scDDboost

Xiuyu Ma, Keegan Korthauerz, Christina Kendziorski, and Michael A. Newton

Contents

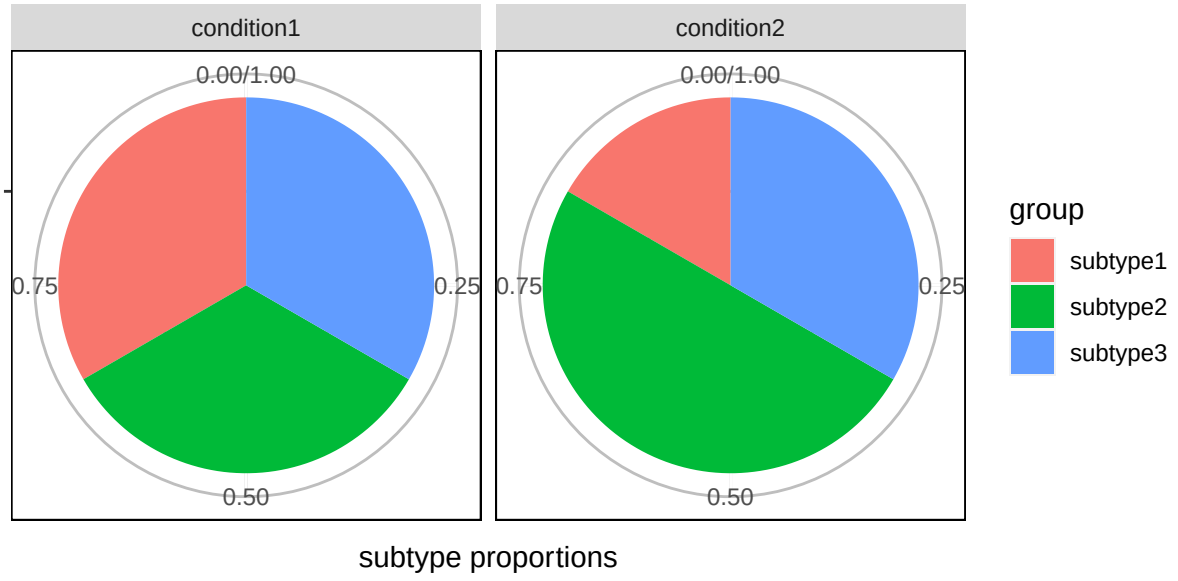
Abstract:

The scDDboost package is designed to identifying distributional changes for transcripts measured by single-cell RNA-seq. It uses mixture model for gene expression, which accounts for cellular heterogeneity and unique varaitional property underlying the data.

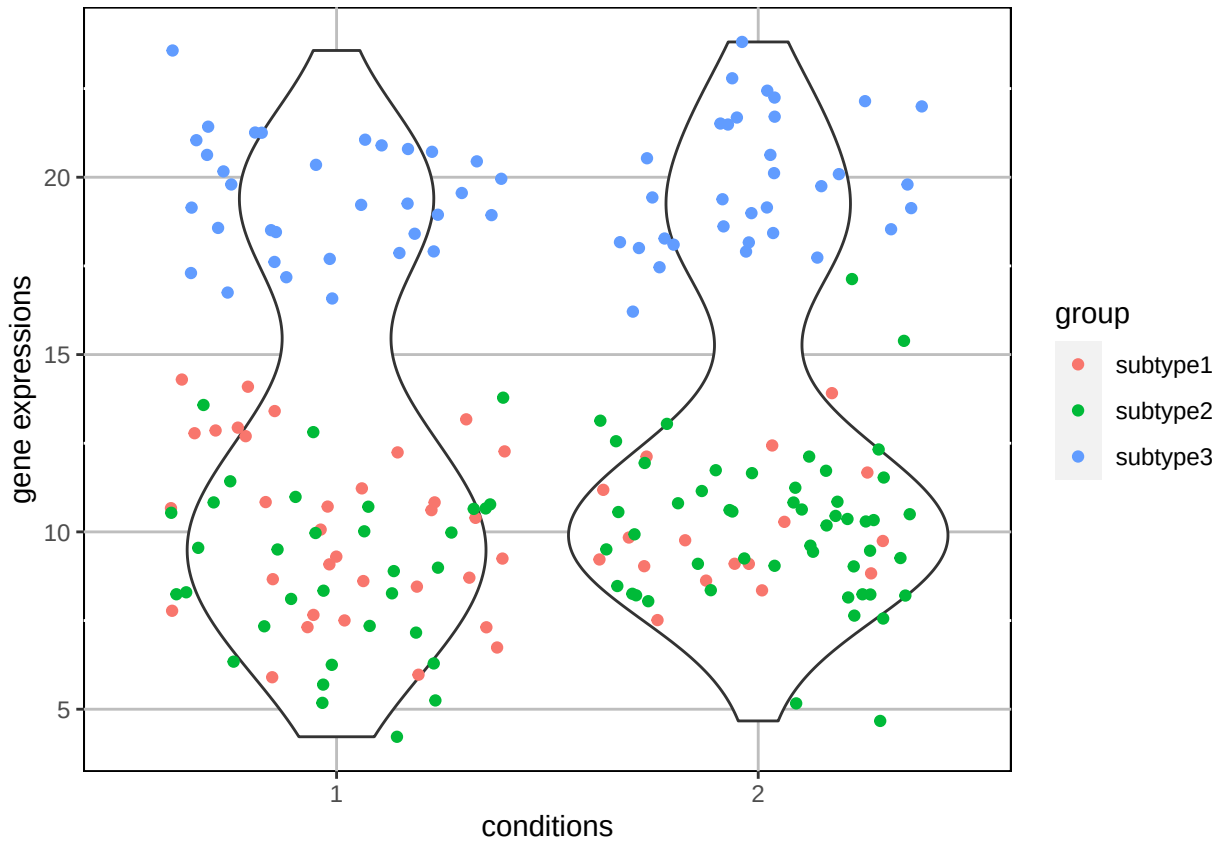
Package scDDboost 1.0.0 Report issues on "github link here"

1.Introduction

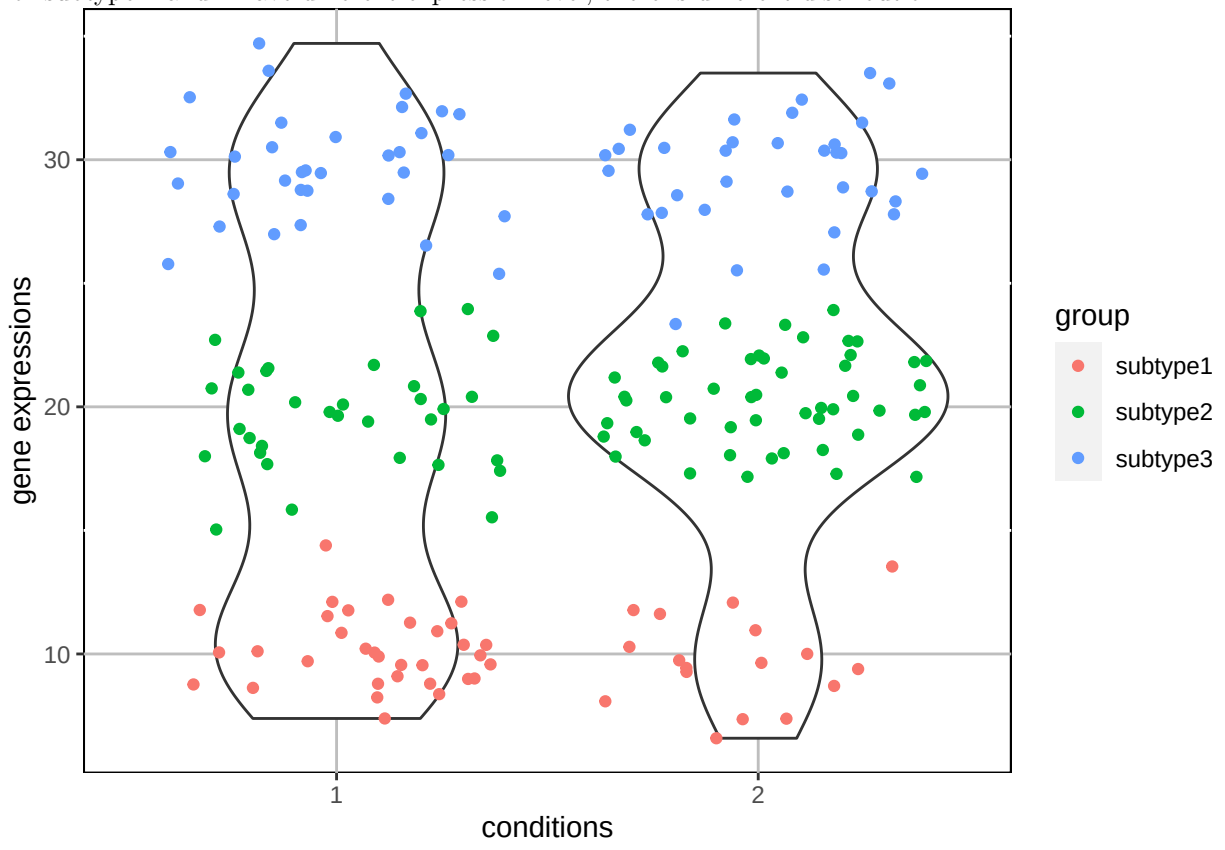
scDDboost scores evidence of a gene being differentially distributed(DD) across two conditions for single cell RNA-seq data. Higher resolution brings several chanllenges for analyzing the data, specifically, the distribution of gene expression tends to have high prevalence of zero and multi-modes. To account for those characteristics and utilizing some biological intuition, we view the expression values sampled from a pool of cells mixed by distinct cellular subtypes blind to condition label. Consequently, the distributional change can be fully determined by the the change of subtype proportions. One tricky part is that not any change of proportions will lead to a distributional change. Given that some genes could be equivalent expressed across several subtypes, even the individual subytpe proportion may differ between conditions but as long as the aggregated proportions over those subtypes remain the same between conditions, it will not introduce different distribution. For example



Proportions of subtypes 1 and 2 changed between the 2 conditions. The gene is not DD if subtype 1 and 2 have the same expression level



For subtype 1 and 2 have different expression level, there is different distribution



2. Posterior probability of a gene being DD

`PDD` is the core function developed to quantify the posterior probabilities of DD for input genes. Let's look at an example,

```
suppressMessages(library(scDDboost))
```

Next, we load the toy simulated example a *SingleCellExperiment* object that we will use for identifying and classifying DD genes.\

```
data(sim_dat)
```

Verify that this object is a member of the SingleCellExperiment class and that it contains 200 cells and 1000 genes. The colData slot (which contains a dataframe of metadata for the cells) should have a column that contains the biological condition or grouping of interest. In this example data, that variable is the `condition` variable. Note that the input gene set needs to be a matrix of normalized counts. We run the function PDD

```
suppressMessages(library(SummarizedExperiment))
data_counts = assays(sim_dat)$counts
```

```
## Loading required package: SingleCellExperiment
```

```
conditions = colData(sim_dat)$conditions
rownames(data_counts) = 1:1000

##here we use 1 core to compute the distance matrix
D_c = cal_D(data_counts,1)
```

```
pdd = PDD(data = data_counts,cd = conditions, ncores = 1, D = D_c)
```

```
## 100 genes are all zero counts, not being considered in DD analysis
```

```
## estimated number of subtypes: 4
```

There are 4 input parameters needed to be specified by user, the dataset, the condition label, number of cpu cores used for computation and a distance matrix of cells. Other input parameters have default settings.

2.1. clustering of cells

We provide a default method of getting the distance matrix, archived by `cal_D`, in general `PDD` accept all valid distance matrix. User can also input a cluster label rather than distance matrix for the argument `D`, but the random distancing mechanism which relies on distance matrix will be disabled and `random` should be set to false.

For the number of sutypes, we provide a default function `detK`, which consider the smallest number of sutypes such that the ratio of difference within cluster between difference between clusters become smaller than a threshold (default setting is 1).

If user have other ways to determine K, K should be specified in `PDD`.

```
## determine the number of subtypes
K = detK(D_c)
```

If we set threshold to be 5% then we have estimated DD genes

```
EDD = which(pdd > 0.95)
```

Notice that, pdd is actually local false discovery rate, this is a conservative estimation of DD genes. We could gain further power, let index gene by $g = 1, 2, ..., G$ and let $p_g = P(DD_g|\text{data})$, $p_{(1)}, ..., p_{(G)}$ be ranked local false discovery rate from small to large. To control the false discovery rate at 5%, our positive set is those genes with the s^* smallest lFDR, where

$$s^* = \text{argmax}_s\{s, \frac{\Sigma_{i=1}^{s} p_{(i)}}{s} \leq 0.05\}$$

```
EDD = lsz(pdd,0.05)
```

Function `lsz` can be used to get the estimated DD genes under the above transformation