Metadata in Bioconductor

Educational Materials

Overview

- Integrative biology - fuse information from biological research databases of many different types.
- In this lecture we focus attention on resources that can help to make use of metadata in different analyses.

Alternative Strategies

- The biomaRt package provides a set of tools that allow you to access online databases such as Ensembl, Vega, Uniprot, MSD, Wormbase.
- We have begun development of a new set of metadata packages that make use of a database (SQLite) rather than R's environments. This will result in smaller size, slightly slower access, but much more general queries.
- Make use of the chip manufacturer's resources (for Affymetrix this is the NetAffx site).

Per chip annotation

An early design decision was that we should provide metadata on a per chip-type basis.

```
> library("hgu133a")
> ls("package:hgu133a")
```

```
 [1] "hgu133a"                "hgu133aACCNUM"
 [3] "hgu133aCHR"             "hgu133aCHRLENGTHS"
 [5] "hgu133aCHRLOC"          "hgu133aENZYME"
 [7] "hgu133aENZYME2PROBE"    "hgu133aGENENAME"
 [9] "hgu133aGO"              "hgu133aGO2ALLPROBES"
[11] "hgu133aGO2PROBE"        "hgu133aLOCUSID"
[13] "hgu133aMAP"             "hgu133aMAPCOUNTS"
[15] "hgu133aOMIM"            "hgu133aORGANISM"
[17] "hgu133aPATH"            "hgu133aPATH2PROBE"
[19] "hgu133aPFAM"            "hgu133aPMID"
[21] "hgu133aPMID2PROBE"      "hgu133aPROSITE"
[23] "hgu133aQC"              "hgu133aQCDATA"
[25] "hgu133aREFSEQ"          "hgu133aSUMFUNC"
[27] "hgu133aSYMBOL"          "hgu133aUNIGENE"
```

A brief description

- These packages contain R `environments`, which are used as hash tables.
- For each package, data provenance information is provided, e. g. `? hgu133a`
- Quality control information is available, e. g. `hgu133a()`. This reports how many of each of the different types of mappings were found.

Accessing annotation packages

You can access the data directly using any of the standard subsetting or extraction tools for environments: `get`, `mget`, `$` and `[[`.

```
> get("201473_at", hgu133aSYMBOL)

[1] "JUNB"

> mget(c("201473_at", "201476_s_at"), hgu133aSYMBOL)

$`201473_at`
[1] "JUNB"

$`201476_s_at`
[1] "RRM1"

> hgu133aSYMBOL$"201473_at"

[1] "JUNB"

> hgu133aSYMBOL[["201473_at"]]

[1] "JUNB"
```

Metadata I

EntrezGene is a catalog of genetic loci that connects curated sequence information to official nomenclature. It replaced `LocusLink`.

UniGene defines sequence clusters. UniGene focuses on protein-coding genes of the nuclear genome (excluding rRNA and mitochondrial sequences).

RefSeq is a non-redundant set of transcripts and proteins of known genes for many species, including human, mouse and rat.

Enzyme Commission (EC) numbers are assigned to different enzymes and linked to genes through EntrezGene.

Metadata II

Gene Ontology (GO) is a structured vocabulary of terms describing gene products according to molecular function, biological process, or cellular component

PubMed is a service of the U.S. National Library of Medicine. PubMed provides a rich resource of data and tools for papers in journals related to medicine and health. While large, the data source is not comprehensive, and not all papers have been abstracted

LITDB curated by the Protein Research Foundation, covers all articles dealing with peptides from journals accessible in Japan

Metadata III

OMIM Online Mendelian Inheritance in Man is a catalog of human genes and genetic disorders.

NetAffx Affymetrix' NetAffx Analysis Center provides annotation resources for Affymetrix GeneChip technology.

KEGG Kyoto Encyclopedia of Genes and Genomes; a collection of data resources including a rich collection of pathway data.

cMAP Pathway data from both KEGG and BioCarta, in a computable form.

Metadata IV

Chromosomal Location Genes are identified with chromosomes, and where appropriate with strand.

Data Archives The NCBI coordinates the Gene Expression Omnibus (GEO); TIGR provides the Resourcerer database, and the EBI supports ArrayExpress.

Working with Metadata

Suppose we are interested in the gene BAD.

```
> gsyms <- unlist(as.list(hgu95av2SYMBOL))
> whBAD <- grep("^BAD$", gsyms)
> gsyms[whBAD]

1861_at
  "BAD"

> hgu95av2GENENAME$"1861_at"

[1] "BCL2-antagonist of cell death"
```

BAD Pathways

Find the pathways that BAD is associated with.

```
> BADpath <- hgu95av2PATH$"1861_at"
> mget(BADpath, KEGGPATHID2NAME)

$`01510`
[1] "Neurodegenerative Disorders"

$`04210`
[1] "Apoptosis"

$`04510`
[1] "Focal adhesion"

$`04910`
[1] "Insulin signaling pathway"

$`05030`
[1] "Amyotrophic lateral sclerosis (ALS)"
```

BAD Pathways

We can get the GeneChip probes and the unique EntrezGene loci in each of these pathways.

```
> allProbes <- mget(BADpath, hgu95av2PATH2PROBE)
> str(allProbes)

List of 5
 $ 01510: chr [1:63] "38974_at" "33831_at" "39334_s_at" "40489_at"
 $ 04210: chr [1:151] "40781_at" "32477_at" "31647_at" "35018_at"
 $ 04510: chr [1:324] "40781_at" "33814_at" "32477_at" "34042_at"
 $ 04910: chr [1:192] "40781_at" "40635_at" "40636_at" "37136_at"
 $ 05030: chr [1:29] "37033_s_at" "34336_at" "32512_at" "32513_at"

> getEG = function(x) unique(unlist(mget(x, hgu95av2LOCUSID)))
> allEG = sapply(allProbes, getEG)
> sapply(allEG, length)

01510 04210 04510 04910 05030
   33    87   189   127    16
```

Annotating a Genome

Bioconductor also provides some comprehensive annotations for whole genomes (e.g. *S. cerevisae*).
These packages are like the chip annotation packages, except a different set of primary keys is used (e.g. for yeast we use the systematic names such as `YBL088C`)

```
> library("YEAST")
> ls("package:YEAST")[1:12]
```

```
 [1] "YEAST"                    "YEASTALIAS"
 [3] "YEASTCHR"                 "YEASTCHRLENGTHS"
 [5] "YEASTCHRLOC"              "YEASTCOMMON2SYSTEMATIC"
 [7] "YEASTDESCRIPTION"         "YEASTENZYME"
 [9] "YEASTENZYME2PROBE"        "YEASTGENENAME"
[11] "YEASTGO"                  "YEASTGO2ALLPROBES"
```

The annotate package

- Functions for harvesting of curated persistent data sources
- functions for simple HTTP queries to web service providers
- interface code that provides common calling sequences for the assay based metadata packages such as `getGI` and `getSEQ` perform web queries to NCBI to extract the GI or nucleotide sequence corresponding to a GenBank accession number.

```
> ggi <- getGI("M22490")
> gsq <- getSEQ("M22490")

> ggi

[1] "179503"

> substring(gsq, 1, 40)

[1] "GGCAGAGGAGGAGGGAGGGAGGGAAGGAGCGCGGAGCCCG"
```

The annotate package

- other interface functions include `getGO`, `getSYMBOL`, `getPMID`, and `getLL`
- functions whose names start with `pm` work with lists of PubMed identifiers for journal articles.

```
> hgu95av2SYMBOL$"37809_at"

[1] "HOXA9"

> pm.getabst("37809_at", "hgu95av2")

$`37809_at`
$`37809_at`[[1]]
An object of class 'pubMedAbst':
Title: Vertebrate homeobox gene nomenclature.
PMID: 1358459
Authors: MP Scott
Journal: Cell
Date: Nov 1992
```

Working with GO

- An ontology is a structured vocabulary that characterizes some conceptual domain.
- The Gene Ontology (GO) Consortium defines three ontologies characterizing aspects of knowledge about genes and gene products.
- These ontologies are
  - molecular function (MF),
  - biological process (BP)
  - cellular component (CC).

GO

molecular function of a gene product is what it does at the biochemical level. This describes what the gene product can do, but without reference to where or when this activity actually occurs. Examples of functional terms include “enzyme," “transporter," or “ligand."
biological process is a biological objective to which the gene product contributes. There is often a temporal aspect to a biological process. Biological processes usually involve the transformation of a physical thing. The terms “DNA replication” or “signal transduction” describe general biological processes.
cellular component is a part of a cell that is a component of some larger object or structure. Examples of cellular components include “chromosome”, “nucleus” and “ribosome”.

GO Characteristics

	Number of Terms
BP	10765
CC	1733
MF	7686

Table 1: Number of GO terms per ontology.

GO hierarchy

GO terms can be linked by two relationships:

- is a: class-subclass relationship, for example, *nuclear chromosome* is a *chromosome*
- part of: C part of D means that when C is present, it is a part of D, but C does not always have to be present. For example, *nucleus* is part of *cell*.

The ontologies are structured as directed acyclic graphs.

DAGs are similar to hierarchies but a child term can have multiple parent terms. For example, the biological process term *hexose biosynthesis* has two parents, *hexose metabolism* and *monosaccharide biosynthesis*.

Working with GO

For precision and conciseness, all indexing of GO resources employs 7-digit tags with prefix `GO:` for example `GO:0008094`. Three basic tasks that are commonly performed in conjunction with GO are

- navigating the hierarchy, determining parents and children of selected terms, and deriving subgraphs of the overall DAG constituting GO;
- resolving the mapping from GO tag to natural language characterizations of function, location, or process;
- resolving the mapping between GO tags or terms and elements of catalogs of genes or gene products.

Navigating the hierarchy

- Finding parents and children of different terms is handled by using the `PARENT` and `CHILDREN` mappings.
- To find the children of `"GO:0008094"` we use:

```
> get("GO:0008094", GOMFCHILDREN)

[1] "GO:0003689" "GO:0015616" "GO:0043142" "GO:0004003"
```

- We use the term *offspring* to refer to all descendants (children, grandchildren, and so on) of a node.
- Similarly we use the term *ancestor* to refer to the parents, grandparents, and so on, of a node.

```
> get("GO:0008094", GOMFOFFSPRING)

[1] "GO:0003689" "GO:0015616" "GO:0043142" "GO:0004003"
[5] "GO:0017116" "GO:0008722" "GO:0043140" "GO:0043141"
```

GO terms

All GO terms are provided in the `GOTERM` environment.

```
> GOTERM$"GO:0002021"

GOID = GO:0002021
Term = response to dietary excess
Definition = The physiological process by which dietary
     excess is sensed by the central nervous system and
     results in a reduction in food intake and increased
     energy expenditure.
Ontology = BP
```

Searching for terms

Let's search for terms containing the word `chromosome` using `eapply` and `grep`.

```
> terms = eapply(GOTERM, Term)
> terms[[18]]

[1] "killing of cells of another organism"

> uterms = unlist(terms)
> re = regexpr("chromosome", uterms)
> chrTerms = uterms[re > 0]
> length(chrTerms)

[1] 75

> chrTerms[1]

                                                    GO:0009047
"dosage compensation, by hyperactivation of X chromosome"
```

Evidence Codes

The mapping of genes to GO terms is carried out by GOA, a project run by the European Bioinformatics Institute that aims to provide assignments of gene products to GO terms.
Four environments in the `GO` package address the association between EntrezGene sequence entries and GO terms:

`GOLOCUSID`	GO $\longrightarrow$ EntrezGene ID (non-redundant)
`GOALLOCUSID`	GO $\longrightarrow$ EntrezGene ID (incl. implied)
`GOLOCUSID2GO`	EntrezGene $\longrightarrow$ GO term
`GOLOCUSID2ALLGO`	EntrezGene $\longrightarrow$ GO term

GO Evidence Codes

```
      Abbreviation Definition
 [1,] "IMP"        "inferred from mutant phenotype"
 [2,] "IGI"        "inferred from genetic interaction"
 [3,] "IPI"        "inferred from physical interaction"
 [4,] "ISS"        "inferred from sequence similarity "
 [5,] "IDA"        "inferred from direct assay"
 [6,] "IEP"        "inferred from expression pattern"
 [7,] "IEA"        "inferred from electronic annotation"
 [8,] "TAS"        "traceable author statement"
 [9,] "NAS"        "non-traceable author statement"
[10,] "ND"         "no biological data available"
[11,] "IC"         "inferred by curator"
```

Find the GO identifier for “transcription factor binding” and use that to get EntrezGene ID with that annotation

```
> tfb <- names(which(uterms == "transcription factor binding"))
> gg1 <- get(tfb, GOLOCUSID)
> table(names(gg1))

IDA IEA IMP IPI ISS NAS  NR TAS
 17  51   1  28  30   4   1  30
```

Ontology

Consider the gene with EntrezGene ID 7355, SLC35A2

```
> z <- get("7355", GOLOCUSID2GO)
> length(z)

[1] 11

> sapply(z, "[[", "Ontology")

GO:0006012 GO:0008643 GO:0015780 GO:0015785 GO:0000139
      "BP"       "BP"       "BP"       "BP"       "CC"
GO:0005795 GO:0016020 GO:0016021 GO:0005338 GO:0005351
      "CC"       "CC"       "CC"       "MF"       "MF"
GO:0005459
      "MF"
```

there are 11 different GO terms. We get those from the BP ontology by using the helper function

```
> getOntology(z, "BP")

[1] "GO:0006012" "GO:0008643" "GO:0015780" "GO:0015785"
```

Evidence Codes

We get the evidence codes using `getEvidence` and can drop codes using `dropEcode`:

```
> getEvidence(z)

GO:0006012 GO:0008643 GO:0015780 GO:0015785 GO:0000139
     "TAS"      "IEA"      "IEA"      "TAS"      "IEA"
GO:0005795 GO:0016020 GO:0016021 GO:0005338 GO:0005351
     "IEA"      "IEA"      "IEA"      "IEA"      "IEA"
GO:0005459
     "TAS"

> zz <- dropECode(z, code = "IEA")
> getEvidence(zz)

GO:0006012 GO:0015785 GO:0005459
     "TAS"      "TAS"      "TAS"
```

GO graphs

For any set of selected genes, and any of the three GO ontologies the *induced GO graph* is the set of GO terms that the genes are associated with, together with all less specific terms.
The term “transcription factor activity” is in the molecular function (MF) ontology and has the GO label `GO:0003700`

```
> library("GO")
> library("GOstats")

> GOTERM$"GO:0003700"

GOID = GO:0003700
Term = transcription factor activity
Secondary = GO:0000130
Definition = Any activity required to initiate or
     regulate transcription; includes the actions of both
     gene regulatory proteins as well as the general
     transcription factors.
Ontology = MF
```

Induced GO graph

The induced graph, based on the MF hierarchy, can be produced using the `GOGraph` function of the package GOstats

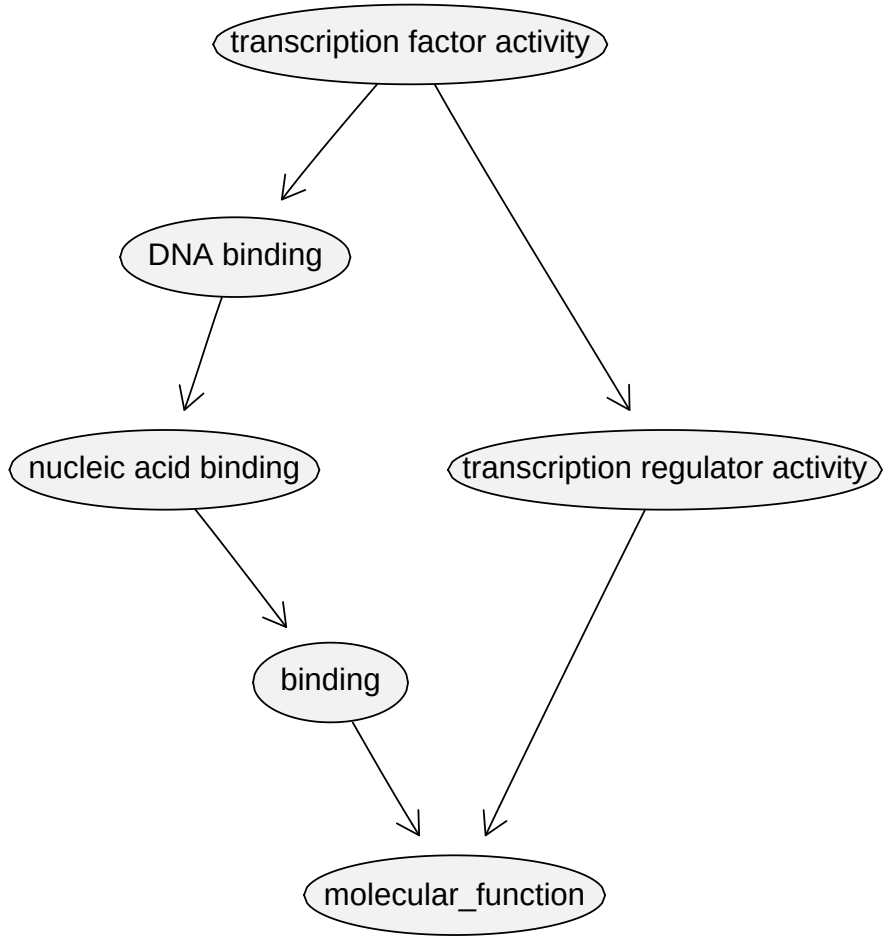
```
> tfG <- GOGraph("GO:0003700", GOMFPARENTS)
```

We can plot the induced GO graph using Rgraphviz and the code below.

```
> library("Rgraphviz")
> tfG = removeNode("all", tfG)
> mt = match(nodes(tfG), names(terms))
> stopifnot(!any(is.na(mt)))
> nattr <- makeNodeAttrs(tfG, label = terms[mt], shape = "ellipse
+     fillcolor = "#f2f2f2", fixedsize = FALSE)

> plot(tfG, nodeAttrs = nattr)
```

GO graph



GO relationships for term “transcription factor activity”.

Induced GO graphs

```
> tfch <- GOMFCHILDREN$"GO:0003700"

[1] "GO:0003705"

> tfchild <- mget(tfch, GOTERM)

$`GO:0003705`
GOID = GO:0003705
Term = RNA polymerase II transcription factor activity,
     enhancer binding
Definition = Functions to initiate or regulate RNA
     polymerase II transcription by binding a promoter or
     enhancer region of DNA.
Ontology = MF
```

KEGG

- KEGG provides mappings from genes to pathways
- We provide these in the package KEGG, you can also query the site directly using KEGGSOAP or other software.
- One problem with the KEGG is that the data is not in a form that is amenable to computation. The cMAP project provides data that is somewhat more useful for constructing networks.

Data in KEGG package

KEGGEXTID2PATHID provides mapping from either EntrezGene (for human, mouse and rat) or Open Reading Frame (yeast) to KEGG pathway ID. `KEGGPATHID2EXTID` contains the mapping in the other direction.

KEGGPATHID2NAME provides mapping from KEGG pathway ID to a textual description of the pathway. Only the numeric part of the KEGG pathway identifiers is used (not the three letter species codes)

Counts per species

	ath	dme	eat	hsa	mmu	rno	sce
Counts	111	118	84	172	168	160	100

Table 2: Pathway Counts Per Species

Exploring KEGG

Consider pathway 00362.

```
> KEGGPATHID2NAME$"00362"

[1] "Benzoate degradation via hydroxylation"
```

Species specific mapping from pathway to genes is indicated by glueing together three letter species code, e. g. textttthsa, and numeric pathway code.

```
> KEGGPATHID2EXTID$hsa00362

[1] "10449"  "1891"   "30"     "3032"   "347381" "59344"
[7] "83875"

> KEGGPATHID2EXTID$sce00362

[1] "YIL160C" "YKR009C"
```

Exploring KEGG

PAK1 has EntrezGene ID 5058 in humans

```
> KEGGEXTID2PATHID$"5058"

[1] "hsa04010" "hsa04360" "hsa04510" "hsa04650" "hsa04660"
[6] "hsa04810" "hsa05120"

> KEGGPATHID2NAME$"04010"

[1] "MAPK signaling pathway"
```

We find that it is involved in 323 pathways. For mice, the MAPK signaling pathway contains

```
> mm <- KEGGPATHID2EXTID$mmu04010
> str(mm)

 chr [1:266] "102626" "109689" "109880" "109905" ...
```

cMAP

The cancer Molecular Analysis Project (cMAP) provides software and molecular data relevant to cancer.
cMAP provides pathway data in a format that is amenable to computational manipulation.

```
> keggproc <- eapply(cMAPKEGGINTERACTION, "[[", "process")
> table(unlist(keggproc))

reaction
    4207

> cartaproc <- eapply(cMAPCARTAINTERACTION, "[[", "process")
> z = table(unlist(cartaproc))
> length(z)

[1] 121

> z[order(-z)[1:6]]

          modification          translocation          transcription
                  2123                    266                    236
           degradation              apoptosis protein ubiquitination
                    60                     41                     37
```

Homology

- Two genes are said to be homologous if they have descended from a common ancestral DNA sequence.
- There is one homology package for each species; a three letter species name (e. g. `hsa`) and suffix `homology`
- The current system is going to be changed and improved. For the time being, one possible alternative will be described in the biomaRt lecture.