Problem 1: Spectral Geometry Of Shapes

1. Implement cotangent weight scheme for the Laplace-Beltrami operator approximations. Write a MATLAB function

```matlab
function [M,A] = cotangentLB(shape)

accepting as the input a struct with fields \{X, Y, Z, TRIV\}. X, Y, Z are the vertex coordinates, each of size \(N_{vertices} \times 1\), and TRIV is the triangulation, of size \(N_{triangles} \times 3\). The function should return the cotangent weight matrix \(M\) and the vertex area diagonal matrix \(A\).

\[
(M)_{ij} = \begin{cases} 
-\frac{1}{2} (\cot \alpha_{ij} + \cot \beta_{ij}) & i \neq j, j \in N_i \\
\sum_{k \in N_i} \frac{1}{2} (\cot \alpha_{ik} + \cot \beta_{ik}) & i = j \\
0 & \text{otherwise}
\end{cases}
\]
$$\begin{align*}
\text{In order to simplify the implementation, you can calculate the area of each vertex, } A_i, \text{ as } \frac{1}{3} \text{ of the sum of areas of its adjacent triangles (as opposite to calculating its Voronoi area, defined in the tutorial).}
\end{align*}$$

2. Use the function to calculate the first 5 eigenfunctions of the Laplace-Beltrami operator of each of the two isometric shapes in armadillos.mat. Show the obtained eigenfunctions of the two shapes using the trisurf function, by colouring the vertices with the eigenfunctions values. Are the eigenfunction approximately isometry invariant? Explain your answer. To calculate the eigendecomposition of the L-B operator, solve a generalized eigenvalue problem $Mv = \lambda Av$ (use the function eigs in Matlab).

3. Using 200 eigenfunctions of the laplace-beltrami operator, compute the Heat Kernel Signature of the two shapes and display the signatures for different times. Use 10 logarithmically sampled time instances - check what time instances work best for the shape you chose and report on them.

**Problem 2: MDS, Non-Linear Dimensionality Reduction**

Implement the dimensionality reduction algorithms: Isomap (MDS) and Laplacian Eigenmaps for the following datasets. Both these algorithms require that you first build a graph matrix from the data points. Define local distance between the datapoints (for eg: Euclidean which should work in most cases). Using the distance you defined, connect close data-points with edges and derive a connected graph. Each data-point should be connected with edges to 'k' closest data points. Estimate the value of 'k' that works best for each dataset. Compute the pairwise geodesic distances between each
pair of points using Dijkstra’s Shortest Path on this graph. In all cases let the final dimension be 2 so that you can visualize your data in a plane.

1. **Swiss-Roll**: Demonstrate the flattening of a swiss roll using the classical MDS scaling algorithm using pairwise geodesic distances and Laplacian Eigenmaps. Show both results side-by-side. Comment on the outputs of both algorithms by making a comparison. Visualize your results using the color index provided.

2. **MNIST digits**: Download the MNIST hand written digits training database. Consider only images of ’4’ and ’9’. Run Isomap (MDS) and Laplacian Eigenmaps on this image dataset. Color the points corresponding to ’4’ and ’9’ differently to show separation.