Bioconductor's edge package
Version 0.99.0

John D. Storey and Andrew J. Bass
Princeton University
http://genomine.org/contact.html

June 24, 2014

Contents

1 Introduction **2**

2 Citing this package **2**

3 Getting help **3**

4 Quick start guide **3**

5 Examples **4**
5.1 Creating the models 4
5.2 edgeSet object 5
5.3 edgeFit 6

6 Static study: gibson dataset **6**
6.1 Independent time course study: kidney dataset 9
6.2 Longitudinal time course study: endotoxin dataset 10

7 Features **11**

1 Introduction

`edge` is an open-source software package for significance analysis of DNA microarray experiments, and is able to identify genes that are differentially expressed between two or more different biological conditions (e.g., healthy versus diseased tissue). There are a number of existing software packages to perform significance analysis but `edge` uses the odp-statistic from the Optimal Discovery Procedure (ODP). Whereas previously existing methods employ statistics that are essentially designed for testing one gene at a time (e.g., t-statistics and F-statistics), the ODP uses information across all genes to test for differential expression.

The improvements in power are substantial; Figure 1 shows a comparison between edge and five leading software packages, based on a well-known breast cancer expression study (Hedenfalk et al. 2001). In addition to the significance analysis procedures for identifying differentially expressed genes, `edge` includes implementations of popular packages such as `snm`, `sva`, `qvalue` and `ExpressionSet`.

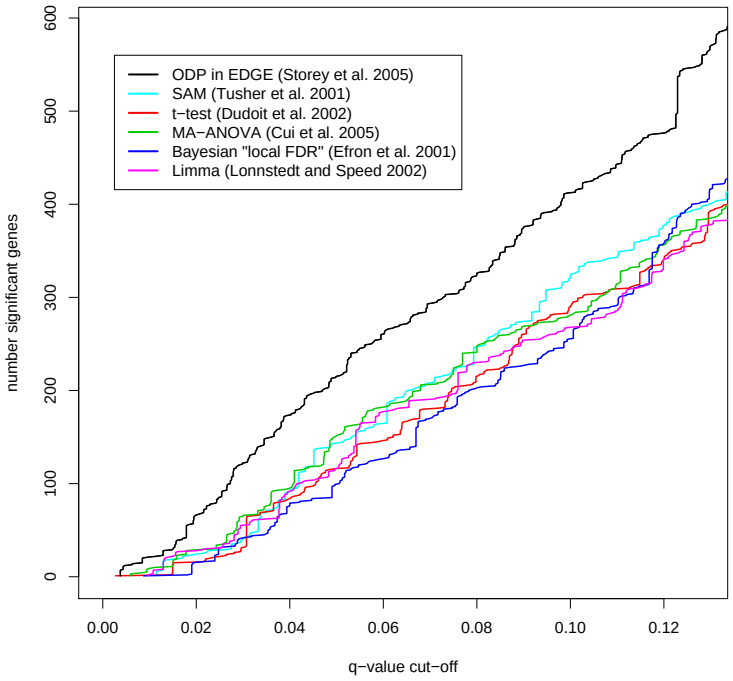


Figure 1: Comparison of EDGE to various other leading methods for identifying differential expressed genes in the Hedenfalk et al., 2001 study. Figure retrieved from Leek et al. (2005).

2 Citing this package

Storey, J. D. The optimal discovery procedure for large-scale significance testing, with applications to comparative microarray experiments. *Journal of the Royal Statistical Society, Series B 69* (2007), 347–368.

Storey, J. D., Dai, J., and Leek, J. T. The optimal discovery procedure for large-scale significance

testing, with applications to comparative microarray experiments. *Biostatistics 8* (2007), 414–432.

Woo, S., Leek, J. T., and Storey, J. D. A computationally efficient modular optimal discovery procedure. *Bioinformatics 27* (2011), 509–515.

3 Getting help

Hopefully most questions relating to the package will be answered in the vignette but to get a more detailed account of how to use the functions simply type, within R:

```
help(package = "edge")
```

Please contact the authors directly with any issues regarding bugs. Otherwise, any questions or problems implementing `edge` will most efficiently be addressed on the Bioconductor mailing list, http://stat.ethz.ch/mailman/listinfo/bioconductor.

4 Quick start guide

There are two ways to use `edge`: (i) using `edgeModel` or (ii) using an `ExpressionSet` object. Using `edgeModel` has less flexibility and is designed to help users create alternative and null models from their experiment.

An example of using `edgeModel` for significance analysis:

```
# Import data
data(kidney)
sex <- kidney$sex[kidney$tissue == "c"]
age <- kidney$age[kidney$tissue == "c"]
kidexpr <- log(kidney$kidexpr[, kidney$tissue ==
    "c"] + 10)

# Create edgeSet object from edgeModel
edgeObj <- edgeModel(data = kidexpr, adj.var = model.matrix(~sex),
    tme = age, sampling = "timecourse", basis.df = 4)

# Optimal Discovery Procedure
edgeODP <- odp(edgeObj)

# Likelihood Ratio Test
edgeLRT <- lrt(edgeObj)
```

Given an `ExpressionSet` object and both the alternative and null hypothesis, the `odp` or `lrt` function can be used as follows:

```
# Create ExpressionSet
expSet <- ExpressionSet(assayData = kidexpr, phenoData = as(data.frame(sex = sex,
    age = age), "AnnotatedDataFrame"))

# Create Models
```

```
nModel <- ~sex
fModel <- ~sex + ns(age, df = 3, intercept = FALSE)

# Create edgeSet object from ExpressionSet object
edgeObj <- edgeSet(expSet, full.model = fModel, null.model = nModel)

# Optimal Discovery Procedure
edgeODP <- odp(edgeObj)

# Likelihood Ratio Test
edgeLRT <- lrt(edgeObj)
```

In the above models, `fModel` is the alternative hypothesis and `nModel` is the null hypothesis. It is recommended to create an `ExpressionSet` object because of the flexibility in creating the null and alternative hypothesis. The following sections of the manual go through a case study to show additional features of the `edge` package.

5 Examples

Three different examples will be used to show the functionality of `edge`. In each example, there will be a different type of experiment: static, longitudinal and independent. It will become evident that in each case, the process is very similar and the only step that differs is the model setup.

The three main steps when using `edge`:

- Write the alternative and null models of the experiment. In this manual they will be called `altMod` and `nullMod`.
- Use an `ExpressionSet` object to create an `edgeSet` object by using function `edgeSet`.
- Either use `edgeFit` to extract the fitted values, residuals and/or coefficients from both models or run functions `odp`/`lrt` directly to obtain the q-value object. See `?qvalue`.

5.1 Creating the models

The example datasets in this section each represent different experimental designs: the kidney dataset is an independent time course design, the endotoxin dataset is a longitudinal time course study and the gibson dataset is a static experiment. We will go through each case to show how hypothesis tests are setup in `edge`.

Kidney dataset Gene expression measurements from kidney samples were obtained from 72 human subjects ranging in age from 27 to 92 years. Only one array was obtained per sample, the age and tissue type of each subject was recorded. There are two covariates in this dataset: sex and age. We are interested in finding the effect of age on gene expression. In this case, we handle the time variable, age, by fitting a natural spline curve as presented by Storey (2005).

```
nullMod <- ~-1 + as.factor(sex)
altMod <- ~-1 + as.factor(sex) + ns(age, df = 4)
```

Endotoxin dataset The endotoxin dataset provide gene expression measurements in an endotoxin study where four subjects were given endotoxin and four subjects were given a placebo. Blood samples were collected and leukocytes were isolated from the samples before infusion and measurement were recorded at times 2, 4, 6, 9, 24 hours. We are interested in identifying genes that vary over time between the endotoxin and control groups. In this example, the models are slightly more complicated. The two covariates are time and class. For the null model we fit a spline curve for the time variable and the full model will contain the class variable and an interaction term between class and time.

```
mNull <- ~-1 + ns(time, df = 4, intercept = FALSE)
mFull <- ~-1 + ns(time, df = 4, intercept = FALSE) +
    ns(time, df = 4, intercept = FALSE):class + class
```

Gibson dataset The gibson dataset provides gene expression measurements in peripheral blood leukocyte samples from three Moroccan Amazigh groups leading distinct ways of life: desert nomadic (DESERT), mountain agrarian (VILLAGE), and coastal urban (AGADIR). Suppose we are interested in finding the genes that differentiate the Moroccan Amazigh groups the most. There are three covariates in this dataset: Gender, Batch and Location. Since we are interested in finding the genes that differentiate the three groups the most, the variable Location will be included in the alternative model.

```
nullMod <- ~Gender + Batch
altMod <- ~Gender + Batch + Location
```

5.2 `edgeSet` object

The `edgeSet` object is the main class for `edge`. The `edgeSet` object contains q-value information among other experimental data inherited from the `ExpressionSet`. The `edgeSet` function requires an `ExpressionSet` object. To create an `ExpressionSet` using the `kidney` dataset as an example:

```
library(edge)
data(kidney)

# Interested in cortex samples
sex <- kidney$sex[kidney$tissue=="c"]
age <- kidney$age[kidney$tissue=="c"]
kidexpr <- kidney$kidexpr[, kidney$tissue=="c"]
expSet <- ExpressionSet(assayData = kidexpr,
                        phenoData = as(data.frame(age=age, sex=sex), "AnnotatedDataFrame"))
```

To access the expression values, one can use the function `exprs(expSet)` or to access the covariates, `pData(expSet)`. The `ExpressionSet` class is a widely used object in Bioconductor and more information can be found http://www.bioconductor.org/packages/2.14/bioc/html/Biobase.html.

Using the `ExpressionSet` object to create an `edgeSet` is very simple once the alternative and null models are known:

```
# null and full models
nullMod <- ~-1 + as.factor(sex)
altMod <- ~-1 + as.factor(sex) + ns(age, df = 4)
edgeObj <- edgeSet(expSet, full.model = altMod, null.model = nullMod)
```

The object contains the following slots:

- `full.model`: the full model of the experiment
- `null.model`: the null model of the experiment
- `full.matrix`: the full model in matrix form
- `null.matrix`: the null model in matrix form
- `individual`: containing information on individuals in the experiment
- `qvalue.obj`: `qvalue` list
- `ExpressionSet`- inherits the slots from `ExpressionSet` object

The qvalue information can be accessed from the object by using `qvalue.obj(object)`. See `?edgeSet` for more details on how to extract and set each slot in the object.

5.3 edgeFit

The `edgeFit` contains information on the model fitting phase of the procedure. The slots for the object are

- fit.full- fitted values for full model
- fit.null- fitted values for null model
- res.full- residuals for the full model
- res.null- residuals for the null models
- dH.full- diaganol elements in the projection matrix for the full model
- beta.coef- the coefficients for the full model
- stat.type- statistic type used, either "odp" or "lrt"

To access the coefficients in the `edgeFit` object, the user only has to type `beta.coef(object)`. Similarly, to access the full residuals, `fit.full(object)`.

6 Static study: `gibson` dataset

Lets begin the analysis by importing the data and creating an `ExpressionSet`,

```
library(edge)
data(gibson)
covar <- data.frame(t(gibson$covar))
expSet <- ExpressionSet(assayData=gibson$exprdat,
                        phenoData=as(covar, "AnnotatedDataFrame"))
```

The next step is to determine the model equations for the experiment. Note that the alternative model must include the null model in its formulation. In this example, there are the covariates `Gender`, `Batch` and `Location`. The `Gender` and `Batch` covariates are the adjustment variables and the `Location` is the biological variable. The models will look like

```
nullMod <- ~Gender + Batch
altMod <- ~Gender + Batch + Location
edgeObj <- edgeSet(expSet, full.model = altMod, null.model = nullMod)
```

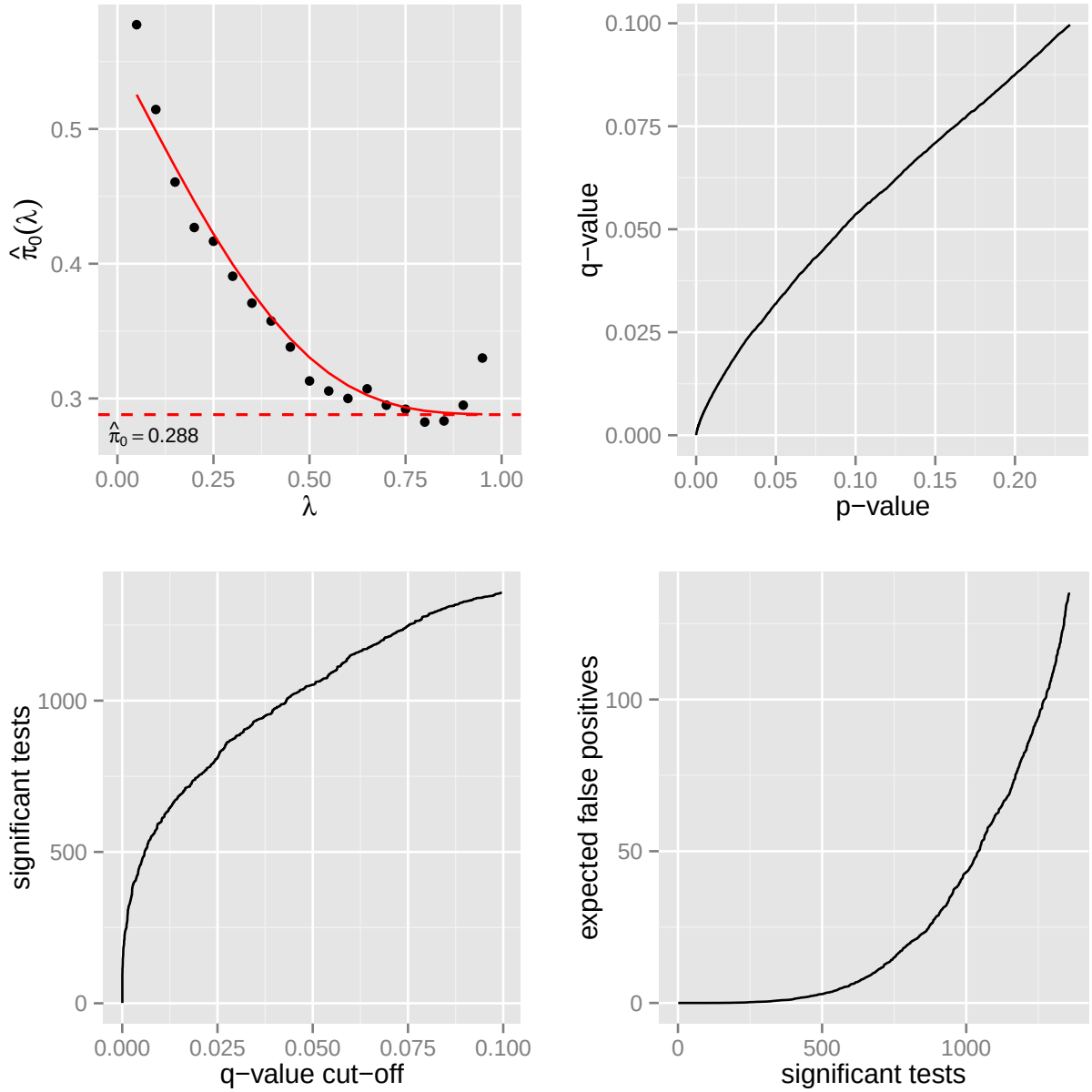
The function `edgeFit` can be used to retrieve the fitted values of the model. See `?edgeFit` for more details. To run `odp` or `lrt` on the dataset to identify differentially expression genes,

```
edge.odp <- odp(edgeObj, bs.its=100, verbose=FALSE)
edge.lrt <- lrt(edgeObj, nullDistn="normal")

## Error:  could not find function "lrtStat"
```

Comparing the findings between both methods:

```
plot(qvalue.obj(edge.odp))
```




```
summary(qvalue.obj(edge.odp))
```

```
##
## Call:
## qvalue(p = pval)
##
## pi0: 0.2883
##
## Cumulative number of significant calls:
##
##           <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1
## p-value      184    323   594    744   903 1074
```

```
## q-value      120    246   597    812  1052 1357
## local FDR     98    173   376    514   650  832
##             <1
## p-value   2000
## q-value   2000
## local FDR 2000

summary(qvalue.obj(edge.lrt))

## Error:  error in evaluating the argument 'object' in selecting a method for function 'qvalue.obj':
Error:  object 'edge.lrt' not found
```

6.1 Independent time course study: `kidney` dataset

Lets begin the analysis by importing the data and in this example use the function `edgeModel` to create the models of the experiment. See `?edgeModel` for more details.

```
data(kidney)

# Interested in cortex samples
sex <- kidney$sex[kidney$tissue=="c"]
age <- kidney$age[kidney$tissue=="c"]
kidexpr <- kidney$kidexpr[, kidney$tissue=="c"]

# Create model
edgeObj <- edgeModel(dat=kidexpr,
                     adj.var=model.matrix(~sex),
                     tme=age,
                     sampling="timecourse",
                     basis.df=4)

## Error:  could not find function "edgeModel"
```

Following similar steps in the previous example, a user can retrieve the fitted values from either statistical method in the experiment and run `odp` or `lrt`.

```
# fitted values
efObject <- edgeFit(edgeObj, stat.type="odp")
efObject <- edgeFit(edgeObj, stat.type="lrt")

edge.odp <- odp(edgeObj, bs.its=100, verbose=FALSE)
edge.lrt <- lrt(edgeObj, nullDistn="bootstrap", bs.its=100, verbose=FALSE)

## Error:  could not find function "lrtStat"
```

Comparing the findings between both methods:

```
summary(qvalue.obj(edge.odp))

##
## Call:
## qvalue(p = pval)
```

```
##
## pi0: 0.3001
##
## Cumulative number of significant calls:
##
##           <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1
## p-value      235    387   639    779   914 1102
## q-value      204    350   649    843  1060 1372
## local FDR    164    235   427    556   662  825
##              <1
## p-value    2000
## q-value    2000
## local FDR 1880
```

```
summary(qvalue.obj(edge.lrt))
```

```
## Error:  error in evaluating the argument 'object' in selecting a method for function 'qvalue.obj':
Error:  object 'edge.lrt' not found
```

6.2 Longitudinal time course study: `endotoxin` dataset

The first step is to create an `ExpressionSet`:

```
data(endotoxin)
# Create ExpressionSet object
expr <- endotoxin$expr
ind <- endotoxin$individual
tme <- endotoxin$time
cls <- endotoxin$class
colnames(expr) <- NULL
expSet <- ExpressionSet(assayData=expr,
          phenoData=as(data.frame(individual=ind, time=tme, class=cls),
                       "AnnotatedDataFrame"))
```

Next step is to create an edgeSet object

```
mNull <- ~-1 + ns(time, df=4, intercept=FALSE)
mFull <- ~-1 + ns(time, df=4, intercept=FALSE) + ns(time, df=4, intercept=FALSE):class + class

# Create edgeSet object
edgeObj <- edgeSet(expSet,
                   full.model=mFull,
                   null.model=mNull,
                   individual=endotoxin$individual)
```

```
## Error:  could not find function "ns"
```

Determine significant genes in experiment by running `odp` or `lrt`.

```
edge.odp <- odp(edgeObj, bs.its=100, verbose=FALSE)
edge.lrt <- lrt(edgeObj, nullDistn="bootstrap", bs.its=10, verbose=FALSE)
```

```
## Error:  could not find function "lrtStat"
```

Comparing the findings between both method:

```
summary(qvalue.obj(edge.odp))

##
## Call:
## qvalue(p = pval)
##
## pi0: 0.2753
##
## Cumulative number of significant calls:
##
##           <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1
## p-value      226    374   640    787   927 1105
## q-value      176    326   669    885  1108 1406
## local FDR    143    227   430    567   702  869
##              <1
## p-value    2000
## q-value    2000
## local FDR 2000

summary(qvalue.obj(edge.lrt))

## Error:  error in evaluating the argument 'object' in selecting a method for function 'qvalue.obj':
Error:  object 'edge.lrt' not found
```

In all three cases we notice that the optimal discovery method finds more significant genes. As shown in previous research by Storey (source), the ODP finds more significant genes for a fixed FDR when compared to the likelihood ratio test and other popular statistical methods.

7 Features

The `edgeSet` object inherits all methods from the `ExpressionSet` object. This makes the use of other packages written for ExprssionSet objects compatible with the edgeSet object.

Finally, there is a method in `edge` to implement supervised normalization of microarrays on the expression matrix and the function is called `edgeSNM` which is a wrapper for the `snm` packages. Also, `edge` has a method `edgeSVA` to create surrogate variables to reduce dependence in significance analysis.

Acknowledgements

This software development has been supported in part by funding from the National Institutes of Health and the Office of Naval Research.

References

[1] Storey, J. D. The optimal discovery procedure for large-scale significance testing, with applications to comparative microarray experiments. *Journal of the Royal Statistical Society, Series B 69* (2007), 347–368.

[2] Storey, J. D., Dai, J., and Leek, J. T. The optimal discovery procedure for large-scale significance testing, with applications to comparative microarray experiments. *Biostatistics 8* (2007), 414–432.

[3] Woo, S., Leek, J. T., and Storey, J. D. A computationally efficient modular optimal discovery procedure. *Bioinformatics 27* (2011), 509–515.