LPEseq Package Introduction

Jungsoo Gim and Taesung Park

26 Aug 2015

1 Introduction

The LPEseq is an R package for performing differential expression (DE) test with RNA sequencing data. Briefly, LPEseq extends local pooled error method, which was developed for microarray data analysis, to sequencing data even with non-replicated sample in each condition. A number of methods are available for both count-based and FPKM-based RNA-Seq data. Among these methods, few (for example, EdgeR and DESeq) can deal with no replicate data, but not accurately. LPEseq was designed for the RNA-Seq data with a small number of replicates, especially with non-replicate in each class. Also LPEseq can be equally applied both count-base and FPKM-based (non-count values) input data. This brief vignette is written for the users who want to use the LPEseq for their DE analysis. An extended documentation about the method can be found in our original manuscript (`http://bibs.snu.ac.kr/software/LPEseq`).

The full LPEseq User's Guide (with figures) is available in our website. Please visit`http://bibs.snu.ac.kr/software/LPEseq` and a pdf file is available.

2 Installation

The source code and a package of LPEseq are freely available from our website and from Bionconductor. You can use it by loading source code in our website.

```
> install.packages("LPEseq")
> library(LPEseq)
```

3 What's in LPEseq

The LPEseq package comes with a number of functions to perform a differential expression test with or without replicates. The main functions are `LPEseq.normalise()` and `LPEseq.test()`, which are designed for running a normalization across the samples and a whole differential expression test, respectively. All the functions that start with the same as the package are newly developed in our method while others are from original LPE package. The most of the functions are described in the manual available in our web-site.

4 A quick example

4.1 Input data

The LPEseq package starts its analysis with read counts. Therefore you have to equip yourself with RNA-Seq read count data sets on your hand first. The package expects count data in the form of a matrix (or a vector) of integer values. But it is not limited to non-count data, for example, FPKM values generated using Cufflinks. If you are not familiar with generating count table, please visit web-sites to learn how to obtain such a data. Good references are GenomicRanges in Bioconductor, htseq-count script written in Python framework, the well-known software and etc. In this LPEseq tutorial, you do not need count dataset right away. Using `generateData()`, you can generate simulated datasets and learn how to use LPEseq library without real dataset.

```
> simData <- generateData(n.rep=3, n.deg=1000)
```

The output of generated data consists of 20000 genes with 6 samples (three replicate per each condition) and differential expression index in the last column (denoted by DEG). DEGs are indexed with 1 otherwise 0.

If you have your own data, you can directly read the data with `read.table()`.Once you loaded your own data, the following analysis procedure is the same.

```
> yourData <- read.table("your_data.txt")
```

To visualize mean and variance of the data, LPEseq provides `AVplot()` function.

```
> par(mfrow=c(1,2))
> AVplot(simData[,1:3])
> AVplot(simData[,1:3], logged=F)
```

4.2 Normalization

As the first step of analysis, we need to remove the effect of sequencing depth. LPEseq follows the similar idea of DESeq. LPEseq divides each column of the count table by the size factor for this column. By doing so, the count values are brought to a common comparable scale. LPEseq adds pseudo-count value 1 to all the values in the data and take log-2 transformation. Because of sparse properties in average bins of RNA-Seq in raw scale, it is recommended to log-2 transform the original data. LPEseq does this by typing

```
> simData.norm <- LPEseq.normalise(simData[,-7])
```

Note that DEG index is removed. If your own data consists of original count values, exactly the same script will do,

```
> youData.norm <- LPEseq.normalise(yourData)
```

But when your data includes normalized count values, such RPKM or FPKM, just take log-transformation to your data for further analyses. We recommend using log-transformed data for LPEseq method.

```
> youData.norm <- log(yourData, base = 2)
```

4.3 Testing Differential Expression

LPEseq provides simple one-step procedure to perform differential expression test. Unlike other methods, LPEseq is applicable to experiments without replicates. By simply providing expression matrix (or vector) per each condition as arguments of `LPEseq.test()` function, LPEseq automatically performs appropriate differential expression test, if the input data is properly given. Replicates are essential to interpret biological experiments. Nevertheless, experiments without any replicates per each condition are frequently undertaken, and LPEseq can deal with them. The followings are the R scripts for differential expression test with or without replicates in each condition, respectively.

```
> sim.result <- LPEseq.test(simData.norm[,1:3], simData.norm[,4:6])
> sim.result.norep <- LPEseq.test(simData.norm[,1], simData.norm[,4])
```

The result of LPEseq.test includes average values in each condition, pooled standard deviation, Z type statistics, nominal p-value and adjusted p-value (with Benjamini-Hochberg multiple testing correction).

To save the output to a file, use the write.table() function.

```
> write.table(sim.result, file="result_file.txt", quote=F, sep="\t")
```