ECE / CS 6501: Geometry of Data **Due Fri 4/3**

Homework 3: Manifold Learning

Instructions: Submit a single Jupyter notebook (.ipynb) of your work to Canvas by 11:59pm on the due date. All code should be written in Python. **Be sure to show all the work involved in deriving your answers! If you just give a final answer without explanation, you may not receive credit for that question.**

You may discuss the concepts with your classmates, but write up the answers entirely on your own. Do not look at another student's answers, do not copy answers from the internet or other sources, and do not show your answers to anyone. **Remember to cite any sources you use, including any prompts given to AI chatbots.**

1. Install the `ellipse` package in Python (e.g., `pip install ellipse`). This package generates an image of a white ellipse on a black background, with a given position, orientation, and axes. The routine you will use is

   ```
   ellipse.get_ellipse(a, b, row, col, theta)
   ```

   - `a, b`: The two axes lengths of the ellipse (in pixels).
   - `row, col`: The x and y position of the ellipse (in pixels).
   - `theta`: The orientation of the ellipse (in radians).

   Use this function to generate a dataset of 1,000 images, using parameters: `a = 12, b = 6, theta = 0` and setting the `row, col` variables to be drawn from a continuous uniform distribution on $[16, 48] \times [16, 48]$. Along with the images, same the corresponding `row` and `col` parameters that generated them.

   Answer the following questions:

   (a) What is the dimensionality of the raw data (that is, the $\mathbb{R}^d$ that the data lives in)?

   (b) What do you expect the intrinsic dimensionality of the data to be (that is, the dimension of the "data manifold")? Explain your thinking why this is the case.

   (c) Perform a PCA of the data. How many non-zero eigenvalues do you find (excluding eigenvalues that are clearly numerical noise)?

   (d) Plot a 3D scatter plot of the data projected onto the top 3 principal components. Color the points based on their `row` parameter. Repeat the plot with colormap based on the `col` parameter for comparison. Is this what you expected the data to look like? Why or why not?

   Hint: You can use matplotlib's `scatter` function to plot points in 3D. But the plotly library's `plotly.express.scatter_3d` is a nicer option that lets you interactively rotate and zoom into the plot.

2. Run an isomap of the same ellipse image dataset. You may use a library to do this (`sklearn` has a good isomap implementation). Answer the following questions:

(a) Before looking at the isomap result, and based on what you observed in the PCA, do you expect that the data will embed isometrically into a Euclidean space? How many dimensions do you think will be needed to embed the data with isomap?

(b) Extract the matrix of estimated pairwise geodesic distances between the data points computed by isomap. Using this matrix, compute the B matrix (Gram matrix) that is used in the MDS step. Compute the eigenanalysis of this matrix. How many of the eigenvalues are positive (again, ignoring numbers relatively close to zero compared to the highest eigenvalue)? Looking at the eigenvalues, do you think that the data manifold is flat (zero curvature)? Why or why not?

(c) Plot the 2D embedding of the isomap result (i.e., using the top two eigenvectors from the final MDS). Color the points based on `row` parameter. Plot it again with color based on the `col` parameter.

(d) Change your embedding dimension for isomap to 3D, and plot a 3D scatter plot of the resulting embedding. Color the points based on either the `row` or `col` parameter. Does the shape of the embedded data correspond to your answer about the curvature and eigenvalues in part (b)?

3. Train a small convolutional autoencoder on the ellipse image dataset using the provided code in the file `Autoencoder.ipynb`. The dimension of the latent space is a parameter. Set this parameter to the intrinsic dimensionality of the ellipse data.

   (a) Build a **test** set of 100 ellipse images that is independent of the original 1,000 images used for training. Run your trained autoencoder on the test images to get reconstructed test images. Plot a 10x10 grid of the test images. Compare this to another 10x10 grid of the autoencoder reconstructed test images and another 10x10 grid of the PCA reconstructed images (use the same latent dimension for PCA as the autoencoder). Discuss what you see: how well do the PCA and autoencoder fit the test data?

   (b) Train another autoencoder, but with an increased latent space dimension (e.g., $d = 16$). Plot the reconstructed test images. How do they compare to the previous model outputs?

   (c) For the larger autoencoder, fit a PCA on the latent embeddings of the images (that is, run the encoder on each image). Plot the 3D scatter plot of the data in the top 3 principal components. Would you say that the embedded data lies on a flat manifold in the latent space?

   (d) Write a function to compute the rank of the Jacobian matrix of the decoder at any latent point. For both of your trained autoencoders compute the rank of the decoder's Jacobian matrix at each of the 100 test data points (at their encoded latent vectors). Summarize these rank values in two histogram plots (one for each autoencoder). Do these histograms indicate that the decoder is a (local) immersion? Do these results match what you would expect to see, given the intrinsic dimensionality of the data?

4. Redo everything in Question 2 and 3 above, but replace the ellipse dataset with the provided `teapot.pth` data. These are 2D color images of a teapot with random 3D orientation. You may realize that the intrinsic dimensionality of this data is greater than 2D. So, you can skip the 2D latent space models (that is, you don't need to report a 2D isomap or train the smaller 2D latent autoencoder).