Testing association of a pathway with a clinical variable

Jelle Goeman Jan Oosting

May 26, 2004

Contents

1 Introduction

This document shows the functionality of the R-package *globaltest*. The main function tests whether a given group of genes is significantly associated with a clinical outcome. The explanation here focuses on practical use of the test. To understand the idea and the mathematics behind the test, and for more details on how to interpret a test result, we refer to the paper (Goeman e.a. 2004).

In the last few years there has been a shift in focus from studying the effects of single genes to studying effects of multiple functionally related genes. Most of the current methods for studying pathways involve looking at increased proportions of differentially expressed genes in pathways of interest. These methods do not identify pathways where many genes have altered their expression in a small way. The globaltest was designed to address this issue.

Essentially the globaltest is based on an emperical Bayesian generalized linear model where the regression coefficients between expression data and clinical outcome are the random variables. The test applies a goodness of fit test on this model. The globaltest is optimal when all coefficients show an effect, even if none of the individual coefficients shows a significant relation between expression and outcome. However the null hypothesis will also be rejected when there are 1 or a few coefficients with a significant relation.

By grouping together genes before testing usually the number of tests will decrease and amount of correction for multiple testing will be decreased. Genes can be grouped together in any meaningful way, for example based in function (KEGG, GO) or location (chromosome, cytoband).

In our examples we use datasets that are available from the BioConductor website. All the necessary packages to repeat the examples below are available from www.bioconductor.org.

First we load some packages from BioConductor and load the data we'll use.

```
> library(globaltest)
> library(golubEsets)
> library(hu6800)
> library(KEGG)
> library(vsn)
> data(golubMerge)
> golubM <- update2MIAME(golubMerge)

> golubX <- vsn(golubM)
```

This gives us a dataset `golubX`, which is of the format *exprSet*, the standard format for gene expression data in BioConductor. It has 7,129 genes for 72 samples. We used *vsn* to normalize the data. Any other normalization method may be used instead. Several phenotype variables are available with `golubX`, among them "ALL.AML", the clinical variable that interests us.

In this document we use the globaltest based on BioConductor *exprSet* input. For examples using simple vector or matrix input, see `help(globaltest)`.

2 Global Testing of a Single Pathway

Suppose we are interested in testing whether AML and ALL have different gene expression pattern for certain pathways from the KEGG database.

First we load all KEGG pathways. We willl use the rest in the next section.

```
> keggids <- ls(env = hu6800PATH2PROBE)
> keggPW <- multiget(keggids, env = hu6800PATH2PROBE)
> cellcycle <- keggPW[["04110"]]
```

This creates a list `keggPW` of 120 pathways, each a vector of gene names. The vector `cellcycle` is one of them. It corresponds to the Cell Cycle pathway, "04110" in the KEGG database, which corresponds to 88 probe sets on the hu6800 chip. Suppose we are predominantly interested in this pathway. We want to know whether this group of genes is associated with the clinical outcome AML versus ALL.

Always first test all genes to see if the overall gene expression pattern is different for different clinical outcomes. We can do this by saying

```
> gt.all <- globaltest(golubX, "ALL.AML")
```

The first input X should be the *exprSet* object, the second input Y the name of the clinical variable in `pData(X)`. Alternatively we can give a matrix of expressions as X and a vector as Y.

The test result is stored in a *gt.result* object, which also contains all the information needed to draw the plots.

```
> gt.all

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic

  genes tested Statistic Q Expected Q sd of Q    p-value
1  7129   7129      53.992         10  1.9035 5.1616e-35
```

We conclude that there is ample evidence that the overall gene expression profile for all 7,129 genes is associated with the clinical outcome: samples with similar AML/ALL status tend to have similar expression profiles. In cases such as this one, in which the overall expression pattern is associated with the clinical variable, we can expect most pathways (especially the larger ones) also to be associated with it.

Because `golubX` is an *exprSet*, we could simply give the name of the phenotype variable "AML.ALL" as our Y input. Alternatively, we can give a vector here. The clinical variable must define two groups (i.e. have two values) or be continuous. For a multi-valued clinical variable, the option *levels* can be used to set which groups are to be tested against each other. The function `globaltest` will automatically choose an appropriate model based on Y. To override the automatic choice, use the option *model*.

Now we test the Cell Cycle pathway that interests us:

```
> gt.cc <- globaltest(golubX, "ALL.AML", cellcycle)
> gt.cc

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
```

```
Model: logistic

  genes tested Statistic Q Expected Q sd of Q    p-value
1    90     90      70.358     10.481  3.4043 4.6074e-18
```

We conclude that the expression pattern of the cellcycle pathway is notably different between AML and ALL samples. However, as the test on all genes was significant we can generally expect most pathways to be significant as well. To get an impression of how "special" this pathway is, one can use the option *sampling*.

```
> gt.cc <- globaltest(golubX, "ALL.AML", cellcycle, sampling = TRUE)
> gt.cc

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic

  genes tested Statistic Q Expected Q sd of Q    p-value comp. p
1    90     90      70.358     10.481  3.4043 4.6074e-18   0.292
```

This gives an extra output column "comparative p", which is the fraction of random genesets of the same size as the cell cycle pathway (88 genes) which have a lower p-value than cell cycle itself. In this case around 28% of 1,000 random 'pathways' of size 88 have a lower p-value than the Cell Cycle pathway. By default 1,000 random sets are sampled; this number can be changed with the option *ndraws*.

By default the p-value of globaltest is calculated using approximate formulas which are more accurate for large sample size, but may be inaccurate for small sample size. For 72 arrays they should be accurate enough. For small sample sizes (below 30, say), it is recommended to use the permutation version of globaltest. This calculates the p-value on the basis of 10,000 permutations of the clinical variable.

```
> gt.cc <- globaltest(golubX, "ALL.AML", cellcycle, permutation = TRUE)
> gt.cc

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic

  genes tested Statistic Q Expected Q sd of Q p-value
1    90     90      70.358     10.601  3.4207       0
```

The permutation p-value is not so accurate in the lower range as it is always a multiple of one over the number of permutations and also has some sampling variance. If desired, the number of permutations can be changed with the option *nperm* to get more accurate p-values.

It is also possible to adjust the globaltest for confounders or for known risk factors. For example in the Golub Data set we may be afraid for a disturbance due to that the fact that some samples were taken from peripheral blood while

others were taken from bone marrow. We can correct for this using the option *adjust*. The option *adjust* can also be used when the study design is different from the simple 'two independent samples' design of the standard global test. In a paired design, for example, put the pair-id (as a factor) in *adjust*.

There may be more than one name of a covariate in *adjust* or *adjust* may be supplied as a *data.frame*. The variables in *adjust* may be numeric or factor.

```
> gt.cc <- globaltest(golubX, "ALL.AML", cellcycle, adjust = "BM.PB")
> gt.cc

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic
Adjusted: 99.8 % of variance of Y remains after adjustment

  genes tested Statistic Q Expected Q sd of Q    p-value
1    90     90       70.82     10.427   3.435 6.7694e-18
```

The test result also gives the percentage of the variance in Y that was left after the adjustment. It is a crude measure like $1 - R^2$. If the percentage is low, the adjustment already explained most of the variance of the outcome Y and there was not much residual variance left to test the influence of the genes. To see an example, adjust for "Source" instead of "BM.PB".

The option *adjust* may be combined with the option *sampling*. However, `globaltest` does not allow *permutation* and *sampling* or *permutation* and *adjust* be used simultaneously.

3 Multiple Global Testing

It is also possible to test many pathways at once. To test all KEGG pathways we proceed as follows:

```
> gt.kegg <- globaltest(golubX, "ALL.AML", keggPW)
```

The result `gt.kegg` can be displayed and prints a matrix whose rows correspond to the KEGG pathways. It gives the test results for each pathway. The same options described above for the single pathway `globaltest` can be applied to the multiple pathway version of `globaltest` as well.

Two functions allow further processing to be done on the test results. The function `result` extracts the whole matrix of test results, while the function `p.value` only extracts the vector of p-values. The latter function can be used for example when a correction for multiple testing is to be done. Note however that due to the extremely high correlations between the tests for different pathways, many multiple testing procedures are inappropriate for the Global Test.

A procedure that is appropriate, although very conservative, is the Benjamini and Yekutieli (2001) method. Applying this with a False Discovery Rate of 0.01, we see that even with this conservative procedure, we can see that more than half of the 120 KEGG pathways can be said to be associated with the clinical variable.

```
> library(multtest)
> mt.res <- mt.rawp2adjp(p.value(gt.kegg), proc = "BY")
> kegg.p <- mt.res$adjp[, "BY"]
> sum(kegg.p < 0.01)
```

```
[1] 82
```

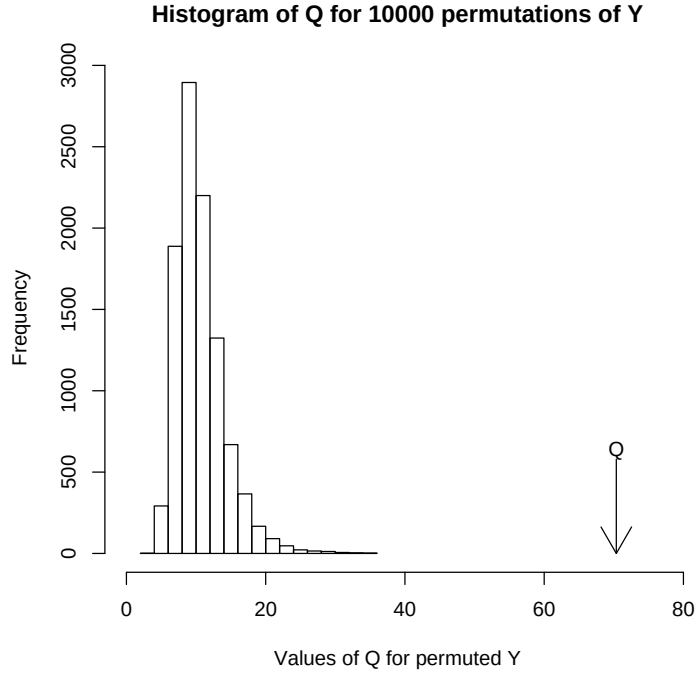
4 Diagnostic plots

There are various types of diagnostic plots available to help the user interpret the `globaltest` result. The plot `permutations` can serve as a check whether the sample size was large enough not to use the permutation version of `globaltest`. The `geneplot` visualizes the influence of individual genes on the test result. The three plots `sampleplot`, `checkerboard` and `regressionplot` all visualize the influence of individual samples. Of these three, `sampleplot` is probably the most useful.

4.1 Permutations plot

The permutations plot plots the values of the test statistic Q calculated for permutations of the clinical outcome in a histogram. The observed value of Q for the true values of the clinical outcome is marked with an arrow. Either of the commands below gives the same output.

```
> permutations(gt.cc)
> permutations(gt.kegg, "04110")
```



The output can be interpreted as a plot of the distribution of the test statistic under the null hypothesis that the pathway is not associated with the clinical variable.

The function `permutations` may not be used when the adjusted version of `globaltest` was used.

4.2 Gene Plot

The second diagnostic plot is the Gene Plot, which can be used to assess the influence of each gene on the outcome of the test. The Gene Plot gives a bar and a reference line for each gene tested. The bar indicates the influence of each gene on the test statistic (the test statistic for the group is the average of the bars for the genes). The reference line gives the expected height of the bar under the null hypothesis that the gene is not associated with the clinical outcome. Marks indicate with how many standard deviations (under the null) the bar exceeds the reference line. Finally the bars are coloured to indicate a positive or a negative association of the gene with the clinical outcome.

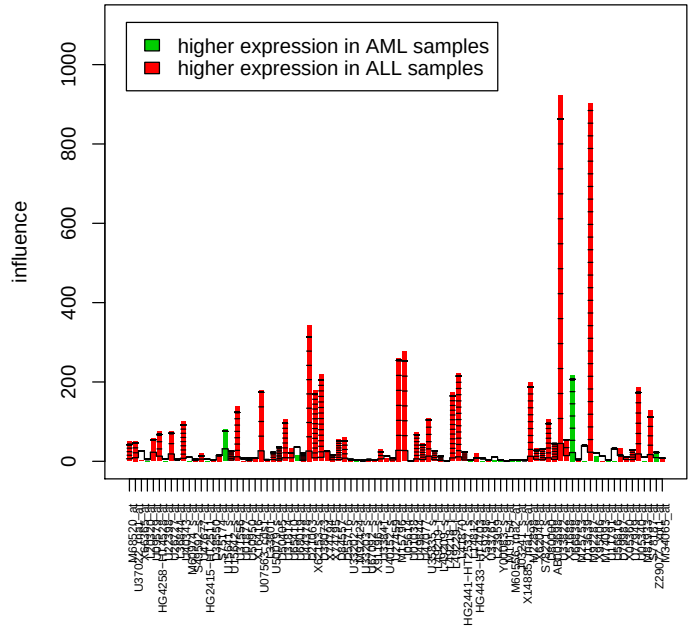
Again, either of the commands below gives the same output.

```
> geneplot(gt.cc)
> geneplot(gt.kegg, "04110")
```

For a large number of genes the plot might become overcrowded. Use the option *genesubset* to plot only a subset of the genes, *labelsize* to resize the gene labels or *drawlabels* = *FALSE* to remove them. In the latter case, use

```
> legend <- geneplot(gt.cc, drawlabels = FALSE)
```

The return of the plot is a legend connecting the gene numbers appearing in the plot if *drawlabels* = *FALSE* to the gene names.



4.3 Sample Plot

The Sample Plot looks very similar to the Gene Plot and visualizes the influence of the individual samples on the test result. It has a bar and a reference line for each sample tested. The bar indicates the influence of each sample on the test statistic, similar to the `geneplot`. The direction of the bar (upward or downward) indicates evidence against or in favour of the null hypothesis. If a sample has a positive bar, its expression profile is relatively similar to that of samples which have the same value of the clinical variable and relatively unlike the profile of the samples which have a different value of the clinical variable. If the bar is negative, it is the other way around: the sample is more similar in expression profile to samples with a different clinical variable. A small p-value will therefore generally coincide with many positive bars. If there are still tall negative bars, these indicate deviating samples.

If the null hypothesis is true the expected influence is zero. Marks on the bars indicate the standard deviation of the influence of the sample under the null hypothesis. Finally the bars are coloured to distinguish the samples. In a logistic model the colours differentiate between the original groups, in an unadjusted linear model they differentiate the above mean from the below mean values of Y. In an adjusted linear model they distinguish positive from negative residuals after adjustment.

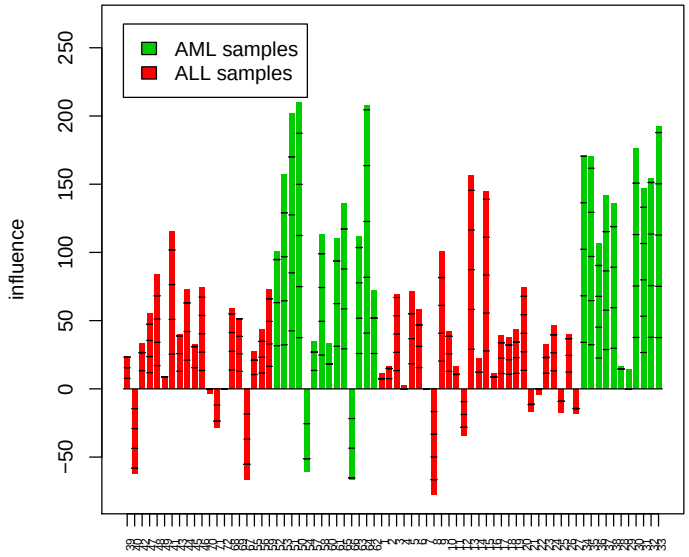
Again, either of the commands below gives the same output.

```
> sampleplot(gt.cc)
> sampleplot(gt.kegg, "04110")
```

For a large number of samples the plot might become overcrowded. Use the option *samplesubset* to plot only a subset of the samples, *labelsize* to resize the sample labels or *drawlabels* = *FALSE* to remove them. In the latter case, use

```
> legend <- sampleplot(gt.cc, drawlabels = FALSE)
```

The return of the plot is a legend connecting the sample numbers appearing in the plot if *drawlabels* = *FALSE* to the sample names.



4.4 Checkerboard plot

The fourth and fifth diagnostic plot can both also be used to assess the influence of each of the samples on the test result. The checkerboard plot visualizes the similarity between samples. It makes a square figure with the samples both on the X and on the Y-axis, so that it has all comparisons between the samples. Samples which are relatively similar are coded white and samples which are relatively dissimilar are coded black.

For easier interpretation the samples are sorted by their clinical outcome. If the test was (very) significant and the clinical outcome has two values, a typical block-like structure will appear. If the clinical outcome was continuous and the test is significant, the black squares will tend to stick together around the corners. By looking at these patterns some things can be learned about the structure of the data. For example, by looking at samples which deviate from the main pattern, outlying samples can be detected.

```
> checkerboard(gt.cc)
> checkerboard(gt.kegg, "04110")
```

The function `checkerboard` also has options *labelsize* and *drawlabels*. It returns a legend to link the numbers appearing in the plot if *drawlabels* = *FALSE* to the sample names.

4.5 Regression Plot

Using the regression plot an assessment can be made of the influence of each sample on the result of the test. It is an alternative visualization of the `sampleplot`.

The regression plot plots all pairs of samples, just like the checkerboard plot, but now showing the covariance between their clinical outcomes on the X-axis and the covariance between their gene expression patterns on the Y-axis. The comparisons of each sample with itself have been excluded.

The test statistic of the Global Test can be seen as a regression-coefficient for this plot, so it is visualized by drawing a least squares regression line. If this regression line is steep, the test statistic has a large value (and is possibly significant).

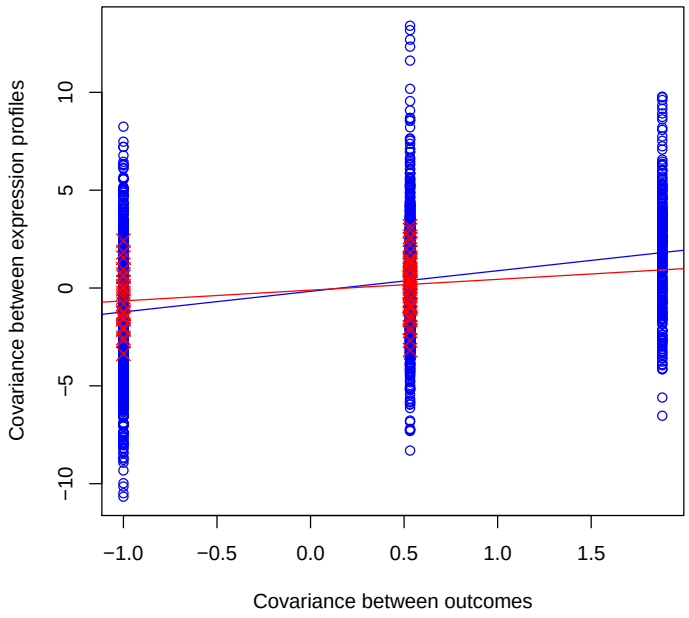
The influence of specific samples can be assessed by drawing a second regression line through only those points in the plot, which are comparisons involving the sample of interest. For example if we are interested the sample with sample name `"1"`, we take the points corresponding to the pairs (1,2) up to (1,72). If the regression line drawn through only these points deviates much from the general line, the sample deviates from the general pattern. This is especially the case if this line has a negative slope, which means that the sample is more similar (in its gene expression pattern) to the samples with a different clinical outcome than to samples with a similar clinical outcome.

If we want to test sample "1", we say:

```
> regressionplot(gt.cc, sampleid = "1")
> regressionplot(gt.kegg, "04110", sampleid = "1")
```

We can also use this plot for a group of samples, saying for example:

```
> regressionplot(gt.cc, sampleid = c("1", "2"))
```



5 Reference

- J. J. Goeman, F. de Kort, S. A. van de Geer and J. C. van Houwelingen, 2004, 'A global test for groups of genes: testing association with a clinical outcome' *Bioinformatics* 20 (1) 93–99.