The *mgsa* package

Sebastian Bauer, Julien Gagneur

13 October 2010

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].

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 positive 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 metabol... The chemical reactio...
GO:0006109 regulation of carboh... Any process that mod...
GO:0008113 peptide-methionine-(... Catalysis of the rea...
GO:0016849 phosphorus-oxygen ly... Catalysis of the cle...
GO:0046527 glucosyltransferase ... Catalysis of the tra...
... and  5  other sets.

Item annotations:
                           name
SFA1    Bifunctional enzyme ...
YJL068C Non-essential intrac...
ADR1    Carbon source-respon...
CAT8    Zinc cluster transcr...
FYV10   Protein of unknown f...
... and  153  other items.
```

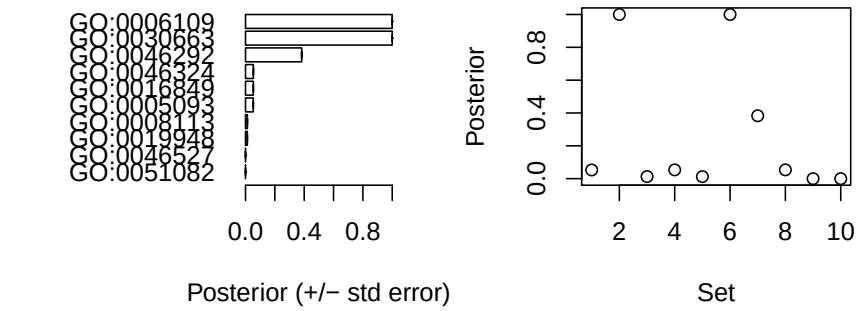
```
> example_o

 [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"
```

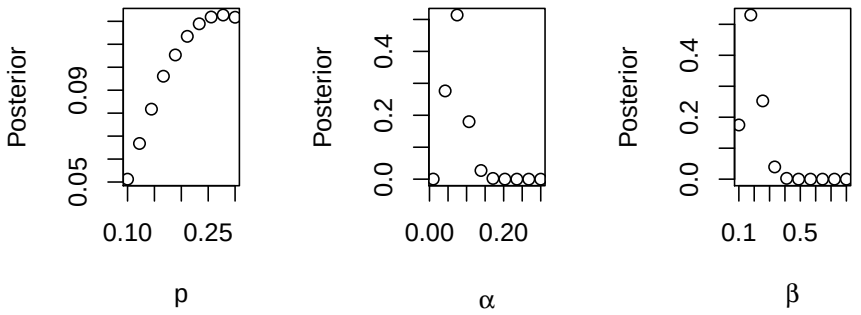
The method `mgsa` fits the MGSA model. It returns a `MgsaMcmcResults` object whose `print` method displays the most likely active terms. On this example, `mgsa` correctly reports largest posterior probabilities for the terms GO:0006109 and GO:0030663. The call to `set.seed()`, which sets the seed of 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.
'data.frame':        10 obs. of  4 variables:
 $ inPopulation: int  1 34 2 1 2 8 2 1 21 86
 $ inStudySet  : int  0 28 0 0 0 7 1 0 1 6
 $ estimate    : num  0.053 1 0.0128 0.0534 0.0124 ...
 $ std.error   : num  9.94e-04 5.50e-06 3.19e-04 6.18e-04 2.99e-04 ...
NULL

Posterior on set activity (decreasing order):
           inPopulation inStudySet  estimate    std.error
GO:0006109           34         28 0.9999814 5.500909e-06
GO:0030663            8          7 0.9999640 1.098180e-05
GO:0046292            2          1 0.3831750 1.474526e-03
GO:0046324            1          0 0.0537238 9.021826e-04
GO:0016849            1          0 0.0533820 6.177710e-04
GO:0005093            1          0 0.0529742 9.942573e-04
GO:0008113            2          0 0.0127592 3.187484e-04
GO:0019948            2          0 0.0124324 2.988773e-04
GO:0046527           21          1 0.0000000 0.000000e+00
GO:0051082           86          6 0.0000000 0.000000e+00
```

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

```
> plot(fit)
```





3 Using the Gene Ontology

The Gene Ontology [2] (GO) provides structured annotations to genes. Genes with the same annotation consitute a gene set. MGSA can be run on these gene sets. GO annotation files for your organism of study can be downloaded at 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
```

```
111 sets over 3 unique items.

Set annotations:
                              term              definition
GO:0000313     organellar ribosome A ribosome contained...
GO:0000314 organellar small rib... The smaller of the t...
GO:0000315 organellar large rib... The larger of the tw...
GO:0003674      molecular_function Elemental activities...
GO:0003735 structural constitue... The action of a mole...
... and  106  other sets.

Item annotations:
             symbol                     name
S000004660     AAC1 Mitochondrial inner ...
S000007287 15S_RRNA Ribosomal RNA of the...
S000007288 21S_RRNA Mitochondrial 21S rR...
```

4 Using custom gene sets

MGSA is not restricted to Gene Ontology and ba 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.
'data.frame':        2 obs. of  4 variables:
 $ inPopulation: int  3 4
 $ inStudySet  : int  2 1
 $ estimate    : num  0.631 0.154
 $ std.error   : num  0.000654 0.000197
NULL

Posterior on set activity (decreasing order):
     inPopulation inStudySet  estimate    std.error
set1            3          2 0.6312916 0.0006535761
set2            4          1 0.1542472 0.0001973723
```

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 = MgsaSets( 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.
'data.frame':        2 obs. of  4 variables:
 $ inPopulation: int  3 4
 $ inStudySet  : int  2 1
 $ estimate    : num  0.63 0.154
 $ std.error   : num  0.000594 0.000378
NULL

Posterior on set activity (decreasing order):
     inPopulation inStudySet  estimate    std.error
set1            3          2 0.6303232 0.0005937597
set2            4          1 0.1542558 0.0003784498
```

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

Object of class MgsaMcmcResults
5 unique elements in population.
2 unique elements both in study set and in population.
'data.frame':        2 obs. of  4 variables:
 $ inPopulation: int  3 4
 $ inStudySet  : int  2 2
 $ estimate    : num  0.527 0.289
 $ std.error   : num  0.000543 0.000175
NULL

Posterior on set activity (decreasing order):
     inPopulation inStudySet  estimate    std.error
set1            3          2 0.5272310 0.0005425706
set2            4          2 0.2889352 0.0001751067
```

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.