Single-cell Panoramic View Clustering (PanoView)

Manual

Contents

1. Introduction

Single-cell transcriptomics has opened new avenues for investigating biological functions at the cellular level. One of the important tasks in analyzing single cell transcriptomics data is to classify cell subpopulations based on the gene expression profile. Current approaches are often sensitive to the input parameters and have difficulty to deal with cell types with different densities. Here, we present *PanoView*, an iterative PCA-based method integrated with a novel density-based clustering, ordering local maximum by convex hull (OLMC) algorithm, to identify cell subpopulations for single-cell RNA-sequencing. The key feature of *PanoView* is to find cell clusters from different sets of variable genes in an iterative fashion and to estimate optimal parameters based on input datasets

2. Installation

PanoView is a python module that uses other common python libraries such as *numpy, scipy, pandas, scikit-learn*, etc. Prior to installing *PanoView* from Github repository, please make sure that *Git* is properly installed or go to https://git-scm.com/ for the installation of *Git*.

To install *PanoView* at your local computer, open your command prompt and type the following

```
pip install git+https://github.com/mhu10/scPanoView.git#egg=scPanoView
```

It will install all the required python libraries for executing *PanoView*. To test the installation of *PanoView*, open the python interpreter or your preferred IDE (Spyder, PyCharm, Jupyter, etc.) and type the following

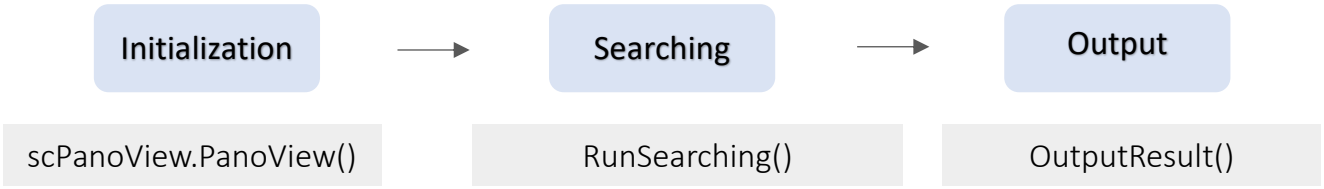
```
from PanoramicView import scPanoView
```

There should not be any error message popping out.

Before proceeding to the tutorial section, please download *"ExamplePollen.zip"* and unzip it into your python working directory.

3. Tutorial

There are three steps to execute *PanoView* algorithm in python. First, initializing expression matrix. Second, searching cell clusters. Third, outputting results of *PanoView* algorithm.



The output files include “Cell_Membership.csv” that stores clustering membership of cells, and “PanoView_output.png” that consists of one cluster hierarchy and two TSNE plots.

3.1 Input expression matrix and initialization

The format of the expression matrix should be comma-separated values (csv). The rows are the genes and the columns are the cells. For demonstration, we use expression data (i.e. “PollenRaw.csv”) from [Pollen et al, Nature Biotechnology 2014] as the input matrix. Please make sure that “PollenRaw.csv” is at your python working directory.

First, import *PanoView* module by `from PanoramicView import scPanoView`. Second, choose a job name (“Pollen” in this tutorial) and initialize *PanoView* by inputting the filename of the expression matrix(“PollenRaw”). You may also input annotation of cells (“PollenAnnotation”) if it is available. Here the annotation file shows 11 different populations described in the Pollen et al. (Note: you need to do the initialization whenever starting a new job)

```
[1] from PanoramicView import scPanoView
[2] Pollen = scPanoView.PanoView( filename = 'PollenRaw', annotation = 'PollenAnnotation')
```

You may check the raw expression values of the first three genes/cells from the input matrix. The raw expression value is stored at `Pollen.raw_exp`

```
[3] Pollen.raw_exp.iloc[:3,:3]
```

Gene_Symbol	Hi_2338_1	Hi_2338_2	Hi_2338_3
A1BG	9.08	0.00	0.00
A1BG-AS1	0.00	0.00	3.47
A1CF	0.00	0.05	0.00

3.2 Generate clusters

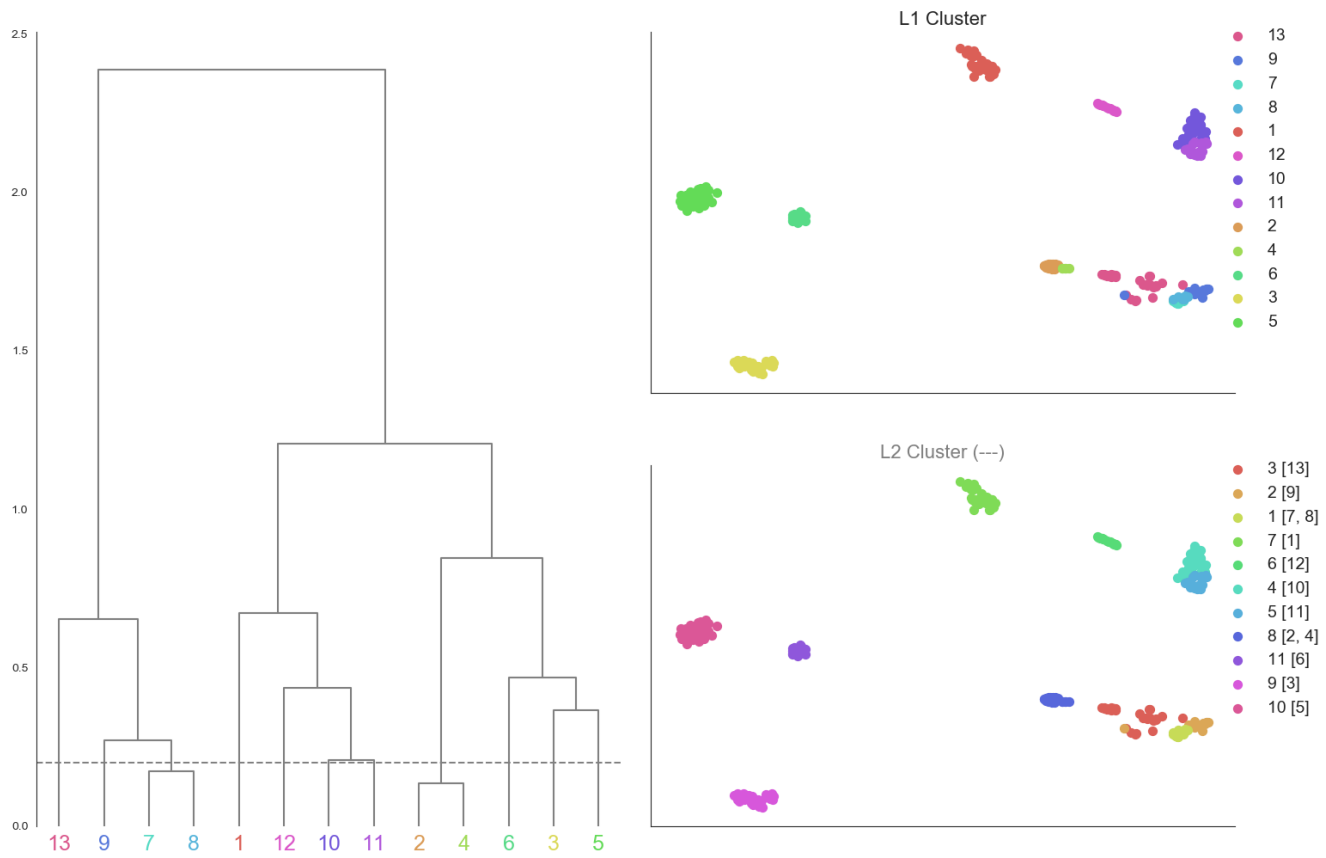
Use `Pollen.RunSearching( )` to find single-cell clusters. It will show the progress of algorithm in percentage which indicates how many cells are being analyzed. It will display "Clusters generated" once the searching is finished.

```
[4] Pollen.RunSearching( )
Percentage of cells being analyzed:  14%
Percentage of cells being analyzed:  20%
  :
Percentage of cells being analyzed:  74%
Percentage of cells being analyzed:  86%
Clusters generated
```

3.3 Output result

Use `Pollen.OutputResult()` to output the result of *PanoView* at your working directory. The result includes a table ("Cell_Membership.csv") storing clustering membership of cells and a figure ("PanoView_output.png") that consists of one cluster hierarchy and two TSNE plots.

```
[5] Pollen.OutputResult()
Output   Cell_Membership.csv
Output   PanoView_output.png
```



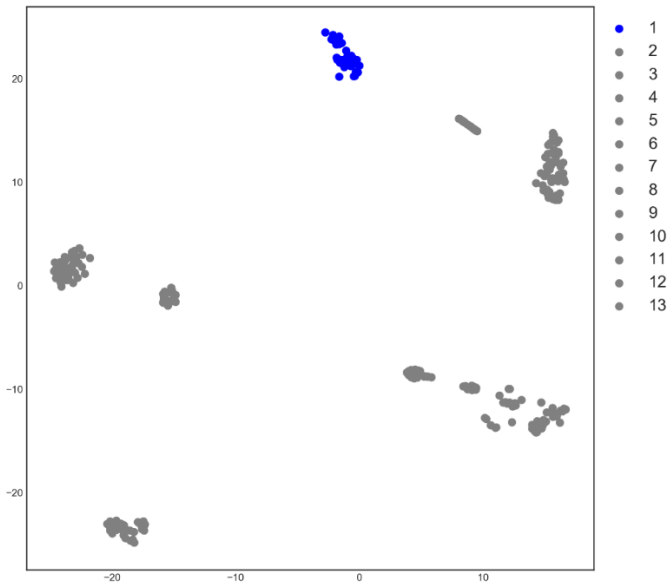
As shown in the figure, *PanoView* identified 13 clusters (Level-1 clusters) and used hierarchical tree to show the similarity of identified clusters. One intuitive parameter in *PanoView* is the height (fclust_dis = 0.2) of the hierarchical tree that would further merge nearby similar clusters. In this tutorial, the default value of 0.2 (marked by dash line) produced a total of 11 Level-2 clusters that merged cluster#2 and cluster#4 as cluster 8, and cluster#7 and cluster#8 as cluster 1.

3.4 Visualization

PanoView provides some simple ways for visualizing the identified single-cell clusters.

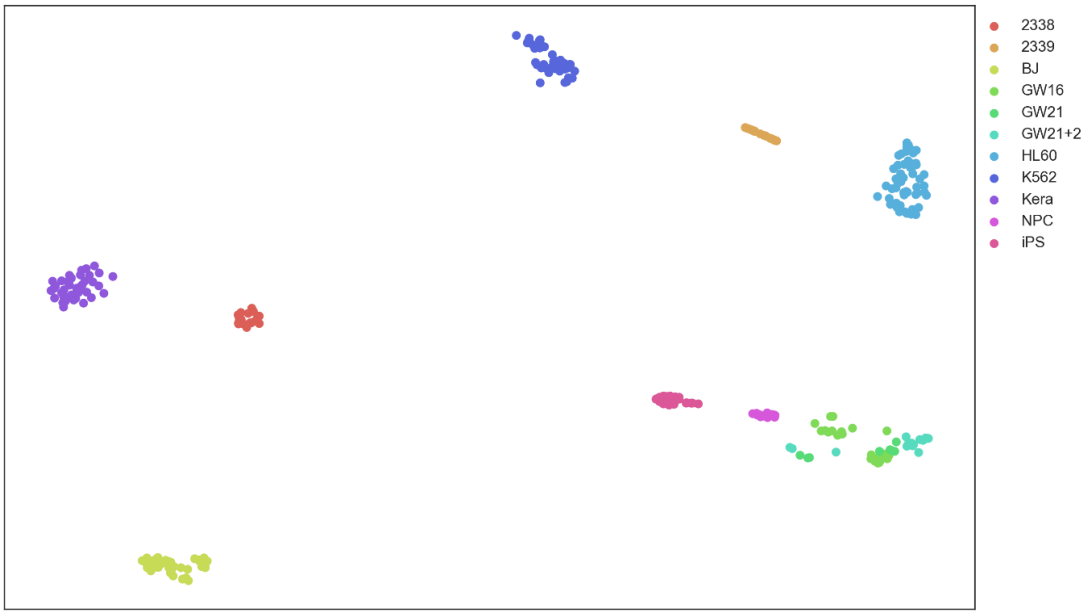
- You may visualize individual cluster on the TSNE map by Pollen.VisCluster(clevel,cnumber). The figure will be stored as a png file at your working directory.

```
[6] Pollen.VisCluster(clevel=1,cnumber=1)
```



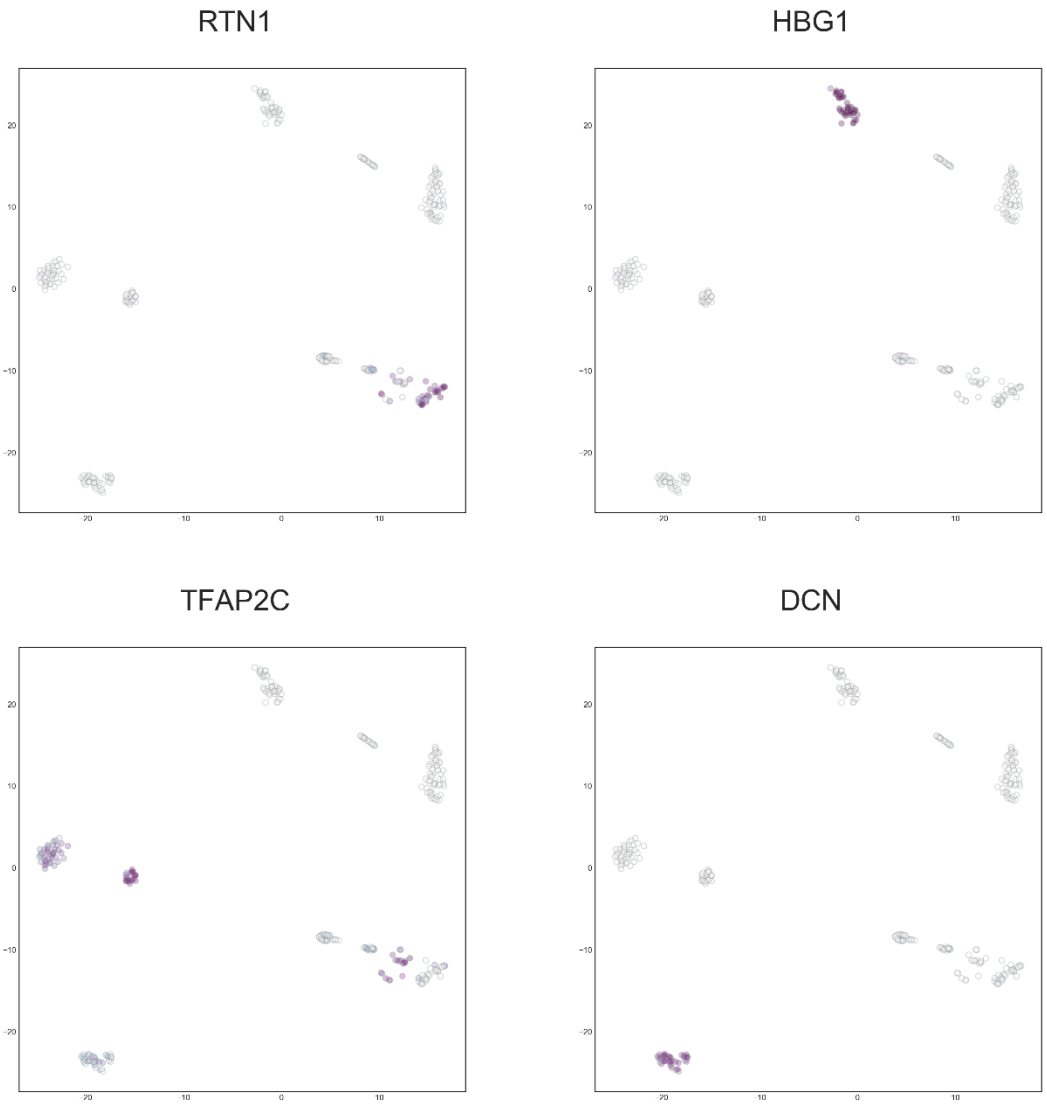
- You may visualize the cell population based on the input annotation by Pollen.VisClusterAnno() in order to compare the clusters of *PanoView*. The figure will be stored at your working directory.

[7] Pollen.VisClusterAnno()



- You may visualize the relative expression level of interested genes on the TSNE map by Pollen.VisGeneExp(genes). All the figures will be stored at your working directory.

```
[8] Pollen.VisGeneExp(genes=['DCN','RTN1','HBG1','TFAP2C'])
```



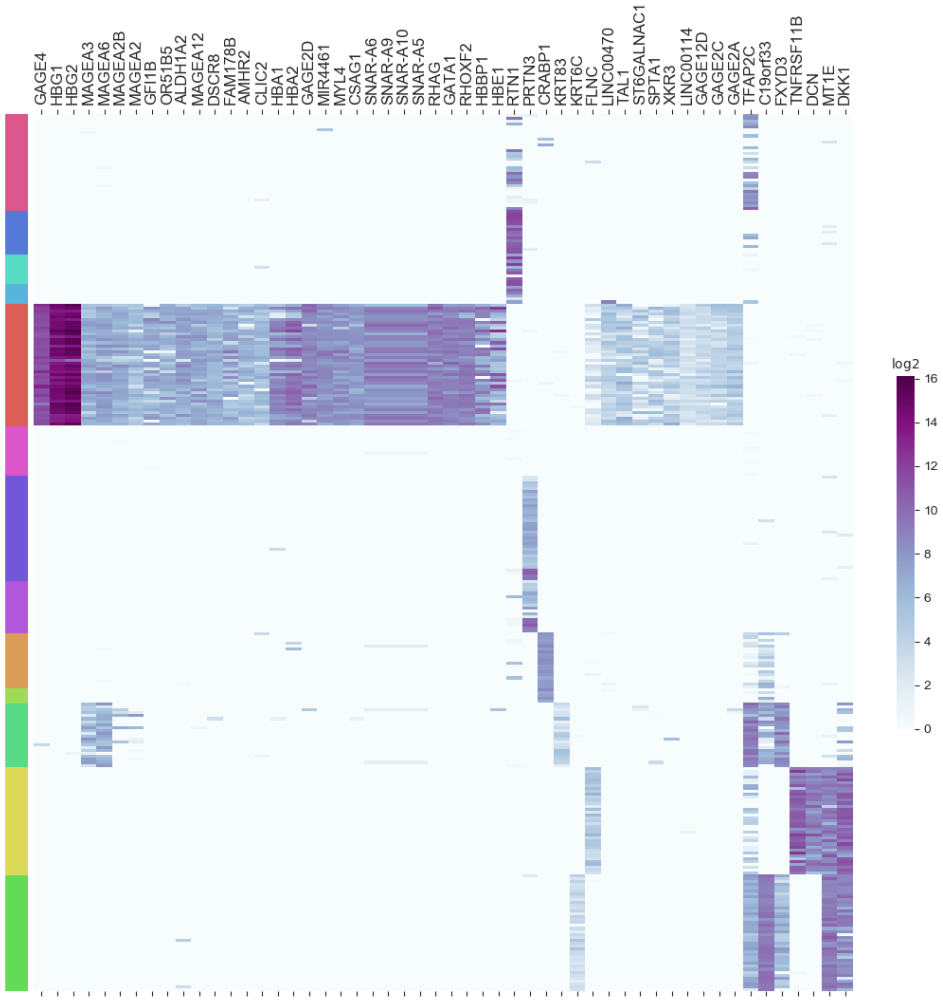
3.5 Variable Genes

PanoView implements the Kruskal-Wallis H-test tests to find variable genes based on the clustering result. Here we choose Level-1 clusters to do the testing. This step may take time due to the size of genes.

```
[9] Pollen.RunVGs(clevel=1)
```

You may visualize them as a heatmap by Pollen.HeatMapVGs(pval,number,fd,clevel,genes_add). pval is the threshold of adjusted pvalue. number is the number of top variable genes, fd is the threshold of log2 fold-change, clevel is the cluster level of *PanoView* that you want to visualize. genes_add is the additional genes that you want to include in the heatmap. These additional genes may be some known markers genes or your interested genes. The heatmap will be saved at your working directory.

```
[10]Pollen.HeatMapVGs(pval=0.05,number=50,fd=2,clevel=1,genes_add=['DCN','RTN1','HBG1','TFAP2C'])
```



4. **Functions in *PanoView***

4.1 RunSearching

```
RunSearching(GeneLow = 'default', Zscore = 'default', Normal=True, Log2=True)
```

To execute *PanoView* algorithm based on input single-cell expression matrix.

Parameters

- GeneLow: The threshold of lower end of expression value for finding variable genes. The default value is 0.5. For 10X/dropseq, the suggestion threshold is 0.01.
- Zscore: The threshold of z-score value for finding variable genes. The default value is 1.5
- Normal: Execution of normalization for expression matrix. The default is True
- Log2: Execution of log2-transfoem for expression matrix. The default value is True

Return

- JobName.cell_id: The name of cells
- JobName.cell_anno: The annotation of cells
- JobName.raw_exp: The raw values of expression
- JobName.log_exp: The log2 values of expression

- JobName.cell_clusters: The collection of arrays/clusters that *PanoView* identifies. Each array represents a sing-cell cluster that stores corresponding cell's id number

4.2 OutputResult

OutputResult(clust_merge = 'default', metric_dis = 'default', fclust_height = 'default', init = 'default', n_PCs = 'default')

To output result of *PanoView* algorithm.

Parameters

- clust_merge: The threshold of differential distance to merge neighboring clusters under the hierarchical tree. The default value is 0.2.
- metric_dis: The property that measures the pairwise distance of cells. *PanoView* uses this to estimate the similarity between identified clusters. The options are 1 (correlation: default) and 2 (euclidean).
- fclust_height: The height for deciding Level-2 clusters in *PanoView*. The default value is 0.2
- init: Initialization of embedding in the TSNE algorithm of scikit-learn. The options are 'pca' (default) and 'random'. User may change it to have preferred visualization for TSNE map
- n_PCs: Number of PCA components used to execute TSNE. The default number is 15. User may change it to have preferred visualization for TSNE map

Return

- JobName.tsne2d: 2D coordinates of TSNE map
- JobName.vargene: The list of variable genes used in the *PanoView* algorithm.

4.3 VisCluster

VisCluster(clevel,cnumber)

To visualize individual *PanoView* cluster in TSNE map

Parameters

- clevel: The level of *PanoView* clusters
- cnumner: The cluster id of *PanoView* clusters

4.4 VisClusterAnno

VisClusterAnno()

To visualize the cell population based on the input annotation.

4.5 VisGeneExp

VisGeneExp(genes)

To visualize expression level of selected genes

Parameters

- genes: The list of genes used for visualization in TSNE map

4.6 RunVGs

RunVGs(clevel)

To execute the Kruskal-Wallis H-test tests on *PanoView* clusters

Parameters

- clevel: The level of *PanoView* clusters used for executing the Kruskal-Wallis H-test tests

4.7 HeatMapVGs

HeatMapVGs(pval,number,fd,clevel,genelist = 'none')

To visualize expression of variable genes from Kruskal-Wallis H-test tests.

Parameters

- pval: The threshold of adjusted pvalue for plotting heatmap
- number: The threshold of number of variable genes for plotting heatmap
- fd: The threshold of log2 fold-change of variable genes for plotting heatmap
- genelist: The list of addition genes for plotting heatmap