scGEAToolbox

James Cai

Sep 23, 2023

MAIN DOCUMENTS

Single-cell RNA sequencing (scRNA-seq) technology has revolutionized the way research is done in biomedical sciences. It provides an unprecedented level of resolution across individual cells for studying cell heterogeneity and gene expression variability. Analyzing scRNA-seq data is challenging though, due to the sparsity and high dimensionality of the data. scGEAToolbox is a MATLAB toolbox for scRNA-seq data analysis. It contains a comprehensive set of functions for data normalization, feature selection, batch correction, imputation, cell clustering, trajectory/pseudotime analysis, and network construction, which can be combined and integrated to building custom workflow. While most of the functions are implemented in native MATLAB, wrapper functions are provided to allow users to call the "third-party" tools developed in Matlab or other languages. Furthermore, scGEAToolbox is equipped with sophisticated graphical user interfaces (GUIs), making it an easy-to-use application for quick data processing.

CHAPTER

ONE

OFFICIAL WEBSITES AND SOCIAL NETWORKS

Please, visit the official website of scGEAToolbox for further information.

1.1 Quick installation

Run the following code in *MATLAB*:

```
tic
disp('Installing scGEAToolbox...')
unzip('https://github.com/jamesjcai/scGEAToolbox/archive/main.zip');
addpath('./scGEAToolbox-main');
toc
if exist('scgeatool.m','file')
    disp('scGEAToolbox installed!')
end
savepath(fullfile(userpath,'pathdef.m'));
% savepath;
```

1.2 Getting Started

Run the following code in *MATLAB* to start SCGEATOOL:

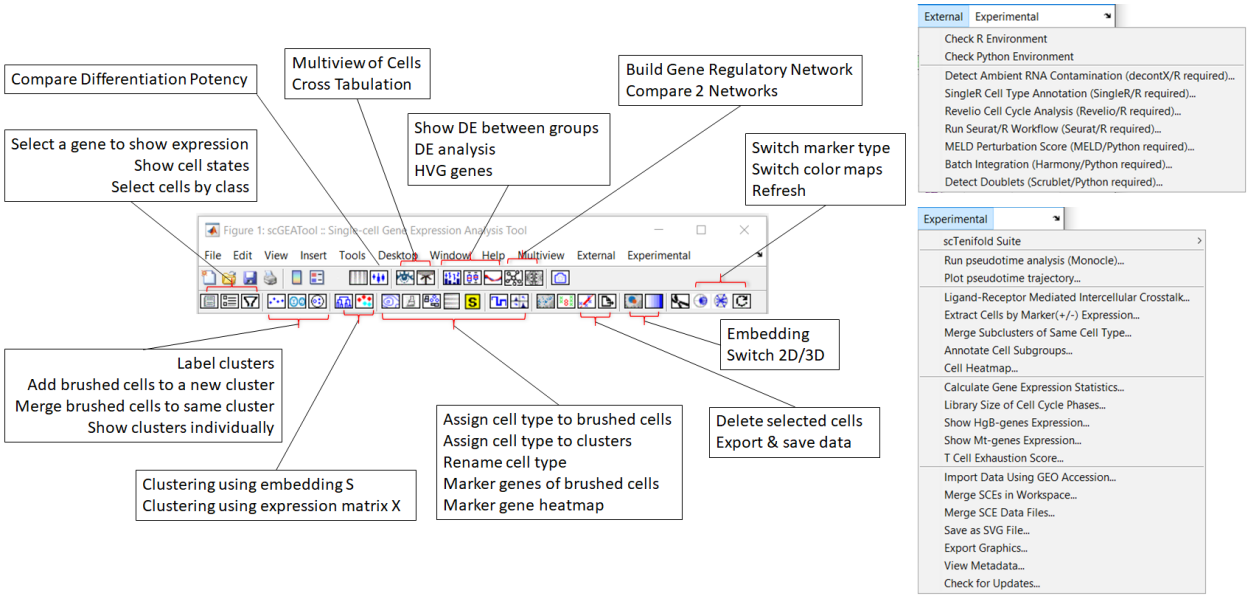
```
scgeatool
```

1.3 SCGEATOOL

SCGEATOOL is a lightweight and blazing fast MATLAB application that provides interactive visualization functionality to analyze single-cell transcriptomic data. SCGEATOOL allows you to easily interrogate different views of your scRNA-seq data to quickly gain insights into the underlying biology.

1.3.1 Overview

In MATLAB, scgeatool function can be used to start SCGEATOOL to visualize *SCE* class/object. Below are links to several case studies and examples using the scgeatool function to explore scRNA-seq data. All examples are below are publically available through GitHub.



1.3.2 Using SCGEATOOL to explore

For a quick exploratry data analysis using *scgeatool* function

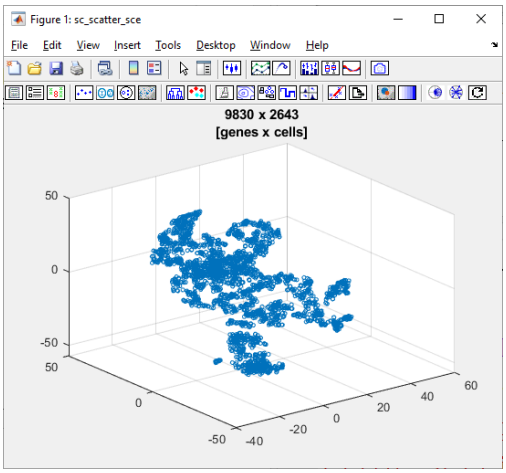
```
cdgea;
load example_data\testXgs.mat
scgeatool(X,g,s)
```

where X is the expression matrix, g is the list of genes, and s is the coordinates of embedding.

You can also load an example SCE (*SingleCellExperiment* object) variable using the following code:

```
cdgea;
load example_data\testSce.mat
scgeatool(sce)
```

If everything goes right, you will see the main inferface of SCGEATOOL like this:



1.3.3 Making scRNA-seq data into *SCE*

SingleCellExperiment defines a Single-cell Experiment (SCE) class in order to store scRNAseq data and variables. To make an SCE class, you need two variables: X and g, which are gene expression matrix and gene list, respectively.

```
cdgea;
load example_data\testXgs.mat
sce=SingleCellExperiment(X,g,s);
scgeatool(sce)
```

1.3.4 SCGEATOOL standalone for Windows

SCGEATOOL standalone is a lightweight and blazing fast desktop application that provides interactive visualization functionality to analyze single-cell transcriptomic data. SCGEATOOL allows you to easily interrogate different views of your scRNA-seq data to quickly gain insights into the underlying biology. SCGEATOOL is a pre-compiled standalone application developed in MATLAB. Pre-compiled standalone releases are meant for those environments without access to MATLAB licenses. Standalone releases provide access to all of the functionality of the SCGEATOOL standard MATLAB release encapsulated in a single application. SCGEATOOL is open-sourced to allow you to experience the added flexibility and speed of the MATLAB environment when needed.

1.4 Code Formulas

Example codes for common tasks.

1.4.1 Import 10x Genomics files

In the 10x Genomics folder, there are three files, namely, matrix.mtx, features.tsv (or genes.tsv) and barcodes.tsv. Here is how to import them:

```
mtxf='GSM3535276_AXLN1_matrix.mtx';
genf='GSM3535276_AXLN1_genes.tsv';
bcdf='GSM3535276_AXLN1_barcodes.tsv';
[X,genelist,barcodelist]=sc_readmtxfile(mtxf,genf,bcdf,2);
```

If the barcodees.tsv is not available, then use the following

```
mtxf='GSM3535276_AXLN1_matrix.mtx';
genf='GSM3535276_AXLN1_genes.tsv';
[X,g]=sc_readmtxfile(mtxf,genf,[],2);
```

1.4.2 Process expression matrix, *X* and gene list, *g*

Here is an example of raw data processing.

```
[X,g,b]=sc_readmtxfile('matrix.mtx','features.tsv','barcodes.tsv',2);
[X,g]=sc_qcfilter(X,g);
[X,g]=sc_selectg(X,g,1,0.05);
[s]=sc_tsne(X);
scgeatool(X,g,s)
```

1.4.3 t-SNE embedding of cells using highly varible genes (HVGs)

```
[~,Xhvg]=sc_hvg(X,g);
[s]=sc_tsne(Xhvg(1:2000,:));
scgeatool(X,g,s)
```

1.4.4 An example pipeline for raw data processing

```
[X,g]=sc_readmtxfile('matrix.mtx','features.tsv');
[X,g]=sc_qcfilter(X,g);                    % basic QC
[X,g]=sc_selectg(X,g,1,0.05);              % select genes expressed in at least 5% of␣
↪cells
[~,Xhvg]=sc_hvg(X,g);                      % identify highly variable genes (HVGs)
[s]=sc_tsne(Xhvg(1:2000,:));               % using expression of top 2000 HVGs for tSNE
sce=SingleCellExperiment(X,g,s);           % make SCE class
sce=sce.estimatepotency(2);                % estimate differentiation potency (1-human; 2-
↪mouse)
sce=sce.estimatecellcycle;                 % estimate cell cycle phase
id=sc_cluster_s(s,10);                     % clustering on tSNE coordinates using k-means
sce.c_cluster_id=id;                       % assigning cluster Ids to SCE class
scgeatool(sce)                             % visualize cells
```

1.4.5 An example pipeline for processing 10x data folder

Assuming the .m file containing the following code is in the folder ./filtered_feature_bc_matrix. In this folder, three files: matrix.mtx.gz, features.tsv.gz, and barcodes.tsv.gz, are present.

```
[X,genelist,celllist]=sc_read10xdir(pwd);
sce=SingleCellExperiment(X,genelist);
sce.c_cell_id=celllist;
sce=sce.qcfilter;
sce=sce.estimatecellcycle;
```

(continues on next page)

(continued from previous page)

```
sce=sce.estimatepotency("mouse");
sce=sce.embedcells('tSNE',true);
save clean_data sce -v7.3
scgeatool(sce)
```

1.4.6 Merge two data sets (WT and KO)

```
load WT/clean_data.mat sce
sce_wt=sce;
load KO/clean_data.mat sce
sce_ko=sce;
sce=sc_mergesces({sce_wt,sce_ko},'union');    % use parameter 'union' or 'intersect' to␣
↪merge genes
sce.c=sce.c_batch_id;
scgeatool(sce)                                % blue - WT and red - KO
```

You may want to re-compute tSNE coordinates after merging.

1.5 Case Studies and Tutorials

1.5.1 Download 10x Genomics data files from GEO

From GEO database, we obtain the FTP links to the data files we need. Here we use a data set from sample GSM3535276 as an example (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM3535276). The sample is human AXLN1 lymphatic endothelial cells.

We can use *gunzip* function directly download and unzip the files.

```
gunzip('https://ftp.ncbi.nlm.nih.gov/geo/samples/GSM3535nnn/GSM3535276/suppl/GSM3535276_
↪AXLN1_matrix.mtx.gz');
gunzip('https://ftp.ncbi.nlm.nih.gov/geo/samples/GSM3535nnn/GSM3535276/suppl/GSM3535276_
↪AXLN1_genes.tsv.gz');
```

We can then use the code below to import data into *MATLAB*.

```
[X,g]=sc_readmtxfile('GSM3535276_AXLN1_matrix.mtx','GSM3535276_AXLN1_genes.tsv');
scgeatool(X,g)
```

1.5.2 Process downloaded 10x Genomics data files

In a 10x Genomics data folder, there should be matrix.mtx and genes.tsv. Here is the commandline code for raw data processing.

```
[X,g]=sc_readmtxfile('matrix.mtx','genes.tsv');
[X,g]=sc_qcfilter(X,g);
[X,g]=sc_selectg(X,g,1,0.05);
[s]=sc_tsne(X);
scgeatool(X,g,s)
```

1.5.3 Download Drop-seq data files from GEO

From GEO database, we obtain the FTP links to the data files we need. Here we use a data set from sample GSM3036814 as an example (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM3036814). The sample is mouse lung cells.

We can use *gunzip* function directly download and unzip the files.

```
gunzip('https://ftp.ncbi.nlm.nih.gov/geo/samples/GSM3036nnn/GSM3036814/suppl/GSM3036814_
↪Control_6_Mouse_lung_digital_gene_expression_6000.dge.txt.gz')
```

We can then use the code below to import data into *MATLAB*.

```
[X,g]=sc_readtsvfile('GSM3036814_Control_6_Mouse_lung_digital_gene_expression_6000.dge.
↪txt');
[X,g]=sc_qcfilter(X,g);
[X,g]=sc_selectg(X,g,1,0.05);
[s]=sc_tsne(X);
scgeatool(X,g,s)
```

1.5.4 Import Seurat RData

For example, we are trying to read files from https://www.synapse.org/#!Synapse:syn22855256. They are described as *pbmc_discovery_v1.RData* and *pbmc_replication_v1.RData* are Seurat objects containing the gene expression raw counts and log normalized data, the phenotype Label ("CI" for MCI, "C" for control) and the inferred cell identity of the discovery and replication cohort, respectively.

```
library(Seurat)
library(Matrix)
load('pbmc_discovery_v1.RData')
countMatrix <- pbmc_discovery@assays$RNA@counts
writeMM(obj = countMatrix, file = 'matrix.mtx')
writeLines(text = rownames(countMatrix), con = 'features.tsv')
writeLines(text = colnames(countMatrix), con = 'barcodes.tsv')
metadata <- pbmc_discovery@meta.data
write.csv(x = metadata, file = 'metadata.csv', quote = FALSE)
```

After exporting Seurate object data into the three files, you can then use MATLAB to read the files:

```
[X,genelist,barcodelist]=sc_readmtxfile('matrix.mtx','features.tsv','barcodes.tsv',1);
sce=SingleCellExperiment(X,genelist);
T=readtable('metadata.csv')
c=string(T.Label);
sce.c_batch_id=c;
scgeatool(sce)
```

1.5.5 Import data from a TSV/Excel file

If your scRNA-seq data is in Excel file, save it as TSV or CSV a file with the format like this:

```
genes X1      X2      X3      X4      X5      X6      X7      X8      X9
NOC2L 1       1       2       3       3       2       0       1       3
HES4  50      15      19      50      8       87      23      25      29
ISG15 279     312     425     180     406     408     335     403     398
AGRN  3       4       9       5       2       3       8       8       9
SDF4  2       2       4       0       5       0       4       2       5
B3GALT6       2       1       0       0       1       0       1       1       0
UBE2J2        1       2       3       1       1       1       6       3       4
SCNN1D        0       1       0       0       0       0       0       0       0
ACAP3 1       3       1       0       1       0       0       1       0
```

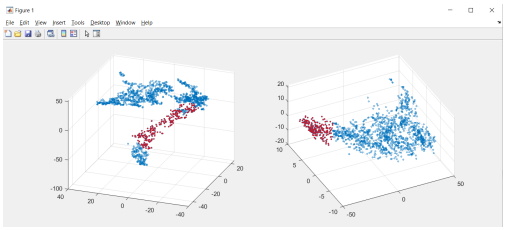
Then you can use function *sc_readtsvfile* to import the data. Here is an example:

```
cdgea;
[X,g]=sc_readtsvfile('example_data\GSM3204304_P_P_Expr.csv');
```

1.5.6 Visualize data in 6D

```
cdgea;
load example_data\example10xdata.mat
% s=sc_tsne(X,6,false,true);
s=s_tsne6;    % using pre-computed 6-d embedding S_TSNE6
gui.sc_multiembeddings(s(:,1:3),s(:,4:6));
```

Here is what you should get:



1.6 Youtube Playlist

You might also want to take a look at scGEAToolbox in action: see the Youtube playlist

1.6.1 Using *SCGEATOOL* to explore scRNA-seq data stored as *SCE* class

1.6.2 Label cell type interactively with *SCGEATOOL*

1.7 Slack Channel

Visit the dedicated Slack Channel if you have questions or to report bugs.

1.8 Twitter Hashtag

If you create amazing visualizations using *scGEAToolbox* and you want to tweet them, remember to include the #scGEAToolbox hashtag: we will be happy to retweet.

1.9 Matlab Central

You might be interested in taking a look at the File Exchange.

1.10 A1: The scGEAToolbox Paper

- Cai JJ, "scGEAToolbox: a Matlab toolbox for single-cell RNA sequencing data analysis," *Bioinformatics*, **btz830**, (2019).

1.11 A2: Papers Citing scGEAToolbox

- https://scholar.google.com/scholar?cites=4661048952867744439&as_sdt=5,44&sciodt=0,44&hl=en

1.12 A3: License Agreement

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.