Testing association of a pathway with a clinical variable

Jelle Goeman   Jan Oosting

September 15, 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(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.

```
> kegg <- as.list(hu6800PATH2PROBE)
> cellcycle <- kegg[["04110"]]
```

This creates a sorted list `kegg` of 127 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 90 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 Global Test allows three different kinds of clinical variables to be tested.

- A clinical variable defining two groups, i.e. having two values (using the logistic model). For a multi-valued clinical variable, the option *levels* can be used to set which groups are to be tested against each other.

- A continuously distributed measurement (using the linear model).

- A survival time (using the Cox model). In that case Y should contain the last observation time of each individual, and an extra argument d should be supplied which contains the event indicator, which has value *event* if an event occurred.

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 function
sampling.

```
> gt.cc <- sampling(gt.cc)
> 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   0.289
```

This gives an extra output column "comparative p", which is the fraction
of random genesets of the same size as the cell cycle pathway (90 genes) which
have a lower p-value than cell cycle itself. In this case around 29 % of 1,000
random 'pathways' of size 90 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 very
small sample size. For 72 arrays they should be accurate enough. For very
small sample sizes an alternative is to use the permutation version of globaltest.
This recalculates the p-value on the basis of 10,000 permutations of the clinical
variable.

```
> permutations(gt.cc)

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic
Using 10000 permutations of Y

  genes tested Statistic Q Expected Q sd of Q p-value
1    90     90        70.358      10.626  3.4417       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*.

The user may supply one or more names of covariates in the option *adjust* or supply *adjust* as a *data.frame*. The easiest way of adjustment, however, is by using a *formula* object as input for Y, as follows:

```
> globaltest(golubX, ALL.AML ~ BM.PB, cellcycle)

Global Test result:
Data: 72 samples with 7129 genes; 1 pathway tested
Model: logistic, ALL.AML ~ BM.PB
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 now 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 again be combined with the function *sampling*, but not with *permutation*.

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", kegg)
```

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. We can also display only some of them:

```
> gt.kegg[1:10]

Global Test result:
Data: 72 samples with 7129 genes; 10 pathways tested
Model: logistic

      genes tested Statistic Q Expected Q sd of Q    p-value
05050    24     24    17.80900     9.4062  3.9614 3.3749e-02
04510    66     66    91.52600    11.2420  3.8982 1.1342e-20
```

```
04010    268    268    47.89200    10.2370  2.2611 2.2964e-21
03020     13     13    41.83900     8.7235  5.4641 1.9498e-04
00960      5      5    35.73100     6.6946  5.1688 6.7735e-04
00910     16     16    35.89700    10.2210  5.0641 4.1299e-04
00140     14     14    18.74100     9.5184  7.2386 1.0662e-01
00460      6      6    66.50600    11.5540  8.2988 1.5242e-04
00780      2      2     0.28754     5.0492  5.9347 8.9299e-01
00280     43     43    47.40400    10.9850  5.1242 1.0228e-05
```

```
> gt.kegg["04110"]
```

```
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
04110    90     90      70.358     10.481  3.4043 4.6074e-18
```

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. See the *multtest* package for details.

We might want to sort the pathways by their p-value, and show the top five. This can be done as follows

```
> sort.gt.kegg <- gt.kegg[sort.list(p.value(gt.kegg))]
> sort.gt.kegg[1:5]
```

```
Global Test result:
Data: 72 samples with 7129 genes; 5 pathways tested
Model: logistic
```

```
      genes tested Statistic Q Expected Q sd of Q    p-value
04610    80     80     114.760     8.9325  3.4808 3.3452e-29
00590    32     32     206.470    13.8340  6.6298 8.0004e-24
04010   268    268      47.892    10.2370  2.2611 2.2964e-21
04510    66     66      91.526    11.2420  3.8982 1.1342e-20
04070    63     63      59.363     8.2905  2.6332 1.7994e-20
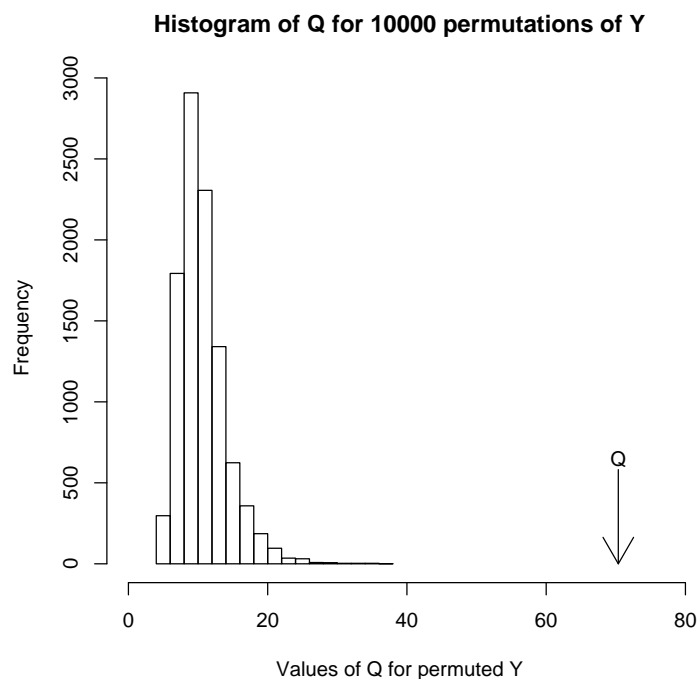```

## 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 histogram

The permutations histogram 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.

```
> hist(permutations(gt.cc))
```

Histogram of Q for 10000 permutations of Y

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. Strictly speaking, however, the permutation version of the Global Test is a different test with different properties (especially for survival data). It may give different p-values for small samples.

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.

A reference line for each bar gives the expected height of the bar under the null hypothesis that the gene is not associated with the clinical outcome (except in a survival model, where the expected height is zero). 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.

The geneplot bars have two interpretations. In the first place, the bars are the Global Test statistic for the single gene pathway containing only that gene. A positive bar that is many standard deviations above the reference line therefore indicates a gene that is significantly associated with the clinical variable in Y. Secondly, the bars indicate the influence of the gene on the test result of the whole pathway (the test statistic for the group is the average of the bars for the genes). Removing a gene with a low bar (relative to the reference line) or a negative bar from the pathway will result in a lower p-value for the pathway, removing a gene with a tall positive bar will have the opposite effect.

To plot the geneplot, use any of the commands below:

```
> geneplot(gt.cc)
> geneplot(gt.kegg, "04110")
> 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. Alternatively, we can plot part of the geneplot later, as follows

```
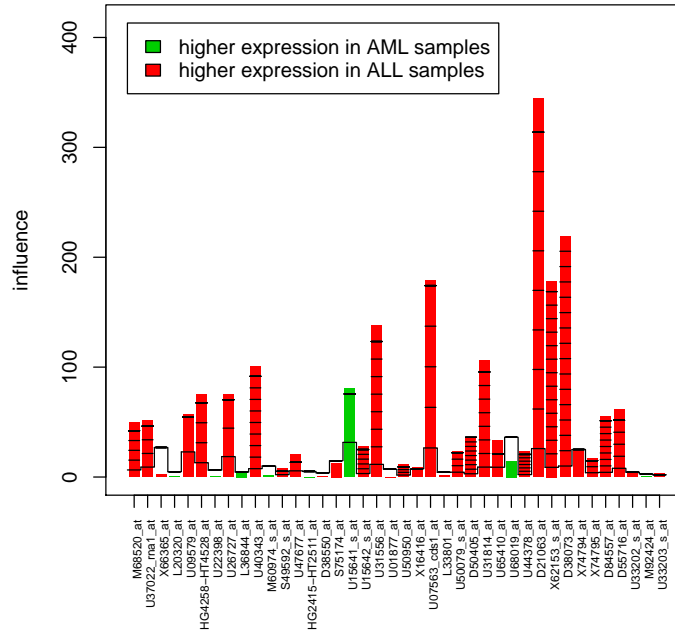> gp.cc <- geneplot(gt.cc)
> plot(gp.cc[1:40])
```

This allows one to look at subsets of a large pathway more closely. The return of the `geneplot` is an object of type *gt.barplot* containing the numbers and names appearing in the plot:

```
> gp.cc[1:10]
```

	influence	expected	sd	z-score	info
M68520_at	49.7145324	6.441802	8.919950	4.851230158	high in ALL
U37022_rna1_at	51.3140447	9.032787	12.431882	3.401034267	high in ALL
X66365_at	2.3985956	26.901314	36.936158	-0.663380264	high in ALL
L20320_at	0.4454403	4.832082	6.743828	-0.650467530	high in AML
U09579_at	57.1092102	22.869922	31.725262	1.079243673	high in ALL
HG4258-HT4528_at	74.8159474	13.040274	18.168260	3.400197667	high in ALL
U22398_at	0.1904305	6.247576	8.688779	-0.697122796	high in AML
U26727_at	75.1724814	18.584538	25.895921	2.185206816	high in ALL
L36844_at	4.5553270	4.493117	6.247460	0.009957692	high in AML
U40343_at	100.1879065	7.591272	10.507570	8.812373605	high in ALL

The option *scale* can be used to rescale the bars to have unit standard deviation.

8

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: removing a sample with a negative bar would result in a lower p-value.

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 values above the mean from the values below the mean of Y. In an adjusted linear or the survival model they distinguish positive from negative residuals after fitting the null model.

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

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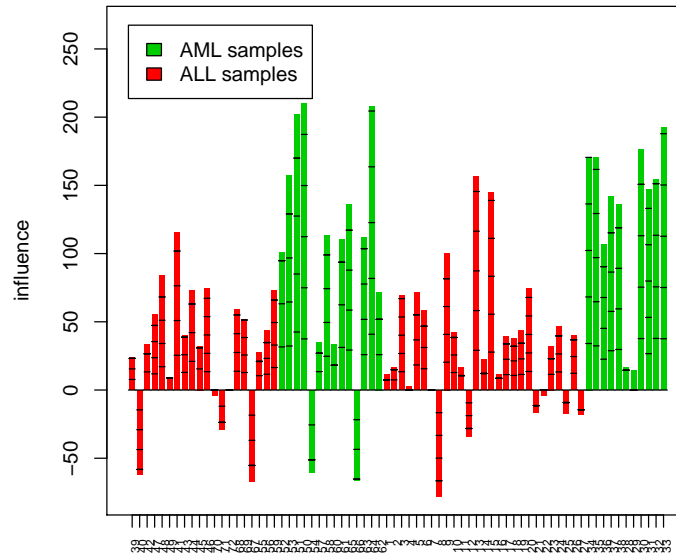> sampleplot(gt.cc)
> sampleplot(gt.kegg, "04110")
```

9

```
> sampleplot(gt.kegg["04110"])
```

The options of `sampleplot` and the resulting *gt.barplot* object are be handled in the same way as described under "geneplot".

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.

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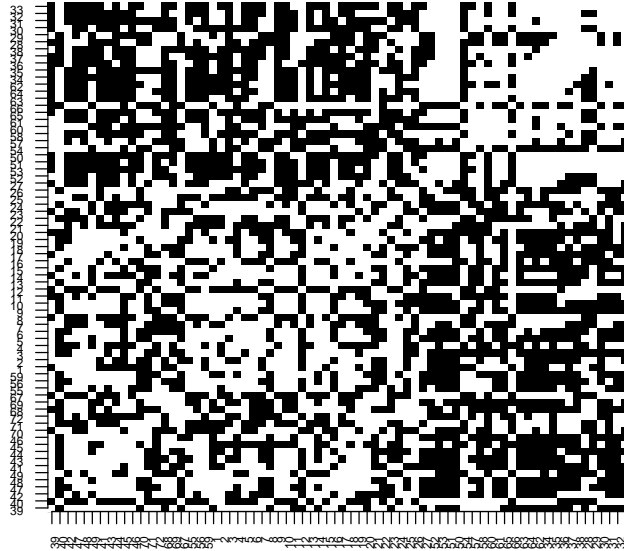> checkerboard(gt.cc)
> checkerboard(gt.kegg, "04110")
```

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:

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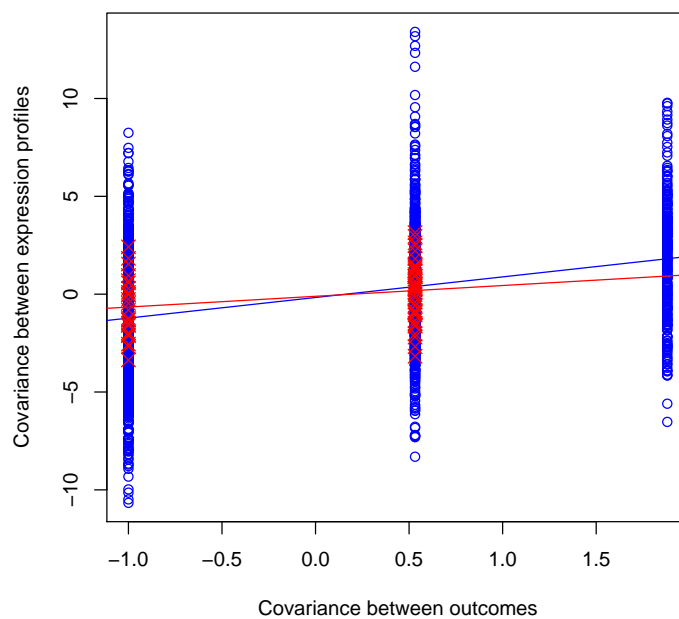> regressionplot(gt.cc, sampleid = "1")
> regressionplot(gt.kegg, "04110", sampleid = "1")
```

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.