

SEPA: Single-Cell Gene Expression Pattern Analysis

Zhicheng Ji

Hongkai Ji

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

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

October 17, 2016

Contents

1	Introductions	1
2	Identify and Visualize Pattern for True Experiment Time Points	2
3	Identify and Visualize Pattern for Pseudo Temporal Cell Ordering	6
4	GO analysis	8
5	SEPA GUI	11
6	Reference	12

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 computation 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 experiment 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)

##
##
##

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':
##
##   Filter, Find, Map, Position, Reduce, anyDuplicated, append,
##   as.data.frame, cbind, colnames, do.call, duplicated, eval,
##   evalq, get, grep, grepl, intersect, is.unsorted, lapply,
##   lengths, mapapply, match, mget, order, paste, pmax, pmax.int,
##   pmin, pmin.int, rank, rbind, rownames, sapply, setdiff, sort,
##   table, tapply, union, unique, unsplit, which, which.max,
##   which.min
## 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
##
## Attaching package: 'S4Vectors'
```

```
## The following objects are masked from 'package:base':
##
##   colMeans, colSums, expand.grid, rowMeans, rowSums
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
## The following object is masked from 'package:base':
##
##   backsolve
##
## groupGOTerms:  GOBPterm, GOMFterm, GOCCterm environments built.
##
## 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)

## ENSG00000000460.12 ENSG00000001630.11 ENSG00000003989.12
##   "down_constant"      "up_constant"      "down_constant"
## ENSG00000005448.12  ENSG00000010292.8  ENSG00000011426.6
##   "down_constant"      "down_constant"      "down_constant"
```

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

```
patternsummary(truepattern)

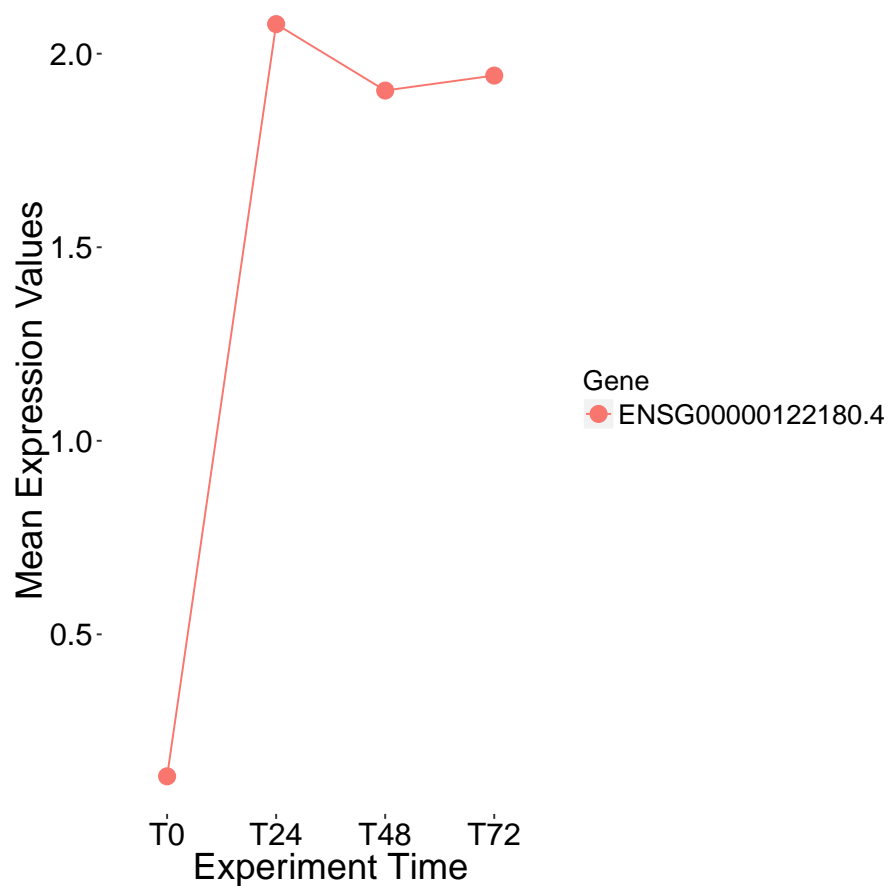
##           Pattern Number
## 1           constant    129
## 2      constant_down     1
## 3 constant_down_constant   6
## 4           constant_up     2
## 5   constant_up_constant   17
## 6           down_constant  196
## 7   down_constant_down     1
## 8   down_constant_up       4
## 9             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.

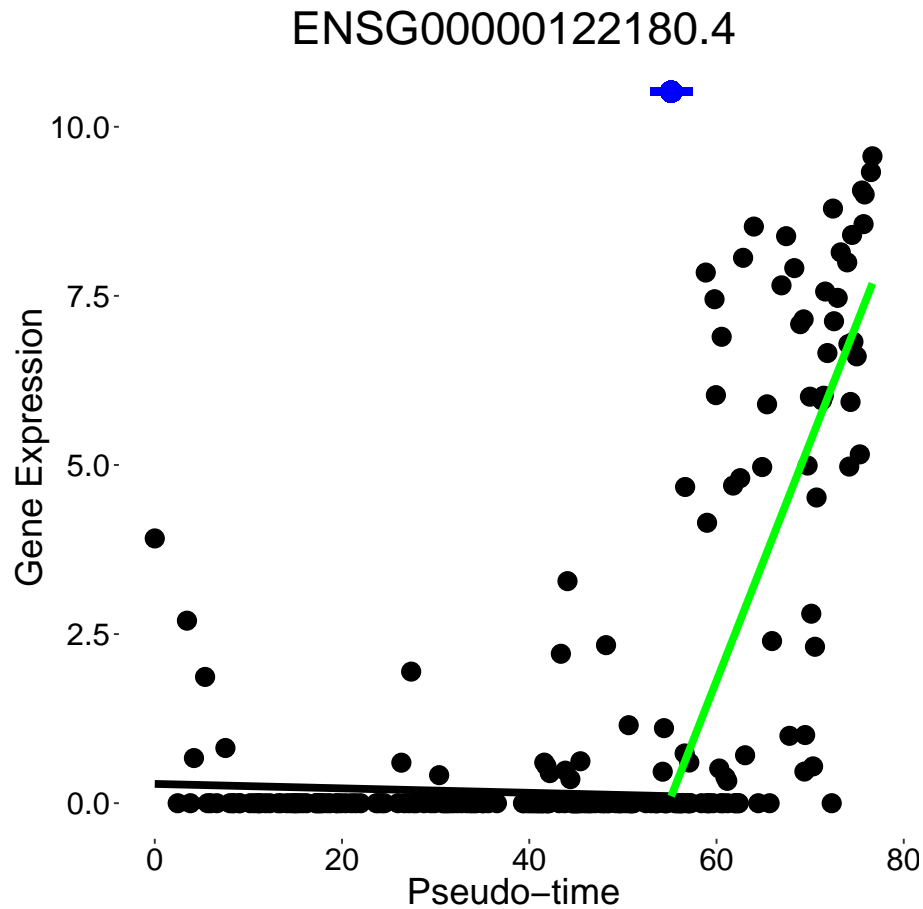
```
head(pseudopattern$patterns$constant_up)

##           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

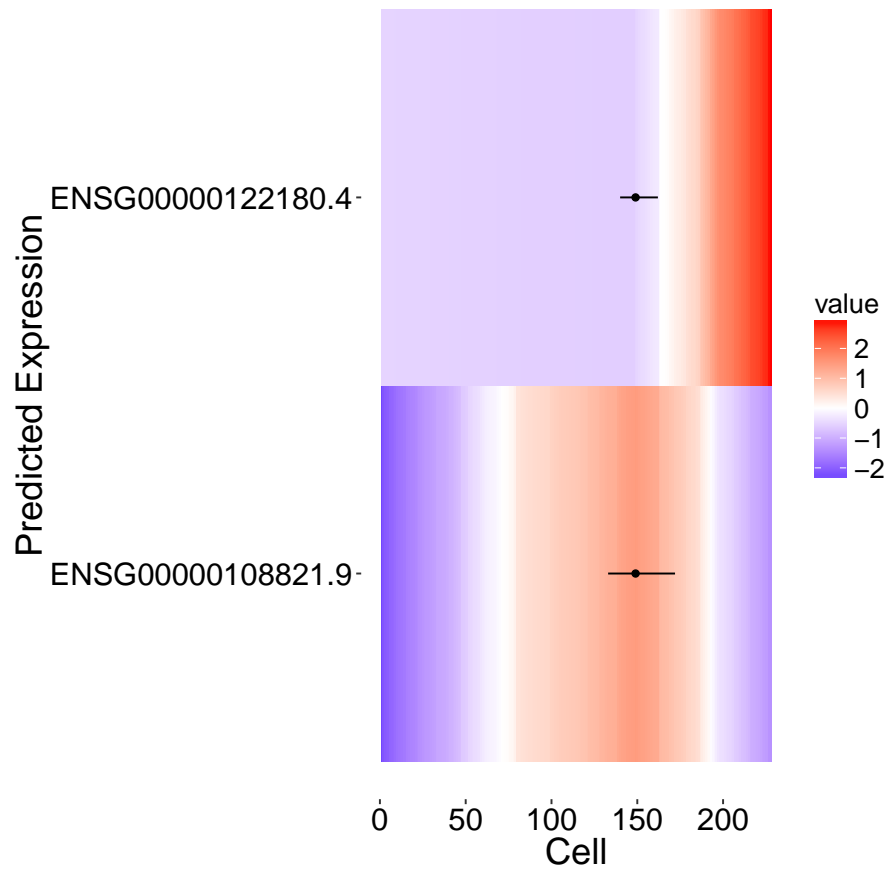
black color of the line stands for constant pattern, green stands for increasing pattern and red stands for decreasing pattern. The blue line on the top of the plot shows the estimated mean and 95% confidence interval of the transition points.

```
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"))
```



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)

## $constant_up
##      GO.ID                                     Term Annotated
## 1 GO:0034097                response to cytokine             28
## 2 GO:0010288                response to lead ion              1
## 3 GO:0033591                response to L-ascorbic acid        1
```



```
## 4 GO:0002480 antigen processing and presentation of e... 2
## 5 GO:0043583 ear development 2
## Significant Expected classicFisher
## 1 2 0.12 0.0036
## 2 1 0.00 0.0044
## 3 1 0.00 0.0044
## 4 1 0.01 0.0087
## 5 1 0.01 0.0087
##
## $constant_down
## GO.ID Term Annotated Significant
## 1 GO:0032259 methylation 12 1
## 2 GO:0008152 metabolic process 313 1
## 3 GO:0008150 biological_process 459 1
## 4 GO:0000002 mitochondrial genome maintenance 1 0
## 5 GO:0000003 reproduction 58 0
## Expected classicFisher
## 1 0.03 0.026
## 2 0.68 0.682
## 3 1.00 1.000
## 4 0.00 1.000
## 5 0.13 1.000
```

GO analysis for the pseudo time.

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

```
## $constant_up
## GO.ID Term Annotated Significant
## 1 GO:0003012 muscle system process 29 8
## 2 GO:0042692 muscle cell differentiation 22 7
## 3 GO:0061061 muscle structure development 43 9
## 4 GO:0006936 muscle contraction 25 7
## 5 GO:0051146 striated muscle cell differentiation 18 6
## Expected classicFisher
## 1 1.52 3.5e-05
## 2 1.15 4.4e-05
## 3 2.25 0.00011
## 4 1.31 0.00011
## 5 0.94 0.00013
##
## $constant_down
## GO.ID Term Annotated Significant
## 1 GO:0007267 cell-cell signaling 27 3
## 2 GO:0007268 chemical synaptic transmission 10 2
## 3 GO:0051216 cartilage development 10 2
```

```
## 4 GO:0098916 anterograde trans-synaptic signaling 10 2
## 5 GO:0099536 synaptic signaling 10 2
## Expected classicFisher
## 1 0.47 0.0084
## 2 0.17 0.0112
## 3 0.17 0.0112
## 4 0.17 0.0112
## 5 0.17 0.0112
```

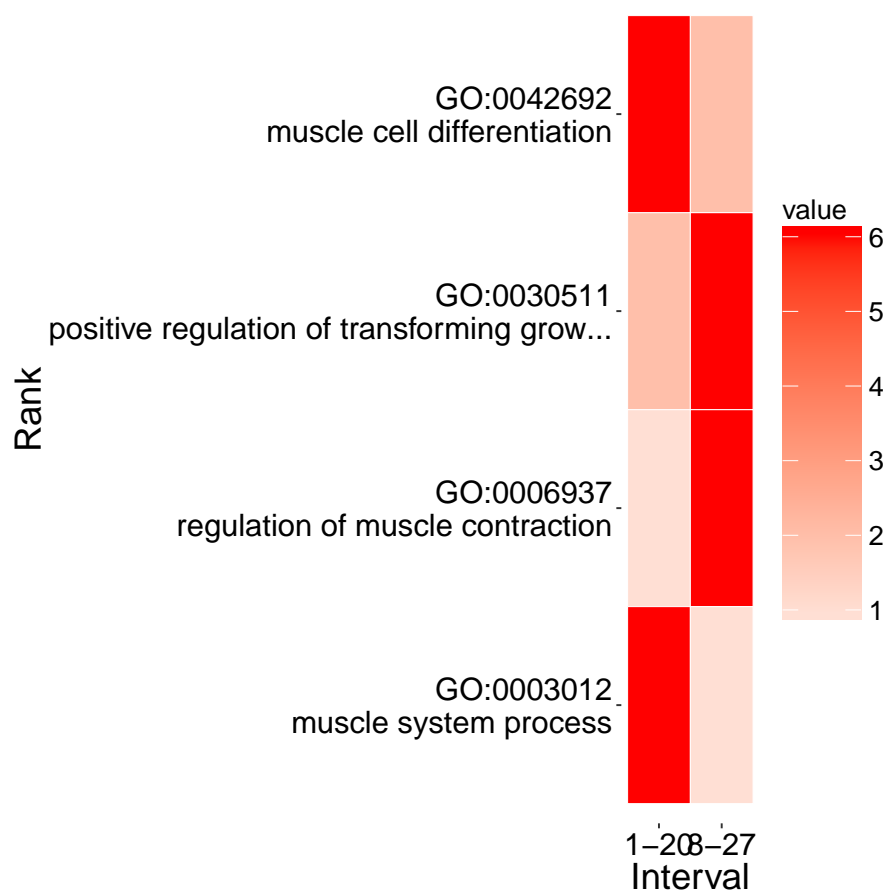
In addition to traditional GO analysis, SEPA also provides a specific kind of GO analysis for genes with transition patterns. For example for all genes with pattern of constant then up, SEPA first orders all genes according to the transition points. Then SEPA performs GO analysis iteratively on the first-20th 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.

```
(G0results <- windowGOanalysis(pseudopattern,type="constant_up",windowsize = 20, termnum=5))

## $`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 25
## 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 0.93 0.0101
## 5 3 0.52 0.0120
##
## $`8-27`
## GO.ID Term Annotated Significant
## 1 GO:0003012 muscle system process 29 8
## 2 GO:0042692 muscle cell differentiation 22 7
## 3 GO:0061061 muscle structure development 43 9
## 4 GO:0006936 muscle contraction 25 7
## 5 GO:0051146 striated muscle cell differentiation 18 6
## Expected classicFisher
## 1 1.20 4.4e-06
## 2 0.91 7.4e-06
## 3 1.78 1.1e-05
## 4 1.03 1.9e-05
## 5 0.75 3.0e-05
```

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

```
windowGOvisualize(GOresults)
```



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 immune 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*.