tmod: Analysis of Transcriptional Modules

January Weiner

November 8th, 2014

The package `tmod` uses blood transcriptional modules described by Chaussabel et al. [1] and by Li et al. [2]. Furthermore, the package includes tools for testing the significance of enrichment of the modules as well as visualisation of the genes and modules. This vignette is a tutorial for the package.

In the following, we will use the Egambia data set included in the package. The data set has been generated by Maertzdorf et al. (2011)[3] and has the GEO ID GSE28623.

The included data set is a simple data frame, so to analyse it conveniently with limma, we will first generate a limma object:

```
> library(limma)
> library(tmod)
> t <- data.frame(ID=1:30,
+                 group=gsub("\\..*", "", colnames(Egambia)[-c(1:3)]))
> e <- new("EList",
+      list(E=as.matrix(Egambia[,-c(1:3)]), genes=Egambia[,c(1:3)], targets=t))
```

The data is already background corrected and normalized, so we can proceed with a differential gene expression analysis. Note that only a bit over 5000 genes from the original set of over 45000 probes is included.

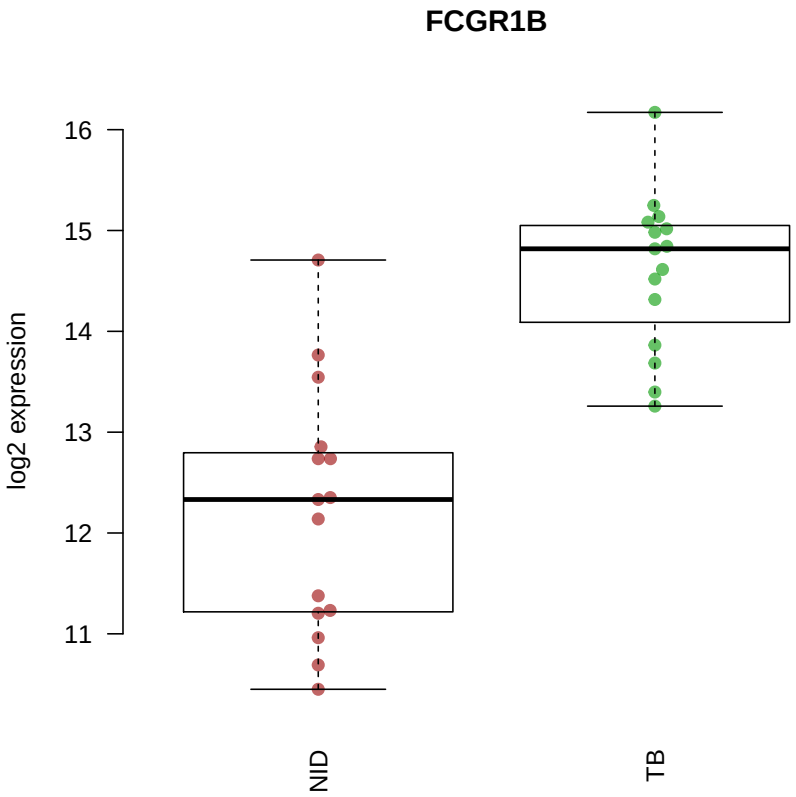
```
> d <- cbind(Intercept=rep(1, 30), TB=rep(c(0,1), each= 15))
> f <- eBayes(lmFit(e, d))
> tt <- topTable(f, coef=2, number=Inf)
> head(tt, 20)
```

```
      GENE_SYMBOL                                                GENE_NAME
4178       FAM20A                family with sequence similarity 20, member A"
20799      FCGR1B       Fc fragment of IgG, high affinity Ib, receptor (CD64)"
4122        BATF2        basic leucine zipper transcription factor, ATF-like 2
23567     ANKRD22                                  ankyrin repeat domain 22
20498       SEPT4                                                  septin 4
20360       CD274                                             CD274 molecule
2513         AIM2                                 absent in melanoma 2
24032      GOLSYN                                Golgi-localized protein
1337         ETV7                                           ets variant 7
```

```
467      SERPING1 serpin peptidase inhibitor, clade G (C1 inhibitor), member 1"
18119       BEND7                                        BEN domain containing 7
14168        GBP5                                    guanylate binding protein 5
19820       DHRS9                 dehydrogenase/reductase (SDR family) member 9
19404       GRB10                       growth factor receptor-bound protein 10
36635      FAM20A                  family with sequence similarity 20, member A
23807     KREMEN1                    kringle containing transmembrane protein 1
44719        NRG1                                                 neuregulin 1
17853        GBP1      guanylate binding protein 1, interferon-inducible, 67kDa
9007         GBP1      guanylate binding protein 1, interferon-inducible, 67kDa
25055        ATF3                              activating transcription factor 3
          EG     logFC   AveExpr         t      P.Value   adj.P.Val        B
4178   54757  2.955829  4.007327  6.200637 3.423267e-07 0.001898886 6.457171
20799   2210  2.391490 13.401207  5.946113 7.552423e-07 0.002094665 5.741043
4122  116071  2.680837 10.398520  5.797752 1.198442e-06 0.002215920 5.322491
23567 118932  2.763908  8.651749  5.624092 2.057601e-06 0.002692116 4.832003
20498   5414  3.286528  4.223270  5.480564 3.215558e-06 0.002692116 4.426508
20360  29126  2.377399  7.334747  5.463149 3.394453e-06 0.002692116 4.377314
2513    9447  1.966342  9.933621  5.462879 3.397298e-06 0.002692116 4.376553
24032  55638 -2.534812  2.221666 -5.362575 4.639596e-06 0.003018586 4.093323
1337   51513  2.844012  8.075046  5.345142 4.897651e-06 0.003018586 4.044119
467      710  2.639069  7.708228  5.150375 8.958000e-06 0.004969002 3.495088
18119 222389  2.892565  4.368001  5.102800 1.037826e-05 0.005037235 3.361233
14168 115362  1.855145 13.912431  5.087016 1.089721e-05 0.005037235 3.316851
19820  10170  1.440288  9.963398  5.004770 1.404789e-05 0.005589313 3.085834
19404   2887  1.781298  9.016137  5.003414 1.410679e-05 0.005589313 3.082028
36635  54757  1.859346  7.952156  4.927883 1.780390e-05 0.005882412 2.870292
23807  83999  2.003210 10.256227  4.924802 1.797352e-05 0.005882412 2.861666
44719   3084  2.342024  7.045822  4.923820 1.802794e-05 0.005882412 2.858916
17853   2633  1.821479  9.791495  4.904735 1.911848e-05 0.005891679 2.805490
9007    2633  1.755501 11.296024  4.815871 2.512025e-05 0.007161573 2.557145
25055    467  2.733576  3.026313  4.806897 2.582143e-05 0.007161573 2.532104
```

OK, we see some of the genes known to be prominent in the human host response to TB. We can display one of these using the showGene function (it's just a boxplot combined with a beeswarm, nothing particular):

```
> showGene(e$E["20799",], e$targets$group, main=e$genes["20799", "GENE_SYMBOL"])
```



Fine, but what about the modules?

Transcriptional module analysis

There are two main functions to understand which modules are significantly enriched. The first one, tmodHGtest, is simply a hypergeometric test on two groups of genes: 'foreground' (fg), or the list of differentially expressed genes, and 'background' (bg) – the gene universe, i.e., all genes present in the analysis. The gene identifiers used currently by tmod are HGNC identifiers, and we will use the GENE_SYMBOL field from the Egambia data set.

In this particular example, however, we have almost no genes which are significantly differentially expressed after correction for multiple testing: the power of the test with 10 individuals in each group is too low. For the sake of the example, we will therefore relax our selection. Normally, I'd use a q-value threhsold of at least 0.001.

```
> fg <- tt$GENE_SYMBOL[tt$adj.P.Val < 0.05 & abs( tt$logFC ) > 1]
```

```
> res <- tmodHGtest(fg=fg, bg=tt$GENE_SYMBOL)
> head(res)
```

```
                ID                                      Title b  B  n    N
LI.M112.0 LI.M112.0                 complement activation (I) 4 11 47 4826
LI.M11.0   LI.M11.0                enriched in monocytes (II) 4 20 47 4826
LI.M75       LI.M75                   antiviral IFN signature 3 10 47 4826
LI.S4         LI.S4                 Monocyte surface signature 3 10 47 4826
LI.S5         LI.S5                      DC surface signature 4 34 47 4826
LI.M165     LI.M165 enriched in activated dendritic cells (II) 3 19 47 4826
                E      P.Value    adj.P.Val
LI.M112.0 37.33849 2.480096e-06 0.0008581134
LI.M11.0  20.53617 3.414323e-05 0.0059067783
LI.M75    30.80426 9.906126e-05 0.0085687989
LI.S4     30.80426 9.906126e-05 0.0085687989
LI.S5     12.08010 2.957367e-04 0.0204649814
LI.M165   16.21277 7.521410e-04 0.0394125446
```

The columns in the above table contain the following:

ID The module ID. IDs starting with "LI" come from Li et al. [2], while IDs starting with "DC" have been defined by Chaussabel et al. [1].

Title The module description

b Number of genes from the given module in the fg set

B Number of genes from the module in the bg set

n Size of the fg set

N Size of the bg set

E Enrichment, calcualted as (b/n)/(B/N)

P.Value P-value from the hypergeometric test

adj.P.Val P-value adjusted for multiple testing using the Benjamini-Hochberg correction

Well, IFN signature in TB is well known. However, the numbers of genes are not high: n is the size of the foreground, and b the number of genes in fg that belong to the given module. N and B are the respective totals – size of bg+fg and number of genes that belong to the module that are found in this totality of the analysed genes. If we were using the full Gambia data set (with all its genes), we would have a different situation.

Another approach is to sort all the genes (for example, by the respective p-value) and perform a U-test on the ranks of (i) genes belonging to the module and (ii) genes that do not belong to the module. This is a bit slower, but

often works even in the case if the power of the statistical test for differential expression is low. That is, even if only a few genes or none at all are significant at acceptable thresholds, sorting them by the p-value or another similar metric can nonetheless allow to get meaningful enrichments[1]. Moreover, we do not need to set arbitrary thresholds, like p-value or logFC cutoff.

```
> l <- topTable(f, coef=2, number=Inf)$GENE_SYMBOL
> res2 <- tmodUtest(l)
> head( res2 )
```

```
               ID                                  Title      U  N1       AUC
LI.M37.0 LI.M37.0 immune activation - generic cluster 352659 100 0.7462103
LI.M37.1 LI.M37.1         enriched in neutrophils (I)  50280  12 0.8703781
LI.S4       LI.S4          Monocyte surface signature  43220  10 0.8974252
LI.M75     LI.M75             antiviral IFN signature  42996  10 0.8927741
LI.M11.0 LI.M11.0          enriched in monocytes (II)  74652  20 0.7766542
LI.M67     LI.M67           activated dendritic cells  28095   6 0.9714730
              P.Value    adj.P.Val
LI.M37.0 1.597067e-17 5.525852e-15
LI.M37.1 4.530577e-06 6.569127e-04
LI.S4    6.853638e-06 6.569127e-04
LI.M75   8.632649e-06 6.569127e-04
LI.M11.0 9.492958e-06 6.569127e-04
LI.M67   3.200305e-05 1.811391e-03
```

This list makes a lot of sense, and also is more stable than the other one: it does not depend on modules that contain just a few genes. Since the statistics is different, the b, B, n, N and E columns in the output have been replaced by the following:

U The Mann-Whitney U statistics

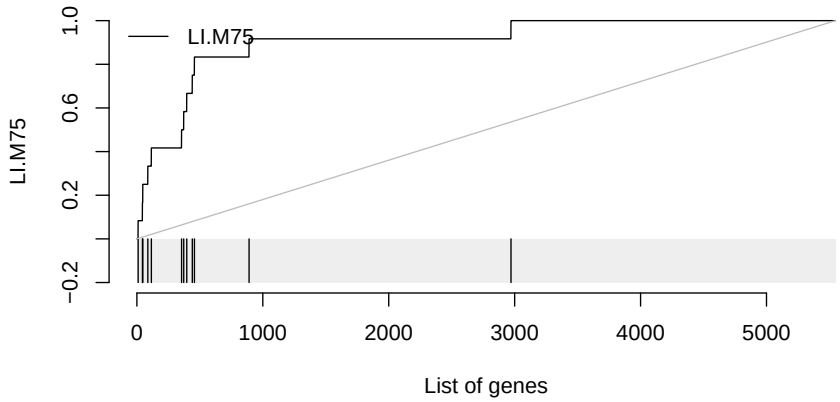
N1 Number of genes in the module

AUC Area under curve – a measure of the effect size

Let us now investigate in more detail the module LI.M75, the antiviral interferon signature. We can use the `evidencePlot` function to see how the module is enriched in the list `l`.

```
> evidencePlot(l, "LI.M75")
```

[1]The rationale is that the non-significant p-values are not associated with the test that we are actually performing, but merely used to sort the gene list. Thus, it does not matter whether they are significant or not.



In essence, this is a receiver-operator characteristic (ROC) curve, and the area under the curve (AUC) is related to the U-statistic, from which the P-value in the tmodUtest is calculated, as $\text{AUC} = \frac{U}{n_1 \cdot n_2}$. Both the U statistic and the AUC are reported. Moreover, the AUC can be used to calculate effect size according to the Wendt's formula[7] for rank-biserial correlation coefficient:

$$r = 1 - \frac{2 \cdot U}{n_1 \cdot n_2} = 1 - 2 \cdot \text{AUC}$$

In the above diagram, we see that nine out of the 10 genes that belong to the LI.M75 module and which are present in the Egambia data set are ranked among the top 1000 genes (as sorted by p-value).

Using other sets of modules

By default, tmod uses the modules published by Li et al. [2] (LI). A second set of modules was published by Chaussabel et al. [1] (DC); the module definitions can be found on a public website[2].

Depending on the `mset` parameter to the test functions, either the LI or DC sets are used, or both, if the `mset=all` has been specified.

```
> l <- topTable(f, coef=2, number=Inf)$GENE_SYMBOL
> res2 <- tmodUtest(l, mset="all")
> head( res2 )
```

```
               ID                                    Title      U  N1       AUC
LI.M37.0 LI.M37.0 immune activation - generic cluster 352659 100 0.7462103
DC.M4.2   DC.M4.2                          Inflammation  91352  20 0.9503953
```

[2] http://www.biir.net/public_wikis/module_annotation/G2_Trial_8_Modules

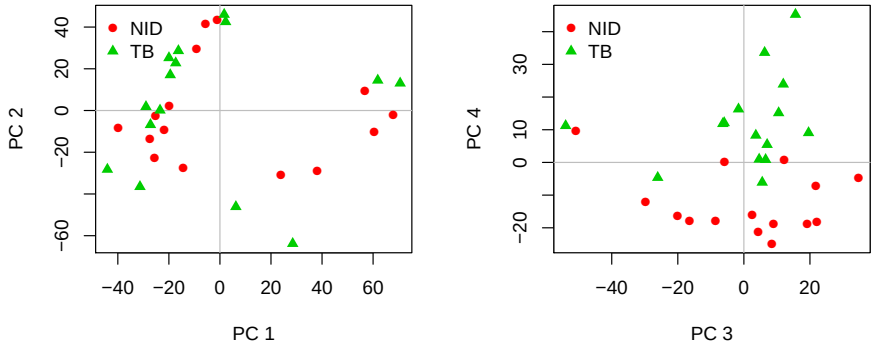
```
DC.M1.2   DC.M1.2                          Interferon  73612  17 0.9004196
DC.M3.2   DC.M3.2                        Inflammation  96366  24 0.8361620
DC.M5.15 DC.M5.15                         Neutrophils  65289  16 0.8483498
DC.M7.29 DC.M7.29                       Not Determined  77738  20 0.8087599
              P.Value    adj.P.Val
LI.M37.0 1.597067e-17 9.678227e-15
DC.M4.2  1.674762e-12 5.074530e-10
DC.M1.2  5.703006e-09 9.623646e-07
DC.M3.2  6.352241e-09 9.623646e-07
DC.M5.15 7.240084e-07 8.774982e-05
DC.M7.29 9.084521e-07 9.175366e-05
```

Combining multivariate analysis and modules

Transcriptional modules can help to understand the biological meaning of the calculated multivariate transformations. For example, consider a principal component analysis (PCA), visualised using the pca3d package [5]:

```
> library(pca3d)
> pca <- prcomp(t(e$E), scale.=TRUE)
> gr <- e$targets$group
> par(mfrow=c(1, 2))
> l<-pca2d(pca, group=gr)

Legend:
----------------------
 group:  color,  shape
----------------------
   NID:    red,     16
    TB: green3,     17

> cols <- as.character(l$colors)
> legend("topleft", as.character(l$groups),
+        pch=l$shapes,
+        col=cols, bty="n")
> l<-pca2d(pca, group=gr, components=3:4)

Legend:
----------------------
 group:  color,  shape
----------------------
   NID:    red,     16
    TB: green3,     17

> legend("topleft", as.character(l$groups),
+        pch=l$shapes,
```

```
+         col=cols, bty="n")
> par(mfrow=c(1, 1))
```



The fourth component looks really interesting. Does it correspond to the modules which we have found before? Each principal component is, after all, a linear combination of gene expression values multiplied by weights (or scores) which are constant for a given component. The i-th principal component for sample j is given by

$$PC_{i,j} = \sum_k w_{i,k} \cdot x_{k,j}$$

where k is the index of the variables (genes in our case), $w_{i,k}$ is the weight associated with the i-th component and the k-th variable (gene), and $x_{k,j}$ is the value of the variable k for the sample j; that is, the gene expression of gene k in the sample j. Genes influence the position of a sample along a given component the more the larger their absolute weight for that component.

For example, on the right-hand figure above, we see that samples which were taken from TB patients have a high value of the principal component 4; the opposite is true for the healthy controls. The genes that allow us to differentiate between these two groups will have very large, positive weights for genes highly expressed in TB patients, and very large, negative weights for genes which are highly expressed in NID, but not TB.

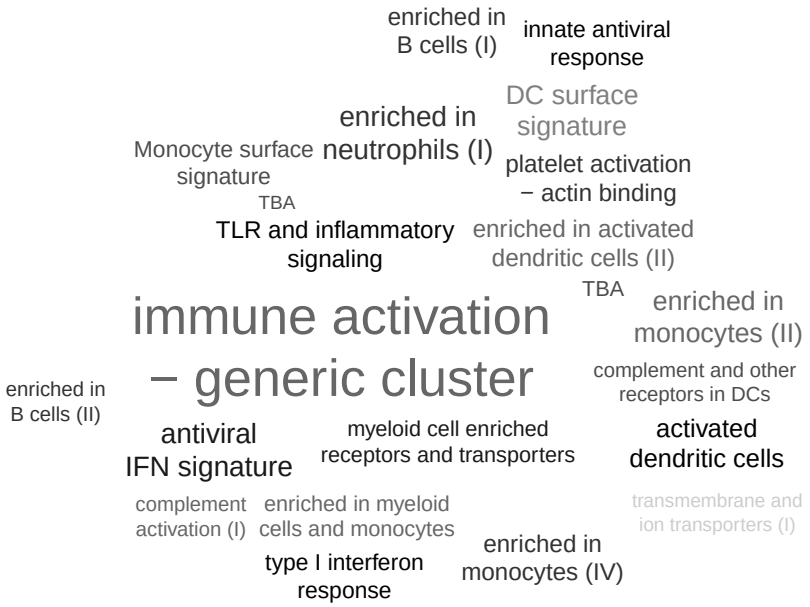
We can sort the genes by their weight in the given component, since the weights are stored in the pca object in the "rotation" slot, and use the tmodUtest function to test for enrichment of the modules.

```
> o <- order(abs(pca$rotation[,4]), decreasing=TRUE)
> l <- e$genes$GENE_SYMBOL[o]
> res <- tmodUtest(l)
> head(res)
```

```
               ID                                Title      U  N1       AUC
LI.M37.0 LI.M37.0 immune activation - generic cluster 339742 100 0.7188785
LI.M37.1 LI.M37.1         enriched in neutrophils (I)  50096  12 0.8671929
LI.M75     LI.M75             antiviral IFN signature  43379  10 0.9007267
LI.M11.0 LI.M11.0          enriched in monocytes (II)  74343  20 0.7734395
LI.S5       LI.S5                DC surface signature 115007  34 0.7058762
LI.M67     LI.M67           activated dendritic cells  28291   6 0.9782503
              P.Value    adj.P.Val
LI.M37.0 3.133111e-14 1.084056e-11
LI.M37.1 5.405722e-06 6.700097e-04
LI.M75   5.809333e-06 6.700097e-04
LI.M11.0 1.185187e-05 1.025187e-03
LI.S5    1.711493e-05 1.184353e-03
LI.M67   2.506730e-05 1.445548e-03
```

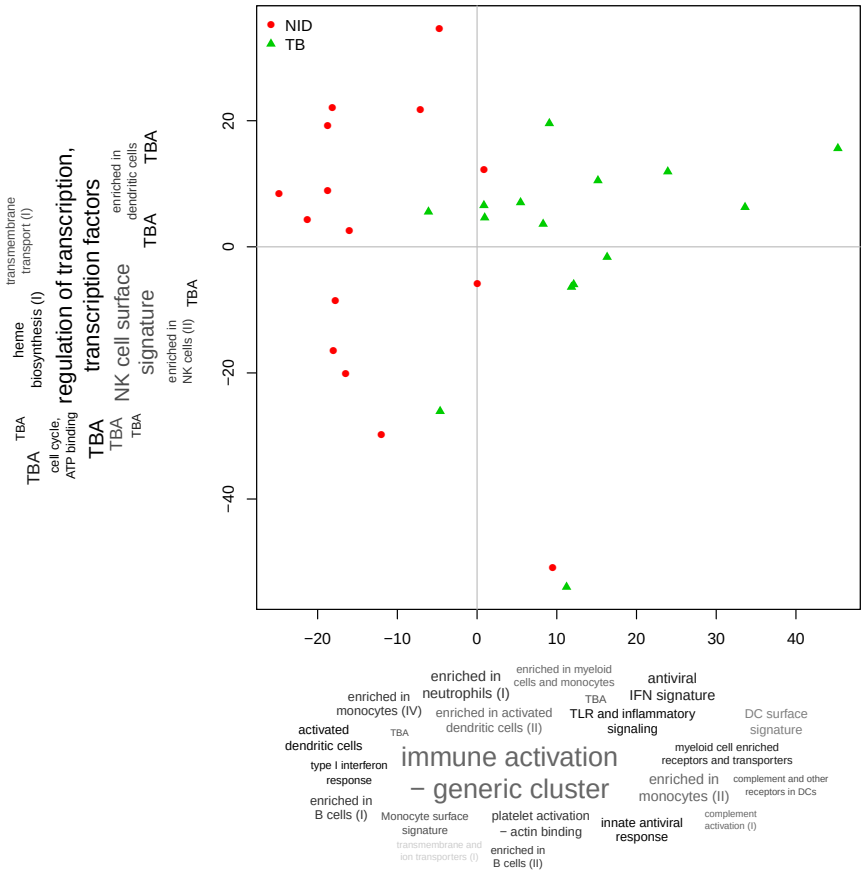
Perfect, this is what we expected: we see that the neutrophil / interferon signature which is the hallmark of the TB biosignature. We can visualise this list of results using the tagcloud package [6]. P-Values will be represented by the size of the tags, while AUC – which is a proxy for the effect size – will be shown by the color of the tag, from grey (AUC=0.5, random) to black (1):

```
> library(tagcloud)
> w <- -log10(res$P.Value)
> c <- smoothPalette(res$AUC, min=0.5)
> tags <- strmultline(res$Title)
> tagcloud(tags, weights=w, col=c)
```



We can now annotate the PCA axes using the tag clouds:

```
> par(mar=c(1,1,1,1))
> o3 <- order(abs(pca$rotation[,3]), decreasing=TRUE)
> l3 <- e$genes$GENE_SYMBOL[o3]
> res3 <- tmodUtest(l3)
> layout(matrix(c(3,1,0,2),2,2,byrow=TRUE),
+   widths=c(0.3, 0.7), heights=c(0.7, 0.3))
> # note -- PC4 is now x axis!!
> l<-pca2d(pca, group=gr, components=4:3)
> cols <- as.character(l$colors)
> legend("topleft",
+   as.character(l$groups),
+   pch=l$shapes,
+   col=cols, bty="n")
> tagcloud(tags, weights=w, col=c, fvert= 0)
> tagcloud(strmultline(res3$Title),
+   weights=-log10(res3$P.Value),
+   col=smoothPalette(res3$AUC, min=0.5),
+   fvert=1)
```



Accessing the tmod data

The `tmod` package stores its data in two data frames and two lists. These objects are loaded when the package is attached via library(), and can be immediately used without calling data(). The names mimick the various environments from Annotation.dbi packages, but currently the objects are just two lists and two data frames.

tmodMODULES is a data frame which contains general module information as defined in the supplementary materials for Li et al. [2] and Chaussabel et al. [1]

tmodGENES is a data frame which contains general gene information, including columns with HGNC ("primary"), as well as ENTREZ and REFSEQ identifiers.

tmodMODULES2GENES is a list with module IDs (same as in the "ID" column of tmodMODULES) as names. Every element of the list is a character vector with IDs ("primary" column of tmodGENES) of the genes which are included in this module.

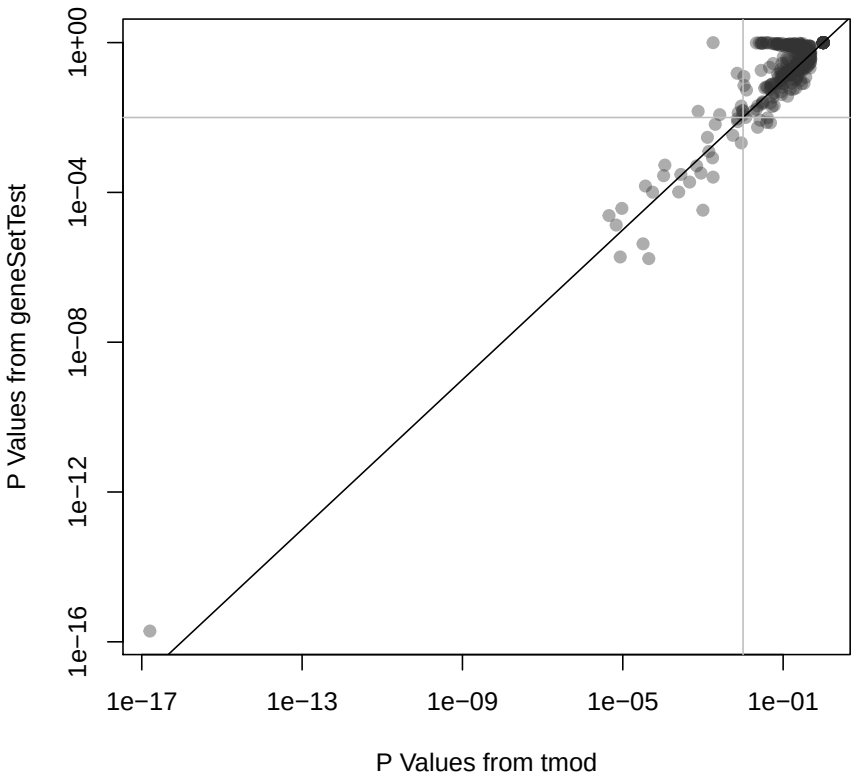
tmodGENES2MODULES is a list with gene IDs (same as in the "primary" column of tmodGENES) as names. Every element of the list is a character vector with IDs of the modules in which the gene is found.

Using these variables, one can apply any other tool for the analysis of enriched module sets available, for example, the `geneSetTest` function from the limma package (Smyth et al. [4]). We will first run `tmodUtest` setting the `qval` to `Inf` to get p-values for all modules. Then, we apply the `geneSetTest` function to each module:

```
> tt <- topTable(f, coef=2, number=Inf)
> res <- tmodUtest(tt$GENE_SYMBOL, qval=Inf)
> gstest <- function(x) {
+   sel <- tt$GENE_SYMBOL %in% tmodMODULES2GENES[[x]]
+   geneSetTest(sel, tt$logFC)
+ }
> gst <- sapply(res$ID, gstest)
```

Are the results of both statistical approaches similar? `tmod` uses a very simple statistical test. The approach from `geneSetTest` is more complex, but similar in principle.

```
> plot(res$P.Value, gst,
+   log="xy", pch=19,
+   col="#33333366",
+   xlab="P Values from tmod",
+   ylab="P Values from geneSetTest")
> abline(0,1)
> abline(h=0.01, col="grey")
> abline(v=0.01, col="grey")
```



On the plot above, the p-values from `tmod` are plotted against the p-values from `geneSetTest`. As you can see, in this particular example, both methods give very similar results.

References

[1] Damien Chaussabel, Charles Quinn, Jing Shen, Pinakeen Patel, Casey Glaser, Nicole Baldwin, Dorothee Stichweh, Derek Blankenship, Lei Li, Indira Munagala, et al. A modular analysis framework for blood genomics studies: application to systemic lupus erythematosus. *Immunity*, 29(1):150–164, 2008.

[2] Shuzhao Li, Nadine Rouphael, Sai Duraisingham, Sandra Romero-Steiner, Scott Presnell, Carl Davis, Daniel S Schmidt, Scott E Johnson, Andrea Milton, Gowrisankar Rajam, et al. Molecular signatures of antibody responses derived from a systems biology study of five human vaccines. *Nature immunology*, 2013.

[3] Jeroen Maertzdorf, Martin Ota, Dirk Repsilber, Hans J Mollenkopf, January Weiner, Philip C Hill, and Stefan HE Kaufmann. Functional correlations of pathogenesis-driven gene expression signatures in tuberculosis. *PloS one*, 6(10):e26938, 2011.

[4] Gordon K Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pages 397–420. Springer, New York, 2005.

[5] January Weiner. *pca3d: Three dimensional PCA plots*, 2013. R package version 0.4.

[6] January Weiner. *tagcloud: Tag Clouds*, 2014. R package version 0.5.

[7] Hans W Wendt. Dealing with a common problem in social science: A simplified rank-biserial coefficient of correlation based on the u statistic. *European Journal of Social Psychology*, 2(4):463–465, 1972.