XLI CILAMCE
XLI Ibero-Latin American Congress on Computational Methods in Engineering
16-19 November, 2020 FOZ DO IGUAÇU PARANÁ - BRAZIL

Mesh generation algorithm involving irregularly contoured regions for numerical simulations of partial differential equations by finite differences

Pedro Zaffalon da Silva[1], Neyva Maria Lopes Romeiro[2], Rafael Furlanetto Casamaximo[1], Iury Pereira de Souza[1], Paulo Laerte Natti[2], Eliandro Rodrigues Cirilo[2]

[1]*Dept. of Computer Science, State University of Londrina*
Rodovia Celso Garcia Cid — PR 445 Km 380, 86.057-970, Paraná/Londrina, Brasil
pedro.zaffalon@uel.br, rafael.furlanetto@uel.br, iury.pereira.souza@uel.br
[2]*Dept. of Math, State University of Londrina*
Rodovia Celso Garcia Cid — PR 445 Km 380, 86.057-970, Paraná/Londrina, Brasil
nromeiro@uel.br, plnatti@uel.br, ercirilo@uel.br

Abstract. This work proposes a mesh generation algorithm, involving irregular contours. In this algorithm, the contour of the physical domain is approximated by mesh segments, allowing numerical simulations of partial differential equations using the finite difference method. Hence, the first code analyzed each node in the mesh, using two loops. However, this algorithm was impracticable for a mesh with many nodes, as a consequence of the high number of operations, which let the algorithm slow. As a possibility to reduce the computational cost, the algorithm was adapted to use only one repetition structure. In order to reduce numerical calculations, the new algorithm calculates the slope of the line defined by a known point of the irregular contours and the neighboring vertices. Then the algorithm calculates the line points and the shortest distance from these points to a mesh node, thus generating a point of the approximate contour. This process is repeated until the approximate contour is obtained. Algorithm results are presented using four geometries. In these figures, the approximated contour properly describes the irregular given contour.

Keywords: Irregular geometry, contour, mesh, finite differences.

1 Introduction

Modeling and simulation, through the manipulation of differential equations, is an important tool for the analysis and mathematical description of several phenomena. However, due to the fact that the vast majority of differential equations have no analytical solution, it is necessary to apply numerical methods to solve them. To apply numerical methods, it is necessary to know information about the geometry of the medium being investigated, considering the computational mesh.

The computational mesh consists of the discretized representation of the physical domain described through a given contour. Thus, the mesh is formed by a set of cells, bounded by the edges, which are called faces, containing vertices, which are called nodes. However, in the natural phenomena modeling, the domain where the boundary conditions of the problem are defined is rarely found under the nodes of the computational mesh [1].

Thus, cartesian meshes in a two-dimensional plane, face serious difficulties when prescribing boundary conditions in a non-regular domain, making it difficult to solve the problem considering the finite difference method [2]. However, discretization using cartesian meshes is attractive in terms of efficiency and low memory storage [3].

Within this context, many authors apply methods that use algebraic polynomial interpolations to construct the equations of differences at the given contour points, making it possible to incorporate the irregular contour into the method, that is, all calculations on irregular domains are reduced to regular domains, therefore obtaining a more precise numerical solution to the problem [2, 4–6].

Thus, this work presents the development of an algorithm, using Octave software, which adequately describes the contour of an irregular region through a finite set of points, resulting in a mapped domain. Through the algorithm, it becomes possible to use the finite difference method to numerically solve partial differential equations using meshes that contain irregular contours. The used procedure employs a technique of representing lines defined by the given contour points, to generate the approximate contour points closest to the mesh points.

This work is described as follows: initially, it is defined how the coordinates of the given contour points are ordered. Then, two cases are presented where the algorithm checks if it is necessary to exclude external nodes in convexities or to add nodes in concavities. After this development, an algorithm resume is presented where it shows how to obtain the approximate contour. As a test, some meshes generated by the algorithm are displayed in the results. Finally, the conclusions are presented.

2 Development

Once the given contour inscribed in a rectangular domain region A=[X_0, X_f]x[Y_0, Y_f], define $\delta_x = (X_f - X_0)/N_x$ and $\delta_y = (Y_f - Y_0)/N_y$, where N_x and N_y are the partitions in x and y, respectively. From these values, the mesh in which the irregular domain will be represented is defined, using the approximate contour.

First, to obtain the approximate contour, an algorithm was considered that used all the nodes of the mesh, as to reproduce the contour of the irregular region, verifying which points under the mesh nodes belonged to the approximate contour, however, due to the high-cost computational approach, a more optimized algorithm was chosen. Therefore, a second algorithm was developed, which started to evaluate only the given contour points, thus reducing the number of operations performed. The final algorithm uses linear functions, which represent lines defined by the given contour points, to generate the approximate contour closer to the nodes of the mesh.

This way, the algorithm receives as input parameters the vector with coordinates x and y of the given contour, the minimum values of the vector coordinates, represented by (x_{min}, y_{min}), and the mesh spacing, that is δ_x and δ_y. Also, the direction in which the contour points are ordered is defined.

Thus, the first point of the approximate contour, or as denoted, the first node, will be used as the basis for calculating the other nodes. This node will be obtained by approaching the first point of the given contour to a closer node, internal to the region of the approximate contour area, under a mesh point. However, for this, it is necessary to check in which direction the internal region, the area of interest, is. To check this region, one must observe the direction in which the coordinates of the contour points are ordered, whether it is clockwise or counterclockwise. To define how the contour is arranged, use the Figures 1a)-d).

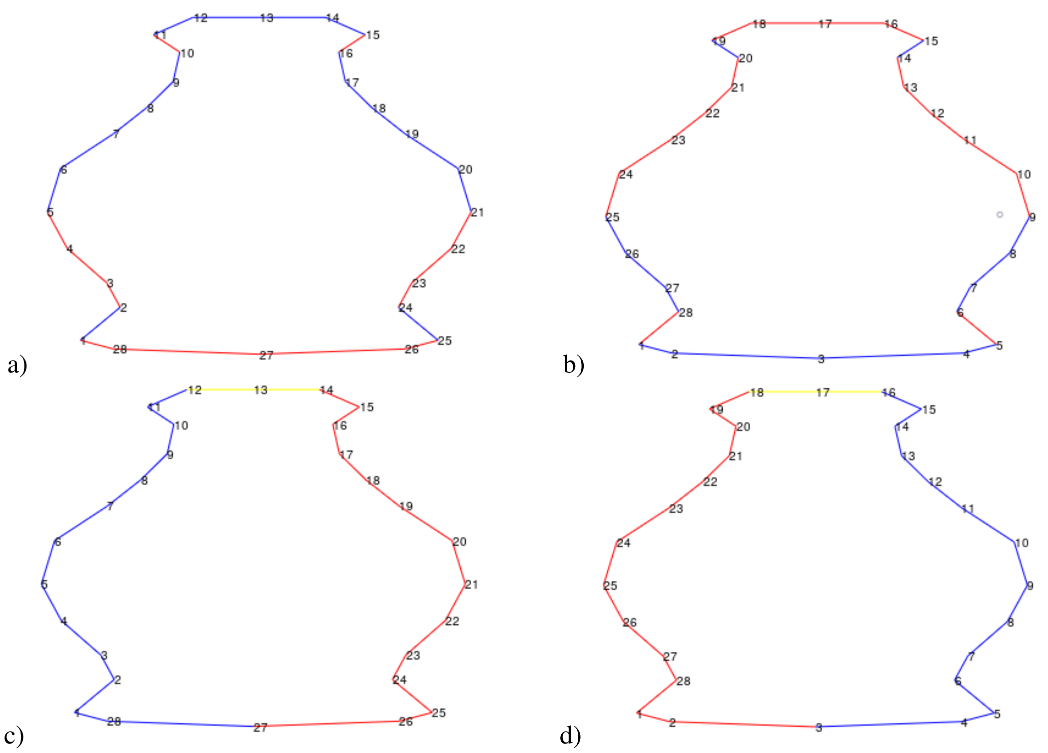

Figure 1. The contour points are arranged in the following directions: a) and c) clockwise; b) and d) counterclockwise.

In Figure 1a), the coordinates are arranged in a clockwise direction, where it can be seen in each part of the blue line segments the subsequent point has a higher value for the coordinate x, and the internal area of the figure is located below the straight line segments. On the other hand, in the red line segments, the subsequent point has a lower value for the coordinate x, and the internal area is above the straight segments.

In Figure 1b), the coordinates are arranged in a counterclockwise. It is observed that in each part of the blue line segments, the subsequent point has a higher value for the coordinate x, and the internal area of the figure is above the straight segments. But, in the red line segments, the subsequent point has a lower value for the coordinate x, and the internal area of the figure is located below the straight line segments.

In Figure 1c), the coordinates are arranged in a clockwise direction. It is observed that in each part of the blue line segments, the subsequent point has a higher value for the coordinate y, and the internal area of the figure is located on the right of the point. On the other hand, in the red line segments, the subsequent point has a lower value for the coordinate y, and the internal area of the figure is located on the left. Still, in this figure, can be seen line segments in yellow, where there is no variation in the value of y, in this case the direction will be the same as the previous point.

In Figure 1d), the coordinates are arranged in a counterclockwise. it is observed that in the blue line segments, the subsequent point has a higher value for y, and the internal area of the figure is located to the left of the point. On the other hand, in the red line segments, the subsequent point has a lower value for de coordinate y, and the internal area of the figure is located on the right. Similarly to Figure 1c), in the yellow line segments, there is no variation in the value of y, in this case the direction will be the same as the previous point.

Once the direction in which the internal area of the region is located is verified, the algorithm approaches the first point of the contour given to the nearest internal node. After obtaining the initial node, the algorithm calculates the coordinates of the approximate nodes up to the second point of the given contour, as shown in the Figures 2 a)-d).

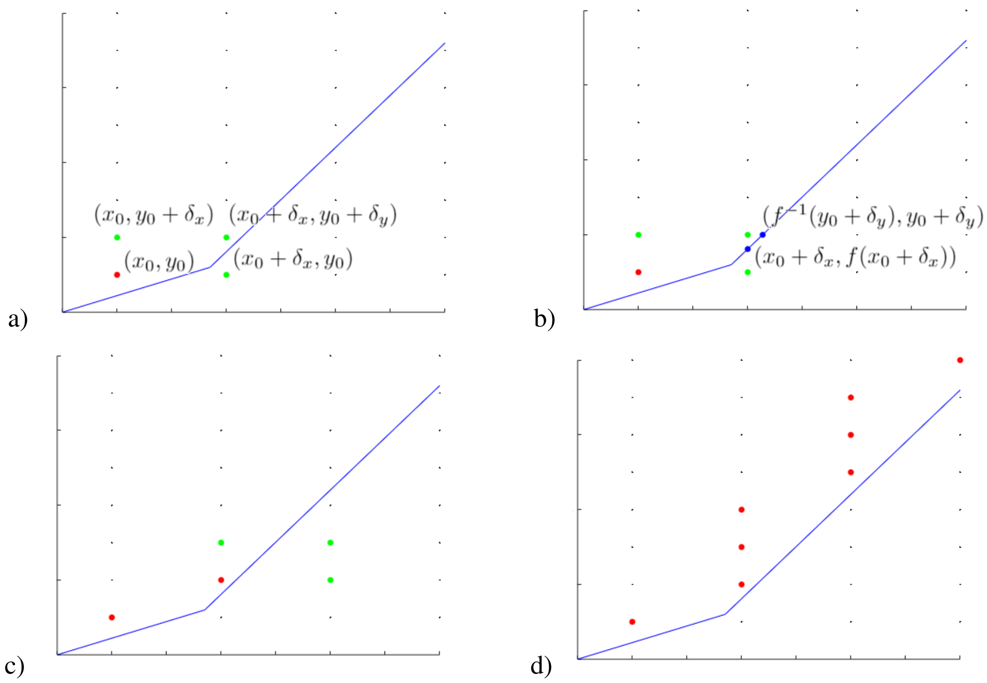

Figure 2. Steps of the algorithm to obtain the nodes between two points.

In Figure 2a), the red point (x_0, y_0), represents the first point of the approximate contour, while the blue lines describe the contour given in counterclockwise.

Since the line is increasing in x and in y, the next node of the approximate contour must have coordinates greater or equal, both in x and in y. So there are three possibilities for the next node, being $(x_0, y_0+\delta_y)$, $(x_0+\delta_x, y_0)$ and $(x_0 + \delta_x, y_0 + \delta_y)$, as can be seen in green at Figure 2a).

In order to identify which node should be added to the approximate contour, it is necessary to find the distance of each point in relation to the given contour. For this, using the function $f(x)$ which represents the line defined by the current point of the given contour and its predecessor. Once the function is obtained, the values of $f(x_0 + \delta_x)$ and $g(y_0 + \delta_y)$, being $g = f^{-1}$ are calculated, which allows to obtain the distance to the analyzed nodes.

To identify which node should be added, to point on the line are analyzed, one where X = $x_0 + \delta_x$, and the other where Y = $y_0+\delta_y$. Thus, obtaining $(x_0+\delta_x, f(x_0+\delta_x))$ and $(f^{-1}(y_0+\delta_y), y_0+\delta_y)$, as shown in Figure 2b). Notice that the point that should be added in the approximate contour is $(x_0 + \delta_x, y_0 + \delta_y)$, because the distance between the line and the point $(x_0, y_0 + \delta_y)$ is bigger than δ_x, and $(x_0 + \delta_x, y_0)$ isn't a internal point in the figure.

Once the second node is obtained, the process continues, performing the same operations, as shown in Figure 2c). This step will be interrupted when reaching the closest node to the current point of the given contour, as shown in Figure 2d). Once the step for the current point is finished, the same procedure will be performed for the other points of the given contour, in order that, when passing through all points, the approximate contour is obtained.

2.1 Exceptions: convexities and concavities

In some cases, in which the direction of x or y changes in relation to the previous line, it is necessary to do some verification to generate the correct approximate contour. The two cases will be described next.

Case 1: In convexities the obtained nodes can be external in relation to the next straight line, as shown in the Figure 3.

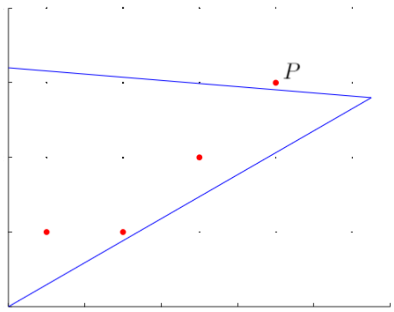

Figure 3. Example of an external node in relation to the next line

In Figure 3 the point P it is the closest node above the first straight line of the contour, but in relation to the next line, it is external to the figure. In order to avoid this situation, a verification is performed to exclude external nodes in convexities.

Case 2: In concavities, the last obtained node may not be the starting node of the next line, as shown in the Figures 4a) and 4b).

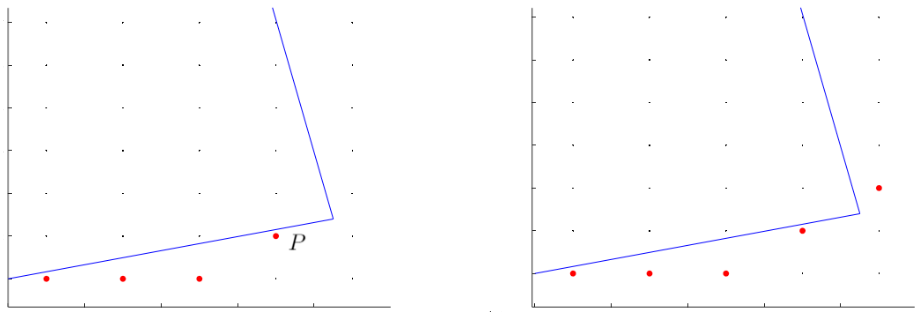

Figure 4. Example of a case where it is necessary to add an additional node

In Figure 4a) the point P it is the last node of the first straight line of the contour and, consequently, it is the starting point for the next line. However, considering that the line has increasing value in y and decreasing in x, the next obtained node would be external to the figure. This occurs because the last obtained node is not suitable as a starting value for the next straight line. As an alternative to avoid incorrect nodes, a new node is added to the contour, in this case, performing the same operation, but verifying in relation to the next line, obtaining the appropriate initial node, as shown in Figure 4b).

3 Algorithm

All the progress described on the algorithm is presented in a summary form at Algorithm 1, which has the vector with the coordinates x and y as input of the given contour, the minimum values of the coordinates of this vector, represented by (x_{min}, y_{min}), and the mesh spacing, that is δ_x and δ_y. Still n_x and n_y describes the number of nodes between a given contour point and its successor. The parameters v_x and v_y represent the difference between two points of the given contour.

Algorithm 1: Algorithm to obtain the given contour

Input: given contour vector, x_{min}, y_{min}, δ_x, δ_y
Output: approximated contour

```
begin
    Approach the first coordinate to the nearest node to the mesh;
    Add the obtained node to the approximate contour;
    for i ← 2; i < size of the given contour; i++ do
        Calculate the difference between the current point and the previous point of the given contour, both for x (v_x) as y (v_y);
        Calculate the number of nodes between the last node and the current point of the given contour, both for x (n_x) as y (n_y);
        if n_x >= 1 or n_y >= 1 then
            if v_x == 0 then
                for j ← 1; j < n_y; j++ do
                    Add a new node in the approximate contour, with the same coordinate as the previous one added with δ_y;
                end
            end
            else if v_y == 0 then
                for j ← 1; j < n_x; j++ do
                    Add a new node in the approximate contour, with the same coordinate as the previous one added with δ_x;
                end
            end
            else
                Calculate the slope of the line defined by the current and the previous point (v_y/v_x);
                while Difference between the last obtained node and the current point in x >= δx or difference in y >= δ_y do
                    Calculate the possible three next nodes;
                    Verify which internal node is closest to the line;
                    Add the obtained node to the approximate contour;
                end
            end
            if Region has convexity then
                while The last node is external to the figure do
                    Remove the last node;
                end
            end
            else if Region present concavity then
                Obtain the next node, according to the previous method;
            end
        end
    end
end
```

4 Results

Using the developed algorithm, results are presented considering four geometries. The geometries of the bottle and the plane are illustrated in Naozuka [7],while the geometries of the foot and the breast were illustrated in Copett *et al.* and Foucher, Ibrahim and Saad, [8] and [9], respectively. Naozuka, used geometries to evaluate a mesh generator in generalized coordinates involving the multiblock technique. In the geometries of the foot and the breast, the authors Copett *et al.* and Foucher, Ibrahim and Saad present numerical simulations using partial differential equations, to describe natural phenomena such as heat transfers and tumor growth. The breast geometry was also used by Maganin [10], to describe tumor growth, using the finite difference method in a cartesian coordinates mesh.

The contour points of the geometries were collected using the program *WebPlotDigitizer 4.3* [11]. The contours, given contour, blue, and the approximate contour, red, are illustrated in the Figures 5-8. All geometries were calculated using the partitions $N_x = N_y = 120$ and $N_x = N_y = 150$, shown on the left and right of the figures, respectively. In the Figures 5-8 in the left, the zoom images highlight the mesh nodes, while the right images present the same details but with refinement.

Geometry details: **bottle**: 30 nodes in the given contour, 338 and 642 nodes in the approximate contours; **plane**: 230 nodes in the given contour, 516 and 872 nodes in the approximate contours; **foot**: 119 nodes in the given contour, 408 and 681 nodes in the approximate contours; **breast**: 84 nodes in the given contour, 398 and 666 nodes in the approximate contours.

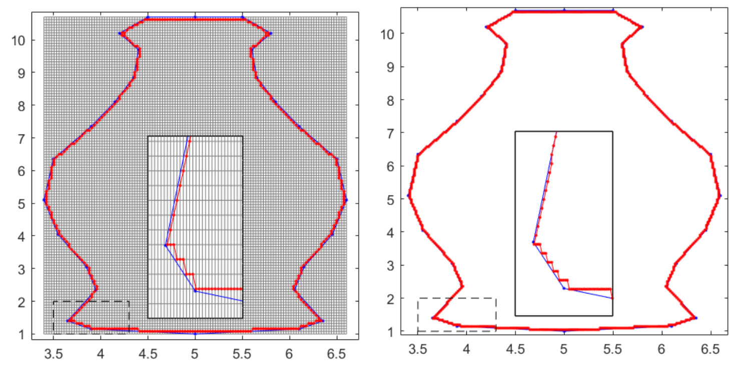

Figure 5. Bottle geometry: 338 and 642 nodes in the approximate contours, left and right of the figure

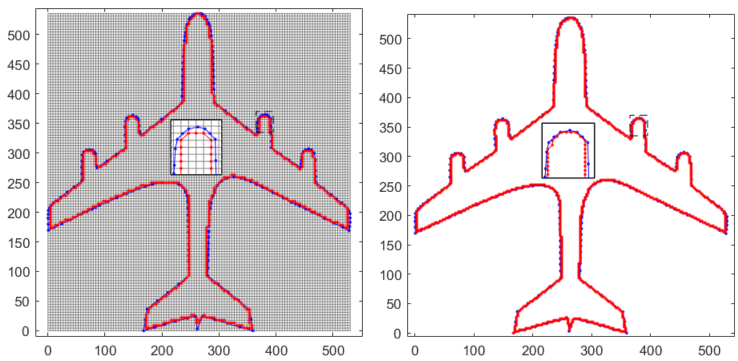

Figure 6. Plane geometry: 516 and 872 nodes in the approximate contours, left and right of the figure

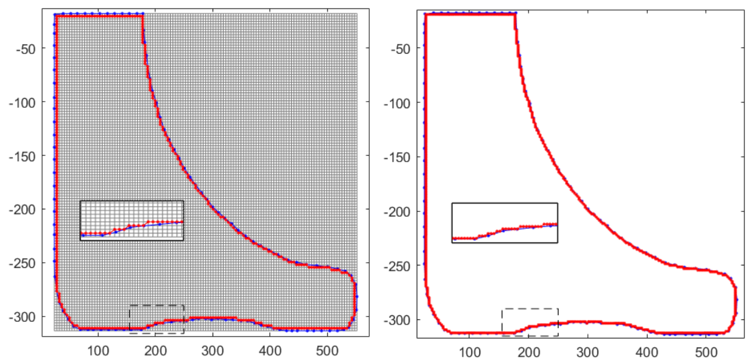

Figure 7. Foot geometry: 408 and 681 nodes in the approximate contours, left and right of the figure

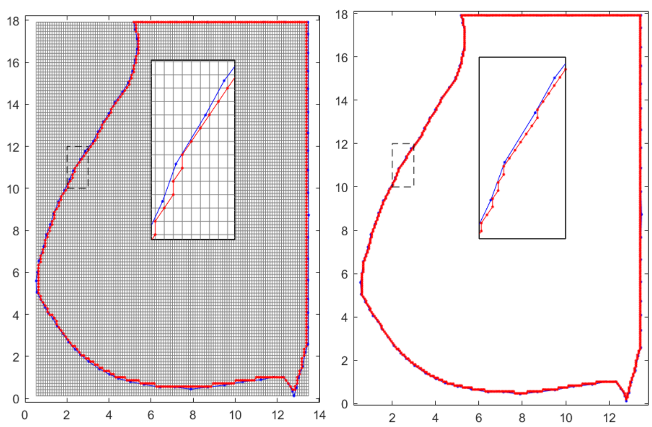

Figure 8. Breast geometry, 398 and 666 nodes in the approximate contours, left and right of the figure

5 Conclusion

In this work, it was developed an algorithm that describes the approximate contour of an irregular region, that is, it describes the physical domain of a problem in a discretized computational mesh, thus allowing the application of numerical simulation techniques to solve differential equations.

However, it was observed that in more complex geometries there were failures during the verification described in the first case of the exceptions, the connections, due to the verification being executed only in the subsequent line. This results in external points present in the final approximate contour. In this way, it is necessary to improve the verification to correct such flaws.

Acknowledgements. The work of Silva, P. Z was partially supported by CNPq under the process 152547/2019-3.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work.

References

[1] Cuminato, J. A. & Meneguette, M., 2013. *Discretização de equações diferenciais parciais: técnicas de diferenças finitas*. Sociedade Brasileira de Matemática.
[2] Othechar, P. F. S., 2013. Analysis of finite difference numerical methods for solving the poisson equation in irregular domains (in portuguese). Master's thesis, Programa de Pós-graduação em Matemática Aplicada e Computacional da Universidade Estadual Paulista Júlio de Mesquita Filho,, Presidente Prudente, SP., Brazil.
[3] Fernández-Fidalgo, J., Clain, S., Ramírez, L., Colominas, I., & Nogueira, X., 2020. Very high-order method on immersed curved domains for finite difference schemes with regular cartesian grids. *Computer Methods in Applied Mechanics and Engineering*, vol. 360, pp. 112782.
[4] Jomaa, Z. & Macaskill, C., 2005. The embedded finite difference method for the poisson equation in a domain with an irregular boundary and dirichlet boundary conditions. *Journal of Computational Physics*, vol. 202, n. 2, pp. 488 – 506.
[5] Fukuchi, T., 2014. Finite difference method and algebraic polynomial interpolation for numerically solving poisson's equation over arbitrary domains. *AIP Advances*, vol. 4, n. 6, pp. 060701.
[6] Codina, R. & Baiges, J., 2009. Approximate imposition of boundary conditions in immersed boundary methods. *International Journal for Numerical Methods in Engineering*, vol. 80, n. 11, pp. 1379–1405.
[7] Naozuka, G. T., 2018. Generation and quality analysis of computational meshes in curvilinear coordinates (in portuguese). Master's thesis, State University of Londrina - Department of Computer Science, Londrina, Pr, Brazil.
[8] Copetti, M., Durany, J., Fernández, J., & Poceiro, L., 2017. Numerical analysis and simulation of a bio-thermal model for the human foot. *Applied Mathematics and Computation*, vol. 305, pp. 103 – 116.
[9] Foucher, F., Ibrahim, M., & Saad, M., 2018. Convergence of a positive nonlinear control volume finite element scheme for solving an anisotropic degenerate breast cancer development model. *Computers Mathematics with Applications*, vol. 76, n. 3, pp. 551 – 578.
[10] Maganin, J., 2020. Simulation of a mathematical model of tumor growth in the breast using finite difference method (in portuguese). Master's thesis, State University of Londrina - Department of Mathematics, Londrina, Pr, Brazil.
[11] Rohatgi, A., 2020. Webplotdigitizer - version 4.3. `https://automeris.io/WebPlotDigitizer`.