Seurat - Guided Clustering/pbmc3k Tutorial (outputs not shown)

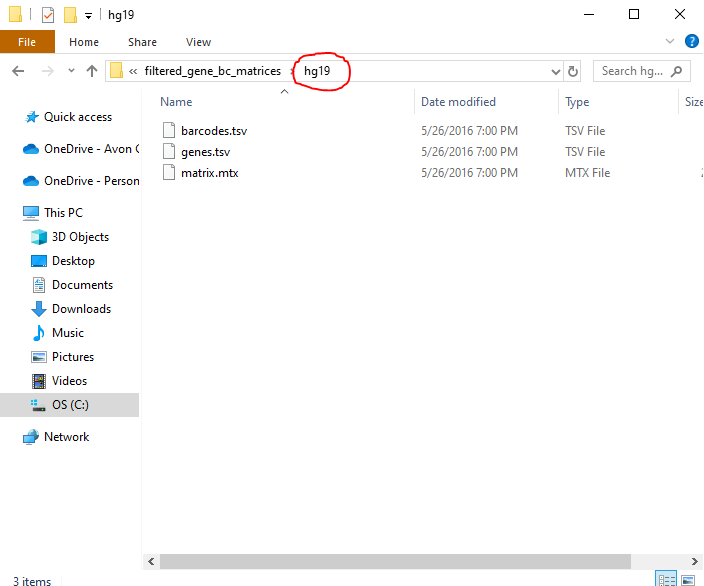
Find the original site here: <https://satijalab.org/seurat/v3.1/pbmc3k_tutorial.html>

First download [this](https://s3-us-west-2.amazonaws.com/10x.files/samples/cell/pbmc3k/pbmc3k_filtered_gene_bc_matrices.tar.gz) and copy it into a safe location

To begin you should load libraries into R. These libraries are created by a third party and make it very easy to get a jumpstart on your code.



After doing this you should copy your file address



The method I found being the easiest is by right clicking what is circled in red and clicking copy address. The only issue is that when you do this you get backslashes instead of forward slashed. I recommend copying the file address into word or a text and editing the slashes like this

C:\Lab08-Data-Wrangling-scRNAseq\pbmc3k\_filtered\_gene\_bc\_matrices\filtered\_gene\_bc\_matrices\hg19

To

C:/Lab08-Data-Wrangling-scRNAseq/pbmc3k\_filtered\_gene\_bc\_matrices/filtered\_gene\_bc\_matrices/hg19

Utilize this whenever it says Your file address and copy and paste it to replace the text there

Next you should load the PBMC dataset. PBMC stands for Peripheral Blood Mononuclear Cells.



Example:



The Read10X function used above helps you by reading the output of the [cellranger](https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/what-is-cell-ranger)pipeline. Then it will return a unique molecular identified or UMI count matrix

After loading the dataset you should initialize the Seurat object with the raw (non-normalized data)



Error:



According to [github](https://github.com/satijalab/seurat/issues/1212) I should be using backticks. This isn’t working for me

*Shown from the second line of code*

Issue: Resolved

**Ignore the warning**



This tells you what the dataset contains

To examine a few genes from the first **thirty** cells do this



Whenever you see “.” These values represent zeroes. In other words it tells you that there are no molecules detected. Due to most values in a scRNA-seq matrix being zeros sparse-matrix representation is used more frequently. This saves time and memory for Drop-seq/inDrop/10x data.

The following are various methods to check the sizes of the data







How to stash QC stats.



The [[adds columns to the object meta data. The MT- helps you use the set of genes starting with MT – as a set of mitochondrial genes. You can calculate mitochondrial QC metrics using the PercentageFeatureSet function. This function calculates the percentage of counts originating from a set of features.

The number of unique genes and the total amount of molecules are calculated when you do CreateSeuratObject

To show the QC metrics for the first 5 cells do the following



To visualize the QC metrics as a violin plot do the following



The following function is commonly used to visualize feature-feature relationships, but it can also be used for other things calculated from the object i.e. columns in object metadata, PC scores etc.





After you remove all of the unwanted cells from the dataset, you want to normalize the data. Usually “LogNormalize” is used. This “normalizes the feature expression measurements for each cell by the total expression, multiplies this by a scale factor (10,000 by default), and log-transforms the result” (satijalab). These normalized values are stored in “pbmc[["RNA"]]@data”



We can do the same thing by doing the following:



But for clarity the longer piece of code is typed.

Now we must calculate a subset of features that exhibit high cell-to-cell variation in the dataset (i.e, they are highly expressed in some cells, and lowly expressed in others).

Off topic: Click [this](https://www.biorxiv.org/content/biorxiv/early/2018/11/02/460147.full.pdf) if you are curious to read about the creator’s process for Seurat.



The *top10* will identify the 10 most highly variable genes. The last section of code for creating the plots, plots variable features with and without labels.

Next we scale the data. This is also known as applying linear transformation. For this we must use a function called “ScaleData” which does the following (satijalab):

* Shifts the expression of each gene, so that the mean expression across cells is 0
* Scales the expression of each gene, so that the variance across cells is 1
  + This step gives equal weight in downstream analyses, so that highly-expressed genes do not dominate
* The results of this are stored in pbmc[["RNA"]]@scale.data

Code shown below:



You may be thinking that this takes too long.

You can just do



This will omit the “features” argument. Through this line of code the PCA and clustering (analysis methods) results will be unaffected. But the heatmaps (“DoHeatmap” function) will require genes for it to be scaled.

After that we will perform PCA on the scaled data. The previously determined variable features are usually used as an input, but through the use of the “features” argument you can choose another subset.



The second line of code beginning with “print” examines and visualizes the PCA results in a multitude of ways.

The “DimHeatmap” helps PCs to include for further downstream analyses. This is a valuable tool for exploring correlated feature sets.



Interesting read: <https://www.cell.com/fulltext/S0092-8674(15)00549-8>

Now we must determine the ‘dimensionality’ of the dataset

The “JackStrawPlot” function helps to provide a visualization tool for comparing the p values for each PC

With a dashed line aka a uniform distribution.

The “ElbowPlot” is another heuristic method. For reading this one we look at where the “elbow is located” in this case it is around PC9-10 which means that the true signal is captured in the first 10 PC’s. This method is based on the percentage of variance.



First line of code takes a while!

Also ignore the warning.

Now we cluster the various cells. We construct a KNN graph based Euclidean distance (the geometry of flat or two-dimensional spaces) in PCA space using the “FindNeighbors” function.

To cluster these cells we use the “FindClusters” function, This means increased values lead to a greater number of clusters. Satijalab suggests that “0.4-1.2 typically returns good results for single-cell datasets of around 3K cells. Optimal resolution often increases for larger datasets.” These cluster can be found through the “Idents” function.



There are many nonlinear techniques that are provided by Seurat. These include tSNE and UMAP to explore datasets.



The last line of code shows how to save the object to make it easy to load it back.

Next we find markers that define clusters via differential expression. The function “FindAllMarkers” helps us by automating the process for all clusters. Although this is true you can slao test groups of clusters against each other or other cells.

\*Descriptions shown below the code”



The first line of code finds all markers of cluster 1

The second line of code finds all markers distinguishing cluster 5 from clusters 0 and 3

The third and fourth lines of code finds markers for every cluster compared to all remaining cells, report only the positive ones

The “FeaturePlot” function visualizes feature expression on a tSNE or PCA plot.

The “DoHeatMap” creates a heatmap for the given cells and features.



In the above code we plot the top 20 markers unless less than 20. In the case that it is less than 20 we plot all markers. This is done for each cluster.

Finally we must assign cell type identity to clusters.

