CALISTA: User Manual

Version 1.1.1 (6 March 2018)

Authors: Nan Papili Gao and Rudiyanto Gunawan
Institute for Chemical and Bioengineering
ETH Zurich
Contact e-mail: nanp@ethz.ch

Table of contents

1 Overview

This user manual is for the MATLAB distribution of CALISTA (Clustering And Lineage Inference in Single Cell Transcriptional Analysis).

CALISTA provides a user-friendly toolbox for the analysis of single cell expression data. CALISTA accomplishes three major tasks:

(1) Identification of cell clusters in a cell population based on single-cell gene expression data;
(2) Reconstruction of lineage progression and produce transition genes;
(3) Pseudotemporal ordering of cells along any given developmental paths in the lineage progression.

For detailed information about CALISTA, please refer to the following manuscript.
Papili Gao N., Hartmann T. and Gunawan R., **CALISTA: Clustering and lineage inference in single-cell transcriptional analysis**, bioRxiv, 2018. https://doi.org/10.1101/257550

2 System requirements

This distribution of CALISTA is written for and developed in MATLAB[1].
CALISTA has been successfully tested on MATLAB 2016b, 2017a and 2018a.

3 CALISTA package

CALISTA package contains the following files and folders:

1. This `CALISTA_USER_MANUAL.doc` file
2. `License.txt` modified BSD license for CALISTA
3. `MAIN.m` example script on how to use CALISTA subroutines
4. `MAIN_GUI.m` example script on how to use GUI version of CALISTA
5. The folder `Two-state model parameters` containing:
    a. `Parameters.mat` steady-state distribution functions of mRNA level
6. The folder `subfunctions` containing the following main subroutines (and other subroutines):
    a. `import_data.m` : upload single-cell expression data and perform preprocessing.
    b. `CALISTA_clustering_main.m`: single-cell clustering in CALISTA.
    c. `CALISTA_transition_main.m`: infer lineage progression among cell clusters.
    d. `CALISTA_transition_genes_main.m`: identify the key genes in lineage progression.
    e. `CALISTA_ordering_main.m`: perform pseudotemporal ordering of cells.
    f. `CALISTA_path_main.m`: perform post-analysis along developmental path(s).
7. The folder `EXAMPLES` containing single-cell expression datasets used in the examples below.
8. The folder `GUI` containing the subroutines used in `MAIN_GUI.m`

[1] http://www.mathworks.com

For further information on running the main subroutines in CALISTA, please use Matlab 'help' command followed by function_name (for example 'help import_data').

4 Examples

In the following, we describe the main steps of CALISTA applied to publicly available single-cell gene expression data. For each dataset, ONLY the most important results are reported. Please refer to the file MAIN.m for an example MATLAB script of CALISTA implementation.

4.1 Example 1. iPSC differentiation into mesodermal and endodermal cells

Analysis of RT-qPCR data of Bargaje et al. (Bargaje, et al, Cell population structure prior to bifurcation predicts efficiency of directed differentiation in human induced pluripotent cells. *Proc. Natl. Acad. Sci. U. S. A*. 114, 2271–2276 (2017)).

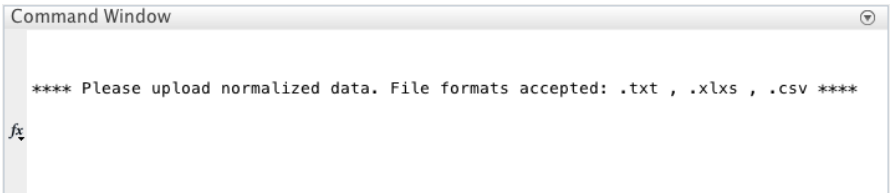
4.1.1 Data Import and Preprocessing

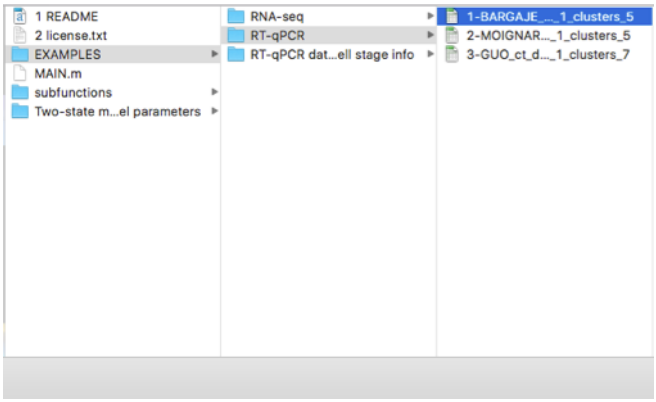
We begin with changing the current directory in MATLAB to the CALISTA folder. Then, we edit the MAIN.m script in the main folder of CALISTA and set the fields of INPUTS as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=1; % Single-cell RT-qPCR CT data
INPUTS.format_data=1; % Rows= cells and Columns = genes with time/stage info in the
last column
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=100; % Remove genes with >= 100% of zeros
INPUTS.perczeros_cells=100; % Remove cells with >= 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells

% Specify single-cell clustering settings
INPUTS.perc_top_genes=100; % Retain only top X of the most variable genes with X =
min(200, INPUTS.perc_top_genes * number of cells/100, number of genes)
INPUTS.optimize=1; % Set the number of clusters based on Eigengap Statistics
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm in clustering
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=50; % Set threshold for transition genes determination to
50 percent

% Specify path analysis settings
INPUTS.plot_fig=1; % Plot figure of smoothened gene expression along path
INPUTS.hclustering=1; % Perform hierarchical clustering of gene expression for each
path
INPUTS.method=2; % Use pairwise correlation for the gene co-expression network
(value_cutoff=0.8, pvalue_cutoff=0.01)
INPUTS.moving_average_window=10; % Set the size of window (percent of cells in each
path) used for the moving averaging
```

We subsequently run MAIN.m and import Bargaje dataset (available in the subfolder EXAMPLES/RT-qPCR).

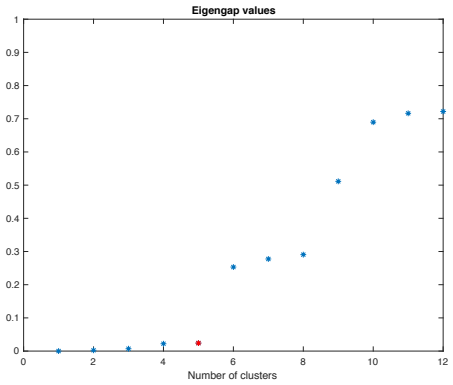
The following are screenshots from running CALISTA on MATLAB.

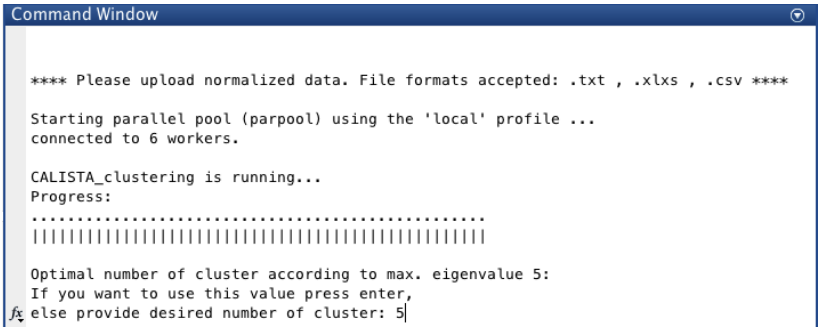


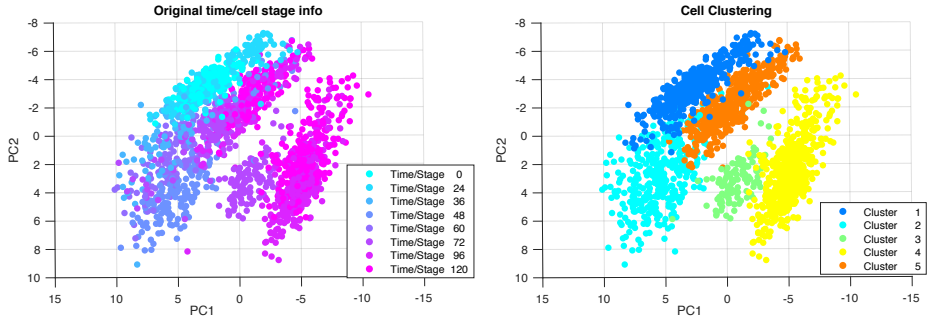


4.1.2 Single-cell clustering

In this case, the number of clusters is determined using the eigengap plot. According to the eigengap plot below, we set the number of clusters to **5**. The following are screenshots from CALISTA single-cell clustering analysis.







If desired, users can remove cells from specific clusters from further analysis. In this example, we do not want to remove any clusters. Hence, we enter **0** (no cluster removal) and then **1** to proceed with lineage inference.

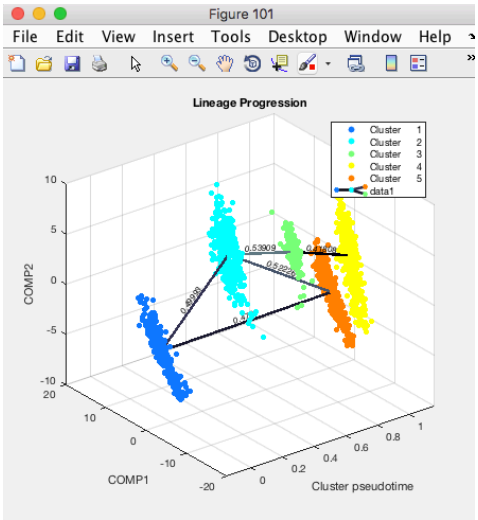
```
Command Window
Plotting...
Press 1 if you want to remove one cell cluster, 0 otherwise: 0
 Press 1 if you want to perform additional analysis (e.g. lineage inference, cell ordering) , 0 otherwise: 1
```

4.1.3 Reconstruction of lineage progression

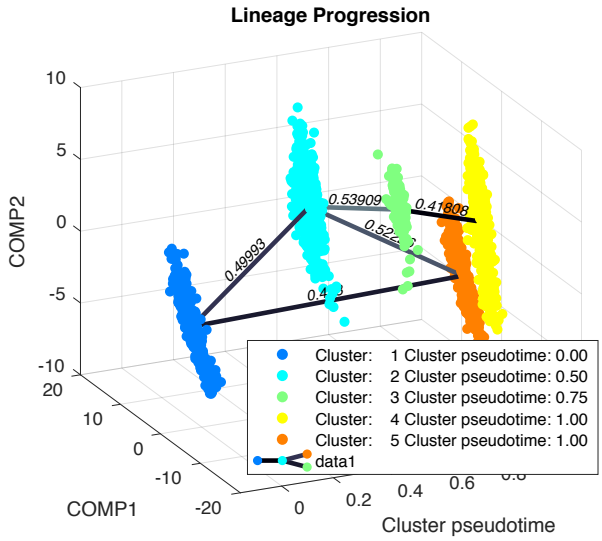
During the lineage inference step, CALISTA automatically generates and displays a lineage graph, obtained by adding an edge between two clusters in increasing cluster distances, until all clusters are connected to at least one other cluster. Subsequently, users can manually add or remove one edge at time based on the cluster distances.

```
Command Window
CALISTA_transition is running...

5 edge(s) have been added and the graph is connected.
If you want to add another edge press "p"
If you want to remove an edge press "m"
If you want to continue with the next step press "enter"
(Please make sure that figure 101 is in the foreground and no additional tools are selected (e.g. zooming, rotatio

-----------------------------------------------
```

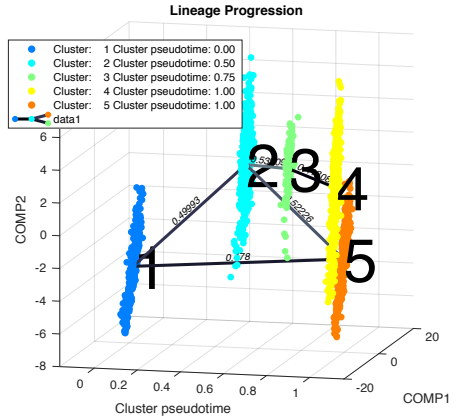
ATTENTION: to add an edge (press "**p**"), remove an edge (press "**m**") or finalize the lineage progression graph (press "**enter**"), the MATLAB figure of the graph must appear in foreground without any modification (e.g., zooming, rotation). Note that the addition/removal of the edges are performed according to increasing/decreasing order of cluster distance.



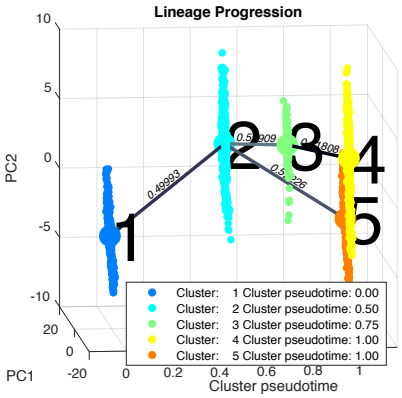
ATTENTION: the final graph must be connected (i.e. there exists a path from any node/cluster to any other node/cluster in the graph), otherwise a warning will be returned.



Since the transition from cluster 1 to cluster 5 is inconsistent with the capture time info (i.e. cluster pseudotime values for cluster 1 and 5 are 0 and 1 respectively) we remove the spurious edge between cluster 1 and 5, by entering **1** and entering **[1 5]**, upon the following query.

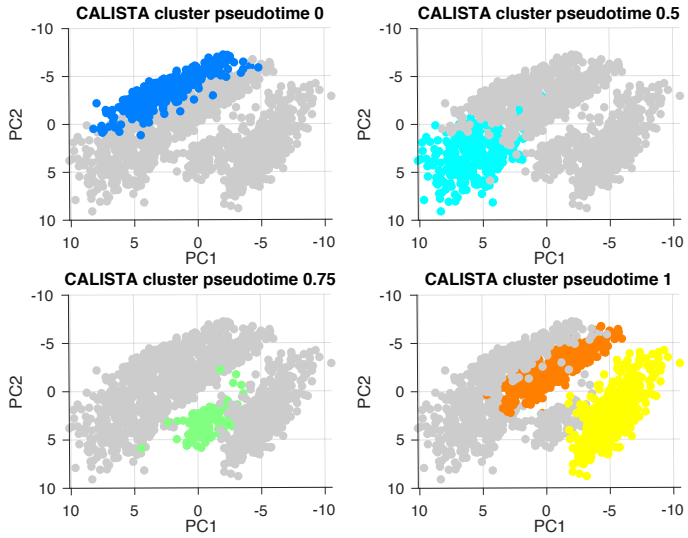


The final inferred lineage relationships are displayed below.

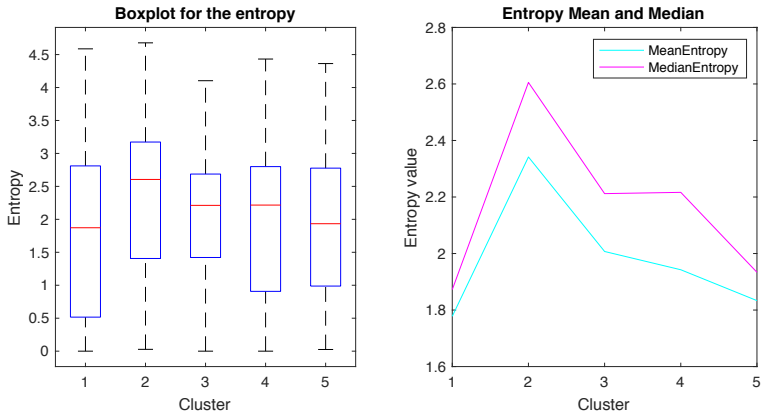


In addition, CALISTA provides the following.

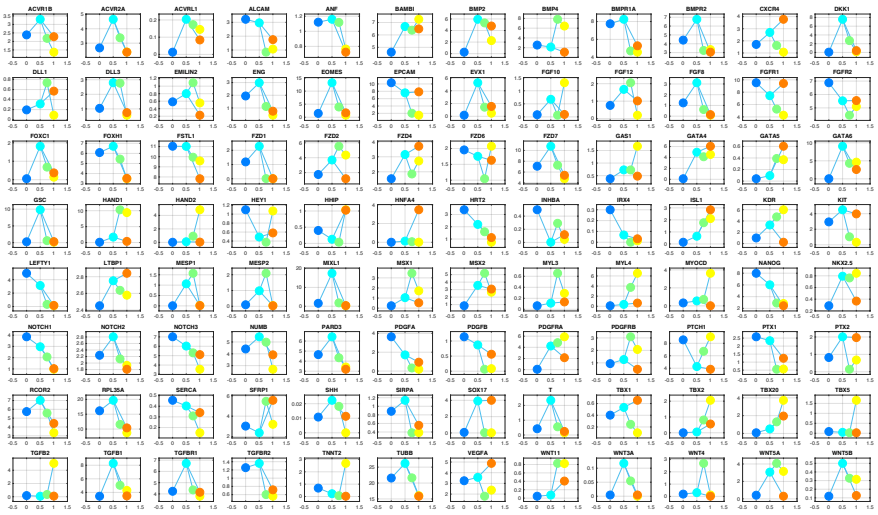
- Cell clustering plot based on the cluster pseudotime



- Boxplot, mean, median entropy values calculated for each cluster

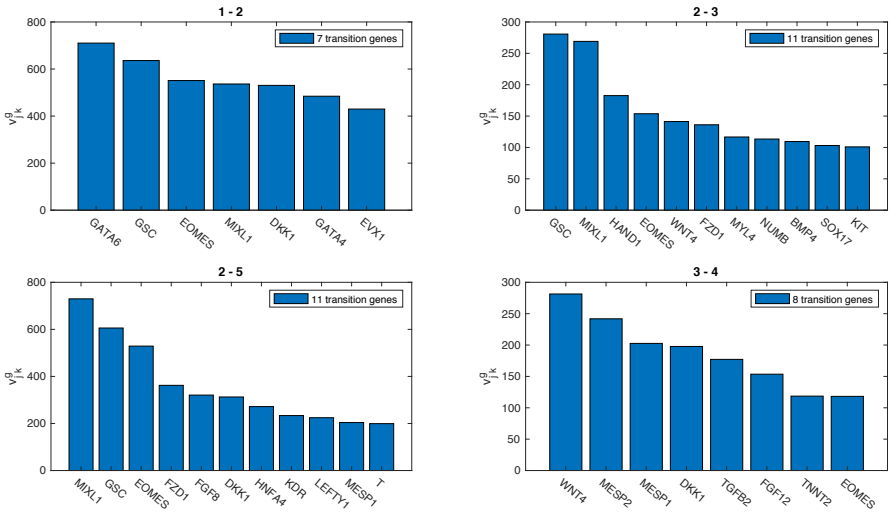


- Plot of mean expression values for each gene based on cell cluster expression level



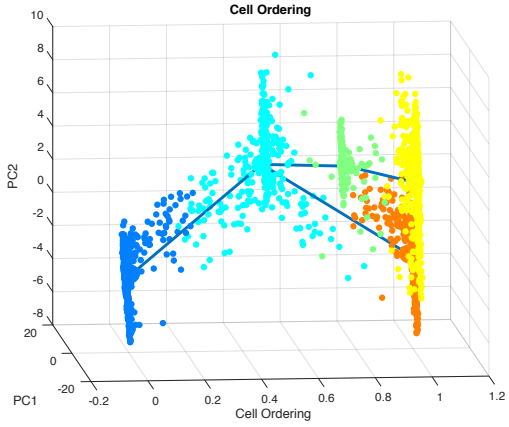
4.1.4 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph, based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.



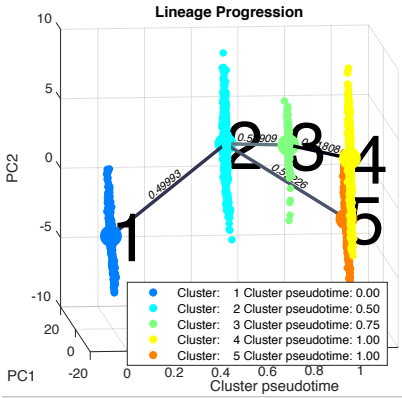
4.1.5 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.



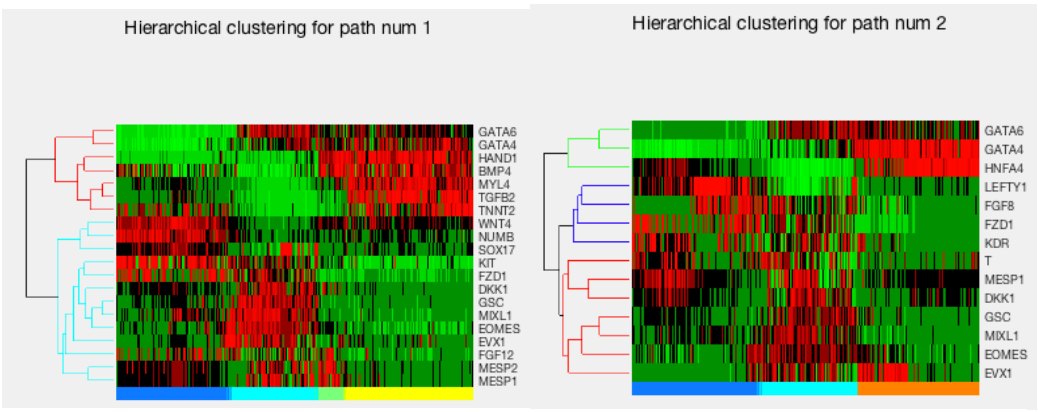
4.1.6 Path analysis

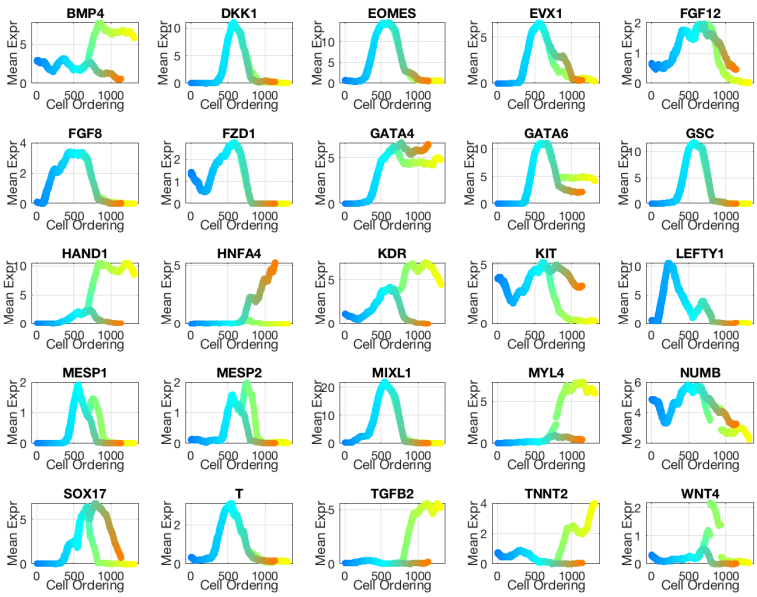
To perform path-specific analysis, users can enter **1** upon queried. In the following, we input two developmental paths of interest: **[1 2 3 4]** (mesodermal fate) and **[1 2 5]** (endodermal fate).

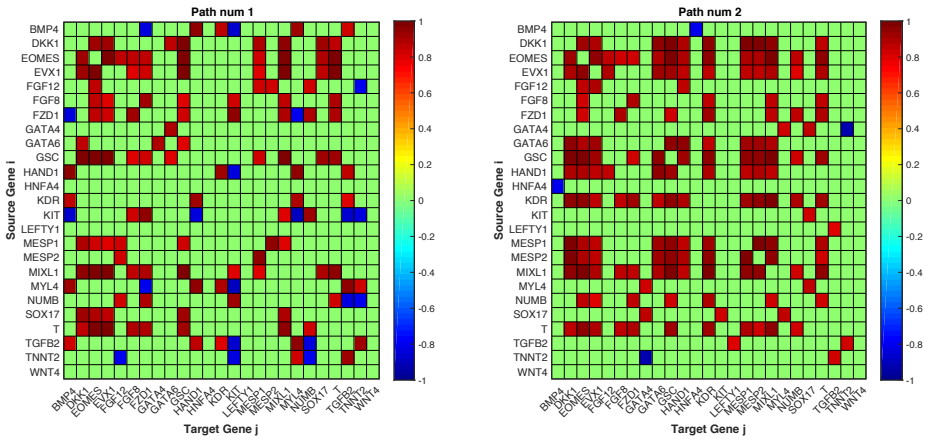



```
Command Window
 Press 1 if you want to select transition paths and perform additional analysis, 0 otherwise: 1
Path num:     1
 ******************************************************
 Key the clusters in the path (e.g. [1 2 3 4]: [1 2 5]
 Press 1 to add another path, 0 otherwise: 1
Path num:     2
 ******************************************************
 Key the clusters in the path (e.g. [1 2 3 4]: [1 2 3 4]
 Press 1 to add another path, 0 otherwise: 0
```

For each path, the post-analysis in CALISTA generates Clustergrams, moving-averaged gene expression profiles and co-expression networks for the transition genes detected previously based on cell orderings.







4.2 Example 2. Hematopoietic stem cell differentiation

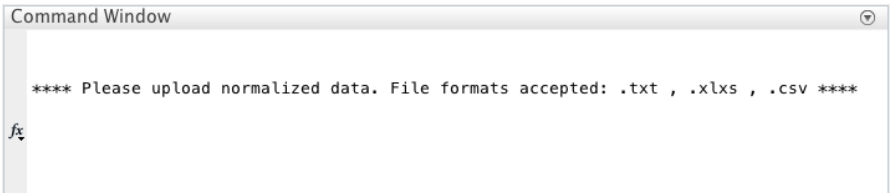
Analysis of RT-qPCR data in Moignard et al., Characterization of transcriptional networks in blood stem and progenitor cells using high-throughput single-cell gene expression analysis, *Nat. Cell Biol*. **15**, 363–72 (2013).

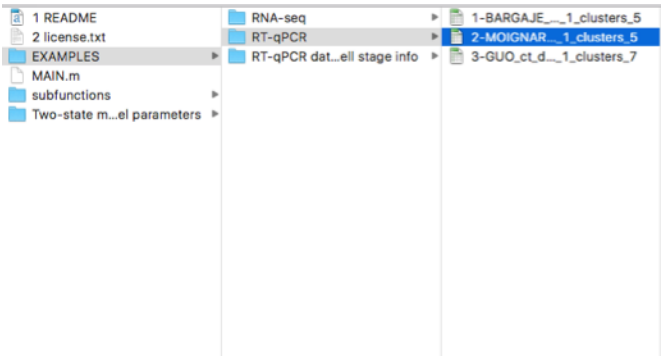
4.2.1 Data Import and Preprocessing

We start by changing the current directory in MATLAB to the CALISTA folder.
We edit the `MAIN.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=1; % Single-cell RT-qPCR CT data
INPUTS.format_data=1; % Rows= cells and Columns= genes with time/stage info in the
last column
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=100; % Remove genes with > 100% of zeros
INPUTS.perczeros_cells=100; % Remove cells with 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells

% Specify single-cell clustering settings
INPUTS.perc_top_genes=100; % Retain only top X the most variable genes with X=min(200,
INPUTS.perc_top_genes * num of cells/100, num of genes)
INPUTS.optimize=0; % The number of cluster is known a priori
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=50; % Set threshold for transition genes determination to
50%

% Specify path analysis settings
INPUTS.plot_fig=1; % Plot figure of smoothened gene expression along path
INPUTS.hclustering=1; % Perform hierarchical clustering of gene expression for each
path
INPUTS.method=2; % Use pairwise correlation for the gene co-expression network
(value_cutoff=0.8, pvalue_cutoff=0.01)
INPUTS.moving_average_window=10; % Set the size of window (percent of cells in each
path) used for the moving averaging
```

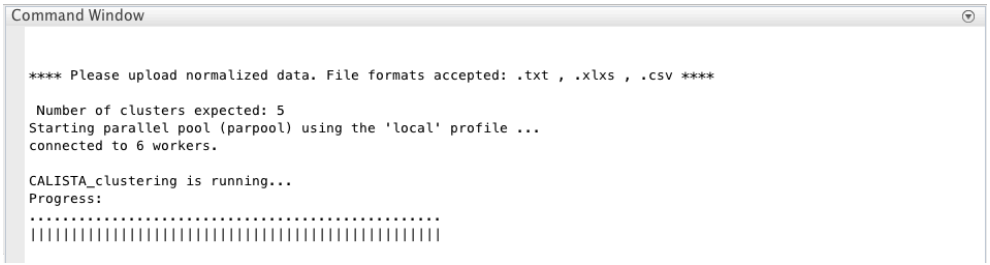
We then run `MAIN.m` from the workspace and load Moignard dataset (in subfolder EXAMPLES/RT-qPCR).



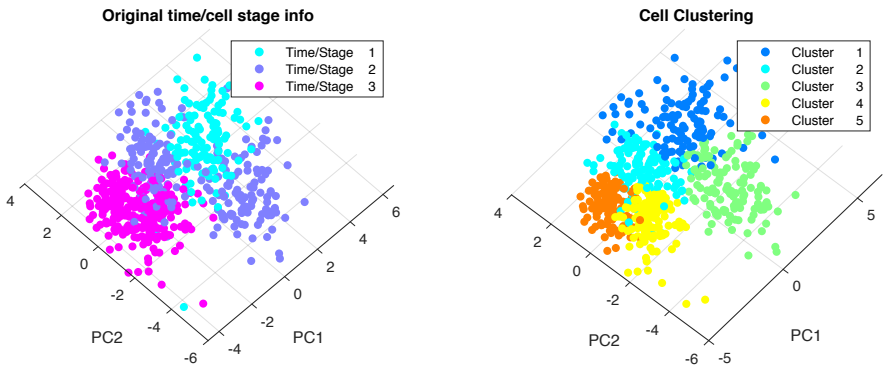


4.2.2 Single-cell clustering

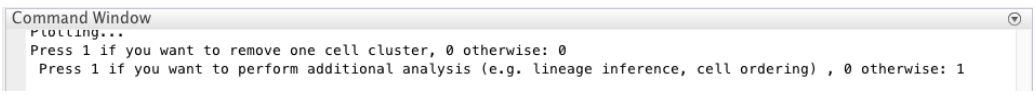
Following the original publication, we set the number of clusters equals to **5**.



CALISTA single-cell clustering results are as follow.

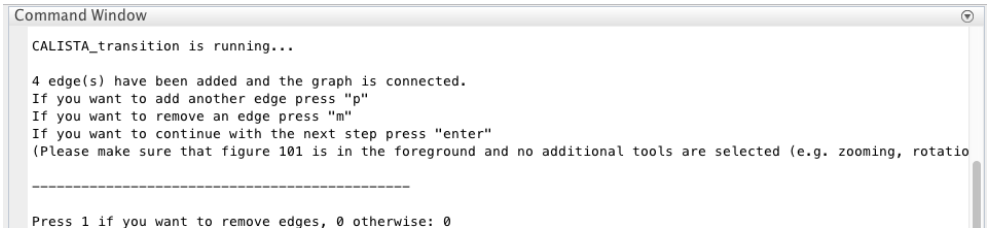


In this case, we do not need to remove any clusters (by pressing **0** upon queried). Then we proceed with further analysis (by pressing **1** upon queried).

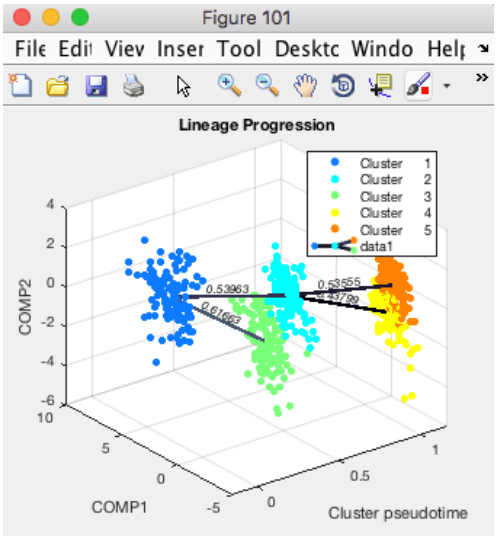


4.2.3 Reconstruction of lineage progression

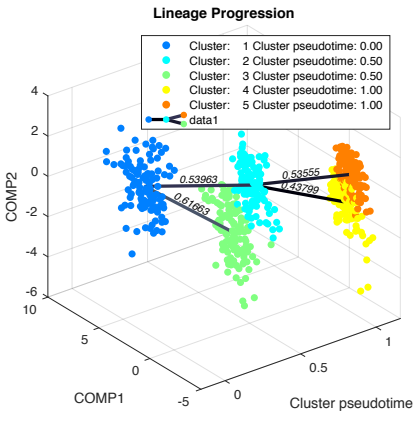
During the lineage inference step, CALISTA automatically generates and displays a lineage graph, obtained by adding an edge between two clusters in increasing cluster distances, until all clusters are connected to at least one other cluster. Subsequently, users can manually add or remove one edge at time based on the cluster distances.



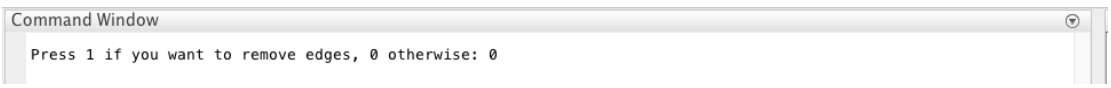
ATTENTION: to add an edge (press "**p**"), remove an edge (press "**m**") or finalize the lineage progression graph (press "**enter**"), the MATLAB figure of the graph must appear in foreground without any modification (e.g., zooming, rotation). Note that the addition/removal of the edges are performed according to increasing/decreasing order of cluster distance.



ATTENTION: The final lineage progression graph must be connected (i.e. there is a path from any node/cluster to any other node/cluster in the graph) otherwise a warning will be returned.



Here, we do not need to remove any spurious edges, and hence we enter **0** upon queried.

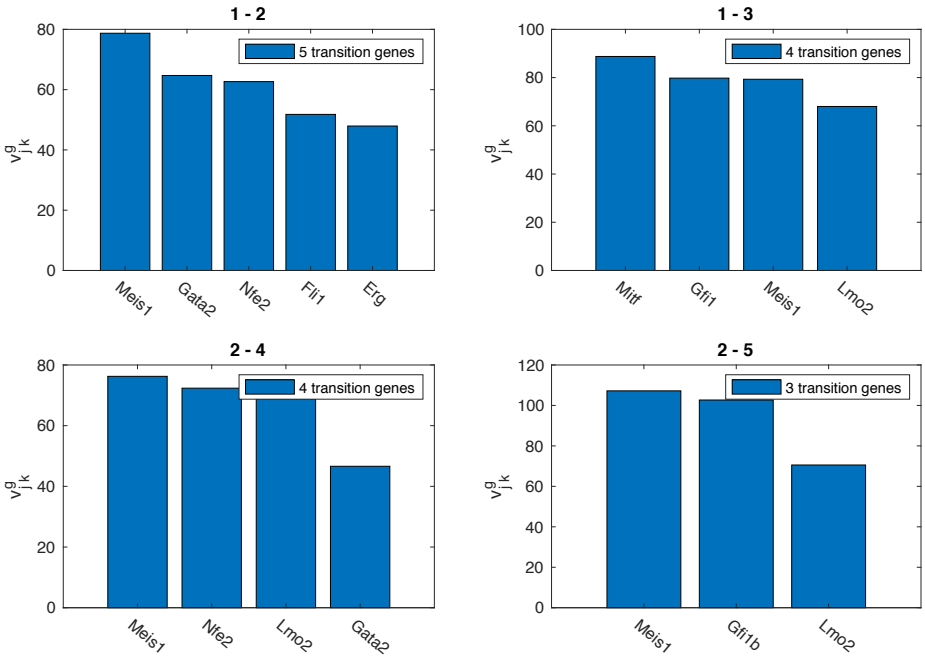


In addition, CALISTA gives (not shown):

- Cell clustering plot based on the cluster pseudotime
- Boxplot, mean, median entropy values calculated for each cluster
- Plot of mean expression values for each gene based on cell cluster expression level

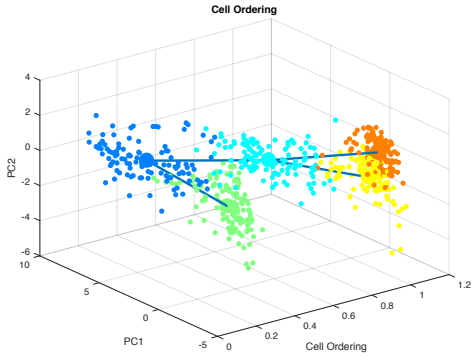
4.2.4 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph, based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.



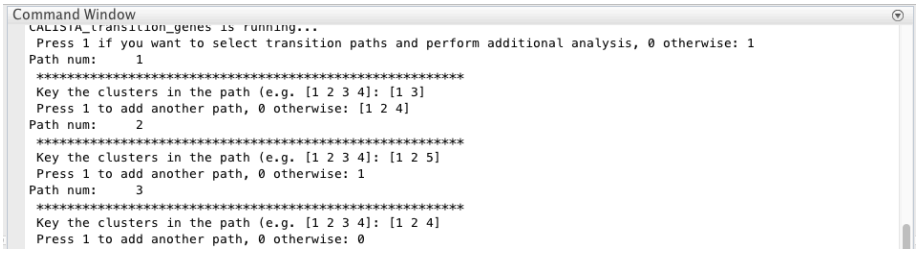
4.2.5 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.



4.2.6 Path analysis

Finally, we perform post-analysis by entering **1** upon queried. Here, we input three developmental paths: **[1 3]**, **[1 2 5]**, and **[1 2 4]**.



For each path, the post-analysis in CALISTA generates Clustergrams, moving-averaged gene expression profiles and co-expression networks for the transition genes detected previously based on cell orderings (not shown).

4.3 Example 3. Mouse embryonic fibroblast differentiation into neurons (Manual data import)

Analysis of RNA-seq data in Treutlein et al., Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-seq, *Nature* **534,** 391–395 (2016).

Please unzip the file "3-TREUTLEIN_data_type_3_format_data_5_clusters_4.txt.zip**" in **EXAMPLES/RNA-seq/** before running CALISTA**

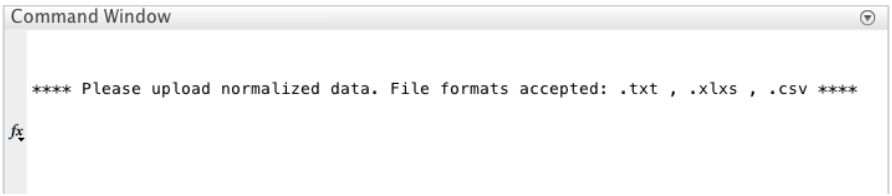
4.3.1 Data Import and Preprocessing

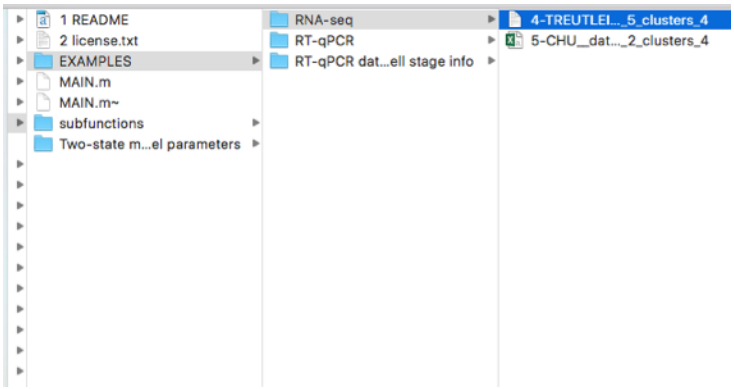
Again, we change the current directory in MATLAB to the CALISTA folder.

Here, we edit the `MAIN.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

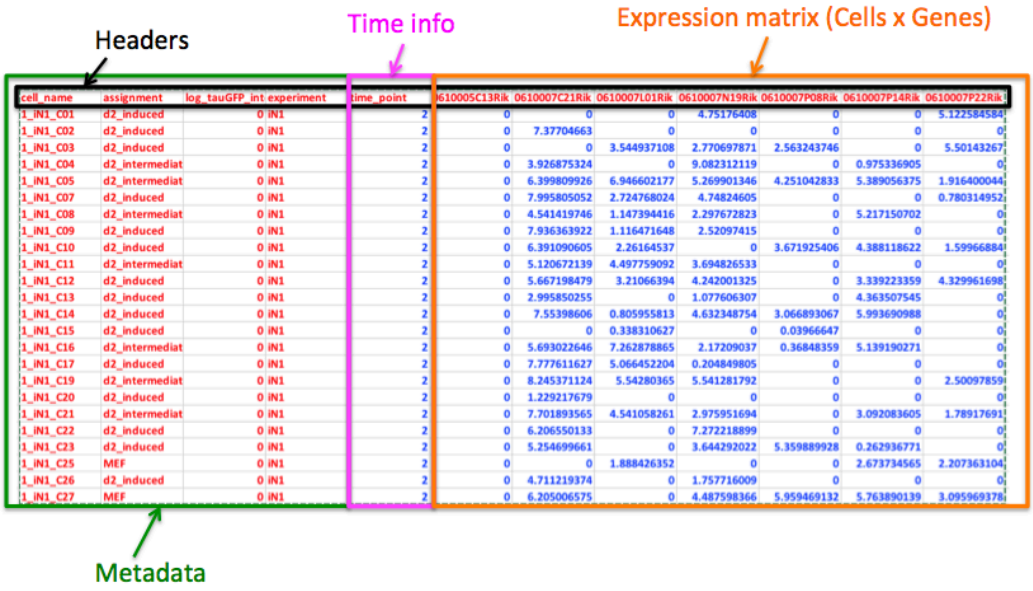
```
% Specify data types and settings for pre-processing
INPUTS.data_type=3; % Single-cell RNA-seq data
INPUTS.format_data=5; % Manual selection from data table
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=90; % Remove genes with > 90% of zeros
INPUTS.perczeros_cells=100; % Remove cells with 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells
INPUTS.perc_top_genes=10; % Retain only top X the most variable genes with X=min(200,
INPUTS.perc_top_genes * num of cells/100, num of genes)

% Specify single-cell clustering settings
INPUTS.optimize=1; % Set the number of clusters based on Eigengap Heuristics
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=75; % Set threshold for transition genes determination to
75%
```

We run `MAIN.m` from the workspace and load Treutlein dataset (in subfolder EXAMPLES/RNA-seq).



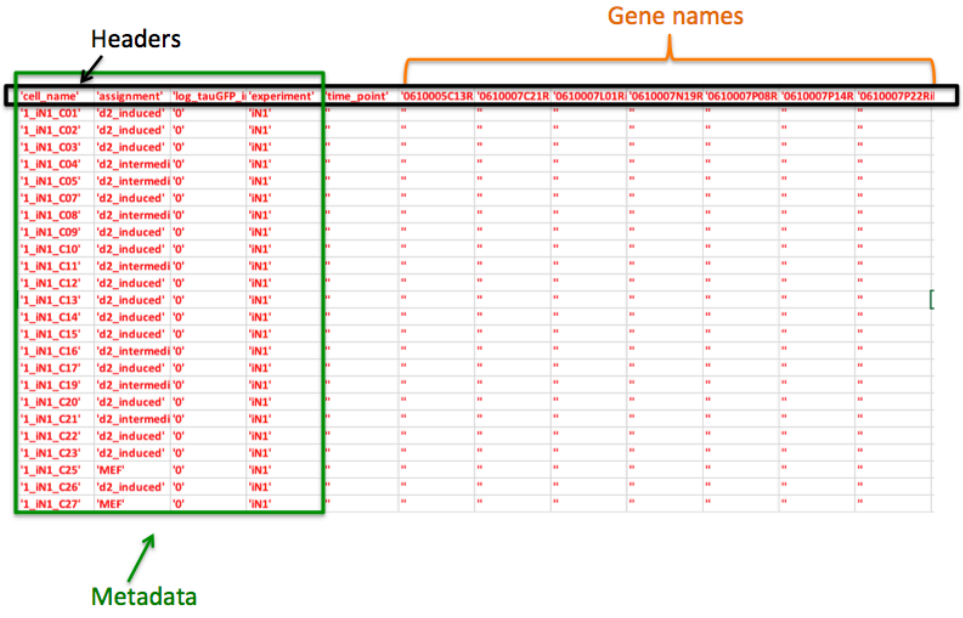


The text file containing the original dataset can be summarized as follows (preview with the first 25 rows and 12 column):



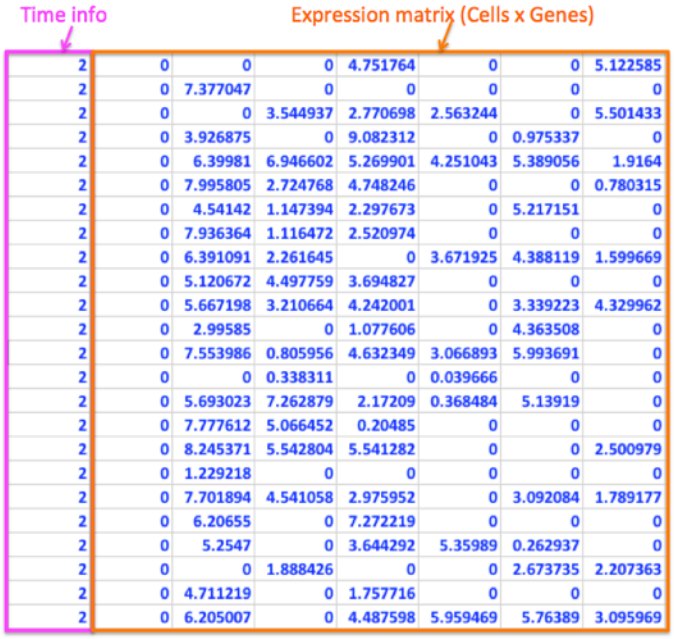
cell_name	assignment	log_tauGFP_int	experiment	time_point	0610005C13Rik	0610007C21Rik	0610007L01Rik	0610007N19Rik	0610007P08Rik	0610007P14Rik	0610007P22Rik
1_iN1_C01	d2_induced	0	iN1	2	0	0	0	4.75176408	0	0	5.122584584
1_iN1_C02	d2_induced	0	iN1	2	0	7.37704663	0	0	0	0	0
1_iN1_C03	d2_induced	0	iN1	2	0	0	3.544937108	2.770697871	2.563243746	0	5.50143267
1_iN1_C04	d2_intermediat	0	iN1	2	0	3.926875324	0	9.082312119	0	0.975336905	0
1_iN1_C05	d2_intermediat	0	iN1	2	0	6.399809926	6.946602177	5.269901346	4.251042833	5.389056375	1.916400044
1_iN1_C07	d2_induced	0	iN1	2	0	7.995805052	2.724768024	4.74824605	0	0	0.780314952
1_iN1_C08	d2_intermediat	0	iN1	2	0	4.541419746	1.147394416	2.297672823	0	5.217150702	0
1_iN1_C09	d2_induced	0	iN1	2	0	7.936363922	1.116471648	2.52097415	0	0	0
1_iN1_C10	d2_induced	0	iN1	2	0	6.391090605	2.26164537	0	3.671925406	4.388118622	1.59966884
1_iN1_C11	d2_intermediat	0	iN1	2	0	5.120672139	4.497759092	3.694826533	0	0	0
1_iN1_C12	d2_induced	0	iN1	2	0	5.667198479	3.21066394	4.242001325	0	3.339223359	4.329961698
1_iN1_C13	d2_induced	0	iN1	2	0	2.995850255	0	1.077606307	0	4.363507545	0
1_iN1_C14	d2_induced	0	iN1	2	0	7.55398606	0.805955813	4.632348754	3.066893067	5.993690988	0
1_iN1_C15	d2_induced	0	iN1	2	0	0	0.338310627	0	0.03966647	0	0
1_iN1_C16	d2_intermediat	0	iN1	2	0	5.693022646	7.262878865	2.17209037	0.36848359	5.139190271	0
1_iN1_C17	d2_induced	0	iN1	2	0	7.777611627	5.066452204	0.204849805	0	0	0
1_iN1_C19	d2_intermediat	0	iN1	2	0	8.245371124	5.54280365	5.541281792	0	0	2.50097859
1_iN1_C20	d2_induced	0	iN1	2	0	1.229217679	0	0	0	0	0
1_iN1_C21	d2_intermediat	0	iN1	2	0	7.701893565	4.541058261	2.975951694	0	3.092083605	1.78917691
1_iN1_C22	d2_induced	0	iN1	2	0	6.206550133	0	7.272218899	0	0	0
1_iN1_C23	d2_induced	0	iN1	2	0	5.254699661	0	3.644292022	5.359889928	0.262936771	0
1_iN1_C25	MEF	0	iN1	2	0	0	1.888426352	0	0	2.673734565	2.207363104
1_iN1_C26	d2_induced	0	iN1	2	0	4.711219374	0	1.757716009	0	0	0
1_iN1_C27	MEF	0	iN1	2	0	6.205006575	0	4.487598366	5.959469132	5.763890139	3.095969378

CALISTA imports the dataset by splitting the text data from the expression (numbers) data. In particular we define the “imported text data” as:



'cell_name'	'assignment'	'log_tauGFP_li	'experiment'	'time_point'	'0610005C13R	'0610007C21R	'0610007L01Ri	'0610007N19R	'0610007P08R	'0610007P14R	'0610007P22Ri
'1_iN1_C01'	'd2_induced'	'0'	'iN1'								
'1_iN1_C02'	'd2_induced'	'0'	'iN1'								
'1_iN1_C03'	'd2_induced'	'0'	'iN1'								
'1_iN1_C04'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C05'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C07'	'd2_induced'	'0'	'iN1'								
'1_iN1_C08'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C09'	'd2_induced'	'0'	'iN1'								
'1_iN1_C10'	'd2_induced'	'0'	'iN1'								
'1_iN1_C11'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C12'	'd2_induced'	'0'	'iN1'								
'1_iN1_C13'	'd2_induced'	'0'	'iN1'								
'1_iN1_C14'	'd2_induced'	'0'	'iN1'								
'1_iN1_C15'	'd2_induced'	'0'	'iN1'								
'1_iN1_C16'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C17'	'd2_induced'	'0'	'iN1'								
'1_iN1_C19'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C20'	'd2_induced'	'0'	'iN1'								
'1_iN1_C21'	'd2_intermedi	'0'	'iN1'								
'1_iN1_C22'	'd2_induced'	'0'	'iN1'								
'1_iN1_C23'	'd2_induced'	'0'	'iN1'								
'1_iN1_C25'	'MEF'	'0'	'iN1'								
'1_iN1_C26'	'd2_induced'	'0'	'iN1'								
'1_iN1_C27'	'MEF'	'0'	'iN1'								

and “imported expression data” as:



2	0	0	0	4.751764	0	0	5.122585
2	0	7.377047	0	0	0	0	0
2	0	0	3.544937	2.770698	2.563244	0	5.501433
2	0	3.926875	0	9.082312	0	0.975337	0
2	0	6.39981	6.946602	5.269901	4.251043	5.389056	1.9164
2	0	7.995805	2.724768	4.748246	0	0	0.780315
2	0	4.54142	1.147394	2.297673	0	5.217151	0
2	0	7.936364	1.116472	2.520974	0	0	0
2	0	6.391091	2.261645	0	3.671925	4.388119	1.599669
2	0	5.120672	4.497759	3.694827	0	0	0
2	0	5.667198	3.210664	4.242001	0	3.339223	4.329962
2	0	2.99585	0	1.077606	0	4.363508	0
2	0	7.553986	0.805956	4.632349	3.066893	5.993691	0
2	0	0	0.338311	0	0.039666	0	0
2	0	5.693023	7.262879	2.17209	0.368484	5.13919	0
2	0	7.777612	5.066452	0.20485	0	0	0
2	0	8.245371	5.542804	5.541282	0	0	2.500979
2	0	1.229218	0	0	0	0	0
2	0	7.701894	4.541058	2.975952	0	3.092084	1.789177
2	0	6.20655	0	7.272219	0	0	0
2	0	5.2547	0	3.644292	5.35989	0.262937	0
2	0	0	1.888426	0	0	2.673735	2.207363
2	0	4.711219	0	1.757716	0	0	0
2	0	6.205007	0	4.487598	5.959469	5.76389	3.095969

CALISTA provides a preview and the dimensions of both imported text and expression data.

```
Command Window
  Text data extracted preview:

     'cell_name'    'assignment'         'log_tauGFP_intens…'    'experiment'    'time_point'    '0610005C13Rik'    '0610007C21Rik'
     '1_iN1_C01'    'd2_induced'         '0'                     'iN1'           ''              ''                 ''
     '1_iN1_C02'    'd2_induced'         '0'                     'iN1'           ''              ''                 ''
     '1_iN1_C03'    'd2_induced'         '0'                     'iN1'           ''              ''                 ''
     '1_iN1_C04'    'd2_intermediate'    '0'                     'iN1'           ''              ''                 ''


  row =

     406


  col =

        22529
```

```
Command Window
  Expression data extracted preview:

     2.0000         0         0         0    4.7518         0         0    5.1226         0         0
     2.0000         0    7.3770         0         0         0         0         0         0         0
     2.0000         0         0    3.5449    2.7707    2.5632         0    5.5014         0         0
     2.0000         0    3.9269         0    9.0823         0    0.9753         0         0         0
     2.0000         0    6.3998    6.9466    5.2699    4.2510    5.3891    1.9164         0         0


  row =

     405


  col =

        22525
```

Based on the expression data preview, we set the starting and ending rows and columns for the expression values: as **[1 405]** and **[2 22525]**, respectively, when queried. We exclude the capture time info in the first column.

```
Command Window
  * Key starting and ending rows for the expression values (e.g. [1 405]): [1 405]
  * Key starting and ending columns for the expression values (e.g. [2 22525]): [2 22525]
```

We press **1** since columns refer genes and rows refer cells.

```
Command Window
  * Press 1 if columns=genes, 0 otherwise: 1
```

We define the gene’s names using the text data preview **[6 22529]** (starting and ending columns).

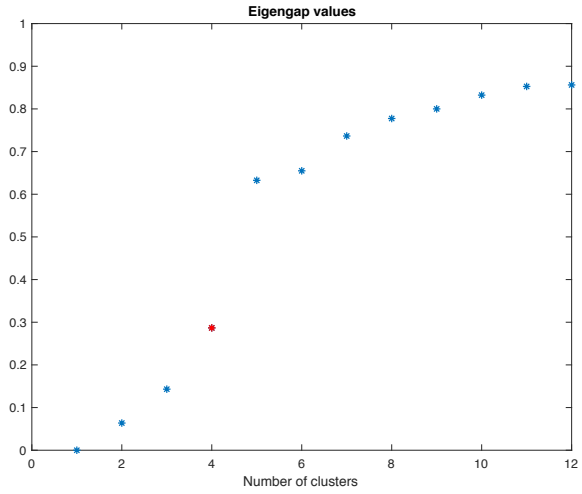
```
Command Window
  * Key starting and ending columns for gene names (e.g. [6 22529]): [6 22529]
```

We load the capture time/cell stage by pressing **1** (i.e. time/cell stage information is in expression data matrix is) and selecting column **1** in the data matrix.

```
Command Window
  * Add time info (1-Yes, 0-No): 1
  * Key the column or row vector (e.g 1) of the EXPRESSION DATA MATRIX with time / cell stage info: 1
```

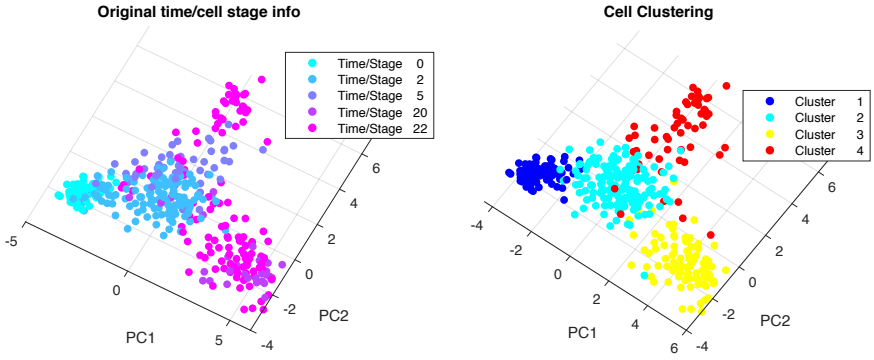
4.3.2 Single-cell clustering

In this case, the number of clusters is determined using the eigengap plot. According to the eigengap plot below, we set the number of clusters to 4.




```
Command Window
 Number of clusters expected: 4
Starting parallel pool (parpool) using the 'local' profile ...
connected to 6 workers.

CALISTA_clustering is running...
Progress:
.................................................
||||||||||||||||||||||||||||||||||||||||||||||||||

Plotting...
```

CALISTA single-cell clustering results are as follow.

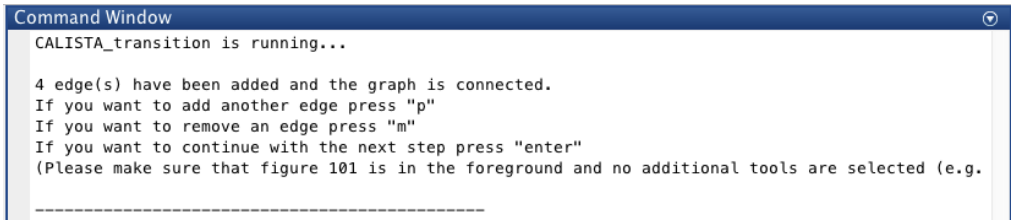


We do not need to remove any cluster (by entering **0** upon queried), and continue with further analysis (by entering **1** upon queried):

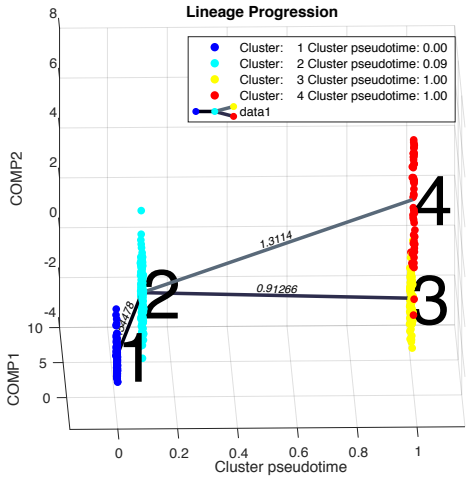
```
Command Window
Plotting...
Press 1 if you want to remove one cell cluster, 0 otherwise: 0
 Press 1 if you want to perform additional analysis (e.g. lineage inference, cell ordering) , 0 otherwise: 1
```

4.3.3 Reconstruction of lineage progression

During the lineage inference step, CALISTA automatically generates and displays a lineage graph, obtained by adding an edge between two clusters in increasing cluster distances, until all clusters are connected to at least one other cluster. Subsequently, users can manually add or remove one edge at time based on the cluster distances.

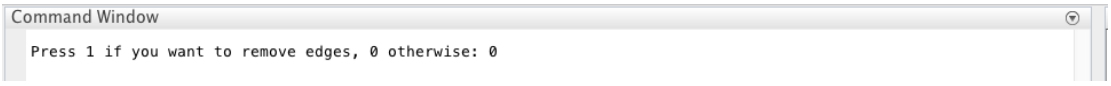


ATTENTION: to add an edge (press "**p**"), remove an edge (press "**m**") or finalize the lineage progression graph (press "**enter**"), the MATLAB figure of the graph must appear in foreground without any modification (e.g., zooming, rotation). Note that the addition/removal of the edges are performed according to increasing/decreasing order of cluster distance.



ATTENTION: the final graph must be connected (i.e. there is a path from any node/cluster to any other node/cluster in the graph) otherwise a warning will be returned.

Here, we do not need to remove any spurious edges, and hence we enter **0** upon queried.



In addition, CALISTA returns (not shown):

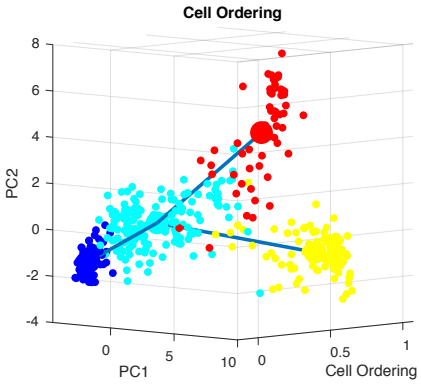
- Cell clustering plot based on the cluster pseudotime
- Boxplot, mean, median entropy values calculated for each cluster
- Plot of mean expression values for each gene based on cell cluster expression level

4.3.4 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph (results not shown here), based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.

4.3.5 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.



4.4 Example 4. Human embryonic stem cell differentiation into endodermal cells

Analysis of RNA-seq data in Chu et al., Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm, *Genome Biol.* **17,** 173 (2016).

Please unzip the file "4-CHU__data_type_4_format_data_2_clusters_4.csv.zip**" in **EXAMPLES/RNA-seq/** before running CALISTA**

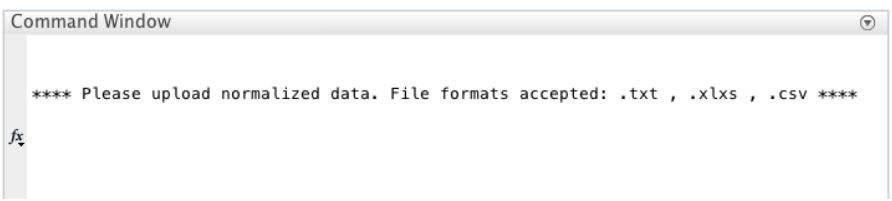
4.4.1 Data Import and Preprocessing

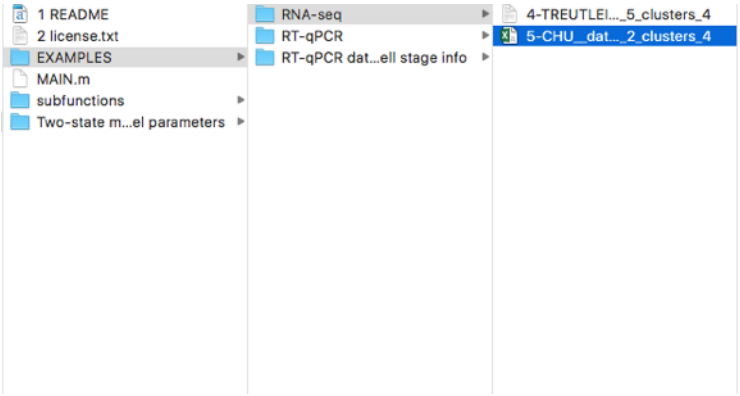
We first change the current directory in MATLAB to the CALISTA folder.

We edit the `MAIN.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=4; % Single-cell RNA-seq data
INPUTS.format_data=2; % Rows= genes and Columns= cells with time/stage info in the
first row
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=90; % Remove genes with > 90% of zeros
INPUTS.perczeros_cells=100; % Remove cells with 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells
INPUTS.perc_top_genes=10; % Retain only top X the most variable genes with X=min(200,
INPUTS.perc_top_genes * num of cells/100, num of genes)

% Specify single-cell clustering settings
INPUTS.optimize=1; % Set the number of clusters based on Eigengap Heuristics
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=75; % Set threshold for transition genes determination to
75%
```

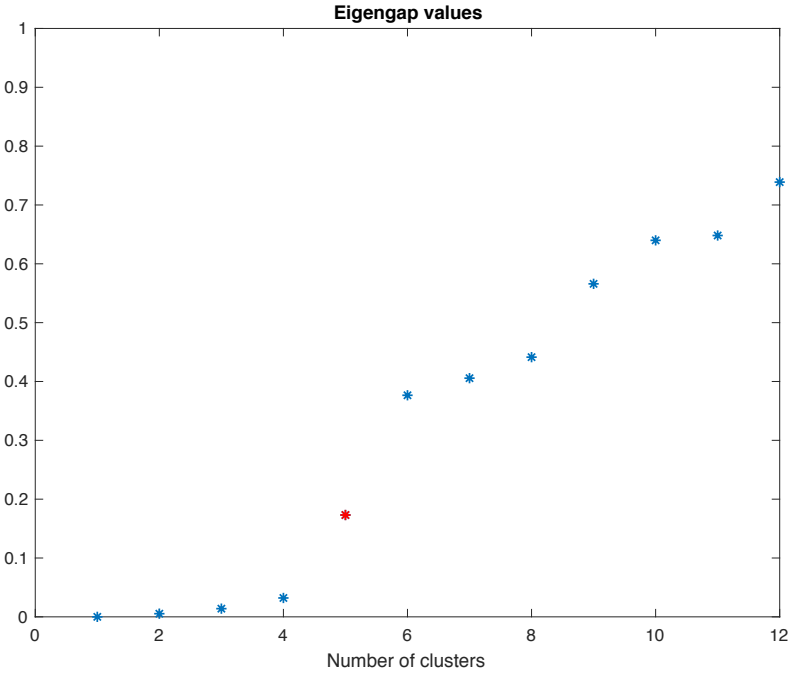
We then run `MAIN.m` from the workspace and import Chu dataset (in subfolder EXAMPLES/RNA-seq):



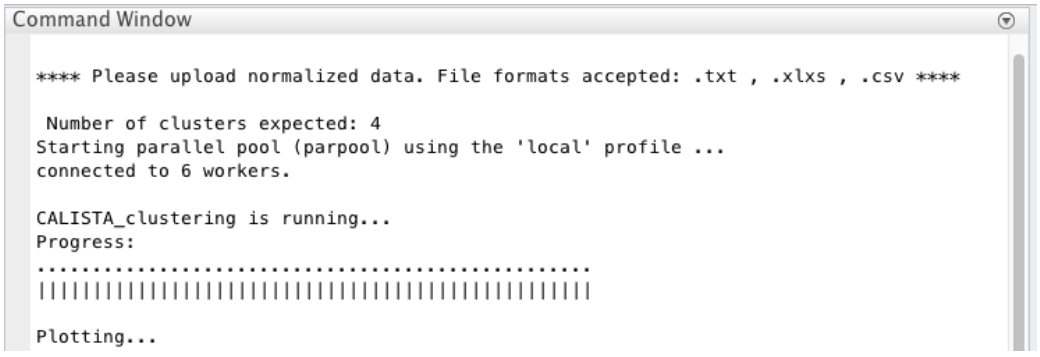


4.4.2 Single-cell clustering

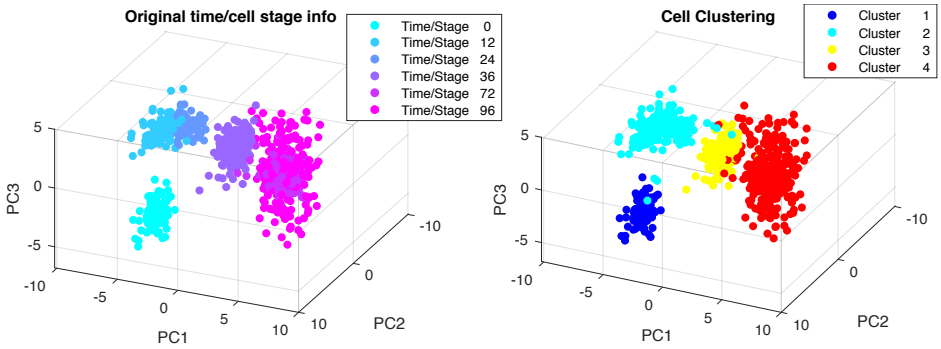
In this case, the number of clusters is determined using the eigengap plot. According to the eigengap plot below, we set the number of clusters to 4.



NOTE: CALISTA automatically returns the optimal number of clusters based on the MAXIMUM eigengap value. However, the user might choose the number of clusters to adopt based on the FIRST eigengap.



CALISTA single-cell clustering result is shown below.



We do not need to remove any clusters (by entering **0** upon queried), and continue with further analysis (by entering **1** upon queried).

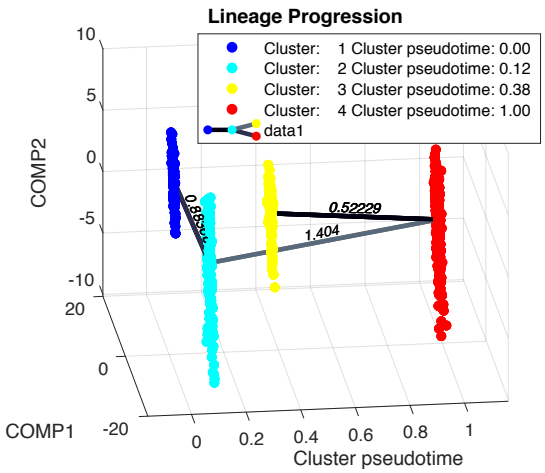
```
Command Window
Plotting...
Press 1 if you want to remove one cell cluster, 0 otherwise: 0
 Press 1 if you want to perform additional analysis (e.g. lineage inference, cell ordering) , 0 otherwise: 1
```

4.4.3 Reconstruction of lineage progression

During the lineage inference step, CALISTA automatically generates and displays a lineage graph, obtained by adding an edge between two clusters in increasing cluster distances, until all clusters are connected to at least one other cluster. Subsequently, users can manually add or remove one edge at time based on the cluster distances.

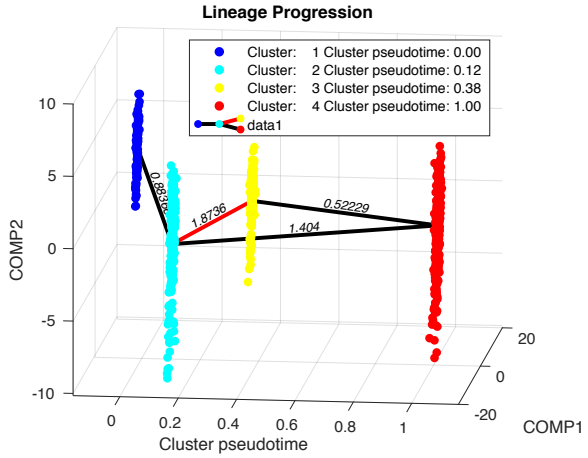
```
Command Window

CALISTA_transition is running...

3 edge(s) have been added and the graph is connected.
If you want to add another edge press "p"
If you want to remove an edge press "m"
If you want to continue with the next step press "enter"
(Please make sure that figure 101 is in the foreground and no additional tools are sele

------------------------------------------------
```

ATTENTION: to add an edge (press "**p**"), remove an edge (press "**m**") or finalize the lineage progression graph (press "**enter**"), the MATLAB figure of the graph must appear in foreground without any modification (e.g., zooming, rotation). Note that the addition/removal of the edges are performed according to increasing/decreasing order of cluster distance.

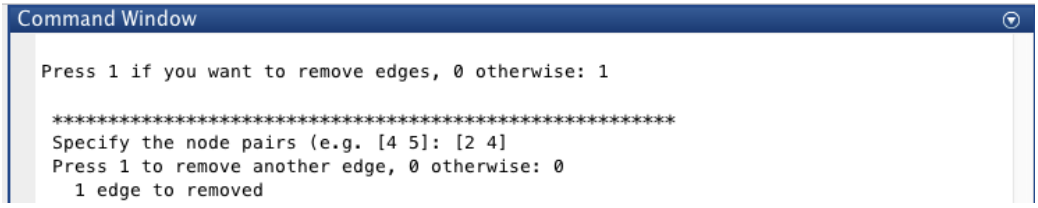


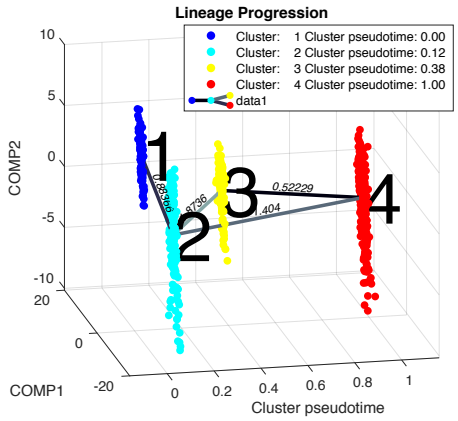
ATTENTION: the final graph must be connected (i.e. there is a path from any node/cluster to any other node/cluster in the graph), otherwise a warning will be returned.

Since the transition from cluster 4 to cluster 3 is inconsistent with the capture time info (i.e. cluster pseudotime values for cluster 4 and 3 are 1 and 0.38 respectively), we add one further edge to produce the following lineage progression graph by pressing "**p**" and then "**enter**":

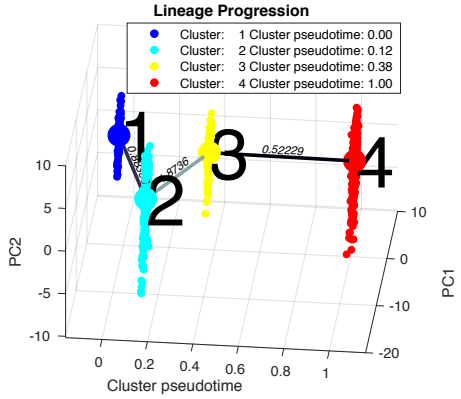


Based on our definition of branching point, we consider the previous inferred lineage graph still linear, since there is only one final cell cluster (cluster 4). Moreover, since the transition from cluster 2 to cluster 4 bypasses a cluster with intermediate pseudotime (i.e. cluster 3), we remove the spurious edge between cluster 2 and 4, by entering **1** and entering **[2 4]**, upon the following query:





The final inferred lineage relationships is shown below.



In addition, CALISTA returns (not shown):

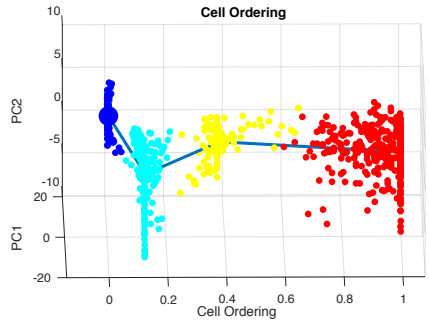
- Cell clustering plot based on the cluster pseudotime
- Boxplot, mean, median entropy values calculated for each cluster
- Plot of mean expression values for each gene based on cell cluster expression level

4.4.4 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph (results not shown here), based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.

4.4.5 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.



4.5 Example 5. Running CALISTA without time or cell stage information

*Analysis of RT-qPCR data in **Moignard et al.** "Characterization of transcriptional networks in blood stem and progenitor cells using high-throughput single-cell gene expression analysis". Nat. Cell Biol. 15, 363–72 (2013).*

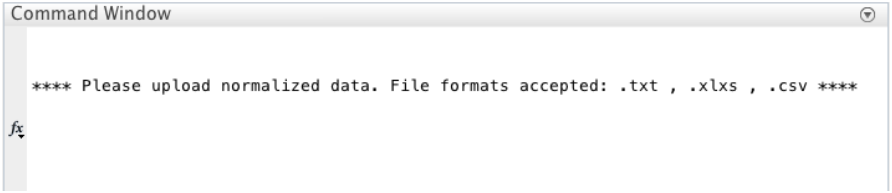
Here, we report only the main steps of the analysis. For the complete analysis please check **Example 4.2**.

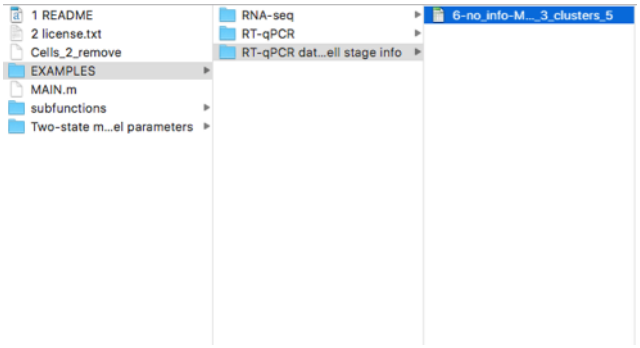
4.5.1 Data Import and Preprocessing

We change the current directory in MATLAB to the CALISTA folder.
We then edit the `MAIN.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=1; % Single-cell RT-qPCR CT data
INPUTS.format_data=3; % Rows= cells and Columns= genes (no time/stage info)
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=100; % Remove genes with > 100% of zeros
INPUTS.perczeros_cells=100; % Remove cells with 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells

% Specify single-cell clustering settings
INPUTS.perc_top_genes=100; % Retain only top X the most variable genes with X=min(200,
INPUTS.perc_top_genes * num of cells/100, num of genes)
INPUTS.optimize=0; % The number of cluster is known a priori
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=50; % Set threshold for transition genes determination to
50%
```

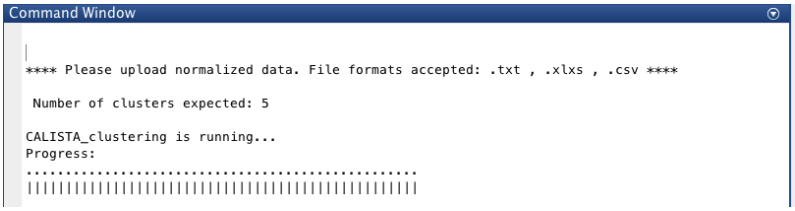
We subsequently run `MAIN.m` from the workspace and load Moignard dataset (in subfolder EXAMPLES/RT-qPCR):





4.5.2 Single-cell clustering

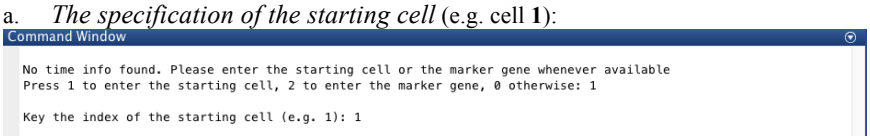
Following the original publication, we set the number of clusters equals to **5**.



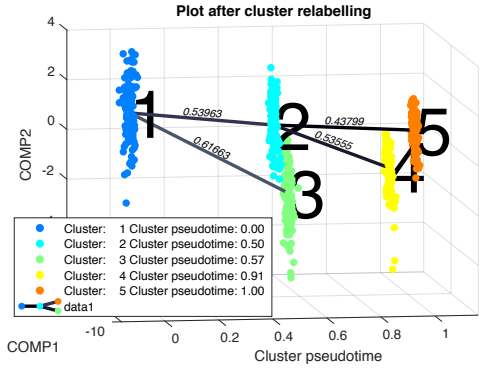
4.5.3 Reconstruction of lineage progression and pseudotemporal ordering of cells

We follow the steps as outlined in the other examples above to infer the lineage progression and carry out pseudotemporal ordering of single cells.
Without the time or cell stage info, CALISTA is still able to recover the cluster progression based on:

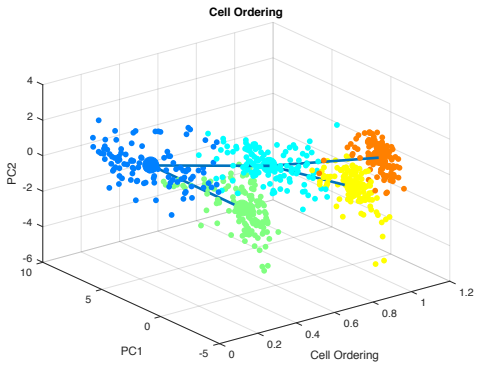
a. *The specification of the starting cell* (e.g. cell **1**):



The final inferred lineage relationships are as follow.



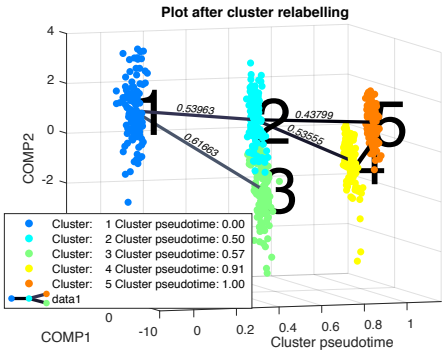
CALISTA pseudotemporal ordering gives the following outcome.



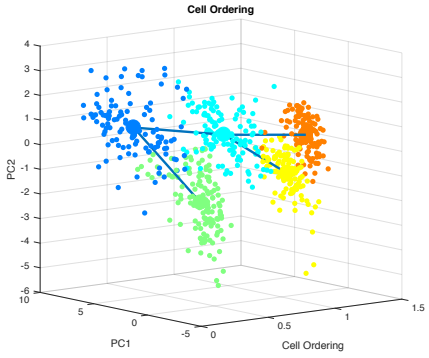
b. *The specification of a marker gene* (e.g. **'Erg'**) which is downregulated (press **2**):

```
Enter the marker gene chosen (e.g. 'Erg', case insensitive): 'Erg'

Press 1 if it is upregulated, 2 if it is downregulated (e.g. 2): 2
```

The final inferred lineage relationships:



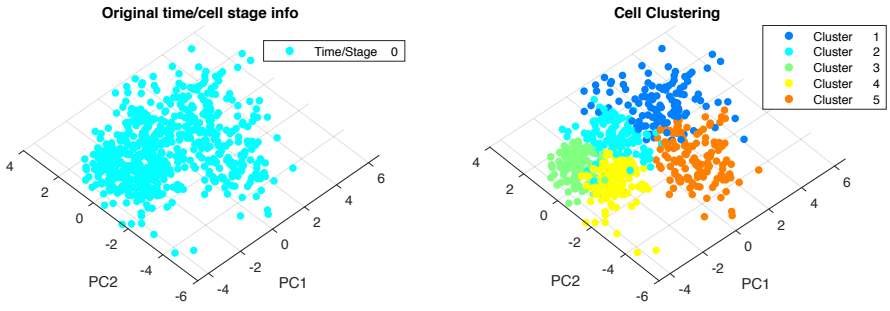
CALISTA pseudotemporal ordering of cells gives the following result.



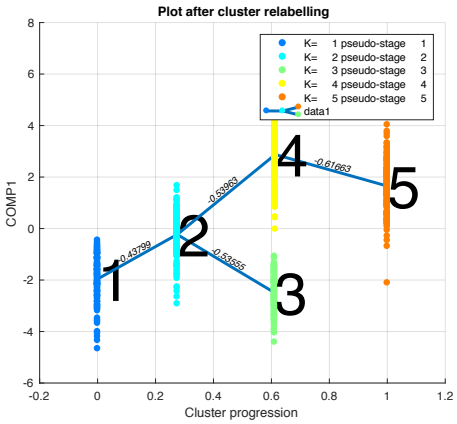
Without any information of the time information, cell stage, starting cell and marker genes, CALISTA is still able to find the topology of the lineage graph, but the edges are undirected.

```
Command Window

**** Please upload normalized data. File formats accepted: .txt , .xlxs , .csv ****

 Number of clusters expected: 5

CALISTA_clustering is running...
Progress:
.................................................
||||||||||||||||||||||||||||||||||||||||||||||||||

No time info found. Please enter the starting cell or the marker gene whenever available
Press 1 to enter the starting cell, 2 to enter the marker gene, 0 otherwise: 0

Skip relabelling step
```

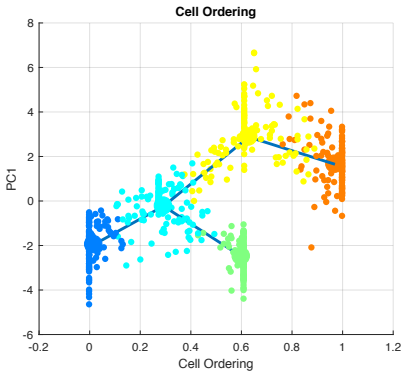
CALISTA single-cell clustering result is as follows.



The final inferred lineage relationships are shown below.



Therefore, CALISTA performs the pseudotemporal ordering of cells as follows:



4.6 Example 6. Removing undesired clusters

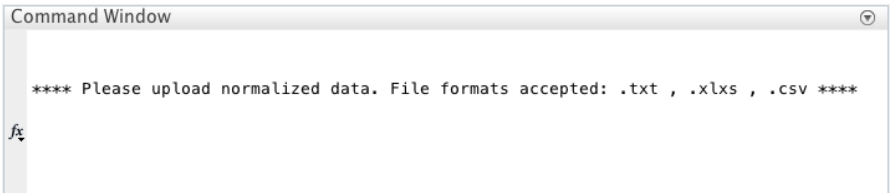
Analysis of RT-qPCR data in Moignard et al., Characterization of transcriptional networks in blood stem and progenitor cells using high-throughput single-cell gene expression analysis, *Nat. Cell Biol.* **15**, 363–72 (2013).

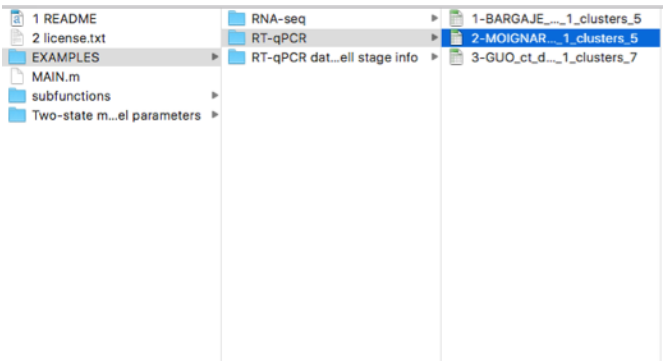
4.6.1 Data Import and Preprocessing

We change the current directory in MATLAB to the CALISTA folder.
We edit the `MAIN.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=1; % Single-cell RT-qPCR CT data
INPUTS.format_data=1; % Rows= cells and Columns= genes with time/stage info in the
last column
INPUTS.data_selection=[]; % Include data from all time points
INPUTS.perczeros_genes=100; % Remove genes with > 100% of zeros
INPUTS.perczeros_cells=100; % Remove cells with 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells
% Specify single-cell clustering settings
INPUTS.perc_top_genes=100; % Retain only top X the most variable genes with X=min(200,
INPUTS.perc_top_genes * num of cells/100, num of genes)
INPUTS.optimize=0; % The number of cluster is known a priori
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering
% Specify transition genes settings
INPUTS.thr_transition_genes=50; % Set threshold for transition genes determination to
50%
```

We then run `MAIN.m` from the workspace and load Moignard dataset (in subfolder EXAMPLES/RT-qPCR):



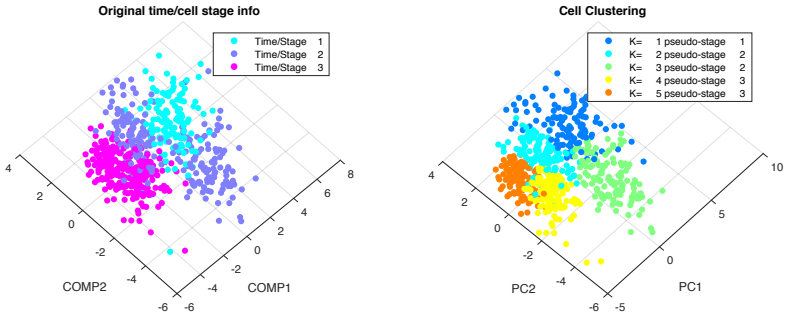


4.6.2 Single-cell clustering

We set the number of clusters equals to **5** following the original publication.

```
Command Window

**** Please upload normalized data. File formats accepted: .txt , .xlxs , .csv ****

 Number of clusters expected: 5
Starting parallel pool (parpool) using the 'local' profile ...
connected to 6 workers.

CALISTA_clustering is running...
Progress:
.................................................
||||||||||||||||||||||||||||||||||||||||||||||||||
```

CALISTA single-cell clustering result is shown below.



Let us proceed with removing cluster 3 and 5, by entering **1** and type **[5 3]** upon queried.

```
Command Window
  Press 1 if you want to remove one cell cluster, 0 otherwise: 1
  Key cluster numbers (e.g 1 or [5 3]): [5 3]
```

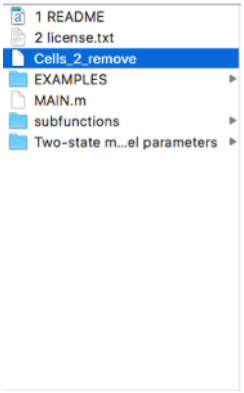
The indices of cells to remove are saved in a csv file.

```
Command Window
  Cell's indices to remove are saved in "Cells_2_remove.csv". Please run MAIN script again
```

We then edit `MAIN.m` script again, but we set the `INPUTS` as described previously except:

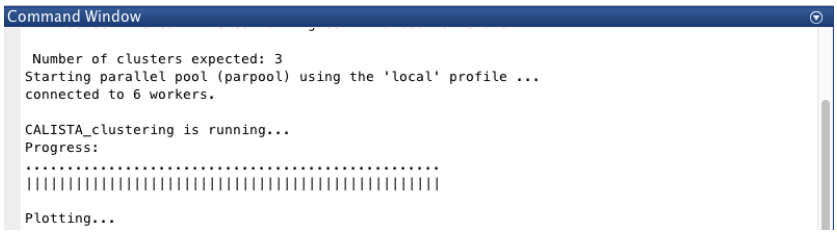
```
INPUTS.cells_2_cut=0; % Manual removal of cells
```

We run `MAIN.m` once more from the workspace and import Moignard dataset (in subfolder EXAMPLES/RT-qPCR). We also upload the csv file containing cell's indices to remove.

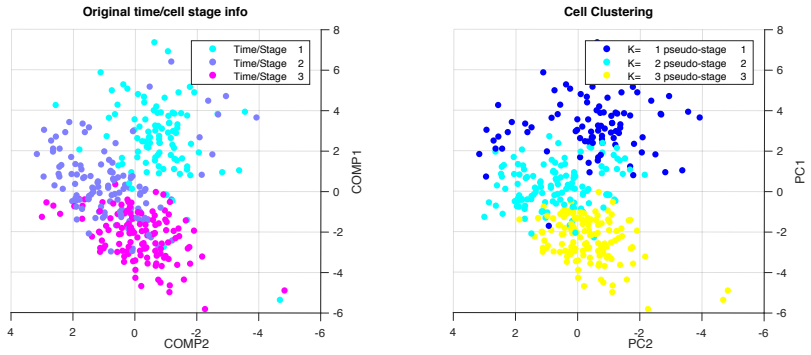


4.6.3 Single-cell clustering after removing undesired clusters

We now set the number of clusters equals to **3**.

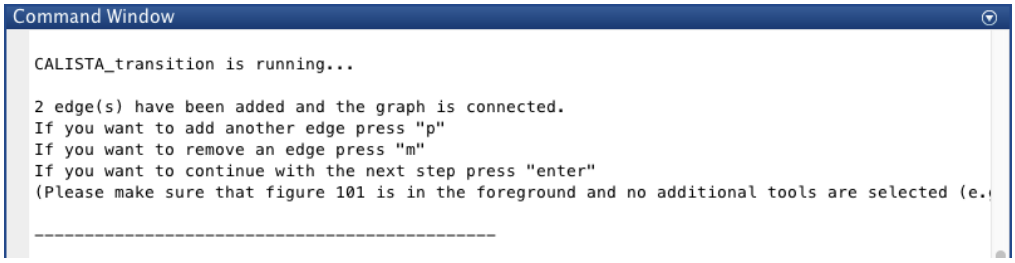


We obtain the following clustering results.



4.6.4 Reconstruction of lineage progression

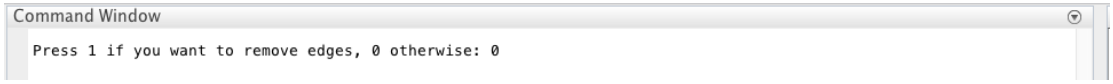
We continue with lineage inference step. During the lineage inference step, CALISTA provides the minimal connected graph (with nodes = cell clusters and edges = state transitions) as starting prediction for the developmental hierarchy. In addition, the user can also manually add or remove one edge at time based on the cluster distance values:



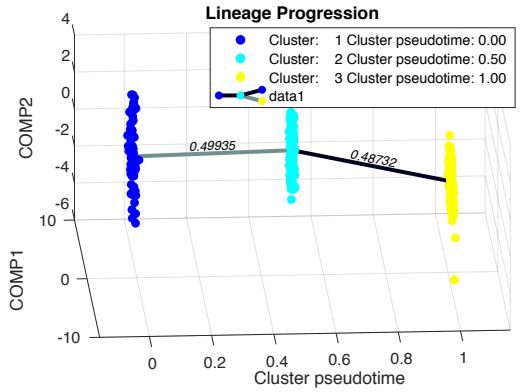
ATTENTION: to add an edge (press "**p**"), remove an edge (press "**m**") or finalize the lineage progression graph (press "**enter**"), the MATLAB figure of the graph must appear in foreground without any modification (e.g., zooming, rotation). Note that the addition/removal of the edges are performed according to increasing/decreasing order of cluster distance.

ATTENTION: the final graph must be connected (i.e. there exists a path from any node/cluster to any other node/cluster in the graph), otherwise a warning will be returned.

We do not need to remove spurious edges (entering **0** upon queried)



The final inferred lineage relationship is shown below



In addition, CALISTA returns (not shown):

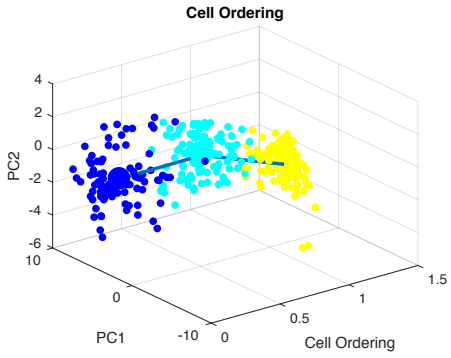
- Cell clustering plot based on the cluster pseudotime
- Boxplot, mean, median entropy values calculated for each cluster
- Plot of mean expression values for each gene based on cell cluster expression level

4.6.5 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph (results not shown here), based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.

4.6.6 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.



4.7 Running CALISTA GUI

Analysis of RT-qPCR data of Bargaje et al. (Bargaje, et al, Cell population structure prior to bifurcation predicts efficiency of directed differentiation in human induced pluripotent cells. *Proc. Natl. Acad. Sci. U. S. A*. 114, 2271–2276 (2017)).

Here, we report only the main steps of the analysis. For the complete analysis please check **Example 4.1**.

4.7.1 Data Import and Preprocessing

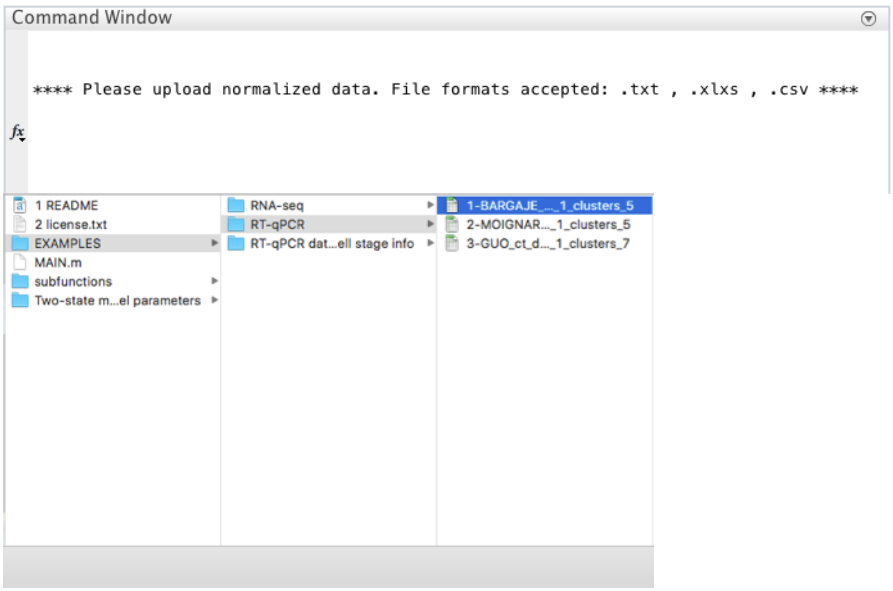
We begin with changing the current directory in MATLAB to the CALISTA folder. Then, we edit the `MAIN_GUI.m` script in the main folder of CALISTA and set the fields of `INPUTS` as follows:

```
% Specify data types and settings for pre-processing
INPUTS.data_type=1; % Single-cell RT-qPCR CT data
INPUTS.format_data=1; % Rows= cells and Columns = genes with time/stage info in the
last column
INPUTS.data_selection=[]; % Include data from all time points
```

```
INPUTS.perczeros_genes=100; % Remove genes with >= 100% of zeros
INPUTS.perczeros_cells=100; % Remove cells with >= 100% of zeros
INPUTS.cells_2_cut=0; % No manual removal of cells

% Specify single-cell clustering settings
INPUTS.perc_top_genes=100; % Retain only top X of the most variable genes with X =
min(200, INPUTS.perc_top_genes * number of cells/100, number of genes)
INPUTS.optimize=1; % Set the number of clusters based on Eigengap Statistics
INPUTS.parallel=1; % Use parallelization option
INPUTS.runs=50; % Perform 50 independent runs of greedy algorithm in clustering
INPUTS.max_iter=100; % Limit the number of iterations in greedy algorithm to 100
INPUTS.cluster='kmedoids'; % Use k-medoids in consensus clustering

% Specify transition genes settings
INPUTS.thr_transition_genes=50; % Set threshold for transition genes determination to
50 percent

% Specify path analysis settings
INPUTS.plot_fig=1; % Plot figure of smoothened gene expression along path
INPUTS.hclustering=1; % Perform hierarchical clustering of gene expression for each
path
INPUTS.method=2; % Use pairwise correlation for the gene co-expression network
(value_cutoff=0.8, pvalue_cutoff=0.01)
INPUTS.moving_average_window=10; % Set the size of window (percent of cells in each
path) used for the moving averaging
```

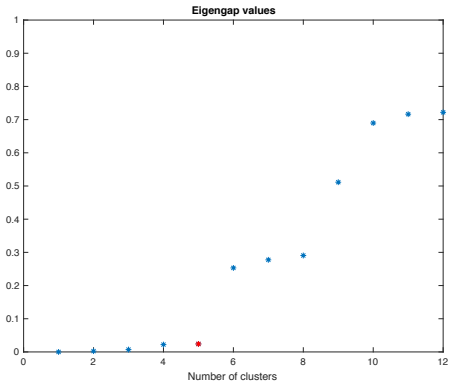
We subsequently run `MAIN_GUI.m` and import Bargaje dataset (available in the subfolder EXAMPLES/RT-qPCR).

The following are screenshots from running CALISTA on MATLAB.



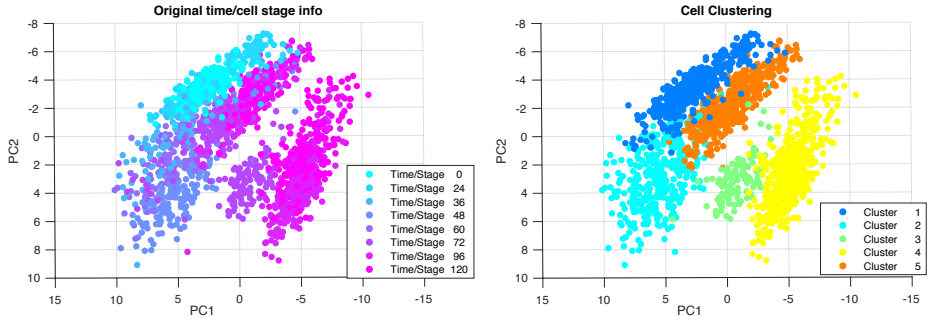
4.7.2 Single-cell clustering

In this case, the number of clusters is determined using the eigengap plot. According to the eigengap plot below, we set the number of clusters to **5**. The following are screenshots from CALISTA single-cell clustering analysis.




```
Command Window

**** Please upload normalized data. File formats accepted: .txt , .xlxs , .csv ****

Starting parallel pool (parpool) using the 'local' profile ...
connected to 6 workers.

CALISTA_clustering is running...
Progress:
.................................................
|||||||||||||||||||||||||||||||||||||||||||||||||

Optimal number of cluster according to max. eigenvalue 5:
If you want to use this value press enter,
else provide desired number of cluster: 5
```

```
CALISTA_clustering is running...
Progress:
.................................................
|||||||||||||||||||||||||||||||||||||||||||||||||

Plotting...
```

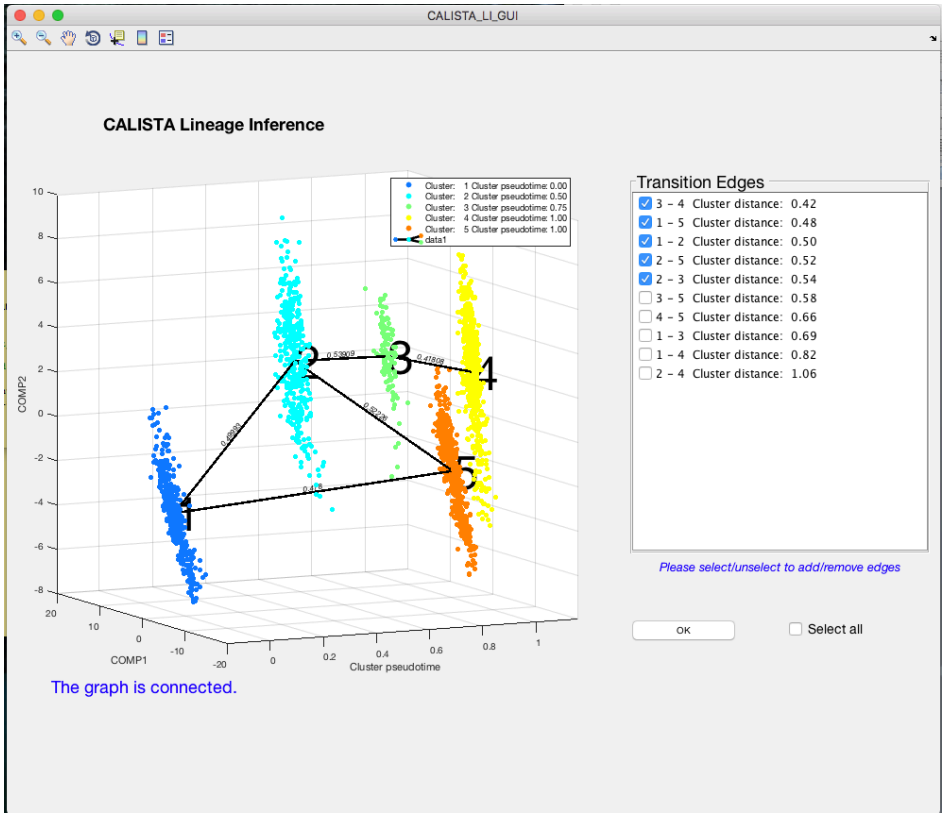


If desired, users can remove cells from specific clusters from further analysis. In this example, we do not want to remove any clusters. Hence, we enter **0** (no cluster removal) and then **1** to proceed with lineage inference.

```
Command Window
Plotting...
Press 1 if you want to remove one cell cluster, 0 otherwise: 0
 Press 1 if you want to perform additional analysis (e.g. lineage inference, cell ordering) , 0 otherwise: 1
```

4.7.3 Reconstruction of lineage progression

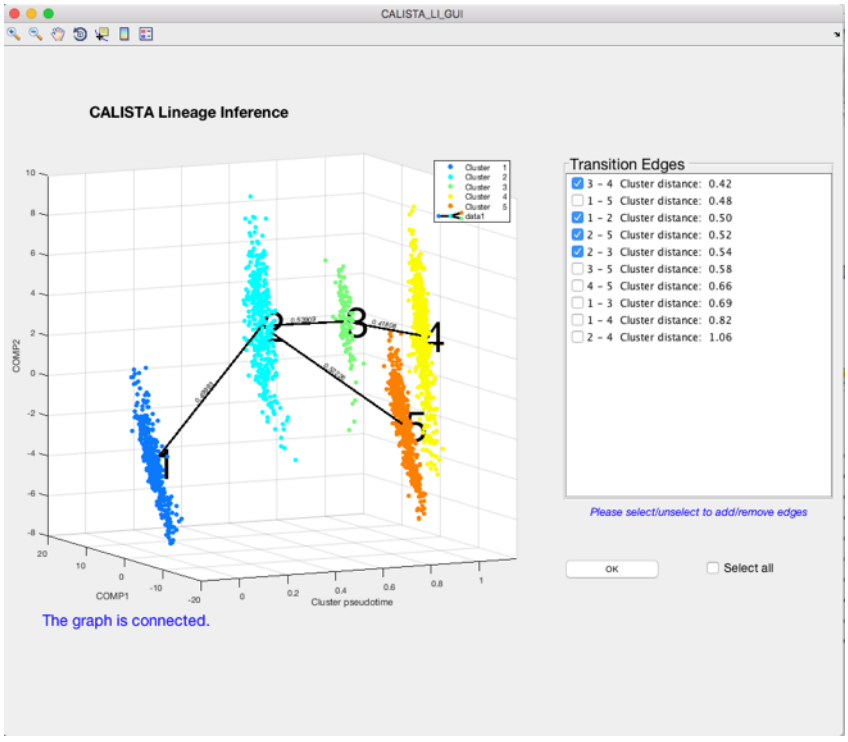
During the lineage inference step, CALISTA automatically generates and displays a lineage graph, obtained by adding an edge between two clusters in increasing cluster distances, until all clusters are connected to at least one other cluster. Subsequently, users can manually add or remove one edge at time based on the cluster distances.



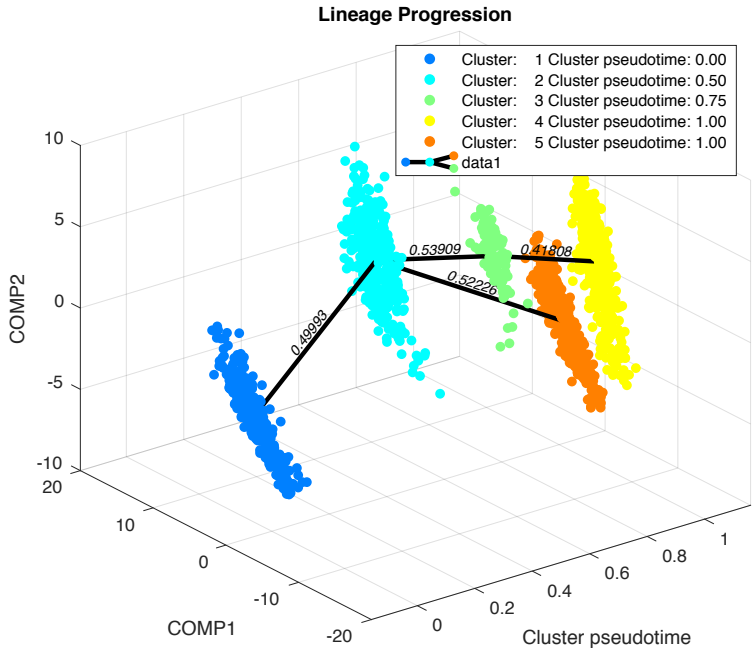
ATTENTION: select (or unselect) checkboxes to add (or remove) specific edges. To select (or unselect) all edges use the "Select all" checkbox.

ATTENTION: to finalize the lineage progression (by pressing the "OK" button), the final graph must be connected (i.e. there exists a path from any node/cluster to any other node/cluster in the graph).

Since the transition from cluster 1 to cluster 5 is inconsistent with the capture time info (i.e. cluster pseudotime values for cluster 1 and 5 are 0 and 1 respectively) we remove the spurious edge between cluster 1 and 5, by unselecting the second checkbox:



We press the “OK” button to confirm and the final inferred lineage relationships are displayed below.



In addition, CALISTA gives (not shown):

- Cell clustering plot based on the cluster pseudotime
- Boxplot, mean, median entropy values calculated for each cluster
- Plot of mean expression values for each gene based on cell cluster expression level

4.7.4 Determination of transition genes

After reconstructing the lineage progression, we identify the key transition genes for any two connected clusters in the graph (results not shown here), based on the gene-wise likelihood difference between having the cells separately as two clusters and together as a single cluster. Larger differences in the gene-wise likelihood point to more informative genes. The transition genes are selected as those whose gene-wise likelihood differences make up to more than a certain percentage of the cumulative sum of the likelihood differences of all genes – set by `INPUTS.thr_transition_genes`.

4.7.5 Pseudotemporal ordering of cells

For pseudotemporal ordering of cells, CALISTA performs maximum likelihood optimization for each cell using a linear interpolation of the cell likelihoods between any two connected clusters. The pseudotimes of the cells are computed by linear interpolation of the cluster pseudotimes, and correspond to the maximum point of the likelihood optimization above. Cells are subsequently assigned to the edges in the lineage progression graph. The following screenshot gives the results of this cell-to-edge assignment.

5 Questions and comments

Please address any problem or comment to: nanp@ethz.ch or rudi.gunawan@chem.ethz.ch.

6 Change log