The \textit{mgsa} package

Sebastian Bauer, Julien Gagneur

28 April 2011

1 Introduction

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

MGSA takes as input \textit{observations} (such as differentially expressed genes) and a list of gene \textit{sets} (for example pathways or annotated terms of the gene ontologies). The model assumes that some sets to be inferred are \textit{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 \cite{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:

- $\alpha$, the false positive rate i.e. the probability of an inactive gene to actually be observed;
- $\beta$, 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 \cite{1}.

2 Quick start

We start with a small simulated dataset which contains \texttt{example\_go}, a random subset of yeast gene ontology annotations with 20 terms and \texttt{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.

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

Object of class MgsaSets
10 sets over 158 unique items.

Set annotations:

<table>
<thead>
<tr>
<th>term</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GO:0046292</td>
<td>formaldehyde metabolic process The chemical reactions and pat...</td>
</tr>
<tr>
<td>GO:0006109</td>
<td>regulation of carbohydrate met... Any process that modulates the...</td>
</tr>
<tr>
<td>GO:0008113</td>
<td>peptide-methionine-(S)-S-oxide... Catalysis of the reactions: pe...</td>
</tr>
<tr>
<td>GO:0016849</td>
<td>phosphorus-oxygen lyase activi... Catalysis of the cleavage of a...</td>
</tr>
<tr>
<td>GO:0046527</td>
<td>glucosyltransferase activity Catalysis of the transfer of a...</td>
</tr>
</tbody>
</table>

... and 5 other sets.

Item annotations:

<table>
<thead>
<tr>
<th>name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFA1</td>
<td>Bifunctional enzyme containing...</td>
</tr>
<tr>
<td>YJL068C</td>
<td>Non-essential intracellular es...</td>
</tr>
<tr>
<td>ADR1</td>
<td>Carbon source-responsive zinc-...</td>
</tr>
<tr>
<td>CAT8</td>
<td>Zinc cluster transcriptional a...</td>
</tr>
<tr>
<td>FYV10</td>
<td>Protein of unknown function, r...</td>
</tr>
</tbody>
</table>

... and 153 other items.

```r

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

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.

```r
> 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):

<table>
<thead>
<tr>
<th>GO</th>
<th>inPopulation</th>
<th>inStudySet</th>
<th>estimate</th>
<th>std.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>GO:0006109</td>
<td>34</td>
<td>28</td>
<td>1.0000</td>
<td>NA</td>
</tr>
<tr>
<td>GO:0030663</td>
<td>8</td>
<td>7</td>
<td>1.0000</td>
<td>NA</td>
</tr>
<tr>
<td>GO:0046292</td>
<td>2</td>
<td>1</td>
<td>0.3886</td>
<td>NA</td>
</tr>
<tr>
<td>GO:0016849</td>
<td>1</td>
<td>0</td>
<td>0.0562</td>
<td>NA</td>
</tr>
<tr>
<td>GO:0005093</td>
<td>1</td>
<td>0</td>
<td>0.0558</td>
<td>NA</td>
</tr>
</tbody>
</table>

term definition
GO:0006109 regulation of carbohydrate metabolism 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 activity Catalysis of the cleavage of a...
GO:0005093 Rab GDP-dissociation inhibitor activity Prevents the dissociation of G...

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

```r
> 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.

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

<table>
<thead>
<tr>
<th>term</th>
<th>inPopulation</th>
<th>inStudySet</th>
<th>estimate</th>
<th>std.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>GO:0006109 regulation of carbohydrate metabolic process</td>
<td>34</td>
<td>28</td>
<td>1</td>
<td>NA</td>
</tr>
<tr>
<td>GO:0030663 COPI coated vesicle membrane</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>NA</td>
</tr>
</tbody>
</table>
```

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:

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

Object of class `MgsaGoSets`

136 sets over 3 unique items.

Set annotations:

<table>
<thead>
<tr>
<th>term</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GO:0000295 adenine nucleotide transmembrane...</td>
<td>Catalysis of the transfer of a...</td>
</tr>
<tr>
<td>GO:0000313 organellar ribosome A ribosome contained within a ...</td>
<td></td>
</tr>
<tr>
<td>GO:0000314 organellar small ribosomal sub...</td>
<td>The smaller of the two subunits...</td>
</tr>
<tr>
<td>GO:0000315 organellar large ribosomal sub...</td>
<td>The larger of the two subunits...</td>
</tr>
<tr>
<td>GO:003674 molecular_function</td>
<td>Elemental activities, such as ...</td>
</tr>
</tbody>
</table>

... and 131 other sets.

Item annotations:

<table>
<thead>
<tr>
<th>symbol</th>
<th>name</th>
</tr>
</thead>
</table>

4
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.

```r
> 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):

<table>
<thead>
<tr>
<th></th>
<th>inPopulation</th>
<th>inStudySet</th>
<th>estimate</th>
<th>std.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>set1</td>
<td>3</td>
<td>2</td>
<td>0.642</td>
<td>NA</td>
</tr>
<tr>
<td>set2</td>
<td>4</td>
<td>1</td>
<td>0.152</td>
<td>NA</td>
</tr>
</tbody>
</table>

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.

```r
> 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):

<table>
<thead>
<tr>
<th></th>
<th>inPopulation</th>
<th>inStudySet</th>
<th>estimate</th>
<th>std.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>set1</td>
<td>3</td>
<td>2</td>
<td>0.6306</td>
<td>NA</td>
</tr>
<tr>
<td>set2</td>
<td>4</td>
<td>2</td>
<td>0.2858</td>
<td>NA</td>
</tr>
</tbody>
</table>

```r
> 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):

<table>
<thead>
<tr>
<th></th>
<th>inPopulation</th>
<th>inStudySet</th>
<th>estimate</th>
<th>std.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>set1</td>
<td>3</td>
<td>2</td>
<td>0.5206</td>
<td>NA</td>
</tr>
<tr>
<td>set2</td>
<td>4</td>
<td>2</td>
<td>0.2858</td>
<td>NA</td>
</tr>
</tbody>
</table>
References
