Genetics & Genomics – Genomics Project Fall 2025

BIO-373: Genomics Practical

General instructions:

In this exercise you will be working with a single cell transcriptomics dataset containing 5'462 single cells from 2 different human cell types. Additionally, a subset of the cells were treated with a cytokine, to study its effect.

Your overall objective is to identify the two cell types present in the dataset and determine which cytokine was used to treat some of the cells. To achieve this objective you will pre-process the dataset, perform differential gene expression analysis across cell-types and conditions (treated and control), and interpret the results.

Input data:

- File name: "single_cell_count_matrix.tsv"
  The main dataset is a single-cell RNA-sequencing count matrix containing raw counts. Rows represent single cells, and columns represent genes.
- File name: "metadata.tsv"
  This file contains metadata, so additional information on cell type and condition, for every cell present in the dataset.

1. Load and visualize the data.

a. Load the data:
Start by loading the tab-separated count matrix (single_cell_count_matrix.tsv). Store it in a pandas Data Frame and display its first rows and columns. Single cells 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. Plot the **total counts per cell**:
- To check if library depth normalization is necessary, plot the sum of the counts for all the genes in a cell, for each cell, as a bar plot. The y-axis label of the plot should represent the total number of counts.
- To facilitate visualization, plot again the barplot, including only a subset of 50 cells.

d. CPM Normalization:
Normalization is recommended if the cumulative counts vary significantly between cells. If you deem that normalization is needed, we recommend that you use CPM* (Count per million) normalization. If you normalize, 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.

- What do the total counts per cell look like after normalization, compared to before the normalization?

e. **Log transformation**:
Another important step of single cell data analysis is log-transformation, which is performed on the normalized data to make cells more comparable.
Perform a log2(1+x) transformation of your data.

f. **Add metadata**:
To be able to identify what cells belong to cell types 1 and 2, and conditions 1 and 2, we need to look at the meta data. As you did in step 1a, load the tab-separated metadata file (metadata.tsv). Store it in a pandas Data Frame and display its first rows and columns.

As you can see, one column represents the cell type of each cell, and one column represents the condition. In the condition column, 'ctrl' stands for 'control', and 'stim' stands for 'stimulated'. Control cells are untreated, stimulated cells are the ones that were treated with the cytokine.

- How many cells are present for each cell type? How many for each condition? Print the answer.

g. **Perform Principal Component Analysis (PCA)**:
To visualize single cells in a lower dimensional space, perform a PCA on the normalized and log2-tranformed data. You can use the sklearn.decomposition.PCA package.
Save the results in a pandas data frame.

h. **Visualize the data in PCA space**:
We want to visualize how single cells distribute in the PCA space, and color them based on cell type and condition.

- From the metadata file loaded in point 1f, extract the cell types and assign a color to each one (you should obtain a pandas Series with matched single cell IDs and colors).
- Generate 2 PCA plots, one plotting PC1 and PC2, and one plotting PC2 and PC3. Color cells according to their cell types.

*Note: make sure to add a legend to the PCA plots, with a label for each color.

Repeat these last 2 steps, this time coloring cells according to their condition.

i. **Interpret the PCA results**:
- How many clusters can you observe on the PCA plot?
- Plotting PC1 and PC2 allows better discriminating between what groups of cells?
- Plotting PC2 and PC3 allows better discriminating between what groups of cells?

2. Differential gene expression across cell types

Now that you've visualized the single cells in the PCA space where we can recognize different groups, you need to identify marker genes for each cluster, to be able to assign an identity to them. A marker gene is a gene that is up-regulated in a particular group of cells (e.g. 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 panda data frames. We will then

interpret the DEGs to annotate our clusters of cells.

a. **Prepare an empty dataframe** that will store the results of the DGE analysis. In this data frame, rows will be the DE 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']

b. You will firstly perform the **DGE analysis across cell types**. Then, you'll repeat the analysis across conditions. Hence:

   - Perform a two-sided Mann-Whitney U test on every gene to check if genes are differentially expressed in cell type 1 when compared to the cell type 2.

   *Notes:
   To do this, you need to find a way to select cells from each cell type, and compare them.
   The DGE analysis should be performed on the cpm normalized matrix, *prior* to log-transformation.
   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.loc[cell_type_1_cells],
                                df.loc[cell_type_2_cells],
                                alternative='two-sided'
                        )
```

   - Store the resulting p-values in a column of the pandas Data Frame described earlier.
   - How many genes are significant with a p-value<0.05?

c. Now, **adjust the p-values** for multiple testing using the **Benjamini-Hochberg** (FDR) correction, and store the output in the correct column of the pandas Data Frame.

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

   - These corrected p-values will now be used instead of the nominal p-values. Why do we need to do this correction?
   - How many genes are significant with a FDR < 0.05?

d. Compute two **arithmetic means** for each gene:
   1) the average normalized gene expression of cell type 1 (or the cluster you are focusing at the time).
   2) the average normalized gene expression of the other cluster.
   Then, place the results in their respective columns of the pandas Data Frame.

   *Note: you should compute the mean on the log-transformed data.

e. **Compute the logFC**:
   Now compute the fold change between the two cell types and add it as a new column in your pandas Data Frame.

*Note: A fold change (FC) is a ratio of means, describing the *effect size* of the gene expression difference across two groups of cells. Having calculated the log2 mean expression in step 2d, you should calculate the fold change as the *difference* between the two log2 means.

f. **Filter DEGs**:
Now that all columns of the data frame are filled, filter the pandas Data Frame and keep only DEGs that have a FDR < 0.05 and abs(log2_fold change) > 1. Sort the genes by decreasing fold change.

*Note: because you performed DGE analysis on cell type 1 vs. cell type 2, the DEGs with the highest log2FC will be the ones up-regulated in cell type 1 (more highly expressed in cell type 1), with respect to cell type 2. Consequently, the DEGs with the lowest log2FC will be the ones up-regulated in cell type 2, (more highly expressed in cell type 2), with respect to cell type 1.

2b. Interpretation of the results
In this section you will interpret the resulting DEGs and annotate cell types. To do so you can rely on the web tool Enrichr, available at: https://maayanlab.cloud/Enrichr/.

a. Inspect **top differentially expressed genes**:
- Visualize the top 50 DEGs (according to log2FC) up-regulated in cell type 1
- Visualize the top 50 DEGs (according to log2FC) up-regulated in cell type 2

b. **Perform Gene Enrichment analysis using EnrichR:**
Gene Enrichment Analysis is a method to evaluate if a list of significant genes (e.g. the genes up-regulated in one group of cells), is over-represented among the genes belonging to a particular biological gene set (e.g. pathways, GO terms, transcription factor targets, diseases, etc.)

You can perform this analysis by copying the list of top 50 DEGs extracted in step 2b,a in the EnrichR web tool, and inspecting the results.
The aim in this case is to identify what is the identity of cell type 1 and cell type 2. For this, we are particularly interested in looking at the “GO Biological Process 2025” category (in “Ontologies”), and in the “CellMarker 2024” category (in “Cell Types”).

*Note: you need to perform this analysis using the up-regulated gene for each cell type.

- What do you think is the identity of Cell Type 1?
- What do you think is the identity of Cell Type 2?

3. Differential gene expression across conditions

a. Perform again all the steps of point 2, this time performing the differential gene expression analysis across conditions (stimulated and control). You can recycle the code already used,

changing only the necessary.

b. Repeat also the steps of point 2b, to interpret the results of the differential gene expression analysis, with minor changes:
- Visualize the top 50 DEGs (according to log2FC) up-regulated in stimulated cells
- Perform Gene Enrichment analysis using EnrichR. This time we are particularly interested in looking at the categories “Reactome Pathways 2024” and “WikiPathways 2024 Human” (in “Pathways”) of EnrichR.
- According to your results, what were the stimulated cells treated with?

To conclude:

Write a PDF summary, including plots and answers to questions. You can for example save a jupyter notebook in .pdf. For the analyses done using the EnrichR web tool, save the plots or tables that you used to justify your answers, and make sure to include them in the report, or attach them as separated files.