Computational and Differential Geometry
Homework 1

Professor: Nicolás Avilán Vargas, Ph.D.

Indicaciones

1. Fecha de entrega: 25 de febrero de 2025 hasta las 11:55 pm.
2. Único medio de entrega e-aulas.
3. Formato de entrega: **Un único** archivo **.ipynb** con códigos en python, descripciones de códigos y procesos, y respuestas a las preguntas. No entregar archivos comprimidos .zip o equivalentes.
4. Solo es permitido el uso de librerías "básicas" (numpy, matplotlib, seaborn, pandas, etc). En ningún caso será válida la solución lograda, total o parcialmente, por el uso de una librería especializada para resolver problemas de geometría computacional.
5. La tarea **debe** realizarse **individualmente**.
6. Cualquier tipo de fraude o plagio es causa de anulación directa de la evaluación y correspondiente proceso disciplinario.
7. Las entregas están sujetas a herramientas automatizadas de detección de plagio en códigos.
8. Las tareas no entregadas antes de la hora indicada tendrán calificación de 0.0.

Support each piece of code with a thorough explanation of its methods, techniques, functions, and tricks. Reference your search source (papers, books, tutorials, websites, etc.). Add any necessary bibliographical references or links.

1. (1 point). Generate a set of points in the plane using the following Python code

```
N = 10
np.random.seed(27)
points = [Point(np.random.randint(0,20), np.random.randint(0,20)) for _
                                        in range(N)]
```

   Next, calculate the point p_c, known as the centroid (or barycenter), which is given by the average of the coordinates of all the points. Then, evaluate the distance r from p_c to the farthest point. Remember, you found this point in the quiz. Plot a circle with center p_c and radius r, ensuring it contains all the points.

   Observe that this circle, which forms a convex region, indeed contains all the points. However, it is not necessarily the smallest possible circle that does so. The challenge is to find the smallest circle that contains all the points.

Implement the following instructions in Python to find the smallest circle containing all the given points

Input: Set of points $P = \{p_1, p_2, ..., p_n\}$, with $n > 3$.
Output: Smallest circle C that encloses all points in P

(a) Let **minCircle** the circle given centered in p_c with radius r.

(b) For each pair of points (p_i, p_j) in P where $i < j$:
   - Compute the circle having $\overline{p_i p_j}$ as the diameter.
   - Check if all points in P lie inside or on this circle.
   - If **true and the circle is smaller than minCircle**, update **minCircle**.

(c) For each triplet of points (p_i, p_j, p_k) where $i < j < k$:
   - Compute the unique circle passing through p_i, p_j, and p_k.
   - Check if all points in P lie inside or on this circle.
   - If **true and the circle is smaller than minCircle**, update **minCircle**.

(d) Return **minCircle**.

Plot the points and the **minCircle**.

2. (1 point). Let's consider some questions:
   - Which models can be implemented using the convex hull algorithm? Which systems can be described by such models? In these cases, what do the points and the hull represent?
   - We already know that the *sweep line* will be one of our best friends. What are the main and most common applications of the sweep line? What role does the *sweep line* play in each of these cases.

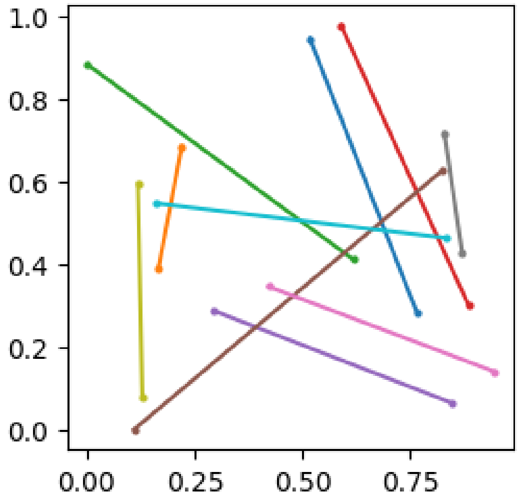
3. (1 point). Generate a set of points using the following code

```
np.random.seed(23)
X = np.random.rand(20, 2)
```

Find the convex hull associated with the generated points by implementing the Jarvis-March algorithm. Illustrate the process with plots, ensuring that the algorithm's steps are clearly visible.

4. (2 points). Implement **the sweep line algorithm** and apply it to the set of segments provided in the attached file (segmentos.csv). Identify intersection points and intersecting segments. Plot the process and the final result.

   Below is the plot of the given segments



Submit:
Upload to the platform **a signle .ipynb file** with answers, codes, descriptions and plots.