An Introduction to *GSEABase*

Martin Morgan

24 September, 2007

```
> library(GSEABase)
> library(hgu95av2.db)
> library(GO.db)
```

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 (see *Biobase* documentation for a description of expression sets) use

```
> data(sample.ExpressionSet)
> 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 expression set. These are accessible using `geneIds`, e.g.,

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

```
[1] "31440_at"   "31441_at"   "31442_at"   "31443_at"
[5] "31444_s_at" "31445_at"
```

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

```
> details(egs)
```

```
setName: Sample
geneIds: 31440_at, 31441_at, ..., 31489_at (total: 50)
geneIdType: Annotation (hgu95av2)
collectionType: ExpressionSet
setIdentifier: LIVERPOOL:8712:2010-12-15 03:06:47
description: Smoking-Cancer Experiment
  (longDescription available)
organism: Homo sapiens
pubMedIds:
urls: www.lab.not.exist
contributor: Pierre Fermat
setVersion: 0.0.1
creationDate: Wed Dec 15 03:06:47 2010
```

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`.

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"
 [3] "GenenameIdentifier"   "RefseqIdentifier"
 [5] "SymbolIdentifier"     "UnigeneIdentifier"
 [7] "ENSEMBLIdentifier"    "AnnotationIdentifier"
 [9] "EntrezIdentifier"     "GOAllFrameIdentifier"
[11] "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: 33)
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 using the `hgu95av2ENTREZID` environment in the `hgu95av2.db` (or `hgu95av2`) package. `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)
> eids <- unique(getEG(geneIds(egs), "hgu95av2"))
> eids <- eids[!is.na(eids)]
> GeneSet(EntrezIdentifier(), geneIds = as.character(eids))
```

```
setName: NA
geneIds: 6932, 643332, ..., 4287 (total: 33)
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: 39)
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:0000165, GO:0000186, ..., GO:0051219 (218 total)
  unique identifiers: 31445_at, 31468_f_at, ..., 31442_at (33 total)
  types in collection:
    geneIdType: AnnotationIdentifier (1 total)
    collectionType: GOCollection (1 total)

> gsc[["GO:0005737"]]

setName: GO:0005737
geneIds: 31445_at, 31451_at, ..., 31489_at (total: 10)
geneIdType: Annotation (hgu95av2)
collectionType: GO
  ids: GO:0005737 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP ISS ISO ISA ISM IGC RCA TAS NAS IC ND IEA
  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:0006412, GO:0008284, ..., GO:0045182 (17 total)
  unique identifiers: 31511_at, 31507_at, ..., 31532_at (11 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.,

```
> 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:0000165, GO:0000186, ..., GO:0051219 (218 total)
  unique identifiers: 6464, 2911, ..., 1048 (29 total)
  types in collection:
    geneIdType: EntrezIdentifier (1 total)
    collectionType: GOCollection (1 total)
```

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

```
setName: GO:0005737
geneIds: 6464, 392, ..., 4287 (total: 7)
geneIdType: EntrezId (hgu95av2)
collectionType: GO
  ids: GO:0005737 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP ISS ISO ISA ISM IGC RCA TAS NAS IC ND IEA
  ontology: CC MF BP
details: use 'details(object)'
```

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