SCPattern: a statistical approach to characterize expression changes in single cell RNA-seq experiments with ordered conditions

Ning Leng , Li-Fang Chu, Jeea Choi, Christina Kendziorski, James Thomson, and Ron Stewart

November 24, 2015

Contents

1 Introduction

SCPattern (as detailed in Leng* and Chu* *et al.*, 2015 (1)) is an empirical Bayes approach to characterize expression changes in single cell RNA-seq (scRNA-seq) experiments with ordered conditions, such as time points, spacial course, etc. In an ordered scRNA-seq experiments, investigators are mainly interested in identifying genes with expression changes along ordered conditions and in classifying them by their expression patterns. Both changes due to mean expression changes of non-zero cells and the number of zeros are important to characterize changes. Figure 1 shows example genes of three classes - genes whose expression mean of non-zero cells changes over time; genes whose percentage of zero cells changes over time; and genes with both changes. SCPattern is based on an empirical Bayes model which identifies genes with expression changes by considering zeros and non-zero cells collectively, and classifies them into directional expression patterns. SCPattern tests distribution changes of a gene across the two conditions using directional Kolmogorov-Smirnov (K-S) statistic and classify the gene into categories Up, Down and equally expressed (EE).

2 The model

To simplify the presentation, we refer to ordered conditions as time points denoted by $t = 1, 2, \ldots, T$, noting that the method directly accommodates other ordered data structures (e.g. space, gradient, etc.). Let $X_g^{t_n}$ be expression of gene g, cell n from time t. The set of observed expression values of gene g at time point t is then denoted by $X_g^t = \left(X_g^{t_1}, \ldots, X_g^{t_{N_t}}\right)$, in which N_t is the number of cells in time point t.

2.1 Two conditions

SCPattern tests whether a gene's distribution changes across the two time points t and $t+1$ by using directional Komogorov-Smirnov (K-S) statistic. Given a value ω, the K-S statistics for up and down regulation are:

$$Y_{g,Up}^{t,t+1} = sup_w(F_g^{t+1}(\omega) - F_g^t(\omega))$$

$$Y_{g,Down}^{t,t+1} = sup_w(F_g^t(\omega) - F_g^{t+1}(\omega))$$

For a pair of time points, we can classify genes into categories Up, Down and EE by taking genes with large $Y_{g,Up}^{t,t+1}$, genes with large $Y_{g,Down}^{t,t+1}$, and genes with large $Y_{g,rest}^{t,t+1}$ $(=1 - Y_{g,Up}^{t,t+1} - Y_{g,Down}^{t,t+1})$. To classify genes to these three categories, we assume $Y_g^{t,t+1}$ follows a mixture of Dirichlet distributions:

$$P(Y_g^{t,t+1}) = \sum_z \pi_z^{t,t+1} P(Y_g^{t,t+t1} | Z_g^{t,t+1} = z),$$

where

$$Y_g^{t,t+t1} | Z_g^{t,t+1} = z \sim \begin{cases} Dir(\alpha, \beta, \beta); \text{if } z = Up \\ Dir(\beta, \alpha, \beta); \text{if } z = Down \\ Dir(\beta, \beta, \alpha); \text{if } z = EE \end{cases}$$

with constraint $\alpha > \beta$.

2.2 Multiple conditions

The main goals in an ordered scRNA-seq experiment is identifying genes that change over time, and specifying each genes' expression path. For each gene g and each transition between t and $t+1$, SCPattern estimates the probability of each state at each transition.

In a time course experiment with T time points, the marginal distribution of gene g along all time points can be written as:

$$P(Y_g) = \prod_{t=1,\ldots,T-1} P(Y_g^{t,t+1})$$

The posterior probability (PP) of gene g coming from pattern k_1 is obtained by Bayes rule:

$$PP(\text{Pattern } k_1 | Y_g) = \frac{P(Y_g|k_1)}{\sum_k P(Y_g|k_1)}$$

3 Run SCPattern

Before analysis can proceed, the SCPattern package must be loaded into the working space:

```
> library(SCPattern)
```

3.1 Required inputs

Data: The object `Data` should be a $G-by-S$ matrix containing the expression values for each gene and each sample, where G is the number of genes and S is the number of samples. These values should exhibit estimates of gene expression across samples. Counts of this nature may be obtained from RSEM (2), Cufflinks (3), or a similar approach. Cross-sample library size normalization should be performed. An cross-sample library size normalization by median normalization are shown in section 3.2.

Conditions: The object `Conditions` should be a factor of length S that indicates to which condition each sample belongs. Note the order of levels in the factor should represent the order in the RNA-seq experiments.

The object `SCPatternExData` is a simulated data matrix containing 1000 rows of genes and 300 columns of samples. The genes are named `g1, g2, ...` and the samples are named `S1, S2, ...`

```
> data(SCPatternExData)
> str(SCPatternExData)
```

```
 num [1:1000, 1:300] 404 2345 1490 209 800 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:1000] "Gene_1" "Gene_2" "Gene_3" "Gene_4" ...
  ..$ : chr [1:300] "S1" "S2" "S3" "S4" ...
```

Here we simulated 60 cells for 5 time points (conditions). To specify which condition each sample belongs, we define:

```
> CondVector <- rep(paste("t",1:5,sep=""),each=60)
> str(CondVector)
```

```
 chr [1:300] "t1" "t1" "t1" "t1" "t1" "t1" "t1" "t1" "t1" "t1" ...
```

Downstream analysis by SCPattern requires the conditions to be specified as a factor. In particular, levels of the factor need to be sorted along the time/spatial course. For example, to generate a factor with ordered conditions from t1 to t5, we define:

```
> Conditions <- factor(CondVector, levels=c("t1","t2","t3","t4","t5"))
> str(Conditions)
```

```
 Factor w/ 5 levels "t1","t2","t3",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
> levels(Conditions)
```

```
[1] "t1" "t2" "t3" "t4" "t5"
```

3.2 Normalization

SCPattern requires cross-sample normalization to be applied to adjust for sequencing depth differences among different samples. Here, the library size factors may be obtained via the function `MedianNorm`, which reproduces the median normalization approach in DESeq (4).

```
> Sizes <- MedianNorm(SCPatternExData)
```

In a case that none of the genes are expressed in all samples due to technical dropouts, we can add `alternative=T` option in `MedianNorm` function (for example, `MedianNorm(SCPatternExData,alternative=T)`).

If quantile normalization is preferred, library size factors may be obtained via the function `QuantileNorm` (for example, `QuantileNorm(GeneMat,.75)` for Upper-Quartile Normalization in (5)).

To obtain the normalized expression matrix, user may used the GetNormalizedMat() function:

```
> DataNorm <- GetNormalizedMat(SCPatternExData, Sizes)
```

3.3 Directional comparison in a two conditions

As detailed in Section 2, SCPattern tests whether a gene's distribution changes across the two conditions, t and $t+1$. Here, we test the gene's distribution between time $t1$ and $t2$, so define the aforementioned variables:

```
> index.use <- which(CondVector=="t1"|CondVector=="t2")
> cond.two <- factor(CondVector[index.use], levels=c("t1","t2"))
> data.two <- SCPatternExData[,index.use]
> data.norm.two <- DataNorm[,index.use]
> size.two <- Sizes[index.use]
```

The function `SCPTest()` is used to test the gene's distribution changes. By setting `Dropout.remove=TRUE`, we can remove dropouts when testing the distribution changes.

```
> res.two <- SCPTest(data.two, cond.two, size.two, Dropout.remove=FALSE, Directional = TRUE)
```

The posterior probabilitis of being no change (NC), down, and up are obtained as follows

```
> pp.two <- res.two$PP
> head(pp.two)
```

```
                 NC      Down        Up
Gene_1 2.951350e-01 0.1433694 0.5614955
Gene_2 6.750864e-16 0.8055903 0.1944097
Gene_3 6.750864e-16 0.8055903 0.1944097
Gene_4 6.743842e-16 0.1953253 0.8046747
Gene_5 1.640612e-01 0.1640612 0.6718776
Gene_6 3.582429e-01 0.5049552 0.1368019
```

The output `res.two$PP` is a matrix of genes showing posterior probability of being NC, Down, and Up pattern between the two conditions. In order to see the genes corresponds to the posterior probability of being Down, we may sort the genes as follows

```
> down.two <- sort(pp.two[,"Down"], decreasing=T)
> head(down.two)
```

```
   Gene_2    Gene_3    Gene_7   Gene_12   Gene_14   Gene_25
0.8055903 0.8055903 0.8055903 0.8055903 0.8055903 0.8055903
```

A gene will be called as a confident assignment to Down or Up if its PP is greater than 0.5. And we note that the 0.5 threshold can be changed. For example, we call below genes to have down pattern.

```
> down.pattern = down.two[which(down.two > 0.5)]
> str(down.pattern)
```

```
 Named num [1:126] 0.806 0.806 0.806 0.806 0.806 ...
 - attr(*, "names")= chr [1:126] "Gene_2" "Gene_3" "Gene_7" "Gene_12" ...
```

3.4 Non-directional comparison in a two conditions

Similar to 3.3, the function `SCPTest()` can be used to identify DE genes by setting option `Directional = FALSE`. This will test whether the gene is DE or not instead of specifying it's pattern as Down or Up.

```
> res.two.nondir <- SCPTest(data.two, cond.two, size.two, Dropout.remove=FALSE, Directional = FALSE)
> pp.two.nondir <- res.two.nondir$PP
> head(pp.two.nondir)
```

```
              NC        DE
Gene_1 0.3509051 0.6490949
Gene_2       NaN       NaN
Gene_3       NaN       NaN
Gene_4 0.1906437 0.8093563
Gene_5 0.2325457 0.7674543
Gene_6 0.4184718 0.5815282
```

```
> two.nondir <- sort(pp.two.nondir[,"DE"], decreasing=T)
> head(two.nondir)
```

```
   Gene_4   Gene_15   Gene_24   Gene_65   Gene_71  Gene_100
0.8093563 0.8093563 0.8093563 0.8093563 0.8093563 0.8093563
```

```
> de.nondir = two.nondir[which(two.nondir > 0.5)]
> str(de.nondir)
```

```
 Named num [1:234] 0.809 0.809 0.809 0.809 0.809 ...
 - attr(*, "names")= chr [1:234] "Gene_4" "Gene_15" "Gene_24" "Gene_65" ...
```

3.5 Comparison in a multiple conditions

For multiple time points data, SCPattern provides gene-specific PP associated with each expression pattern. The MAP of gene g is defined as $\text{argmax}_k PP(\text{Pattern}_k)$. The function SCPTest() can be used and we can remove dropouts when testing the distribution changes by setting `Dropout.remove=TRUE`.

```
> res.multi <- SCPTest(SCPatternExData, CondVector, Sizes, Dropout.remove=FALSE)
```

The posterior probabilities of being specific pattern are obtained as follows

```
> head(res.multi$sortedlist)
```

```
        PP_marginal Path
Gene_2  "0.447"     "Down-Down-Down-Down"
Gene_3  "0.447"     "Down-Down-Up-Up"
Gene_12 "0.447"     "Down-Down-Up-Up"
Gene_14 "0.447"     "Down-Down-Up-Up"
Gene_43 "0.447"     "Up-Up-Down-Down"
Gene_58 "0.447"     "Up-Up-Down-Down"
```

```
> str(res.multi$sortedlist)
```

```
 chr [1:314, 1:2] "0.447" "0.447" "0.447" "0.447" "0.447" "0.447" ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:314] "Gene_2" "Gene_3" "Gene_12" "Gene_14" ...
  ..$ : chr [1:2] "PP_marginal" "Path"
```

The output `res.multi$sortedlist` is a matrix with two columns. The first column shows PP(most likely path) for each gene. And the second column shows a gene's most likely path (the path with highest PP). The higher the PP is, the more likely that this gene is following the particular path. Rows are genes that are defined as DE.

The PP of EE (not change along all time points) can be shown in the output `res.multi$NCPP` and the PP of all possible patterns can be seen in the output `res.multi$PP.all`.

```
> head(res.multi$NCPP)
```

```
      Gene_1       Gene_2       Gene_3       Gene_4       Gene_5       Gene_6
3.847057e-03 4.271224e-65 4.271224e-65 4.262397e-65 1.319215e-03 2.240769e-02
```

```
> str(res.multi$PP.all)
```

```
 num [1:1000, 1:81] 3.85e-03 4.27e-65 4.27e-65 4.26e-65 1.32e-03 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:1000] "Gene_1" "Gene_2" "Gene_3" "Gene_4" ...
  ..$ : chr [1:81] "NC-NC-NC-NC" "Down-NC-NC-NC" "Up-NC-NC-NC" "NC-Down-NC-NC" ...
```

4 Plotting

We demonstrate the plotting routine that is implemented in the `VioFun` function. This function produces side-by-side violin plots (where the curves represent a smoothed kernel density estimate). Each condition is represented by one viloin plot.

We illustrate the function by displaying the three types of results obtained from simulated dataset in section sec:quickstart. Each results are shown with and without log-transformation. For log-transformed data, a count of 1 is added before log-transformation so that zeros can be displayed. The option `Log="y"` can be used to plot log-scale y axes.

To visualize the top 6 Down pattern identified genes from section 3.3 as an original scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(names(down.two)[i],data.norm.two, cond.two)
```

To visualize the top 6 Down pattern identified genes from section 3.3 as log2-transformed scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(names(down.two)[i],log2(data.norm.two+1), cond.two)
```

To visualize the top 6 Down pattern identified genes from section 3.4 as an original scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(names(de.nondir)[i],data.norm.two, cond.two)
```

To visualize the top 6 Down pattern identified genes from section 3.4 as log2-transformed scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(names(de.nondir)[i],log2(data.norm.two+1), cond.two)
```

To visualize the top 6 DE identified genes from section 3.5 as an original scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(rownames(res.multi$sortedlist)[i], DataNorm, Conditions)
```

To visualize the top 6 DE identified genes from section 3.5 as log2-transformed scale:

```
> par(mfrow=c(2,3))
> for(i in 1:6) VioFun(rownames(res.multi$sortedlist)[i], log2(DataNorm+1), Conditions)
```

5 Session info

Here is the output of sessionInfo on the system on which this document was compiled:

```
> print(sessionInfo())
```

```
R version 3.1.3 (2015-03-09)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.10.5 (Yosemite)
```

```
locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] splines   stats4    stats     graphics  grDevices utils     datasets  methods
[9] base

other attached packages:
 [1] SCPattern_0.0.1     VGAM_1.0-0          DirichletReg_0.6-3  rgl_0.93.1098
 [5] Formula_1.2-1       gtools_3.4.2        vioplot_0.2         sm_2.2-5.4
 [9] EBSeq_1.11.0        testthat_0.10.0     gplots_2.17.0       blockmodeling_0.1.8

loaded via a namespace (and not attached):
 [1] BiocStyle_1.4.1    bitops_1.0-6       caTools_1.17.1     crayon_1.3.1
 [5] digest_0.6.8       gdata_2.16.1       grid_3.1.3         KernSmooth_2.23-14
 [9] lattice_0.20-31    maxLik_1.3-4       memoise_0.2.1      miscTools_0.6-16
[13] sandwich_2.3-4     tools_3.1.3        zoo_1.7-12
```

References

[1] Ning Leng, Li-Fang Chu, Jeea Choi, Christina Kendziorski, James Thomson, and Ron Stewart. Scpattern: A statistical approach to characterize expression changes in single cell rna-seq experiments with ordered conditions. *XX*, XX(XX):XX, 2016.

[2] Bo Li and Colin N Dewey. Rsem: accurate transcript quantification from rna-seq data with or without a reference genome. *BMC bioinformatics*, 12(1):323, 2011.

[3] Cole Trapnell, Adam Roberts, Loyal Goff, Geo Pertea, Daehwan Kim, David R Kelley, Harold Pimentel, Steven L Salzberg, John L Rinn, and Lior Pachter. Differential gene and transcript expression analysis of rna-seq experiments with tophat and cufflinks. *Nature protocols*, 7(3):562–578, 2012.

[4] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome biol*, 11(10):R106, 2010.

[5] James H Bullard, Elizabeth Purdom, Kasper D Hansen, and Sandrine Dudoit. Evaluation of statistical methods for normalization and differential expression in mrna-seq experiments. *BMC bioinformatics*, 11(1):94, 2010.