SEPA: Single-Cell Gene Expression Pattern Analysis

Zhicheng Ji

Johns Hopkins University,
Baltimore, Maryland, USA
zji4@jhu.edu

Hongkai Ji

Johns Hopkins University,
Baltimore, Maryland, USA
hji@jhsph.edu

October 15, 2015

Contents

1 Introductions

Single-cell high-throughput transcriptomics is a powerful approach to study the heterogeneity of gene expression activities on single-cell level. Compared to traditional bulk transcriptomics experiments, single-cell RNA-seq significantly increase the resolution of gene expression and thus can lead to many new biological discoveries. Many of the single-cell transcriptomics researches focus on the differentiation of various cell types and hope to find how the expression of genes and key regulatory factors changes over the differentiation process [1,2]. These single-cell RNA-seq data often contain the gene expression profiles of cells extracted from multiple experimental time points. There is also existing computatioin tool such as Monocle [3] that uses unsupervised machine learning methods to order whole-transcriptome profiles of single cells along 'pseudotime',

a hypothesized time course which can quantitatively measure the real biological process of differentiation. This pseudotime course is then used to study how gene expressions change over the differentiation process. Such pseudo time cell ordering concept provides a novel method of exploring single-cell RNA-seq data. If one has available true experimental time or pseudo temporal cell ordering information, a natural question to ask is what expression patterns do these genes have along the true or pseudo time axis. The expression patterns could be constant, monotonic change, or some transition patterns like first increasing followed by decreasing in gene expressions. It would be even more interesting to investigate what kind of biological functions do genes with similar expression patterns share. These results may provide deeper insights into the biological process and guidelines for researchers to study the function of individual genes. Such kind of analysis is unique for single-cell RNA-seq data, since the bulk RNA-seq experiments usually do not have enough sample size to obtain a robust estimate of the patterns while single-cell RNA-seq data often contain hundreds of cells for each experiment time point. Pseudo time cell ordering is also a unique feature of single-cell RNA-seq.

However, currently there is no available computational tool for such kind of down stream analysis. In Monocle [3], gene expression patterns are clustered and identified manually, which is somehow arbitrary. Thus we propose SEPA: a systematic and comprehensive tool to study the gene expression patterns for single-cell RNA-seq. SEPA is designed for general single-cell RNA-seq data. SEPA takes input true experiement time or pseudo temporal cell ordering information. SEPA first computationally identify the gene expression patterns using a set of t-tests (for true experiment time) or segmented linear regression (for pseudo temporal cell ordering). SEPA then performs GO analysis on genes with similar expression patterns. A special kind of GO analysis with moving windows is also available for pseudo temporal cell ordering information. Finally, users also have the option to identify genes with different gene expression patterns in true experiment time or pseudo-time axis. This function is important if users want to find genes with so-called "Simpsons Paradox".

SEPA comes with a powerful Graphical User Interface written in R shiny package. It allows users without prior programming knowledge to conveniently identify and explore the gene expression patterns. SEPA has an online user interface which is freely available at https://zhiji.shinyapps.io/SEPA/. However, the online user interface only allows one concurrent user so it is recommended that users launch SEPA on their own computers.

2 Identify and Visualize Pattern for True Experiment Time Points

The function truetimepattern can be used to identify gene expression patterns given true experiment time points. The output of the function will be a named vector of patterns. Note that appropriate transformations on gene expression

matrix (e.g. log2 transformation) should be done before calling this function.

```
library(SEPA)

## Loading required package:  DBI
## Warning:  replacing previous import by 'graph::.__C__dist' when loading
'topGO'

library(topGO)

## Loading required package:  BiocGenerics
## Loading required package:  parallel
##
## Attaching package:  'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##    clusterExport, clusterMap, parApply, parCapply, parLapply,
##    parLapplyLB, parRapply, parSapply, parSapplyLB
##
## The following objects are masked from 'package:stats':
##
##    IQR, mad, xtabs
##
## The following objects are masked from 'package:base':
##
##    anyDuplicated, append, as.data.frame, as.vector, cbind,
##    colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##    grep, grepl, intersect, is.unsorted, lapply, lengths, Map,
##    mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##    pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##    setdiff, sort, table, tapply, union, unique, unlist, unsplit
##
## Loading required package:  graph
## Loading required package:  Biobase
## Welcome to Bioconductor
##
##    Vignettes contain introductory material; view with
##    'browseVignettes()'.  To cite Bioconductor, see
##    'citation("Biobase")', and for packages 'citation("pkgname")'.
##
## Loading required package:  GO.db
## Loading required package:  AnnotationDbi
## Loading required package:  stats4
## Loading required package:  IRanges
## Loading required package:  S4Vectors
```

```
##
## Loading required package:  SparseM
##
## Attaching package:  'SparseM'
##
## The following object is masked from 'package:base':
##
##    backsolve
```

```
##
## Attaching package:  'topGO'
##
## The following object is masked from 'package:IRanges':
##
##    members
```

```
library(ggplot2)
library(org.Hs.eg.db)
data(HSMMdata)
truepattern <- truetimepattern(HSMMdata,truetime)
head(truepattern)
```

Use patternsummary function to check the number of genes for each pattern.

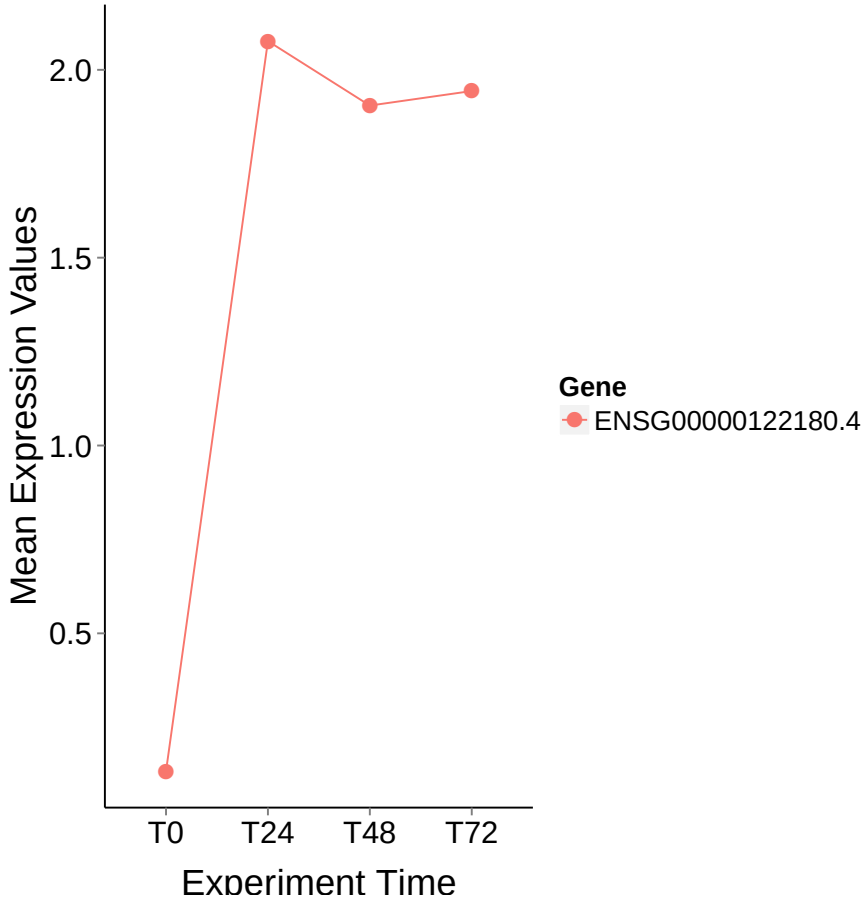
```
patternsummary(truepattern)
```

```
##                   Pattern Number
## 1                constant    129
## 2           constant_down      1
## 3  constant_down_constant      6
## 4             constant_up      2
## 5    constant_up_constant     17
## 6           down_constant    196
## 7      down_constant_down      1
## 8        down_constant_up      4
## 9                 down_up      1
## 10       down_up_constant      3
## 11            up_constant    146
## 12       up_constant_down      1
```

```
## 13          up_constant_up       1
## 14       up_down_constant       9
## 15             up_down_up       1
```

Visualize the mean gene expression using the pseudotimevisualize function.

```
truetimevisualize(HSMMdata,truetime,"ENSG00000122180.4")

## Warning in if (mode == "mean") {:  the condition has length > 1
and only the first element will be used
## Warning in if (mode == "mean") {:  the condition has length > 1
and only the first element will be used
## Warning in if (mode == "mean") {:  the condition has length > 1
and only the first element will be used
## Warning in if (mode == "mean") {:  the condition has length > 1
and only the first element will be used
```



3 Identify and Visualize Pattern for Pseudo Temporal Cell Ordering

The function pseudotimepattern can be used to identify gene expression patterns given pseudo-time. The pseudo-time can be obtained by the Monocle package. Note that if there are multiple cell paths, users should perform the analysis one path at a time. The output of the function will be a list. Note that appropriate transformations on gene expression matrix (e.g. log2 transformation) should be done before calling this function.

```
data(HSMMdata)
pseudopattern <- pseudotimepattern(HSMMdata,pseudotime)

## [1] "Running davies test"
## [1] "Determining potential transition point positions"
## [1] "Fitting segmented regression models"
```

Use patternsummary function to check the number of genes for each pattern.

```
patternsummary(pseudopattern)

##         Pattern Number
## 1 constant_down      9
## 2   constant_up     27
## 3 down_constant    131
## 4       down_up     28
## 5   up_constant     49
## 6       up_down     64
## 7            up    132
## 8          down     72
## 9      constant      6
```

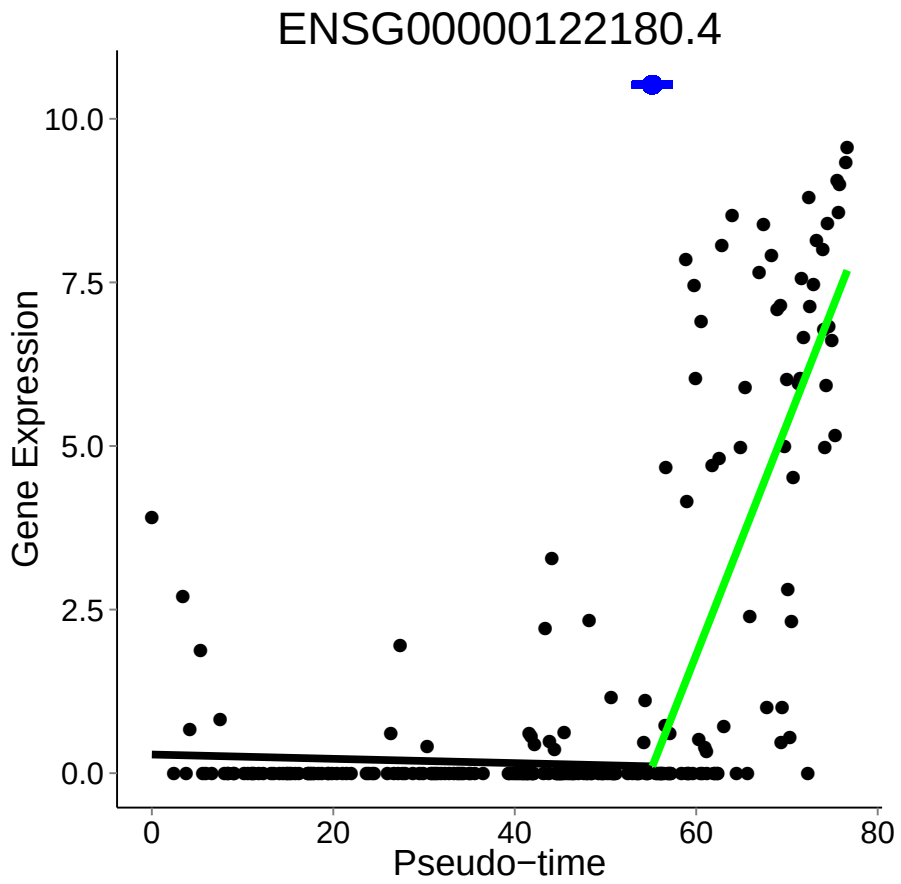
Check the transition points for transition patterns.

```
head(pseudopattern$pattern$constant_up)

##                    transpoint   LCI   UCI
## ENSG00000219481.6       8.495 4.416 12.58
## ENSG00000130741.5       9.727 2.571 16.88
## ENSG00000079482.11     10.850 4.110 17.59
## ENSG00000197746.9      11.500 4.512 18.48
## ENSG00000119669.3      12.230 5.900 18.55
## ENSG00000185585.15     14.090 8.143 20.03
```

Visualize the gene expression using the pseudotimevisualize function. For visualizing the expression of a single gene, a scatter plot will be displayed. The

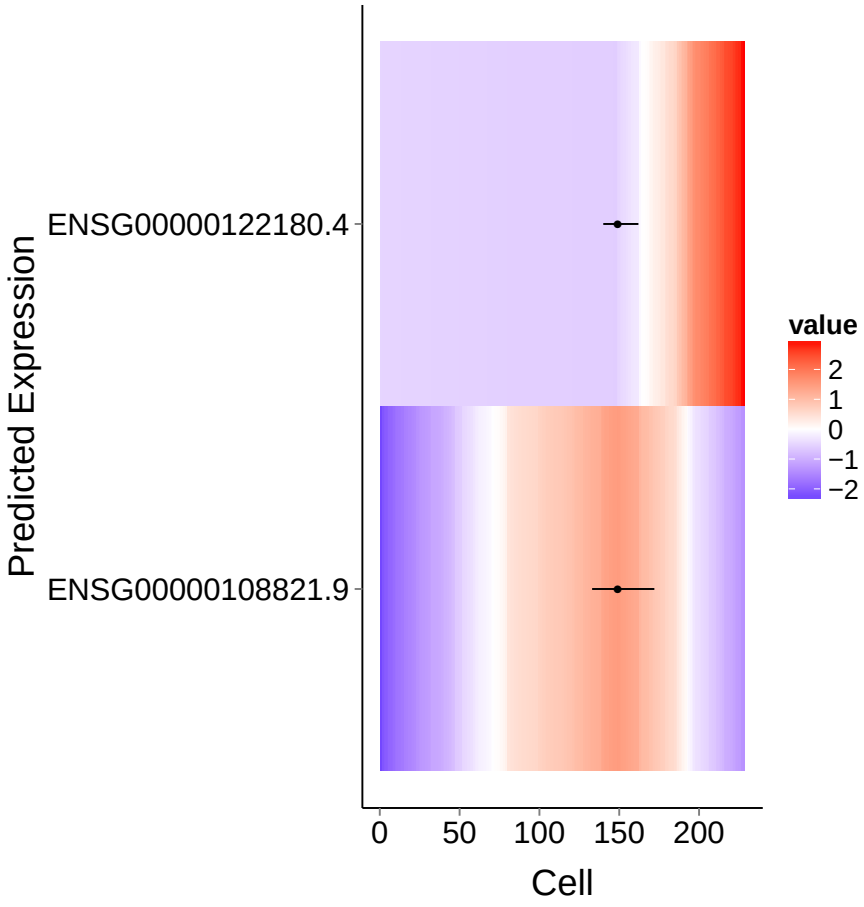
black color of the line stands for constant pattern, green stands for increasing pattern and red stands for decreasing pattern. The blue line on the top of the plot shows the estimated mean and 95% confidence interval of the transition points.

```
pseudotimevisualize(pseudopattern,"ENSG00000122180.4")
```



For visualizing the expression of multiple genes, a heatmap will be displayed. The black dots and line segments show the estimated mean and 95% confidence interval of the transition points.

```
pseudotimevisualize(pseudopattern,c("ENSG00000122180.4","ENSG00000108821.9"))

## Warning:  Non Lab interpolation is deprecated
```



4 GO analysis

The function patternGOanalysis will perform GO analysis for genes with the same expression patterns. The background gene list is all genes in the expression profile (518 genes in this example)
GO analysis for the true experiment time points. The result is a list. Each element is a data.frame of GO analysis results.

```
patternGOanalysis(truepattern,type=c("constant_up","constant_down"),termnum = 5)

##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
```

```
## Annotating nodes ..... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 280 nontrivial nodes
##   parameters:
##   test statistic:  fisher
##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
## Annotating nodes ..... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 3 nontrivial nodes
##   parameters:
##   test statistic:  fisher
## $constant_up
##        GO.ID                                        Term Annotated
## 1 GO:0009617                      response to bacterium        17
## 2 GO:0034097                       response to cytokine        30
## 3 GO:0001911 negative regulation of leukocyte mediate...         1
## 4 GO:0002249                          lymphocyte anergy         1
## 5 GO:0002484 antigen processing and presentation of e...         1
##   Significant Expected classicFisher
## 1           2     0.07        0.0013
## 2           2     0.13        0.0041
## 3           1     0.00        0.0044
## 4           1     0.00        0.0044
## 5           1     0.00        0.0044
##
## $constant_down
##        GO.ID                            Term Annotated Significant
## 1 GO:0032259                     methylation        12           1
## 2 GO:0008152               metabolic process       342           1
## 3 GO:0008150              biological_process       459           1
## 4 GO:0000002 mitochondrial genome maintenance         1           0
## 5 GO:0000003                    reproduction        41           0
##   Expected classicFisher
## 1     0.03         0.026
## 2     0.75         0.745
## 3     1.00         1.000
## 4     0.00         1.000
```

```
## 5     0.09         1.000
```

GO analysis for the pseudo time.

```
patternGOanalysis(pseudopattern,type=c("constant_up","constant_down"),termnum = 5)

##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
## Annotating nodes ............... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 1137 nontrivial nodes
##   parameters:
##   test statistic:  fisher
##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
## Annotating nodes ............... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 610 nontrivial nodes
##   parameters:
##   test statistic:  fisher
## $constant_up
##        GO.ID                                Term Annotated Significant
## 1 GO:0003012               muscle system process        28           8
## 2 GO:0042692          muscle cell differentiation        21           7
## 3 GO:0061061         muscle structure development        39           9
## 4 GO:0006936                muscle contraction        24           7
## 5 GO:0051146 striated muscle cell differentiation        19           6
##   Expected classicFisher
## 1     1.46       2.6e-05
## 2     1.10       3.1e-05
## 3     2.04       4.7e-05
## 4     1.25       8.3e-05
## 5     0.99       0.00018
##
## $constant_down
##        GO.ID                                     Term Annotated
```

```
## 1 GO:0007267                          cell-cell signaling        14
## 2 GO:0007268                        synaptic transmission        10
## 3 GO:0051216                        cartilage development        10
## 4 GO:0061448                connective tissue development        12
## 5 GO:0000060 protein import into nucleus, translocati...         1
##   Significant Expected classicFisher
## 1           3     0.24        0.0012
## 2           2     0.17        0.0112
## 3           2     0.17        0.0112
## 4           2     0.21        0.0161
## 5           1     0.02        0.0174
```

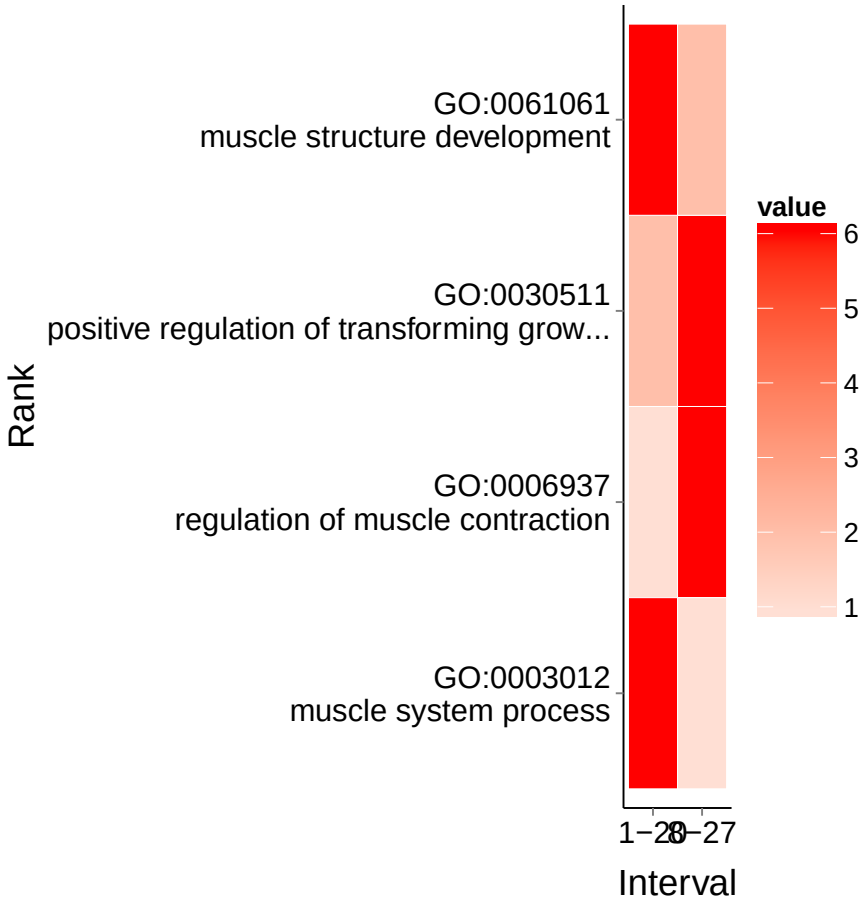
In additional to tranditional GO analysis, SEPA also provides a specific kind of GO analysis for genes with transition patterns. For example for all genes with pattern of constant then up, SEPA first orders all genes according to the transition points. Then SEPA performs GO analysis interatively on the first-20th genes, then 11th-30th genes, then 18th-37th genes (See example below). In this way users can explore the continuous change of biological functions associated with genes with similar expression patterns.

```
(GOresults <- windowGOanalysis(pseudopattern,type="constant_up",windowsize = 20, termnum=5))

##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
## Annotating nodes ............... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 729 nontrivial nodes
##   parameters:
##   test statistic:  fisher
##
## Building most specific GOs ..... ( 2176 GO terms found. )
##
## Build GO DAG topology .......... ( 5202 GO terms and 12160 relations. )
##
## Annotating nodes ............... ( 459 genes annotated to the GO terms. )
##
##   -- Classic Algorithm --
##
##   the algorithm is scoring 1049 nontrivial nodes
##   parameters:
##   test statistic:  fisher
```

```
## $`1-20`
##        GO.ID                                          Term Annotated
## 1 GO:0006937            regulation of muscle contraction        11
## 2 GO:0030511 positive regulation of transforming grow...         4
## 3 GO:1903846 positive regulation of cellular response...         4
## 4 GO:0006936                          muscle contraction        24
## 5 GO:0090257         regulation of muscle system process        14
##   Significant Expected classicFisher
## 1           3     0.41        0.0058
## 2           2     0.15        0.0074
## 3           2     0.15        0.0074
## 4           4     0.89        0.0087
## 5           3     0.52        0.0120
##
## $`8-27`
##        GO.ID                                  Term Annotated Significant
## 1 GO:0003012               muscle system process        28           8
## 2 GO:0061061         muscle structure development        39           9
## 3 GO:0042692           muscle cell differentiation        21           7
## 4 GO:0006936                     muscle contraction        24           7
## 5 GO:0051146 striated muscle cell differentiation        19           6
##   Expected classicFisher
## 1     1.16       3.3e-06
## 2     1.61       4.6e-06
## 3     0.87       5.2e-06
## 4     0.99       1.4e-05
## 5     0.79       4.2e-05
```

windowGOvisualize function can be used to visualize the GO analysis results from windowGOanalysis function.

```
windowGOvisualize(GOresults)

## Warning:  Non Lab interpolation is deprecated
```



5 SEPA GUI

In addition to the basic command lines tools discussed above, SEPA provides a powerful which provides more comprehensive and convenient functions for gene expression pattern analysis. For example, users can easily transform raw gene expression data and convert gene identifiers before the analysis; save the result tables and plots of publication quality; identify genes with different expression patterns on true time and pseudo-time axis. Users are encouraged to use SEPA GUI.

6 Reference

[1] Bendall, S. C., Simonds, E. F., Qiu, P., El-ad, D. A., Krutzik, P. O., Finck, R., ... & Nolan, G. P. (2011). Single-cell mass cytometry of differential im-

mune and drug responses across a human hematopoietic continuum. Science, 332(6030), 687-696.
[2] Tang, F., Barbacioru, C., Bao, S., Lee, C., Nordman, E., Wang, X., ... & Surani, M. A. (2010). Tracing the derivation of embryonic stem cells from the inner cell mass by single-cell RNA-Seq analysis. Cell stem cell, 6(5), 468-478.
[3] Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., ... & Rinn, J. L. (2014). The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nature biotechnology.