

**Department of Applied Mathematics**

**Cancer detection in CT scans**

**Final Project for Bachelor of Science (BSc) in Applied Mathematics**

**Sara Hamoud and Amal Nasiralden**

**(שרה חמוד - אמאל נסר אלדין)**

**Supervisor: Prof. Emil Saucan**

**Contents:**

1. **Abstract…………………………………………………………………………….....3**
2. **Definitions …………………………………………………………………….......... 4**
3. **Introduction ……………………………………………………………………...… 8**

**3.1** Relationship Between Gaussian Curvature and Surface Geometry **………..…..... 9**

**3.2** General Formulas for Gaussian Curvature **………………………………………11**

**3.3** Examples of Gaussian Curvature **……………………………………………......11**

**3.4** Ricci Curvature **……………………………………………………………….... 13**

**3.5** The Relationship Between the Heat Equation and Ricci /Gaussian

Curvature **…………………………………………………………………………...13**

**3.6** The connection between the Heat Equation and Gaussian and Ricci Curvature **………………………………………………………………………….. 13**

1. **Bochner’s Method ……………………………………………………………...…..14**

**4.1** Forman’s Combinatorial Ricci Curvature **……………………………………… 14**

**4.2** The Combinatorial Weitzenböck Formula **………………………………………15**

**4.3** The relationship between Curvature and the Heat Kernel **….…….………….….18**

**4.4** Euler Characteristic **……………………………………………………...……... 20**

1. **Practical Application ……………………………………………………………… 21**

**5.1** Introduction **…………………………………………………………………….. 21**

**5.2** Stages of MATLAB code implementation **…………………………………...…21**

**5.3.** Results analysis of MATLAB code **……………...……………………….…….41**

**5.4.** Python code implementation **…………………………………………….……. 46**

**5.5.** Results analysis for Python code **………………………………………….……54**

1. **Comparative Evaluation of MATLAB and Python Implementations………….54**
2. **Conclusions and Comments……………………….………………….……………55**
3. **Summary.….…………………………….…………………………………..…….……..…57**
4. **Bibliography …...………………….…....….……………………….……………… 58**

### 

### **Abstract:**

The goal of this project is to examine the applicability of Forman-Ricci curvature as a tool for detecting cancerous tumors in medical CT images. This approach is grounded in concepts from differential and combinatorial geometry and integrates theoretical mathematics with computational image processing. The main rationale is that cancerous tumors often exhibit sharp geometric deformations along their boundaries, which can be captured via pixel-wise curvature computation.

The methodology involved a sequence of steps: Preprocessing the CT image via binarization; representing white pixels as 0-cells and constructing a cell complex by connecting adjacent white pixels with 1-cells (edges); computing the Forman-Ricci curvature for each edge using a simplified combinatorial formula; constructing a curvature map by assigning each edge's curvature value to its midpoint; and finally thresholding the map to highlight regions with significantly negative curvature values.

The results demonstrate a clear correlation between highly negative curvature regions and geometrically irregular boundaries - features characteristic of tumor-like structures. In the resulting heatmap, such areas were distinctly visible and allowed accurate localization of suspicious regions, even when the anatomical structure was non-uniform or asymmetric.

The general conclusion is that advanced mathematical tools can be effectively combined with medical image analysis for the early detection of cancer. The Forman-Ricci method is particularly advantageous in that it relies solely on the local connectivity structure of the binarized image, without requiring pixel intensity or geometric distance information. This makes it both efficient, easy to implement, and promising for further development in computational diagnostics. To validate this approach, we implemented the method independently in both MATLAB and Python, using the same CT images in each environment. This allowed us to verify the consistency and robustness of the results across platforms.

**2.Definitions:**

**1) Curvature:** In differential geometry, curvature measures how a geometric object deviates from being flat or straight. For a surface, the Gaussian curvature at a point is the product of the principal curvatures at that point (see [6]).

**2) Principal curvatures:** The minimal and maximal values kmin(p), kmax(p) of normal curvatures are called the principal curvatures (at p) (see [11]).

**3) Manifold:** A manifold is a topological space that near each point resembles Euclidean space, allowing for the application of calculus and differential geometry (see [2]).

**4) Cell complexes:** cell complex is a type of topological space constructed by "gluing together" cells of various dimensions (such as points, line segments, polygons) in a specific manner, used in algebraic topology to study the properties of spaces by breaking them down into simpler pieces (see [8]).

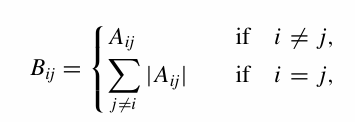
**5) Riemann-Laplace operator:** A generalization of the Laplace operator to Riemannian manifolds, defined as Δgf = div(∇f), where ∇f is the gradient and {div} is the divergence with respect to the metric g (see [7]).

**6) p-forms:** In mathematics, particularly in differential geometry, p-forms are mathematical objects that generalize the concept of integration to higher dimensions (see [6], Chapter 14).

**7) Bochner (or rough) Laplacian:** is a second-order differential operator defined on sections of a vector bundle with a connection ∇ over a Riemannian manifold. It is given by Δ = ∇\*∇, where ∇ is the covariant derivative, and ∇\* is its formal adjoint. This operator acts as a generalized Laplacian on vector-valued functions (sections of the bundle) (see [5]).

**8) Curvature tensor:** In Riemannian geometry, the Riemann curvature tensor is a fourth-order tensor that describes the curvature of a manifold. It is defined using the Levi-Civita connection and measures the change of a vector after parallel transport around a closed loop in the manifold (see [7]).

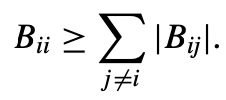
**9) Bochner matrix:** Let A = (Aij) be a symmetric n ×n matrix. Bochner matrix defined as : B(A) = (Bij) the symmetric n × n matrix whose entries are given by:

**** (see [11])

**10) Curvature matrix:** Let A = (Aij) be a symmetric n ×n matrix. The curvature matrix defined as: F(A)=(Fij) the n ×n diagonal matrix whose:

 (For further details see [1]).

**11) Strongly nonnegative matrix:** Say a symmetric n ×n matrix B is strongly nonnegative if, for each 1 ≤i ≤n,



Note that for every A the associated Bochner matrix B(A) is strongly nonnegative (see [1]).

**12)** **P-cell:** is a building block of a topological space that has **p dimensions**. It generalizes the idea of points, edges, faces, and higher-dimensional counterparts:

* **0-cells** → Points (vertices)
* **1-cells** → Line segments (edges)
* **2-cells** → Polygons (faces, like triangles, squares)
* **3-cells** → Polyhedra (volumes, like tetrahedra)
* **p-cells** → p-dimensional generalization (such as a 4D hypercube in higher dimensions) (see [3]).

**13)** **B-neighbors:** If α1 and α2 are p-cells of M, say α1 and α2 are neighbors if

**(1)** α1 and α2 share a (p +1)-cell, that is, there is a (p + 1)-cell β with β > α1 and β>α2. (β > α means α is a face of β).

**(2)** α1 and α2 share a (p −1)-cell, that is, there is a (p − 1)-cell γ with γ<α1and γ<α2. We say that α1 and α2 are parallel neighbors if either (1) or (2) is true but not both. If both (1) and (2) are true, we say α1 and α2 are transverse neighbors (see [1]).

**14) Cp(M, R):** denotes the space of real-valued (R) p-cochains on M. (see [3]).

**15):** means the function has infinitely many derivatives, and all of them are continuous (see [6]).

**16)** **Compact manifold:** A Riemannian manifold M, that is closed and bounded (see [4]).

**17) Eigenfunction:** A function Δ on a Riemannian manifold M is called an eigenfunction of Δ, if there exists a scalar λ such that Δf = λf, where Δ denotes the Laplace Beltrami operator (see [13]).

**18) Tensors:** Multidimensional arrays of numerical values and therefore generalize matrices to multiple dimensions (see [9]).

**19) Euclidean plane:** The Euclidean plane is the set ℝ² = {(x, y) | x, y ∈ ℝ}, equipped with a distance function

This makes ℝ² into a metric space. We define a line to be the set of points satisfying a linear equation ax + by + c = 0, where a and b are not both 0. The Euclidean plane, together with these lines, is the setting for Euclidean geometry (see [12]).

**20) Covariant derivative:** Let M be a smooth manifold and the space of smooth vector fields on M. A **covariant derivative** is defined as a bilinear operator

denoted , satisfying these properties:

1. **Linearity in the first argument**:
2. **Leibniz rule in the second argument**:

The quantity is called the **covariant derivative** of Y in the direction of X

(see [7]).

**3. Introduction**

In the field of Image Processing, *curvature* solved a problem related to shape detection, edge features, and object tracking. In the past, when trying to detect shapes or track them in an image, simpler approaches like detecting straight edges were used [10], which were not always accurate for objects with curved or irregular lines.

By calculating curvature, it became possible to better analyze the edge features of irregular shapes and differentiate between straight and curved elements. This enables more effective detection of complex objects, such as body organs in medical scans (like CT or MRI), fingerprint recognition, or facial feature detection in images.

Classically, curvature is a geometric property that describes the degree of bending or change in direction of a line or surface (or, more generally, a manifold) at a specific point. For a curve, it reflects the rate of angular change between adjacent segments of a curve, where the sharper the turns of the curve, the greater the curvature.

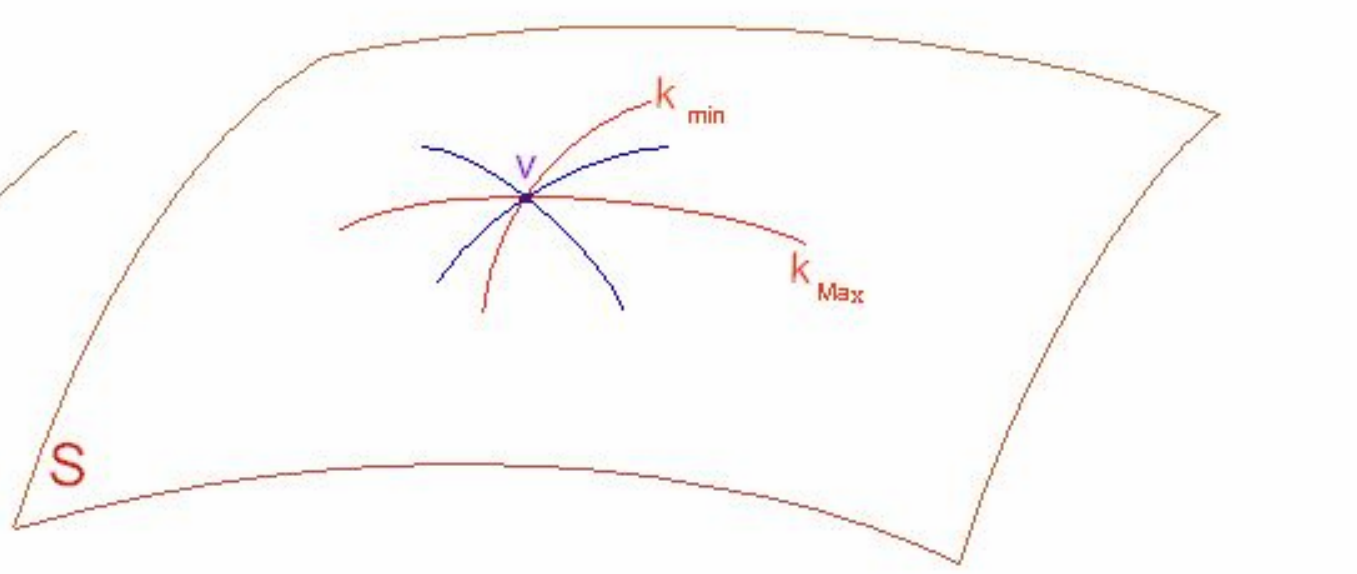
Mathematically, curvature is classically defined as the second derivative of a curve as parametrized by the arc length. In other words, it provides information about the rate of angular change between neighboring points along a contour or boundary of an object. The sharper the change, the higher the curvature; if the contour is completely straight, the curvature is zero.

In the field of Image Processing, curvature is used to analyze the lines and edges of objects within an image. By calculating curvature, it is possible to more accurately detect curves and non-linear edges, enhancing the performance of tasks such as shape recognition, segmentation, and object tracking. Curvature calculations allow the distinction between straight and curved elements of an object's contour, aiding in the detection and analysis of geometric shapes, especially in situations where the boundaries are complex or irregular.

In Image Processing, two main types of curvature are used: Gaussian curvature and more recently Ricci curvature. Each curvature provides a unique measure of how surfaces bend, serving distinct purposes in image analysis and pattern recognition processes.

**Gaussian curvature** (K): Gaussian curvature is a measure of the direction and intensity of the local “bending” of a surface at a given point. It is defined as the product of the extreme values (maximum and minimum) of the normal curvature in different directions.

If and ​ are the *principal curvatures* (see Figure 1) of a surface at a point p, then

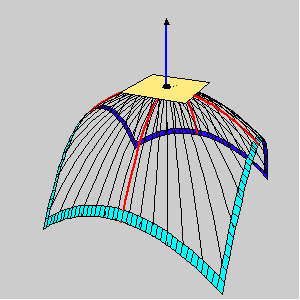


### **Figure 1:** The principal curvature of a surface at a point

### **3.1. The Relationship Between Gaussian Curvature and Surface Geometry**

Gaussian curvature classifies the points on a surface as follows:

1. **Elliptic Point (K>0):** (see Figure 2)
   * > 0 and > 0 or < 0 and < 0.
   * The surface curves in the same direction (e.g., a sphere).
   * Locally the shape of the surface is like the shape of "bowl".



**Figure 2:** an elliptic point

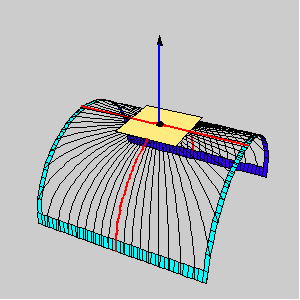
1. **Hyperbolic Point (K<0):** (see Figure 3)
   * < 0, > 0
   * Locally, the surface curves in opposite directions (e.g., a saddle).

תמונה שמכילה ציור, עיצוב, אומנות

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

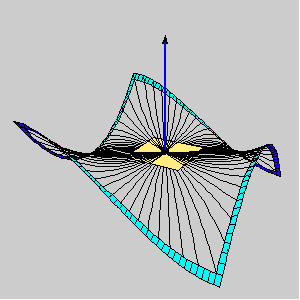
**Figure 3:** a hyperbolic point

1. **Parabolic Point (K=0):** (see Figure 4)
   * Only one principal curvature is non-zero (e.g., a cylinder).



**Figure 4:** a parabolic point

1. **Planar Point (K=0):** (see Figure 5)
   * Locally, in the vicinity of the point, the surface is entirely flat



**Figure 5:** a planar point

### **3.2. General Formulas for Gaussian Curvature**

If the surface S is represented as a graph of a function z = f (x, y) , Gaussian curvature can be computed using:

Where we used the classical notation:

* ​: are first partial derivatives of f (x, y).
* , , are second partial derivatives.

### 

### **3.3. Examples of Gaussian Curvature**

1. **Sphere of Radius R:**

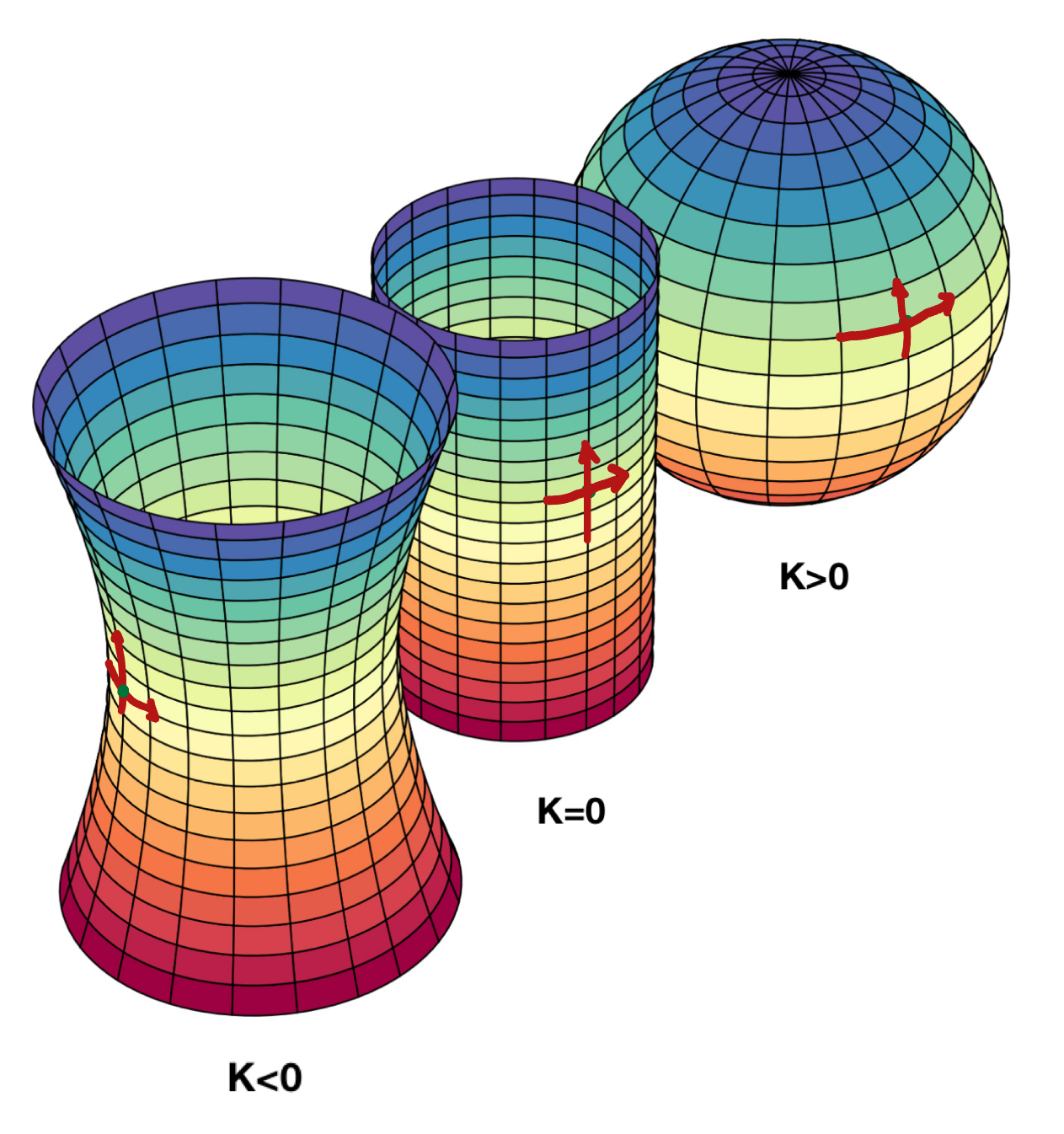
Both principal curvatures (k1 = k2​) are positive which means both curve in the same direction, so their product is also positive . (See the image below).

1. **Hyperbolic Saddle:**

One principal curvature is positive, and the other is negative which means the two curves bend in opposite directions, resulting in a negative curvature (See the image below).

1. **Cylinder of Radius R:**

It represents a flat surface, where the surface has zero curvature if it is flat in atleast one direction**,** since one principal curvature is zero, the curvature is also zero (See the image below).



**Figure 6**

From left to right: a surface of negative Gaussian curvature ([hyperboloid](https://en.wikipedia.org/wiki/Hyperboloid)), a surface of zero Gaussian curvature ([cylinder](https://en.wikipedia.org/wiki/Cylinder_(geometry))), and a surface of positive Gaussian curvature ([sphere](https://en.wikipedia.org/wiki/Sphere)).

**3.4. Ricci Curvature:**  Ricci curvature is a concept from differential geometry used to describe the geometric properties of manifolds. Generally, it measures how a *manifold* curves in different directions around a specific point. Unlike Gaussian curvature, which focuses on a single direction, Ricci curvature is an average of the curvatures in all different tangential directions.

**3.5.** **The Relationship Between the Heat Equation and Ricci /Gaussian Curvature:**

The connection between the heat equation and Ricci / Gaussian curvature stems from the heat equation’s ability to describe heat diffusion on a manifold, specifically how heat "spreads" over a surface endowed with geometric properties like curvature.

**3.6. The connection between the Heat Equation and Gaussian and Ricci Curvature:**

* **Gaussian Curvature:**Gaussian curvature is a geometric invariant that measures how much a surface deviates from being flat. The heat equation on a non-flat surface (such as a surface embedded in three-dimensional space) varies depending on the surface’s curvature. In regions with high curvature, heat diffusion behaves differently compared to areas with low curvature. In medical image processing, the heat equation models the propagation of heat (or noise) propagates and changes across regions with varying geometric structures, such as in CT or MRI scans.
* **Ricci Curvature:**Ricci curvature quantifies volume growth in space. The heat equation serves as a tool to describe how phenomena like tissue changes, tumors, or pathological growth spread in regions with significant variations in Ricci curvature.

This understanding aids in improving the detection of anomalies such as tumors or other pathological areas.

**4. Bochner’s Method**

**4.1. Forman’s Combinatorial Ricci Curvature:**

Forman suggests a way to generalize the concept of Ricci curvature so that it also applies to combinatorial structures (objects) (such as *cell complexes)*, as follows:

One starts with the following form of the Bochner-Weitzenbock formula for the *Riemann-Laplace operator* on *p-forms* on (compact) Riemannian manifolds:

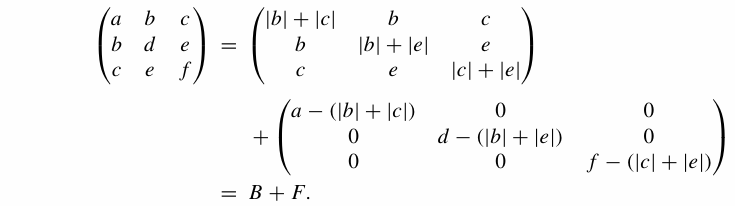
Where is the exterior derivative, which maps   
The operator is the adjoint of with respect to the inner product on forms, mapping .

is the *Bochner (or rough) Laplacian* (12)and where *Curv(R)* is an expression with linear coeﬃcients of the *curvature tensor* (Here is the *covariant derivative* operator.)

Moreover, there exists a canonical decomposition of the form:

where Bp is a nonnegative operator and, when expressed as a matrix with respect to a basis of *Cp(M, )* consisting of the *p-cells* of M, Fp is a diagonal matrix.

To present this decomposition, we begin by defining a decomposition for any symmetric matrix. Instead of providing a general definition, we illustrate the decomposition for a symmetric 3×3 matrix.



The linear map defined by matrix B is nonnegative (with respect to the standard inner product on ). This is a special case of the fact that any symmetric matrix () such that for each i is nonnegative. The nonnegativity of B implies that in this simple example we can already make the Bochner-like statement that if each of the three numbers

is positive, then the original 3 × 3 matrix has a trivial kernel. (Since any positive definite matrix has a trivial kernel)

**4.2. The Combinatorial Weitzenböck** **Formula:**

The Weitzenböck decomposition expresses any symmetric n×n matrix A as the sum:

**A=B(A)+F(A)**

This decomposition separates the matrix into two components:

B(A), which is a nonnegative definite symmetric matrix called the *Bochner matrix*.

F(A) a diagonal correction matrix known as the curvature matrix.

A key property of this decomposition is that B(A) retains the off-diagonal entries of A while adjusting the diagonal to be nonnegative, making it nonnegative definite. This decomposition provides a useful framework for analyzing symmetric matrices, particularly in the context of the combinatorial Laplace operator. Specifically, B(A) captures the nonnegative structure of the matrix and is used to study spectral properties and positive semi-definiteness, while F(A) encodes the curvature information and is useful in geometric interpretations and corrections.

**Theorem 1:** Suppose B is a *strongly nonnegative matrix*. Then B is nonnegative definite, i.e., for every v ∈ , the following holds:

**Proof:**

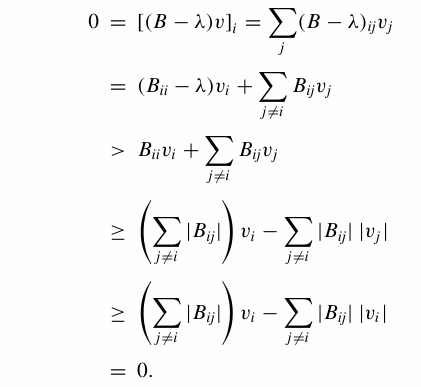
Suppose B is not non-negative definite. Since B is symmetrical, it is diagonalizable. Thus, there must be a λ<0 and a nonzero vector ∈ such that B = λ. The contradiction will be reached via a maximum principle:

Let: ||= ||.

Since , || > 0. Multiplying by −1, if necessary, we may assume that > 0.

Then

(B −λI) = 0,



This is a contradiction.  ****

Denote by *Cp*(M, ) the oriented real chains of M. The (real-) homology of M is the homology of the chain complex



where *n* = dim M , and ∂p : Cp → Cp−1 denotes the standard boundary operator.

Choose a positive weight > 0 for each cell α, and endow each with an inner product, which we denote by ,by declaring the p-cells to be orthogonal, and setting for each p-cell α,

האם מניחים שאיברי Cp אורתוגולנלים בהמשך פי זה?

תמונה שמכילה גופן, לבן, קליגרפיה, טיפוגרפיה

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

We can now define the adjoint operator



By



for each cp ∈ Cp and cp+1∈ Cp+1. This leads us to the combinatorial Laplace operator



**\***

Our next goal is to find a more explicit representation of . For any cell *α* of M, we write if dim(*a*) = *p*. For cells α and *β* we write *α < β* if and say *α* is a face of *β*. \*להסביר במילים שמדובר בפאות במובן הרגיל

Choose an orientation for each cell. For any (p+1)-cell β we can write

תמונה שמכילה גופן, טקסט, לבן, גרפיקה

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

Where is the incidence number of *β* relative to *α*. Let *α < β* , for each oriented p-cell *α* and (p+1)-cell *β*, לציין שאפסילון תמיד אחד בפיקסלים

תמונה שמכילה טקסט, גופן, כתב יד, לבן

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

**\***

Plugging this expression into we see that for any p-cell ,

תמונה שמכילה גופן, לבן, קו, טקסט

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

We observe that is a self-adjoint operator with respect to the inner product It is natural to express as a matrix with respect to a basis of that is orthonormal with respect to the inner product. For each p-cell *α*, with its chosen orientation, define

תמונה שמכילה טקסט, גופן, מספר, לבן

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

Then the ’s form an orthonormal basis. Let denote the set of normalized p-cells. For we have

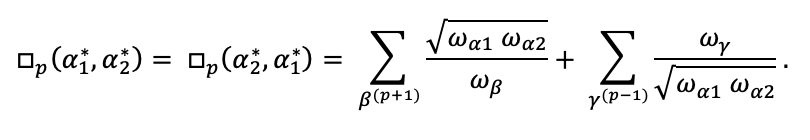
......תמונה שמכילה גופן, לבן, טקסט, קו

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

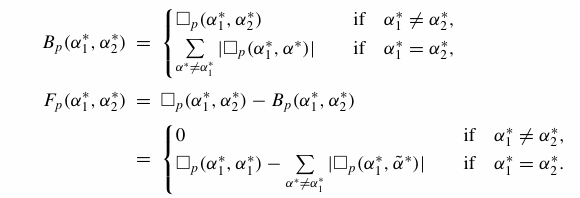
תמונה שמכילה גופן, לבן, קו, טקסט

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

We can think of the Laplace operator as a symmetric matrix, with the rows and columns indexed by the elements of .For and in , let denote the corresponding element of the matrix (equivalently,

…..

we now define two new operators, =( , the combinatorial Bochner Laplacian, and =( , the p-th combinatorial curvature operator, in matrix form, by:



These operators have the following desirable properties:

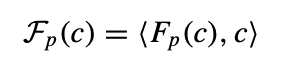
(1) = (The combinatorial Bochner–Weizenbock formula).

(2) is a strongly non-negative matrix.

(3) is a diagonal matrix.

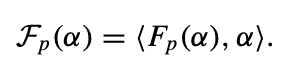
We now take a closer look at the curvature operator *Fp*. For any *p*-chain *c* ∈

define the *p-*th curvature of *c* by:



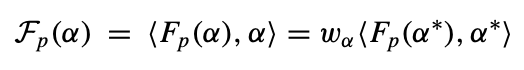
(here we are thinking of Fpas a linear map from Cpto Cp). For any *p*-

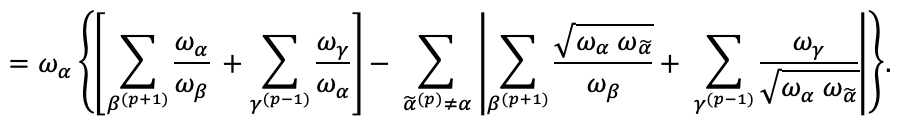
cell α,



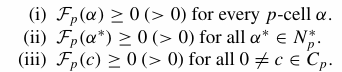
Note that is independent of the orientation of . More explicitly, we have:

**Theorem 2:** For any p-cell α, the p-th curvature function applied to α, Fp(α), is given by:

**

**

We say that the operator (> 0) if any of the following three equivalent conditions hold:



In analogy with the Riemannian case, we make the following definition:

For any 1-cell α, we define the Ricci curvature of α by:



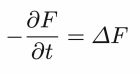
**4.3. The relationship between curvature and the heat kernel:**

Let M be a *compact* *Riemannian manifold*. There is a function:



which is

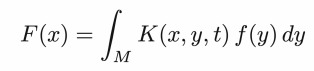
1. Given any initial data the solution of the heat equation:



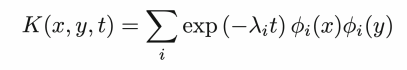
with:



is given by:

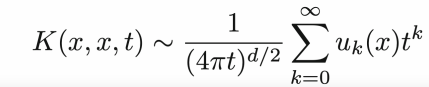


1. *K* is given by the convergent series: A



(where the *eigenfunctions*  of the Laplace operator Δ are chosen so that they form an orthonormal basis on M)

1. For every there is an asymptotic expansion as of the form:



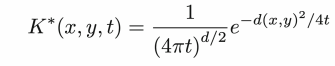
where the are functions given by universal formulae expressing in terms of the curvature *tensor* of *M* and its covariant derivatives at the point X.

The three-argument function K is called the *fundamental solution* of the heat equation on M, or the *heat kernel of M.*

After expanding the heat kernel using a sum over eigenfunctions, we can now compare the heat kernel on a Riemannian manifold to its Euclidean counterpart. The Euclidean heat kernel in (without curvature i.e is given by:



More generally, in dimension *d*, the Euclidean heat kernel is:

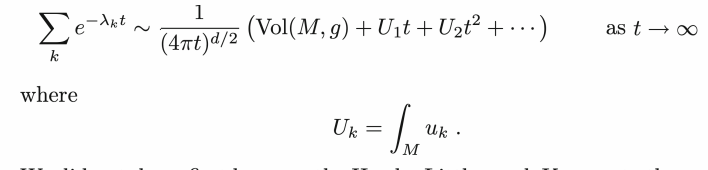


This expression allows us to interpret the heat kernel K(x,x,t) on a Riemannian manifold as a deformation of the Euclidean heat kernel. In fact, the asymptotic expansion illustrates this. By integrating the heat kernel over the manifold



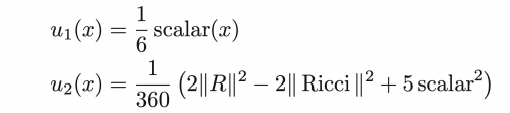
This result highlights the connection between the asymptotic expansion of the heat kernel and the volume of the manifold.

The asymptotic expansion above is taken from the full asymptotic expansion:



The curvature is neglected in the previous formula because it is difficult to understand geometrically, and starting from ​ onward, the curvature terms become very hard to compute.

Thus, the computation of ​ and ​ is:



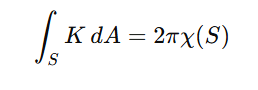
The formula above represents the deep connection between the curvature and the heat kernel.

# **4.4. Euler characteristic:**

For a compact, two-sided surface s such as a sphere or a torus, for example. the Euler characteristic is:

Where is the genus of a surface that is the number of "holes" or handles it has.

**The Gauss–Bonnet Theorem:** states that for a compact, orientable surface without boundary:



;

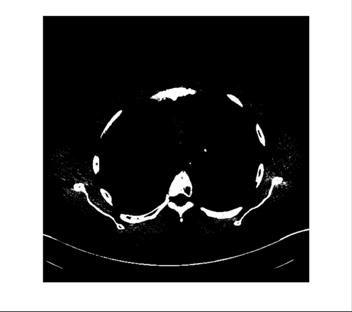
where:

* K is the Gaussian curvature of S,
* dA is the area element on S,
* χ (S) is the Euler characteristic of the surface.

**5. Practical Application: Computing Forman-Ricci Curvature on a CT Image for Cancer Detection**

**5.1. Introduction – Why Use Curvature?**

As discussed in previous chapters, Forman-Ricci curvature is a discrete version of Ricci curvature that can be applied to networks or pixelated images. When analyzing a CT scan of the human body, cancerous tumors often appear as irregular shapes or sharp boundaries within tissue, geometric features that can be detected using curvature. In simpler terms: Low or negative Forman-Riccicurvature may indicate sharp edges, convex boundaries, or irregular structures, features that are often associated with cancerous growths.



**Figure 7 - Pelvis CT scan**

To apply curvature-based analysis to medical scans and detect cancerous tumors, we used a processed CT image of the chest area (see **Figure 7**). The image was preprocessed and binarized, meaning that only the relevant anatomical structures (e.g., a suspected tumor) appear in white, while the background (air, soft tissues, etc.) appears in black.

**5.2. Stages of MATLAB Code Implementation**:

To compute Forman-Ricci curvature on this image, we followed these steps:

**1. pelvis CT scan**

#### **Step 1: Loading the CT Image and Preprocessing (Binarization):**

img = imread('pelvis.png'); %Load the image

% Convert to grayscale if the image is in color

if size(img,3) == 3

img\_gray = rgb2gray(img);

else

img\_gray = img;

end

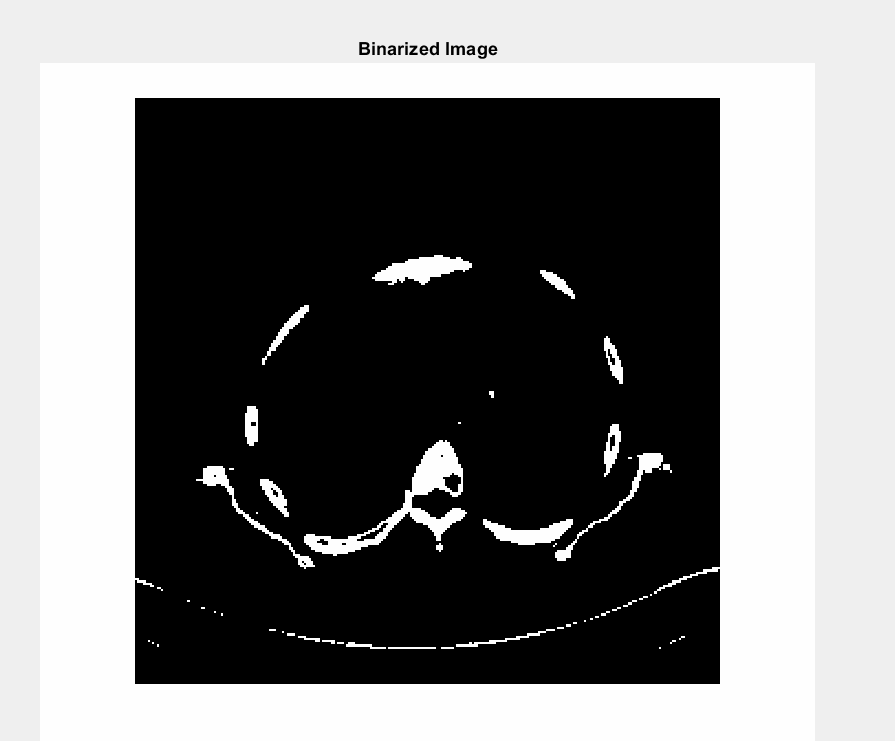
% Binarize the image (e.g., using automatic thresholding)

bw = imbinarize(img\_gray);

% Display the binary image

imshow(bw);

title('Binarized Image');



**Figure 7 - Binarized CT image**

**Step 2: Converting White Pixels into a Cell Complex:**

Each white pixel represents a 0-cell (vertex). Edges (1-cells) are defined between neighboring white pixels (including diagonals).

**Matlab code:**

% Identify white pixels in the binary image (potential vertices of the graph)

[row, col] = find(bw);

numVertices = length(row);

% Map each white pixel's position to a unique vertex index in the graph

pixelIndexMap = zeros(size(bw));

pixelIndexMap(sub2ind(size(bw), row, col)) = 1:numVertices;

% Define neighborhood directions (8-connectivity in 2D grid)

neighbors = [ -1, -1; -1, 0; -1, 1;

0, -1; 0, 1;

1, -1; 1, 0; 1, 1 ];

% Estimate maximum possible number of edges (each vertex may have up to 8 neighbors)

estimatedMaxEdges = numVertices \* 8;

edges = zeros(estimatedMaxEdges, 2);

edgeCount = 0;

for n = 1:size(neighbors,1)

dr = neighbors(n,1);

dc = neighbors(n,2);

r2 = row + dr;

c2 = col + dc;

% Ensure neighboring pixels lie within the image boundaries

valid = r2 >= 1 & r2 <= size(bw,1) & c2 >= 1 & c2 <= size(bw,2);

r1 = row(valid);

c1 = col(valid);

r2 = r2(valid);

c2 = c2(valid);

ind1 = sub2ind(size(bw), r1, c1);

ind2 = sub2ind(size(bw), r2, c2);

v1 = pixelIndexMap(ind1);

v2 = pixelIndexMap(ind2);

% Construct directed edges from each vertex to its valid neighbor,

% ensuring each undirected edge is represented only once (v1 < v2)

validEdge = v1 < v2 & v2 > 0;

newEdges = [v1(validEdge), v2(validEdge)];

nEdges = size(newEdges,1);

edges(edgeCount+1:edgeCount+nEdges, :) = newEdges;

edgeCount = edgeCount + nEdges;

end

% Remove unused rows from the edge list

edges = edges(1:edgeCount, :);

% Remove duplicate edges to obtain a simple undirected graph

edges = unique(edges, 'rows');

% Visualization: overlay the resulting graph (cell complex) on the binary image

if ~isempty(edges)

figure;

imshow(bw);

hold on;

plot(col, row, 'go', 'MarkerSize', 3, 'LineWidth', 1); % Plot vertices

for k = 1:size(edges,1)

v1 = edges(k,1);

v2 = edges(k,2);

x = [col(v1), col(v2)];

y = [row(v1), row(v2)];

plot(x, y, 'r-', 'LineWidth', 0.5); % Plot edges

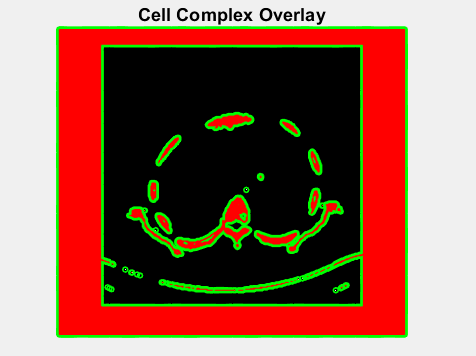
end

title('Optimized Cell Complex Overlay');

else

disp('No cell complex generated: no edges found.');

end

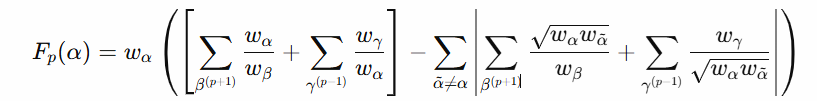
**Output:**

**Figure 8 - Cell complex generated from white pixels, with 0-cells (green dots) and 1-cells (red edges) overlaid on the binary image.**

Figure 8: A graph structure constructed from the binary image of a CT scan. Each green pixel represents a vertex, and green lines connect adjacent pixels—edges. This results in a graph describing the shape of the suspicious tissues as preparation for computing the Forman-Ricci curvature.

**Step 3: Computing Forman-Ricci Curvature for Each Edge:**

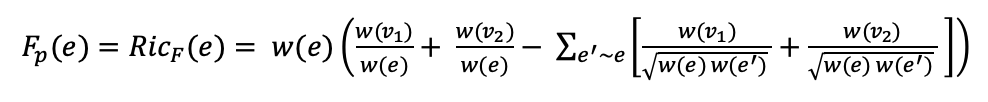
The simplified Forman-Ricci curvature formula used in our project is a direct consequence of the general formulation introduced by Forman. The general expression for the curvature of a p-cell α is:



**(Formula 1)**

In our setting:

* p=1, so α is a 1-cell (an edge).
* Each edge connects two 0-cells (vertices).
* There are no 2-cells (no higher-dimensional faces), so the terms involving vanish.



* We assume uniform weights: , By substituting these simplifications into the Formula 1 above: the first term becomes

Each neighboring edge contributes a value of 2 in the subtraction term.

This leads to the simplified combinatorial formula:



This version is particularly well-suited for analyzing binary images converted to graphs, as it captures local geometric irregularities using only the connectivity structure without relying on pixel intensities or geometric distances.

**MATLAB code:**

% Create a graph from the edge list

G = graph(edges(:,1), edges(:,2)); % Each edge connects two vertices

% Get the total number of edges in the graph

numEdges = numedges(G);

% Get a list of all edges as pairs of connected node indices

edgeList = G.Edges.EndNodes;

% Build a neighborhood map: for each vertex, store which edges are connected to it

% This is stored as a cell array; each cell contains a list of edge indices

numVertices = numnodes(G);

vertexEdgeMap = cell(numVertices, 1);

for e = 1:numEdges

v1 = edgeList(e,1); % First vertex of edge e

v2 = edgeList(e,2); % Second vertex of edge e

% Append the edge index to the list of each connected vertex

vertexEdgeMap{v1} = [vertexEdgeMap{v1}, e];

vertexEdgeMap{v2} = [vertexEdgeMap{v2}, e];

end

% Compute the Forman-Ricci curvature efficiently

RicF = zeros(numEdges,1); % Preallocate vector for curvature values

for k = 1:numEdges

v1 = edgeList(k,1); % First vertex of edge k

v2 = edgeList(k,2); % Second vertex of edge k

% Get the union of all edges connected to v1 and v2 (neighboring edges)

neighboringEdges = unique([vertexEdgeMap{v1}, vertexEdgeMap{v2}]);

% Exclude the current edge itself from the list of neighbors

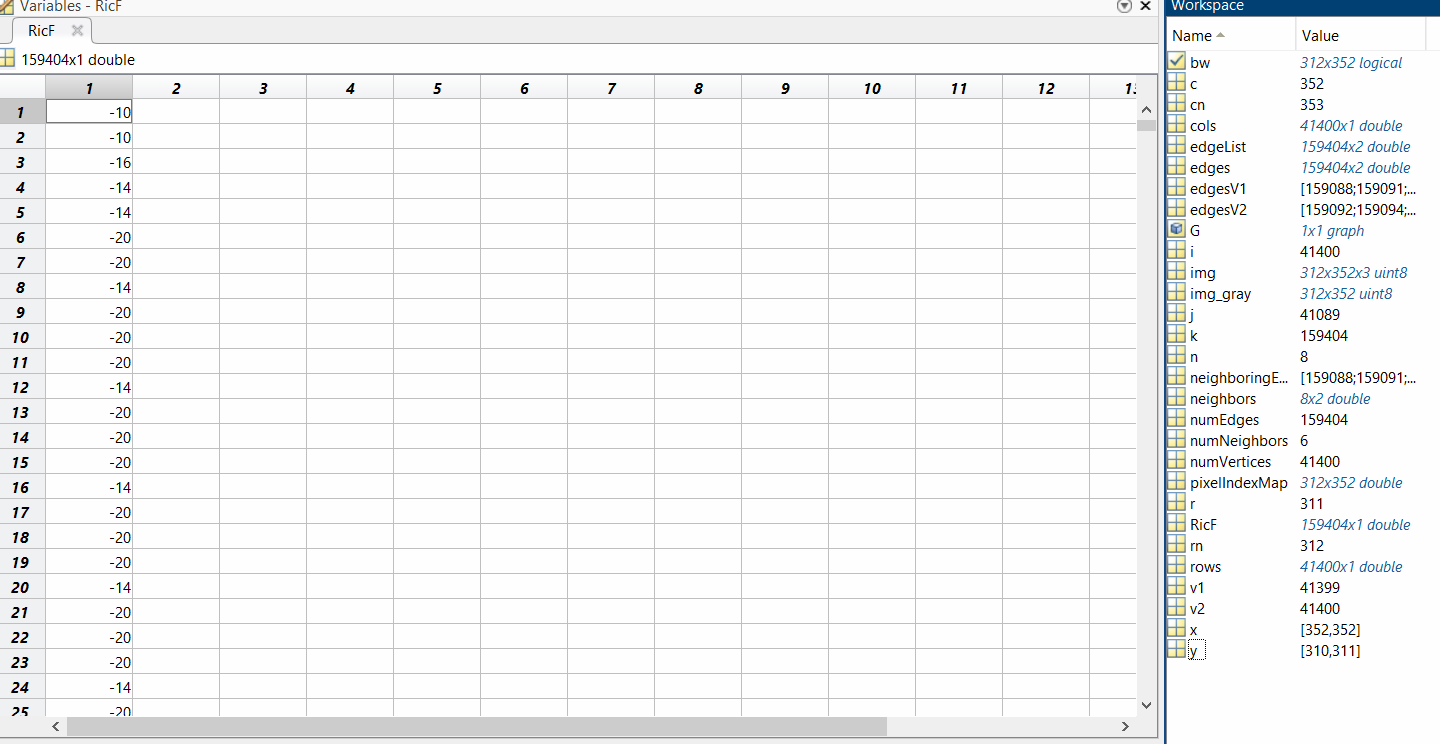
neighboringEdges(neighboringEdges == k) = [];

% Compute the curvature: RicF = 2 - 2 × (number of neighboring edges)

RicF(k) = 2 - 2 \* numel(neighboringEdges);

end

**Output:**



**Table 1– Table of Forman-Ricci curvature for each edge in the graph**

**Forman-Ricci Curvature Output Explanation:**

After computing the Forman-Ricci curvature for each edge in the graph derived from the binary CT image, we obtained a vector RicF of size 159,404×1, where each element represents the curvature value of a specific edge. The curvature was calculated using the simplified formula:



For example, an edge with 8 neighboring edges received a curvature value of:

while an edge with 10 neighbors received:

This pattern demonstrates that edges located in highly connected or geometrically irregular regions (e.g., tumor boundaries) tend to have lower or more negative curvature values. These results serve as the basis for highlighting suspicious areas in the image through curvature visualization.

**Step 4: Construction of a Curvature Map for Image Overlay:**

To visualize the curvature values along the boundary, each edge is associated with the midpoint between its two adjacent pixels. The curvature map is then overlaid on the image using a color representation.

**MATLAB code:**

% Initialize an image-sized matrix to accumulate curvature values

curvatureMap = zeros(size(bw));

% Initialize a matrix to count how many times each pixel is assigned a value

curvatureCount = zeros(size(bw));

% Loop over all edges in the graph

for k = 1:numEdges

v1 = edgeList(k,1); % Index of the first vertex of edge k

v2 = edgeList(k,2); % Index of the second vertex of edge k

% Calculate the midpoint between the two vertices (pixels)

rMid = round((row(v1) + row(v2)) / 2); % Row of midpoint

cMid = round((col(v1) + col(v2)) / 2); % Column of midpoint

% Accumulate the Ricci curvature value at the midpoint location

curvatureMap(rMid, cMid) = curvatureMap(rMid, cMid) + RicF(k);

% Increment the count of values added at this location

curvatureCount(rMid, cMid) = curvatureCount(rMid, cMid) + 1;

end

% Initialize a matrix to store the final averaged curvature values

curvatureAvg = zeros(size(bw));

% Only compute the average where at least one value was accumulated

nonzero = curvatureCount > 0;

% Compute the average curvature at each pixel (to avoid overwriting)

curvatureAvg(nonzero) = curvatureMap(nonzero) ./ curvatureCount(nonzero);

% Display the averaged curvature map as a heatmap

figure;

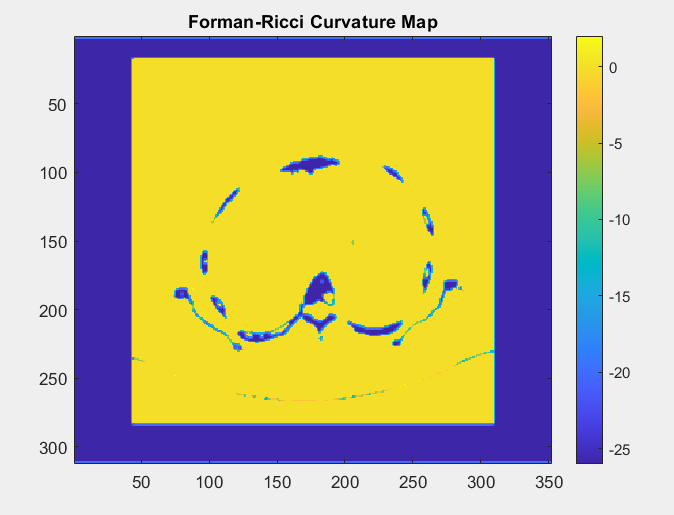
imagesc(curvatureAvg); % Show the image with curvature values

colorbar; % Show the color scale

title('Forman-Ricci Curvature Map (Averaged)');

**Output:**

To visualize the computed curvature values in a spatial context, we generated a 2D curvature map corresponding to the binary image. For each edge in the graph, we identified its two end nodes (pixels) and calculated the midpoint between them. The curvature value of the edge was then assigned to that midpoint location in a new image matrix. The resulting curvature map was displayed using a heatmap, where color intensity represents the magnitude of curvature. This visualization highlights regions with strong geometric irregularities—such as sharp, jagged, or highly connected boundary zones—through more extreme (typically negative) curvature values. It provides a clear spatial representation of where potential tumor-like structures may be located within the image.

****

**Figure 9 - 2D Forman-Ricci curvature map**

**Step 5: Analysis and Differentiation of Suspicious Regions**

Regions exhibiting highly negative or extreme curvature values can be marked as suspicious.

**MATLAB code:**

threshold = -10; % Set the threshold for identifying significantly negative curvature values

suspiciousMask = curvatureMap < threshold;

imshow(bw); % Display the original binary image

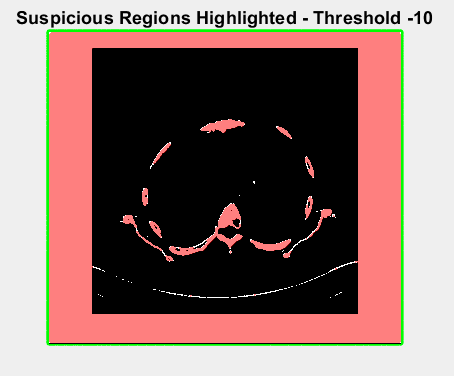
hold on;

redMask = cat(3, ones(size(bw)), zeros(size(bw)), zeros(size(bw))); % Create a red color layer

h = imshow(redMask); % Overlay the red mask

set(h, 'AlphaData', 0.5 \* suspiciousMask); % Apply transparency only where suspiciousMask is true

title('Suspicious Regions Highlighted - Threshold -10');

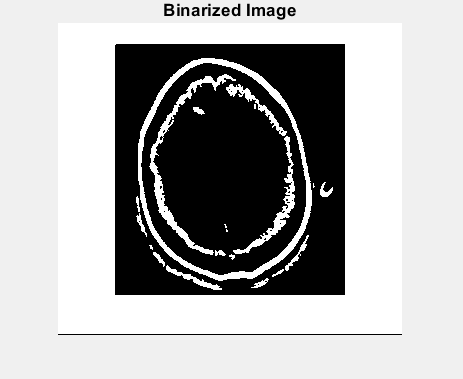
**Output :**

**Figure 10 - Suspicious regions (curvature < −10) shown in red**

The red-highlighted regions represent points in the curvature map where the Forman-Ricci curvature falls below a specified threshold (e.g., -10). These areas correspond to highly irregular boundary zones and are thus considered geometrically suspicious for potential tumor detection.

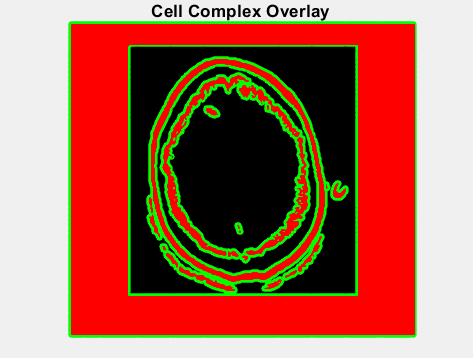
**Outputs for another CT scans:**

**2. Brain CT scan 1:**

**Step1:**

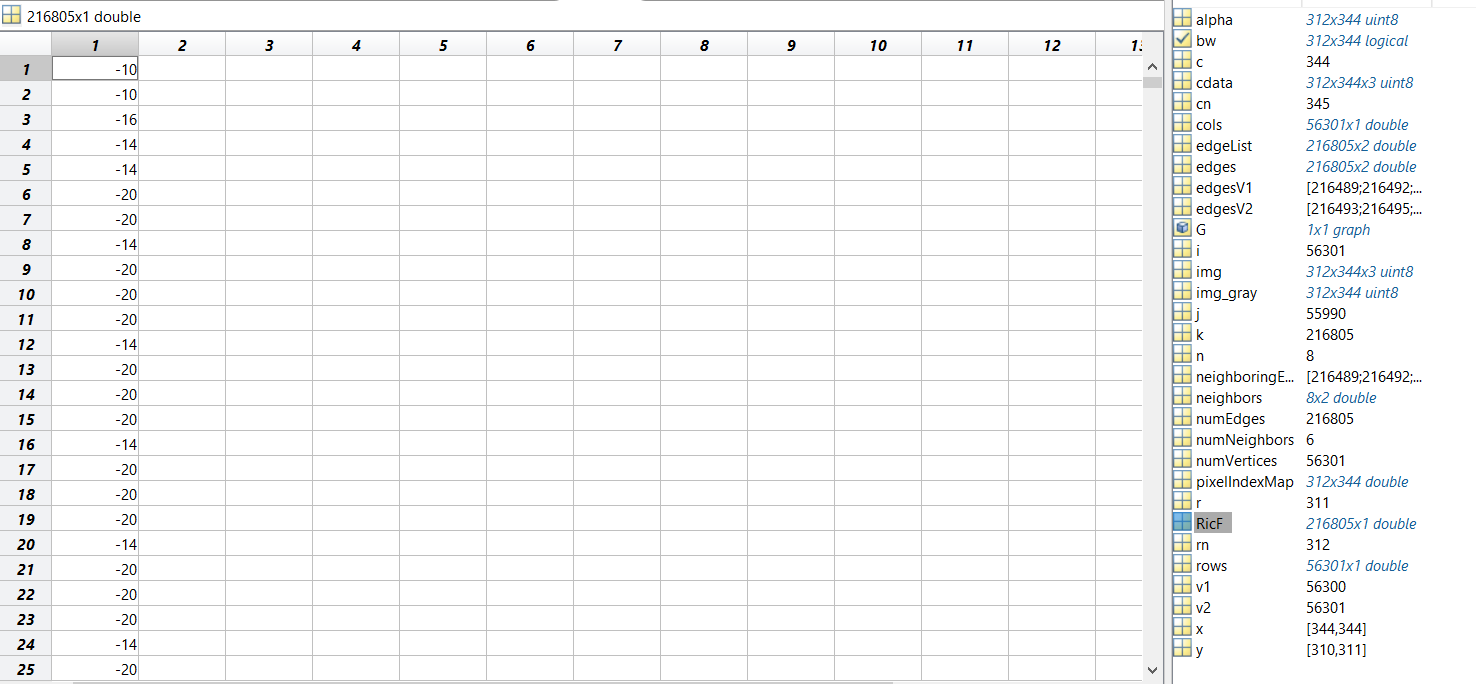
**Figure 11** - **Binarized CT Image**

**Step2:**



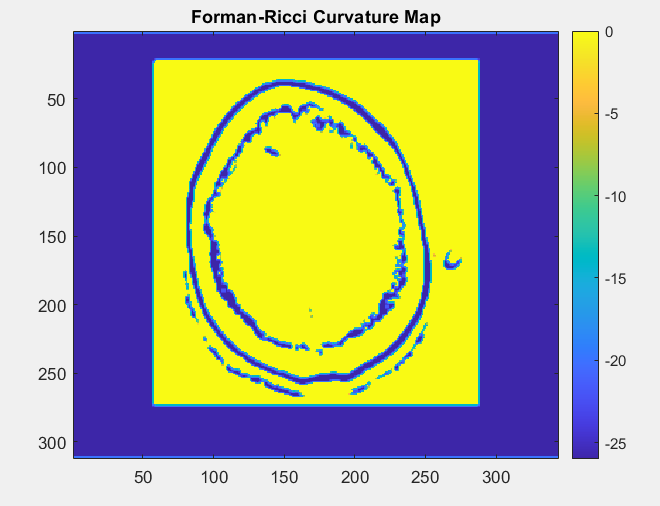
**Figure 12 –** **Cell complex generated from white pixels, with 0-cells (green dots) and 1-cells (red edges) overlaid on the binary image.**

**Step3:**



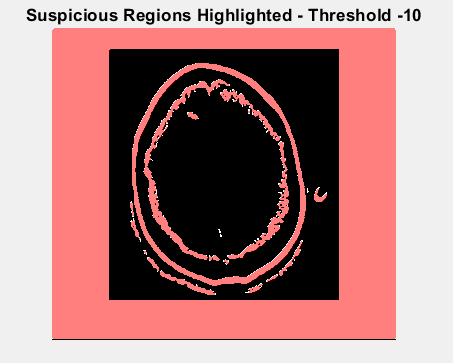
**Table 2 - Table of Forman-Ricci curvature for each edge in the graph**

**Step 4:**



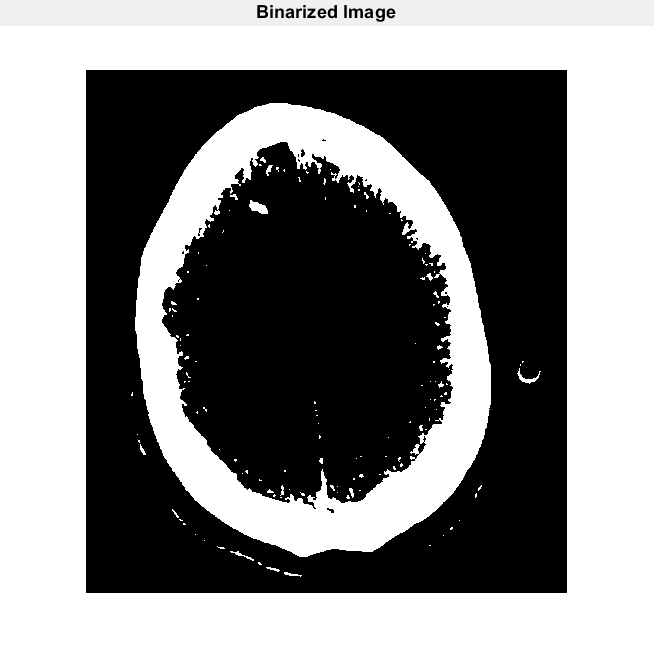
**Figure 13 -** **2D Forman-Ricci curvature map**

**Step5:**



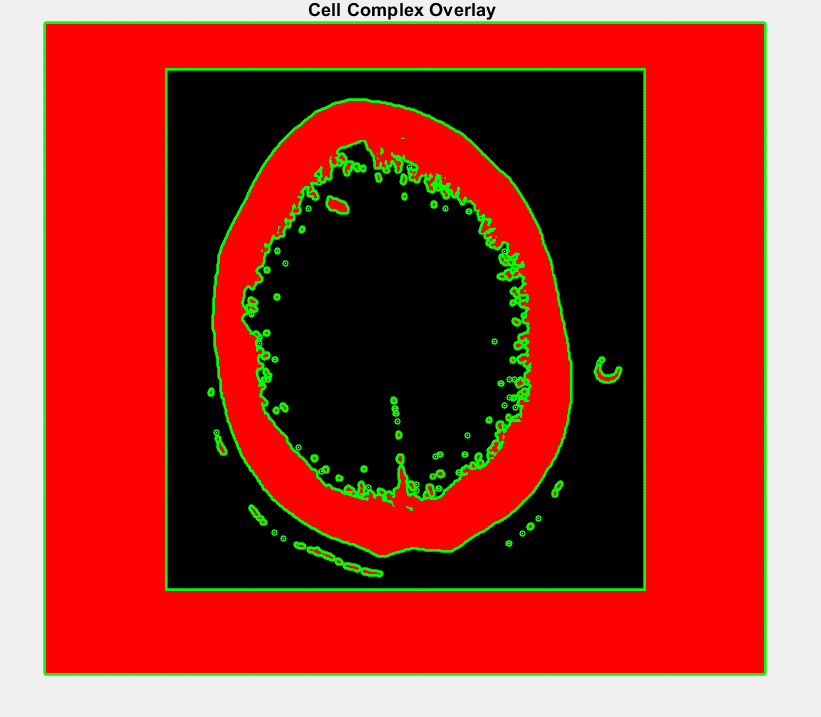
**Figure 14 - Suspicious regions (curvature < −10) shown in red**

**3.** **Brain CT scan 2:**

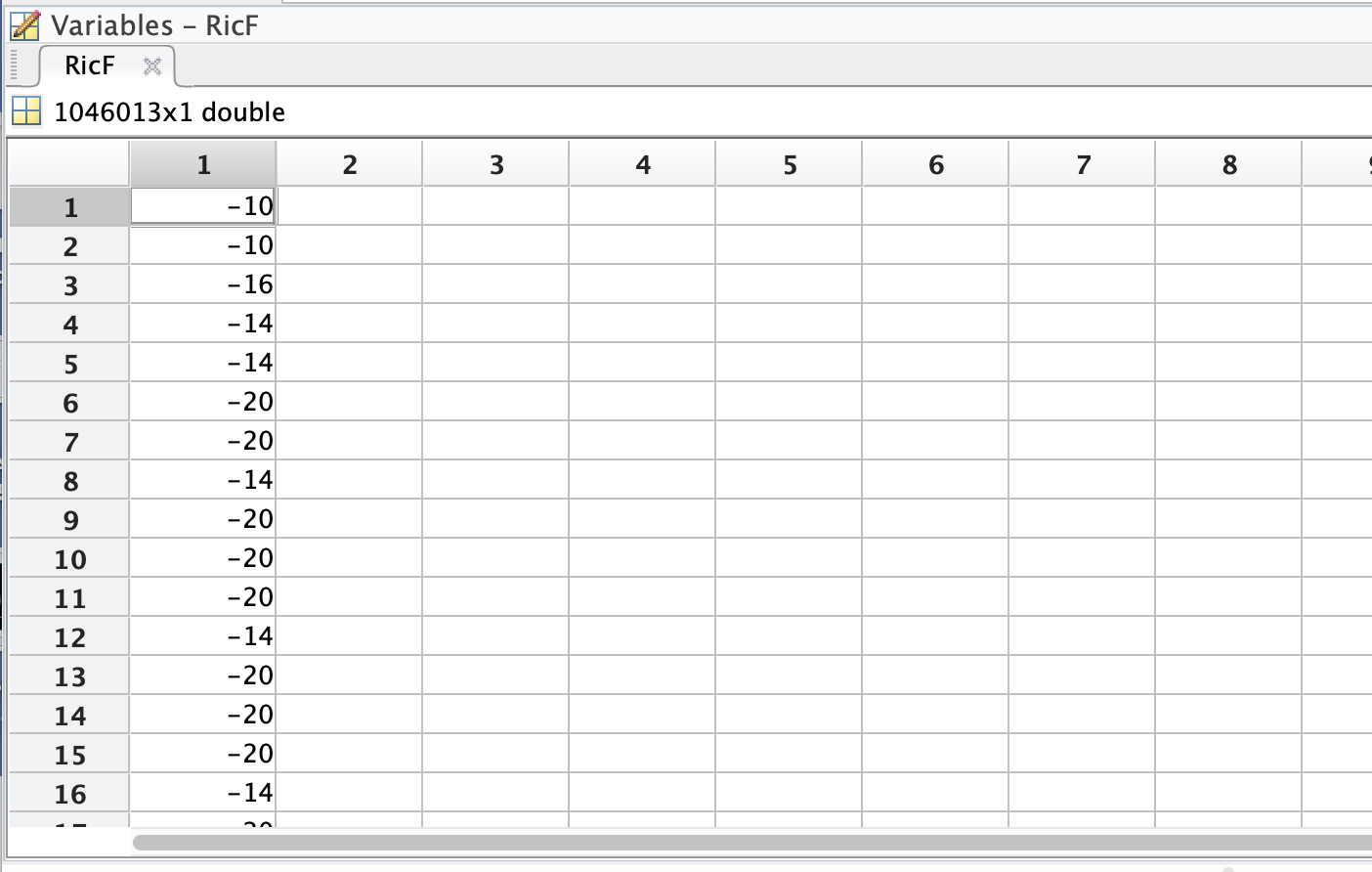
**Step 1:**

**Figure 15 -** **Binarized CT Image**

**Step 2:**

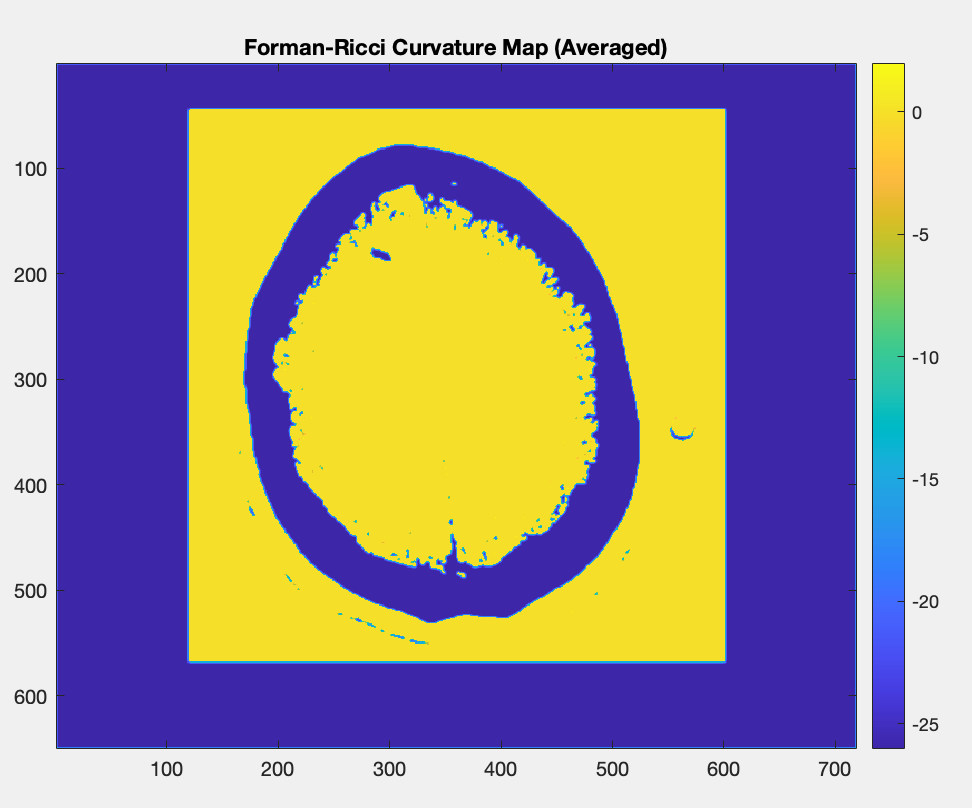


**Figure 16 -** **Cell complex generated from white pixels, with 0-cells (green dots) and 1-cells (red edges) overlaid on the binary image.**

**Step 3:**

**Table 3 -** **Table of Forman-Ricci curvature for each edge in the graph**

**Step 4:**



**Figure 17 - 2D Forman-Ricci curvature map**

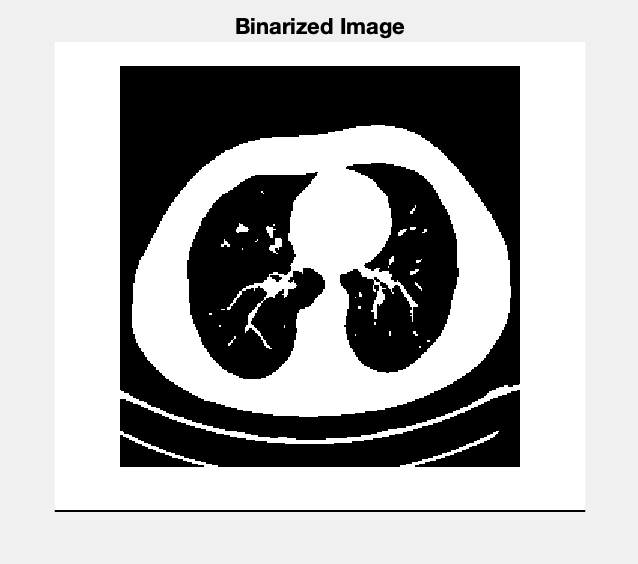
**Step 5:**

**Figure 18 - Suspicious regions (curvature < −10) shown in red**

****

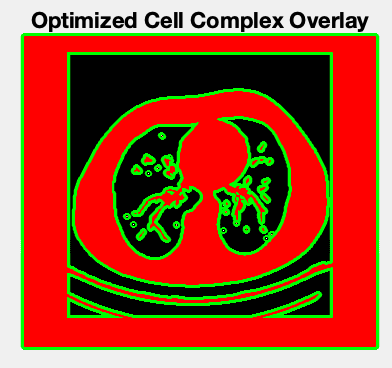
**4. Lungs CT scan:**

**Step1:**



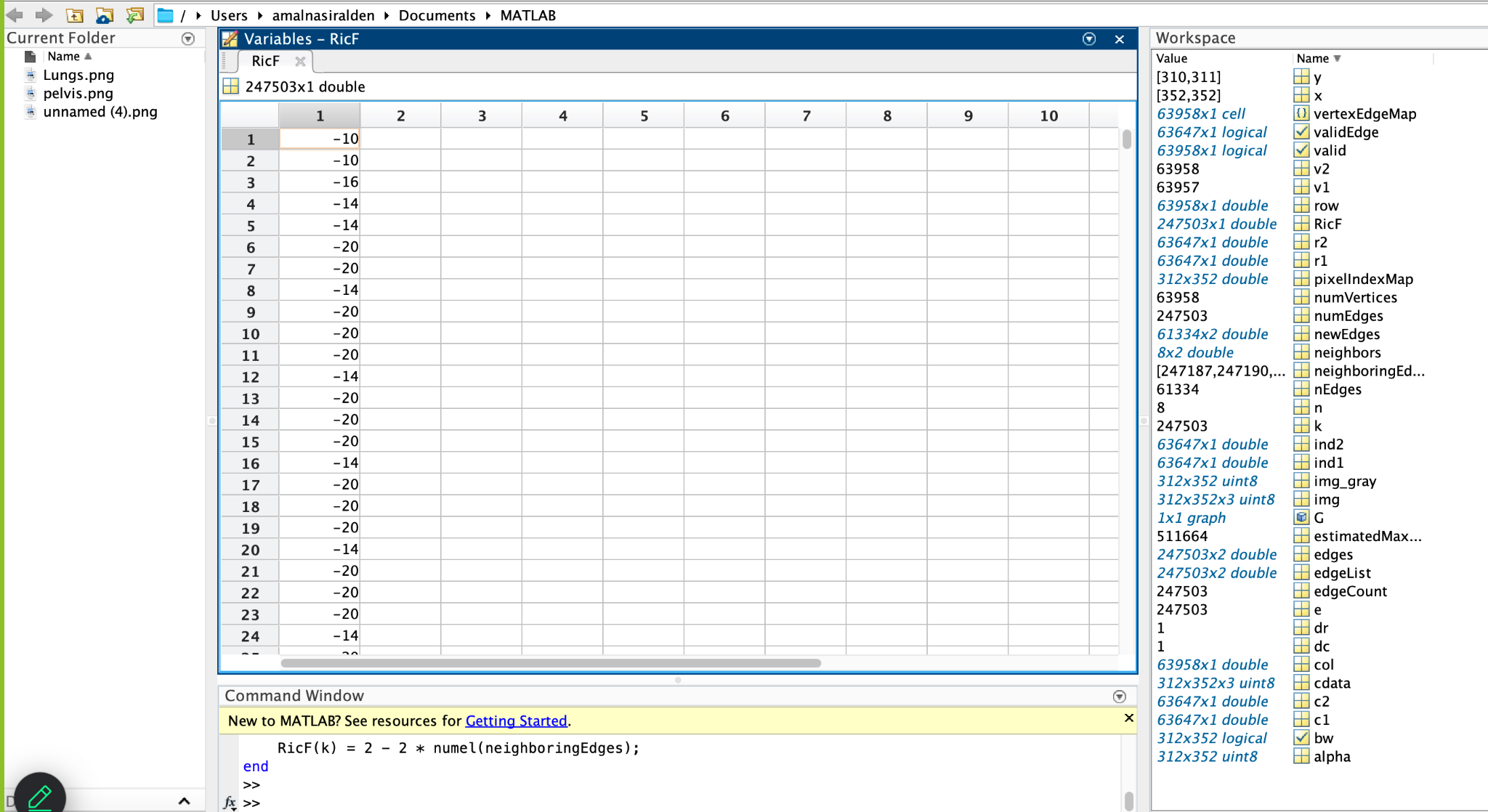
**Figure 19 - Binarized CT Image**

**Step 2:**



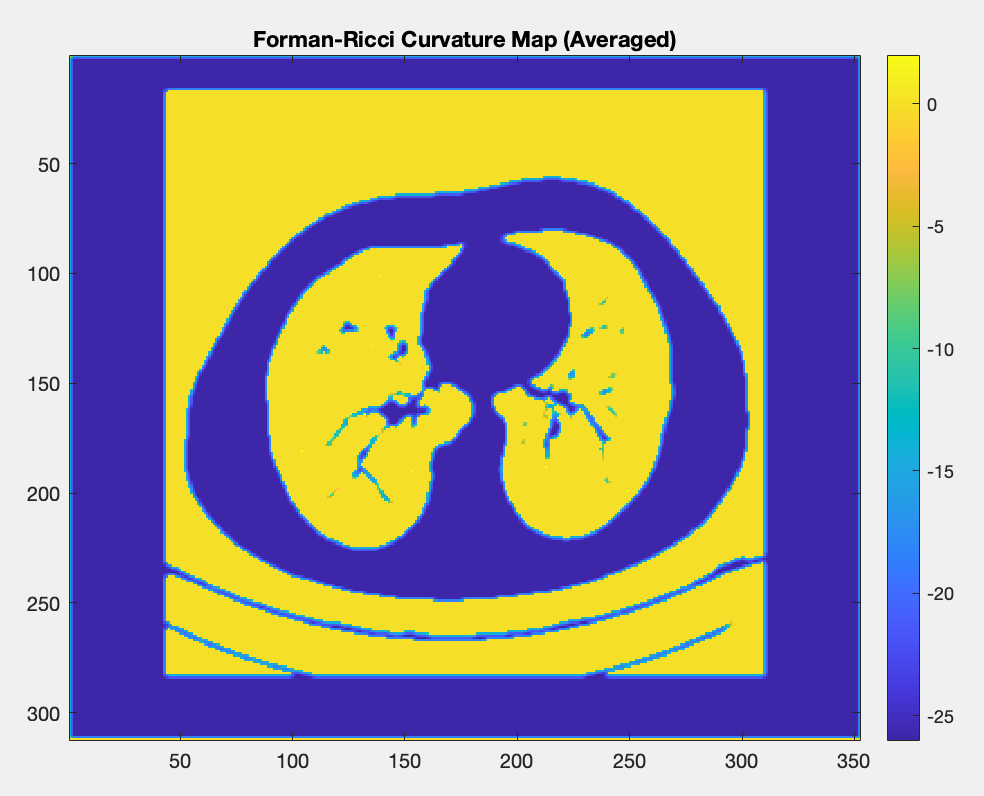
**Figure 20 - Cell complex generated from white pixels, with 0-cells (green dots) and 1-cells (red edges) overlaid on the binary image.**

**Step 3:**



**Table 4 - Table of Forman-Ricci curvature for each edge in the graph**

**Step 4:**



**Figure 21 - 2D Forman-Ricci curvature map**

**Step 5:**



**Figure 22 - Suspicious regions (curvature < −10) shown in red**

**5.3. Results analysis of MATLAB code:**

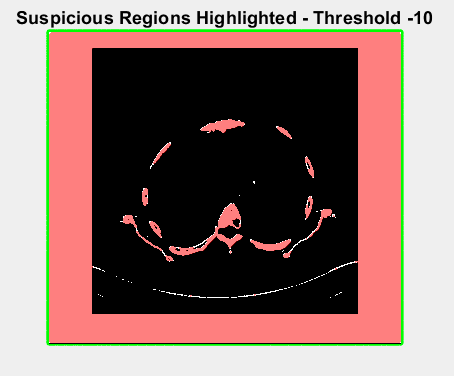
During the project, the Forman-Ricci Curvature method was applied to CT images of different human body organs to identify regions suspected as cancerous tumors. The method is based on a geometric processing of the image as a graph, where white pixels are connected to each other, and a curvature is computed for each edge between them. Very negative curvature values (below a threshold of -10) indicate distortions along the tissue boundary, which may suggest the presence of a cancerous tumor.

The entire process included five main stages: image binarization, construction of a cell complex, curvature computation for each edge, mapping the values to their location in the image, and finally a colored heatmap visualization highlighting in red the regions with extreme negative curvature. These regions were examined in order to draw medical conclusions.

To determine suspicion of cancer, three main criteria were examined in each of the images:  
 (1) The size of the red-marked area – the larger the area, the more significant the suspicion;  
 (2) Location – a red area appearing deep within the tissue, rather than just on the edges, is more suspicious;  
 (3) Boundary shape – when the red area is not symmetrical, has sharp edges, protrusions, or sudden breaks, there is an indication of pathological deformation.

Based on these criteria, we analyzed four CT images generated during the algorithm execution, and the possible clinical conclusions are presented below. It is important to note that the analysis was based solely on the resulting output images (i.e., regardless of file names or their order in the code) and focuses on the geometric pattern of the results.

**Detailed Analysis of the Four Images**

**  
Figure 23 – Final output for Pelvis CT scan**

**Observation**: Small and scattered red markings appear mainly along the boundary of the tissue, with almost no presence in the interior.

**Analysis**:  
 **1)** **Size** – Very small.  
 **2) Location** – Periphery only.  
 **3) Shape** – Symmetrical, narrow, corresponding to normal boundaries.

**Conclusion**: No pathological findings in this image. The tissue structure is considered normal according to the criteria.

תמונה שמכילה טקסט, צילום מסך, עיגול, גופן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

**Figure 24 – Final output for Brain CT scan 1**

**Observation**: Thin red lines appear along the edges of the tissue, mainly on the periphery, with no central focus.

**Analysis**:  
 **1) Size** – Small, only a few lines.  
 **2) Location** – Periphery only.  
 **3)** **Shape** – Relatively straight and symmetrical on both sides.

**Conclusion**: No suspicion of a tumor. The markings correspond to normal anatomical boundaries of a healthy organ.



**Figure 25 – Final output for Brain CT scan 2**

**Observation**: A clearly visible and relatively large red region appears on the right side of the tissue. The shape is non-symmetrical, sharply bounded, and localized.

**Analysis**:  
 **1) Size** – Large and prominent.  
 **2) Location** – Internal, not along the boundary.  
 **3) Shape** – Non-circular, with sharp angles.

**Conclusion**: The image indicates a high suspicion of a cancerous tumor. All three criteria are met.



**Figure 25 – Final output for Lungs CT scan**

**Observation**: A non-uniform, wide red region is observed at the center of the image. The shape is dispersed and non-symmetrical, including sharp boundaries.

**Analysis**:  
 **1) Size** – Medium to large.  
 **2)** **Location** – Internal, close to the tissue core.  
 **3) Shape** – Irregular and distorted.

**Conclusion**: There is a strong indication of suspicion for a cancerous condition. All three criteria are clearly met.

**Summary of outputs**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Image Position** | **Red Region Size** | **Location** | **Shape** | **Conclusion** |
| **1** | Very small | Peripheral | Normal | Normal |
| **2** | Small | Peripheral | Symmetrical | Normal |
| **3** | Large | Internal | Non-symmetrical | Suspicion of tumor |
| **4** | Large | Internal | Irregular | Suspicion of tumor |

**Table 5 - Summary of outputs**

**5.4.** **Python code implementation:**

import numpy as np

import matplotlib.pyplot as plt

from skimage import io, color, filters

import networkx as nx

from typing import Dict, Tuple

**Python code:**

# Step 1: Load and Binarize the Image

img = io.imread('pelvis.jpg') # Load the image

if img.ndim == 3:

img\_gray = color.rgb2gray(img) # Convert to grayscale if RGB

else:

img\_gray = img # Image is already grayscale

bw = img\_gray > filters.threshold\_otsu(img\_gray) # Binarize using Otsu's method

# Find white pixels (potential vertices in the graph)

row, col = np.nonzero(bw)

coords = np.column\_stack((row, col))

# Map each pixel coordinate to a unique index (for graph representation)

pixel\_index: Dict[Tuple[int, int], int] = {tuple(p): i for i, p in enumerate(coords)}

num\_vertices = len(coords)

#Step 2: Create Edges Based on 8-Connectivity

neighbor\_offsets = np.array([

[-1, -1], [-1, 0], [-1, 1],

[ 0, -1], [ 0, 1],

[ 1, -1], [ 1, 0], [ 1, 1]])

# Relative positions of 8-connected neighbors

edges = []

coord\_set = set(map(tuple, coords)) # Fast lookup of existing pixels

# Check each pixel’s 8 neighbors and add an edge if the neighbor exists

for p in coords:

for offset in neighbor\_offsets:

neighbor = tuple(p + offset)

if neighbor in coord\_set:

i1 = pixel\_index[tuple(p)]

i2 = pixel\_index[neighbor]

if i1 < i2: # Avoid duplicate edges

edges.append((i1, i2))

#Step 3: Create Graph Using NetworkX

G = nx.Graph()

G.add\_edges\_from(edges) # Build an undirected graph from the edges

# Step 4: Compute Forman-Ricci Curvature

RicF = np.zeros(len(G.edges)) # Array to store curvature values

edge\_to\_index = {e: i for i, e in enumerate(G.edges)} # Map edge to index

# Calculate curvature for each edge using a simple Forman-Ricci formula

for i, (u, v) in enumerate(G.edges):

neighbors\_u = set(G[u])

neighbors\_v = set(G[v])

neighbors\_u.discard(v)

neighbors\_v.discard(u)

total\_neighbors = neighbors\_u.union(neighbors\_v)

RicF[i] = 2 - 2 \* len(total\_neighbors)

# Step 5: Generate Curvature Map

curvature\_map = np.zeros(bw.shape) # Map to accumulate curvature values

curvature\_count = np.zeros(bw.shape, dtype=int) # Counter to compute averages

coord\_array = np.array(coords)

# For each edge, add its curvature to the midpoint pixel

for i, (u, v) in enumerate(G.edges):

p1 = coord\_array[u]

p2 = coord\_array[v]

midpoint = np.round((p1 + p2) / 2).astype(int)

r, c = midpoint

if 0 <= r < bw.shape[0] and 0 <= c < bw.shape[1]:

curvature\_map[r, c] += RicF[i]

curvature\_count[r, c] += 1

# Compute average curvature only where there were contributions

nonzero\_mask = curvature\_count > 0

curvature\_avg = np.zeros\_like(curvature\_map)

curvature\_avg[nonzero\_mask] = curvature\_map[nonzero\_mask] / curvature\_count[nonzero\_mask]

# Step 6: Visualization

plt.figure()

plt.imshow(curvature\_avg, cmap='hot') # Display the averaged curvature map

plt.colorbar()

plt.title('Forman-Ricci Curvature Map (Averaged)')

plt.axis('off')

plt.show()

# Step 7: Highlight Suspicious Regions

threshold = -10 # Curvature threshold to detect anomalies

suspicious\_mask = curvature\_map < threshold # Mark low-curvature areas

plt.figure()

plt.imshow(bw, cmap='gray') # Show the original binary image

red\_overlay = np.zeros((\*bw.shape, 3)) # Create an RGB overlay

red\_overlay[..., 0] = suspicious\_mask # Highlight in red (only red channel)

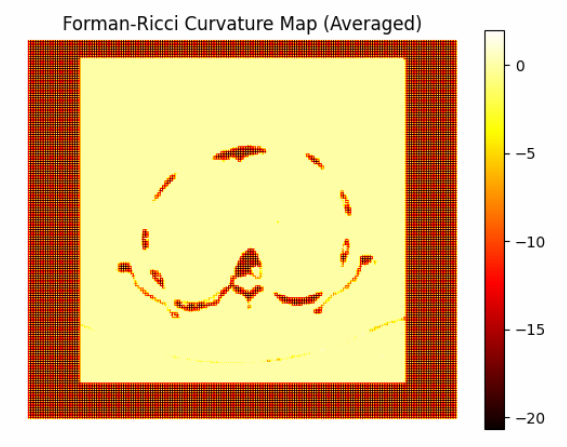
plt.imshow(red\_overlay, alpha=0.5) # Overlay with transparency

plt.title('Suspicious Regions Highlighted')

plt.axis('off')

plt.show()

**Output of pelvis CT scan:**

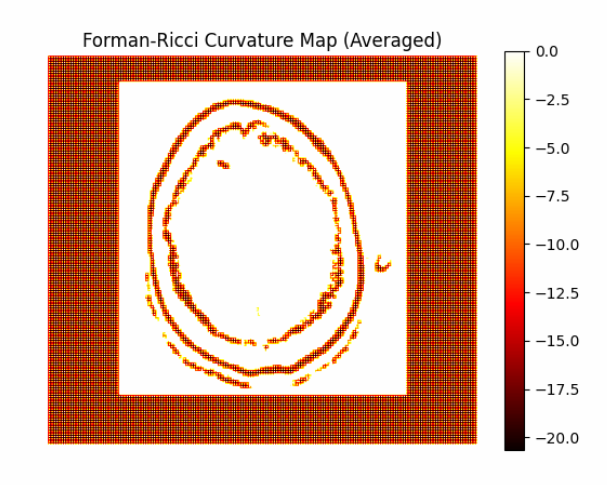
****

**Figure 26 - Averaged Forman-Ricci curvature heatmap of the pelvis binary image**

****

**Figure 27 - Suspicious regions (curvature < −10) highlighted in red**

**Output of Brain1 CT scan:**

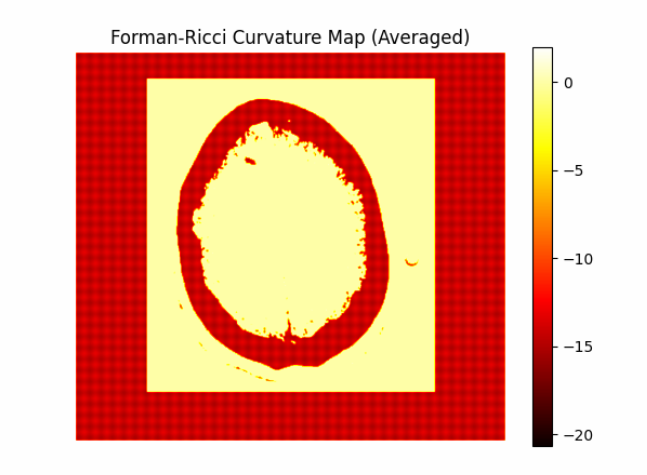
****

**Figure 28 - Averaged Forman-Ricci curvature heatmap of the brain1 binary image**

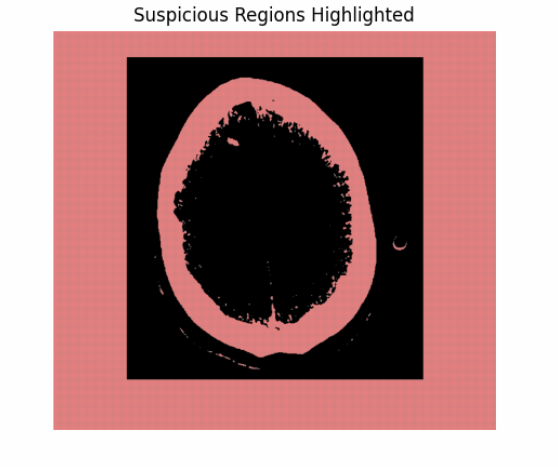
****

**Figure 29 - Suspicious regions (curvature < −10) highlighted in red**

**Output of Brain2 CT scan:**

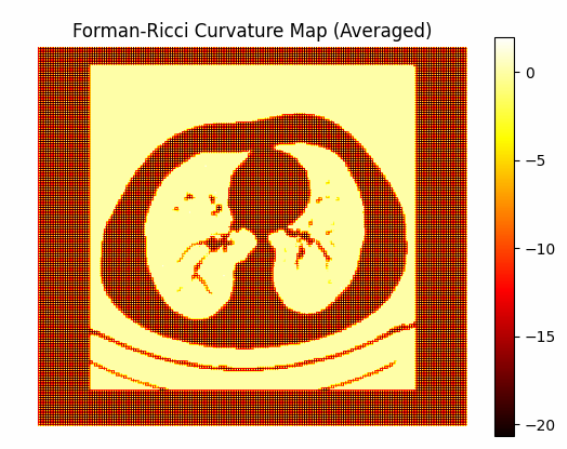
****

**Figure 28 - Averaged Forman-Ricci curvature heatmap of the brain1 binary image**

****

**Figure 29 - Suspicious regions (curvature < −10) highlighted in red**

**Output of Lungs CT scan:**

****

**Figure 28 - Averaged Forman-Ricci curvature heatmap of the brain1 binary image**

****

**Figure 29 - Suspicious regions (curvature < −10) highlighted in red**

**5.5. Results analysis for python code:**

**Curvature Map:** each pair visualizes the curvature levels of the graph edges extracted from the image, based on the Forman-Ricci curvature computation and its aggregation into a 2D map. Dark colors (e.g., black or deep red) indicate low or negative curvature values, which may correspond to sharp boundaries or structural transitions in the image.

**Suspicious Regions Highlighted:** shows a binary mask highlighting suspected regions (curvature < -10) in red, overlaid on the binary version of the original image.

The results obtained using MATLAB were found to be consistent with those computed in Python, confirming the reliability of the implementation across both platforms.

**6.** **Comparative Evaluation of MATLAB and Python Implementations:** To evaluate the effectiveness of the implementation across both environments, we compared the MATLAB and Python versions of the algorithm based on three core criteria: accuracy of tumor detection, visual clarity, and execution speed. The following table summarizes the comparative findings:

|  |  |  |
| --- | --- | --- |
| **Criterion** | **MATLAB** | **Python** |
| **Tumor Detection Accuracy** | High accuracy with focused marking. Suspicious regions are highlighted only where curvature is strongly negative. | Lower accuracy – highlights are overly broad, sometimes covering non-suspicious tissue as well. |
| **Visual Clarity of Output** | Clear and sharp visualization. Red highlights appear only where needed and blend well with the binary image. | Less sharp – red overlay dominates the image, making it harder to locate the exact suspicious areas. |
| **Execution Speed** | Slightly slower, especially during graph construction and rendering in MATLAB. | Noticeably faster. Python executed all processing and visualization steps more efficiently. |

**Table 6 - Comparison between MATLAB and Python**

Although the Python implementation demonstrated faster execution, the MATLAB version produced significantly more accurate and visually precise results. In the context of medical figures, where diagnostic accuracy is critical, MATLAB’s ability to provide focused and well-defined detection outweighs the minor delay in execution time. Therefore, despite Python’s computational efficiency, MATLAB is the preferred environment for this project due to its superior performance in correctly identifying tumor-suspected regions.

**7. Conclusions and Comments:**

In this project, a mathematical method was developed and implemented for detecting regions suspected to be cancerous tumors in CT images of various human organs, using the concept of Forman-Ricci curvature from discrete geometry. The method is based on representing each CT image as a graph, where white pixels represent vertices and the connections between them form edges. The Forman curvature is computed for each edge using a combinatorial formula, aiming to identify abnormal structural distortions along tissue boundaries.

The proposed algorithm follows five main stages:

1. Binarization of the image;
2. Construction of a discrete cell complex;
3. Computation of Forman curvature for each edge;
4. Mapping curvature values back onto the image;
5. Graphical visualization, where regions with highly negative curvature (lower than 10) are marked in red.

To identify regions suspected to be cancerous, three primary criteria were defined:

* Size of the red area: larger regions are considered more suspicious.
* Location of the region: regions located deeper within the tissue (not only at the edges) raise the level of suspicion.
* Shape of the boundary: asymmetric or irregularly shaped regions with sharp, jagged edges are considered abnormal.

The algorithm was applied to four CT images of different human organs and executed both in MATLAB and Python using the same input images. The outputs included curvature maps as well as suspicious region masks, where red indicates areas with curvature below the defined threshold.

The analysis yielded the following findings:

* In the second brain image (Image 3) and the lung image (Image 4), large, deep, and asymmetric regions were detected, clearly indicating a strong suspicion of cancer.
* In the pelvis image (Image 1) and the first brain image (Image 2), only thin peripheral lines or small symmetric regions appeared — therefore, no suspicion of tumors was found.

The results obtained in Python additionally included color heatmaps visualizing the geometric values across the image, which closely matched the corresponding suspicious masks.

The results computed in both MATLAB and Python were consistent with each other, confirming the reliability and accuracy of the implementation across different platforms.

In conclusion, the use of Forman-Ricci curvature analysis has demonstrated its capability to detect geometric irregularities in medical images that may correspond to pathological findings. The combination of rigorous mathematical analysis and graphical visualization enables the construction of a computational support tool for medical decision-making in the field of diagnostic imaging.

## 

## **8. Summary:**

Throughout the work on this project, we experienced a significant learning process both technically and personally. From the very beginning, we understood that this was a challenge combining advanced mathematical knowledge, image processing concepts, and programming across different platforms and we had to deepen our understanding in areas we were not previously fully familiar with.

Our work required us to thoroughly understand the concept of Forman-Ricci curvature, and how abstract mathematical ideas can be applied to a real-world problem in the medical field. We learned how to convert a CT image into a mathematical graph, how to compute discrete curvature values for edges, and how to interpret the results in a way that supports clinical conclusions.

We worked in parallel in both MATLAB and Python environments which required precision, step-by-step verification, and dealing with challenges such as differences in graphic libraries, image formats, and threshold calibration. Comparing results across both environments taught us the importance of consistency and reliability in algorithm development.

In addition, we learned the importance of proper planning, time management, and equitable division of tasks. We worked closely together, consulting with one another, sharing responsibilities, supporting each other, and showing willingness to learn from one another.

**9. Bibliography:**

[1] **Forman, R.** (2003). Bochner’s method for cell complexes and combinatorial Ricci curvature. *Discrete & Computational Geometry, 29*(3), 323–374.

[2] **Gowers, T. (Ed.).** (2008). *The Princeton companion to mathematics* (p. 352). Princeton University Press.

[3] **Hatcher, A.** (2002). *Algebraic topology*. Cambridge University Press.

[4] **Jost, J**. (2017). *Riemannian geometry and geometric analysis* (7th ed.). Springer.

[5] **Kobayashi, S., & Nomizu, K.** (1963). *Foundations of differential geometry* (Vol. 1). Wiley-Interscience.

[6] **Lee, J. M.** (2013). *Introduction to smooth manifolds* (2nd ed.). Springer.

[7] **Lee, J. M.** (2018). *Introduction to Riemannian manifolds* (2nd ed.). Springer.

[8] **Lundell, A. T., & Weingram, S.** (1969). *The topology of CW complexes*. Van Nostrand Reinhold.

[9] **Rabanser, S., Shchur, O., & Günnemann, S.** (2017). *Introduction to tensor decompositions and their applications in machine learning* [Preprint]. arXiv. <https://arxiv.org/pdf/1711.10781>

[10] **E. Saucan**, Curvature - Smooth, Piecewise-Linear and Metric, In: What is Geometry? , *Advanced Studies in Mathematics and Logic*, Polimetrica, Milano, 2006, 237-268. <https://www.ee.technion.ac.il/people/semil/curvature-book-chapter1.pdf>

[11] **E. Saucan and D. X. Gu,** Classical and Discrete Differential Geometry, Taylor & Francis/CRC Press, 2022.

[12] **Wang, R**. (n.d.). *Lecture notes in modern geometry*. University of California, Berkeley. <https://math.berkeley.edu/~ruiwang/pdf/161.pdf>

[13] **Zelditch, S.** (n.d.). *Eigenfunctions of the Laplacian of Riemannian manifolds Updated*. Northwestern University.