An Introduction to GSEABase

Martin Morgan [1]

[1]Roswell Park Cancer Institute, Buffalo, NY

November 1, 2022

Abstract

The *GSEABase* package implements data structures and methods to represent, manipulate, and analyze gene sets in the context of gene set enrichment analysis. This includes construction of gene sets from reference resources, ID mapping, coloring according to phenotype association, and storing in gene set collections.

Package

GSEABase 1.60.0

Report issues on https://github.com/Bioconductor/GSEABase/issues

Contents

1 *GeneSet*

A *GeneSet* stores a set of related gene identifiers. Important components of the gene set are a vector of identifiers, general descriptive information about the set, and information about how the gene set was constructed. To construct a gene set, use `GeneSet`. For example, to create a gene set from the identifiers in a subset of the sample *ExpressionSet* in the *Biobase* package use

```
data(sample.ExpressionSet) # from Biobase
egs <- GeneSet(sample.ExpressionSet[201:250,], setName="Sample")
egs
```

```
## setName: Sample
## geneIds: 31440_at, 31441_at, ..., 31489_at (total: 50)
## geneIdType: Annotation (hgu95av2)
## collectionType: ExpressionSet
## details: use 'details(object)'
```

Each gene set may have a name. The gene set contains 50 identifiers ('genes') from the *ExpressionSet*. These are accessible using `geneIds`, e.g.,

```
head(geneIds(egs))
```

```
## [1] "31440_at"   "31441_at"   "31442_at"   "31443_at"   "31444_s_at"
## [6] "31445_at"
```

The gene set records that the identifiers are probeset names from the annotation package *hgu95av2.db*, and that the source of the gene set was an *ExpressionSet*. Additional details are available:

```
details(egs)
```

```
## setName: Sample
## geneIds: 31440_at, 31441_at, ..., 31489_at (total: 50)
## geneIdType: Annotation (hgu95av2)
## collectionType: ExpressionSet
## setIdentifier: nebbiolo2:1831558:Tue Nov  1 17:42:57 2022:1
## description: Smoking-Cancer Experiment
##   (longDescription available)
## organism: Homo sapiens
## pubMedIds:
## urls: www.lab.not.exist
## contributor: Pierre Fermat
## setVersion: 0.0.1
## creationDate:
```

The set identifier, set version, and creation date provide mechanisms for carefully curating gene sets. Additional information is automatically copied from the expression set used to create `egs`.

```
## FIXME: GeneSet(AnnotationIdentifier("hgu95av2")) --> non-empty
## FIXME: GeneSet(AnnotationIdentifier("hgu95av2"),
## collectionType=GOCollection()) filters on GOCollection (or KEGG)
```

View additional methods for creating gene sets with:

```
showMethods("GeneSet", inherited=FALSE)
```

```
## Function: GeneSet (package GSEABase)
## type="BroadCollection"
## type="ExpressionSet"
## type="GOCollection"
## type="GeneIdentifierType"
## type="character"
## type="missing"
```

These are described on the `GeneSet` help page.

The identifier type of gene sets created from expression sets is AnnotationIdentifier. Additional predefined identifiers are available:

```
names(slot(getClass("GeneIdentifierType"), "subclasses"))
```

```
##  [1] "NullIdentifier"       "EnzymeIdentifier"     "GenenameIdentifier"
##  [4] "RefseqIdentifier"     "SymbolIdentifier"     "UniprotIdentifier"
##  [7] "ENSEMBLIdentifier"    "AnnotationIdentifier" "EntrezIdentifier"
```

```
## [10] "GOAllFrameIdentifier" "KEGGFrameIdentifier"
```

It is possible to map between identifier types (provided the corresponding map in the annotation package exists):

```
mapIdentifiers(egs, EntrezIdentifier())
```

```
## setName: Sample
## geneIds: 6932, 643332, ..., 4287 (total: 31)
## geneIdType: EntrezId (hgu95av2)
## collectionType: ExpressionSet
## details: use 'details(object)'
```

`mapIdentifiers` consults the gene set to determine that annotation (probeset) identifiers are to be converted to Entrez IDs based on the mapping in the *hgu95av2.db* package. The function `mapIdentifiers` can automatically determine many of the common maps; it is also possible to provide as a third argument an environment containing an arbitrary map. Use the `verbose` argument of `mapIdentifiers` to be informed when the map between identifier types is not 1:1.

A gene set can be created with different types of identifier, e.g., to create a gene set with Entrez IDs, use

```
library(annotate)                       # getEG
eids <- unique(getEG(geneIds(egs), "hgu95av2"))
eids <- eids[!is.na(eids)]
GeneSet(EntrezIdentifier(), geneIds=as.character(eids))
```

```
## setName: NA
## geneIds: 6932, 643332, ..., 4287 (total: 31)
## geneIdType: EntrezId
## collectionType: Null
## details: use 'details(object)'
```

The `collectionType` of a gene set provides additional information about a gene set. Available collection types are

```
names(slot(getClass("CollectionType"), "subclasses"))
```

```
##  [1] "NullCollection"          "ExpressionSetCollection"
##  [3] "ComputedCollection"      "CollectionIdType"
##  [5] "BroadCollection"         "KEGGCollection"
##  [7] "OMIMCollection"          "PMIDCollection"
##  [9] "ChrCollection"           "ChrlocCollection"
## [11] "MapCollection"           "PfamCollection"
## [13] "PrositeCollection"       "GOCollection"
## [15] "OBOCollection"
```

Collection types are most important when creating gene sets. For instance, the *GOCollection* class allows creation of gene sets based on gene ontology (GO) terms. The following command creates a gene set from two terms, including all genes with a particular evidence code.

```
GeneSet(GOCollection(c("GO:0005488", "GO:0019825"),
                     evidenceCode="IDA"),
```

```
        geneIdType=EntrezIdentifier("org.Hs.eg.db"),
        setName="Sample GO Collection")
```

```
## setName: Sample GO Collection
## geneIds:  (total: 0)
## geneIdType: EntrezId (org.Hs.eg.db)
## collectionType: GO
##   ids: GO:0005488, GO:0019825 (2 total)
##   evidenceCode: IDA
##   ontology: CC MF BP
## details: use 'details(object)'
```

This creates a gene set by consulting an object in the *GO.db* package. A gene set created from an expression set, and with collection type *GOCollection* consults the appropriate environment in the annotation package associated with the expression set.

Gene sets encoded in XML following the schema and conventions of the Broad Institute can be read into *R* as follows:

```
fl <- system.file("extdata", "Broad1.xml", package="GSEABase")
bgs <- GeneSet(BroadCollection(), urls=fl)
bgs
```

```
## setName: chr16q24
## geneIds: GALNS, C16ORF44, ..., TRAPPC2L (total: 129)
## geneIdType: Symbol
## collectionType: Broad
##   bcCategory: c1 (Positional)
##   bcSubCategory: NA
## details: use 'details(object)'
```

The example above uses a local file, but the source for the gene set might also be a web address accessible via the `http://` protocol. The file name is added to the url of the gene set. The set name and category of the Broad collection indicate that the gene set contains gene symbols from band 24 of the q arm of chromosome 16. The probe sets in chip `hgu95av2` corresponding to these symbols can be determined by mapping identifiers

```
bgs1 <- mapIdentifiers(bgs, AnnotationIdentifier("hgu95av2"))
bgs1
```

```
## setName: chr16q24
## geneIds: 32100_r_at, 32101_at, ..., 35807_at (total: 37)
## geneIdType: Annotation (hgu95av2)
## collectionType: Broad
##   bcCategory: c1 (Positional)
##   bcSubCategory: NA
## details: use 'details(object)'
```

Subsetting creates sets with just the symbols identified. Subsetting can use indices or symbol names.

```
bgs[1:5]
```

```
## setName: chr16q24
```

```
## geneIds: GALNS, C16ORF44, ..., LOC646365 (total: 5)
## geneIdType: Symbol
## collectionType: Broad
##   bcCategory: c1 (Positional)
##   bcSubCategory: NA
## details: use 'details(object)'
```

```
bgs[c("GALNS", "LOC646365")]
```

```
## setName: chr16q24
## geneIds: GALNS, LOC646365 (total: 2)
## geneIdType: Symbol
## collectionType: Broad
##   bcCategory: c1 (Positional)
##   bcSubCategory: NA
## details: use 'details(object)'
```

Logical operations provide a convenient way to identify genes with particular properties. For instance, the intersection

```
egs & bgs1
```

```
## setName: (Sample & chr16q24)
## geneIds:  (total: 0)
## geneIdType: Annotation (hgu95av2)
## collectionType: Computed
## details: use 'details(object)'
```

is empty (note that the identifiers in the two sets were of the same type), indicating that none of the identifiers in `egs` are on `16q24`. Additional operations on sets include union (performed with `|`) and difference (`setdiff`).

Methods exist to directly subset expression sets using gene sets

```
sample.ExpressionSet[bgs,]
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 2 features, 26 samples
##   element names: exprs, se.exprs
## protocolData: none
## phenoData
##   sampleNames: A B ... Z (26 total)
##   varLabels: sex type score
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation: hgu95av2
```

Remember that `sample.ExpressionSet` contains just 500 probe sets, so the small size of the subset is not surprising. Note also that subsetting required mapping of symbol identifiers in `bgs` to *AnnotationIdentifier*s; this map used the annotation information in the expression set, and is not necessarily 1:1.

2 GeneColorSet

A *GeneColorSet* is a gene set with "coloring" to indicate how features of genes and phenotypes are associated. The following sample data describes how changes in expression levels of several genes (with Entrez and Symbol names) influence cisplatin resistance phenotype.

```
tbl
```

```
##   Entrez.ID Gene.Symbol Expression.level Phenotype.response
## 1      1244       ABCC2         Increase          Resistant
## 2       538       ATP7A         Increase          Resistant
## 3       540       ATP7B         Increase          Resistant
## 4      9961         MVP         Increase          Resistant
## 5      7507         XPA         Increase          Resistant
## 6      2067       ERCC1         Increase          Resistant
## 7       672       BRCA1         Increase          Resistant
## 8      3725         JUN         Increase          Resistant
## 9      2730        GCLM         Increase          Resistant
```

Note that three different aspects of data influence coloring: the phenotype under consideration (cisplatin resistance), whether expression responses refer to increasing or decreasing levels of gene expression, and whether the phenotypic response represents greater resistance or sensitivity to cisplatin. Here is the resulting gene color set:

```
gcs <- GeneColorSet(EntrezIdentifier(),
                    setName="A color set",
                    geneIds=as.character(tbl$Entrez.ID),
                    phenotype="Cisplatin resistance",
                    geneColor=tbl$Expression.level,
                    phenotypeColor=tbl$Phenotype.response)
gcs
```

```
## setName: A color set
## geneIds: 1244, 538, ..., 2730 (total: 9)
## geneIdType: EntrezId
## collectionType: Null
## phenotype: Cisplatin resistance
## geneColor: Increase, Increase, ..., Increase
##   levels: Increase
## phenotypeColor: Resistant, Resistant, ..., Resistant
##   levels: Resistant
## details: use 'details(object)'
```

Gene color sets can be used in the same way as gene sets, e.g., for subsetting expression sets (provided the map between identifiers is 1:1, so that coloring corresponding to identifiers can be determined). The `coloring` method allows access to the coloring information with a data frame interface; `phenotype`, `geneColor` and `phenotypeColor` provide additional accessors.

3 GeneSetCollection

A *GeneSetCollection* is a collection of gene sets. Sets in the collection must have distinct `setName`s, but can be a mix of *GeneSet* and *GeneColorSet*. Two convenient ways to create a gene set collection are by specifying a source of identifiers (e.g., an *ExpressionSet* or *AnnotationIdentifier*) and how the identifiers are to be induced into sets (e.g., by consulting the GO or KEGG ontologies):

```
gsc <- GeneSetCollection(sample.ExpressionSet[201:250,], setType=GOCollection())
gsc
```

```
## GeneSetCollection
##   names: GO:0000122, GO:0000209, ..., GO:1990837 (357 total)
##   unique identifiers: 31471_at, 31480_f_at, ..., 31477_at (31 total)
##   types in collection:
##     geneIdType: AnnotationIdentifier (1 total)
##     collectionType: GOCollection (1 total)
```

```
gsc[["GO:0005737"]]
```

```
## setName: GO:0005737
## geneIds: 31450_s_at, 31451_at, ..., 31489_at (total: 10)
## geneIdType: Annotation (hgu95av2)
## collectionType: GO
##   ids: GO:0005737 (1 total)
##   evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA ISM IGC IBA IBD IKR IRD RCA TAS NA
##   ontology: CC MF BP
## details: use 'details(object)'
```

In this example, the annotation identifiers of the sample expression set are organized into gene sets based on their presence in GO pathways. Providing arguments such as `evidenceCode` to *GOCollection* act to select just those pathways satisfying the GO collection constraint:

```
GeneSetCollection(sample.ExpressionSet[201:300,],
                  setType=GOCollection(evidenceCode="IMP"))
```

```
## GeneSetCollection
##   names: GO:0000122, GO:0000226, ..., GO:1902282 (120 total)
##   unique identifiers: 31520_at, 31489_at, ..., 31487_at (26 total)
##   types in collection:
##     geneIdType: AnnotationIdentifier (1 total)
##     collectionType: GOCollection (1 total)
```

Sets in the collection are named after the GO terms, and can be accessed by numeric index or name.

A file or url containing several gene sets defined by Broad XML can be used to create a gene set collection, e.g.,

```
  ## FIXME: BroadCollection default to paste("c", 1:4, sep="")
  ## FIXME: GeneSetCollection(BroadCollection(), urls=fl); filters on bcCategory
fl <- system.file("extdata", "Broad.xml", package="GSEABase")
gss <- getBroadSets(fl)
gss
```

```
## GeneSetCollection
##   names: chr5q23, chr16q24 (2 total)
##   unique identifiers: ZNF474, CCDC100, ..., TRAPPC2L (215 total)
##   types in collection:
##     geneIdType: SymbolIdentifier (1 total)
##     collectionType: BroadCollection (1 total)
```

```
names(gss)
```

```
## [1] "chr5q23"  "chr16q24"
```

Identifiers within a gene set collection can be mapped to a common type (provided maps are available) with, for example,

```
gsc <- mapIdentifiers(gsc, EntrezIdentifier())
gsc
```

```
## GeneSetCollection
##   names: GO:0000122, GO:0000209, ..., GO:1990837 (357 total)
##   unique identifiers: 641339, 4111, ..., 7033 (27 total)
##   types in collection:
##     geneIdType: EntrezIdentifier (1 total)
##     collectionType: GOCollection (1 total)
```

```
gsc[["GO:0005737"]]
```

```
## setName: GO:0005737
## geneIds: 6014, 392, ..., 4287 (total: 7)
## geneIdType: EntrezId (hgu95av2)
## collectionType: GO
##   ids: GO:0005737 (1 total)
##   evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA ISM IGC IBA IBD IKR IRD RCA TAS NA
##   ontology: CC MF BP
## details: use 'details(object)'
```

A convenient way to visualize a *GeneSetCollection* is with the *ReportingTools* package.

```
## 'interesting' gene sets
idx <- sapply(gsc, function(x) length(geneIds(x))) > 2

library(ReportingTools)
```

```
## Loading required package:  knitr
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
```

```
gscReport <- HTMLReport(
    shortName="gsc_example",
    title="GSEABase Vignette GeneSetCollection",
    basePath=tempdir())
publish(gsc[idx], gscReport, annotation.db="org.Hs.eg")
url <- finish(gscReport)
```

The report can be viewed with

```
browseURL(url)
```

This concludes a brief tour of gene sets and gene set collections available in the *GSEABase* package.