Genetics & Genomics – Genomics Project *fall 2024*
Title: *Where do my single cells come from?*

Background story: *Hannah is a PhD student that has a conference presentation coming up in a couple weeks about her single cell analysis of three tissues in mice. However, the Master student that was working with her forgot to label the cells according to their corresponding tissues. Luckily, nearby there was a group of EPFL SV students who took Genetics and Genomics and offered to help her. Are you ready to help Hannah?*

Exercise:
In this exercise we are working with a single cell transcriptomics dataset containing 54 samples (or single cells) from 3 types of mice tissues:

- *Fat brown adipose tissue cells*
- *Heart myocardial tissue cells*
- *Serotonergic neurons in the dorsal raphe nucleus*

Your objective is to cluster them into 3 groups and find the differentially expressed genes (DEGs) to identify their cell type, or tissue of origin. The main dataset is a count matrix containing genes as rows, and single cells or samples as columns.

1. Load and visualize the data.

a. Start by loading the tab-separated count matrix (*single_cell_count_matrix.txt*) and display its first rows and columns. Store it in a *pandas* Data Frame. Genes are represented by their unique Ensembl IDs, and can be used as row/indexes. What is the size of the matrix?

b. What type of data do you have, float or integer? Normalized or raw read counts?

c. Filtering: Remove the genes that are not expressed in any sample (the genes that have 0 counts for all samples)

Normalization: To check if library depth normalization is necessary, plot the sum of all the genes in a sample, for each sample, as a bar plot. The y-axis label of the plot should be in counts per million. Normalization is recommended if the cumulative counts vary significantly between samples, let's say >20%. If you deem that normalization is needed, we recommend that you use **CPM* (Count per million)** normalization. If you normalized, also plot the depth after normalization.

***Note:** CPM normalization is the number of raw reads, or counts, mapped to a gene divided, or scaled, by the cumulative number of sequencing reads in your sample multiplied by a million.

d. Visualization: Create a principal component analysis (PCA) plot to visualize your data. You can use the *sklearn.decomposition*.PCA package. Do you see well the 3 expected clusters? There should be one cluster per cell type?

e. Transformation: Perform a log2(1+x) transformation of your data and plot the PCA on the log-transformed data. Which PCA allows you to better segregate the data clusters? Continue to the next questions using the more appropriate Normalization/Transformation for your dataset.

f. Clustering: Identify the 3 groups of cells, or cell types, and label them accordingly (for example, by using a dictionary). You can use for example the **k-means** clustering*, or another clustering technique of your preference. Of note, you can run the clustering on the normalized data, or on the PCA data (using all the principal components or PCs). Plot a PCA with the cells colored according to their cluster label.

Note: You can use the **KMeans** function from the *sklearn.cluster*.KMeans package.

2. **Differential gene expression**

Now that we've identified our 3 clusters or groups of cells, we need to identify marker genes for each cluster. A marker gene is a gene that is up-regulated in a particular cell-type, as compared to the other groups of cells. In this part, we will compute Differentially Expressed (DE) genes, or DEGs, and store the results in 3x panda Data Frames (one for each cluster/group). Rows will be the filtered genes, and we will need at least 5 columns, which are detailed below.

For example: *columns = ['p-value', 'fdr', 'mean', 'mean_other', 'log2_fold_change']*

First, we recommend that you focus on **cluster 1,** which is the cluster you identified containing the most cells. Then, you will have to repeat the steps below for each of the remaining clusters (cluster 2 and cluster 3). You can also try to perform the operations below in parallel for all the clusters.

a. Perform a **two-tailed independent t-test**, or another appropriate pairwise comparison, on every gene to check if genes are differentially expressed in **one cluster** when **compared to all the other clusters combined.** Store these p-values in a column of the pandas Data Frame described earlier. How many genes are significant with a p-value<0.05?

Note: You can use the *scipy.stats.mannwhitneyu* function or the *scipy.stats.ttest_ind* function

For example:
```
t_value_1, p_value_1 = stats.mannwhitneyu(df.T.loc[G1_samples],
                                          df.T.loc[G2_samples+G3_samples], alternative='two-sided')
```

Warning: try to parallelize the pairwise comparison as shown in the example above. Pairwise comparisons that are done in a loop, gene-by-gene, are slow.

b. Now, adjust the p-values for multiple testing using the FDR correction (Benjamini-Hochberg) correction, and store the output in the correct column of the pandas Data Frame. These corrected p-values will now be used instead of the nominal p-values. Why do we need to do this correction?

Note: You can use the *statsmodels.stats.multitest.*multipletests package

c. Compute two **arithmetic** means for each gene:
   1) the average gene expression of **cluster 1**, or the cluster you are focusing at the time.
   2) the average normalized gene expression of all the other clusters

Then, place the results in their respective columns of the pandas Data Frame.

d. Now compute the fold change between the two groups and add it as a new column in your pandas Data Frame. We recommend you calulate the log2 fold change using this equation: ***log2(1+ mean) - log2(1+ mean_other)***

Note: A fold change (FC) is a ratio of means, describing the effect size of the gene expression difference.

e. Filter the pandas Data Frame and keep only **upregulated genes** that have FDR < 5% and fold change > 2 (log2_fold change>1). Sort by decreasing fold change.

f. **[BONUS—optional]** Display the results of the DE calculations as a Volcano Plot (log2 fold change in x-axis, -log10(FDR) in y-axis). In general, we color differentially expressed genes (both upregulated and

downregulated) differently from those that are not significantly expressed. You can also plot the lines of the FC and p-value thresholds. All genes should appear (not only the significant genes).

g. If you haven't done so already, **repeat** steps 2.a to 2.e (as well as the optional 2.f) for clusters 2 and 3.

3. Investigating top marker genes

a. Use the gene annotation file (*gene_name.txt*) to annotate the Ensembl IDs of the genes in the pandas Data Frame with the differentially expressed genes by adding a 'gene_name' column.

b. Look at the top marker genes for each group. Top marker genes are the differentially expressed genes with the highest fold-change. Then, create:
   1) A boxplot showing the gene expression of the cells in each of the three groups. Place the gene name in the title of the plot.
   2) A PCA with a gradient of colors that matches the expression of the gene in those cells.
   3) Make sure to have **at least** 1 top marker plotted for each clustered group. If the marker is repeated in between clusters, pick another marker that you haven't used yet. In total, you should have plotted at least 3 boxplots and 3 PCAs—one pair for each cluster.

   Note: If possible, for plotting, avoid repetition of similar code and build functions. This is highly recommended for all other parts of the code too.

c. Identify the cell type corresponding to each group: use the differentially expressed genes associated with each group to identify which cell type/tissue they come from. For this, use the Enrichr library to match your list of differentially expressed genes to a specific cell type.

   Note: Use this link to access the Enrichr library (***https://maayanlab.cloud/Enrichr/***)

d. **[BONUS—optional]** Create a clustered heatmap with the at least the 5 top marker genes for each cell type. Use the log2 fold change as the magnitude. You can reference the seaborn clustermap documentation (https://seaborn.pydata.org/generated/seaborn.clustermap.html)

e. Write a PDF summary, with visually convincing plots (for example, the plots generated in 3b, or 3d) that Hannah can use during her conference presentation to justify her tissue group assignments.

   Reference: the data used in this exercise is adapted from Dueck et al. 2015, Genome Biology (DOI 10.1186/s13059-015-0683-4)