User Manual
RoDEO: Robust Differential Gene Expression

July 18, 2014

Contents

1 Overview

RoDEO* is a framework for detecting differentially expressed genes and stable genes between RNA-seq experiments.

For detecting differentially expressed genes, RoDEO uses normalization that is not based on the relative values of the gene expression but on the relative order of the expression within a sample. Indeed, the expression values of all genes in an experiment are utilized in a re-sampling approach, to tease out robust relative ranks of the genes in several re-generated instances of the RNA-seq experiments. RoDEO may be applied on data with one or several replicates per experiment. The input to RoDEO are read counts per gene from RNA sequencing experiments.

RoDEO performs the following three main steps, described in detail in [1] and implemented in [2]:

1. Simulated re-sampling of an RNA-seq experiment, transforming the raw read counts into scale-free robust expression distributions.
2. Computing distance measures between two experiments, based on the distributions constructed in step 1.
3. Identifying stable genes between two experiments, based on the distributions constructed in step 1.

2 Requirements and availability

The RODEO algorithm is provided as a command line version in the form of a `C++` unix executable. We have compiled (with `g++`) and tested the system on Linux Fedora operating system. RoDEO executable files are available for download at
`http://researcher.ibm.com/project/5513`

The compressed `RoDEO.zip` folder contains six files:

- `RoDEOsampling` - executable program for simulated re-sampling
- `RoDEOde` - executable program for detecting differentially expressed genes
- `RoDEOstable` - executable program for detecting stable genes
- *LICENSE.txt* - copy of the licence agreement that the user agrees to when downloading the program
- *ExampleA.txt, ExampleB.txt* - example input files for RoDEO

*__Ro__bust __DE O__perator

When RoDEO is used in published analysis, it should be cited as:

Parida, L., Zhou, Z., Haiminen, N.: **RoDEO: Non-parametric Framework for DE & Stable Genes Characterization in HTS data**. Under review (2014).

For further information visit `http://researcher.ibm.com/project/5513` or contact parida@us.ibm.com.

3 `RoDEOsampling` for constructing dispersion functions

3.1 Input format

Re-sampling is first performed for each experiment separately by `RoDEOsampling`, and the results are then processed by `RoDEOde` or `RoDEOstable`. `RoDEOsampling` takes as input one file, where the only column (on row i) is the number of reads mapping to the (ith) gene. All the results will be indexed in the order of the input file rows. If there are several replicates, each replicate needs to be processed separately and the results stored in separate results files, for further processing.

`RoDEOsampling` constructs dispersion functions for each gene, denoting their expression bins' distribution across several iterations of re-assigning and binning the read counts. An example input file is shown in Table 1.

Sometimes it is desired to the use read counts after taking the logarithm of the counts $c_g + 1$ per gene g (count plus one, to avoid negative results). In that case, the second column of the input may be given as non-negative real numbers $\log(c_g + 1)$. It is up to the user to determine if log-transformation is required, e.g., by examining the cumulative curve of input read counts for smoothness with and without taking the log. See the figure in [2] for an example of a smooth cumulative curve.

13
0
2457

Table 1: An example input file for three genes.

3.2 Output format

The output of `RoDEOsampling` is a file with one row per iteration, and one column per gene, as demonstrated in Table 2. Larger values denote higher expression, and the columns are in the same order as the input rows.

2	1	5
3	1	6
2	1	6
2	1	5

Table 2: An example `RoDEOsampling` output file (.bins format) for three genes and 4 iterations.

3.3 Running `RoDEOsampling`

The binary file `RoDEOsampling` can be executed with command line parameters:

```
RoDEOsampling Bins Genes Iterations Reads Gap
```

where:

- `Bins`: Number of bins, P, for the expression values (e.g., 20);
- `Genes`: Number of genes G (number of rows in the input);
- `Iterations`: Number of independent re-sampling simulations, I (e.g., 100);
- `Reads`: Number or reads, R, for the re-sampling (e.g., 10000000);
- `Gap`: For assigning the bins, speed up the linear segmentation by increasing gap between sampled values (e.g., 10).

Please note that all input parameters are required to be set, and are assumed to be **positive integers** (other values may result in execution failure).

For example, to execute `RoDEOsampling` using the input file `ExampleA.txt` possible command lines include:

```
RoDEOsampling 10 1000 50 1000000 1 < ExampleA.txt > ExampleA.bins
```

(10 bins, 1000 genes, 50 iterations and 1M reads, with no gap (gap= 1))

```
RoDEOsampling 10 1000 200 10000000 10 < ExampleA.txt > ExampleA.bins
```

(10 bins, 1000 genes, 200 iterations, 10M reads, with gap 10 to speed up the iterations)

4 `RoDEOde` for detecting differentially expressed genes

4.1 Input format

`RoDEOde` compares two experiments and ranks all genes w.r.t. differential expression between the experiments. `RoDEOde` takes as input constructed dispersion functions (distributions) between two experiments A and B, the output from

`RoDEOsampling` as shown in Table 2 (.bins format). Each sample may consist of several replicates; the number of replicates and their respective sampling files are given as input.

4.2 Output format

The output of `RoDEOde` is a file with one row per gene, the gene index (1...G) followed by (i) the maximum norm distance, (ii) the mode distance, and (iii) the direction of change +/- from the first to the second experiment. The genes are ordered w.r.t. decreasing DE. Example output is shown in Table 3.

2	1.0	7	+
3	1.0	5	-
1	0.8	2	-

Table 3: An example `RoDEOde` output file for three genes.

4.3 Running `RoDEOde`

The binary file `RoDEOde` can be executed with command line parameters:

```
RoDEOde Bins Genes Iterations ReplicatesA ReplicatesB FileA1.bins
[FileA2.bins FileA3.bins ...]  FileB1.bins [FileB2.bins ...]
```

where:

- `Bins`: Number of bins, P, for the expression values (e.g., 20);
- `Genes`: Number of genes G (number of columns in the input);
- `Iterations`: Number of independent re-sampling simulations, I, that were used to construct the input (number of rows in the input);
- `ReplicatesA`: Number of replicates in the first experiment;
- `ReplicatesB`: Number of replicates in the second experiment;
- `FileA1.bins`: Sampled dispersion function in the first replicate in the first experiment;
- `FileB1.bins`: Sampled dispersion function in the first replicate in the second experiment.

The parameters denoted [ ] are optional, for including data more than one replicate, and their number must reflect the respective `Replicates` parameter minus one. Please note that all input parameters are required to be set, and are assumed to be **positive integers** (other values may result in execution failure).

Also note, the number of bins and iterations need to be the same that were used to construct all input files.

For example, to execute `RoDEOde` using the input files `ExampleA.bins` and `ExampleB.bins` with one replicate per experiment:

```
RoDEOde 10 1000 50 1 1 ExampleA.bins ExampleB.bins > ExampleAB.de
```

(10 bins, 1000 genes, 50 iterations were used to construct A and B bins)

5 `RoDEOstable` for detecting stable genes

5.1 Input format

`RoDEOstable` compares two experiments A and B, and detects genes whose expression remains stable. `RoDEOstable` takes as input two re-sampled constructed distributions, the output from `RoDEOsampling` as shown in Table 2 (.bins format).

5.2 Output format

The output of `RoDEOstable` are one or several files, with one row per iteration, listing the stable genes (genes from 1...G) between those iterations in A and B. Example output is shown in Table 4. When there are several replicates, several files are constructed, one for each pair of replicates between experiment A and experiment B. The files are named as follows for each pair of files (replicates): `<file1>VS<file2>.stable`, e.g., `FileA1.binsVSFileB3.bins.stable`.

1	5	82		
2	5	29	82	
1	5	29		
1	5	29	82	99

Table 4: An example `RoDEOstable` output file for 4 iterations and option to output only 1 best solution per iteration. Here only gene 5 occurs in all stable sets.

Additionally, `RoDEOstable` outputs a file with a **count** for each gene, denoting the number of iterations where it appears in the stable gene list (in at least one solution per iteration). The file is sorted by decreasing count. Those genes with the maximum count (= `Iterations*ReplicatesA*ReplicatesB`) are the most likely stable candidates.

5.3 Running `RoDEOstable`

The binary file `RoDEOstable` can be executed with command line parameters:

```
RoDEOstable Bins Genes Iterations Solutions ReplicatesA ReplicatesB
FileA1.bins [FileA2.bins FileA3.bins ...]  FileB1.bins [FileB2.bins
...]
```

where:

- `Bins`: Number of bins, P, for the expression values (e.g., 20);
- `Genes`: Number of genes G (number of columns in the input);
- `Iterations`: Number of independent re-sampling simulations, I, that were used to construct the input (number of rows in the input);
- `Solutions`: Output options "1" output only one (randomly chosen) best solution; "0" output all the (equally good) best solutions; "N" output at most N best solutions (where $N > 1$, note that this may also include suboptimal solutions);
- `ReplicatesA`: Number of replicates in the first experiment;
- `ReplicatesB`: Number of replicates in the second experiment;
- `FileA1.bins`: Sampled dispersion function in the first replicate in the first experiment;
- `FileB1.bins`: Sampled dispersion function in the first replicate in the second experiment.

The parameters denoted [ ] are optional, for including data in the additional replicates, and their number must reflect the respective `Replicates` parameter minus one. Please note that all input parameters are required to be set, and are assumed to be **positive integers**, except 0 is allowed for `Solutions` (other values may result in execution failure). Also note, the number of bins and iterations need to be the same that were used to construct all input files.

For example, to execute `RoDEOstable` using the input files `ExampleA.bins` and `ExampleB.bins`:

```
RoDEOstable 10 1000 50 1 1 1 ExampleA.bins ExampleB.bins >
ExampleAB.stablecounts
```

(10 bins, 1000 genes, and 50 iterations were used to construct A and B bins, output is one (randomly chosen) best solution per iteration.)

6 Example

Here is described an example analysis using the files *ExampleA.txt* and *ExampleB.txt* included with the RoDEO software distribution.

- Sampling experiment A with 10 bins, 1000 genes, 100 iterations, 1M reads, with gap 10 to speed up the iterations: `RoDEOsampling 10 1000 200 1000000 10 < ExampleA.txt > ExampleA.bins`
- Sampling experiment B with 10 bins, 1000 genes, 100 iterations, 1M reads, with gap 10 to speed up the iterations: `RoDEOsampling 10 1000 200 1000000 10 < ExampleB.txt > ExampleB.bins`
- DE ranking of genes with 10 bins, 1000 genes, 100 iterations, 1 replicate for both A and B: `RoDEOde 10 1000 100 1 1 ExampleA.bins ExampleB.bins > ExampleAB.de`
- Stable gene set with 10 bins, 1000 genes, 100 iterations, 1 best solution per iteration, 1 replicate for both A and B: `RoDEOstable 10 1000 100 1 1 1 ExampleA.bins ExampleB.bins > ExampleAB.stablecounts`

Result file *ExampleAB.de* contains an ordered ranking of genes s.t. strongest DE candidates are on the first rows. Result file *ExampleAB.stablecounts* contains an ordered list of genes s.t. strongest stable gene candidates are on the first rows (particularly all those with count 100).

7 Troubleshooting

Here is a list of some possible problems and solutions when running the RODEO analysis:

- **Problem**: There is an error in the execution of RoDEO.
- *Solution*: The most likely reason is that the input file is formatted incorrectly, or the parameter values have erroneous values. Please take a look at the examples in the sections regarding the input format.
- **Problem**: `RoDEOsampling` takes a long time to complete, or runs out of memory.
- *Solution*: Increase the parameter `Gap` to relax the linear segmentation accuracy in favor or increased speed and reduced memory footprint. Another alternative for increasing speed is to reduce the number of `Iterations` I or `Reads` R.
- **Problem**: The executables do not run on a specific unix system, or there are other issues.
- *Solution*: Contact parida@us.ibm.com for assistance.

References

[1] N. Haiminen, M. Klaas, Z. Zhou, F. Utro, P. Cormican, T. Didion, C. S. Jensen, C. Mason, S. Barth, and L. Parida. Comparative Exomics of Phalaris cultivars under salt stress. *(under review)* 2014.

[2] L. Parida, Z. Zhou, and N. Haiminen. RoDEO: Non-parametric Framework for DE & Stable Genes Characterization in HTS data *(under review)* 2014.