Theory

1. Assume we have our data , where the th sample has features. Let be the first principal component. Then we have learned in the class is the projection of the samples on .

1.1 prove , where . is the identity matrix and is a dimensional vector with all ones as its elements. means the variance. Hint: unitize the definition of variance and write out everything using matrix formula.

1.2 show is the covariance matrix between features.

2. Nonnegative Matrix Factorization (NMF) can be formulated as

,

where . Derive the algorithm to solve NMF by integrative coordinate descent and projected gradient decent.

Programing

Description:

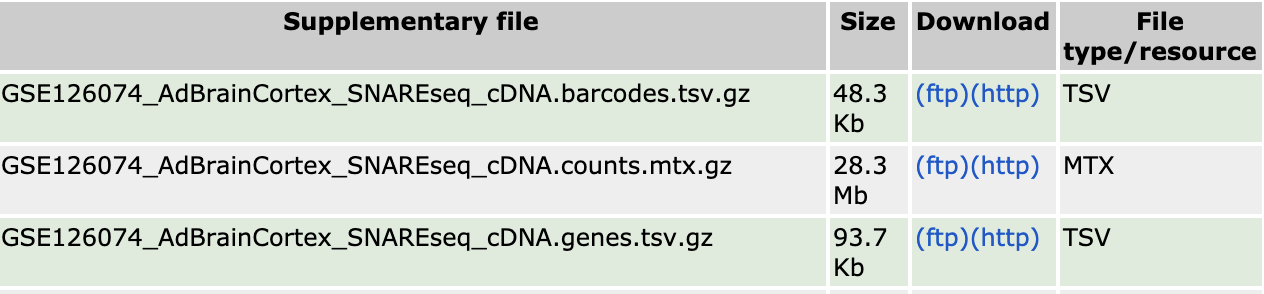
We are going to use what we learned in the class to analyze real-world single-cell RNA-seq data. You will need to use the dimension reduction techniques introduced in the class. And you will need to use clustering algorithms to cluster the data. In the end, you will need to visualize the clustering results.

Write everything in a jupyter notebook. Write your code in the “code” section in jupyter notebook and your answers to the questions in the “Markdown” section. You can use python package to do the following tasks.

1. Data preparation:

1.1 Download the single-cell RNA-seq data from the GEO website: <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE126074>

You need to download the following three files



Hint: you can use scanpy.read\_mtx to read in the data.

2. Dimension reduction:

Use the built-in pca function in scanpy to do dimension reduction on the scRNA-seq data. Assuming the raw scRNA-seq data is in a matrix of the format data , where is the number of genes and is the number cells.

2.1 Use PCA to reduce to . Please select to make sure 90% variance are preserved.

2.2 After PCA, becomes to . Please project back to the original space and get .

3. Clustering:

Find clusters using , , and , respectively.

4. Visualization:

Using Umap to plot three clustering results from using , , and .

5. Marker genes:

5.1 Using and the clustering result, we can identify marker genes for each cluster. Use this scanpy code (more details can be found here <https://scanpy-tutorials.readthedocs.io/en/latest/pbmc3k.html#Finding-marker-genes> )

sc.tl.rank\_genes\_groups(adata, 'leiden', method='wilcoxon')

sc.pl.rank\_genes\_groups(adata, n\_genes=25, sharey=False)

5.2 Using and the clustering result, we can identify marker genes for each cluster.

5.3 Compare the top 20 marker genes identified from using and . Point out the difference for each cluster.