Along signal paths: an empirical gene set approach exploiting pathway topology

1.1 clipper approach

Different experimental conditions are usually compared in terms of their gene expression mean differences. In the univariate case, if a gene set changes significantly its multivariate mean expression in one condition with respect to the other, it is said to be differentially expressed. However, the difference in mean expression levels does not necessarily result in a change of the interaction strength among genes. In this case, we will have pathways with significant altered mean expression levels but unaltered biological interactions. On the contrary, if transcripts abundances ratios are altered, we expect a significant alteration not only of their mean, but also of the strength of their connections, resulting in pathways with completely corrupted functionality. Therefore, to look for pathways strongly involved in a biological process, we should look at pathways with both mean and variance significantly altered. clipper is based on a two-step approach: 1) it selects pathways with covariance matrices and/or means significantly different between experimental conditions and 2) on such pathways, it identifies the sub-paths mostly associated to the phenotype. This is a very peculiar feature in pathway analysis. To our knowledge this is the first approach able to systematically inspect a pathway deep in its different portions.

1.2 Other approaches on Bioconductor

Currently there are some pathway analysis methods implemented in Bioconductor (probably the most famous is GSEA), but very few of them try to exploit pathway topology. Example of the latter category are SPIA and DEGraph.

In Martini et al. 2012 is provided a detailed comparison of the performance of non-topological analysis (GSEA) and topological analysis (SPIA and clipper) using both real and simulated data. In the next few words, we are going to highlight the differences of these approaches.

GSEA uses pathway as a list of genes without taking into account the structure of the pathway while SPIA takes into account pathway topological information, gene fold-changes and pathway enrichment scores. Then SPIA takes as input only the list of differentially expressed genes. So, from a practical point of view clipper and SPIA test different null hypotheses.

More importantly, clipper is able to highlight the portions (sub-path) of the pathway that are mostly involved in the phenotype under study using graph decomposition theory.

For more details please refer to Martini et al. 2012.
# 2 Performing pathway analysis

In this section, we describe how to perform the topological pathway analysis on a whole pathway. As an example we used the gene expression data published by Chiaretti et al. on acute lymphocytic leukemia (ALL) cells associated with known genotypic abnormalities in adult patients. Several distinct genetic mechanisms lead to ALL malignant transformations deriving from distinct lymphoid precursor cells that have been committed to either T-lineage or B-lineage differentiation. Chromosome translocations and molecular rearrangements are common events in B-lineage ALL and reflect distinct mechanisms of transformation. The BCR breakpoint cluster region and the c-abl oncogene 1 (BCR/ABL) gene rearrangement occurs in about 25% of cases in adult ALL.

The expression values (available through Bioconductor) deriving from Affymetrix single channel technology, consist of 37 observations from one experimental condition (class 2, n1 37, BCR; presence of BCR/ABL gene rearrangement) and 42 observations from another experimental condition (class 1, n2 41, NEG; absence of rearrangement). The *clipper* method is based on Gaussian graphical models, therefore it is strongly recommended to use log-transformed data.

In this example, we are going to evidence the differences between BCR/ABL (class 2) and NEG (class 1) though a topological pathway analysis.

We use the *graphite* bioconductor R package as a source of pathway topological information. In our test dataset, given the presence of the BCR/ABL chimera, we expect that all the pathways including BCR and/or ABL1 will be impacted. Here we retrieve, for example, the KEGG "Chronic myeloid leukemia" pathway.

```r
> library(graphite)
> kegg <- pathways("hsapiens", "kegg")
> graph <- convertIdentifiers(kegg["Chronic myeloid leukemia"], "entrez")
> graph <- pathwayGraph(graph)
> genes <- nodes(graph)
> head(genes)

[1] "10000" "1019" "1021" "1026" "1029" "1147"
```

Once the pathway (converted to a graphNEL object) is loaded in the workspace, we need to retrieve the expression matrix and the corresponding sample annotations (2 denoting samples with translocation and 1 denoting samples with no BCR/ABL translocation). We use as an example the ALL Bioconductor package.

```r
> library(ALL)
> data(ALL)

Fist of all, we should take a look at the phenoData.

```r
> head(pData(ALL))

<table>
<thead>
<tr>
<th>cod</th>
<th>diagnosis</th>
<th>sex</th>
<th>age</th>
<th>BT</th>
<th>remission</th>
<th>CR</th>
<th>date.cr</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>t(4;11)</th>
<th>t(9;22)</th>
<th>cyto.normal</th>
<th>citog mol.biol</th>
</tr>
</thead>
<tbody>
<tr>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>t(9;22) BCR/ABL</td>
</tr>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>simple alt.</td>
<td>NEG</td>
</tr>
<tr>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>&lt;NA&gt; BCR/ABL</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>t(4;11)</td>
<td>ALL1/AF4</td>
</tr>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>del(6q)</td>
<td>NEG</td>
</tr>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>complex alt.</td>
<td>NEG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>fusion protein</th>
<th>mdr</th>
<th>kinet</th>
<th>ccr</th>
<th>relapse</th>
<th>transplant</th>
</tr>
</thead>
<tbody>
<tr>
<td>p210</td>
<td>NEG</td>
<td>dyploid</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
</tr>
<tr>
<td>&lt;NA&gt;</td>
<td>POS</td>
<td>dyploid</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
This data frame summarized all the phenotypic features of the samples. In our analysis, we are interested in B-cell. This information is hosted in the column called ‘BT’.

> pData(ALL)$BT

```
[55] B3 B3 B4 B3 B3 B1 B1 B1 B3 B3 B3 B3 B3 B3 B3 B3 B3 B3
[91] B2 B1 B2 B B B T T3 T2 T2 T2 T4 T2 T3 T3 T T2
[109] T3 T2 T2 T2 T1 T4 T T2 T3 T2 T2 T2 T2 T3 T3 T T3 T3 T
[127] T2 T
```

Levels: B B1 B2 B3 B4 T T1 T2 T3 T4

> pAllB <- pData(ALL)[grep("B", pData(ALL)$BT),]

> dim(pAllB)

```
[1] 95 21
```

After this selection, we are interested in the isolation of sample with translocation from those without translocation. This information is hosted in the column ‘mol.biol’.

> pAllB$'mol.biol'

```
[1] BCR/ABL NEG BCR/ABL ALL1/AF4 NEG NEG
[7] NEG NEG NEG BCR/ABL BCR/ABL NEG
[13] E2A/PBX1 NEG BCR/ABL NEG BCR/ABL BCR/ABL
[19] BCR/ABL BCR/ABL NEG BCR/ABL BCR/ABL NEG
[25] ALL1/AF4 BCR/ABL ALL1/AF4 NEG ALL1/AF4 BCR/ABL
[31] NEG BCR/ABL NEG BCR/ABL BCR/ABL ALL1/AF4
[37] NEG BCR/ABL BCR/ABL BCR/ABL NEG E2A/PBX1
[43] BCR/ABL NEG NEG NEG BCR/ABL p15/p16
[49] ALL1/AF4 BCR/ABL BCR/ABL NEG E2A/PBX1 NEG
[55] NEG NEG BCR/ABL BCR/ABL NEG NEG
[61] ALL1/AF4 NEG ALL1/AF4 NEG BCR/ABL
[67] NEG NEG NEG NEG BCR/ABL ALL1/AF4
[73] BCR/ABL NEG E2A/PBX1 NEG BCR/ABL BCR/ABL
[79] NEG NEG NEG NEG BCR/ABL BCR/ABL
[85] BCR/ABL BCR/ABL BCR/ABL ALL1/AF4 NEG NEG
[91] BCR/ABL NEG BCR/ABL BCR/ABL E2A/PBX1
```

Levels: ALL1/AF4 BCR/ABL E2A/PBX1 NEG NUP-98 p15/p16
Now we have to build the vector of classes.

```r
classesUn <- as.character(pAll$'mol.biol')
classesUn[classesUn=='BCR/ABL'] <- 2
classesUn[classesUn=='NEG'] <- 1
names(classesUn) <- row.names(pAll)
classes <- sort(classesUn)
```

Now that we have the vector of classes, we can isolate the subset of sample from the original expression set and subsequently we convert affymetrix probe names into entrez gene ids.

```r
library("hgu95av2.db")
all <- ALL[,names(classes)]
probesIDS <- row.names(exprs(all))
featureNames(all@assayData)<-unlist(mget(probesIDS, hgu95av2ENTREZID))
all <- all[(!is.na(row.names(exprs(all))))]
```

At this point, we compute the intersection between pathway nodes and the genes for which an expression value is available. Thus we obtain a subgraph of the original graph. Moreover, we can extract from the expression set a smaller expression set corresponding to the expression of the genes in the pathway under investigation.

```r
library(graph)
genes <- intersect(genes, row.names(exprs(all)))
graph <- subGraph(genes, graph)
exp <- all[genes,,drop=FALSE]
```

ExpressionSet (storageMode: lockedEnvironment)

- assayData: 67 features, 79 samples
  - element names: exprs
- protocolData: none
- phenoData
  - sampleNames: 01010 04007 ... 84004 (79 total)
  - varLabels: cod diagnosis ... date last seen (21 total)
  - varMetadata: labelDescription
- featureData: none
- experimentData: use 'experimentData(object)'
  - pubMedIds: 14684422 16243790
- Annotation: hgu95av2

```r
dim(exprs(exp))
```

[1] 67 79

Note that the usage of 'exp' as ExpressionSet or exprs(exp) as expression matrix leads exactly at the same result. Here we will use the ExpressionSet but all our function can be used with a simple expression matrix.

The analysis is performed using `pathQ` function as follows:

```r
library(clipper)
pathwayAnalysis <- pathQ(exp, classes, graph, nperm=100, alphaV=0.05, b=100)
```

```r
pathwayAnalysis
```

```r
$alphaVar
```

[1] 0.21
$\text{alphaMean} \\
[1] 0 \\

The returned list contains the p-value for the test on the concentration matrices (alphaVar) and the p-value for the test on the means (alphaMean).

3 Performing clipper analysis

After a global inspection and identification of the most interesting/impacted pathways (the global analysis we have seen in the previous section), it is important to focus on the genes that drive the differences between the two phenotypes. The following example shows how to identify the sub-paths mostly associated to the phenotype.

```r
> clipped <- clipper(exp, classes, graph, "var", trZero=0.01, permute=FALSE)
```

The analyzed cliques are indexed by the maximum cardinality search (mcs) algorithm and identified hereafter with the index number. The result of the clipper analysis is the matrix described in the following. For each analyzed path (named as `<starting clique index>;<ending clique index>`) the following information are reported:

1. Index of the starting clique
2. Index of the ending clique
3. Index of the clique where the maximum value is reached
4. Length of the path
5. Maximum score of the path
6. Average score along the path
7. Percentage of path activation
8. Impact of the path on the entire pathway
9. Involved and significant cliques
10. Cliques forming the path
11. Genes forming the significant cliques
12. Genes forming the path

A deeper look at the clipper matrix reveals that many paths overlap. To help users in focusing on the best candidates, we devised a function to prune the paths that are already part of other ones. The pruning process is performed according to a dissimilarity threshold `thr` (if paths dissimilarity value is greater than `thr` they are retained).

```r
> clipped <- prunePaths(clipped, thr=0.2)
```

After the pruning, the results are smaller and much clear to read and interpret.
4 Visualizing clipper results.

WARNING: RCytoscape is no longer support in some Platform. clipper uses the RCytoscape package to connect to Cytoscape and display its results. Cytoscape is a Java based software specifically built to manage biological network complexity and for this reason it is widely used by the biological community.

> plotInCytoscape(graph, clipped[1,])

After the export of the pathway to Cytoscape, you can choose between different layouts. In figure 1 we show the KEGG “Chronic myeloid leukemia” pathway with an hierarchical layout and with the genes that belong to the most impacted path highlighted in blue.

5 Easy clipper analysis

The package provides also a function to easily run the analysis described in Martini et al. 2012. This analysis is able to start from a expression matrix and a pathway and returns the paths in the pathway that are altered between the two conditions.

> clipped <- easyClip(exp, classes, graph, method="mean")
> clipped[,1:5]

<table>
<thead>
<tr>
<th>startIdx</th>
<th>endIdx</th>
<th>maxIdx</th>
<th>length</th>
<th>maxScore</th>
</tr>
</thead>
<tbody>
<tr>
<td>1;17</td>
<td>1</td>
<td>17</td>
<td>9</td>
<td>51.0841685716396</td>
</tr>
<tr>
<td>1;13</td>
<td>1</td>
<td>13</td>
<td>3</td>
<td>12.7058667162116</td>
</tr>
</tbody>
</table>

A short summary of the results can be obtained with easyLook function.

> easyLook(clipped)

maxScore
1;17 51.0841685716396
6 Bibliography

References


