- This homework is due at 11:59pm on Friday, February 28, 2025. **Please submit as a PDF** on Canvas homework 1 upload.
- There are 3 data files provided for the following questions, including, `Levine_matrix.csv`, `cell_graph.edgelist`, and `population_names_Levine_13dim.txt`. Instructions for how to use these data will be provided in each homework problem.
- Office hours are Monday at 1pm if you need help.
- Please feel free to work in groups, but please write up your own independent solution.
- You are welcome to use Python, Julia, or R here. All hints and sparse code are given for Python.
- You are welcome to write up your assignment using the `HW1_683-2024.tex` template, or write up the solutions in the method of your choice.
- This homework is worth 65 points total.

Problem 1

Nice Properties of the Graph Laplacian (25 Points Total)

- Here we will walk through computing the Graph Laplacian and exploring nice properties about it that relate to graph structure. You are welcome to paste the code that you write for each of these sub-problems.

Consider the following graph, $\mathcal{G}$ of 6 nodes. This graph has two components. We will see how what happens with our graph Laplacian in this situation.

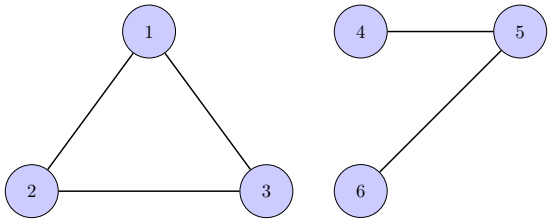

Figure 1: Graph, $\mathcal{G}$ with six nodes and two components.

1. (5 points) Create the adjacency matrix, $\mathbf{A}$, corresponding to $\mathcal{G}$. (Note that node i should correspond to the ith row of $\mathbf{A}$.
2. (5 points) We will go through some steps to write the code to compute the Graph Laplacian. The first matrix we need to define is the diagonal degree matrix, $\mathbf{D}$. Create the degree matrix ($\mathbf{D}$) for $\mathcal{G}$.
3. (5 points) Define the Laplacian matrix, $\mathbf{L}$ as $\mathbf{L} = \mathbf{D} - \mathbf{A}$.
4. (5 points) An eigenvector ($\mathbf{v} \in \mathbb{R}^6$) and its corresponding eigenvalue (λ) of the graph Laplacian, $\mathbf{L}$, must satisfy $\mathbf{L}\mathbf{v} = \lambda\mathbf{v}$. As this graph has two components, we will be able to find two unique eigenvectors ($\mathbf{v}_1$ and $\mathbf{v}_2$) that correspond to the zero eigenvalue, such that $\mathbf{L} \times \mathbf{v}_1 = 0 \times \mathbf{v}_1$ and $\mathbf{L} \times \mathbf{v}_2 = 0 \times \mathbf{v}_2$. Consider $\mathbf{v}_1 = [1, 1, 1, 0, 0, 0]^T$. You will notice that there are 1s in the first three entries, corresponding to the nodes that are together in a component. Check that $\mathbf{v}_1$ does indeed correspond to eigenvalue 0 and satisfies $\mathbf{L} \times \mathbf{v}_1 = 0 \times \mathbf{v}_1$.

   If you are using numpy, you can check by computing,

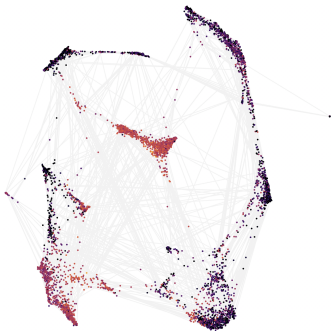

Figure 2: A visualization of the graph in homework problem 2.

```
L @ np.matrix([[1],[1],[1],[0],[0],[0]])
```

Discuss your findings.

5. (5 points) Given this pattern you just observed with $\mathbf{v}_1$, find $\mathbf{v}_2$. Show that your $\mathbf{v}_2$ satisfies $\mathbf{L} \times \mathbf{v}_2 = 0 \times \mathbf{v}_2$.

   You can fill in the following with your entries of $\mathbf{v}_2$.

   ```
   L @ np.matrix([[ ],[ ],[ ],[ ],[ ],[ ]])
   ```

Problem 2

(25 Points Total) **Practice with Single Cell Data**
We will consider data from a CyTOF experiment obtained from `http://flowrepository.org/id/FR-FCM-ZZPH`. Here, we are considering the expression of 13 different protein markers across a set of cells. It has already been pre-processed for you. From the entire dataset, 5,000 cells were sampled for further analysis. You can use the following accompanying data as follows.

- `Levine_matrix.csv` is the cell × marker matrix. Note that the last column is labels for the cells. Let's call this matrix, $\mathbf{X}$. **You should not use the last column (named label) for any kind of clustering.** Some cells are not labeled (hence are called NaN).
- `population_names_Levine_13dim.txt` maps the cell labels from the last column of $\mathbf{X}$ (number labels) to biologically-interpretable cell-type names.
- `cell_graph.edgelist` is an edgelist for a between-cell graph. We will call this, $\mathbf{G}$. Note that the nodes in $\mathbf{G}$ correspond to the rows in $\mathbf{X}$. So, node i maps to row i of $\mathbf{X}$, etc. Please make sure you interpret this as an undirected graph, where the adjacency matrix is symmetric.

1) **Clustering on cell × marker data** (5 points): Use a clustering algorithm of your choice to generate a cell-to-cluster partition for the cells, using the matrix, $\mathbf{X}$. Use normalized mutual information (NMI) to compute overlap between the true and predicted cell labels. **Note that because not all cells are labeled, you can compute this only based on the labeled cells.** Feel free to use an available implementation, such as, `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.normalized_mutual_info_score.html`.

2) **Graph Partitioning** (5 points): Use a graph clustering algorithm to partition **G** into clusters. Similar to part 1 (1) compute NMI between the labels obtained in graph partitioning and the true cell labels.

3) (5 points) Comment on any observations you observe between the quality of the partitions obtained clustering on **X** in comparison to partitioning **G**. Which approach do you think works better, using the original data, or the graph?

4) **Rare Cell-types** (5 points) `Plasmacytoid_DC_cells`, or pDCs (label 21) are a popular rare cell type, meaning many clustering algorithms will not be able to reliably find them. Report the number of distinct clusters where you found pDCs in both the clustering of **X** and in the partitioning of **G**.

5) **Cell Classification** (5 points) Select cells from **X** with the following labels, $\{11, 12, 17, 18\}$ and $\{1, 2, 3\}$. In general, cells with labels $\{11, 12, 17, 18\}$ are T-cells and cells with labels $\{1, 2, 3\}$ are monocytes. Convert this to a binary classification problem by labeling T-cells with 0 and monocytes with 1. Use your favorite classifier to predict the labels of these cells. Use an ROC curve to visualize the performance. Please be sure to use an appropriate training/testing or cross validation procedure. If the performance was not good, explain what could have gone wrong. If your performance is very good, can you identify features from **X** that were helpful in predicting labels?

Problem 3

(15 points total) **node2vec**

We will use the implementation of node2vec available in github, `https://github.com/eliorc/node2vec`.

1) (**Clustering on Node2Vec Features** (5 points)) First, use default parameters and follow the instructions in the README on the graph in `cell_graph.edgelist`. This will create a 128-dimensional vector for each node. Cluster the nodes based on these vectors and compare to the ground truth labels in the last column of `Levine_matrix.csv` using NMI. Does an embedding of the graph offer any apparent advantages in classifying cells? (hint: make sure the indices of the embedding correspond to the original nodes indices)

2) (**Parameters, part 1** (5 points)) Try a few different values for the number of dimensions `--dimensions`, such that some of them are less than 128, and some of them are more than 128. Cluster cells again with the embeddings obtained in different dimensions. Again, you can compute the NMI between the cluster assignments and the ground truth labels. Comment on some observations, and show a plot of NMI plotted against the number of dimensions used.

3) (**Parameters, part 2** (5 points)) Recall that the parameters p and q control the 'breadth' vs 'depth of the walk'. Choose one of these parameters to vary, and repeat the previous question using the default 128 dimensions, but varying values for either p or q. Comment on some observations, and show a plot of NMI against p or q (whichever one you chose).