The *mgsa* package

Sebastian Bauer, Julien Gagneur

28 April 2011

1 Introduction

Model-based Gene Set Analysis (MGSA, Bauer et al. [1]) is a Bayesian modeling approach for gene set enrichment. The package *mgsa* implements MGSA and tools to use MGSA together with the Gene Ontology [2].

MGSA takes as input *observations* (such as differentially expressed genes) and a list of gene *sets* (for example pathways or annotated terms of the gene ontologies). The model assumes that some sets to be inferred are *active* and that all genes member of active sets are themselves active. Active genes are more likely to belong to the observations and inactive genes not. Fitting the model amounts to infer the probability of every set to be active given the observations.

This procedure provides a useful alternative to classical gene set enrichment analysis [1]. Classical methods analyze each set in isolation. Because sets such as biological pathways often share genes with each other, the returned list of enriched sets is usually long and redundant. In contrast, MGSA takes set overlap into account by working on all sets simultaneously and substantially reduces the number of redundant sets.

The model have three parameters:

- α, the false positive rate i.e. the probability of an inactive gene to actually be observed;
- β, the false negative rate i.e. the probability of an active gene to actually be not observed;
- p, the prior probability for any set to be active, a typically small number ensuring sparse solutions to be inferred.

MGSA is Bayesian about these parameters too. Relatively uninformative priors are specified for them and the algorithm performs inference on the parameters as well.

Technical details about the algorithm and benchmarks of the method are given in [1].

2 Quick start

We start with a small simulated dataset which contains `example_go`, a random subset of yeast gene ontology annotations with 20 terms and `example_o`, a simulated set of observed genes. These genes could for example be the "hits"

of some screen or a set of differentially expressed genes. In the simulation, the terms GO:0006109 and GO:0030663 were active, implying that genes annotated to these terms were more likely to be observed positives than other genes.

```
> library(mgsa)
> data("example")
> example_go
```

```
Object of class MgsaSets
10 sets over 158 unique items.

Set annotations:
                                          term                         definition
GO:0046292    formaldehyde metabolic process The chemical reactions and pat...
GO:0006109 regulation of carbohydrate met... Any process that modulates the...
GO:0008113 peptide-methionine-(S)-S-oxide... Catalysis of the reactions: pe...
GO:0016849 phosphorus-oxygen lyase activi... Catalysis of the cleavage of a...
GO:0046527      glucosyltransferase activity Catalysis of the transfer of a...
... and  5  other sets.

Item annotations:
                                     name
SFA1    Bifunctional enzyme containing...
YJL068C Non-essential intracellular es...
ADR1    Carbon source-responsive zinc-...
CAT8    Zinc cluster transcriptional a...
FYV10   Protein of unknown function, r...
... and  153  other items.
```

```
> example_o
```

```
 [1] "SFA1"    "ADR1"    "CAT8"    "FYV10"   "GCR1"    "GCR2"    "GID7"
 [8] "HAP2"    "HAP3"    "HAP4"    "HAP5"    "PCL10"   "PCL6"    "PCL7"
[15] "PCL8"    "PFK26"   "PFK27"   "PHO85"   "PIG1"    "PIG2"    "REG1"
[22] "SIP4"    "SNF1"    "SNF4"    "TYE7"    "UBC8"    "UBP14"   "VID28"
[29] "YLR345W" "GSC2"    "CCT5"    "CPR6"    "CPR7"    "HSC82"   "PET100"
[36] "TIM9"    "COP1"    "GLO3"    "RET2"    "RET3"    "SEC21"   "SEC26"
[43] "SEC27"
```

The method `mgsa` fits the MGSA model. It returns a `MgsaMcmcResults` object whose `print` method displays the most likely active terms. On this example, `mgsa` correctly reports largest posterior probabilities for the terms GO:0006109 and GO:0030663. The call to `set.seed()`, which sets the seed of the random number generator, simply ensures the example of this vignette to be reproducible. It is not required for `mgsa()` to work.

```
> set.seed(0)
> fit = mgsa(example_o, example_go)
> fit
```

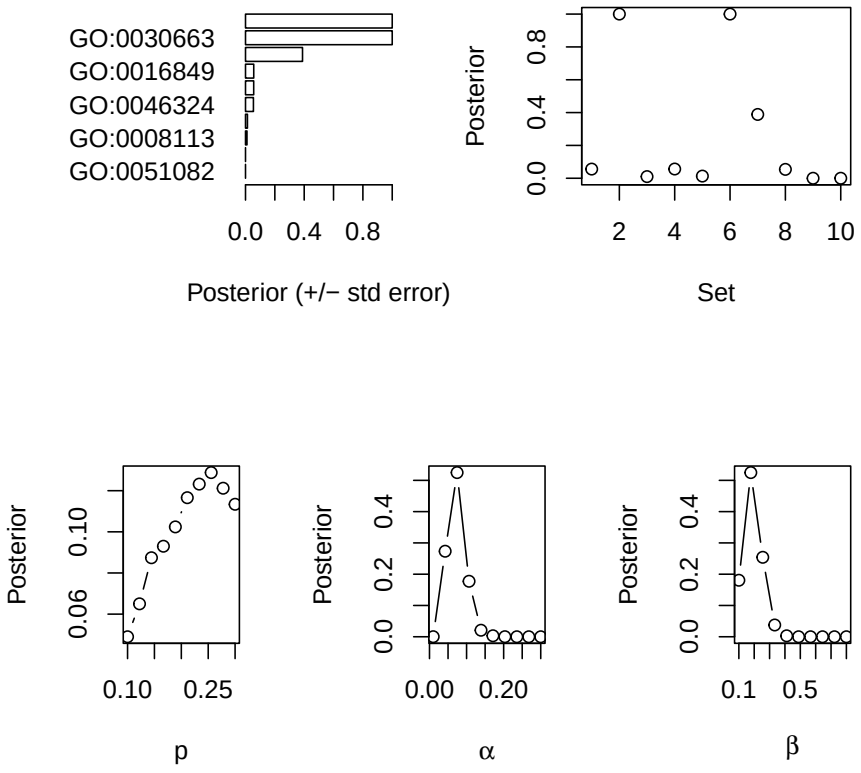
```
Object of class MgsaMcmcResults
158 unique elements in population.
```

```
43 unique elements both in study set and in population.

Posterior on set activity (decreasing order):
           inPopulation inStudySet estimate std.error
GO:0006109           34         28   1.0000        NA
GO:0030663            8          7   1.0000        NA
GO:0046292            2          1   0.3886        NA
GO:0016849            1          0   0.0562        NA
GO:0005093            1          0   0.0558        NA
                                        term                          definition
GO:0006109 regulation of carbohydrate met... Any process that modulates the...
GO:0030663      COPI coated vesicle membrane The lipid bilayer surrounding ...
GO:0046292    formaldehyde metabolic process The chemical reactions and pat...
GO:0016849 phosphorus-oxygen lyase activi... Catalysis of the cleavage of a...
GO:0005093 Rab GDP-dissociation inhibitor... Prevents the dissociation of G...
```

The method `plot` provides a graphical visualization of the fit.

```
> plot(fit)
```



Finally, inference results, i.e., the set activities, can be extracted with the accessor function `setsResults()`. The data frame returned by the function can

be sorted or filtered with R's standard data frame operations. The following code extract the set results into a data frame named `res` and filter for those with posterior probability estimate greater than 0.5.

```
> res = setsResults(fit)
> subset(res, estimate > 0.5)

           inPopulation inStudySet estimate std.error
GO:0006109           34         28        1        NA
GO:0030663            8          7        1        NA
                                                 term
GO:0006109 regulation of carbohydrate metabolic process
GO:0030663                 COPI coated vesicle membrane

GO:0006109 Any process that modulates the frequency, rate or extent of the chemical reacti
GO:0030663                                                                      The lipid b
```

3 Using the Gene Ontology

The Gene Ontology [2] (GO) provides structured annotations to genes. Genes with the same annotation belong to the same gene set. MGSA can be run on these gene sets. GO annotation files for the studied organism can be downloaded from the GO web page: http://www.geneontology.org.

The function `readGAF` creates an `MgsaGoSets` object, a particular `MgsaSets`, from such a gene annotation file. Note that `readGAF` requires the package `GO.db` and `RSQLite` to be installed.

For illustration purposes, a simplified GO annotation file with only three yeast genes is provided:

```
> readGAF(
+         system.file(
+                 "example_files/gene_association_head.sgd",
+                 package="mgsa"
+         )
+ )

Object of class MgsaGoSets
116 sets over 3 unique items.

Set annotations:
                                          term                          definition
GO:0000295 adenine nucleotide transmembra... Enables the transfer of adenin...
GO:0000313               organellar ribosome A ribosome contained within a ...
GO:0000314 organellar small ribosomal sub... The smaller of the two subunit...
GO:0000315 organellar large ribosomal sub... The larger of the two subunits...
GO:0003674                molecular_function A molecular process that can b...
... and  111  other sets.

Item annotations:
             symbol                               name
```

```
S000004660     AAC1 Mitochondrial inner membrane A...
S000007287 15S_RRNA Ribosomal RNA of the small mit...
S000007288 21S_RRNA            Mitochondrial 21S rRNA
```

4 Using custom gene sets

MGSA is not restricted to Gene Ontology and can be applied to any gene sets. The method `mgsa` can directly be called on such gene sets provided as `list` as in the example below.

```
> mgsa( c("A", "B"), list(set1=LETTERS[1:3], set2=LETTERS[2:5]) )

Object of class MgsaMcmcResults
5 unique elements in population.
2 unique elements both in study set and in population.

Posterior on set activity (decreasing order):
     inPopulation inStudySet estimate std.error
set1            3          2   0.6290        NA
set2            4          1   0.1496        NA
```

Internally, the method `mgsa` indexes all elements of the sets before fitting the model. In case `mgsa` must be run on several observations with the same gene sets, computations can be speeded up by performing this indexing once for all. This can be achieved by building a `MgsaSets`.

```
> myset = new( "MgsaSets", sets=list(set1=LETTERS[1:3], set2=LETTERS[2:5]) )
> mgsa(c("A", "B"), myset)

Object of class MgsaMcmcResults
5 unique elements in population.
2 unique elements both in study set and in population.

Posterior on set activity (decreasing order):
     inPopulation inStudySet estimate std.error
set1            3          2   0.5354        NA
set2            4          1   0.0932        NA

> mgsa(c("B", "C"), myset)

Object of class MgsaMcmcResults
5 unique elements in population.
2 unique elements both in study set and in population.

Posterior on set activity (decreasing order):
     inPopulation inStudySet estimate std.error
set1            3          2   0.4454        NA
set2            4          2   0.2360        NA
```

References

[1] S. Bauer, J. Gagneur, and P. N. Robinson. GOing Bayesian: model-based gene set analysis of genome-scale data. *Nucleic acids research*, 2010.

[2] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29,2000.