mdgsa Library

David Montaner

(2014-11-10)

Contents

Introduction

The `mdgsa` library implements the *gene set analysis* methodology developed in Montaner and Dopazo (2010). It presents a flexible framework for analyzing the enrichment of *gene sets* along a given *ranking* of genes. The novelty is that, not just one *ranking index* but two, may be analyzed and jointly explored in a **multidimensional gene set analysis**.

As classical *GSEA*, our approach allows for the functional profiling of isolated genomic characteristics; differential gene expression, copy number analyses, or variant to disease associations may be interpreted in terms of *gene sets* using the `mdgsa` package. But more interestingly, our multivariate approach may be used to find out *gene set* enrichments due to the combined effect of two of such genomic dimensions. We could for instance detect *gene sets* affected by the interaction of gene expression changes and copy number alterations.

Citation

Further description of the `mdgsa` methods may be found at:

> *Multidimensional gene set analysis of genomic data.*
> David Montaner and Joaquin Dopazo.
> PLoS One. 2010 Apr 27;5(4):e10348. doi: 10.1371/journal.pone.0010348.

Functional Profiling of Gene Expression Data

In this tutorial we use the data in the Acute Lymphocytic Leukemia expression dataset package of Bioconductor. First we will use the limma library to compute a differential gene expression analysis. Then we will use the functions `uvGsa` and `mdGsa` in the `mdgsa` package to perform uni-dimensional and bi-dimensional gene set analyses respectively. The functional interpretation will be done in terms of KEGG Pathways. The annotation will be taken from the hgu95av2.db library in Bioconductor.

First we load the data and describe the design matrix of the experiment

```
library (ALL)
data (ALL)

des.mat <- model.matrix (~ 0 + mol.biol, data = ALL)
colnames (des.mat) <- c("ALL", "BCR", "E2A", "NEG", "NUP", "p15")
head (des.mat)
```

```
##       ALL BCR E2A NEG NUP p15
## 01005   0   1   0   0   0   0
## 01010   0   0   0   1   0   0
## 03002   0   1   0   0   0   0
## 04006   1   0   0   0   0   0
## 04007   0   0   0   1   0   0
## 04008   0   0   0   1   0   0
```

Then we can use `limma` to carry out some gene expression comparisons. We can for instance compare *ALL* samples to *NEG* control samples or explore gene differential expression between *BCR* and *NEG*.

```
library (limma)
cont.mat <- makeContrasts (ALL-NEG, BCR-NEG, levels = des.mat)
cont.mat
```

```
##        Contrasts
## Levels ALL - NEG BCR - NEG
##    ALL         1         0
##    BCR         0         1
##    E2A         0         0
##    NEG        -1        -1
##    NUP         0         0
##    p15         0         0
```

```
fit <- lmFit (ALL, design = des.mat)
fit <- contrasts.fit (fit, cont.mat)
fit <- eBayes (fit)
```

From this analysis we get *test statistics* and *p-values* for each of the two contrasts

```
fit$t[1:3,]
```

```
##            Contrasts
##              ALL - NEG  BCR - NEG
##    1000_at   -2.2610931 -0.7684296
##    1001_at   -1.0962463  0.2064388
##    1002_f_at  0.7978798 -1.7527367
```

```
fit$p.value[1:3,]
```

```
##             Contrasts
##              ALL - NEG  BCR - NEG
##    1000_at    0.02548234 0.44368153
##    1001_at    0.27507879 0.83678412
##    1002_f_at 0.42645360 0.08209929
```

These gene level information may be now interpreted in terms of *gene sets*. For this example we will carry out a *gene set analysis* using the functional blocks described in KEGG, but any other functional data base such as the Gene Ontology or even a customized one may be analyzed using mdgsa. We can get the KEGG annotation from the hgu95av2.db library as follows.

```
library (hgu95av2.db)
anmat <- toTable (hgu95av2PATH)
anmat[1:3,]
```

```
##    probe_id path_id
## 1   1000_at   04010
## 2   1000_at   04012
## 3   1000_at   04062
```

Univariate Gene Set Analysis

We can now carry out the functional interpretation of the contrast "BCR - NEG" for instance.

The data needed for the gene set analysis are the p-values and test statistics returned by limma at gene level.

```
fit$t[1:3, "BCR - NEG"]
```

```
##    1000_at     1001_at   1002_f_at
## -0.7684296   0.2064388  -1.7527367
```

```
fit$p.value[1:3, "BCR - NEG"]
```

```
##    1000_at     1001_at   1002_f_at
## 0.44368153  0.83678412  0.08209929
```

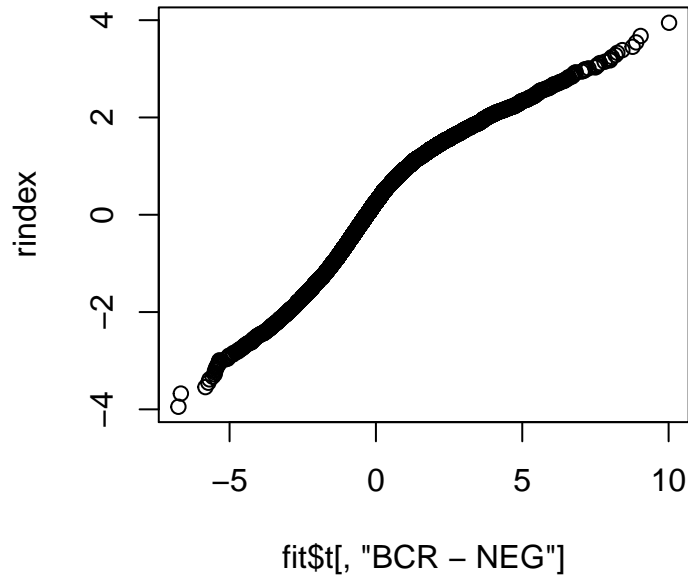We load the mdgsa library.

```
library (mdgsa)
```

The first step in the procedure is to combine p-values and test statistics in a single ranking value.

```
rindex <- pval2index (pval = fit$p.value[,"BCR - NEG"], sign = fit$t[,"BCR - NEG"])
rindex <- indexTransform (rindex)
rindex[1:3]
```

```
##    1000_at     1001_at   1002_f_at
## -0.4038315   0.4309088  -1.2145063
```
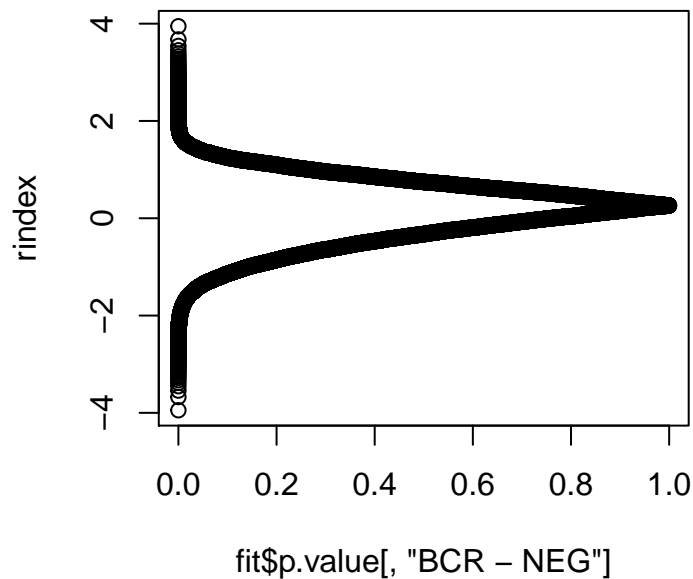
This *ranking index* keeps the sign of the test statistic; that is, the information of whether the gene is over or underexpressed.

```
plot (fit$t[,"BCR - NEG"], rindex)
```


fit$t[, "BCR – NEG"]

but the evidence of the differential expression is taken directly from the p-value

```
plot (fit$p.value[,"BCR - NEG"], rindex)
```


fit$p.value[, "BCR – NEG"]

Next we will need to format the annotation. The function `annotMat2list` in the `mdgsa` library converts the annotation matrix into a list.

```
anmat[1:3,]
```

```
##    probe_id path_id
```

```
## 1   1000_at    04010
## 2   1000_at    04012
## 3   1000_at    04062
```

```
annot <- annotMat2list (anmat)
length (annot)
```

```
## [1] 228
```

Each element of the list contains the gene names of a gene set.

```
lapply (annot[1:3], head, n= 3)
```

```
## $`00010`
## [1] "2035_s_at"  "31488_s_at" "32210_at"
##
## $`00020`
## [1] "160044_g_at" "32332_at"    "32546_at"
##
## $`00030`
## [1] "31485_at" "32210_at" "32336_at"
```

It is also important to make sure that the *gene universe* described by the *ranking index* and the *annotation* are concordant. The function `annotFilter` encompasses the annotation list to the names in the ranking index; it is also used to exclude functional blocks too small to be considered gene sets, or too big to be specific of any biological process of interest.

```
annot <- annotFilter (annot, rindex)
```

```
##
## 42.57% of the genes in the index are annotated in the list.
##
## Filtering annotation by size:
##     18 small blocks removed.
##     2 big blocks removed.
##     208 blocks remain in the annotation.
```

Now everything is ready to carry out the *univariate* gene set analysis.

```
res.uv <- uvGsa (rindex, annot)
```

```
## Analyzed blocks:
## 100, 200,
## time in seconds:
##    user  system elapsed
##  11.289   0.008  11.291
```

The output of the `uvGsa` function is a data frame which rows correspond to functional blocks analyzed.

```
res.uv[1:3,]
```

```
##       N        lor       pval      padj
## 00010 65 -0.08900436 0.47430323 1.0000000
## 00020 33 -0.26525586 0.12586858 1.0000000
## 00030 20 -0.53224486 0.01979982 0.3164805
```

The `uvGsa` function fits a *logistic regression model* relating, the probability of genes belonging to the gene set, with the value of the ranking statistic.

Significant and **positive** *log odds ratio* (`lor`) indicate that the gene set is enriched in those genes with **high values** of the ranking statistic. In our example this means that genes up regulated in *BCR* compared to *NEG* are more likely to belong to the functional block. We could also say that the block of genes shows a significant degree of over expression *BCR* compared to *NEG*.

On the other hand, when a gene set has a **negative** *log odds ratio* we can say that the genes in the set are more likely to be associated to **low values**, negative in our case, of the ranking statistics. In our case this means that the gene set is down regulated in *BCR* compared to *NEG*.

The function `uvPat` helps you classifying the analyzed gene sets

```
res.uv[,"pat"] <- uvPat (res.uv, cutoff = 0.05)
table (res.uv[,"pat"])
```

```
##
## -1   0   1
## 11 155  42
```

Positive values (1) correspond to significant and positive *log odds ratios*. Negative values (-1) correspond to significant and negative *log odds ratios*. Zeros correspond to non enriched blocks.

As in this example we are analyzing KEGG pathways we can use the function `getKEGGnames` to find out the "name" of the pathways[1].

```
res.uv[,"KEGG"] <- getKEGGnames (res.uv)
```

Finally, the function `uvSignif` may help us displaying just the enriched blocks.

```
res <- uvSignif (res.uv)
res[,c("pat", "KEGG")]
```

```
##       pat                                               KEGG
## 05332   1                        Graft-versus-host disease
## 05416   1                                Viral myocarditis
## 05330   1                                Allograft rejection
## 04612   1                  Antigen processing and presentation
## 05131   1                                       Shigellosis
## 04142   1                                          Lysosome
## 04145   1                                          Phagosome
## 04940   1                          Type I diabetes mellitus
## 04670   1                 Leukocyte transendothelial migration
```

---

[1]Similar function `getGOnames` is available to be used with Gene Ontology terms.

```
## 05100   1              Bacterial invasion of epithelial cells
## 05323   1                             Rheumatoid arthritis
## 05145   1                                    Toxoplasmosis
## 05140   1                                    Leishmaniasis
## 04520   1                                 Adherens junction
## 04380   1                         Osteoclast differentiation
## 04010   1                            MAPK signaling pathway
## 04621   1                  NOD-like receptor signaling pathway
## 04510   1                                   Focal adhesion
## 05150   1                   Staphylococcus aureus infection
## 05322   1                     Systemic lupus erythematosus
## 04672   1           Intestinal immune network for IgA production
## 05220   1                         Chronic myeloid leukemia
## 05310   1                                           Asthma
## 04060   1              Cytokine-cytokine receptor interaction
## 05146   1                                        Amoebiasis
## 04144   1                                      Endocytosis
## 04350   1                        TGF-beta signaling pathway
## 04810   1                    Regulation of actin cytoskeleton
## 05130   1                 Pathogenic Escherichia coli infection
## 04722   1                    Neurotrophin signaling pathway
## 04514   1                      Cell adhesion molecules (CAMs)
## 04062   1                       Chemokine signaling pathway
## 05142   1             Chagas disease (American trypanosomiasis)
## 05320   1                         Autoimmune thyroid disease
## 05144   1                                          Malaria
## 04210   1                                        Apoptosis
## 05120   1 Epithelial cell signaling in Helicobacter pylori infection
## 05412   1      Arrhythmogenic right ventricular cardiomyopathy (ARVC)
## 04530   1                                    Tight junction
## 04662   1                   B cell receptor signaling pathway
## 04360   1                                    Axon guidance
## 04666   1                  Fc gamma R-mediated phagocytosis
## 00670  -1                        One carbon pool by folate
## 00900  -1                  Terpenoid backbone biosynthesis
## 03013  -1                                    RNA transport
## 03420  -1                        Nucleotide excision repair
## 03450  -1                         Non-homologous end-joining
## 03008  -1                 Ribosome biogenesis in eukaryotes
## 00280  -1         Valine, leucine and isoleucine degradation
## 03440  -1                        Homologous recombination
## 00100  -1                              Steroid biosynthesis
## 03430  -1                                   Mismatch repair
## 03030  -1                                  DNA replication
```

Multivariate Gene Set Analysis

But with the `mdgsa` library we can analyze not just one but two ranking statistics at a time.

In our example we were interested not in just one differential expression contrast but in two: ALL vs. NEG and BCR vs. NEG. We used `limma` to fit this two contrasts (see previous sections) and computed gene statistics and p-values for for each of them.

```
fit$t[1:3,]
```

```
##            Contrasts
##              ALL - NEG  BCR - NEG
##    1000_at   -2.2610931 -0.7684296
##    1001_at   -1.0962463  0.2064388
##    1002_f_at  0.7978798 -1.7527367
```

```
fit$p.value[1:3,]
```

```
##            Contrasts
##              ALL - NEG  BCR - NEG
##    1000_at    0.02548234 0.44368153
##    1001_at    0.27507879 0.83678412
##    1002_f_at 0.42645360 0.08209929
```

We can combine this two matrices in a single one containing a *ranking statistic* for each contrast, just as we did in the *univariate* example.

```
rindex <- pval2index (pval = fit$p.value, sign = fit$t)
rindex <- indexTransform (rindex)
rindex[1:3,]
```

```
##            Contrasts
##              ALL - NEG  BCR - NEG
##    1000_at   -1.6649597 -0.4038315
##    1001_at   -0.9075447  0.4309088
##    1002_f_at  0.6548653 -1.2145063
```

Now we can explore the bi-dimensional distribution of this ranking indexes

```
plot (rindex, pch = ".")
```
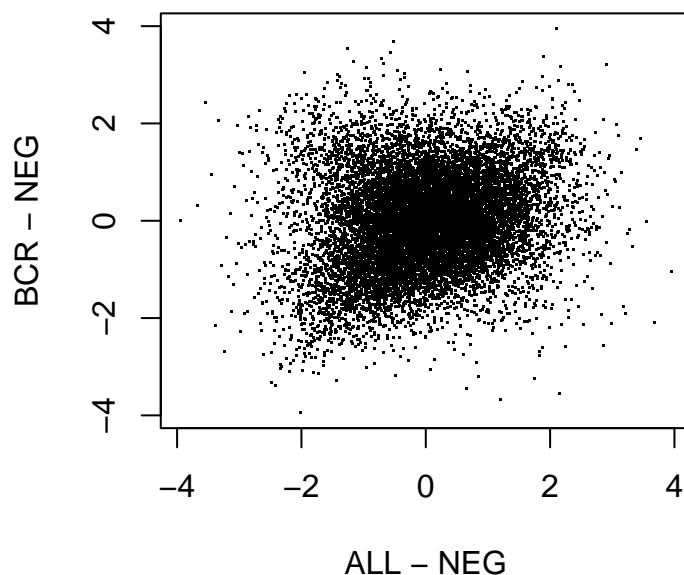
and search for gene sets enrichment patterns in both *dimensions*

The same annotation list we **filtered** in the previous section may be used in this second example. Thus everything is ready to carry out our *multidimensional* analysis.

```
res.md <- mdGsa (rindex, annot)
```

```
## Analyzed blocks:
## 100, 200,
## time in seconds:
##    user  system elapsed
##  13.021   0.016  13.032
```

As in the *univariate* analysis, the output of the `mdGsa` function is a data frame with a row per analyzed gene set.

```
res.md[1:3,]
```

```
##          N lor.ALL - NEG lor.BCR - NEG       lor.I pval.ALL - NEG
## 00010 65     0.1975707    -0.1309072  0.01695617       0.1227818
## 00020 33     0.0604673    -0.2965314 -0.07414872       0.7399391
## 00030 20     0.1044882    -0.5664453 -0.01135237       0.6959869
##        pval.BCR - NEG    pval.I padj.ALL - NEG padj.BCR - NEG padj.I
## 00010     0.31386075 0.8729905             1      1.0000000      1
## 00020     0.09865711 0.6210410             1      1.0000000      1
## 00030     0.01887660 0.9536525             1      0.3226767      1
```

The column of the row contain *log odds ratios* and p-values for each of the analyzed *dimensions* and also for their *interaction* effect. The function `mdPat` helps clarifying the bi-dimensional pattern of enrichment.

```
res.md[,"pat"] <- mdPat (res.md)
table (res.md[,"pat"])
```

```
##
## b13  NS q2f q3f  yh  yl
##   1 153   1   1  43   9
```

And as before we can incorporate the KEGG names to our results.

```
res.md[,"KEGG"] <- getKEGGnames (res.md)
res.md[1:3,]
```

Thus we could for instance explore the KEGG classified as having a with a **q3f** pattern. The **q3f** classification means that the genes of this gene set are located in the **third quadrant** of the *bivariate ranking index* representation.

The `plotMdGsa` function help us understanding such pattern.

```
Q3 <- rownames (res.md)[res.md$pat == "q3f"]
Q3
```

```
## [1] "03030"
```

```
plotMdGsa (rindex, block = annot[[Q3]], main = res.md[Q3, "KEGG"])
```

DNA replication
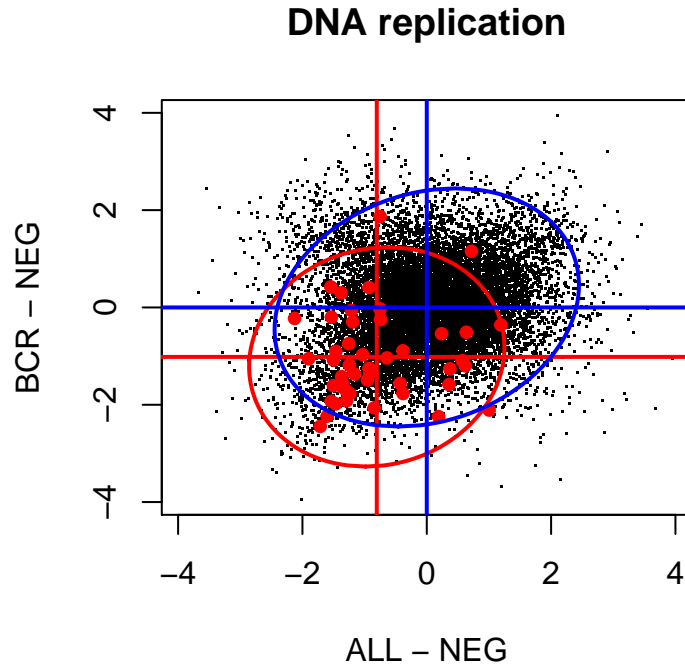
Red dots in the figure represent the genes within the gene set. They show, in both dimensions of our ranking, values significantly more negatives that the remaining genes in the study. This same pattern may be appreciated in the ellipses drawn in the figure. The blue one represents a *confidence region* for all the genes in the study. The red one shows the same *confidence region* but just for those genes within the gene set. We can see how the distribution of the genes in KEGG *03030* is displaced towards the third quadrant of the plot. This indicates us that "DNA replication" is a *pathway* jointly down regulated in both *ALL* and *BCR* when compared to the controls in the *NEG* group.

Similarly we can explore a *bimodal* (**b13**) KEGG. This pattern classification indicates us that the functional block has two **sub-modules** of genes with opposite patterns of expression. One subset of the KEGG is up-regulated in both conditions while the other subset is down-regulated also in both dimensions analyzed.

It may be worth pointing here that, this *bimodal* pattern will be missed by standard *univariate* gene set methods, and that it may be just detected in the *multidimensional* analysis.

```
BI <- rownames (res.md)[res.md$pat == "b13"]
plotMdGsa (rindex, block = annot[[BI]], main = res.md[BI, "KEGG"])
```

Primary immunodeficiency
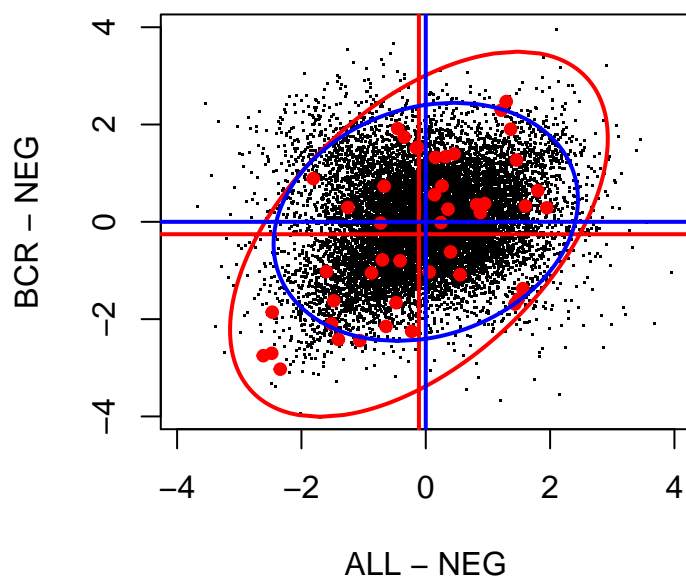
As third example we could display some KEGG enriched in one dimension but not in the other. The **yh** pattern indicates gene set over-representation just in the *Y* axis but not in the horizontal axis. In our case, KEGG pathways with at **yh** pattern are those over expressed in *BRC* compared to *NEG* but not enriched in the comparison between *ALL* and *NEG*.

Similarly an **yl** pattern indicates a down-regulation of the block in the *BRC vs. NEG* comparison but not in the *ALL vs. NEG* one.

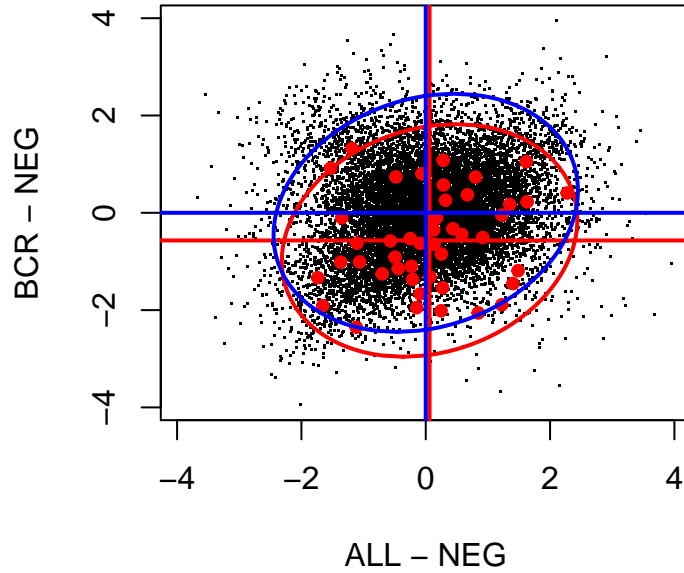The plot below displays one of such **yl** classified KEGGs.

```
rownames (res.md)[res.md$pat == "yl"]
```

```
## [1] "00100" "00280" "00900" "03008" "03013" "03430" "03440" "03450" "04146"
```

```
YL <- "00280"
plotMdGsa (rindex, block = annot[[YL]], main = res.md[YL, "KEGG"])
```

Valine, leucine and isoleucine degradatio

All possible multidimensional enrichment **patterns** are listed in the Appendix.

Appendix

1.- Multidimensional Functional Classification

All possible functional block classifications in the bi-dimensional gene set analysis are:

- **q1i**: block displaced toward quadrant **1** ($0 < X$ & $0 < Y$) with interaction.

- **q2i**: block displaced toward quadrant **2** ($0 > X$ & $0 < Y$) with interaction.

- **q3i**: block displaced toward quadrant **3** ($0 > X$ & $0 > Y$) with interaction.

- **q4i**: block displaced toward quadrant **4** ($0 < X$ & $0 > Y$) with interaction.

- **q1f**: block displaced toward quadrant **1**, no interaction.

- **q2f**: block displaced toward quadrant **2**, no interaction.

- **q3f**: block displaced toward quadrant **3**, no interaction.

- **q4f**: block displaced toward quadrant **4**, no interaction.

- **xh**: block shifted to **positive X** values.

- **xl**: block shifted to **negative X** values.

- **yh**: block shifted to **positive Y** values.

- **yl**: block shifted to **negative Y** values.

- **b13**: bimodal block. Half of the genes displaced towards quadrant **1** and the other half towards quadrant **3**.

- **b24**: bimodal block. Half of the genes displaced towards quadrant **2** and the other half towards quadrant **4**.

- **NS**: **non significant** block.

A detailed description of each of the patterns can be found in Montaner and Dopazo (2010).

The function `mdPat` in the `mdgsa` package is devised to help the user classifying bi-dimensional GSA results in such patterns.

Session Info

- R version 3.1.2 (2014-10-31), `x86_64-pc-linux-gnu`

- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=es_ES.UTF-8`, `LC_COLLATE=en_US.UTF-8`, `LC_MONETARY=es_ES.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=es_ES.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=es_ES.UTF-8`, `LC_IDENTIFICATION=C`

- Base packages: base, datasets, graphics, grDevices, methods, parallel, stats, stats4, utils

- Other packages: ALL 1.7.1, AnnotationDbi 1.28.1, Biobase 2.26.0, BiocGenerics 0.12.0, DBI 0.3.1, devtools 1.6.1, GenomeInfoDb 1.2.2, hgu95av2.db 3.0.0, IRanges 2.0.0, knitr 1.7, limma 3.22.1, markdown 0.7.4, mdgsa 0.99.0, org.Hs.eg.db 3.0.0, RSQLite 1.0.0, S4Vectors 0.4.0

- Loaded via a namespace (and not attached): cluster 1.15.3, compiler 3.1.2, digest 0.6.4, evaluate 0.5.5, formatR 1.0, GO.db 3.0.0, grid 3.1.2, htmltools 0.2.6, KEGG.db 3.0.0, lattice 0.20-29, Matrix 1.1-4, rmarkdown 0.3.3, stringr 0.6.2, tools 3.1.2, yaml 2.1.13