Model-based Gene Set Analysis (MGSA) is a Bayesian modeling approach for gene set enrichment. The package mgsa implements MGSA and tools to use MGSA together with the Gene Ontology.

License Artistic-2.0

URL https://github.com/sba1/mgsa-bioc

biocViews Pathways, GO, GeneSetEnrichment

LazyLoad yes


NeedsCompilation yes

R topics documented:

  mgsa-package .................................................. 2
  alphaMcmcPost ................................................. 3
  alphaPost ...................................................... 3
  betaMcmcPost ................................................... 4
  betaPost ......................................................... 4
  createMgsaGoSets ............................................. 5
  example-go ..................................................... 5
  example-o ....................................................... 5
  itemAnnotations ................................................. 6
  itemIndices ..................................................... 6
  length,MgsaSets-method ....................................... 7
Model-based Gene Set Analysis (MGSA) is a Bayesian modeling approach for gene set enrichment. The package mgsa implements MGSA and tools to use MGSA together with the Gene Ontology.

Author(s)

Sebastian Bauer <Sebastian.Bauer@charite.de>, Julien Gagneur <julien.gagneur@embl.de>

References

alphaMcmcPost

**Description**

Posterior estimates of the parameter alpha for each MCMC run.

**Usage**

```r
alphaMcmcPost(x)
```

## S4 method for signature 'MgsaMcmcResults'

```r
alphaMcmcPost(x)
```

**Arguments**

- `x`: a `MgsaMcmcResults`.

**Value**

- `matrix`: Posterior estimates of the parameter alpha for each MCMC run.

---

alphaPost

**Description**

Realization values, posterior estimate and standard error for the parameter alpha.

**Usage**

```r
alphaPost(x)
```

## S4 method for signature 'MgsaResults'

```r
alphaPost(x)
```

**Arguments**

- `x`: a `MgsaResults`.

**Value**

- `data.frame`: realization values, posterior estimate and standard error for the parameter alpha.
betaMcmcPost

posterior estimates of the parameter beta for each MCMC run

Description
Posterior estimates of the parameter beta for each MCMC run.

Usage
betaMcmcPost(x)
## S4 method for signature 'MgsaMcmcResults'
betaMcmcPost(x)

Arguments
x a MgsaMcmcResults.

Value
matrix: Posterior estimates of the parameter beta for each MCMC run.

betaPost

Posterior for beta

Description
Realization values, posterior estimate and standard error for the parameter beta.

Usage
betaPost(x)
## S4 method for signature 'MgsaResults'
betaPost(x)

Arguments
x a MgsaResults.

Value
data.frame: realization values, posterior estimate and standard error for the parameter beta.
createMgsaGoSets

This function takes a 1:1 mapping of go.ids to items and returns a full MgsaGOSets instance. The structure of GO is gathered from GO.db. It is sufficient to specify just the directly asserted mapping (or annotation), i.e., the most specific ones. The true path rule is taken account, that is, if an item is annotated to a term then it will be also annotated to more general terms (some people prefer to say that just the transitive closure is calculated).

Usage

createMgsaGoSets(go.ids, items)

Arguments

go.ids a character vector of GO ids (GO:00001234)
items a vector of identifiers that are annotated to the term in the corresponding position of the go.ids vector.

Example GO sets for mgsa

This data is an example GO set for mgsa.

Example objects for mgsa

This data is an example objects for mgsa.
### itemAnnotations

**Description**

Item annotations of a `MgsaSets`

**Usage**

```r
itemAnnotations(sets, items)
```

#### S4 method for signature 'MgsaSets,missing'

```r
itemAnnotations(sets, items)
```

#### S4 method for signature 'MgsaSets,character'

```r
itemAnnotations(sets, items)
```

**Arguments**

- `sets` an instance of class `MgsaSets`
- `items` character an optional vector specifying the items of interest.

**Value**

a `data.frame`: the item annotations.

### itemIndices

**Description**

Returns the indices corresponding to the items

**Usage**

```r
itemIndices(sets, items)
```

#### S4 method for signature 'MgsaSets,character'

```r
itemIndices(sets, items)
```

#### S4 method for signature 'MgsaSets,numeric'

```r
itemIndices(sets, items)
```

**Arguments**

- `sets` an instance of class `MgsaSets`
- `items` character or numeric the items of interest.

**Value**

a `integer`: the item indices.
length.MgsaSets-method

Length of a MgsaSets.

Description

Length (number of sets) of MgsaSets.

Usage

## S4 method for signature 'MgsaSets'
length(x)

Arguments

x an instance of class MgsaSets.

Value

integer vector.

mgsa

Performs an MGSA analysis

Description

Estimate marginal posterior of the MGSA problem with an MCMC sampling algorithm.

Usage

mgsa(o, sets, population = NULL, p = seq(min(0.1, 1/length(sets)), min(0.3, 20/length(sets)), length.out = 10), ...)

## S4 method for signature 'integer,list'
mgsa(o, sets, population = NULL, p = seq(1, min(20, floor(length(sets)/3)), length.out = 10)/length(sets), ...)

## S4 method for signature 'numeric,list'
mgsa(o, sets, population = NULL, p = seq(1, min(20, floor(length(sets)/3)), length.out = 10)/length(sets), ...)

## S4 method for signature 'character,list'
mgsa(o, sets, population = NULL, p = seq(1, min(20, floor(length(sets)/3)), length.out = 10)/length(sets), ...)

## S4 method for signature 'logical,list'
mgsa(o, sets, population = NULL, p = seq(min(0.1, 1/length(sets)), min(0.3, 20/length(sets)), length.out = 10), ...)
## S4 method for signature 'character,MgsaSets'
mgsa(o, sets, population = NULL,
    p = seq(min(0.1, 1/length(sets)), min(0.3, 20/length(sets)), length.out =
    10), ...)

### Arguments

- **o**: The observations. It can be a numeric, integer, character or logical. See details.
- **sets**: The sets. It can be an `MgsaSets` or a list. In this case, each list entry is a vector of type numeric, integer, character. See details.
- **population**: The total population. Optional. A numeric, integer or character vector. Default to `NULL`. See details.
- **p**: Grid of values for the parameter p. Values represent probabilities of term activity and therefore must be in \([0,1]\].
- **...**: Optional arguments that are passed to the methods. Supported parameters are
  - **alpha**: Grid of values for the parameter alpha. Values represent probabilities of false-positive events and hence must be in \([0,1]\). numeric.
  - **beta**: Grid of values for the parameter beta. Values represent probabilities of false-negative events and hence must be in \([0,1]\). numeric.
  - **steps**: The number of steps of each run of the MCMC sampler. integer of length 1. A recommended value is 1e6 or greater.
  - **burnin**: The number of burn-in MCMC steps, until sample collecting begins. integer of length 1. A recommended value is half of total MCMC steps.
  - **thin**: The sample collecting period. An integer of length 1. A recommended value is 100 to reduce autocorrelation of subsequently collected samples.
  - **flip.freq**: The frequency of MCMC Gibbs step that randomly flips the state of a random set from active to inactive or vice versa. numeric from \((0,1]\).
  - **restarts**: The number of different runs of the MCMC sampler. integer of length 1. Must be greater or equal to 1. A recommended value is 5 or greater.
  - **threads**: The number of threads that should be used for concurrent restarts. A value of 0 means to use all available cores. Default to 0.

### Details

The function can handle items (such as genes) encoded as character or integer. For convenience numeric items can also be provided but these values should essentially be integers. The type of items in the observations `o`, the sets and in the optional population should be consistent. In the case of character items, `o` and population should be of type character and sets can either be an `MgsaSets` or a list of character vectors. In the case of integer items, `o` should be of type integer, numeric (but essentially with integer values), or logical and entries in sets as well as the population should be integer. When `o` is logical, it is first coerced to integer with a call on `which`. Observations outside the population are not taken into account. If population is `NULL`, it is defined as the union of all sets.

The default grid value for `p` is such that between 1 and 20 sets are active in expectation. The lower limit is constrained to be lower than 0.1 and the upper limit lower than 0.3 independently of the total number of sets to make sure that complex solutions are penalized. Marginal posteriors of activity of each set are estimated using an MCMC sampler as described in Bauer et al., 2010. Because convergence of an MCMC sampler is difficult to assess, it is recommended to run it several
times (using restarts). If variations between runs are too large (see \texttt{MgsaResults}), the number of steps (steps) of each MCMC run should be increased.

\textbf{Value}

An \texttt{MgsaMcmcResults} object.

\textbf{References}


\textbf{See Also}

\texttt{MgsaResults, MgsaMcmcResults}

\textbf{Examples}

```r
## observing items A and B, with sets \{A,B,C\} and \{B,C,D\}
mgsa(c("A", "B"), list(set1 = LETTERS[1:3], set2 = LETTERS[2:4]))

## same case with integer representation of the items and logical observation
mgsa(c(TRUE,TRUE,FALSE,FALSE), list(set1 = 1:3, set2 = 2:4))

## a small example with gene ontology sets and plot
data(example)
fit = mgsa(example_o, example_go)
## Not run:
plot(fit)
## End(Not run)
```

\textbf{MgsaGoSets-class}

\textit{Gene Ontology annotations}

\textbf{Description}

This class represents gene ontology annotations.

\textbf{Details}

For now, it is identical to the parental class \texttt{MgsaSets}.

\textbf{See Also}

\texttt{readGAF}
Instances of this class are used to hold the additional information that was provided by running (possibly multiple times) an MCMC algorithm.

**Description**

Instances of this class are used to hold the additional information that was provided by running (possibly multiple times) an MCMC algorithm.

**Slots**

- `nsamples` how many samples collected per MCMC run
- `steps` how many steps per MCMC run
- `restarts` how many MCMC runs
- `alphaMcmcPost` posterior estimates for each MCMC run of the parameter alpha
- `betaMcmcPost` posterior estimates for each MCMC run of the parameter beta
- `pMcmcPost` posterior estimates for each MCMC run of the parameter p
- `setsMcmcPost` posterior estimates for each MCMC run of the sets marginal posterior probabilities

The columns of the matrices `alphaMcmcPost`, `betaMcmcPost`, `pMcmcPost` and `setsMcmcPost` stores the posterior estimates for each individual MCMC run. The row order matches the one of the slot `alphaPost`, `betaPost`, `pPost`, and `setsResults` respectively.

Accessor methods exist for each slot.

**See Also**

`mgsa`

---

**Description**

The results of an MGSA analysis.

**Slots**

- `populationSize` The number of items in the population.
- `studySetSizeInPopulation` The number of items both in the study set and in the population.
- `alphaPost` with columns `value`, `estimate` and `std.error`.
- `betaPost` with columns `value`, `estimate` and `std.error`.
- `pPost` with columns `value`, `estimate` and `std.error`.
setsResults with columns inPopulation, inStudySet, estimate and std.error.
The columns of the slots alphaPost, betaPost, and pPost contains a realization value, its
posterior estimate and standard error for the parameters alpha, beta and p respectively.
The columns of the slot setsResults contains the number of items of the set in the population,
the number of items of the set in the study set, the estimate of its marginal posterior probability
and its standard error. The rownames are the names of the sets if available.
Because an MgsaResults is the outcome of an MGSA analysis (see mgsa), accessor but no
replacement methods exist for each slot.

See Also

mgsa

MgsaSets-class  Sets of items and their annotations

Description

This class describes sets, items and their annotations.

Details

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. In order
to ensure consistency of the indexing, no replace method for any slot is provided. Accessors are
available.

The data frames setAnnotations and itemAnnotations allow to store annotations. No constraint
is imposed on the number and names of their columns.

Slots

sets A list whose elements are vector of item indices.
itemName2ItemIndex The mapping of item names to index.
numberOfItems How many items?
setAnnotations Annotations of the sets. The rownames are set names.
itemAnnotations Annotations of the items. The rownames are item names.

See Also

MgsaGoSets, readGAF, mgsa

Examples

new("MgsaSets", sets=list(set1=c("a", "b"), set2=c("b", "c")))
**nSamples**

*How many samples per MCMC run collected*

**Description**

How many samples collected per MCMC run.

**Usage**

```r
nSamples(x)
```

## S4 method for signature 'MgsaMcmcResults'

```r
nSamples(x)
```

**Arguments**

- `x`: a `MgsaMcmcResults`.

**Value**

`integer`: how many samples per MCMC run collected.

---

**plot,MgsaResults-method**

*Plot method for MgsaResults objects*

**Description**

Plot method for `MgsaResults` objects.

**Usage**

```r
## S4 method for signature 'MgsaResults'
plot(x, y, ...)
```

**Arguments**

- `x`: a `MgsaResults`
- `y`: unused
- `...`: unused
**pMcmcPost**

*posterior estimates of the parameter p for each MCMC run*

**Description**

Posterior estimates of the parameter p for each MCMC run.

**Usage**

```r
pMcmcPost(x)
```

```r
## S4 method for signature 'MgsaMcmcResults'
pMcmcPost(x)
```

**Arguments**

- `x`: a `MgsaMcmcResults`.

**Value**

matrix: Posterior estimates of the parameter p for each MCMC run.

---

**populationSize**

*Size of the population of a MgsaResults*

**Description**

The size of the population on which the analysis was run.

**Usage**

```r
populationSize(x)
```

```r
## S4 method for signature 'MgsaResults'
populationSize(x)
```

**Arguments**

- `x`: a `MgsaResults`.

**Value**

integer: the size of the population.
**pPost**  
*Posterior for beta*

**Description**
Realization values, posterior estimate and standard error for the parameter p.

**Usage**

```r
pPost(x)
```

## S4 method for signature 'MgsaResults'

```r
pPost(x)
```

**Arguments**

- `x` a `MgsaResults`.

**Value**

data.frame: realization values, posterior estimate and standard error for the parameter p.

---

**readGAF**  
*Read a Gene Ontology annotation file*

**Description**
Creates a `MgsaGoSets` using gene ontology annotations provided by a file in GAF 1.0 or 2.0 format.

**Usage**

```r
readGAF(filename, evidence=NULL, aspect=c("P", "F", "C"))
```

**Arguments**

- `filename` The name of the Gene Ontology annotation file. It must be in the GAF 1.0 or 2.0 format. It may be gzip-compressed.
- `evidence` character or NULL. Only annotations with evidence code in evidence are returned. If NULL (default), annotations of all evidence codes are returned.
- `aspect` character with values in P, C or F. Only annotations of the listed GO namespaces P (biological process), F (molecular function) or C (cellular component) are returned. By default, annotations of the three namespaces are returned.

**Details**
The function extracts from the annotation file all direct gene annotations and infers from the Gene Ontology all the indirect annotations (due to term relationships). This is done using the package `Go.db` which provides the ontology as a database and `RSQLite` for querying the database.
Value

An `MgsaGoSets` object.

References


See Also

`MgsaGoSets`, `mgsa`

Examples

```r
## parsing provided example file (yeast)
gofile = system.file("example_files/gene_association_head.sgd", package="mgsa")
readGAF(gofile)
## only annotations inferred from experiment or a direct assay
readGAF(gofile, evidence=c("EXP", "IDA"))
```

### Parsing Provided Example File (Yeast)

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

### Only Annotations Inferred from Experiment or a Direct Assay

```r
readGAF(gofile, evidence=c("EXP", "IDA"))
```

Description

how many MCMC runs.

Usage

```r
restarts(x)
```

Arguments

- `x`: a `MgsaMcmcResults`.

Value

integer: how many MCMC runs.
setAnnotations  Set annotations of a MgsaSets

Description
Set annotations of a MgsaSets.

Usage
setAnnotations(sets, names)

## S4 method for signature 'MgsaSets,missing'
setAnnotations(sets, names)

## S4 method for signature 'MgsaSets,character'
setAnnotations(sets, names)

Arguments
sets an instance of class MgsaSets.
names character an optional vector specifying the names of interest.

Value
a data.frame: the set annotations.

setsMcmcPost posterior estimates of the set marginal probabilities for each MCMC run

Description
Posterior estimates of the set marginal probabilities for each MCMC run.

Usage
setsMcmcPost(x)

## S4 method for signature 'MgsaMcmcResults'
setsMcmcPost(x)

Arguments
x a MgsaMcmcResults.

Value
matrix: Posterior estimates of the set marginal probabilities for each MCMC run.
### setsResults

**Posterior for each set**

**Description**

Number of items of the set in the population, the number of items of the set in the study set, the estimate of its marginal posterior probability and its standard error.

**Usage**

```r
setsResults(x)
```

## S4 method for signature 'MgsaResults'

```r
setsResults(x)
```

**Arguments**

- `x` 
  a `MgsaResults`.

**Value**

data.frame: For each set, number of items of the set in the population, number of items of the set in the study set, estimate of its marginal posterior probability and standard error.

---

### show,MgsaResults-method

**Show an MgsaResults**

**Description**

Show an `MgsaResults`.

**Usage**

```r
## S4 method for signature 'MgsaResults'
show(object)
```

**Arguments**

- `object` 
  an instance of class `MgsaResults`.

**Value**

an invisible null
### show,MgsaSets-method

*Show an MgsaSets*

#### Description

Show an `MgsaSets`.

#### Usage

```r
## S4 method for signature 'MgsaSets'
show(object)
```

#### Arguments

- `object`: an instance of class `MgsaSets`.

#### Value

an invisible `NULL`.

---

### steps

*How many steps per MCMC run*

#### Description

how many steps per MCMC run.

#### Usage

```r
steps(x)
```

#### Arguments

- `x`: a `MgsaMcmcResults`.

#### Value

integer: how many steps per MCMC run.
studySetSizeInPopulation

Size of the study set of a MgsaResults

Description
The size of the study set on which the analysis was run.

Usage
studySetSizeInPopulation(x)

## S4 method for signature 'MgsaResults'
studySetSizeInPopulation(x)

Arguments
x a MgsaResults.

Value
integer: the size of the study set.

subMgsaSets
Subset of an MgsaSets

Description
Returns a subset of an MgsaSets that contains only the specified items. Empty sets are removed.

Usage
subMgsaSets(sets, items)

## S4 method for signature 'MgsaSets,character'
subMgsaSets(sets, items)

Arguments
sets an MgsaSets.
items character. The items to restrict on.

Value
an MgsaSets.
Index

*Topic data
  example-go, 5
  example-o, 5

*Topic package
  mgsa-package, 2

alphaMcmcPost, 3
alphaMcmcPost,MgsaMcmcResults-method (alphaMcmcPost), 3
alphaPost, 3
alphaPost,MgsaResults-method (alphaPost), 3

betaMcmcPost, 4
betaMcmcPost,MgsaMcmcResults-method (betaMcmcPost), 4
betaPost, 4
betaPost,MgsaResults-method (betaPost), 4

createMgsaGoSets, 5
example-go, 5
example-o, 5
example_go (example-go), 5
example_o (example-o), 5

itemAnnotations, 6
itemAnnotations,MgsaSets,character-method (itemAnnotations), 6
itemAnnotations,MgsaSets,missing-method (itemAnnotations), 6
itemIndices, 6
itemIndices,MgsaSets,character-method (itemIndices), 6
itemIndices,MgsaSets,numerical-method (itemIndices), 6

length,MgsaSets-method, 7
mgsa, 7, 10, 11, 15
mgsa,character,list-method (mgsa), 7
mgsa,character,MgsaSets-method (mgsa), 7
mgsa,numeric,list-method (mgsa), 7
mgsa-package, 2
MgsaGoSets, 11, 15
MgsaGoSets-class, 9
MgsaMcmcResults, 3, 4, 9, 12–14, 17, 19
MgsaMcmcResults-class, 10
MgsaResults, 3, 4, 9, 12–14, 17, 19
MgsaResults-class, 10
MgsaSets, 6–9, 11, 16, 18, 19
MgsaSets-class, 11

nsamples, 12
nsamples,MgsaMcmcResults-method (nsamples), 12

plot,MgsaResults-method, 12
pMcmcPost, 13
pMcmcPost,MgsaMcmcResults-method (pMcmcPost), 13
populationSize, 13
populationSize,MgsaResults-method (populationSize), 13
pPost, 14
pPost,MgsaResults-method (pPost), 14

readGAF, 9, 11, 14
restarts, 15
restarts,MgsaMcmcResults-method (restarts), 15
rownames, 11

setAnnotations, 16
setAnnotations,MgsaSets,character-method (setAnnotations), 16
setAnnotations,MgsaSets,missing-method (setAnnotations), 16
setsMcmcPost, 16
setsMcmcPost,MgsaMcmcResults-method (setsMcmcPost), 16
setsResults, 17
setsResults,MgsaResults-method (setsResults), 17
show,MgsaResults-method, 17
show,MgsaSets-method, 18
INDEX

steps, 18
steps,MgsaMcmcResults-method(steps), 18
studySetSizeInPopulation, 19
studySetSizeInPopulation,MgsaResults-method
(studySetSizeInPopulation), 19
subMgsaSets, 19
subMgsaSets,MgsaSets,character-method
(subMgsaSets), 19

which, 8