Bioconductor: annotation databases

Thomas Lumley
Ken Rice

UW Biostatistics

Liège, September 2007
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

- RSNPper
- biomaRt
- goTools and GOstats.

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

This is an interface to the SNPper service, part of the Children’s Hospital Informatics Program (CHIP) at Boston Children’s Hospital.

There are five basic functions

- `geneInfo()` information on a gene: location, name, coding strand, id in various databases
- `geneLayout()` information on exon locations
- `geneSNPs()` known SNPs in a gene
- `SNPinfo()` location, alleles, amino acid alleles, dbSNP id.
- `itemsInRange()` genes, SNPs, or counts of SNPs in segment of chromosome.
Example: Angiotensinogen

> geneInfo("AGT")
SNPper Gene metadata:
There are 1 entries.
Basic information:
    GENEID  NAME  CHROM  STRAND
    1       AGT    chr1   -

1 angiotensinogen (serine (or cysteine) proteinase inhibitor, clade A (alpha-1 antiproteinase, antitrypsin), member 8)

    NSNPS   TX.START   TX.END   CODSEQ.START   CODSEQ.END   LOCUSLINK   OMIM   UNIGENE
    1       211       227145020 227156602       227145622       227153331       183   106150   Hs.19383

    SWISSPROT  MRNAACC  PROTACC  REFSEQACC
    1       P01019   NM_000029  NP_000020  NULL

SNPper info:
    SOURCE    VERSION    GENOME    DBSNP
[1,] "*RPCSERV-NAME*" "$Revision: 1.38 "$ "hg17" "123"

[Note that the output also includes build numbers for dbSNP and the Human Genome.]

The ID number for angiotensinogen is 1986, which is the key for other queries
Example: Angiotensinogen

> geneLayout(1986)

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>CHROM</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot; &quot;</td>
<td>&quot;AGT&quot;</td>
<td>&quot;chr1&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TRANSSCRIPT.START</th>
<th>CODINGSEQ.START</th>
<th>TRANSSCRIPT.END</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;227145020&quot;</td>
<td>&quot;227145622&quot;</td>
<td>&quot;227156602&quot;</td>
</tr>
<tr>
<td>CODINGSEQ.END</td>
<td>exon1.start</td>
<td>exon1.end</td>
</tr>
<tr>
<td>&quot;227153331&quot;</td>
<td>&quot;227145020&quot;</td>
<td>&quot;227145810&quot;</td>
</tr>
<tr>
<td>exon2.start</td>
<td>exon2.end</td>
<td>exon3.start</td>
</tr>
<tr>
<td>&quot;227146674&quot;</td>
<td>&quot;227146818&quot;</td>
<td>&quot;227148414&quot;</td>
</tr>
<tr>
<td>exon3.end</td>
<td>exon4.start</td>
<td>exon4.end</td>
</tr>
<tr>
<td>&quot;227148681&quot;</td>
<td>&quot;227152476&quot;</td>
<td>&quot;227153334&quot;</td>
</tr>
<tr>
<td>exon5.start</td>
<td>exon5.end</td>
<td></td>
</tr>
<tr>
<td>&quot;227156567&quot;</td>
<td>&quot;227156602&quot;</td>
<td></td>
</tr>
</tbody>
</table>

attr(,"toolInfo")

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>VERSION</th>
<th>GENOME</th>
<th>DBSNP</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;<em>RPCSERV-NAME</em>&quot;</td>
<td>&quot;$Revision: 1.38 &quot;$</td>
<td>&quot;hg17&quot;</td>
<td>&quot;123&quot;</td>
</tr>
</tbody>
</table>
Example: Angiotensinogen

> agtsnps<-geneSNPs(1986)
> length(agtsnps)
[1] 213
> agtsnps[[1]]
TSCID " "
CHROMOSOME "chr1"
POSITION "227135034"
ALLELES "A/G/T"
ROLE "UTR"
RELPOS "18297"
AMINO " "
AMINOPOS " "
HUGO "AGT"
LOCUSLINK "183"
NAME "angiotensinogen (serine (or cysteine) proteinase inhibitor, mRNA "NM_000029"
Example: Angiotensinogen

> itemsInRange("genes", "chr1", "227140000", "227160000")[[1]][-3]
   NAME   CHROM   NSNPS
   "AGT"   "chr1"   "211"

> itemsInRange("countsnps", "chr1", "227140000", "227160000")
   total exonic nonsyn
       149   30   11

For some SNPs there is additional information available from the SNPinfo function

> b<-SNPinfo("372")
> b

SNP per SNP metadata:
   DBSNPID  CHROMOSOME  POSITION  ALLELES  VALIDATED
[1,] "rs372"  "chr13"  "31383542"  "A/G"  "Y"

There are details on 4 populations and 1 connections to gene features

> popDetails(b)

   PANEL    SIZE MAJOR.ALLELE MINOR.ALLELE  majorf  minorf
   1  Yoruba-30-trios illumina   A     G  0.883333  0.116667
   2       Japanese illumina     A     G  0.977273  0.0227273
   3  Han_Chinese illumina        A     G  0.955556  0.0444444
   4    CEPH-30-trios illumina    A     G  0.966667  0.0333333
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

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

Now call SNPper

```r
library(RSNPper)
chroms <- sapply(ugenes,  
    function(gene) geneInfo(gene, useOldOutput=TRUE)["CHROM"]
```

Works for all except one gene, where the name VDRDIL wasn't recognized
Under the hood

SNPper responds to URLs like http://snpper.chip.org/bio/rpcserv/dummy?cmd=geneinfo&name=CRP with XML (structured text) format descriptions of the gene.

RSNPper downloads the information in the same way that read.table() downloads data from a web page, and then uses the XML package to process the information.
Under the hood

```xml
> useSNPper("geneinfo&","name=CRP")
<SNPPER-RPC SOURCE="*RPCSERV-NAME*" VERSION="$Revision: 1.38 $"
GENOME="hg17" DBSNP="123">
<GENEINFO>
  <GENE ID="1440">
    <GENEID>1440</GENEID>
    <NAME>CRP</NAME>
    <CHROM>chr1</CHROM>
    <STRAND>--
  </GENE>
</GENEINFO>
```
Under the hood

C-reactive protein, pentraxin-related

TRANSCRIPT
  START=156495525
  END=156497437

CODINGSEQ
  START=156496388
  END=156497348

ACCESSION
  MRNAACC=NМ_000567
  PROTACC=NP_000558
  REFSEQACC START="NIL" END="NIL"

LINKS
  LOCUSLINK=1401
  OMIM=123260
  UNIGENE=Hs.76452
  SWISSPROT=P02741

NSNPS=101

GENE
  GENEINFO

SNPPER-RPC
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.
We begin by choosing which BioMart to use

```r
> library(biomaRt)
Loading required package: RCurl
> listMarts()

<table>
<thead>
<tr>
<th>name</th>
<th>version</th>
</tr>
</thead>
<tbody>
<tr>
<td>ensembl</td>
<td>ENSEMBL 44 GENES (SANGER)</td>
</tr>
<tr>
<td>compara_mart_homology_44</td>
<td>ENSEMBL 44 HOMOLOGY (SANGER)</td>
</tr>
<tr>
<td>compara_mart_pairwise_ga_44</td>
<td>ENSEMBL 44 PAIRWISE ALIGNMENTS (SANGER)</td>
</tr>
<tr>
<td>snp</td>
<td>ENSEMBL 44 VARIATION (SANGER)</td>
</tr>
<tr>
<td>vega</td>
<td>VEGA 21 (SANGER)</td>
</tr>
<tr>
<td>uniprot</td>
<td>UNIPROT PROTOTYPE (EBI)</td>
</tr>
<tr>
<td>msd</td>
<td>MSD PROTOTYPE (EBI)</td>
</tr>
<tr>
<td>ENSEMBL_MART_GRAMENE</td>
<td>GRAMENE (CSHL)</td>
</tr>
<tr>
<td>dicty</td>
<td>DICTYBASE (NORTHWESTERN)</td>
</tr>
<tr>
<td>rgd_mart</td>
<td>RGD GENES (MCW)</td>
</tr>
<tr>
<td>SSLP_mart</td>
<td>RGD MICROSATELLITE MARKERS (MCW)</td>
</tr>
<tr>
<td>pepseekerGOLD_mart</td>
<td>PEPSEEKER (UNIVERSITY OF MANCHESTER)</td>
</tr>
<tr>
<td>pride</td>
<td>PRIDE (EBI)</td>
</tr>
</tbody>
</table>

> ens <- useMart("ensembl")
```
BioMart

We then choose a database to use

> listDatasets(ens)

<table>
<thead>
<tr>
<th>dataset</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oanatinus_gene_ensembl</td>
<td>Ornithorhynchus anatinus genes (OANA5)</td>
</tr>
<tr>
<td>gaculeatus_gene_ensembl</td>
<td>Gasterosteus aculeatus genes (BROADS1)</td>
</tr>
<tr>
<td>cporcellus_gene_ensembl</td>
<td>Cavia porcellus genes (GUINEAPIG)</td>
</tr>
<tr>
<td>lafricana_gene_ensembl</td>
<td>Loxodonta africana genes (BROADE1)</td>
</tr>
<tr>
<td>hsapiens_gene_ensembl</td>
<td>Homo sapiens genes (NCBI36)</td>
</tr>
<tr>
<td>cfamiliaris_gene_ensembl</td>
<td>Canis familiaris genes (BROADD2)</td>
</tr>
</tbody>
</table>

> 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

```r
> getGene(id=1440, type="entrezgene", mart=ens)  
     entrezgene  hgnc_symbol
   1       1440          CSF3
   2       1440          CSF3
```

1 Granulocyte colony-stimulating factor precursor (G-CSF) (Pluripoietin) (Filgrastim)
2 Granulocyte colony-stimulating factor precursor (G-CSF) (Pluripoietin) (Filgrastim)

<table>
<thead>
<tr>
<th>chromosome_name</th>
<th>band</th>
<th>strand</th>
<th>start_position</th>
<th>end_position</th>
<th>ensembl_gene_id</th>
<th>ensembl_transcript_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>q21.1</td>
<td>1</td>
<td>35425214</td>
<td>35427592</td>
<td>ENSG00000108342</td>
<td>ENST00000225474</td>
</tr>
<tr>
<td>17</td>
<td>q21.1</td>
<td>1</td>
<td>35425214</td>
<td>35427592</td>
<td>ENSG00000108342</td>
<td>ENST00000331769</td>
</tr>
</tbody>
</table>

```r
> getGene(id=c("AGT","AGTR1"), type="hgnc_symbol", mart=ens)  
     hgnc_symbol  hgnc_symbol
   1          AGTR1        AGTR1
   2           AGT         AGT
```

1 Type-1 angiotensin II receptor (AT1) (AT1AR) (AT1BR). [Source:Uniprot/SWISSPROT;]
2 Angiotensinogen precursor [Contains: Angiotensin-1 (Angiotensin I) (Ang I); Angi
(Angiotensin II) (Ang II); Angiotensin-3 (Angiotensin III) (Ang III) (Des-Asp[1]-angiotensin II).

[Source: Uniprot/SWISSPROT; Acc: P01019]

<table>
<thead>
<tr>
<th>chromosome_name</th>
<th>band</th>
<th>strand</th>
<th>start_position</th>
<th>end_position</th>
<th>ensembl_gene_id</th>
<th>ensembl_transcript_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>149898355</td>
<td>149943478</td>
<td>ENSG00000144891</td>
<td>ENST00000349243</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-1</td>
<td>228904892</td>
<td>228916666</td>
<td>ENSG00000135744</td>
<td>ENST00000366667</td>
</tr>
</tbody>
</table>
BioMart

getHomolog() finds homologous genes in other species. For example, we can look up the mouse equivalents of a particular Affy transcript, or of the AGT gene.

```r
> mouse = useMart("ensembl","mmusculus_gene_ensembl")
> homolog = getHomolog( id = "1939_at", to.type = "affy_mouse430_2",
>                      from.type = "affy_hg_u95av2", from.mart = ens, to.mart = mouse )
> homolog
   V1      V2
1 1939_at 1426538_a_at
2 1939_at 1427739_a_at
> homolog2 = getHomolog( id = "AGT", to.type = "affy_mouse430_2",
>                       from.type = "hgnc_symbol", from.mart = ens, to.mart = mouse )
> homolog2
   V1      V2
1 AGT 1423396_at
```
The `getSequence` function looks up DNA or protein sequences by chromosome position or gene identifiers

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

1 MRKRAPQSEMAGVSLRATILCLLLAWAGLAAGDRVYIHPFHHLVHNESTCEQLAKANAGKPDKPDTFIFPAPIQAKTSPVDEKALQDQLVLVAAKLDTEDKLRAAMVGMLANFLGFRYGMHSELWGVVHGATVLSPTAVFGTLASLYLGALDHTADRLQAILGVPWKDKNCTSRDLAHKVLSAALGAVQGGLLVAAQGRADSQAQLLLSTVGVFAPGLHLKQPFVQGLALYTPVVLPRSLDFTELDVAAEKMDFQAVTGWKTGCSLMGASVDSLAFNTYVHFQGKMKGFSLLLAEFPQEFWDNSTSVSVPMLS
GMGTFQHWSDIQDNSVTQVPFTESAACLQLIQPHYASDLKVEGLTFQQNSLNNMKKLSPRTIHLTMPQLVQLQGYSYDLQDLLAQAELPAIHTELNLKLSNDRIRVGEVLNSIFFELEADEREPTQQLNKPEVLEVTLNRPFLFAVYDQSATALHFLGRVANPLSTA*
Gene Ontology

The GO project has developed three structured controlled vocabularies (ontologies) that describe gene products in terms of their associated biological processes, cellular components and molecular functions in a species-independent manner. (http://www.geneontology.org/)

Each gene will be in nested categories of increasing specificity, and may be in more than one set of categories (structure is a 'directed acyclic graph').

Several Bioconductor packages allow queries by GO labels (eg biomaRt) and others provide further analyses based on GO categories.
Gene Ontology

Using biomaRt, we can find the genes in a particular category, eg MAP kinase activity

```r
> mapk <- getFeature(type = "entrezgene", GOID = "GO:0004707", mart = ens)
> mapk
   go  entrezgene
  1 GO:0004707     5594
  2 GO:0004707     5596
  3 GO:0004707     5597
  4 GO:0004707     6300
  5 GO:0004707     5600
  6 GO:0004707     5595
  7 GO:0004707     5602
  8 GO:0004707     5598
 10 GO:0004707     51701
 12 GO:0004707     225689
 13 GO:0004707     5601
 14 GO:0004707     5599
 16 GO:0004707     1432
 17 GO:0004707     5603
```
Gene Ontology

goTools::ontoCompare takes lists of genes and looks up their GO categories. It can report categories for a single list of genes or compare categories for multiple lists.

The GOstats package does a more sophisticated test for whether a list of genes is overrepresented in certain GO categories.

This example is taken from the goTools package:

```r
> library(goTools)
> data(ProbeID)
> str(operonlist)
List of 2
$ L1: chr [1:30] "H200000481" "H200012124" "H200016088" "H200001913" ...
$ L2: chr [1:85] "H200018146" "H200019124" "H200008091" "H200004721" ...

> ontoCompare(operonlist,probeType="operon",plot=TRUE,goType="MF")

binding catalytic activity enzyme regulator activity
L1 0.55000 0.15000 0.15
L2 0.63158 0.47368 0.00
```
Gene Ontology

motor activity signal transducer activity
L1  0.100000  0.250000
L2  0.035088  0.052632

structural molecule activity
L1  0.050000
L2  0.017544

translation regulator activity transporter activity
L1  0.050000  0.100000
L2  0.017544  0.052632

transcription regulator activity NotFound
L1  0.000000  0.050000
L2  0.08772  0.15789