	351,68816	5819,44
5	439,6102	9092,875
6	527,53224	13093,74
7	615,45428	17822,035
8	703,37632	23277,76
9	791,29836	29460,915

We prove that, given any polygon F, to $k = \dfrac{P(F)}{\sqrt{A(F)}}$, all of the polygons obtained from F through homothety have fractal dimension equal to 1. In fact, given a polygon F with vertices $v_i = (x_i, y_i), i \in \{1, \ldots, n\}$. We know that the area of F is given by the equation

$$A(F) = \frac{(x_1y_2 + \ldots + x_ny_1) - (y_1x_2 + \ldots + y_nx_1)}{2}$$

while the perimeter of F is given by

$$P(F) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} + \ldots + \sqrt{(x_n - x_1)^2 + (y_n - y_1)^2}$$

If F_λ is a homothety of F of ratio λ, with $\lambda \in \mathbb{R}^*$. Then the vertices of F_λ are the points $\lambda v_i = (\lambda x_i, \lambda y_i), i \in \{1, \ldots, n\}$. Consequently the area of F_λ is equal to

$$A(F_\lambda) = \frac{(\lambda x_1\lambda y_2 + \ldots + \lambda x_n\lambda y_1) - (\lambda y_1\lambda x_2 + \ldots + \lambda y_n\lambda x_1)}{2} =$$

$$\lambda^2\left[\frac{(x_1y_2 + \ldots + x_ny_1) - (y_1x_2 + \ldots + y_nx_1)}{2}\right] = \lambda^2 A(F)$$

while its perimeter is

$$P(F_\lambda) = \sqrt{(\lambda x_1 - \lambda x_2)^2 + (\lambda y_1 - \lambda y_2)^2} + \ldots + \sqrt{(\lambda x_n - \lambda x_1)^2 + (\lambda y_n - \lambda y_1)^2}$$

$$\lambda\left[\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} + \ldots + \sqrt{(x_n - x_1)^2 + (y_n - y_1)^2}\right] = \lambda P(F)$$

If $k = \dfrac{P(F)}{\sqrt{A(F)}}$, then the fractal dimension of F_λ is

$$D(F_\lambda) = 2\frac{\ln\left(\dfrac{P(F_\lambda)}{k}\right)}{\ln(A(F_\lambda))} = 2\frac{\ln\left(\dfrac{P(F_\lambda)}{\dfrac{P(F)}{\sqrt{A(F)}}}\right)}{\ln(A(F_\lambda))} = 2\frac{\ln\left(\dfrac{\lambda P(F)}{\dfrac{P(F)}{\sqrt{A(F)}}}\right)}{\ln(\lambda^2 A(F))} = \frac{\ln\left(\left(\lambda\sqrt{A(F)}\right)^2\right)}{\ln(\lambda^2 A(F))} = 1$$

Tabela 6 – Values of k, according to Equation (5.1) to some of the polygons presented in this thesis.

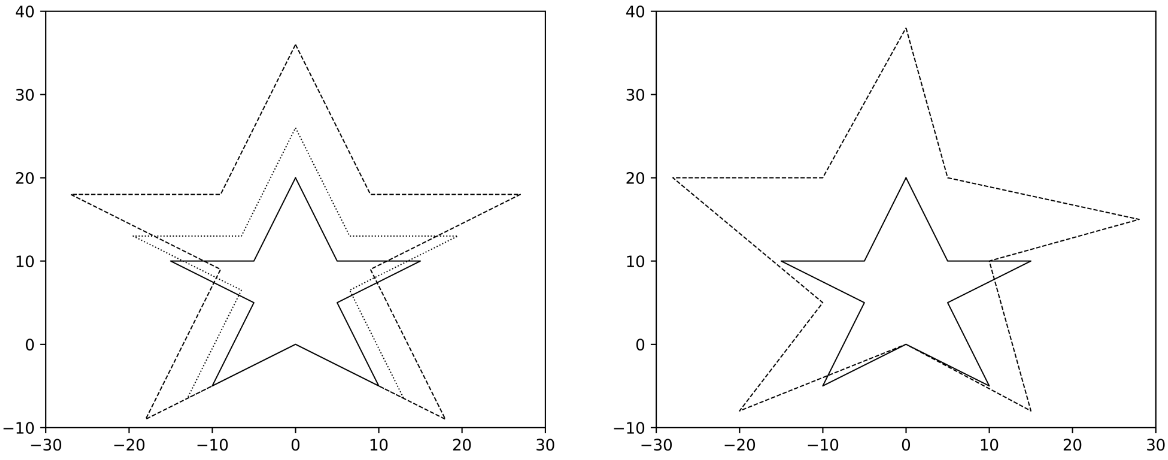
Figure Nº	k
2	4,61017
10b	4,0567
10c	3,73703
11	3,89018
12	4,02644
13a	4,01143
13b	4,00444

5.3 The constant's in the analysis of natural elements

Given any natural element, it is possible to calculate its approximate proportionality constant. If we fix k for this object, then when we calculate the fractal dimension of a sample of the same type of element by Equation (1.3), the dimension value therefore becomes a measure of how much the perimeter-area ratio of the sample deviates from the standard (TRIPATHI et al., 2015).

In natural elements that follow a specific shape, such as honeycombs or leaves of certain plants, samples with a dimension very far from 1 indicate samples that have lost their specific shape, which in many cases, like that of plants, can be a sign of disease.

Figura 23 – For natural elements where shape is expected to be maintained during growth, each subsequent phase is an approximate homothety of the original and should maintain a perimeter-area dimension close to 1 for the original object constant.



(a) Shape is maintained by growth.

(b) Shape is deformed by growth.

In forest patches, by defining k from a patch that presents a good perimeter-area ratio, it is possible to adapt the expansion developed in Chapter 3 exchanging the value 4 in Equation (3.5) by the desired value of k.

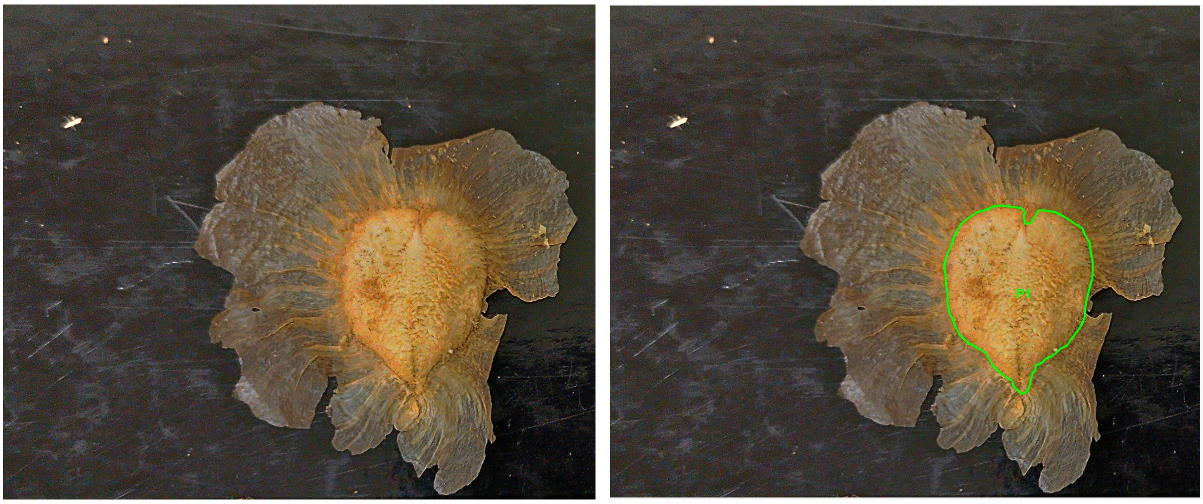
Thus, if $V_1 = (x_1, y_1), V_2 = (x_2, y_2), \cdots, V_n = (x_n, y_n)$ are successive vertices of a polygon F. The points V such that the polygon of successive vertices $V_1V_2 \cdots V_nV$ has fractal dimension equal to 1, are the points (x, y) that satisfy the equation

$$P(F) - d + \sqrt{(x - x_1)^2 + (y - y_1)^2} + \sqrt{(x - x_n)^2 + (y - y_n)^2} = k\sqrt{A(F) + \frac{x_1y_n + x_ny + y_1x - x_1y - y_1x_n - y_nx}{2}} \quad (5.2)$$

5.3.1 Application

As an example of the application of the proportionality constant's analysis a study of the shape of 33 samples of the seed *Zeyheria tuberculosa* (Vell.) Bureau ex Verl was developed. With the use of the program FracFor, the seeds had their outline delineated so that the values of perimeter, area and proportionality constant of each one of them could be measured.

Figura 24 – Photo of Sample 01 of the seeds of *Zeyheria tuberculosa* (Vell.) Bureau ex Verl. The image on the right shows the outline of the seed made with the use of the FracFor program.



Then the seeds were ordered according to the value of their areas and the samples were then divided into 3 groups, A, B and C, with 11 seeds each. The seeds of each group then had their perimeter-area dimension calculated using the proportionality constant of the seed with the smallest area in that group (Figure 25).

The area (A), perimeter (P), proportionality constant k and perimeter-area dimension (D_F) values of the samples of each group can be seen in the Tables 7, 8 and 9.

The area and perimeter values are respectively in mm^2 and mm, while the perimeter-area constant and dimension values are dimensionless. In Group A, the dimension was calculated using the constant from Sample 03 (Figure 25a), while in Group B used the

Tabela 7 – Data of Group A.

Seed Nº	A (mm^2)	P (mm)	k	D_F
14	194,68684	54,12209	3,87888	0,96390
12	190,93159	55,23719	3,99754	0,97524
21	189,04608	51,41560	3,73948	0,94974
11	187,00769	52,44809	3,83531	0,95931
27	186,67888	54,19304	3,96639	0,97215
06	181,67755	51,62298	3,82994	0,95854
07	181,02114	51,70817	3,84321	0,95984
04	180,83186	54,63497	4,06287	0,98122
01	180,29314	53,17112	3,95992	0,97133
16	174,16705	51,39544	3,89441	0,96467
03	168,06523	55,30505	4,26605	1,00000

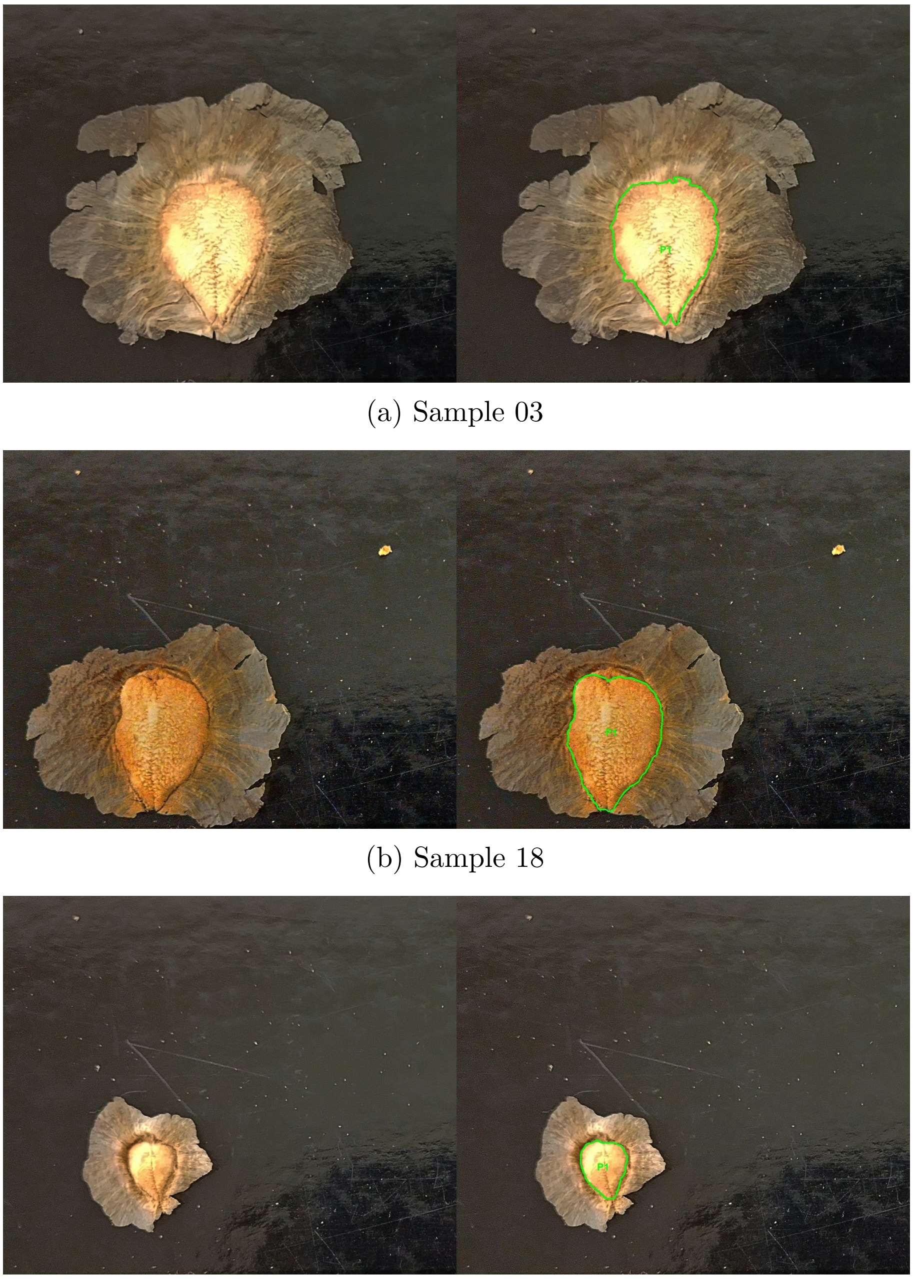
Tabela 8 – Data of Group B.

Seed Nº	A (mm^2)	P (mm)	k	D_F
33	163,53224	49,25707	3,85183	0,98918
05	161,42348	51,54862	4,05727	1,00959
02	160,09852	50,16492	3,96466	1,00051
13	159,78184	49,54568	3,91960	0,99600
20	156,15400	50,07425	4,00717	1,00473
30	150,85782	47,63910	3,87864	0,99177
17	150,84690	49,53350	4,03302	1,00733
28	147,97132	47,07926	3,87027	0,99087
19	147,47628	49,24914	4,05544	1,00958
29	146,27023	49,69670	4,10913	1,01488
18	145,67813	47,79063	3,95955	1,00000

Tabela 9 – Data of Group C.

Seed Nº	A (mm^2)	P (mm)	k	D_F
08	144,90524	47,03896	3,90765	1,00799
10	134,67689	46,21284	3,98214	1,01581
26	129,28122	47,48729	4,17647	1,03555
15	124,16826	48,10404	4,31694	1,04956
09	113,27865	43,38847	4,07662	1,02631
31	101,55306	39,35124	3,90492	1,00830
32	78,51076	37,92161	4,27979	1,05081
22	75,03094	39,28655	4,53549	1,07822
24	60,62996	30,80901	3,95671	1,01577
25	40,45597	26,39739	4,15021	1,04330
23	30,35138	21,10429	3,83073	1,00000

Figura 25 – Seeds of *Zeyheria tuberculosa* (Vell.) Bureau ex Verl whose perimeter-area proportionality constants were used to calculate the group dimension.



(a) Sample 03

(b) Sample 18

(c) Sample 23

constant from Sample 18 (Figure 25b) and in Group C, the constant from Sample 23 (Figure 25c). All values are approximated to 5 decimal places.

Analyzing each table, it can be seen that all the seeds in Group A presented a perimeter-area dimension less than 1, while in Group C all the samples had a dimension greater than 1, Group B, however, presented samples with dimensions greater and less than 1.

The standard deviation of samples' dimension values of each of the groups was also calculated (Table 10). Groups B and C presented the lowest and highest standard deviation, respectively.

Tabela 10 – Standard deviation of the values of perimeter-area dimension to each group.

	σ (mm)
Group A	0,01366
Group B	0,00864
Group C	0,02371

6 Program

In order to apply patch expansion in practice, it was planned the development of a program that enables the analysis of forest patches in GIS environments. The first step was the creation, in Python language, of different functions to the calculus and exhibition of the expansion curves.

6.1 Support Functions

Aiming to simplify the calculus in order to decrease the data's processing period, the program was based on the simplified version of triangle expansion equation (Equation (3.6)), that is, in cased in which the side of expansion is overlapped with the *oy* axis and one of the vertices coincident with the origin.

Therefore, it was necessary to create functions to translate (Function 6.2) and rotate (Function 6.4) the side of expansion, but it was necessary to primarily define Functions 6.1 and 6.3 to automate the definitions of translation reference point and the rotation angle. Additionally, a parameter *tipo* was added to Functions 6.2 and 6.4 in order to allow those functions to perform the reverse processes of rotation and translation, taking the points to their original coordinate.

Listing 6.1 – Suport function of translation.

```python
def referencia_translacao(ponto1,ponto2):
    if ponto1[1] < ponto2[1]:
        return ponto1
    elif ponto1[1] == ponto2[1]:
        return ponto2
    elif ponto1[1] > ponto2[1]:
        return ponto2
```

Listing 6.2 – Function to translate points.

```python
def translacao(ponto,referencia,tipo=""):
    if tipo == "r":
        return (ponto[0]+referencia[0],ponto[1]+referencia
           ↪ [1])
    else:
        return (ponto[0]-referencia[0],ponto[1]-referencia
           ↪ [1])
```

Listing 6.3 – Suport function of rotation.

```
def angulo_rotacao(ponto1,ponto2):
    ponto3 = (ponto1[0],ponto2[1])
    if ponto1[1] == ponto2[1]:
        return math.radians(90)
    else:
        return math.radians(math.degrees(math.atan2(ponto3[1]-ponto1[1], ponto3[0]-ponto1[0]) - math.atan2(ponto2[1]-ponto1[1], ponto2[0]-ponto1[0])))
```

Listing 6.4 – Function to rotate points.

```
def rotacao(ponto,angulo,tipo=""):
    if tipo == "r":
        x = ponto[0]*math.cos(-angulo)-ponto[1]*math.sin(-angulo)
        y = ponto[0]*math.sin(-angulo)+ponto[1]*math.cos(-angulo)
        x,y = aproximar(x,9), aproximar(y,9)
        return (x,y)
    else:
        x = ponto[0]*math.cos(angulo)-ponto[1]*math.sin(angulo)
        y = ponto[0]*math.sin(angulo)+ponto[1]*math.cos(angulo)
        x,y = aproximar(x,9), aproximar(y,9)
        return (x,y)
```

Function 6.5 was created to round the values, in order to accept the options to round up and down.

Listing 6.5 – Function to round values.

```
def aproximar(numero,casas="",tipo=""):
    if casas == "":
        return numero
    else:
        if len(tipo) != 0:
            if str(tipo)[0].lower() == "c":
                return round(numero+5*10**((-1)*(casas+1)), casas)
            elif str(tipo)[0].lower() == "b":
                return round(numero-5*10**((-1)*(casas+1)),
                   ↪  casas)
            else:
                return round(numero,casas)
        else:
            return round(numero,casas)
```

To the calculus of the constants of Equation (3.6) it was created Functions 6.6, 6.7, 6.8 and 6.9 respectively to the calculus of patch's expansion side length, perimeter, area and dimension.

Listing 6.6 – Function to the calculus of a polygon's side length.

```
def comprimento_lado(ponto1,ponto2,casas=""):
    cl = float(sp.sqrt((ponto1[0]-ponto2[0])**2+(ponto1[1]-
       ↪  ponto2[1])**2))
    return aproximar(cl,casas)
```

Listing 6.7 – Function to the calculus of a polygon's perimeter.

```
def perimetro(poligono,casas="",tipo=""):
    p = math.fabs(poligono.perimeter)
    return aproximar(p,casas,tipo)
```

Listing 6.8 – Function to the calculus of a polygon's area.

```
def area(poligono,casas="",tipo=""):
    a = math.fabs(poligono.area)
    return aproximar(a,casas,tipo)
```

Listing 6.9 – Function to the calculus of a polygon's perimeter-area dimension.

```
def dimensao(poligono,constante=4,casas="",tipo=""):
    d = 2*sp.log(perimetro(poligono)/constante)/sp.log(area(
       ↪  poligono))
    return aproximar(d,casas,tipo)
```

Listing 6.10 – Function to the calculus of the perimeter-area dimension' constant of proportionality.

```
def constante_geral(poligono,casas="",tipo=""):
    k = perimetro(poligono)/math.sqrt(area(poligono))
    return aproximar(k,casas,tipo)
```

An additional parameter was inserted in Function 6.9 so that the perimeter-area dimension could be calculated to $k \neq 4$ and Function 6.10 was defined according to Equation 5.1 to the calculus of perimeter-area dimension's proportionality constant.

6.2 Function to Triangle Expansion

After finishing the definition of the support functions, the functions to the construction of the triangle expansion curve were created. Foremost Equation (3.6) was defined in Function 6.11. An optional parameter k was added to allow expansion with constant unlike 4.

Listing 6.11 – Function with Equation (3.6).

```
def equacao_expansao_triangulo(poligono,ponto1,ponto2,k=4):
    x, y = sp.symbols(’x␣y’)
    P = perimetro(poligono)
    A = area(poligono)
    l = comprimento_lado(ponto1,ponto2)
    return sp.Eq(P-l+sp.sqrt((x-ponto1[0])**(2)+(y-ponto1[1])
       ↪ **(2))+sp.sqrt((x-ponto2[0])**(2)+(y-ponto2[1])
       ↪ **(2))-k*sp.sqrt(A+((x*ponto1[1]+ponto1[0]*ponto2
       ↪ [1]+ponto2[0]*y-(y*ponto1[0]+ponto1[1]*ponto2[0]+
       ↪ ponto2[1]*x))/(2))), 0)
```

After that, the curve existence condition equation (Equation (3.7)) was defined in Function 6.12, also with an optional parameter k. It was necessary to limit the solution set to real results, it was also necessary to convert the results to float and order them to be used as the curve boundary.

Listing 6.12 – Function to test the curve existence as defined in Equation (3.7).

```
def existencia_curva_triangulo(poligono,ponto1,ponto2,k=4):
    x = sp.symbols(’x’)
    P = perimetro(poligono)
    A = area(poligono)
    l = comprimento_lado(ponto1,ponto2)
    equacao = sp.Eq(P-l+2*sp.sqrt(x**2+l**2/4)-4*sp.sqrt(A+l*
       ↪ x/2),0)
    raizes = list(sp.solveset(equacao,x,domain=sp.S.Reals))
    if len(raizes) == 2:
        raizes.sort()
        raizes[0] = float(raizes[0])
        raizes[1] = float(raizes[1])
        return raizes
    else:
        return []
```

Function 6.13 returns two lists with the x and y coordinates of the expansion curve points and, since the curve is calculated point to point, the optional parameter *passo* sets the distance between the calculated points. Generally, the side of expansion is translated and rotated to *oy* axis, then the curve points are calculated to the values within the boundary set by Function 3.7 and then the rotation and translation processes are reversed, taking the side and the curve to the original coordinates. The function counts with a mechanism to detect if the expansion should take place to right or left based on the order and position of the side's vertices.

Listing 6.13 – Function that calculates the points of the triangle expansion curve.

```
def curva_expansao_triangulo(poligono,ponto1,ponto2,k=4,passo
  ↪ =1):
    x, y = sp.symbols('x␣y')
    dominio = [list(),list()]
    imagem = [list(),list()]
        if (ponto1[0] == ponto2[0] and ponto1[1] > ponto2[1])
          ↪  or (ponto1[0] < ponto2[0] and ponto1[1] >
          ↪ ponto2[1]) or (ponto1[0] > ponto2[0] and ponto1
          ↪ [1] == ponto2[1]) or (ponto1[0] > ponto2[0] and
          ↪  ponto1[1] > ponto2[1]):
        referencia = referencia_translacao(ponto1,ponto2)
        angulo = angulo_rotacao(ponto1,ponto2)
        ponto1t,ponto2t = [translacao(ponto1,referencia),
          ↪ translacao(ponto2,referencia)]
        ponto1r,ponto2r = [rotacao(ponto1t,angulo),rotacao(
          ↪ ponto2t,angulo)]
        limites = existencia_curva_triangulo(poligono,ponto1r
          ↪ ,ponto2r,k)
        for m in np.append(np.arange(limites[0],limites[1],
          ↪ passo),limites[1]):
            solucao = sp.solve(equacao_expansao_triangulo(
              ↪ poligono,ponto1r,ponto2r,k).subs(x, m), y)
            if len(solucao) != 0:
                for n in range(0,len(solucao)):
                    ponto_curva = (m,converter_float(solucao[
                        ↪ n]))
                    ponto_curva = rotacao(ponto_curva,angulo,
                        ↪ 'r')
                    ponto_curva = translacao(ponto_curva,
                        ↪ referencia,'r')
                    dominio[n].append(ponto_curva[0])
                    imagem[n].append(ponto_curva[1])
    elif (ponto1[0] == ponto2[0] and ponto1[1] < ponto2[1])
       ↪ or (ponto1[0] < ponto2[0] and ponto1[1] == ponto2
       ↪ [1]) or (ponto1[0] < ponto2[0] and ponto1[1] <
       ↪ ponto2[1]) or (ponto1[0] > ponto2[0] and ponto1[1]
       ↪ < ponto2[1]):
        referencia = referencia_translacao(ponto2,ponto1)
        angulo = angulo_rotacao(ponto2,ponto1)
        ponto1t,ponto2t = [translacao(ponto2,referencia),
           ↪ translacao(ponto1,referencia)]
        ponto1r,ponto2r = [rotacao(ponto2t,angulo),rotacao(
           ↪ ponto1t,angulo)]
        limites = existencia_curva_triangulo(poligono,ponto2r
           ↪ ,ponto1r,k)
        for m in np.append(np.arange(limites[0],limites[1],
           ↪ passo),limites[1]):
            solucao = sp.solve(equacao_expansao_triangulo(
               ↪ poligono,ponto2r,ponto1r,k).subs(x, m), y)
            if len(solucao) != 0:
                for n in range(0,len(solucao)):
                    ponto_curva = (-m,converter_float(solucao
                        ↪ [n]))
                    ponto_curva = rotacao(ponto_curva,angulo,
                        ↪ 'r')
                    ponto_curva = translacao(ponto_curva,
                        ↪ referencia,'r')
                    dominio[n].append(ponto_curva[0])
                    imagem[n].append(ponto_curva[1])
    return dominio, imagem
```

Depending on the computer's processing speed, the curve may take a long time to be calculated, thus Function 6.14 was created in order to calculate a single point of the curve.

Listing 6.14 – Function to obtain a single point of the triangle expansion curve.

```
def ponto_curva_triangulo(poligono,ponto1,ponto2,coordenada1,
  ↪ k=4,tipo='x',numero_solucao=1):
    x, y = sp.symbols('x␣y')
    if (ponto1[0] == ponto2[0] and ponto1[1] > ponto2[1]) or
      ↪ (ponto1[0] < ponto2[0] and ponto1[1] > ponto2[1])
      ↪ or (ponto1[0] > ponto2[0] and ponto1[1] == ponto2
      ↪ [1]) or (ponto1[0] > ponto2[0] and ponto1[1] >
      ↪ ponto2[1]):
        referencia = referencia_translacao(ponto1,ponto2)
        angulo = angulo_rotacao(ponto1,ponto2)
        ponto1t,ponto2t = [translacao(ponto1,referencia),
          ↪ translacao(ponto2,referencia)]
        ponto1r,ponto2r = [rotacao(ponto1t,angulo),rotacao(
          ↪ ponto2t,angulo)]
        equacao = equacao_expansao_triangulo(poligono,ponto1r
          ↪ ,ponto2r,k)
        if tipo == 'x':
            equacao = equacao.subs(x, coordenada1)
            solucao = sp.solve(equacao, y)
            coordenada2 = converter_float(solucao[
              ↪ numero_solucao])
            ponto_curva = (coordenada1,coordenada2)
        elif tipo == 'y':
            equacao = equacao.subs(y, coordenada1)
            solucao = sp.solve(equacao, x)
            coordenada2 = converter_float(solucao[
              ↪ numero_solucao])
            ponto_curva = (coordenada2,coordenada1)
    elif (ponto1[0] == ponto2[0] and ponto1[1] < ponto2[1])
      ↪ or (ponto1[0] < ponto2[0] and ponto1[1] == ponto2
      ↪ [1]) or (ponto1[0] < ponto2[0] and ponto1[1] <
      ↪ ponto2[1]) or (ponto1[0] > ponto2[0] and ponto1[1]
      ↪ < ponto2[1]):
        referencia = referencia_translacao(ponto2,ponto1)
        angulo = angulo_rotacao(ponto2,ponto1)
        ponto1t,ponto2t = [translacao(ponto2,referencia),
          ↪ translacao(ponto1,referencia)]
        ponto1r,ponto2r = [rotacao(ponto2t,angulo),rotacao(
          ↪ ponto1t,angulo)]
        equacao = equacao_expansao_triangulo(poligono,ponto2r
          ↪ ,ponto1r,k)
        if tipo == 'x':
            equacao = equacao.subs(x, coordenada1)
            solucao = sp.solve(equacao, y)
            coordenada2 = converter_float(solucao[
              ↪ numero_solucao])
            ponto_curva = (-coordenada1,coordenada2)
        elif tipo == 'y':
            equacao = equacao.subs(y, coordenada1)
            solucao = sp.solve(equacao, x)
            coordenada2 = converter_float(solucao[
              ↪ numero_solucao])
            ponto_curva = (-coordenada2,coordenada1)
    ponto_curva = rotacao(ponto_curva,angulo,'r')
    ponto_curva = translacao(ponto_curva,referencia,'r')
    return ponto_curva
```

6.3 Function to Trapezium Expansion

After finishing the triangle expansion's codes, the development of the trapezium expansion's codes began. In Function 6.15, we defined Equation (4.3), while Function 6.16 provides the trapezium expansion curve.

Listing 6.15 – Function to definition of Equation (4.3).

```
def equacao_expansao_trapezio(poligono,ponto1,ponto2,k=4):
    i, j = sp.symbols('i␣j')
    P = perimetro(poligono)
    A = area(poligono)
    l = comprimento_lado(ponto1,ponto2)
    return sp.Eq(P-l+2*j+2*sp.sqrt(i**2+(j-l/2)**2)-k*sp.sqrt
      ↪ (A+(l+2*j)*i/2), 0)
```

As in Function 6.13, Function 6.16 rotates and translates the expansion side up to the origin, calculates the curve's points and then performs the reverse processes of rotation and translation, returning lists with the points' x and y coordinates. Since an existence condition to the trapezium expansion curve wasn't found, the function was programmed to stop when i becomes bigger than the expansion side length.

Listing 6.16 – Function to create the trapezium expansion curve.

```
def curva_expansao_trapezio(poligono,ponto1,ponto2,k=4,passo
   ↪ =1):
    i, j = sp.symbols('i␣j')
    dominio = [[[],[]],[[],[]]]
    imagem = [[[],[]],[[],[]]]
    l = comprimento_lado(ponto1,ponto2)
    grafico = list()
    m = 0
    if (ponto1[0] == ponto2[0] and ponto1[1] > ponto2[1]) or
       ↪ (ponto1[0] < ponto2[0] and ponto1[1] > ponto2[1])
       ↪ or (ponto1[0] > ponto2[0] and ponto1[1] == ponto2
       ↪ [1]) or (ponto1[0] > ponto2[0] and ponto1[1] >
       ↪ ponto2[1]):
        referencia = referencia_translacao(ponto1,ponto2)
        angulo = angulo_rotacao(ponto1,ponto2)
        ponto1t,ponto2t = [translacao(ponto1,referencia),
           ↪ translacao(ponto2,referencia)]
        ponto1r,ponto2r = [rotacao(ponto1t,angulo),rotacao(
           ↪ ponto2t,angulo)]
        while True:
            m = round(m,5)
            solucao = sp.solve(equacao_expansao_trapezio(
               ↪ poligono,ponto1r,ponto2r,k).subs(i, m), j)
            if len(solucao) != 0:
                for n in range(0,len(solucao)):
                    if str(round(solucao[n]-converter_float(
                       ↪ solucao[n]),10)) == '0':
                        grafico.append(solucao)
                        solucao[n] = converter_float(solucao[
                           ↪ n])
                        if solucao[n] >= 0:
                            trapezio1,trapezio2 = (m,l/2+
                               ↪ converter_float(solucao[n])
                               ↪ ), (m,l/2-converter_float(
                               ↪ solucao[n]))
                            trapezio1,trapezio2 = [rotacao(
                               ↪ trapezio1,angulo,'r'),
                               ↪ rotacao(trapezio2,angulo,'r
                               ↪ ’)]
                            trapezio1,trapezio2 = [translacao
                               ↪ (trapezio1,referencia,’r’),
                               ↪ translacao(trapezio2,
                               ↪ referencia,’r’)]
                            dominio[n][0].append(trapezio1
                               ↪ [0])
                            dominio[n][1].append(trapezio2
                               ↪ [0])
                            imagem[n][0].append(trapezio1[1])
                            imagem[n][1].append(trapezio2[1])
                    else:
                        if len(grafico) != 0:
                            if dominio[1][0][0] < dominio
                               ↪ [0][0][-1]:
                                return [dominio[1][0]+dominio
                                   ↪ [0][0][::-1],dominio
                                   ↪ [1][1]+dominio
                                   ↪ [0][1][::-1]], [imagem
                                   ↪ [1][0]+imagem
                                   ↪ [0][0][::-1],imagem
                                   ↪ [1][1]+imagem
                                   ↪ [0][1][::-1]]
                            else:
                                return [dominio[0][0][::-1]+
                                   ↪ dominio[1][0],dominio
                                   ↪ [0][1][::-1]+dominio
                                   ↪ [1][1]], [imagem
                                   ↪ [0][0][::-1]+imagem
                                   ↪ [1][0],imagem
                                   ↪ [0][1][::-1]+imagem
                                   ↪ [1][1]]
            if m > l:
                return dominio, imagem
            m += passo
    elif (ponto1[0] == ponto2[0] and ponto1[1] < ponto2[1])
       ↪ or (ponto1[0] < ponto2[0] and ponto1[1] == ponto2
       ↪ [1]) or (ponto1[0] < ponto2[0] and ponto1[1] <
       ↪ ponto2[1]) or (ponto1[0] > ponto2[0] and ponto1[1]
       ↪ < ponto2[1]):
         referencia = referencia_translacao(ponto2,ponto1)
         angulo = angulo_rotacao(ponto2,ponto1)
         ponto1t,ponto2t = [translacao(ponto2,referencia),
            ↪ translacao(ponto1,referencia)]
         ponto1r,ponto2r = [rotacao(ponto2t,angulo),rotacao(
            ↪ ponto1t,angulo)]
         while True:
            m = round(m,5)
            solucao = sp.solve(equacao_expansao_trapezio(
               ↪ poligono,ponto1r,ponto2r,k).subs(i, m), j)
            if len(solucao) != 0:
                for n in range(0,len(solucao)):
                    if str(round(solucao[n]-converter_float(
                       ↪ solucao[n]),10)) == ’0’:
                        grafico.append(solucao)
                        solucao[n] = converter_float(solucao[
                           ↪ n])
                        if solucao[n] >= 0:
                            trapezio1,trapezio2 = (-m,1/2+
                               ↪ converter_float(solucao[n])
                               ↪ ), (-m,1/2-converter_float(
                               ↪ solucao[n]))
                            trapezio1,trapezio2 = [rotacao(
                               ↪ trapezio1,angulo,’r’),
                               ↪ rotacao(trapezio2,angulo,’r
                               ↪ ’)]
                            trapezio1,trapezio2 = [translacao
                               ↪ (trapezio1,referencia,’r’),
                               ↪ translacao(trapezio2,
                               ↪ referencia,’r’)]
                            dominio[n][0].append(trapezio1
                               ↪ [0])
                            dominio[n][1].append(trapezio2
                               ↪ [0])
                            imagem[n][0].append(trapezio1[1])
                            imagem[n][1].append(trapezio2[1])
                    else:
                        if len(grafico) != 0:
```

```
                            if dominio[1][0][0] < dominio
                               ↪ [0][0][-1]:
                                return [dominio[1][0]+dominio
                                   ↪ [0][0][::-1],dominio
                                   ↪ [1][1]+dominio
                                   ↪ [0][1][::-1]], [imagem
                                   ↪ [1][0]+imagem
                                   ↪ [0][0][::-1],imagem
                                   ↪ [1][1]+imagem
                                   ↪ [0][1][::-1]]
                            else:
                                return [dominio[0][0][::-1]+
                                   ↪ dominio[1][0],dominio
                                   ↪ [0][1][::-1]+dominio
                                   ↪ [1][1]], [imagem
                                   ↪ [0][0][::-1]+imagem
                                   ↪ [1][0],imagem
                                   ↪ [0][1][::-1]+imagem
                                   ↪ [1][1]]
            if m > l:
                return dominio, imagem
            m += passo
```

Also due to the curve's processing duration, Function 6.17 was created to return a single point of the trapezium expansion curve.

Listing 6.17 – Function to obtain a single point of the trapezium expansion curve.

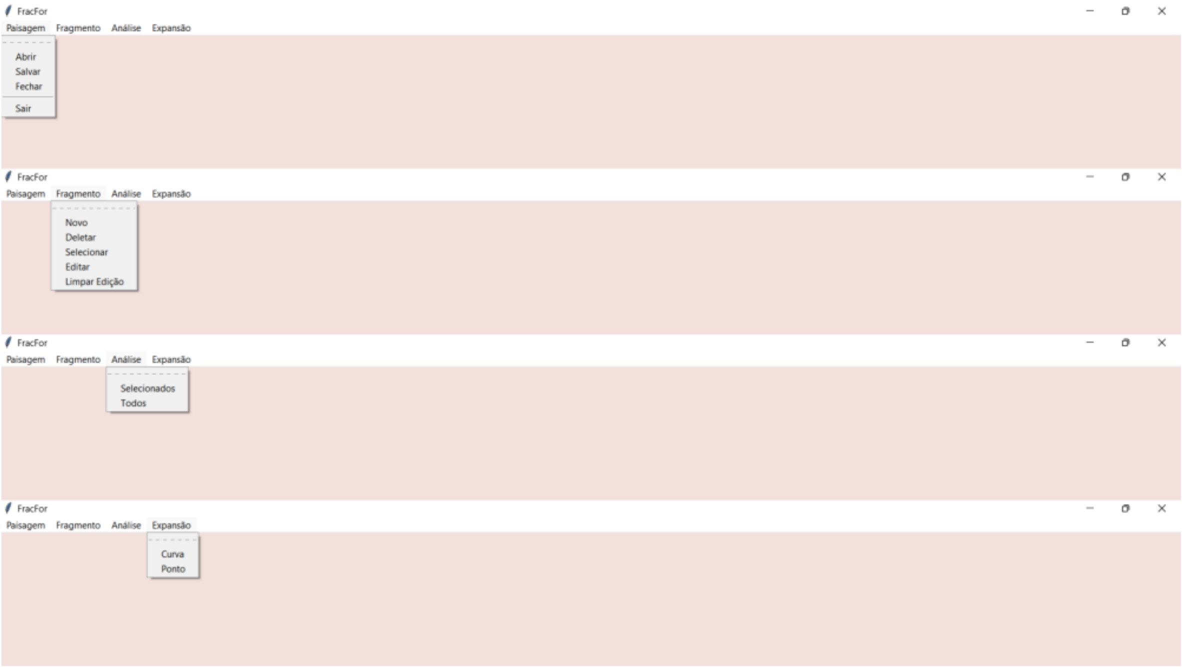
```
def ponto_curva_trapezio(poligono,ponto1,ponto2,altura,k=4):
    i, j = sp.symbols('i␣j')
    l = comprimento_lado(ponto1,ponto2)
    if (ponto1[0] == ponto2[0] and ponto1[1] > ponto2[1]) or
       ↪ (ponto1[0] < ponto2[0] and ponto1[1] > ponto2[1])
       ↪ or (ponto1[0] > ponto2[0] and ponto1[1] == ponto2
       ↪ [1]) or (ponto1[0] > ponto2[0] and ponto1[1] >
       ↪ ponto2[1]):
        referencia = referencia_translacao(ponto1,ponto2)
        angulo = angulo_rotacao(ponto1,ponto2)
        ponto1t,ponto2t = [translacao(ponto1,referencia),
           ↪ translacao(ponto2,referencia)]
        ponto1r,ponto2r = [rotacao(ponto1t,angulo),rotacao(
          ↪ ponto2t,angulo)]
        solucao = sp.solve(equacao_expansao_trapezio(poligono
          ↪ ,ponto1r,ponto2r,k).subs(i, altura), j)
        trapezio1,trapezio2 = (altura,l/2+converter_float(
          ↪ solucao[1])), (altura,l/2-converter_float(
          ↪ solucao[1]))
    elif (ponto1[0] == ponto2[0] and ponto1[1] < ponto2[1])
      ↪ or (ponto1[0] < ponto2[0] and ponto1[1] == ponto2
      ↪ [1]) or (ponto1[0] < ponto2[0] and ponto1[1] <
      ↪ ponto2[1]) or (ponto1[0] > ponto2[0] and ponto1[1]
      ↪ < ponto2[1]):
        referencia = referencia_translacao(ponto2,ponto1)
        angulo = angulo_rotacao(ponto2,ponto1)
        ponto1t,ponto2t = [translacao(ponto2,referencia),
          ↪ translacao(ponto1,referencia)]
        ponto1r,ponto2r = [rotacao(ponto2t,angulo),rotacao(
          ↪ ponto1t,angulo)]
        solucao = sp.solve(equacao_expansao_trapezio(poligono
          ↪ ,ponto2r,ponto1r,k).subs(i, altura), j)
        trapezio1,trapezio2 = (-altura,l/2+converter_float(
          ↪ solucao[1])), (-altura,l/2-converter_float(
          ↪ solucao[1]))
    trapezio1,trapezio2 = [rotacao(trapezio1,angulo,’r’),
      ↪ rotacao(trapezio2,angulo,’r’)]
    trapezio1,trapezio2 = [translacao(trapezio1,referencia,’r
      ↪ ’),translacao(trapezio2,referencia,’r’)]
    return trapezio1,trapezio2
```

6.4 Features

Although it has not yet been completed, the development of the program for the analysis of landscape patches’ images has already begun. The program currently presents four menus: Paisagem, Fragmento, Análise and Expansão, with different features (Figure 26) in operation and the addition of more features is planned as the application continues to be developed.

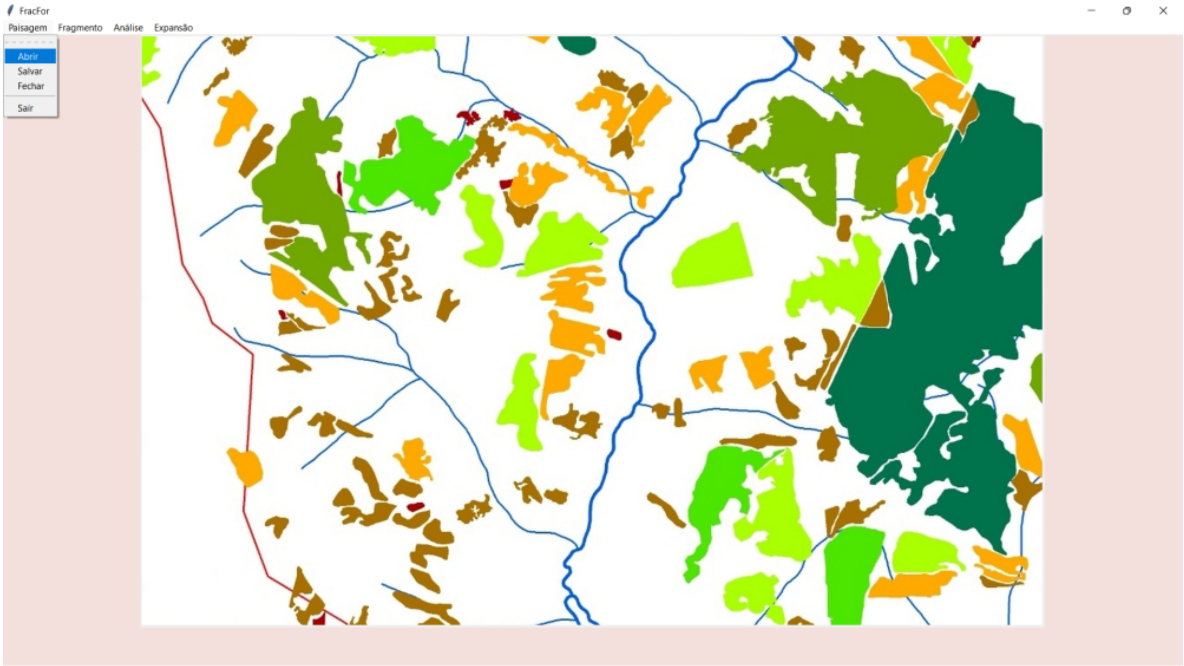
The Paisagem menu features the functions Abrir, Salvar, Fechar and Sair. The function Abrir, an image in the formats .JPG, .PNG or .TIFF, can be loaded to edition (Figure 27). The Salvar function, a sheet with the patch’s area, perimeter, proportionality

Figura 26 – Program interface.



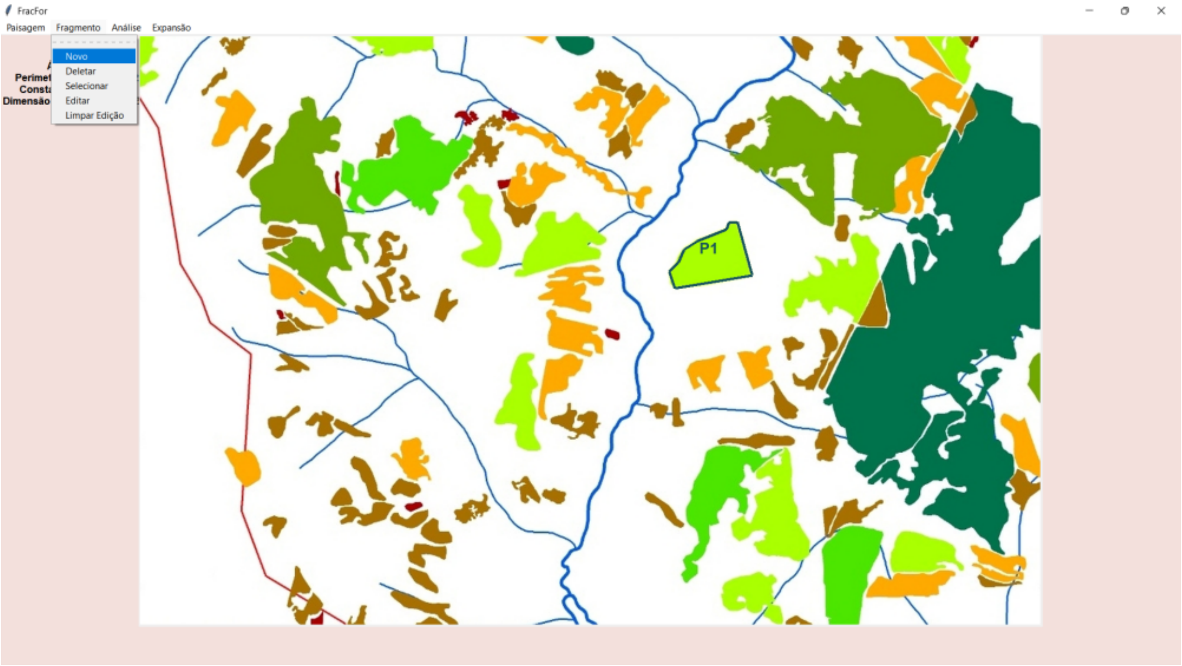
constant and perimeter-area dimension (to k=4) data and an image file, again is possible to choose the formats .JPG, .PNG and .TIFF is saved. The Fechar function ends the work with the opened image, making it possible to start analyzing a new one, while Sair closes the application.

Figura 27 – Image of a landscape loaded via the Abrir function.



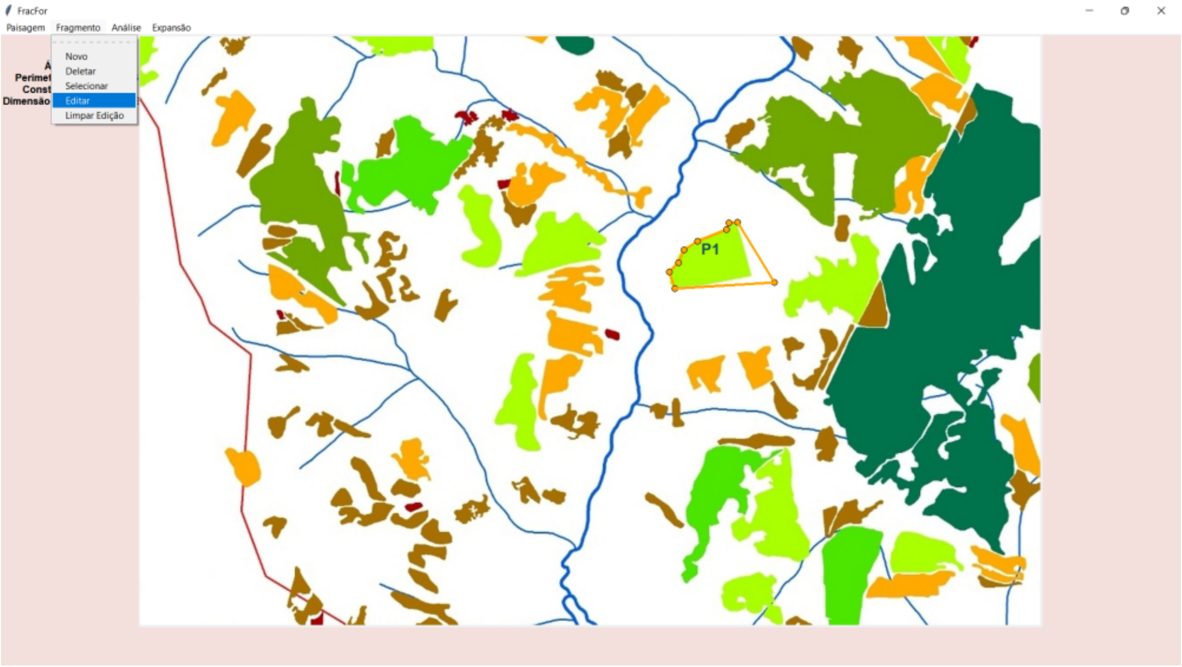
In the Fragmento menu, the Novo function allows the user to outline the patch by defining the polygon's vertices (Figure 28). Immediately after their creation, the polygon's area (in pixels), perimeter (in pixels), proportionality constant and perimeter-

Figura 28 – Polygons created using the Novo function.



area dimension (to k=4) data are shown in the left side of the program. In order to delete any of the outlined polygons, the user selects the Delete function and clicks inside the desired polygon, which also deletes the polygon data.

Figura 29 – Displacement of a vertex done through the Editar function.



Selecting the Editar function and clicking inside a polygon allows the user to change the coordinates of the chosen polygon's vertices. Editing can be performed in three ways: moving a vertex by clicking on it with the mouse's left button and dragging it to a different location (Figure 29), creating a new vertex by clicking with the mouse's right button on

Figura 30 – Creation of a new vertex done through the Editar function.

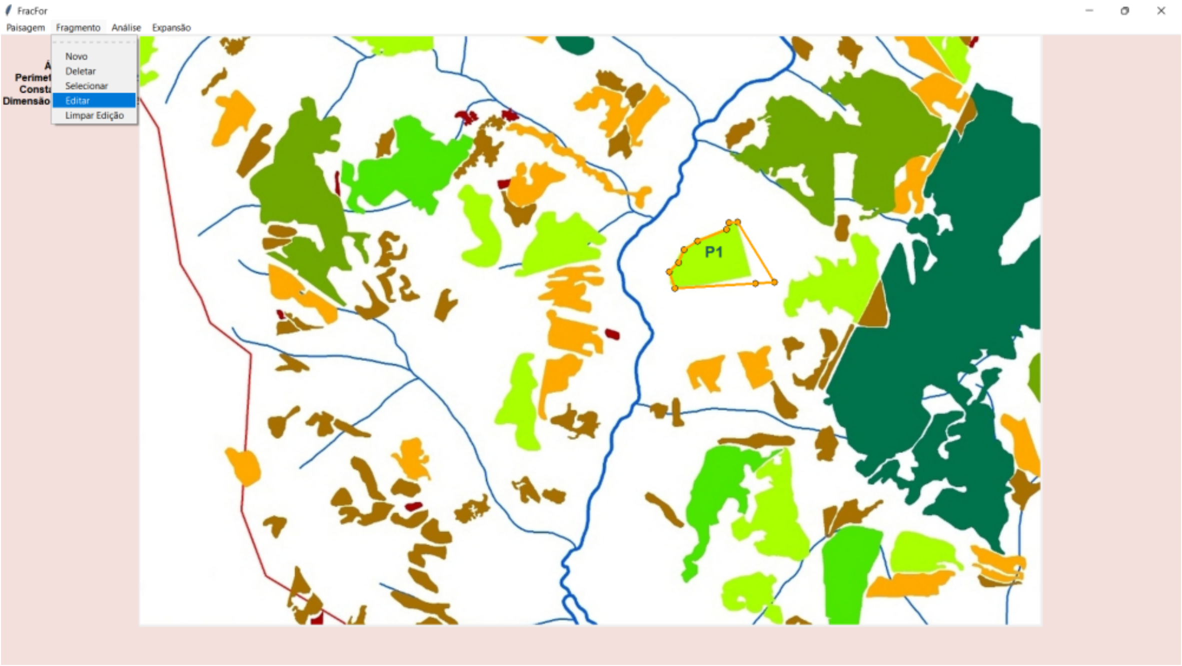
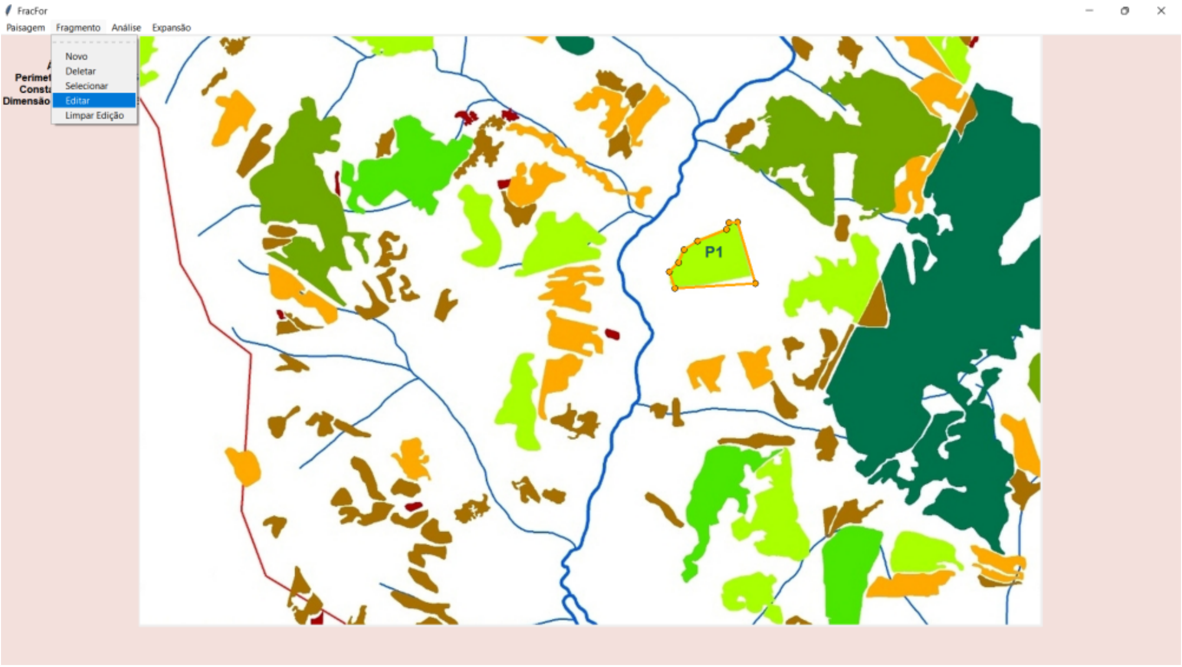


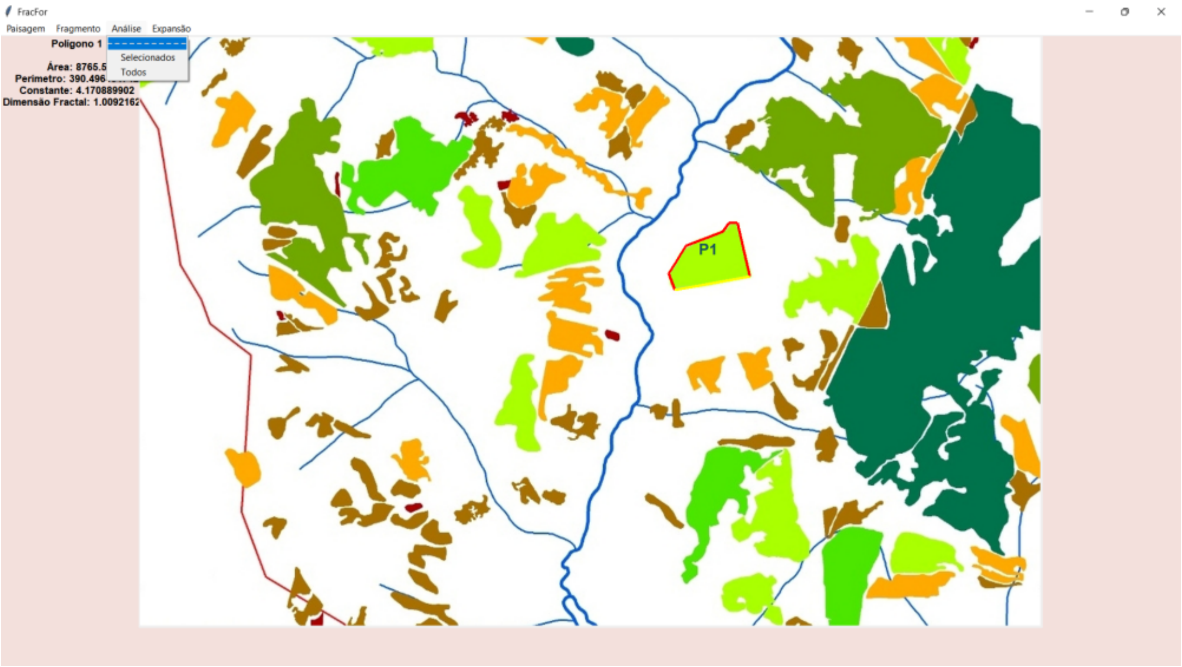
Figura 31 – Removal of a vertex done through the Editar function.



one of the sides (Figure 30) or removing a vertex by clicking on it with the mouse's right button (Figure 31).

The Análise menu contains the Todo function, that analyzes the possibility of triangle expansion of all outlined polygons, and Selecionado, that analyzes only the polygons selected through the Selecionar function in the Fragment menu. In both cases, the sides that cannot be expanded are highlighted in red, while the sides that can be expanded are shown in yellow (Figure 32).

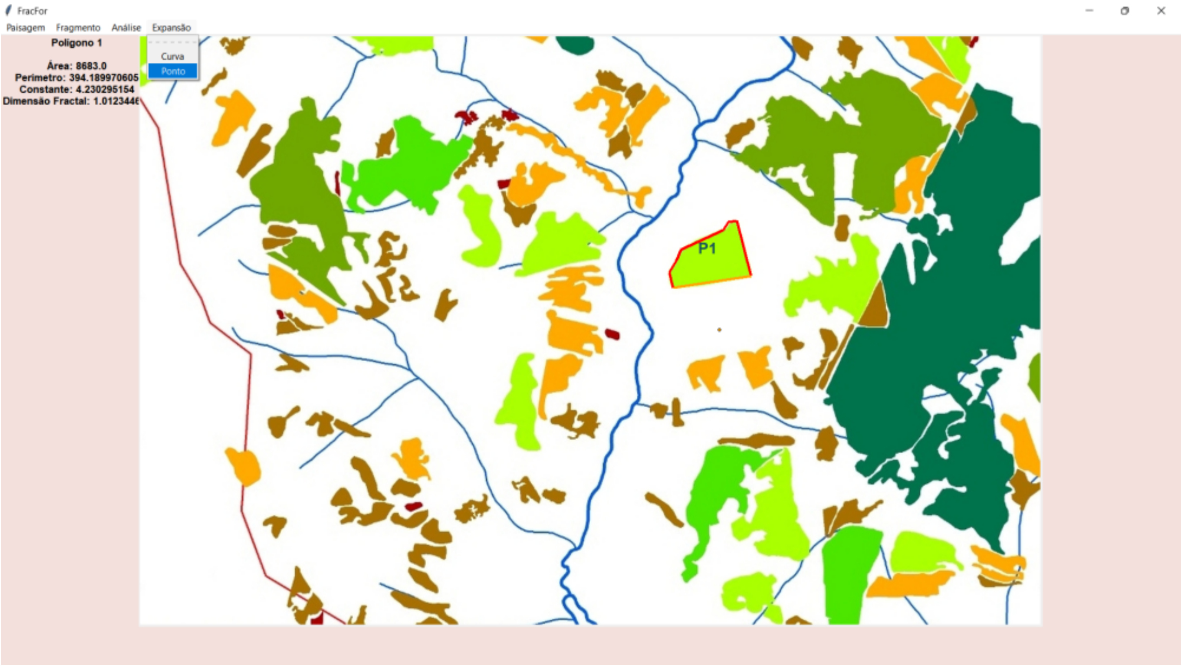
Figura 32 – Result of a polygon's analysis.



Poligono 1

Área: 8765.5
Perimetro: 390.496
Constante: 4.170889902
Dimensão Fractal: 1.0092162

Figura 33 – Point on the expansion curve that would expand the analyzed polygon by the addition of an isosceles triangle.

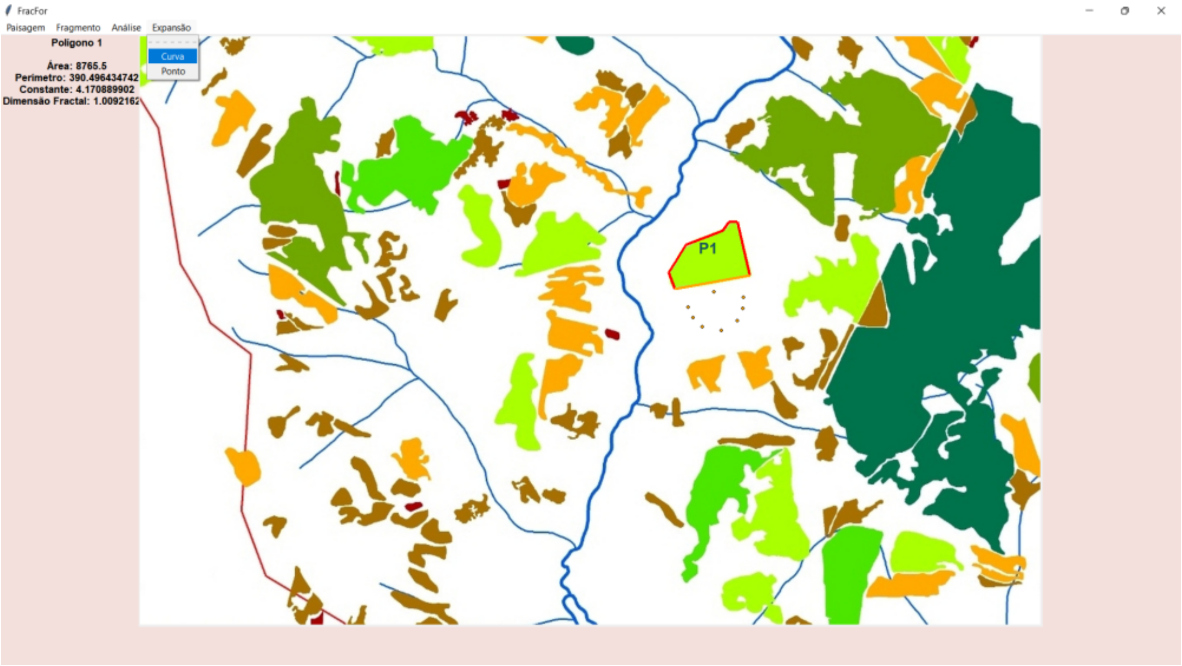


Poligono 1

Área: 8683.0
Perimetro: 394.189970605
Constante: 4.230295154
Dimensão Fractal: 1.012344

Finally, the Expansão menu contains the Curva and Ponto functions. The Ponto function shows only a single point of the expansion curve (Figure 33), while the Curva function shows several points of the expansion curve (Figure 34). Both functions are activated by clicking on the function and then on the expansion side.

Figura 34 – Expansion curve of selected side.



Poligono 1

Área: 8765.5
Perimetro: 390.496434742
Constante: 4.170889902
Dimensão Fractal: 1.0092162

6.4.1 Analysis of Forest Patches from the Catu River Basin

For the patches' analysis, 3 sub-landscapes of Figure 1 were chosen and, using FracFor, we delimited 5 patches of each sub-landscape (Figures 35, 36 and 37). Tables 11, 12 and 13 show area, perimeter and perimeter-area dimension data for each sub-landscape.

Figura 35 – Analysis of fragments in sub-landscape A.

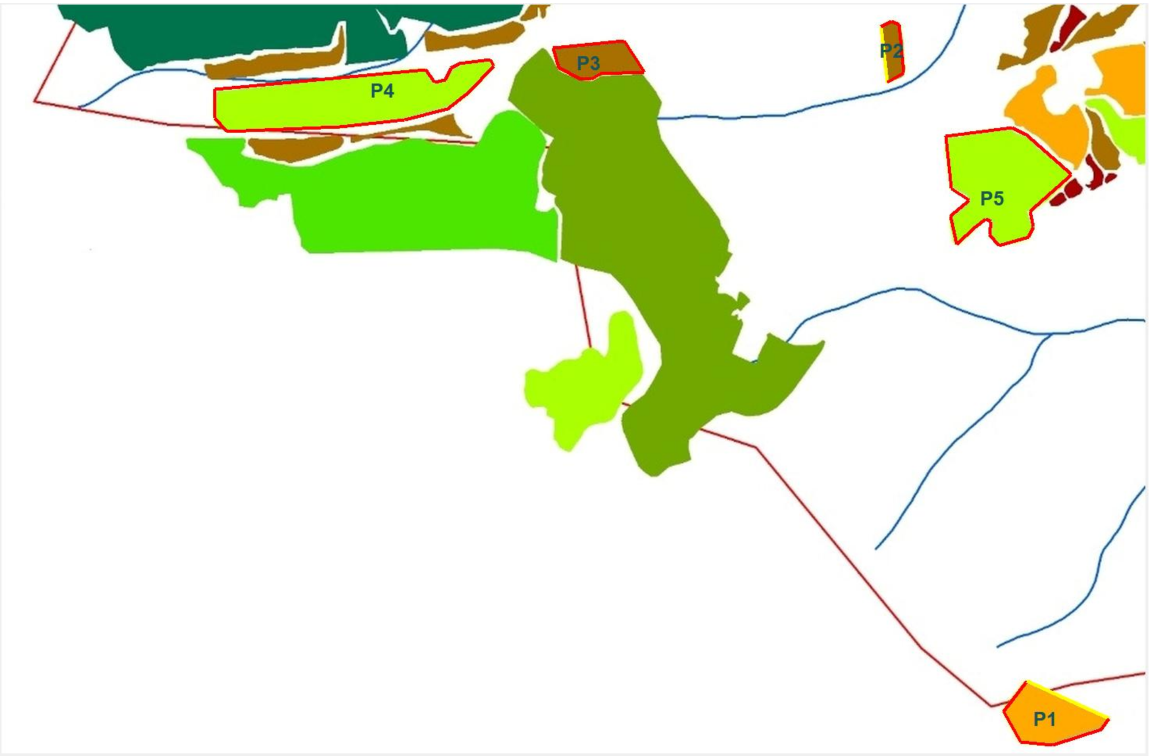


Figura 36 – Analysis of fragments in sub-landscape B.

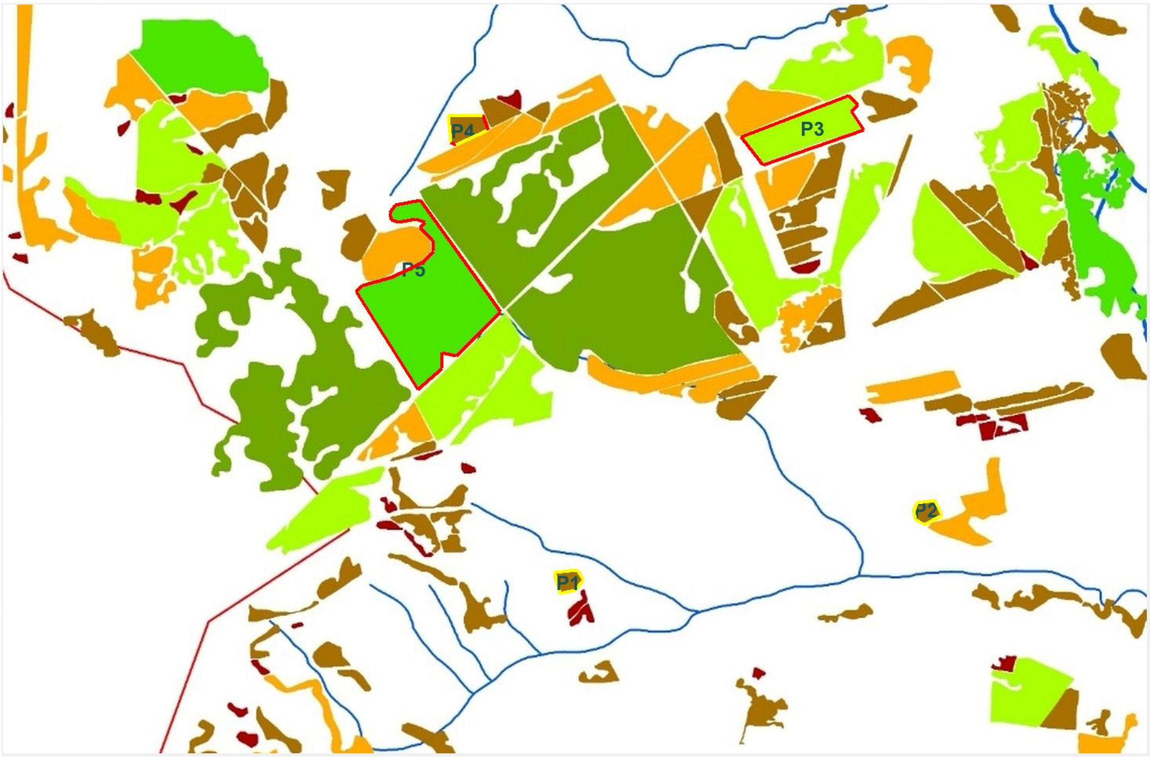
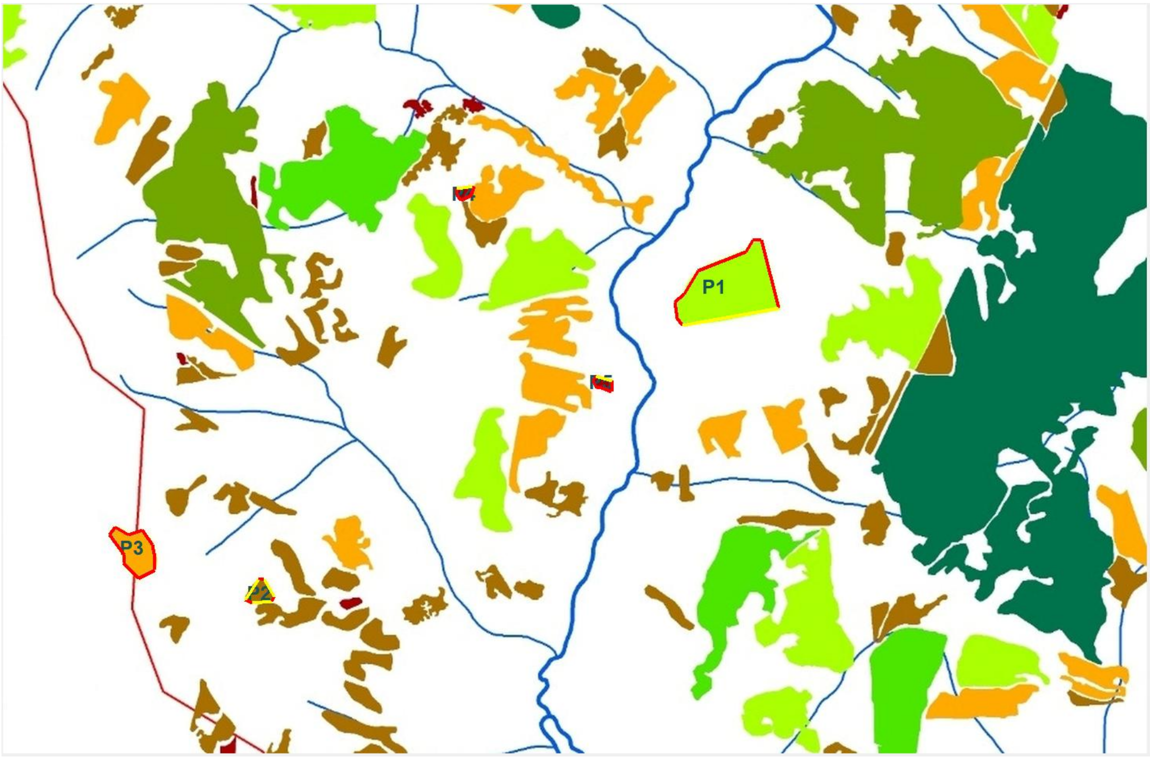


Figura 37 – Analysis of fragments in sub-landscape C.



As previously mentioned, the program returns area and perimeter values in pixels and not in the usual units of length and area, however the conversion of units can be done by calculating for each image, how many pixels correspond to each unit of measurement, as noted in Tables 7, 8 and 9.

Tabela 11 – Table with the values of area (A), perimeter (P) and fractal dimension (D_F) of the patches in sub-landscape A (Figure 35).

Fragmento	A (px)	P (px)	D_F
P_1	6476	332,97747	1,00771
P_2	1576,5	183,46085	1,03918
P_3	3911	280,80583	1,02795
P_4	19153	809,14966	1,07699
P_5	15790,5	583,25254	1,03077

Tabela 12 – Table with the values of area (A), perimeter (P) and fractal dimension (D_F) of the patches in sub-landscape B (Figure 36).

Fragmento	A (px)	P (px)	D_F
P_1	766,5	108,75762	0,99455
P_2	764	105,07373	0,98466
P_3	6469,5	387,91476	1,04264
P_4	1389,5	151,19618	1,00385
P_5	19412,5	742,16348	1,05802

Tabela 13 – Table with the values of area (A), perimeter (P) and fractal dimension (D_F) of the patches in sub-landscape C (Figure 37).

Fragmento	A (px)	P (px)	D_F
P_1	8522,5	391,95603	1,01318
P_2	688	105,25525	1,00098
P_3	2214,5	199,73284	1,01540
P_4	258,5	65,77286	1,00809
P_5	305,5	73,38273	1,01692

In sub-landscape A, the patches to admit expansion were patches P_1 and P_2, in sub-landscape B, patches P_1, P_2 and P_4 admitted expansion, while sub-landscape C, patches P_1, P_2, P_4 and P_5 could be expanded. As highlighted earlier, in the same patch there may be some sides that admit expansion while others do not. Among the presented patches, only patches P_1 and P_2 of sub-landscape B allowed expansion on all sides, this is due to the fact that this is the only fragment with a perimeter-area dimension smaller than 1.

7 Papers

As a result of this research, the article Interpretação Geométrica da Constante de Proporcionalidade da Dimensão Perímetro-Área, was presented in the I Simpósio Baiano de Modelagem e Simulação de Biossistemas. The article is a summary of the idea shown in Chapter 5, it describes the role of the proportionality constant of the perimeter-area dimension, and how the choice of the constant can make the dimension invariant by homothety.

Currently, the article Fractal Stability Applied to Forestry Patches, that summarizes Chapter 3, presenting the triangle expansion, has ended and is being formatted in order to be published in a journal.

In addition to these, it's planned to publish articles showing the analysis of the fractal dimension with a modified constant in the morphology of different organisms, such as leaves of plants germinated at different levels of hydration. Additionally, the continuous development of the FracFor program is planned and ew formats such as shp (shapefile) might be accepted.

8 Conclusion

This thesis was formulated with the objective of defining a method for planning of recovery of forestry patches so that they had the desired perimeter-area fractal dimension. Due to the hypothesis that the perimeter-area dimension could be used in the development of the method, the study of applications of fractal geometry in landscape ecology focused on this dimension, which proved to be an appropriate tool for the elaboration of the method.

During the analysis of the perimeter-area dimension, it was observed that the constant of proportionality influenced the alteration of the dimension with the variation in the size of the fragments, which led to a deeper study of the role played by the constant in the dimension. During the research, it was observed that changes in the constant made the dimension invariable for fragments that had the same shape, but different sizes, leading to the definition of a dimension with a modified constant for analysis of changes in the shape of objects in nature.

With the study of the constant, it was also concluded that forms quite different from the square could have a perimeter-area dimension equal to 1 for a constant equal to 4, which led to the elaboration of the expansion method, which consists of finding polygons that have a format close to that of the fragment and fractal dimension equal to 1.

Once it was verified that the triangle expansion method was not applicable for any landscape fragment, expansion methods with the addition of different figures, such as the isosceles trapezoid, were thought. Although it can be applied to a larger number of fragments, the trapeze expansion has its own limitations, such as not having an existence condition, which raises the investigation of expansion by new forms.

Although normally landscape ecologists do not want landscape fragments to present perimeter-area dimensions with values very close to 1, since this is an indication of anthropic action, the expansion method was proposed for the recovery of fragments already degraded by human action and since patches with dimension close to 1, for $k = 4$, have a good perimeter-area ratio, which is desirable in comparison to fragments that present recesses or elongated rectangular shapes in which the edge effect is more accentuated.

Referências

ALLAIN, C.; CLOITRE, M. Characterizing the lacunarity of random and deterministic fractal sets. *Phys. Rev. A*, American Physical Society, v. 44, p. 3552–3558, Sep 1991. Citado 2 vezes nas páginas 9 e 13.

BARNSLEY, M. F. et al. *The Science of Fractal Images*. [S.l.]: Springer, 1988. Citado 2 vezes nas páginas 9 e 12.

BROADBENT, E. N. et al. Forest fragmentation and edge effects from deforestation and selective logging in the brazilian amazon. *Biological Conservation*, v. 141, n. 7, p. 1745–1757, 2008. ISSN 0006-3207. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0006320708001377>. Citado na página 9.

CHEN, Y. Two sets of simple formulae to estimating fractal dimension of irregular boundaries. *Mathematical Problems in Engineering*, v. 2020, 2020. Citado 3 vezes nas páginas 16, 17 e 38.

FALCONER, K. *Fractal Geometry: Mathematical Foundations and Applications*. 3. ed. [S.l.]: John Wiley & Sons, Ltd, 2014. Citado 4 vezes nas páginas 9, 11, 12 e 13.

FRASER, J. M. *Assouad Dimension and Fractal Geometry*. [S.l.]: Cambridge University Press, 2020. (Cambridge Tracts in Mathematics). Citado 3 vezes nas páginas 11, 12 e 15.

GUSTAFSON, E. J. How has the state-of-the-art for quantification of landscape pattern advanced in the twenty-first century? *Landscape Ecol*, v. 34, p. 2065–2072, 2019. Citado 2 vezes nas páginas 14 e 16.

HALLEY, J. M. et al. Uses and abuses of fractal methodology in ecology. *Ecology Letters*, v. 7, n. 3, p. 254–271, 2004. Citado 5 vezes nas páginas 12, 13, 14, 15 e 16.

KENKEL, N. C.; WALKER, D. J. Fractals in the biological sciences. *Coenoses*, Akadémiai Kiadó, v. 11, n. 2, p. 77–100, 1996. ISSN 03939154. Citado na página 16.

KRUMMEL, J. R. et al. Landscape patterns in a disturbed environment. *Oikos*, [Nordic Society Oikos, Wiley], v. 48, n. 3, p. 321–324, 1987. ISSN 00301299, 16000706. Citado 5 vezes nas páginas 15, 16, 20, 31 e 32.

LOPEZ, R.; FROHN, R. *Remote Sensing for Landscape Ecology: Monitoring, Modeling, and Assessment of Ecosystems*. 2. ed. [S.l.]: CRC Press, 2017. Citado 6 vezes nas páginas 9, 14, 15, 16, 31 e 32.

LOVEJOY, S. Area-perimeter relation for rain and cloud areas. *Science*, v. 216, n. 4542, p. 185–187, 1982. Citado na página 15.

MANDELBROT, B. B. *The Fractal Geometry of Nature*. [S.l.]: H. B. Fenn and Company Ltd., 1982. Citado 3 vezes nas páginas 11, 12 e 14.

METZGER, J. P. O que é ecologia de paisagens? *Biota Neotrop.*, 2001. Citado na página 13.

MILNE, B. T. Measuring the fractal geometry of landscapes. *Applied Mathematics and Computation*, v. 27, n. 1, p. 67–79, 1988. ISSN 0096-3003. Citado 2 vezes nas páginas 13 e 15.

MONTELLO, D. R. Scale in geography. In: WRIGHT, J. D. (Ed.). *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*. Second edition. Oxford: Elsevier, 2015. p. 1–3. ISBN 978-0-08-097087-5. Citado 2 vezes nas páginas 13 e 14.

MURCIA, C. Edge effects in fragmented forests: implications for conservation. *Trends in Ecology & Evolution*, v. 10, n. 2, p. 58–62, 1995. ISSN 0169-5347. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169534700889776>. Citado na página 9.

NEWMAN, E. A. et al. Scaling and complexity in landscape ecology. *Front. Ecol. Evol.*, 2019. Citado na página 12.

OLSEN, E. R.; RAMSEY, R. D.; WINN, D. S. A modified fractal dimension as a measure of landscape diversity. *Photogrammetric Engineering & Remote Sensing*, v. 59, n. 10, p. 1517 – 1520, 1993. Cited by: 71. Citado 2 vezes nas páginas 16 e 17.

O'NEILL, R. V. et al. Indices of landscape pattern. *Landscape Ecol*, v. 1, p. 153–162, 1988. Citado na página 16.

RICOTTA, C. et al. A generalizaed non-regression technqiue for evaluating the fractal dimension of raster gis layers consisting of non-square cells. *Coenoses*, Akadémiai Kiadó, v. 12, n. 1, p. 23–26, 1997. ISSN 03939154. Citado 2 vezes nas páginas 16 e 17.

ROSENBERG, E. *Fractal Dimensions of Networks*. [S.l.]: Springer, 2020. Citado 4 vezes nas páginas 9, 11, 12 e 14.

SEURONT, L. *Fractals and multifractals in ecology and aquatic science*. [S.l.]: Taylor and Francis Group, LLC, 2010. Citado 4 vezes nas páginas 11, 12, 14 e 15.

SUGIHARA, G.; MAY, R. M. Applications of fractals in ecology. *Trends in Ecology & Evolution*, v. 5, n. 3, p. 79–86, 1990. ISSN 0169-5347. Citado na página 13.

TRIPATHI, S. K. et al. Measuring ecosystem patterns and processes through fractals. *Current Science*, Current Science Association, v. 109, n. 8, p. 1418–1426, 2015. Citado 3 vezes nas páginas 16, 21 e 42.

TURNER, M. G. Landscape ecology: The effect of pattern on process. *Annual Review of Ecology and Systematics*, Annual Reviews, v. 20, p. 171–197, 1989. ISSN 00664162. Citado na página 13.

TURNER, M. G. Spatial and temporal analysis of landscape patterns. *Landscape Ecol*, v. 4, p. 21–30, 1990. Citado na página 16.

TURNER, M. G.; GARDNER, R. H. *Landscape Ecology in Theory and Practice: Pattern and Process*. 2. ed. [S.l.]: Springer, 2015. Citado 2 vezes nas páginas 9 e 13.

TURNER, M. G.; RUSCHER, C. L. Changes in landscape patterns in georgia, usa. *Landscape Ecol*, v. 1, p. 241–251, 1988. Citado na página 16.

VAZ, G.; MATOS, M. Mapeamento e avaliação ecológica de Áreas de preservação permanente do rio catu, alagoinhas, bahia, brasil. *SITIENTIBUS Série Ciências Biológicas*, v. 14, 2014. Citado na página 10.

VRANKEN, I. et al. A review on the use of entropy in landscape ecology: heterogeneity, unpredictability, scale dependence and their links with thermodynamics. *Landscape Ecol*, v. 30, p. 51–65, 2015. Citado na página 16.

WU, J. Effects of changing scale on landscape pattern analysis: scaling relations. *Landscape Ecol*, v. 19, p. 125–138, 2004. Citado na página 13.

YU, H. et al. Landscape ecology development supported by geospatial technologies: A review. *Ecological Informatics*, v. 51, p. 185–192, 2019. ISSN 1574-9541. Citado 2 vezes nas páginas 9 e 15.