`MulCom`: a `Mul`tiple `Com`parison statistical test for microarray data in Bioconductor.

Claudio Isella, Tommaso Renzulli, Davide Corà and Enzo Medico

October 31, 2011

Abstract

Many microarray experiments compare a common control group with several "test" groups, like in the case , for example of a *time-course* experiments where time zero serves as a common reference point. The `MulCom` package described here implements the Dunnett's t-test, which has been specifically developed to handle multiple comparisons against a common reference, in a version tailored for genomic data analysis that we named `MulCom` (`Mul`tiple `Com`parisons) test. The implementation includes two test parameters, namely the t value and an optional minimal fold-change value, m, with automated, permutation-based estimation of False Discovery Rate (FDR) for parameter combinations of choice.

The package permits automated optimization of the test parameters to obtain the maximum number of significant genes at a given FDR value. In this vignette we present the rationale, implementation and usage of the `MulCom` package, plus a practical application on a *time-course* microarray experiment.

Contents

1 Introduction

Together with the technical advancing of the -omics platforms and in particular of microarray analysis of gene expression, the bioinformatics community has developed several statistical tools to handle data and discriminate between truly differential gene regulation and systematic/random measurement errors deriving from the highly parallel -but poorly replicated- microarray data. Indeed, statistical analysis severely suffers from the limited number of replicates usually performed in these experiments. Typical analysis algorithms use t-test or similars to identify gene lists discriminating two experimental subgroups. However, several gene expression experiments encompass multiple experimental points to be compared with a reference point. This is the case, for example, of a *time-course*, or other experimental designs involving, multiple different treatments versus a control condition. The analysis is then aimed at assessing, for each gene, which experimental group means are significantly different from the control group mean.

In principle, it is possible to run a t-test-based method for each comparison. However, when applied to this type of data, t-test-based methods have several problems: (i) they do not correct the result of each comparison for the total number of comparisons made; (ii) information about experimental variability (the standard error) is extracted only from the a limited number of samples (the two groups actually compared) and the possibility of very low standard error occurring by chance (and therefore of false positives) are very high, (iii) the estimation of the standard error could be overextimated, leading to a high rate of false negatives. Alternatively, one could use a procedure such as the *Tukey*'s "Honestly Significantly Different" (HSD) test, designed to compare each mean with each other mean. The problem with using the *Tukey*'s HSD is that, like any other method designed for systematic pairwise comparison of all means, it over corrects for the number of comparisons made, when the only comparisons of interest are those with the common reference.

To tackle these issues and enable optimal analysis of multiple comparisons versus a reference group we designed the `MulCom` test and implemented an R-Bioconductor [3] package which implements on the basis of a modified Dunnett's t-test, adapted to microarray data analysis. The test includes an optional tunable fold-change threshold (m) and Montecarlo simulation to assess FDR. We also implemented a streamlined procedure for automated optimization of test parameters, to maximize the number of significant genes at a given FDR.

The package is loaded with the command

```
> library(Mulcom)
```

2 The `MulCom` Test

The Dunnett's t-test is a procedure designed to compare different experimental conditions with a common control condition.

The procedure for comparing each experimental mean with the control mean is called "Dunnett's test".

Dunnett's test controls the Experiment-wise Error Rate and is more powerful than tests designed to compare each mean with each other mean. The test is conducted by computing a modified t-test between each experimental group and the control group. The Dunnett's t-test formula is:

$$t = \frac{|M_i - M_c|}{\sqrt{\frac{2*MSE}{n_h}}}$$

Where M_i is the mean of the i^{th} experimental group, M_c is the mean of the control group, MSE is the mean square error as computed from the analysis of variance, and n_h is the harmonic mean of the sample sizes of the experimental group and the control group.

We integrated the Dunnett's formula a tunable m parameter to set a minimum threshold for differential expression, thereby obtaining what we called the `MulCom` test:

$$t = \frac{|M_i - M_c| - m}{\sqrt{\frac{2*MSE}{n_h}}}$$

where the m parameter is subtracted from the numerator of the test.

To define the t value threshold for significance, instead of the alpha-based Dunnett's tables, the FDR of the test is estimated by extensive sample permutations. The same approach can be used to optimally tune the m parameter, which we found to substantially improve the number of significant genes at a given FDR.

It should be anyway notices that the `MulCom` test can be converted into the classical Dunnett's t-test, by simply setting m to zero.

3 Practical Example

In this vignette th `MulCom` test is applied to benchmark dataset provided together with this vignette. It is a *time-course* experiment in which MDA-MB-435 cells have been stimulated with Hepatocyte Growth Factor (HGF) for 0, 1, 6 and 24 hours. We also added an experimental point in which MDA-MB-435 cells are transduced with integrin B4 (ITGB4) as a control (detection of significantly differential ITGB4 expression is expected). The experiment has been performed on Affymetrix (HGU133A) and Illumina (RS-8 Human Beadchip) arrays, to assess statistical test reproducibility on two independent platforms. In the "benchVign" file we included the Affymetrix, "Affy", data and the cross-annotation table, "AffyIlmn". Data was filtered for expression calls.

```
> data("benchVign")
```

3.1 MulCom Analysis

The MulCom implementation requires as starting imput files an ExpressionSet, a matrix or a data.frame, and a class vector corresponding to groups in the analysis. In this example the class vector is Affy$Groups:

```
> Affy$Groups

 [1] 0 0 1 1 2 2 3 3 4 4
```

MulCom can be applied with the following procedure:

```
> mulcomScore <- mulScores(Affy, Affy$Groups)
```

Where *mulScores* calculates the numerator and the denominator of the test without the parameters m and t.

On this output, *mulCalc* calculates the number of significant genes derived from the analysis in all the comparisons, with specific m and t (m = 0.3, t = 1.7) ("sg"):

```
> sg <- mulCalc(mulcomScore, m = 0.3, t = 3)
> sg

[1] 201 246  10  37
```

3.2 Montecarlo simulation for MulCom Test

To validate the performance of the test with the chosen parameters, the package implements the function *mulPerm* to perform a Montecarlo analysis of the data, which reiterates *mulScores* on permutated samples. The function requires the ExpressionSet, a matrix or a data.frame, the number of permutations, *np*, and *seed*, to generate reproducible permutation:

```
> permutation <- mulPerm(Affy, Affy$Groups, np = 100, seed=7)
> #permutationIlmn <- mulPerm(Ilmn, Ilmn$Groups, 5, 7)
> #load("permutation.rda")
> #load("permIlmn.rda")
```

On the basis of the permutations, *mulFSG* can be used to estimate the median number of genes called in the Montecarlo model ("fsg") and thus estimate the FDR.

```
> fsg <- mulFSG(permutation, m = 0.3, t = 3)
> fsg

[1] 2.5 5.5 3.5 3.0

> fdr <- fsg/sg
> fdr

[1] 0.01243781 0.02235772 0.35000000 0.08108108
```

This result revealed that with $m = 0.3$ and $t = 3$ applied to the whole dataset, two out of four comparison have a FDR below 0.05. Indeed, every comparison requires an independent tuning of the `MulCom` test parameters.

To facilitate optimization of the test parameters, we developed a streamlined analysis strategy. The *mulOpt* function will recursively calculate the `MulCom` test across all the experiments with specific series of m and t defined by the user, provided as vectors.

```
> optimization <- mulOpt(permutation, vm = seq(0,0.5, 0.1), vt = seq(1,3, 0.1))
```

The function *mulParOpt* is designed to identify the optimal m and t values combination leading to the maximum number of differentially regulated genes satisfying an user define FDR threshold.

The function requires the `ExpressionSet`, a matrix or a dafata.frame, the output of the *mulOpt*, an index, *ind* defining the comparison to optimize and a threshold, *ths*, defining the FDR required. In case of equal number of genes, the combination of m and t with the lower FDR will be prioritized. In case of both identical number of genes and FDR, the function will chose the highest t. The function optionally will define a graphical output to visually inspect the performance of the test at given m and t parameters for a certain comparison.

```
> h1Opt <- mulParOpt(permutation, optimization, ind = 1, th = 0.05)
```

```
[1] 1190
```

In this case the optimization with FDR = 0.05 yielded $t = 1.7$ and $m = 0.3$

```
> h1Opt
```

```
  t   m
2.0 0.2
```

```
[1] 1190
```

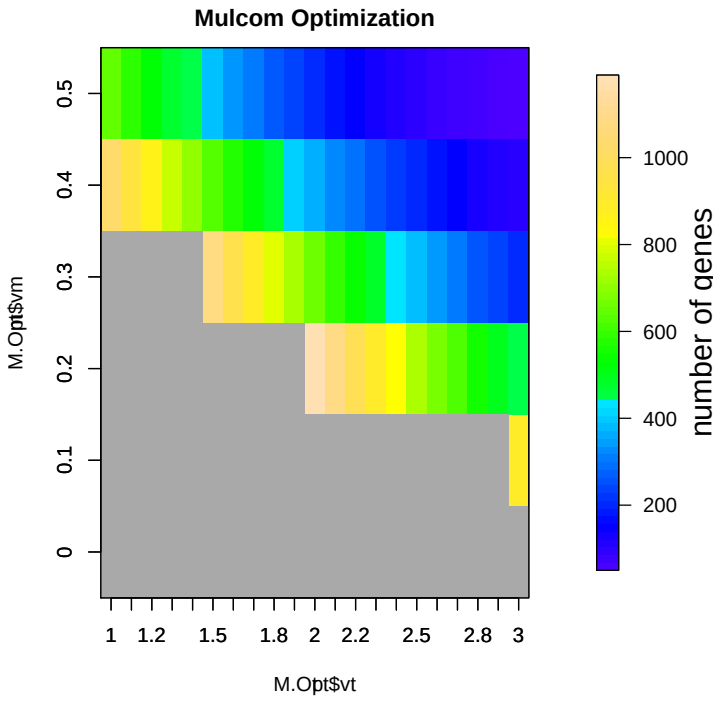

Figure 1: *plot for MulCom Figure 1: plot for MulCom Test optimization: The plot shows the number of significant genes for a series of combinations of m and t. The grey area represents the combinations that do not reach the FDR of choice*

In the example below we adopt the automated method to optimize m and t with *mulParOpt* and the function *mulDiff* to identify differentially regulated genes. *mulDiff* requires the `ExpressionSet`, a matrix or a data.frame, the permutation object and the m and t parameters:

```
> ## 1h
> h1Opt <- mulParOpt(permutation, optimization, ind = 1, th = 0.05)
```

```
[1] 1190
```

```
> affyMulc1Probes <- mulDiff(Affy, permutation, m = h1Opt[2], t = h1Opt[1], ind = 1)
```

The analysis performed on first comparison yielded 902 differentially expressed genes, with an FDR below 5%.

```
> ## 6h
> h6Opt <- mulParOpt(permutation,optimization, ind = 2, th = 0.05)
```

```
[1] 592
```

```
> affyMulc6Probes <- mulDiff(Affy, permutation, h6Opt[2], h6Opt[1], 2)
```

```
> ## 24h
> h24Opt <- mulParOpt(permutation,optimization, ind = 3, th = 0.1)

[1] 2

> affyMulc24Probes <- mulDiff(Affy, permutation, h24Opt[2], h24Opt[1], 3)

> ## B4
> b4Opt <- mulParOpt(permutation,optimization, ind = 4, th = 0.05)

[1] 28

> affyMulcB4Probes <- mulDiff(Affy, permutation, b4Opt[2], b4Opt[1], 4)

> mulcomGeneList <- unique(c(affyMulc1Probes, affyMulc6Probes, affyMulcB4Probes))
```

The analysis yielded to 1555 significant genes; we performed the same analysis on Illumina dataset: with 763 significant genes User can take advantage of the function mulInt to evalute the intersection of different lists of genes as it follows:

```
> intersection <- mulInt(affyMulc1Probes,affyMulc6Probes)
```

4 Discussion

The `MulCom` package provides a simple and powerful tool for the identification of differentially expressed genes between a control and several experimental conditions. This algorithm is better suited for multiple comparisons than t-tests performing single pairwise comparisons, as it estimates the within-group variability across all experimental groups.

```
> mulIntSym/mulTotSym

[1] 0.1397436

> samIntSym/samTotSym

[1] 0.04916512

> limIntSym/limTotSym

[1] 0.09670782
```

Notably the `Mulcom` test provided a good of cross-platform reproducibility: the signatures obtained by Mulcom in Illumina and Affymetrix higher percentage of common gene in respect to t-test, Limma or Sam [4, 5].

MulCom can also be applied to any -omics datasets such as mirnomic, proteomic and metabolomic studies, provided as an `ExpressionSet`, or any matrix formats.

5 References

References

[1] Dunnett, C. *A Multiple Comparison Procedure for Comparing Several Treatments with a Control* Journal of the American Statistical Association, Vol. 50, No. 272 Dec., 1955, pp.

[2] Gautier, L., Cope, L., Bolstad, B. M., and Irizarry, R. A. *affy—analysis of Affymetrix GeneChip data at the probe level* Bioinformatics 20, 3 (Feb. 2004), 307-315.

[3] Gentleman, R.C., et al. *Bioconductor: open software development for computational biology and bioinformatics.* Genome Biol, 2004,5:R80.

[4] Smyth, G. K. *Linear models and empirical Bayes methods for assessing differential expression in microarray experiments.* Statistical Applications in Genetics and Molecular Biology, Vol. 3, No. 1, Article 3. http://www.bepress.com/sagmb/vol3/iss1/art3

[5] Tusher VG, Tibshirani R, Chu G. *Significance analysis of microarrays applied to the ionizing radiation response.* Proc Natl Acad Sci U S A. 2001 Apr 24;98(9):5116-21. Epub 2001 Apr 17.