Database mining with biomaRt

Steffen Durinck
Lawrence Berkeley National Laboratory

Statistics and Genomics seminar 2010
UC Berkeley
Overview

- The BioMart software suite
- biomaRt package
- Workshop style discussion to show the variety of different data types that can be retrieved for many organisms
BioMart 0.7

- BioMart is a query-oriented data management system developed jointly by the European Bioinformatics Institute (EBI) and Cold Spring Harbor Laboratory (CSHL).
- Originally developed for the Ensembl project but has now been generalized.
BioMart 0.7

- BioMart data can be accessed using either web, graphical, or text based applications, or programmatically using web services or software libraries written in Perl and Java.

- http://www.biomart.org
Example BioMart databases

- Ensembl
- Wormbase
- Reactome
- Gramene
- ....
BioMart databases

- De-normalized
- Tables with ‘redundant’ information
- Query optimized
- Fast and flexible
- Well suited for batch querying
biomaRt

- R interface to BioMart databases
- Performs online queries
- Current release version 2.2.0
- Depends on Rcurl and XML packages
Download Statistics

biomaRt

Downloads
Distinct IPs
List available BioMart databases

> library(biomaRt)
Loading required package: XML
Loading required package: Rcurl
> listMarts()
## List available BioMarts

<table>
<thead>
<tr>
<th>biomart</th>
<th>version</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ensembl</td>
<td>ENSEMBL 55 GENES (SANGER UK)</td>
</tr>
<tr>
<td>2 snp</td>
<td>ENSEMBL 55 VARIATION (SANGER UK)</td>
</tr>
<tr>
<td>3 functional_genomics</td>
<td>ENSEMBL 55 FUNCTIONAL GENOMICS</td>
</tr>
<tr>
<td>4 vega</td>
<td>VEGA 35 (SANGER UK)</td>
</tr>
<tr>
<td>5 msd</td>
<td>MSD PROTOTYPE (EBI UK)</td>
</tr>
<tr>
<td>6 htgt</td>
<td>HIGH THROUGHPUT GENE TARGETING AND TRAPPING</td>
</tr>
<tr>
<td>7 QTL_MART</td>
<td>GRAMENE 29 QTL DB (CSHL US)</td>
</tr>
<tr>
<td>8 ENSEMBL_MART_ENSEMBL</td>
<td>GRAMENE 29 GENES</td>
</tr>
<tr>
<td>9 ENSEMBL_MART_SNP</td>
<td>GRAMENE 29 SNPs</td>
</tr>
<tr>
<td>10 GRAMENE_MARKER_29</td>
<td>GRAMENE 29 MARKERS</td>
</tr>
</tbody>
</table>
Ensembl

- Ensembl is a joint project between EMBL - European Bioinformatics Institute (EBI) and the Wellcome Trust Sanger Institute (WTSI)
- A software system which produces and maintains automatic annotation on selected eukaryotic genomes.
- http://www.ensembl.org
Ensembl - BioMart

> ensembl=useMart("ensembl")
> listDatasets(ensembl)

Returns:
- name: hsapiens_gene_ensembl
- description: Homo sapiens genes
- version: (GRCh37)

Ensembl currently contains 50 datasets~species
Ensembl - Datasets

A dataset can be selected using the useMart function

```r
> ensembl = useMart("ensembl", dataset="hsapiens_gene_ensembl")
```

Checking attributes ... ok
Checking filters ... ok
biomaRt query: Attributes

- Attributes define the values which the user is interested in.
- Conceptually equal to output of the query
- Example attributes:
  - chromosome_name
  - band
biomaRt query: Filters

- Filters define restrictions on the query
- Conceptually filters are inputs

- Example filters:
  - entrezgene
  - chromosome_name
biomaRt query

Attributes (e.g., chromosome and band)

Filters (e.g., “entrezgene”)

Values (e.g., EntrezGene identifiers)
Three main biomaRt functions

- **listFilters**
  - Lists the available filters
- **listAttributes**
  - Lists the available attributes
- **getBM**
  - Performs the actual query and returns a data.frame
Microarrays & Ensembl

- Ensembl does an independent mapping of array probe sequences to genomes (Affymetrix, Illumina, Agilent, ...)
- If there is no clear match then that probe is not assigned to a gene
Annotate the following Affymetrix probe identifiers from the human u133plus2 platform with hugo gene nomenclature symbol (hgnc_symbol) and chromosomal location information:

211550_at, 202431_s_at, 206044_s_at
TASK 1 - Ensembl

- Filters: affy_hg_u133_plus_2
- Attributes: affy_hg_u133_plus_2, chromosome_name, start_position, end_position, band, strand

- Values: 211550_at, 202431_s_at, 206044_s_at
> affyids = c("211550_at","202431_s_at","206044_s_at")

> annotation = getBM(attributes=c("affy_hg_u133_plus_2","ensembl_gene_id","hgnc_symbol","chromosome_name","start_position","end_position","band","strand"), filters="affy_hg_u133_plus_2", values=affyids, mart = ensembl)
### TASK 1 - Ensembl

<table>
<thead>
<tr>
<th>annotation</th>
<th>chromosome_name</th>
<th>ensembl_gene_id</th>
<th>hgnc_symbol</th>
<th>start_position</th>
<th>end_position</th>
<th>band</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>202431_s_at</td>
<td>ENSG00000136997</td>
<td>128747680</td>
<td>128753674</td>
<td>q24.21</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>211550_at</td>
<td>ENSG00000146648</td>
<td>55086714</td>
<td>55324313</td>
<td>p11.2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>206044_s_at</td>
<td>ENSG00000157764</td>
<td>140424943</td>
<td>140624564</td>
<td>q34</td>
<td>-1</td>
</tr>
</tbody>
</table>
TASK 1* - Ensembl

Retrieve GO annotation for the following Illumina
human_wg6_v2 identifiers:
ILMN_1728071, ILMN_1662668

```r
> illuminalIDs = c("ILMN_1728071","ILMN_1662668")
> goAnnot = getBM(c("illumina_humanwg_6_v2", "go_biological_process_id", "go_biological_process_linkage_type"), filters="illumina_humanwg_6_v2", values=illuminalIDs, mart = ensembl)
```
## TASK 1* - Ensembl

<table>
<thead>
<tr>
<th>illumina_humanwg_6_v2</th>
<th>go_biological_process_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILMN_1662668</td>
<td>GO:0000281</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0006461</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0006974</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0007026</td>
</tr>
<tr>
<td>ILMN_1662668</td>
<td>GO:0007050</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>go_biological_process_linkage_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP</td>
</tr>
<tr>
<td>IDA</td>
</tr>
<tr>
<td>IDA</td>
</tr>
<tr>
<td>IDA</td>
</tr>
<tr>
<td>IDA</td>
</tr>
</tbody>
</table>
Using more than one filter

- `getBM` can be used with more than one filter
- Filters should be given as a vector
- Values should be a list of vectors where the position of each vector corresponds with the position of the associated filter in the filters argument
TASK 2 - Ensembl

Retrieve all genes that are involved in Diabetes Mellitus Type I or Type II and have transcription factor activity
1. Diabetes Mellitus type I MIM accession: 222100
2. Diabetes Mellitus type II MIM accession: 125853
3. GO id for “transcription factor activity”: GO: 0003700
diab=getBM(c("ensembl_gene_id","hgnc_symbol"),
            filters=c("mim_morbid_accession","go"),
            values=list(c("125853","222100"),"GO:0003700"),
            mart=ensembl)
<table>
<thead>
<tr>
<th>ensembl_gene_id</th>
<th>hgnc_symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENSG00000139515</td>
<td>PDX1</td>
</tr>
<tr>
<td>ENSG00000108753</td>
<td>HNF1B</td>
</tr>
<tr>
<td>ENSG00000148737</td>
<td>TCF7L2</td>
</tr>
<tr>
<td>ENSG00000106331</td>
<td>PAX4</td>
</tr>
<tr>
<td>ENSG00000162992</td>
<td>NEUROD1</td>
</tr>
<tr>
<td>ENSG00000135100</td>
<td>HNF1A</td>
</tr>
</tbody>
</table>
Boolean filters

- Filters can be either numeric, string or boolean
- Boolean filters should have either TRUE or FALSE as values
  - TRUE: return all information that comply with the given filter (e.g. return only genes that have a hgnc_symbol)
  - FALSE: return all information that doesn’t comply with the given filter (e.g. with no hgnc_symbol)
Boolean filters/ filterType

The function `filterType` allows you to figure out which type each filter is (this function is currently only available in the devel version of biomaRt)

```r
> filterType("affy_hg_u133_plus_2", mart=ensembl)
[1] "id_list"

> filterType("with_affy_hg_u133_plus_2", mart=ensembl)
[1] "boolean_list"
```
TASK 3 - Ensembl

Retrieve all miRNAs known on chromosome 13 and their chromosomal locations
TASK 3 - Ensembl

```r
> miRNA = getBM(c("mirbase","ensembl_gene_id","start_position","chromosome_name"), filters=c("chromosome_name","with_mirbase"), values=list(13,TRUE), mart=ensembl)
> miRNA[1:5,]
```
<table>
<thead>
<tr>
<th>mirbase</th>
<th>ensembl_gene_id</th>
<th>start_position</th>
<th>chromosome_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 MI0008190</td>
<td>ENSG00000211491</td>
<td>41301964</td>
<td>13</td>
</tr>
<tr>
<td>2 MI0003635</td>
<td>ENSG00000207652</td>
<td>41384902</td>
<td>13</td>
</tr>
<tr>
<td>3 MI0000070</td>
<td>ENSG00000208006</td>
<td>50623109</td>
<td>13</td>
</tr>
<tr>
<td>4 MI0000069</td>
<td>ENSG00000207718</td>
<td>50623255</td>
<td>13</td>
</tr>
<tr>
<td>5 MI0003636</td>
<td>ENSG00000207858</td>
<td>90883436</td>
<td>13</td>
</tr>
</tbody>
</table>
TASK 4 - Ensembl

Retrieve all entrezgene identifiers on chromosome 22 that have a non-synonymous coding SNP
> filterOptions("snptype_filters", ensembl)
[1] "[STOP_GAINED,STOP_LOST,COMPLEX_INDEL,FRAME_SHIFT_CODING,
NON_SYNONYMOUS_CODING,STOP_GAINED,SPlice_SITE,STOP_LOST,
SPlice_SITE,FRAME_SHIFT_CODING,SPlice_SITE,
NON_SYNONYMOUS_CODING,SPlice_SITE,SYNONYMOUS_CODING,
SPlice_SITE,SYNONYMOUS_CODING,SPlice_SITE,SYNONYMOUS_CODING,5PRIME_UTR,SPlice_SITE,
5PRIME_UTR,3PRIME_UTR,SPlice_SITE,3PRIME_UTR,INTRONIC,
ESSENTIAL_SPlice_SITE,INTRONIC,SPlice_SITE,INTRONIC,UPSTREAM,
DOWNSTREAM]"

> entrez = getBM("entrezgene", filters=c("chromosome_name",
 snptype_filters"), values=list(22,"NON_SYNONYMOUS_CODING"),
 mart=ensembl)

> entrez[1:5,]
> [1] 23784 81061 150160 150165 128954
getSequence

- Retrieving sequences from Ensembl can be done using the `getBM` function or the `getSequence` wrapper function.
- Output of `getSequence` can be exported to FASTA file using the `exportFASTA` function.
getSequence

- Available sequences in Ensembl:
  - Exon
  - 3’UTR
  - 5’UTR
  - Upstream sequences
  - Downstream sequences
  - Unspliced transcript/gene
  - Coding sequence
  - Protein sequence
getSequence

Arguments of getSequence:
- \textit{id} : identifier
- \textit{type} : type of identifier used e.g. hgnc\_symbol or affy\_hg\_u133\_plus\_2
- \textit{seqType} : sequence type that needs to be retrieved e.g. gene\_exon, coding, 3utr, 5utr,
- \textit{upstream/downstream} : specify number of base pairs upstream/downstream that need to be retrieved
TASK 5 - Ensembl

Retrieve all exons of CDH1
> seq = getSequence(id="**CDH1**", type="hgnc_symbol", seqType="gene_exon", mart = ensembl)

> seq[1,]

`gene_exon`

```
1
TACAAGGGTCAGGTGCCTGAGAACGAGGCTAACGTCGTAATCACCA
CACTGAAAGTGACTGATGCTGATGCCCCCAATACCCCGAGCGTGGGA
GGCTGTATACACCATATTGAATGATGATGATGTTTGGGACAATTTTGTCGTCA
CCACAAATCCAGTGAACAACGATGGCATTTTGAAAACAGCAAAG
```

`hgnc_symbol`

```
1 CDH1
```
TASK 6 - Ensembl

Retrieve 2000bp sequence upstream of the APC and CUL1 translation start site
> promoter=getSequence(id=c("APC","CUL1"),type="hgnc_symbol", seqType="coding_gene_flank",upstream =2000, mart=ensembl)

> promoter
coding_gene_flank
1 TTGTTTACATCTGAAGAGATTGATTTTTTTATTCCTGTAATA..................
2 TCCGTAGCAGTTGAATGTG .....................

hgnc_symbol
1 APC
2 CUL1
Homology - Ensembl

- The different species in Ensembl are interlinked
- biomaRt takes advantage of this to provide homology mappings between different species
Linking two datasets

- Two datasets (e.g. two species in Ensembl) can be linked to each other by using the `getLDS` (get linked dataset) function.
- One has to connect to two different datasets and specify the linked dataset using `martL`, `filtersL`, `attributesL`, `valuesL ` arguments.
Retrieve human gene symbol and affy identifiers of their homologs in chicken for the following two identifiers from the human affy_hg_u95av2 platform: 1434_at, 1888_s_at
TASK 7 - Ensembl

```r
> human=useMart("ensembl", dataset="hsapiens_gene_ensembl")
Checking attributes and filters ... ok
> chicken=useMart("ensembl", dataset="ggallus_gene_ensembl")
Checking attributes and filters ... ok

> out = getLDS(attributes=c("affy_hg_u95av2","hgnc_symbol"), filters="affy_hg_u95av2", values=c("1888_s_at","1434_at"),mart=human, attributesL="affy_chicken", martL=chicken)
> out
V1 V2 V3
1 1434_at PTEN GgaAffx.25913.1.S1_a
2 1888_s_at KIT Gga.606.1.S1_at
```
Variation BioMart

- dbSNP mapped to Ensembl

```r
> snp = useMart("snp", dataset="hsapiens_snp")
```
TASK 8 - Variation

Retrieve all refsnp_ids and their alleles and position that are located on chromosome 8 and between bp 148350 and 158612.
> out=getBM(attributes=c("refsnp_id","allele","chrom_start"),
filters=c("chr_name","chrom_start","chrom_end"), values=list(8,148350, 158612), mart=snp)

> out[1:5,]

<table>
<thead>
<tr>
<th>refsnp_id</th>
<th>allele</th>
<th>chrom_start</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENSSNP4490669</td>
<td>C/G</td>
<td>148729</td>
</tr>
<tr>
<td>ENSSNP5558526</td>
<td>T/C</td>
<td>148909</td>
</tr>
<tr>
<td>ENSSNP4089737</td>
<td>T/A</td>
<td>149060</td>
</tr>
<tr>
<td>ENSSNP9060169</td>
<td>C/T</td>
<td>149245</td>
</tr>
<tr>
<td>ENSSNP4351891</td>
<td>C/G</td>
<td>149250</td>
</tr>
</tbody>
</table>
Ensembl Archives

- Provide alternate host

```r
> listMarts(host="may2009.archive.ensembl.org/biomart/martservice/")
biomart version
1 ENSEMBL_MART_ENSEMBL Ensembl 54
2 ENSEMBL_MART_SNP Ensembl Variation 54
3 ENSEMBL_MART_VEGA Vega 35
4 REACTOME Reactome(CSHL US)
5 wormbase_current WormBase (CSHL US)
6 pride PRIDE (EBI UK)

> ensembl54=useMart("ENSEMBL_MART_ENSEMBL", host="may2009.archive.ensembl.org/biomart/martservice/")
```
Ensembl Archives

- Access to archives by setting `archive=TRUE` or connect to specific host (Note that this is currently not up to date in the central repository)

```r
> listMarts(archive=TRUE)
biomart version
1 ensembl_mart_51 Ensembl 51
2 snp_mart_51 SNP 51
3 vega_mart_51 Vega 32
4 ensembl_mart_50 Ensembl 50
5 snp_mart_50 SNP 50

> ensembl51 = useMart("ensembl_mart_51", archive=TRUE, dataset="hsapiens_gene_ensembl")
```
Gramene

Gramene is a curated, open-source, data resource for comparative genome analysis in the grasses.

- Rice, Maize and Arabidopsis
TASK 9 - Gramene

Retrieve affy ATH1 ids and CATMA ids that map to the *Arabidopsis thaliana* chromosome 1 between basepair 30.000 and 41.000
TASK 9 - Gramene

```r
> gramene = useMart(
    "ENSEMBL_MART_ENSEMBL", dataset="
    athaliana_gene_ensembl")

> getBM(c("affy_ath1_id","catma_tigr5_id"),
    filters=c("chromosome_name","start","end"),
    values=list("1", "30000","41000"),
    mart=gramene)
```
# TASK 9 - Gramene

<table>
<thead>
<tr>
<th>affy_ath1_id</th>
<th>catma_tigr5_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 261579_at</td>
<td>CATMA1a00040</td>
</tr>
<tr>
<td>2 261569_at</td>
<td>CATMA1a00045</td>
</tr>
<tr>
<td>3 261569_at</td>
<td>CATMA1a00045</td>
</tr>
<tr>
<td>4 261569_at</td>
<td>CATMA1a00045</td>
</tr>
<tr>
<td>5 261576_at</td>
<td>CATMA1a00050</td>
</tr>
<tr>
<td>6 261576_at</td>
<td>CATMA1a00050</td>
</tr>
</tbody>
</table>
Wormbase

- Database on the genetics of C elegans and related nematodes.
TASK 10 - Wormbase

Determine the RNAi ids and the observed phenotypes for the gene with wormbase gene id: WBGene00006763
> worm = useMart("wormbase176", dataset="wormbase_rnai")

> pheno = getBM(c("rnai", "phenotype_primary_name"), filters="gene", values="WBGene00006763", mart=worm)
## TASK 10 - Wormbase

> pheno

<table>
<thead>
<tr>
<th>ID</th>
<th>ID</th>
<th>phenotype_primary_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WBRNAi00021278</td>
<td>slow_growth</td>
</tr>
<tr>
<td>2</td>
<td>WBRNAi00021278</td>
<td>postembryonic_development_abnormal</td>
</tr>
<tr>
<td>3</td>
<td>WBRNAi00021278</td>
<td>embryonic_lethal</td>
</tr>
<tr>
<td>4</td>
<td>WBRNAi00021278</td>
<td>larval_lethal</td>
</tr>
<tr>
<td>5</td>
<td>WBRNAi00021278</td>
<td>larval_arrest</td>
</tr>
<tr>
<td>6</td>
<td>WBRNAi00021278</td>
<td>maternal_sterile</td>
</tr>
<tr>
<td>7</td>
<td>WBRNAi00021278</td>
<td>Abnormal</td>
</tr>
<tr>
<td>8</td>
<td>WBRNAi00021278</td>
<td>sterile_progeny</td>
</tr>
<tr>
<td>9</td>
<td>WBRNAi00026915</td>
<td>slow_growth</td>
</tr>
<tr>
<td>10</td>
<td>WBRNAi00026915</td>
<td>postembryonic_development_abnormal</td>
</tr>
</tbody>
</table>
Discussion

- Using biomaRt to query public web services gets you started quickly, is easy and gives you access to a large body of metadata in a uniform way.
- Need to be online.
- Online metadata can change behind your back; although there is possibility of connecting to a particular, immutable version of a dataset.
Reporting bugs

- Check with MartView if you get the same output
  - Yes: contact database e.g. helpdesk@ensembl.org
  - No: contact me - sdurinck@lbl.gov
Acknowledgements

- EBI
  - Rhoda Kinsella
  - Arek Kasprzyk
  - Ewan Birney

- EMBL
  - Wolfgang Huber

Bioconductor users