

Gene expression

Improved scoring of functional groups from gene expression data by decorrelating GO graph structure

Adrian Alexa*, Jörg Rahnenführer and Thomas Lengauer

Max-Planck-Institute for Informatics, Stuhlsatzenhausweg 85, D-66123 Saarbrücken, Germany

Received on September 28, 2005; revised on March 30, 2006; accepted on April 4, 2006

Advance Access publication April 10, 2006

Associate Editor: Martin Bishop

ABSTRACT

Motivation: The result of a typical microarray experiment is a long list of genes with corresponding expression measurements. This list is only the starting point for a meaningful biological interpretation. Modern methods identify relevant biological processes or functions from gene expression data by scoring the statistical significance of predefined functional gene groups, e.g. based on Gene Ontology (GO). We develop methods that increase the explanatory power of this approach by integrating knowledge about relationships between the GO terms into the calculation of the statistical significance.

Results: We present two novel algorithms that improve GO group scoring using the underlying GO graph topology. The algorithms are evaluated on real and simulated gene expression data. We show that both methods eliminate local dependencies between GO terms and point to relevant areas in the GO graph that remain undetected with state-of-the-art algorithms for scoring functional terms. A simulation study demonstrates that the new methods exhibit a higher level of detecting relevant biological terms than competing methods.

Availability: topgo.bioinf.mpi-inf.mpg.de

Contact: alexa@mpi-sb.mpg.de

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

The result of a microarray experiment is a list of genes with corresponding expression profiles. Such a gene list is the starting point for an investigation of the biology manifested by the experimental data. Often, genes are ranked according to differential expression between disease groups or according to correlation of expression values with a phenotype measurement. The result of such an analysis is an ordered list of genes.

In many cases the list of differentially expressed genes is not sufficient for accurate inference of the underlying biology. Additional biological knowledge needs to be included to enhance the interpretation of such a list of genes. With the development of biological knowledge databases (Ashburner *et al.*, 2000), information for augmenting gene expression data is available. Biologically interesting sets of genes, for example genes that belong to a pathway or genes known to have the same biological function, can now be compiled. A popular choice for gene sets are genes collected under Gene Ontology (GO) terms, see GO Consortium (2004).

Methods for gene set enrichment analyze the positions of genes with a common function in an ordered list of genes. The biological function is expected to be more relevant if the gene set members are among the top-ranked genes in the ordered list obtained in the primary stage of the analysis. The top k genes from the ordered list are selected as interesting genes, and enrichment refers to over-representation of these genes in the group of gene set members. The amount of over-representation is assessed with a statistical score.

Methods that test for enrichment of GO terms have been proposed by Draghici *et al.* (2003), Zeenerg *et al.* (2003), Al-Shhrour *et al.* (2004) and Beissbarth and Speed (2004). A comparative study of commonly used tools for analyzing GO term enrichment was recently presented by Khatri and Draghici (2005). None of the methods compared in this study integrates the knowledge encapsulated in the hierarchical structure of the GO database. Recently Grossmann *et al.* (2006) discussed scoring enrichment of GO terms in a local sense. The over-representation of a GO term is quantified with respect to its direct less specific neighbors in the GO hierarchy. In the present article we propose algorithms that identify over-represented terms in a more global sense, integrating the whole GO topology in the score.

Several other methods integrating the hierarchical structure of the GO have related but different goals. Balasubramanian *et al.* (2004) test the association between multiple sources of functional genomics data. Each data source is represented by a single graph nodes representing genes and edges representing functional links. For a group of genes a graph based on GO is obtained by placing edges between all gene pairs that are annotated to GO terms with a small distance in GO graph topology. The proposed tests statistically compare the occurrence of edges observed in different graphs. Joslyn *et al.* (2004) define distances between nodes in the GO graph for ordering GO terms with respect to a group of genes. A GO term is considered more important if many genes in the group are annotated to GO terms close in graph topology. The resulting rank-ordered list of GO terms is then clustered in order to identify summarizing nodes for the characteristics of the gene group.

The complex structure of the GO introduces strong dependencies among the GO terms. Figure 1 depicts significantly enriched GO terms in a microarray study. The color of the GO terms represents the relative significance of the terms. The interpretation of such results is hampered by the implicit dependence between neighboring GO terms. Even when multiple testing correction procedures are applied the results are biased due to this correlation. The interpretation of the results can be improved if the correlation between the neighboring GO terms is integrated into the score.

*To whom correspondence should be addressed.

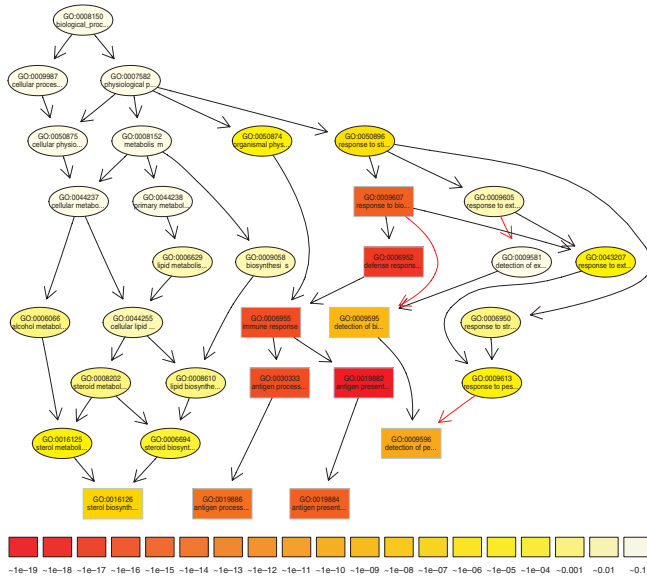


Fig. 1. The subgraph induced by the 10 most significant GO terms identified by a current state-of-the-art method for scoring GO terms for enrichment. Boxes indicate the 10 most significant terms. Box color represents significance, ranging from dark red (most significant) to light yellow (least significant). The numbers in the legend represent the magnitude of the raw p -values associated with the respective color. Black arrows indicate is-a relationships and red arrows part-of relationships.

In this article we present two novel methods that improve the enrichment analysis of GO terms by integrating GO graph topology on a global scale. The first method, called *elim*, is intuitive and simple to interpret. It iteratively removes the genes mapped to significant GO terms from more general (higher level) GO terms. In the second method, called *weight*, genes annotated to a GO term receive weights based on the scores of neighboring GO terms.

When analyzing the GO graph structure, local dependencies between GO terms can be identified and removed. The algorithms are evaluated on two publicly available datasets. However, an evaluation on real datasets always yields results biased towards specifically designed algorithms. To avoid a subjective assessment of the quality of the enrichment methods, we introduce a novel evaluation scheme in which a predefined number of GO terms are artificially enriched and the performance of the methods is quantified with respect to the number of correctly identified enriched terms. The results from both real and simulated data show that the proposed algorithms perform better than current state-of-the-art methods.

2 METHODS

The GO provides an ontology that describes the roles of genes and gene products in various organisms. GO has a hierarchical structure that forms a directed acyclic graph (DAG). For such a graph we can use the notions of child and parent, where a child can have multiple parents. Every GO term is represented by a node in this graph. The nodes are annotated with a set of genes. For an inner node of the GO graph, the corresponding set of genes also comprises all genes annotated to all children of this node.

Various test statistics are used for scoring significance of GO terms. Standard implementations of GO testing compute the significance of a node independent of the significance of the neighboring nodes. Independently

Table 1. Contingency table for the set of genes annotated to node u and the genes found significant in an expression study

	\mathcal{A}	$\bar{\mathcal{A}}$	Sum
\mathcal{B}	$ sigGenes \cap genes[u] $	$ \overline{sigGenes} \cap genes[u] $	$ genes[u] $
$\bar{\mathcal{B}}$	$ sigGenes \cap genes[\bar{u}] $	$ \overline{sigGenes} \cap genes[\bar{u}] $	$ genes[\bar{u}] $
Sum	$ sigGenes $	$ \overline{sigGenes} $	$ allGenes $

of the test statistic used, this standard scoring is referred as the *classic* method in this article. Two novel approaches as alternatives for this basic method are introduced further.

If a GO term contains the same genes as one of its children the *classic* method gives the same score to both terms. In such a case the child is biologically more interesting since its associated definition is more specific than the definition of its ancestors. Thus, a promising idea is to compute the significance of a node dependent on the significance of its children. The *elim* method directly implements this idea by removing all genes that are annotated to a significantly enriched node from all its ancestors. Removing genes from a GO term can be regarded as a weighting scheme where weights are restricted to be either 0 or 1. The *weight* algorithm generalizes this idea to weights in the interval $[0,1]$. In order to decide if a GO term u better represents the interesting genes than any other term from its neighborhood, the enrichment score of node u is compared with the scores of its children. Children with a better score than u better represent the interesting genes. The corresponding genes annotated to these children should contribute less to the score of any ancestor of node u . Thus, these genes are assigned small weights in all ancestors of node u . The children with a lower score than u should not be reported as significant. To achieve this, the genes annotated to these children receive small weights, and the score is recomputed based on the newly assigned weights.

Before describing the algorithms in more detail, we introduce some definitions and notations.

2.1 Definitions and notations

Let $genes[u]$ denote the set of genes annotated to a GO term u . When the union of two or more sets of genes is computed, duplicates are removed. With $\overline{genes[u]}$ we denote all genes that are not annotated to node u .

The edges of the GO graph are directed from parent to child. The *level* of a node u is defined as the length of the longest path from the *root* to node u . Note that there can be leaves at each level (except level 0) and that nodes at the same level never have an edge between them. For a set \mathcal{U} of nodes, $upperInducedGraph(\mathcal{U})$ is defined to be the subgraph induced by all nodes reachable from \mathcal{U} if all edges in the graph are reversed.

To test the enrichment of a node u we use the degree of independence between the two properties:

- \mathcal{A} : gene is in the list of significant genes,
- \mathcal{B} : gene is member of GO term u .

Let $sigGenes$ denote the set of interesting genes in a microarray experiment, e.g. the list of differentially expressed genes. Analogously, $\overline{sigGenes}$ denotes all other genes from the microarray. For a node u the contingency table (Table 1) summarizes the counts of genes according to their membership to $genes[u]$ and $sigGenes$.

Testing the association between the characteristics \mathcal{A} and \mathcal{B} for the contingency table (Table 1) corresponds to Fisher's exact test, see Lehmann (1986). The p -value returned by this test is the probability of observing at least the same amount of enrichment when significant genes are randomly selected out of all genes. Thus, a very small p -value gives strong evidence for an association between \mathcal{A} and \mathcal{B} .

Algorithm 1 elim

```

markedGenes ← ∅; nodeSig ← ∅
get the DAG levels list dagLevels
for i from max(dagLevels) to 1
  for u in nodes(dagLevels, i)
    genes[u] ← genes[u] \ markedGenes[u]
    nodeSig[u] ← FisherTest(genes[u], sigGenes)
    if nodeSig[u] ≤ threshold then
      for x in upperInducedGraph(u)
        markedGenes[x] ← markedGenes[x] ∪ genes[u]
    end
  end
end
return nodeSig

```

2.2 First approach: eliminating genes

The *elim* method investigates the nodes in the GO graph bottom-up. It starts processing the nodes from the highest (bottommost) level and then iteratively moves to nodes from a lower level. Since nodes from the same level share no edge, they can be investigated independently. The bottom-up strategy assures that for a currently investigated node all children have been scored.

When a node *u* is processed, the genes that have been marked as removed in a previous step are removed from the set *genes*[*u*]. The score for the resulting group of genes is the *p*-value returned by Fisher's exact test, and node *u* is marked as significant if the *p*-value is smaller than a previously defined threshold. Typically this threshold is set to be 0.01 divided by the number of nodes in the GO graph with at least one annotated gene. This corresponds to a Bonferroni adjustment of the *p*-values. If node *u* is found significant then all genes mapped to it are marked as removed in all nodes of *upperInducedGraph*(*u*), that is in all ancestors of node *u*.

The algorithm finishes when all nodes have been processed. The *elim* method is summarized in Algorithm 1.

2.3 Second approach: weighting genes

The purpose of the bottom-up strategy implemented in the *elim* algorithm is to identify the most specific nodes with minimum required significance. A node is considered to be significant if its *p*-value is below a given threshold. More significant nodes on higher levels in the graph thus can be missed because of the gene removal process.

An alternative is implemented in the *weight* method. Here, significance scores of connected nodes (a parent and its child) are compared in order to detect the locally most significant terms in the GO graph. This is achieved by down-weighting genes in less significant neighbors. We first describe how weights are assigned to genes and how the significance score of a group of genes with corresponding weights is computed.

Let *u* be a node in the graph and let *v* be one of its children. The weight associated with this pair of nodes is defined by

$$w = \text{sigRatio}(\text{score}(v), \text{score}(u)).$$

The function *sigRatio*(*a*, *b*) has the form

$$\text{sigRatio}(a, b) = \frac{f(a)}{f(b)},$$

where *f*(*·*) is an arbitrary increasing function that can be adapted to the desired degree of weighting between pairs of nodes. Note that the function *sigRatio* always attains a value in the interval [0, 1] if both scores are non-negative and if the child node *v* is less significant than its parent *u*.

The score for a weighted set of genes is computed by applying Fisher's exact test on a weighted contingency table. $X = |\text{sigGenes} \cap \text{genes}[u]|$ is the number of significant genes that are annotated to node *u* (Table 1). This

quantity is replaced by summing up the weights of the genes and subsequent rounding up to the next integer:

$$X = \left\lceil \sum_{i \in \{\text{sigGenes} \cap \text{genes}[u]\}} \text{weight}[i] \right\rceil. \quad (1)$$

In the same way, all other quantities in Table 1 are computed. The cardinality function $|S|$ for a set *S* is always replaced by the function $\lceil \sum_{i \in S} \text{weight}[i] \rceil$. The idea behind using a weighted contingency table is to first decorrelate the GO terms and then to test for enrichment in the transformed set of genes.

In the *weight* method the nodes are processed bottom-up level by level as in the *elim* method. However, for each node a vector of gene weights is memorized and updated during the process. Initially, all weights for the genes annotated to a node are set to 1. Let *u* be the currently processed node in the bottom-up process. The main principle is to reinforce differences in significance between *u* and its neighbors. If *u* is more significant, genes contained in the children are down-weighted, yielding a decreased significance of the children. If at least one child is more significant than *u*, the genes common to the child and *u* are down-weighted in node *u* and its ancestors, further decreasing the significance of node *u*.

The function *computeTermSig*(*u*, *children*) is the core of the *weight* algorithm. It recursively eliminates children of *u* that are more significant than *u*. Each time this function is called, the score of *u* is recomputed using the updated genes weights. Based on the new score, in turn, the weights for all children are recomputed using the *sigRatio*() function.

More precisely, if in the iterative process node *u* has a lower *p*-value than its children and thus can be regarded as a better candidate, for each child the old weights (assigned in some previous step) are multiplied with the weights given by the *sigRatio*() function. After updating the scores for all children the processing of node *u* finishes.

The alternative is that at least one child has a better score than node *u*. All genes annotated to each of these more relevant children are down-weighted in the ancestors of node *u*, including *u* itself. Again, gene weights are multiplied with a factor obtained with the *sigRatio*() function. By calling *computeTermSig*() for node *u* and all its less significant children, the score for *u* then is recomputed based on the new weights. Owing to the modified weights for node *u*, the scores of current children can now become more significant than the updated score of *u*. Thus the process is iterated until all remaining children have lower scores than *u*. Then the algorithm moves to the next node.

The full details of the algorithm *weight* are shown in Algorithm 2.

2.4 Combining algorithms

In addition to the presented algorithms and the *classic* method, we consider a combined algorithm called all.M. For this algorithm, the reported score for a node is defined as the mean of the *p*-values obtained with *classic*, *elim* and *weight* algorithm, computed on a log scale. If *p*-Value denotes the vector of the three *p*-values for a GO term, then the new score is defined by

$$\bar{p} = \exp \left(\frac{1}{3} \sum_{i=1}^3 \log(\text{pValue}[i]) \right).$$

3 RESULTS

We evaluated the GO scoring algorithms *classic*, *elim* and *weight* on two real gene expression datasets (Chiaretti et al., 2004; Cario et al., 2005) and on simulated data. Both real datasets were collected in order to distinguish subgroups of Leukemia patients. However, the discrimination criterion is quite different. In the first case, patients are split into B-cell and T-cell type leukemias and in the second case into children with or without minimal residual disease (MRD) after therapy.

The algorithms were implemented in the R programming language (www.r-project.org). The results were obtained using

Table 2. Statistics for significant GO terms for the ALL dataset (Chiaretti et al., 2004).

GO ID	Term	Observed	Expected	Annotated	p-values				
					classic	elim	weight.log	weight.ratio	all.M
1	GO:0019882 antigen presentation	22	2.287	41	1.6e-17	0.2821	1.6e-17	1.6e-17	1.8e-13
2	GO:0006952 defense response	107	47.143	845	8.3e-17	0.0065	1.4e-09	1.1e-06	5.4e-09
3	GO:0030333 antigen processing	20	2.12	38	7.8e-16	1.0000	7.8e-16	7.8e-16	4.7e-12
4	GO:0006955 immune response	98	43.293	776	2.7e-15	5.9e-06	3.0e-05	0.024	3.3e-07
5	GO:0019884 antigen presentation, exogenous ...	14	1.004	18	5.9e-15	5.9e-15	2.2e-10	0.054	1.4e-10
6	GO:0009607 response to biotic stimulus	112	53.949	967	9.5e-15	0.6873	1.0e-05	0.404	1.3e-05
7	GO:0019886 antigen processing, exogenous ...	14	1.116	20	6.8e-14	6.8e-14	1.5e-11	0.054	2.5e-10
8	GO:0009596 detection of pest, pathogen or ...	9	0.725	13	2.9e-09	2.9e-09	2.9e-09	2.9e-09	2.9e-09
9	GO:0009595 detection of biotic stimulus	9	0.893	16	3.9e-08	1.0000	1.0e-05	0.107	0.00046
10	GO:0016126 sterol biosynthesis	9	1.395	25	4.5e-06	0.0015	4.5e-06	4.5e-06	1.9e-05

The column Expected represents the expected number of interesting genes mapped to the GO term if the interesting genes were randomly distributed over all GO terms.

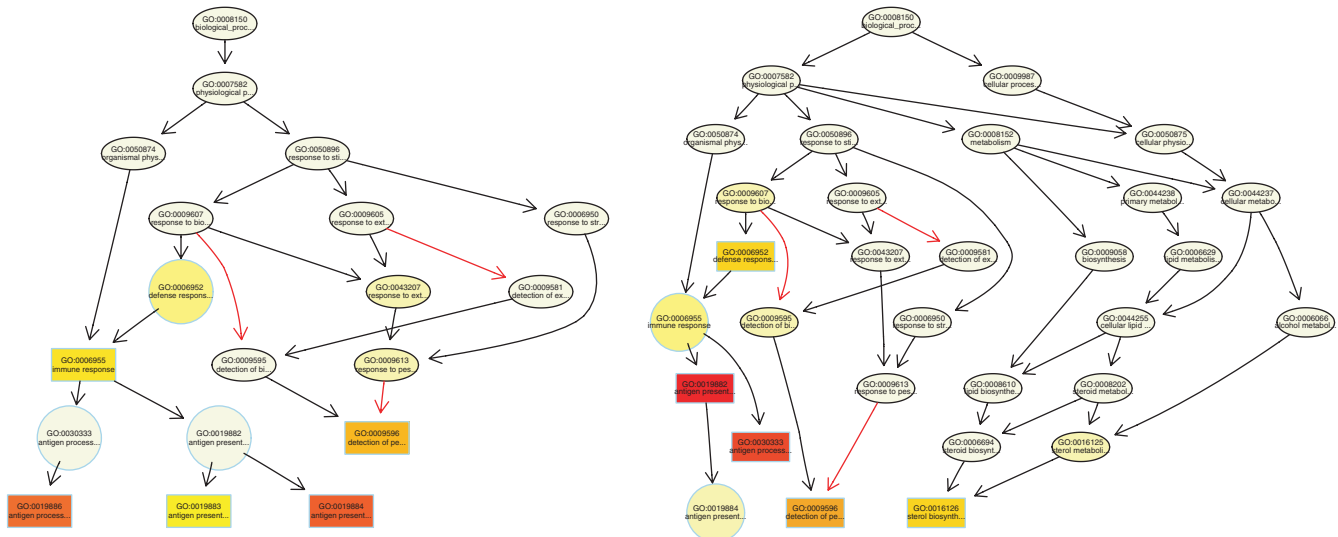


Fig. 2. The subgraph induced by the five most significant GO terms found by the *elim* method (left graph) and by the *weight.ratio* method (right graph). For a detailed description of the figures see the legend of Figure 1.

detectable minimal residual disease (MRD-SR) and 21 patients with high MRD load (MRD-HR).

The data were imported and processed in R. All probes on the array with more than missing values for the 51 samples were removed from the analysis. The expression values of multiple probes with the same LocusLink identifier coding for one respective gene were averaged. This preprocessing resulted in 13236 genes. Only 6853 of these genes are annotated to at least one GO term from the BP ontology. The induced GO graph contains 2733 nodes. Compared with the first Leukemia study discussed above the genes are annotated to more specific GO terms and they are more widely spread across the hierarchy. Applying a two-sided *t*-test to the groups defined by patients with MRD-SR and MRD-HR respectively, a total of 682 genes were detected as differentially expressed. *p*-values were adjusted with the FDR procedure (Benjamini and Yekutieli, 2001).

The induced GO subgraph plots for all methods and a results table in the form of Table 2 are available in the Supplementary Material.

A comparison of the results for the two studies shows important differences but also remarkable similarities. Although both studies are related, they address different biological problems and the data are obtained with different experimental technologies. The GO term immune response is found significant for the first dataset but not for the second and cell adhesion is specific only to the second dataset. Further interesting GO terms for the second study are peripheral nervous system development and activation of MAPK activity. These processes should be investigated for obtaining more knowledge on MRD.

On the other hand, in both studies the GO terms GO:0019884 and GO:0019886 consistently are among the most significant terms for all algorithms, thus underlining the general importance of antigen presentation and antigen processing for ALL, Table 2. Similar concordances between the two Leukemia datasets are described in the results section of the article of Cario et al. (2005).

It is common to both studies that very general GO terms are discarded by the more sophisticated algorithms, in the second

Table 3. Average numbers of correctly identified *enriched nodes* over 100 simulation runs with 50 true *enriched nodes*, 10% noise level and 10 – 50 genes annotated to the *enriched nodes*, for different values of k

k	class	weight.log	weight.ratio	elim	all.M
25	5.5	13	14.2	16.9	15.4
50	14.5	25.5	28	27	28.5
75	22.5	36	38	31	38
100	31	42	39.5	33.5	43

study specifically the originally high-scoring terms organogenesis and development.

3.3 Comparison of algorithms on simulated data

In order to objectively compare the quality of different GO scoring methods we need to know the true relevant GO terms. On real datasets like the ones introduced above the true significant GO terms are not known. To address this issue the algorithms were also tested on simulated data. In our simulation setup the significant GO terms are known and the performance of an algorithm is measured w.r.t. the number of correctly identified GO terms. To mimic the real data as best possible the GO biological process ontology was chosen as the underlying graph. The complete list of genes contains all 9231 probes on the HGU95aV2 chip that can be annotated to this graph. The resulting graph contains 2677 nodes.

In the first step, a set of truly enriched nodes is determined. These nodes represent the relevant biological functions and are denoted as *enriched nodes*. We select the *enriched nodes* randomly from all nodes whose number of annotated genes lies in a fixed pre-specified interval. Such a constraint is imposed in order to overcome two problems. Nodes with too few annotated genes are too specific and cannot synthesize the underlying biology, and nodes with too many annotated genes are too general and close to the root of the graph.

In the second step, given the *enriched nodes*, the list of interesting genes is obtained by combining all genes annotated to these nodes to a single set. To better model reality, some noise is introduced in the list of interesting genes. A fraction of the interesting genes is randomly selected and replaced with other genes.

The performance of the methods is assessed with the following measure. For each method \mathcal{M} , the nodes are sorted in ascending order of their computed p -values. For a fixed cutoff k the score is the number of *enriched nodes* found among the top k nodes:

$$score_k^0(\mathcal{M}) = |top_k(\mathcal{M}) \cap enriched|.$$

$top_k(\mathcal{M})$ denotes the set of the k top-scoring nodes for method \mathcal{M} , and *enriched* denotes the set of *enriched nodes*. Methods that obtain a higher score better retrieve the true relevant nodes.

In the first simulation study 50 *enriched nodes* are selected, each having between 10 and 50 annotated genes. The noise introduced in the list of interesting genes is set to 10%, and results of 100 simulated datasets are averaged. Table 3 shows that all new methods perform better than the classic method. The *weight* methods always outperform the classic method by a considerable amount. For small values of k the *elim* method is the best, outperforming the classic method by a factor of 3. From the top 25 nodes selected by *elim* an

average of 17 nodes are *enriched nodes*, compared with an average of only 5.5 *enriched nodes* for the classic method. Combining the p -values of all methods also helps, providing robust results.

For real datasets the list of interesting genes contains much more noise than 10%. In a second simulation study with more stringent conditions we selected 50 *enriched nodes* from all nodes with 10–1000 annotated genes, and the noise level was set to 40%. With these parameters recovering the *enriched nodes* is much more difficult. As in the previous simulation study, the methods *weight* and *elim* clearly outperform the method *classic*, see the Supplementary Material, Table 5.

To obtain more insight into how each method accounts for the topology of the graph, the following scores are defined:

$$score_k^4(\mathcal{M}) = |level_k^1(\mathcal{M}) \cap enriched|, \pm$$

$$score_k^{1p}(\mathcal{M}) = |level_k^{1p}(\mathcal{M}) \cap enriched|,$$

$$level_k^1 = top_k(\mathcal{M}) \cup par(top_k(\mathcal{M})) \cup ch(top_k(\mathcal{M})),$$

$$level_k^{1p} = top_k(\mathcal{M}) \cup par(top_k(\mathcal{M})).$$

The score $score_k^1$ considers the first k nodes together with all children and parents of these nodes. $score_k^{1p}$ only parents and not children are included. Analogously, we define $score_k^2(\mathcal{M})$ and $score_k^{2p}(\mathcal{M})$ that also include nodes with a distance of two levels from the first k nodes.

We compute such scores based on the observation that some methods, in particular the *elim* method, often yield more specific significant GO terms. If for a method \mathcal{M} , the difference $score_k^{1p}(\mathcal{M}) - score_k^0(\mathcal{M})$ is large, then many true *enriched nodes* are among the parents of the detected nodes.

Figure 3 shows plots of $score_k$ as a function of k . In Figure 3a and 3b $score_k^0$ and $score_k^{1p}$ are plotted for the simulation study with 10% noise. In Figure 3c $score_k^0$ is plotted for the simulation with 40% noise. Adding parents leads to no improvement for the classic method, but to a 15–20% improvement for the *elim* method. The best performing methods are *weight.ratio* and the combination of all methods all.M for $score_k^0$ and *elim* for the $score_k^{1p}$.

The results obtained with the *elim* method show that including the children of the top scoring nodes gives no significant improvement (40 versus 41 *enriched nodes* for $k = 50$, see Supplementary Material, Table 4, for a detailed results table). Including only parents ($score_k^{1p}$) improves the results for the *elim* method. These observations show that for *elim* the true *enriched nodes* are among the top scoring nodes and their parents. Thus *elim* successfully identifies the areas in the graph with the *enriched nodes*, but sometimes fails to precisely point to the correct nodes.

3.4 Robustness of algorithms

The analysis of high throughput microarray experiments adopted in this article is sometimes referred to as two-stage analysis. In the first step genes receive scores, e.g. quantifying the correlation of a phenotype with the gene expression values. Based on the distribution of the scores a threshold is chosen and genes with a score above the threshold are called differentially expressed genes. In the second step, the enrichment of a set of genes is tested based on test statistics that depend on the list of differentially expressed genes.

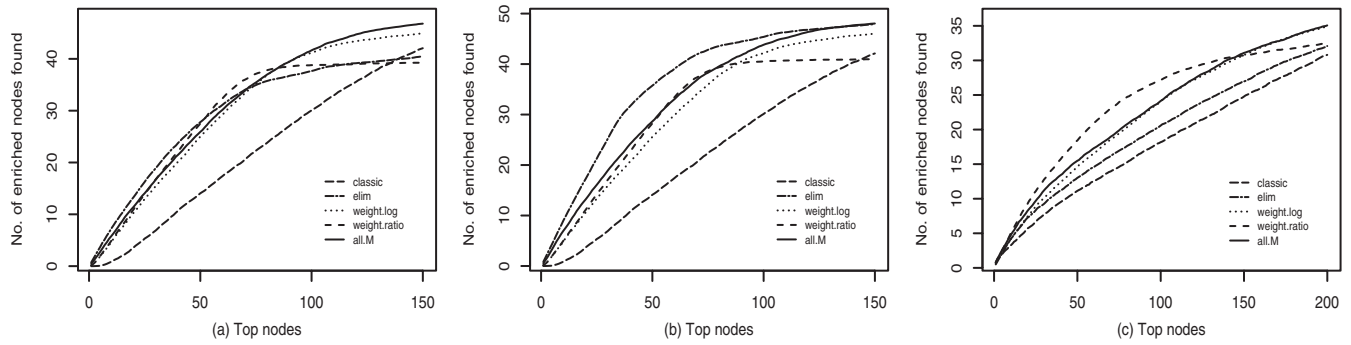


Fig. 3. Comparison of the quality of different GO scoring algorithms in a simulation study. Each curve represents the average of the numbers of preselected GO terms, over 100 simulation runs, that are among the top k GO terms according to the different algorithms. The proposed methods clearly outperform the method classic. In (a) and (b) $score_k^0$ and $score_k^p$ are plotted, respectively, with 10% noise introduced in the list of interesting genes, in (c) $score_k^e$ is plotted, with 40% noise.

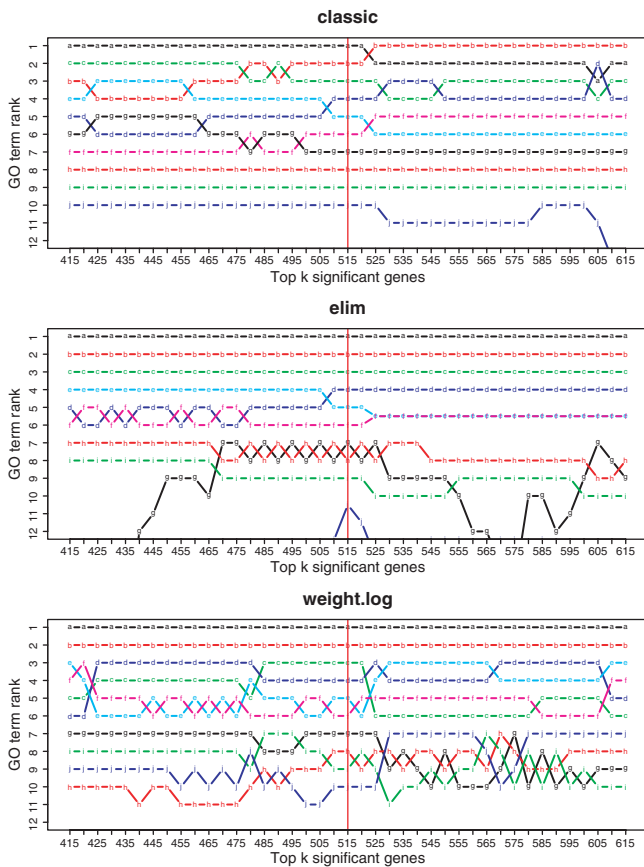


Fig. 4. Analysis of the influence of the threshold on the ranking of GO terms. Including all genes with FDR-adjusted p -value $p \leq 0.01$ yields $k = 515$ differentially expressed genes. Changing the threshold according to an increase or decrease of numbers of genes in steps of size 10 the ranks of the originally 10 top-ranking GO terms are shown. A colored line describes the rank of a fixed GO term for different thresholds.

It is believed that in many real-life cases the list of differentially expressed genes contains only a small fraction of truly differentially expressed genes. Thus, the result of the enrichment analysis can be hampered by the choice of the threshold. One method addressing this problems has been introduced in Subramanian *et al.* (2005).

To evaluate the influence of the threshold, all methods were run with different threshold values for the leukemia dataset discussed in Section 3.1. The results summarized in Figure 4 demonstrate the stability of the methods. Modifying the threshold value, and thus considering more or fewer differentially expressed genes, does not significantly change the order of the most significant GO terms. For the *elim* method adding more genes to the list of differentially expressed genes does not affect the top five GO terms. The *weight* method is the best in preserving the ordering; only few swaps between the GO terms are observed.

4 CONCLUSIONS

We addressed the problem of finding enriched functional groups of genes based on gene expression data. We proposed two novel heuristics for integrating the DAG structure of the GO in testing for group enrichment. The key idea is to compute the significance of a GO term based on its neighborhood. The experimental results show that the introduced methods improve over existing methods.

There is no clear measure to tell which method performs better on a real dataset. The evaluation is performed by the researcher and thus is inherently subjective. To eliminate this subjectivity, we evaluate the methods on simulated data.

With the classical approach in which each node is scored independently only few true significant nodes remain undiscovered. However, the dependencies between top scoring nodes yield a high false-positive rate. The simulation results show that the *weight* algorithm manages to reduce the false-positive rate, while not missing many true enriched nodes. The *elim* method further reduces the false-positive rate, but with a higher risk of discarding relevant nodes. For finding the important areas in the graph, the *elim* method is to be preferred, given its simplicity.

Other test statistics for assessing GO term significance than Fisher's exact test can be used, e.g. a hypergeometric, binomial or χ^2 -test. In particular, test statistics that do not rely on a fixed list of genes, such as the normalized Kolmogorov–Smirnov test introduced in Subramanian *et al.* (2005), can be implemented in the method *elim*. In this article we restricted ourselves to a single test statistic in order to better emphasize the differences between the different algorithms.

In conclusion we showed that integrating the dependency structure between nodes is enhancing the inference. Moreover, combining

the results of multiple methods even further improves the result of the analysis.

ACKNOWLEDGEMENTS

Financial support was provided by BMBF grant No. 01GR0453 [AA, JR] and by the IMPRS [AA]. This work has been performed in the context of the BioSapiens Network of Excellence (EU contract no. LSHG-CT-2003-503265).

Conflict of Interest: none declared.

REFERENCES

- Al-Shahrour, F. *et al.* (2004) FatiGO: a web tool for finding significant associations of Gene Ontology terms with groups of genes. *Bioinformatics*, **20**, 578–580.
- Ashburner, M. *et al.* (2000) Gene ontology: tool for the unification of biology. *The Gene Ontology Consortium. Nat. Genet.*, **25**, 25–29.
- Balasubramanian, R. *et al.* (2004) A graph-theoretic approach to testing associations between disparate sources of functional genomics data. *Bioinformatics*, **20**, 3353–3362.
- Beissbarth, T. and Speed, T.P. (2004) Gostat: find statistically overrepresented Gene Ontologies within a group of genes. *Bioinformatics*, **20**, 1464–1465.
- Benjamini, Y. and Yekutieli, D. (2001) The control of the false discovery rate in multiple testing under dependency. *Ann. Stat.*, **29**, 1165–1188.
- Cario, G. *et al.* (2005) Distinct gene expression profiles determine molecular treatment response in childhood acute lymphoblastic leukemia. *Blood*, **105**, 821–826.
- Chiaretti, S. *et al.* (2004) Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. *Blood*, **103**, 2771–2778.
- Draghici, S. *et al.* (2003) Global functional profiling of gene expression. *Genomics*, **81**, 98–104.
- GO Consortium (2004), The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.*, **32**, D258–D261.
- Grossmann, S., Bauer, S., Robinson, P.N. and Vingron, M. (2006) An improved statistic for detecting over-represented Gene Ontology annotations in gene sets. In *Proceedings of the Lecture Notes in Computer Science 3909*, March 2006 pp. 85–98.
- Joslyn, C.A. *et al.* (2004) The gene ontology categorizer. *Bioinformatics*, **20** (Suppl. 1) i169–i177.
- Khatri, P. and Draghici, S. (2005) Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, **21**, 3587–3595.
- Lehmann, E.L. (1986) *Testing Statistical Hypotheses. Springer Texts in Statistics*. 2nd edn. Springer-Verlag, New York.
- Subramanian, A. *et al.* (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl Acad. Sci. USA*, **102**, 15545–15550.
- Zeeberg, B.R. *et al.* (2003) GoMiner: a resource for biological interpretation of genomic and proteomic data. *Genome Biol.*, **4**, R28.