Practical: Annotation

Martin Morgan (mtmorgan@fhcrc.org)

June 26, 2014

Contents

1 Gene annotation

1.1 Data packages

Organism-level ('org') packages contain mappings between a central identifier (e.g., Entrez gene ids) and other identifiers (e.g. GenBank or Uniprot accession number, RefSeq id, etc.). The name of an org package is always of the form `org.<Sp>.<id>.db` (e.g. *org.Sc.sgd.db*) where `<Sp>` is a 2-letter abbreviation of the organism (e.g. `Sc` for *Saccharomyces cerevisiae*) and `<id>` is an abbreviation (in lower-case) describing the type of central identifier (e.g. `sgd` for gene identifiers assigned by the *Saccharomyces* Genome Database, or `eg` for Entrez gene ids). The "How to use the '.db' annotation packages" vignette in the *AnnotationDbi* package (org packages are only one type of ".db" annotation packages) is a key reference. The '.db' and most other *Bioconductor* annotation packages are updated every 6 months.

Annotation packages usually contain an object named after the package itself. These objects are collectively called *AnnotationDb* objects, with more specific classes named *OrgDb*, *ChipDb* or *TranscriptDb* objects. Methods that can be applied to these objects include `cols`, `keys`, `keytypes` and `select`. Common operations for retrieving annotations are summarized in Table 1.

Exercise 1
This exercise illustrates basic use of the 'select' interface to annotation packages.

a. *What is the name of the org package for* Homo sapiens*? Load it. Display the OrgDb object for the org.Hs.eg.db package. Use the `columns` method to discover which sorts of annotations can be extracted from it.*
b. *Use the `keys` method to extract ENSEMBL identifiers and then pass those keys in to the `select` method in such a way that you extract the SYMBOL (gene symbol) and GENENAME information for each. Use the following ENSEMBL ids.*

```
ensids <- c("ENSG00000130720", "ENSG00000103257", "ENSG00000156414",
            "ENSG00000144644", "ENSG00000159307", "ENSG00000144485")
```

Solution: The *OrgDb* object is named `org.Hs.eg.db`.

Table 1: Common operations for retrieving and manipulating annotations.

Category	Function	Description
Discover	`columns`	List the kinds of columns that can be returned
	`keytypes`	List columns that can be used as keys
	`keys`	List values that can be expected for a given keytype
	`select`	Retrieve annotations matching `keys`, `keytype` and `columns`
Manipulate	`setdiff`, `union`, `intersect`	Operations on sets
	`duplicated`, `unique`	Mark or remove duplicates
	`%in%`, `match`	Find matches
	`any`, `all`	Are any `TRUE`? Are all?
	`merge`	Combine two different `data.frames` based on shared keys
*GRanges**	`transcripts`, `exons`, `cds`	Features (transcripts, exons, coding sequence) as *GRanges*.
	`transcriptsBy` , `exonsBy` `cdsBy`	Features group by gene, transcript, etc., as *GRangesList*.

```
library(org.Hs.eg.db)
keytypes(org.Hs.eg.db)
```

```
##  [1] "ENTREZID"     "PFAM"         "IPI"          "PROSITE"      "ACCNUM"
##  [6] "ALIAS"        "CHR"          "CHRLOC"       "CHRLOCEND"    "ENZYME"
## [11] "MAP"          "PATH"         "PMID"         "REFSEQ"       "SYMBOL"
## [16] "UNIGENE"      "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS" "GENENAME"
## [21] "UNIPROT"      "GO"           "EVIDENCE"     "ONTOLOGY"     "GOALL"
## [26] "EVIDENCEALL"  "ONTOLOGYALL"  "OMIM"         "UCSCKG"
```

```
columns(org.Hs.eg.db)
```

```
##  [1] "ENTREZID"     "PFAM"         "IPI"          "PROSITE"      "ACCNUM"
##  [6] "ALIAS"        "CHR"          "CHRLOC"       "CHRLOCEND"    "ENZYME"
## [11] "MAP"          "PATH"         "PMID"         "REFSEQ"       "SYMBOL"
## [16] "UNIGENE"      "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS" "GENENAME"
## [21] "UNIPROT"      "GO"           "EVIDENCE"     "ONTOLOGY"     "GOALL"
## [26] "EVIDENCEALL"  "ONTOLOGYALL"  "OMIM"         "UCSCKG"
```

```
cols <- c("SYMBOL", "GENENAME")
select(org.Hs.eg.db, keys=ensids, columns=cols, keytype="ENSEMBL")
```

```
##           ENSEMBL SYMBOL
## 1 ENSG00000130720 FIBCD1
## 2 ENSG00000103257 SLC7A5
## 3 ENSG00000156414  TDRD9
## 4 ENSG00000144644  GADL1
## 5 ENSG00000159307 SCUBE1
## 6 ENSG00000144485   HES6
##                                                                        GENENAME
## 1                                                fibrinogen C domain containing 1
## 2 solute carrier family 7 (amino acid transporter light chain, L system), member 5
## 3                                                      tudor domain containing 9
## 4                                                glutamate decarboxylase-like 1
## 5                                          signal peptide, CUB domain, EGF-like 1
## 6                                          hes family bHLH transcription factor 6
```

Table 2: Selected packages querying web-based annotation services.

Package	Description
AnnotationHub	Ensembl, Encode, dbSNP, UCSC data objects
biomaRt	http://biomart.org, Ensembl and other annotations
PSICQUIC	https://code.google.com/p/psicquic.org, protein interactions
uniprot.ws	http://uniprot.org, protein annotations
KEGGREST	http://www.genome.jp/kegg, KEGG pathways
SRAdb	http://www.ncbi.nlm.nih.gov/sra, sequencing experiments.
rtracklayer	http://genome.ucsc.edu, genome tracks.
GEOquery	http://www.ncbi.nlm.nih.gov/geo/, array and other data
ArrayExpress	http://www.ebi.ac.uk/arrayexpress/, array and other data

1.2 Internet resources

A short summary of select *Bioconductor* packages enabling web-based queries is in Table 2.

Using biomaRt The *biomaRt* package offers access to the online biomart resource. this consists of several data base resources, referred to as 'marts'. Each mart allows access to multiple data sets; the *biomaRt* package provides methods for mart and data set discovery, and a standard method `getBM` to retrieve data.

Exercise 2
warning: This exericse requires INTERNET ACCESS

a. *Load the biomaRt package and list the available marts. Choose the* ensembl *mart and list the datasets for that mart. Set up a mart to use the* ensembl *mart and the* hsapiens_gene_ensembl *dataset.*
b. *A biomaRt dataset can be accessed via `getBM`. In addition to the mart to be accessed, this function takes filters and attributes as arguments. Use `filterOptions` and `listAttributes` to discover values for these arguments. Call `getBM` using filters and attributes of your choosing.*

Solution:

```
## NEEDS INTERNET ACCESS !!
library(biomaRt)
head(listMarts(), 3)                        ## list the marts
head(listDatasets(useMart("ensembl")), 3) ## mart datasets
ensembl <-                                  ## fully specified mart
    useMart("ensembl", dataset = "hsapiens_gene_ensembl")

head(listFilters(ensembl), 3)               ## filters
myFilter <- "chromosome_name"
substr(filterOptions(myFilter, ensembl), 1, 50) ## return values
myValues <- c("21", "22")
head(listAttributes(ensembl), 3)            ## attributes
myAttributes <- c("ensembl_gene_id","chromosome_name")

## assemble and query the mart
res <- getBM(attributes =  myAttributes, filters =  myFilter,
             values =  myValues, mart = ensembl)
```

Use `head(res)` to see the results.

Exercise 3
As an optional exercise, annotate the genes that are differentially expressed in the DESeq2 laboratory, e.g., find the `GENENAME` *associated with the five most differentially expressed genes. Do these make biological sense? Can you* `merge` *the annotation results with the 'top table' results to provide a statistically and biologically informative summary?*

2 Genome annotation

There are a diversity of packages and classes available for representing large genomes. Several include:

TxDb.* For transcript and other genome / coordinate annotation.
BSgenome For whole-genome representation. See `available.packages` for pre-packaged genomes, and the vignette 'How to forge a BSgenome data package' in the
Homo.sapiens For integrating *TxDb** and *org.** packages.
SNPlocs.* For model organism SNP locations derived from dbSNP.
`FaFile` (*Rsamtools*) for accessing indexed FASTA files.
SIFT.*, PolyPhen, ensemblVEP Variant effect scores.

2.1 Transcript annotation packages

Genome-centric packages are very useful for annotations involving genomic coordinates. It is straight-forward, for instance, to discover the coordinates of coding sequences in regions of interest, and from these retrieve corresponding DNA or protein coding sequences. Other examples of the types of operations that are easy to perform with genome-centric annotations include defining regions of interest for counting aligned reads in RNA-seq experiments and retrieving DNA sequences underlying regions of interest in ChIP-seq analysis, e.g., for motif characterization.

Exercise 4
This exercise uses annotation resources to go from a gene symbol 'BRCA1' through to the genomic coordinates of each transcript associated with the gene, and finally to the DNA sequences of the transcripts.

a. *Use the org.Hs.eg.db package to map from the gene symbol 'BRCA1' to its Entrez identifier. Do this using the* `select` *command.*
b. *Use the TxDb.Hsapiens.UCSC.hg19.knownGene package to retrieve the transcript names (*`TXNAME`*) corresponding to the BRCA1 Entrez identifier. (The org* packages are based on information from NCBI, where Entrez identifiers are labeled ENTREZID; the TxDb* package we are using is from UCSC, where Entrez identifiers are labelled GENEID).*
c. *Use the* `cdsBy` *function to retrieve the genomic coordinates of all coding sequences grouped by transcript, and select the transcripts corresponding to the identifiers we're interested in. The coding sequences are returned as an GRangesList, where each element of the list is a GRanges object representing the exons in the coding sequence. As a sanity check, ensure that the sum of the widths of the exons in each coding sequence is evenly divisble by 3 (the R 'modulus' operator %% returns the remainder of the division of one number by another, and might be helpful in this case).*
d. *Use the Bsgenome.Hsapiens.UCSC.hg19 package and* `extractTranscriptSeqs` *function to extract the DNA sequence of each transcript.*

Solution: Retrieve the Entrez identifier corresponding to the BRCA1 gene symbol

```
library(org.Hs.eg.db)
eid <- select(org.Hs.eg.db, "BRCA1", "ENTREZID", "SYMBOL")[["ENTREZID"]]
```

Map from Entrez gene identifier to transcript name

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
txid <- select(txdb, eid, "TXNAME", "GENEID")[["TXNAME"]]
```

```
## Warning:  'select' resulted in 1:many mapping between keys and return rows
```

Retrieve all coding sequences grouped by transcript, and select those matching the transcript ids of interest, verifying that each coding sequence width is a multiple of 3

```
cds <- cdsBy(txdb, by="tx", use.names=TRUE)
brca1cds <- cds[names(cds) %in% txid]
class(brca1cds)
```

```
## [1] "GRangesList"
## attr(,"package")
## [1] "GenomicRanges"
```

```
length(brca1cds)
```

```
## [1] 20
```

```
brca1cds[[1]]                          # exons in cds
```

```
## GRanges with 22 ranges and 3 metadata columns:
##        seqnames               ranges strand   |    cds_id    cds_name exon_rank
##           <Rle>            <IRanges>  <Rle>   | <integer> <character> <integer>
##    [1]    chr17 [41276034, 41276113]      -   |    186246        <NA>         1
##    [2]    chr17 [41267743, 41267796]      -   |    186245        <NA>         2
##    [3]    chr17 [41258473, 41258550]      -   |    186243        <NA>         3
##    [4]    chr17 [41256885, 41256973]      -   |    186241        <NA>         4
##    [5]    chr17 [41256139, 41256278]      -   |    186240        <NA>         5
##    ...      ...                  ...    ... ...       ...         ...       ...
##   [18]    chr17 [41209069, 41209152]      -   |    186218        <NA>        18
##   [19]    chr17 [41203080, 41203134]      -   |    186217        <NA>        19
##   [20]    chr17 [41201138, 41201211]      -   |    186215        <NA>        20
##   [21]    chr17 [41199660, 41199720]      -   |    186214        <NA>        21
##   [22]    chr17 [41197695, 41197819]      -   |    186212        <NA>        22
##   ---
##   seqlengths:
##                     chr1                  chr2 ...        chrUn_gl000249
##                249250621             243199373 ...                 38502
```

```
cdswidth <- width(brca1cds)            # width of each exon
all((sum(cdswidth) %% 3) == 0)         # sum within cds, modulus 3
```

```
## [1] TRUE
```

Extract the coding sequences of each transcript

```
library(BSgenome.Hsapiens.UCSC.hg19)
genome <- BSgenome.Hsapiens.UCSC.hg19
tx_seq <- extractTranscriptSeqs(genome, brca1cds)
tx_seq
```

```
##   A DNAStringSet instance of length 20
##      width seq                                                   names
##  [1]  2280 ATGGATTTATCTGCTCTTCGCGTTGAAG...CCCCAGATCCCCCACAGCCACTACTGA uc010whl.2
##  [2]  5379 ATGAGCCTACAAGAAAGTACGAGATTTA...CCCCAGATCCCCCACAGCCACTACTGA uc002icp.4
##  [3]   522 ATGGATGCTGAGTTTGTGTGTGAACGGA...CCCCAGATCCCCCACAGCCACTACTGA uc010whm.2
##  [4]  2100 ATGGATTTATCTGCTCTTCGCGTTGAAG...CTTCCATGCAATTGGGCAGATGTGTGA uc002icu.3
##  [5]  5451 ATGCTGAAACTTCTCAACCAGAAGAAAG...CCCCAGATCCCCCACAGCCACTACTGA uc010cyx.3
##  ...   ... ...
## [16]  4095 ATGGATTTATCTGCTCTTCGCGTTGAAG...GAAGAGCAAAGCATGGATTCAAACTTA uc010cyy.1
```

```
## [17]  4095 ATGGATTTATCTGCTCTTCGCGTTGAAG...GAAGAGCAAAGCATGGATTCAAACTTA uc010whs.1
## [18]  3954 ATGCTGAAACTTCTCAACCAGAAGAAAG...GAAGAGCAAAGCATGGATTCAAACTTA uc010cyz.2
## [19]  4017 ATGGATTTATCTGCTCTTCGCGTTGAAG...GAAGAGCAAAGCATGGATTCAAACTTA uc010cza.2
## [20]  3207 ATGAATGTAGAAAAGGCTGAATTCTGTA...GAAGAGCAAAGCATGGATTCAAACTTA uc010wht.1
```

Intron coordinates can be identified by first calculating the range of the genome (from the start of the first exon to the end of the last exon) covered by each transcript, and then taking the (algebraic) set difference between this and the genomic coordinates covered by each exon

```
introns <- psetdiff(range(brca1cds), brca1cds)
```

Retrieve the intronic sequences with `getSeq` (these are *not* assembled, the way that `extractTranscriptSeqs` assembles exon sequences into mature transcripts); note that introns start and end with the appropriate acceptor and donor site sequences.

```
seq <- getSeq(genome, introns)
names(seq)
```

```
##  [1] "uc010whl.2" "uc002icp.4" "uc010whm.2" "uc002icu.3" "uc010cyx.3" "uc002icq.3"
##  [7] "uc002ict.3" "uc010whn.2" "uc010who.3" "uc010whp.2" "uc010whq.1" "uc002idc.1"
## [13] "uc010whr.1" "uc002idd.3" "uc002ide.1" "uc010cyy.1" "uc010whs.1" "uc010cyz.2"
## [19] "uc010cza.2" "uc010wht.1"
```

```
seq[["uc010whl.2"]]                    # 21 introns
```

```
##   A DNAStringSet instance of length 21
##      width seq
##  [1]  1840 GTAAGGTGCCTGCATGTACCTGTGCTATATGGGGTCCT...AATTGACACTAATCTCTGCTTGTGTTCTCTGTCTCCAG
##  [2]  1417 GTAAGTATTGGGTGCCCTGTCAGAGAGGGAGGACACAA...TTGACACTTTGAATGCTCTTTCCTTCCTGGGGATCCAG
##  [3]  1868 GTAAGAGCCTGGGAGAACCCCAGAGTTCCAGCACCAGC...ATTACTGCAGTGATTTTACATCTAAATGTCCATTTTAG
##  [4]  5934 GTAAAGCTCCCTCCCTCAAGTTGACAAAAATCTCACCC...CCATTCCCCTGTCCCTCTCTCTTCCTCTCTTCTTCCAG
##  [5]  6197 GTAAGTACTTGATGTTACAAACTAACCAGAGATATTCA...TTCTCTTATCCTGATGGGTTGTGTTTGGTTTCTTTCAG
##  ...   ... ...
## [17]  4241 GTAAAACCATTTGTTTTCTTCTTCTTCTTCTTCTTCTT...CCAACAATTGCTTGACTGTTCTTTACCATACTGTTTAG
## [18]   606 GTAAGTGTTGAATATCCCAAGAATGACACTCAAGTGCT...CTAACTGCAAACATAATGTTTTCCCTTGTATTTTACAG
## [19]  1499 GTATATAATTTGGTAATGATGCTAGGTTGGAAGCAACC...CACTTGCTGAGTGTGTTTCTCAAACAATTTAATTTCAG
## [20]  9192 GTAAGTTTGAATGTGTTATGTGGCTCCATTATTAGCTT...GTAAATTAAATTGTTCTTTCTTTCTTTATAATTTATAG
## [21]  8237 GTAAGTCAGCACAAGAGTGTATTAATTTGGGATTCCTA...CTCATTTATTTTCTTTTTCTCCCCCCCTACCCTGCTAG
```

2.2 rtracklayer

The *rtracklayer* package allows us to query the UCSC genome browser, as well as providing `import` and `export` functions for common annotation file formats like GFF, GTF, and BED.

Exercise 5
warning: This exericse requires INTERNET ACCESS

Here we use rtracklayer to retrieve estrogen receptor binding sites identified across cell lines in the ENCODE project. We focus on binding sites in the vicinity of a particularly interesting region of interest.

a. *Define our region of interest by creating a GRanges instance with appropriate genomic coordinates. Our region corresponds to 10Mb up- and down-stream of a particular gene.*
b. *Create a session for the UCSC genome browser*
c. *Query the UCSC genome browser for ENCODE estrogen receptor ERalpha$_a$ transcription marks; identifying the appropriate track, table, and transcription factor requires biological knowledge and detective work.*
d. *Visualize the location of the binding sites and their scores; annotate the mid-point of the region of interest.*

Solution: Define the region of interest

```
library(GenomicRanges)
roi <- GRanges("chr10", IRanges(92106877, 112106876, names="ENSG00000099194"))
```

Create a session

```
library(rtracklayer)
session <- browserSession()
```

Query the UCSC for a particular track, table, and transcription factor, in our region of interest

```
trackName <- "wgEncodeRegTfbsClusteredV2"
tableName <- "wgEncodeRegTfbsClusteredV2"
trFactor <- "ERalpha_a"
ucscTable <- getTable(ucscTableQuery(session, track=trackName,
    range=roi, table=tableName, name=trFactor))
```

Visualize the result

```
plot(score ~ chromStart, ucscTable, pch="+")
abline(v=start(roi) + (end(roi) - start(roi) + 1) / 2, col="blue")
```