Package 'BAGS'

October 14, 2021

Type Package

Title A Bayesian Approach for Geneset Selection

Version 2.32.0

Date 2013-06-12

Author Alejandro Quiroz-Zarate

Maintainer Alejandro Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

Description R package providing functions to perform geneset significance analysis over simple cross-sectional data between 2 and 5 phenotypes of interest.

Depends R (>= 2.10), breastCancerVDX, Biobase

License Artistic-2.0

Collate DataGeneSets.R Gibbs5.R Gibbs4.R Gibbs3.R Gibbs2.R GrpMean.R MCMCDataSet.R ReadGMT.R

LazyLoad yes

URL http //compbio.dfci.harvard.edu/

biocViews Bayesian

git_url https://git.bioconductor.org/packages/BAGS

git_branch RELEASE_3_13

git_last_commit d94e763

git_last_commit_date 2021-05-19

Date/Publication 2021-10-14

R topics documented:

`BAGS-package` *A Bayesian Approach for Geneset Selection (BAGS).*

Description

Functions to perform geneset significance analysis for data sets with cross-sectional or time course data design. This method is described in detail in the paper soon to be published. This work is done by Alejandro Quiroz-Zarate in collaboration with Benjamin Haibe-Kains and John Quackenbush.

Details

Package:	BAGS
Type:	Package
Version:	2.0
Date:	2013-06-12
License:	Artistics-2.0
LazyLoad:	yes

For a detailed example on the use and manipulation of the functions provided on this package please see the package Vignette.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

References

Quiroz-Zarate, A., Haibe-Kains, B. and Quackenbush, J. Manuscript in preparation (2013).

Examples

```
#vignette("BAGS")
```

`AnnotationMFGO` *List containing a collection of gene symbols with their associated GO term for the Molecular function ontology*

Description

This dataset contains list containing all the GO terms for the Molecular function ontology with their associated gene symbols, base on the collections provided by the MSigDB from the Broad Institute

Usage

```
data(AnnotationMFGO)
```

Format

A list containing the information in the following way:

- `AnnotationMFGO[1:3]`: The first 3 GO terms of the Molecular Function ontology with their associated gene symbols.

Details

This dataset contains list containing gene symbols associated with their respective GO term, based on the from the MSigDB at the Broad Institute. This dataset enables the construction of the gene groups of interest for the analysis of the methodology proposed.

Author(s)

A. Quiroz-Zarate. <aquiroz@jimmy.harvard.edu>

Source

http://www.broadinstitute.org/gsea/msigdb/collections.jsp#C5

References

Quiroz-Zarate A and Quackenbush J (2012). "Manuscript in preparation".

Examples

```
#- load the dataset
data(AnnotationMFGO,package="BAGS")
#- show the first 20 rows and columns.
AnnotationMFGO[1:3]
```

DataGeneSets *Function that filters the gene sets to work with the desired size.*

Description

This function provides the gene sets with the desired size. It provides two lists of gene sets, one with the gene identifiers of interest, and the other with the position of the gene identifiers with respect to the dataset. Finally it provides a list of the sizes of all the gene sets considered.

Usage

```
DataGeneSets(output.ReadGMT, data.gene.symbols, size)
```

Arguments

output.ReadGMT	Output of the function ReadGMT.
data.gene.symbols	Vector with the gene identifiers associated to the dataset interest. These gene identifiers have to the same as the ones on the .gmt file of interest.
size	Integer with the minimum number of genes in a gene set.

Details

This function constructs the gene sets that are going to be considered in the analysis based on the desired size.

Value

This function returns a list with three items

DataGeneSetsIds	A list of gene sets with the positions of the gene identifiers with respect to the dataset of interest.
DataGeneSetsNms	A list of gene sets of the gene identifiers.
Size	A vector with the size of the gene sets

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use this function and the help of the function Gibbs5 for a detailed example of its use.

Examples

```
library(breastCancerVDX)
library(Biobase)

data(vdx,package="breastCancerVDX")
gene.expr=exprs(vdx)   # Gene expression of the package
vdx.annot=fData(vdx)   # Annotation associated to the dataset
vdx.clinc=pData(vdx)   # Clinical information associated to the dataset

# Identifying the sample identifiers associated to ER+ and ER- breast cancer
er.pos=which(vdx.clinc$er==1)
er.neg=which(vdx.clinc$er==0)

# Only keep columns 1 and 3, probeset identifiers and Gene symbols respectively
vdx.annot=vdx.annot[,c(1,3)]

all(rownames(gene.expr)==as.character(vdx.annot[,1])) # Checking if the probeset are ordered with respect to the c
all(colnames(gene.expr)==as.character(vdx.clinc[,1])) # Checking if the sample identifiers are order with respect
rownames(gene.expr)=as.character(vdx.annot[,2])     # Changing the row identifiers to the gene identifiers of int

#===== Because we have several measurements for a gene (multiple rows for a gene), we filter the genes
#===== Function to obtain the genes with highest variabilty among phenotypes
gene.nms.u=unique(rownames(gene.expr))
gene.nms=rownames(gene.expr)
indices=NULL
for(i in 1:length(gene.nms.u))
{
aux=which(gene.nms==gene.nms.u[i])
if(length(aux)>1){
var.r = apply(cbind(apply(gene.expr[aux,er.pos],1,mean),apply(gene.expr[aux,er.neg],1,mean)),1,var)
aux=aux[which.max(var.r)]
}
indices=c(indices,aux)
}
#===== Only keep the genes with most variability among the phenotypes of interest
gene.expr=gene.expr[indices,]
gene.nams=rownames(gene.expr)     # The gene symbols of interest are stored here


#===== In the following R dataset it is stored the .gmt file associated to the MF from GO.
#===== So "reading the GMT" is the only step that we skip. But an example is provided on the
#===== help file associated to the function "ReadGMT".
data(AnnotationMFGO,package="BAGS")

data.gene.grps=DataGeneSets(AnnotationMFGO,gene.nams,10)
```

`Gibbs2` *Function obtains the MCMC chains for the parameters of interest that will form their posterior distribution.*

Description

This function provides the MCMC chains for the parameters of interest that will form their posterior distribution. This function is to obtain the gene sets that are differentially expressed among five phenotypes of interest, taking into account one as baseline.

Usage

```
Gibbs2(noRow,noCol,iter,GrpSzs,YMu,L0,V0,L0A,V0A,MM,AAPi,ApriDiffExp,result1)
```

Arguments

noRow	Number of row of the dataset
noCol	Total number of subjects considered.
iter	Number of iterations for the Gibbs sampler.
GrpSzs	Vector with the sizes of the gene sets considered. Output from the function DataGeneSets.
YMu	Output y.mu from the MCMCDataSet
L0	Vector with the prior parameters.
V0	Vector with the prior parameters.
L0A	Vector with the prior parameters.
V0A	Vector with the prior parameters.
MM	Parameter of the prior.
AAPi	Parameter of the prior.
ApriDiffExp	Number of differentially expressed gene sets apriori
result1	Matrix for the MCMC chains for the parameter that identifies the difference in geneset expression from phenotype 1 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.

Details

This function provides the MCMC chains for the estimation of the posterior distribution for the parameters of interest for each gene set.

Value

This function returns a list with four items

alfa.1	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 1 with respect to the phenotype chosen as baseline.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use this function and the help of the function Gibbs2 for a detailed example of its use. This function can also be used when the gene expression data has a time series experimental design. In this case, there will be two time points on the time course sampling. The assumption is that measurements between time points are independent. This assumption is reasonable when there is irregular and sparse time course sampling.

Examples

```
library(breastCancerVDX)
library(Biobase)

data(vdx)
gene.expr=exprs(vdx)   # Gene expression of the package
vdx.annot=fData(vdx)   # Annotation associated to the dataset
vdx.clinc=pData(vdx)   # Clinical information associated to the dataset

# Identifying the sample identifiers associated to ER+ and ER- breast cancer
er.pos=which(vdx.clinc$er==1)
er.neg=which(vdx.clinc$er==0)

# Only keep columns 1 and 3, probeset identifiers and Gene symbols respectively
vdx.annot=vdx.annot[,c(1,3)]

all(rownames(gene.expr)==as.character(vdx.annot[,1])) # Checking if the probeset are ordered with respect to the c
all(colnames(gene.expr)==as.character(vdx.clinc[,1])) # Checking if the sample identifiers are order with respect
rownames(gene.expr)=as.character(vdx.annot[,2])     # Changing the row identifiers to the gene identifiers of int

#===== Because we have several measurements for a gene (multiple rows for a gene), we filter the genes
#===== Function to obtain the genes with highest variabilty among phenotypes
gene.nms.u=unique(rownames(gene.expr))
gene.nms=rownames(gene.expr)
indices=NULL
for(i in 1:length(gene.nms.u))
{
aux=which(gene.nms==gene.nms.u[i])
if(length(aux)>1){
var.r = apply(cbind(apply(gene.expr[aux,er.pos],1,mean),apply(gene.expr[aux,er.neg],1,mean)),1,var)
aux=aux[which.max(var.r)]
}
indices=c(indices,aux)
}
#===== Only keep the genes with most variability among the phenotypes of interest
gene.expr=gene.expr[indices,]
gene.nams=rownames(gene.expr)     # The gene symbols of interest are stored here

dim(gene.expr)

#===== In the following R dataset it is stored the .gmt file associated to the MF from GO.
#===== So "reading the GMT" is the only step that we skip. But an example is provided on the
#===== help file associated to the function "ReadGMT".
data(AnnotationMFGO,package="BAGS")
```

```
data.gene.grps=DataGeneSets(AnnotationMFGO,gene.nams,5)

phntp.list=list(er.pos,er.neg)
data.mcmc=MCMCDataSet(gene.expr,data.gene.grps$DataGeneSetsIds,phntp.list)

noRow=dim(data.mcmc$y.mu)[1]
noCol=unlist(lapply(phntp.list,length))
iter=10000
GrpSzs=data.gene.grps$Size
YMu=data.mcmc$y.mu
L0=rep(2,2)
V0=rep(3,2)
L0A=rep(3,2)
V0A=rep(3,2)
MM=0.55
AAPi=10
ApriDiffExp=floor(dim(data.mcmc$y.mu)[1]*0.03)
results=matrix(0,noRow,iter)

mcmc.chains=Gibbs2(noRow,noCol,iter,GrpSzs,YMu,L0,V0,L0A,V0A,MM,AAPi,ApriDiffExp,results)

burn.in=2000
alfa.pi=apply(mcmc.chains[[1]][,burn.in:iter],1,function(x){
                                y=length(which(x!=0))/length(burn.in:iter);return(y)})
plot(alfa.pi,type="h",main="Probabilities of MF differentially expressed between ER status")
cut.off=0.9
abline(h=cut.off,col="red")
differential.processes=names(data.gene.grps$Size)[which(alfa.pi>cut.off)]
```

`Gibbs3` *Function obtains the MCMC chains for the parameters of interest that will form their posterior distribution.*

Description

This function provides the MCMC chains for the parameters of interest that will form their posterior distribution. This function is to obtain the gene sets that are differentially expressed among five phenotypes of interest, taking into account one as baseline.

Usage

```
Gibbs3(noRow,noCol,iter,GrpSzs,YMu,L0,V0,L0A,V0A,MM,AAPi,ApriDiffExp,result1,result2)
```

Arguments

`noRow`	Number of row of the dataset
`noCol`	Total number of subjects considered.
`iter`	Number of iterations for the Gibbs sampler.

GrpSzs	Vector with the sizes of the gene sets considered. Output from the function DataGeneSets.
YMu	Output y.mu from the MCMCDataSet
L0	Vector with the prior parameters.
V0	Vector with the prior parameters.
L0A	Vector with the prior parameters.
V0A	Vector with the prior parameters.
MM	Parameter of the prior.
AAPi	Parameter of the prior.
ApriDiffExp	Number of differentially expressed gene sets apriori
result1	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 1 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
result2	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 2 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.

Details

This function provides the MCMC chains for the estimation of the posterior distribution for the parameters of interest for each gene set.

Value

This function returns a list with four items

alfa.1	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 1 with respect to the reference phenotype.
alfa.2	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 2 with respect to the reference phenotype.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use function Gibbs2. This function can also be used when the gene expression data has a time series experimental design. In this case, there will be three time points on the time course sampling. The assumption is that measurements between time points are independent. This assumption is reasonable when there is irregular and sparse time course sampling.

Examples

```
# Similar to the example on Gibbs2, but in this case there are three different phenotypes of interest. The user has t
```

`Gibbs4` *Function obtains the MCMC chains for the parameters of interest that will form their posterior distribution.*

Description

This function provides the MCMC chains for the parameters of interest that will form their posterior distribution. This function is to obtain the gene sets that are differentially expressed among five phenotypes of interest, taking into account one as baseline.

Usage

```
Gibbs4(noRow,noCol,iter,GrpSzs,YMu,L0,V0,L0A,V0A,MM,AAPi,ApriDiffExp,result1,result2,result3)
```

Arguments

`noRow`	Number of row of the dataset
`noCol`	Total number of subjects considered.
`iter`	Number of iterations for the Gibbs sampler.
`GrpSzs`	Vector with the sizes of the gene sets considered. Output from the function `DataGeneSets`.
`YMu`	Output y.mu from the `MCMCDataSet`
`L0`	Vector with the prior parameters.
`V0`	Vector with the prior parameters.
`L0A`	Vector with the prior parameters.
`V0A`	Vector with the prior parameters.
`MM`	Parameter of the prior.
`AAPi`	Parameter of the prior.
`ApriDiffExp`	Number of differentially expressed gene sets apriori
`result1`	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 1 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
`result2`	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 2 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
`result3`	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 3 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.

Details

This function provides the MCMC chains for the estimation of the posterior distribution for the parameters of interest for each geneset.

Value

This function returns a list with four items

`alfa.1`	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 1 with respect to the reference phenotype.
`alfa.2`	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 2 with respect to the reference phenotype.
`alfa.3`	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 3 with respect to the reference phenotype.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use function Gibbs2. This function can also be used when the gene expression data has a time series experimental design. In this case, there will be four time points on the time course sampling. The assumption is that measurements between time points are independent. This assumption is reasonable when there is irregular and sparse time course sampling.

Examples

```
# Similar to the example on Gibbs2, but in this case there are four different phenotypes of interest.  The user has to
```

Gibbs5 *Function obtains the MCMC chains for the parameters of interest that will form their posterior distribution.*

Description

This function provides the MCMC chains for the parameters of interest that will form their posterior distribution. This function is to obtain the gene sets that are differentially expressed among five phenotypes of interest, taking into account one as baseline.

Usage

```
Gibbs5(noRow,noCol,iter,GrpSzs,YMu,L0,V0,L0A,V0A,MM,AAPi,ApriDiffExp,result1,result2,result3,result
```

Arguments

noRow	Number of row of the dataset
noCol	Total number of subjects considered.
iter	Number of iterations for the Gibbs sampler.
GrpSzs	Vector with the sizes of the gene sets considered. Output from the function DataGeneSets.
YMu	Output y.mu from the MCMCDataSet
L0	Vector with the prior parameters.
V0	Vector with the prior parameters.
L0A	Vector with the prior parameters.
V0A	Vector with the prior parameters.
MM	Parameter of the prior.
AAPi	Parameter of the prior.
ApriDiffExp	Number of differentially expressed gene sets apriori
result1	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 1 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
result2	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 2 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
result3	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 3 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.
result4	Matrix for the MCMC chains for the parameter that identifies the difference in gene set expression from phenotype 4 in comparison with the phenotype chosen as baseline. The rows are for the gene sets and the columns for the number of iterations.

Details

This function provides the MCMC chains for the estimation of the posterior distribution for the parameters of interest for each gene set.

Value

This function returns a list with four items

alfa.1	A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 1 with respect to the reference phenotype.

`alfa.2` A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 2 with respect to the reference phenotype.

`alfa.3` A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 3 with respect to the reference phenotype.

`alfa.4` A list with the MCMC chains for the estimation of the posterior distribution for the parameter associated with the comparison of phenotype 4 with respect to the reference phenotype.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use function Gibbs2. This function can also be used when the gene expression data has a time series experimental design. In this case, there will be five time points on the time course sampling. The assumption is that measurements between time points are independent. This assumption is reasonable when there is irregular and sparse time course sampling.

Examples

```
# Similar to the example on Gibbs2, but in this case there are five different phenotypes of interest.  The user has to
```

MCMCDataSet *Function that transform the dataset into the required format for the Gibbs sampler.*

Description

This function makes the necessary transformation of the dataset in order for the Gibbs sampler to perform the iterations. This transformation is based on the number of phenotypes of interest considered.

Usage

```
MCMCDataSet(data,output.DataGeneSets,list.phenotype.ids)
```

Arguments

`data` The dataset of interest. The data has as rows the gene identifiers of interest (same as the .gmt file) and as columns the samples considered.

`output.DataGeneSets` List of gene sets with the positions of the gene identifiers with respect to the dataset of interest. This list is part of the output from DataGeneSets.

`list.phenotype.ids`
: A list that has as elements the vectors with the column positions of the phenotypes considered in the analysis.

Details

This function constructs the gene sets that are going to be considered in the analysis based on the desired size.

Value

This function returns a list:

`y.mu`
: A matrix with the means across samples and genes for each gene set. The rows are gene sets and columns are the different phenotypes considered.

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use this function and the help of the function Gibbs5 for a detailed example of its use.

Examples

```
library(breastCancerVDX)
library(Biobase)

data(vdx)
gene.expr=exprs(vdx)   # Gene expression of the package
vdx.annot=fData(vdx)   # Annotation associated to the dataset
vdx.clinc=pData(vdx)   # Clinical information associated to the dataset

# Identifying the sample identifiers associated to ER+ and ER- breast cancer
er.pos=which(vdx.clinc$er==1)
er.neg=which(vdx.clinc$er==0)

# Only keep columns 1 and 3, probeset identifiers and Gene symbols respectively
vdx.annot=vdx.annot[,c(1,3)]

all(rownames(gene.expr)==as.character(vdx.annot[,1])) # Checking if the probeset are ordered with respect to the c
all(colnames(gene.expr)==as.character(vdx.clinc[,1])) # Checking if the sample identifiers are order with respect
rownames(gene.expr)=as.character(vdx.annot[,2])     # Changing the row identifiers to the gene identifiers of int

#===== Because we have several measurements for a gene (multiple rows for a gene), we filter the genes
#===== Function to obtain the genes with highest variabilty among phenotypes
gene.nms.u=unique(rownames(gene.expr))
gene.nms=rownames(gene.expr)
indices=NULL
for(i in 1:length(gene.nms.u))
```

```
{
aux=which(gene.nms==gene.nms.u[i])
if(length(aux)>1){
var.r = apply(cbind(apply(gene.expr[aux,er.pos],1,mean),apply(gene.expr[aux,er.neg],1,mean)),1,var)
aux=aux[which.max(var.r)]
}
indices=c(indices,aux)
}
#===== Only keep the genes with most variability among the phenotypes of interest
gene.expr=gene.expr[indices,]
gene.nams=rownames(gene.expr)     # The gene symbols of interest are stored here

dim(gene.expr)

#===== In the following R dataset it is stored the .gmt file associated to the MF from GO.
#===== So "reading the GMT" is the only step that we skip. But an example is provided on the
#===== help file associated to the function "ReadGMT".
data(AnnotationMFGO,package="BAGS")

data.gene.grps=DataGeneSets(AnnotationMFGO,gene.nams,10)

phntp.list=list(er.pos,er.neg)
data.mcmc=MCMCDataSet(gene.expr,data.gene.grps$DataGeneSetsIds,phntp.list)
```

ReadGMT *Function that extracts the gene set definition from a .gmt file.*

Description

This function reads the gene set definitions provided by a .gmt file (MSigDB annotation files) an stores the information into a list.

Usage

```
ReadGMT(path)
```

Arguments

path	The path where the .gmt file is stored..

Details

This function reads the gene set definitions provided by a .gmt file (MSigDB annotation files) an stores the information into a list. The .gmt file needs to have the name of the gene set in column 1. A description/properties of the gene set in column 2. And finally the gene identifiers associated to the gene set from column 3 and on. These gene identifiers can be Ensembl, Affymetrix, Gene Symbols, etc...

Value

This function returns a list of the same length as gene sets in the .gmt file provided. The entries of the lists are the associated gene identifier associated to their respective gene set

Author(s)

A. Quiroz-Zarate <aquiroz@jimmy.harvard.edu>

See Also

See the BAGS Vignette for examples on how to use function Gibbs2.

Examples

```
# An example on the use of this function:
# gene.sets.DB=ReadGMT("/Users/Bioinformatics/Projects/c5.mf.v3.0.symbols.gmt")
#
# #Where the path is where the gmt file from the MSigDB is stored.
```

Index