9. Bioconductor: annotation databases

Thomas Lumley
Ken Rice

UW Biostatistics

August 2010, 北京

Outline

One goal of Bioconductor is to provide efficient access inside R to the genome databases that are vital to interpreting associations.

We will look at a few of these

- annotate
- biomaRt
- genomeGraphs

The reason to have an R interface to these databases is to be able to analyze annotation data for many SNPs or RNA transcripts.

Online or stored data

Annotation data can be downloaded in a single file or retrieved for each query from an online database.

Local storage is faster, but may require too much space (eg Ensembl) or become obsolete too quickly.

Local storage is ideal for fixed annotation data such as gene names for a microarray or SNP chip.

Types of database

Translations of names: Affy probe `32972_at` is the gene NADPH oxidase 1 with symbol `NOX1` and Ensembl gene id `ENSG00000007952`

Location: NOX1 is on Xq22.1, from 99984969 to 100015990, coded on the negative strand. There are 120 known polymorphisms (SNPs or indels) in this range.

Homology: The mouse version of NOX1 is also on the X chromosome, starting at 130621066.

Structure and function: `NOX1` is a membrane protein (location), involved in voltage-gated ion channel activity (molecular function), and involved in signal transduction (biological process).

Annotate

Bioconductor distributes annotation packages for a wide range of gene expression microarrays. The `annotate` package is one way to use this annotation information.

```
> library("annotate")
> library("hgu95av2.db")
> library("GO.db")
```

loads the `annotate` package and the databases for the Gene Ontology and one of the Affymetrix human microarray chips.

Lookups

The databases are queried with `get()` or `mget()` for multiple queries

```
> mget(c("738_at", "40840_at", "32972_at"), hgu95av2GENENAME)
$‘738_at‘
[1] "5’-nucleotidase, cytosolic II"

$‘40840_at‘
[1] "peptidylprolyl isomerase F (cyclophilin F)"

$‘32972_at‘
[1] "NADPH oxidase 1"

> go<-get("738_at", hgu95av2GO)
> names(go)
[1] "GO:0009117" "GO:0005829" "GO:0005737" "GO:0000166" "GO:0000287"
[6] "GO:0008253" "GO:0008253" "GO:0016787"
```

Lookups

```
> get("GO:0009117",GOTERM)
GOID: GO:0009117
Term: nucleotide metabolic process
Ontology: BP
Definition: The chemical reactions and pathways involving a nucleotic
    a nucleoside that is esterified with (ortho)phosphate or an
    oligophosphate at any hydroxyl group on the glycose moiety; may h
    mono-, di- or triphosphate; this definition includes cyclic
    nucleotides (nucleoside cyclic phosphates).
Synonym: nucleotide metabolism
```

BioMart

BioMart (www.biomart.org)is a query-oriented data management system developed jointly by the European Bioinformatics Institute (EBI) and Cold Spring Harbor Laboratory (CSHL).

`biomaRt` is an R interface to BioMart systems, in particular to Ensembl (www.ensembl.org). Ensembl is a joint project between EMBL - European Bioinformatics Institute (EBI) and the Wellcome Trust Sanger Institute (WTSI) to develop a software system which produces and maintains automatic annotation on selected eukaryotic genomes.

BioMart

We begin by choosing which BioMart to use

```
> library(biomaRt)
Loading required package: RCurl
> listMarts()
                          name                                    version
1                     ensembl                ENSEMBL 44 GENES (SANGER)
2     compara_mart_homology_44              ENSEMBL 44 HOMOLOGY (SANGER)
3  compara_mart_pairwise_ga_44 ENSEMBL 44 PAIRWISE ALIGNMENTS (SANGER)
4                         snp           ENSEMBL 44 VARIATION  (SANGER)
5                        vega                        VEGA 21  (SANGER)
6                     uniprot                UNIPROT PROTOTYPE (EBI)
7                         msd                    MSD PROTOTYPE (EBI)
8         ENSEMBL_MART_GRAMENE                         GRAMENE (CSHL)
9                       dicty               DICTYBASE (NORTHWESTERN)
10                    rgd_mart                         RGD GENES (MCW)
11                   SSLP_mart       RGD MICROSATELLITE MARKERS (MCW)
12          pepseekerGOLD_mart   PEPSEEKER (UNIVERSITY OF MANCHESTER)
13                      pride                             PRIDE (EBI)
> ens <- useMart("ensembl")
```

BioMart

We then choose a database to use

```
> listDatasets(ens)
                          dataset                                    description
1          oanatinus_gene_ensembl         Ornithorhynchus anatinus  genes (OANA5)
2         gaculeatus_gene_ensembl         Gasterosteus aculeatus genes (BROADS1)
3         cporcellus_gene_ensembl              Cavia porcellus genes (GUINEAPIG)
4          lafricana_gene_ensembl             Loxodonta africana genes (BROADE1)
...
12          hsapiens_gene_ensembl                    Homo sapiens genes (NCBI36)
...
37       cfamiliaris_gene_ensembl                Canis familiaris genes (BROADD2)

> ens <- useDataset("hsapiens_gene_ensembl",mart=ens)
```

BioMart

The `getGene` function queries the database for gene information. It accepts many forms of gene identifier, eg Entrez, HUGO, Affy transcript

```
> getGene(id=1440, type="entrezgene", mart=ens)
   entrezgene hgnc_symbol
1        1440        CSF3

1 Granulocyte colony-stimulating factor Precursor (G-CSF)(Pluripoietin)
(Filgrastim)(Lenograstim) [Source:UniProtKB/Swiss-Prot;Acc:P09919]
  chromosome_name  band strand start_position end_position ensembl_gene_id
1              17 q21.1      1       35425140     35427592 ENSG00000108342

>  getGene(id=c("AGT","AGTR1"), type="hgnc_symbol", mart=ens)
  hgnc_symbol hgnc_symbol
1       AGTR1       AGTR1
2         AGT         AGT
1
2 Angiotensinogen Precursor (Serpin A8) [Contains Angiotensin-1(Angiotensin I)
(Ang I);Angiotensin-2(Angiotensin II)(Ang II);Angiotensin-3(Angiotensin III)
(Ang III)(Des-Asp[1]-angiotensin II)] [Source:UniProtKB/Swiss-Prot;Acc:P01019]
  chromosome_name  band strand start_position end_position ensembl_gene_id
1               3   q24      1      149898348    149943478 ENSG00000144891
2               1 q42.2     -1      228904897    228916564 ENSG00000135744
```

BioMart

`getBM` is more general than `getGene`. It specifies a list of filters for selecting genes or SNPs and attributes to return from the database.

```
> affyids <- c("202763_at", "209310_s_at", "207500_at")
>  getBM(attributes = c("affy_hg_u133_plus_2", "hgnc_symbol", "chromosome_name",
    "start_position", "end_position", "band"), filters = "affy_hg_u133_plus_2",
     values = affyids, mart = ens)
     affy_hg_u133        hgnc   chromosome_name start_position end_position  band
1          202763_at       CASP3           4      185785844    185807623 q35.1
2          207500_at       CASP5          11      104370180    104384957 q22.3
3        209310_s_at       CASP4          11      104318804    104344535 q22.3
```

Homology

`getLDS()` combines two data marts, for example to homologous genes in other species. We can look up the mouse equivalents of a particular Affy transcript, or of the NOX1 gene.

```
> human = useMart("ensembl", dataset = "hsapiens_gene_ensembl")
> mouse = useMart("ensembl", dataset = "mmusculus_gene_ensembl")
> getLDS(attributes = c("hgnc_symbol","chromosome_name", "start_position"),
     filters = "hgnc_symbol",  values = "NOX1", mart = human,
     attributesL = c("chromosome_name","start_position"),
     martL = mouse)
    V1 V2       V3 V4        V5
1 NOX1  X 99984969  X 130621066
```

Homology

The `getSequence` function looks up DNA or protein sequences by chromosome position or gene identifiers

```
> agt<-getSequence(id="AGT",type="hgnc_symbol", seqType="peptide",mart=ens)
> agt

1 MRKRAPQSEMAPAGVSLRATILCLLAWAGLAAGDRVYIHPFHLVIHNESTCEQLAKANAGKPKDPTFIPAPIQAKTS
PVDEKALQDQLVLVAAKLDTEDKLRAAMVGMLANFLGFRIYGMHSELWGVVHGATVLSPTAVFGTLASLYLGALDHTAD
RLQAILGVPWKDKNCTSRLDAHKVLSALQAVQGLLVAQGRADSQAQLLLSTVVGVFTAPGLHLKQPFVQGLALYTPVVL
PRSLDFTELDVAAEKIDRFMQAVTGWKTGCSLMGASVDSTLAFNTYVHFQGKMKGFSLLAEPQEFWVDNSTSVSVPMLS
GMGTFQHWSDIQDNFSVTQVPFTESACLLLIQPHYASDLDKVEGLTFQQNSLNWMKKLSPRTIHLTMPQLVLQGSYDLQ
DLLAQAELPAILHTELNLQKLSNDRIRVGEVLNSIFFELEADEREPTESTQQLNKPEVLEVTLNRPFLFAVYDQSATAL
HFLGRVANPLSTA*
```

Example: finding chromosomes

We had a set 1524 SNPs, of which 409 did not have their chromosome listed.

I needed to know which SNPs were on the X chromosome, to estimate sex from DNA intensity and heterozygous X-chromosome loci, for QC.

```
> head(unknown)
[1] "UGT1A3-001449-0_B_R_1538822" "LIPC-002761-0_B_R_1538453"
[3] "CETP-001265-0_B_R_1538254"   "F8-165293-0_T_F_1538626"
[5] "CPB2-051208-0_B_F_1539402"   "VDRDIL-1355-0_T_F_1539404"
```

A hand-search would be easy but tedious, so we want an automated approach

Example: finding chromosomes

First extract the gene names

```
genes <- sapply(unknown, function(snp) strsplit(snp,"-")[[1]][1])
ugenes <- unique(genes)
```

Now call out to Ensembl

```
getBM(attributes="chromosome_name", filters="hgnc_symbol",values=ugenes,
  mart=ensembl)
```

works for all except VRDIL, which isn't recognized.

Finding SNPs

Human SNPs are in a separate database from gene information. We can look up known SNPs and other polymorphisms for the NOX1 gene

```
> snpmart = useMart("snp", dataset = "hsapiens_snp")
Checking attributes ... ok
Checking filters ... ok
> getBM(c("refsnp_id", "allele", "chrom_start", "chrom_strand"),
      filters = c("chr_name", "chrom_start","chrom_end"),
      values = list("X",99984969,100015990), mart = snpmart)
        refsnp_id             allele chrom_start chrom_strand
1       rs35477500              AAG/-    99985012            1
2       rs41310727                G/T    99985014            1
3       rs61639376              GAA/-    99985014            1
4       rs34181451                G/A    99985489            1
5       rs34009592                G/A    99985531            1
```

GenomeGraphs

This package makes pretty pictures from the annotation data.

For example, a pictures showing the standard and alternative splices for the NOX1 gene and the location of the gene on the X chromosome

```
> library(GenomeGraphs)
> gene <- makeGene(id = "NOX1", type = "hgnc_symbol",
     biomart = ensembl)
> transcript<-makeTranscript(id="NOX1",type="hgnc_symbol",
     biomart=ensembl)
> ideogram <- makeIdeogram(chromosome ="X")
> gdPlot(list(ideogram,gene,transcript))
```

GenomeGraphs