Model-based gene set analysis for Bioconductor

Sebastian Bauer1, Peter N. Robinson1,2,3,∗ and Julien Gagneur4, ∗

1 Institute for Medical Genetics, Charité–Universitätsmedizin Berlin, Augustenburger Platz 1, 13353 Berlin, 2 Max Planck Institute for Molecular Genetics, Ihnestrasse 73, 14195 Berlin, 3 Berlin-Brandenburg Center for Regenerative Therapies (BCRT), Charité–Universitätsmedizin Berlin, Berlin and 4 European Molecular Biology Laboratory, Meyerhofstrasse 1, 69117 Heidelberg, Germany

ABSTRACT

Summary: Gene Ontology and other forms of gene-category analysis play a major role in the evaluation of high-throughput experiments in molecular biology. Single-category enrichment analysis procedures such as Fisher’s exact test tend to flag large numbers of redundant categories as significant, which can complicate interpretation. We have recently developed an approach called model-based gene set analysis (MGSA), that substantially reduces the number of redundant categories returned by the gene-category analysis. In this work, we present the Bioconductor package mgsa, which makes the MGSA algorithm available to users of the R language. Our package provides a simple and flexible application programming interface for applying the approach.

Availability: The mgsa package has been made available as part of Bioconductor 2.8. It is released under the conditions of the Artistic license 2.0.

Contact: peter.robinson@charite.de; julien.gagneur@embl.de

Received on March 29, 2011; revised on May 5, 2011; accepted on May 6, 2011

1 INTRODUCTION

Gene Ontology (GO) analysis and other forms of gene-set enrichment analysis have become a standard exploratory tool for understanding the results of large-scale genomics experiments and for generating new hypotheses (Robinson and Bauer, 2011). Most early approaches investigated each GO term one at a time, for example testing for significant enrichment of responder genes in each term using Fisher’s exact test. In contrast, two recent methods, GenGO and model-based gene set analysis (MGSA) have been developed as global approaches, in which one aims to find the best combination of GO terms to explain the observed biological response (Bauer et al., 2010; Ott et al., 2011). In our original work, we integrated a first implementation of MGSA into the Ontologizer application, which is a tool for GO analysis and allows user to inspect the results in an interactive environment (Bauer et al., 2008). Here, we present an implementation of MGSA for users of Bioconductor (Gentleman et al., 2004). The mgsa package wraps a fast C-based implementation of the MGSA algorithm into a flexible application programming interface (API) and utilizes OpenMP to take advantage of the multicore processing units that modern computer hardware offers (Dagum and Menon, 1998).

2 AVAILABILITY AND USAGE

The mgsa package is part of Bioconductor 2.8, therefore it can be installed directly within the R environment together with all its dependencies. Refer to the Bioconductor Web page at http://www.bioconductor.org/ for installation procedures.

Once the package is installed and loaded, the method can be readily accessed using the function mgsa. To invoke the function, one needs to specify the observations, a vector of gene identifiers corresponding to the study set (e.g. the set of differentially expressed genes), and the gene sets, a list of vectors of gene identifiers for each of the GO terms (or other gene sets or categories) to be analyzed. To simplify the usage of GO, the readGAF function takes a GAF (Gene Annotation Format) file as input, in which gene annotations are stored, and computes the gene sets of all GO categories including direct and indirect annotations. GAF files are available from the GO homepage and updated regularly. The function takes advantage of the GO.db package to load the structure of the GO, so no external file is needed for the ontology itself. If goa.filename contains the location of a GAF file, observations is a vector of character strings describing the genes of the study set, then an MGSA analysis is as simple as entering the following R code.

```R
library(mgsa)
mapping <- readGAF(goa.filename)
results <- mgsa(observations,mapping)
plot(results)
```

∗ To whom correspondence should be addressed.

© The Author(s) 2011. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/2.5), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
A detailed tutorial is provided in the package vignette that can be invoked with:

```
 vignette("mgsa")
```

### 3 APPLICATION

The MGSA package is not restricted to the GO but allows analysis with arbitrary gene sets. This flexibility is illustrated on a dataset in which gene expression for two yeast strains that differ by a single allele (PHO84; Gagneur et al., 2009) is compared. We ask which transcription factor(s) could together best explain the set of 84 transcripts that show differential expression. We stored these as vector of gene identifiers, observations:

```
"YBR050C" "YBR054W" "YBR093C" ...
```

MacIsaac et al. (2006) have compiled a regulatory network for yeast by integrating data of *in vivo* transcription factor binding from ChIP/chip together with transcription factor motif analysis and sequence conservation. We defined as gene sets the sets of targets of each transcription factor of the network with intermediate cutoffs for binding intensities and conservation (MacIsaac et al., 2006). This network contains a total of 2514 targets across 116 transcription factors. We simply stored it as a named list of vectors of gene identifiers, *sets*:

```
$ABF1
[1] "YPL242C" "YPL159C" "YPL012W" ...

$ACE2
[1] "YOR140W" "YOR138C" "YNL327W" ...
```

For instance, the first item of the list contains a vector of genes that are targets of the transcription factor *ABF1* as predicted by MacIsaac et al. (2006). We can now call the *mgsa* method and plot results:

```
results <- mgsa(observations, sets)
plot(results)
```

The plot displays the marginal probabilities of the 10 most likely sets (Fig. 1). MGSA infers changes in activity for the transcription factor (posterior = 0.9995 ± 2 × 10⁻⁴). Allele variation in the transporter *PHO84* affects cellular phosphate levels and regulation of the whole PHO pathway (Gagneur et al., 2009). These transcriptional changes are known to be mediated by the transcription factor *PHO4*, which MGSA precisely identified.

### 4 CONCLUSION

The *mgsa* package gives users of Bioconductor programmatic access to MGSA. Thus, it can be incorporated into scripts and pipelines written in R and be combined with many other packages of the bioinformatics community. The package comes with a simple but flexible API, which allows researchers not only to use GO as

![Fig. 1. Transcription factor target set enrichment. The posterior probability is shown for the 10 transcription factors with highest marginal probabilities. Categories whose posterior is above 0.5 are interpreted as ‘active’ according to the MGSA model (Bauer et al., 2010).](image)

**REFERENCES**


Conflict of Interest: none declared.