Activity : Concluding Conquest of Arrakis! [6 marks]

In your lab, you implemented the LAG(Lisan-Al-Gaib) Algorithm. The algorithm begins with an initial guess estimate of the number of spice centers, denoted by K. Upon execution, the algorithm returns K spice centers and assigns N labels corresponding to N spicepoints in the dataset. The implementation of the algorithm is provided in the 'LAG.py' file. Check the function LAG() which is responsible for executing the algorithm.

Part 1: Mean-Intra Cluster Distance **[4 marks]**

Fill the TODO1 in mean_intra_cluster_distance() in 'submission.py' that takes in two arguments: a np.array of 2D spice points and their corresponding labels. Each spice point represents a location in a two-dimensional space, and each point is labeled to belong to a certain cluster.

For each spice point i, the mean intra-cluster distance a(i) needs to be calculated. The mean intra-cluster distance a(i) for a point i is the average distance from i to all other points j that belong to the same cluster as i.

Example:
Lets say we have points N1, N2, N3, N4,N5,N6 and corresponding labels are 0,1,1,0,1,2.
Let notation d(N1,N2) represent the euclidean distance between points N1 and N2.

```
a(0) = a(N1) = d(N1,N4)                      // a(0) corresponds to N1. Only N4 is in the same cluster
a(1) = a(N2) = [d(N2,N3) + d(N2,N5)]/2   // a(1) corresponds to N2. {N3, N5} also lie in the same cluster
a(2) = a(N3)  = [d(N3,N2)+d(N3,N5)]/2
a(3) = a(N4) = d(N4,N1)
a(4) = a(N5)= [d(N5,N2) + d(N5,N3)]/2
a(5) = a(N6) = 0       // a(5) corresponds to N6. No other point in the same cluster as N6

return np.array(a(0),a(1),a(2),a(3),a(4),a(5))
```

Note: You have to implement this function without loops(for/while) and only numpy functions. Using a single loop will fetch at max 1 mark for this section in the server side scoring.

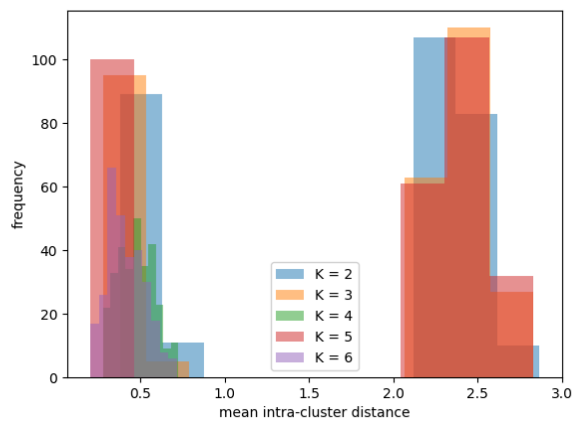
Part 2: Generate Plot **[2 marks]**

In this section, we plot the histogram of a(i) values obtained on runs of LAG for different values of K in the function generate_plot(). Arguments passed are data_path:str and max_K:int

Fill the TODO2. Given max_K, run the LAG algorithm for each **K = 2,3,....max_K** and pass the output labels obtained for each K into the function defined in Part 1. Plot the histogram of these intra_mean_cluster_distance values. Final plot must contain all (max_K - 1) histograms in one figure with different plot labels to each of them. Also set up various other annotations on the plot as shown in the figure(like the legend, xlabel, ylabel). Save the figure as **plot.png**.

Below is one possible output of plot.png when given data_path = spicepoints.txt and max_K = 6. It's fine to not match the color sequencing and distributions because everything is random. When plotting histogram, set the **transparency(alpha) to around 0.5** to be able to differentiate between different hist plots. You are allowed to use for loops here. Legend should have 5 characters. Eg: "K = 2" . There should be 1 space to the left and right of "=".

Note: When running the code for generate_plot you may see a QT wayland error. Please ignore it.



Note: In the entire problem, do not create new functions and implement your code only within the TODO sections. Part 2 will be evaluated independently from your implementation of Part 1. In case you haven't implemented Part 1, then you can return a randomly generated np.ndarray using np.random. The autograder for Part 2 only checks for the xlabel, ylabel, legend and hist plots. The rest of your code will be checked manually.

General Note: Your submission will be evaluated against hidden testcases, which may be different from those provided to you. So, hardcoding won't help.

Marking scheme:
On server side
Part 1:
1. Will be evaluated on 4 testcases. Each testcase = 1 mark. Total marks = 4.
2. If you use 1 for loop. Each testcase = 0.25 mark. Total marks = 1.
3. If you use more than 1 for loop => 0 marks

Part 2:
1. 1.5 marks for basic checks(xlabel, ylabel, (maxK-1) histograms in one plot, correct legend, correct transparency, different colors

2. 0.5 marks for correct code for correctly calling the functions LAG() and mean_intra_cluster_distance()