Building and Using Ensembl Based Annotation Packages with ensembldb

Johannes Rainer

June 22, 2016

Introduction

- `TxDb` objects from `GenomicFeatures` provide gene model annotations:
  - Used for RNA-seq, CHiP-seq, etc.
  - Providing mostly UCSC annotations.
- `ensembldb` package defines the `EnsDb` class:
  - Same functionality as `TxDb` objects, plus:
  - Designed for Ensembl: all genes, attributes *gene biotype* and *tx biotype*.
  - Allows to query specific annotations using a simple filter framework.

Query gene, transcript, exon information

- Available methods to extract data:
    - `genes`
    - `transcripts`
    - `transcriptsBy`
    - `exons`
    - `exonsBy`
    - `cdsBy`
    - `fiveUTRsByTranscripts`
    - `threeUTRsByTranscripts`

Query gene, transcript, exon information

- Example: get all genes' annotations.

```r
## Load an EnsDb package matching Ensembl version 81
library(EnsDb.Hsapiens.v81)
edb <- EnsDb.Hsapiens.v81

## Now just get all genes
genes(edb)
```

```
          ...
ENSG00000185220          1 [248906196, 248919946]      + | ENSG00000185220
ENSG00000200495          1 [248912690, 248912795]      - | ENSG00000200495
ENSG00000233084          1 [248936581, 248937043]      + | ENSG00000233084
                    gene_name    entrezid         gene_biotype seq_coord_system
                  <character> <character>          <character>      <character>
ENSG00000278806    AF065393.4                            miRNA         scaffold
ENSG00000210049         MT-TF                          Mt_tRNA       chromosome
ENSG00000211459       MT-RNR1                          Mt_rRNA       chromosome
            ...           ...         ...                  ...              ...
ENSG00000185220         PGBD2      267002       protein_coding       chromosome
ENSG00000200495     RNU6-1205P                           snRNA       chromosome
ENSG00000233084      RPL23AP25             processed_pseudogene       chromosome
-------
seqinfo: 338 sequences from GRCh38 genome
```

Query gene, transcript, exon information

- Example: get all genes encoded on chromosome Y.

```
## Create a filter object
sf <- SeqnameFilter("Y")

## Retrieve the data.
genes(edb, filter=sf)
```

```
           ...
ENSG00000237917        Y [26594851, 26634652]      - | ENSG00000237917
ENSG00000231514        Y [26626520, 26627159]      - | ENSG00000231514
ENSG00000235857        Y [56855244, 56855488]      + | ENSG00000235857
                  gene_name    entrezid           gene_biotype
                <character> <character>            <character>
        LRG_186     LRG_186        1438               LRG_gene
ENSG00000251841   RNU6-1334P                             snRNA
ENSG00000184895         SRY        6736         protein_coding
            ...         ...         ...                    ...
ENSG00000237917     PARP4P1             unprocessed_pseudogene
ENSG00000231514     FAM58CP               processed_pseudogene
ENSG00000235857     CTBP2P1               processed_pseudogene
                seq_coord_system
                     <character>
        LRG_186       chromosome
ENSG00000251841       chromosome
ENSG00000184895       chromosome
            ...              ...
ENSG00000237917       chromosome
ENSG00000231514       chromosome
ENSG00000235857       chromosome
-------
```

Available filters

- For genes: `GeneidFilter`, `GenenameFilter`, `EntrezidFilter` and `GenebiotypeFilter`.
- For transcripts: `TxidFilter` and `TxbiotypeFilter`.
- For exons: `ExonidFilter` and `ExonrankFilter`.
- *Generic* filters: `SeqnameFilter`, `SeqstrandFilter`, `SeqstartFilter`, `SeqendFilter` and `GRangesFilter`.
- Multiple filters are combined with a logical *AND*.
- Each filter supports 1:n values and also a *like* condition.

Available filters

- Example: combine filters.

```r
## Example for a GRangesFilter:
grf <- GRangesFilter(GRanges(17, IRanges(59000000, 59200000)),
                     condition="within")
## Combine with a GenebiotypeFilter to get all genes in the region
## EXCEPT pre-miRNAs and snRNAs.
genes(edb, filter=list(grf,
                      GenebiotypeFilter(c("miRNA", "snRNA"),
                                        condition="!=")))
```

```
GRanges object with 4 ranges and 5 metadata columns:
                 seqnames               ranges strand |         gene_id
                    <Rle>            <IRanges>  <Rle> |     <character>
 ENSG00000263558       17 [59059226, 59059493]      + | ENSG00000263558
 ENSG00000224738       17 [59106598, 59118267]      + | ENSG00000224738
 ENSG00000182628       17 [59109951, 59155269]      - | ENSG00000182628
 ENSG00000266537       17 [59174983, 59181787]      - | ENSG00000266537
                   gene_name    entrezid           gene_biotype
                 <character> <character>            <character>
 ENSG00000263558   RN7SL716P                           misc_RNA
 ENSG00000224738  AC099850.1                          antisense
 ENSG00000182628        SKA2      348235         protein_coding
 ENSG00000266537    SPDYE22P             unprocessed_pseudogene
                 seq_coord_system
                      <character>
 ENSG00000263558       chromosome
 ENSG00000224738       chromosome
 ENSG00000182628       chromosome
 ENSG00000266537       chromosome
```

ensembldb and the AnnotationDbi API

- EnsDb support all AnnotationDbi methods with filters.
- Example: use AnnotationDbi's select method to fetch annotations.

```r
## Get all data for the gene SKA2
Res <- select(edb, keys="SKA2", keytype="GENENAME")
head(Res, n=3)
```

```
  ENTREZID          EXONID EXONIDX EXONSEQEND EXONSEQSTART    GENEBIOTYPE
1   348235 ENSE00001324111       1   59155269     59155131 protein_coding
2   348235 ENSE00003636954       2   59131367     59131281 protein_coding
3   348235 ENSE00003478713       3   59119495     59119319 protein_coding
           GENEID GENENAME GENESEQEND GENESEQSTART ISCIRCULAR SEQCOORDSYSTEM
1 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
2 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
3 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
  SEQLENGTH SEQNAME SEQSTRAND      TXBIOTYPE TXCDSSEQEND TXCDSSEQSTART
1  83257441      17        -1 protein_coding    59155163      59112277
2  83257441      17        -1 protein_coding    59155163      59112277
3  83257441      17        -1 protein_coding    59155163      59112277
             TXID          TXNAME TXSEQEND TXSEQSTART
1 ENST00000330137 ENST00000330137 59155269   59109951
2 ENST00000330137 ENST00000330137 59155269   59109951
3 ENST00000330137 ENST00000330137 59155269   59109951
```

ensembldb and the AnnotationDbi API

```r
## Or: pass filters with keys parameter to have more control:
## For the gene SKA2: get all exons except exons 1 and 2
## for all tx targeted for nonsense mediated decay.
select(edb, keys=list(GenenameFilter("SKA2"),
                      TxbiotypeFilter("nonsense_mediated_decay"),
                      ExonrankFilter(1:2, condition="!=")))
```

```
  ENTREZID          EXONID EXONIDX EXONSEQEND EXONSEQSTART     GENEBIOTYPE
1   348235 ENSE00002710994       3   59124428     59124307 protein_coding
2   348235 ENSE00003552567       4   59119495     59119319 protein_coding
3   348235 ENSE00002729093       5   59112345     59111890 protein_coding
4   348235 ENSE00003594135       3   59119495     59119319 protein_coding
5   348235 ENSE00002695019       4   59112345     59112262 protein_coding
           GENEID GENENAME GENESEQEND GENESEQSTART ISCIRCULAR SEQCOORDSYSTEM
1 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
2 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
3 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
4 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
5 ENSG00000182628     SKA2   59155269     59109951          0     chromosome
  SEQLENGTH SEQNAME SEQSTRAND               TXBIOTYPE TXCDSSEQEND TXCDSSEQSTART
1  83257441      17        -1 nonsense_mediated_decay    59155163      59124363
2  83257441      17        -1 nonsense_mediated_decay    59155163      59124363
3  83257441      17        -1 nonsense_mediated_decay    59155163      59124363
4  83257441      17        -1 nonsense_mediated_decay    59155083      59119474
5  83257441      17        -1 nonsense_mediated_decay    59155083      59119474
             TXID          TXNAME TXSEQEND TXSEQSTART
1 ENST00000578519 ENST00000578519 59155182   59111890
2 ENST00000578519 ENST00000578519 59155182   59111890
3 ENST00000578519 ENST00000578519 59155182   59111890
4 ENST00000583976 ENST00000583976 59155177   59112262
5 ENST00000583976 ENST00000583976 59155177   59112262
```

Annotation for feature counting

- exonsBy: provide gene model information for feature counting.
- Example: feature counting using GenomicAlignments' summarizeOverlaps method.

```r
## Get exons by gene, for chromosomes 1:22, X, Y, excluding also locus reference
## genomic genes (LRG)
exns <- exonsBy(edb, by="gene", filter=list(SeqnameFilter(c(1:22, "X", "Y")),
                                            GeneidFilter("ENSG%", "like")))
## Load the required libraries.
library(GenomicAlignments)
library(BiocParallel)
## Get the Bam files.
bfl <- BamFileList(dir("data/bam", pattern=".bam$", full.names=TRUE),
                   asMates=TRUE, yieldSize=1e+6, obeyQname=TRUE)
## Define a ScanBamParam with a mapping quality filter.
sbp <- ScanBamParam(mapqFilter=30)
## Do the gene counting
geneCounts <- bplapply(bfl, FUN=summarizeOverlaps, features=exns,
                       mode="IntersectionStrict", ignore.strand=TRUE,
                       singleEnd=FALSE, fragments=TRUE, param=sbp)
geneCounts <- do.call(cbind, geneCounts)
```

Annotation for feature counting

- Example: gene models for Rsubread'2 featureCount function.

```r
## Convert the exon list to SAF format
saf <- toSAF(exns)

head(saf)

####
##  Do the feature counting using the Rsubread package
library(Rsubread)
bamf <- dir("data/bam", pattern=".bam$", full.names=TRUE)
cnts <- featureCounts(files=bamf, annot.ext=saf, isPairedEnd=TRUE, nthreads=1)
```

Integrating UCSC and Ensembl annotations

- UCSC and Ensembl use different chromosome naming styles.
- Example: How to integrate Ensembl based annotation with UCSC data?

```r
## Get chromosome names
head(seqlevels(edb))
## Different from UCSC style: chr1...
```

```
[1] "1"  "10" "11" "12" "13" "14"
```

```r
## Get genes on chromosome Y, UCSC style.
genes(edb, filter=SeqnameFilter("chrY"))
```

```
GRanges object with 0 ranges and 5 metadata columns:
   seqnames    ranges strand |     gene_id   gene_name    entrezid gene_biotype
      <Rle> <IRanges>  <Rle> | <character> <character> <character>  <character>
   seq_coord_system
        <character>
  -------
  seqinfo: no sequences
```

Integrating UCSC and Ensembl annotations

```r
## Solution: change the chromosome naming style:
seqlevelsStyle(edb) <- "UCSC"
## Get chromosome names
head(seqlevels(edb))
```

```
[1] "chr1"  "chr10" "chr11" "chr12" "chr13" "chr14"
Warning message:
In .formatSeqnameByStyleFromQuery(x, sn, ifNotFound) :
  More than 5 seqnames with seqlevels style of the database (Ensembl) could not be mapped to the seq
```

- Sequence names are mapped between *styles* using the GenomeInfoDb package.

```r
genes(edb, filter=SeqnameFilter("chrY"))
```

```
             ...
ENSG00000237917     chrY [26594851, 26634652]      - | ENSG00000237917
ENSG00000231514     chrY [26626520, 26627159]      - | ENSG00000231514
ENSG00000235857     chrY [56855244, 56855488]      + | ENSG00000235857
                  gene_name    entrezid          gene_biotype
                <character> <character>           <character>
        LRG_186     LRG_186        1438              LRG_gene
ENSG00000251841  RNU6-1334P                             snRNA
ENSG00000184895         SRY        6736        protein_coding
            ...         ...         ...                   ...
ENSG00000237917     PARP4P1             unprocessed_pseudogene
ENSG00000231514      FAM58CP               processed_pseudogene
ENSG00000235857      CTBP2P1               processed_pseudogene
                seq_coord_system
                     <character>
```

Integrating UCSC and Ensembl annotations

```r
## Use case:
## Get mRNA sequences for SKA2 using BSgenome.
library(BSgenome.Hsapiens.UCSC.hg38)  ## <- UCSC based
## Get exons by transcript
ska2tx <- exonsBy(edb, by="tx", filter=GenenameFilter("SKA2"))
## Use GenomicFeatures' extractTranscriptSeqs
head(extractTranscriptSeqs(BSgenome.Hsapiens.UCSC.hg38, ska2tx))
```

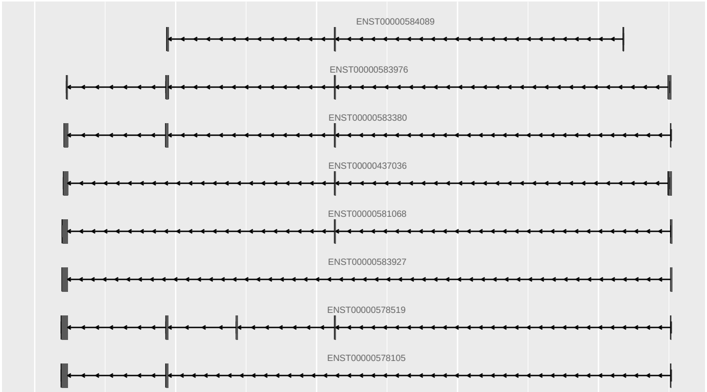
```
  A DNAStringSet instance of length 6
    width seq                                               names
[1]  2798 AATGAGTGCGAGATGTTGAGTGA...AACCTACAATCCTCTTTCTAAAA ENST00000330137
[2]   625 GCCGCGGTCTGCGGAATGTCAAC...AATGAGAATAAAACGATTTAAAT ENST00000437036
[3]   689 GCGGAATGTCAACTATTCAACAT...TGTACATTTCAGTCATTCGGTAT ENST00000578105
[4]   894 GGAATGTCAACTATTCAACATGG...TATGTACATTTCAGTCATTCGGT ENST00000578519
[5]   689 GCGGAATGTCAACTATTCAACAT...TACATTTCAGTCATTCGGTATGT ENST00000580541
[6]   595 GACAGCTGTCCAATGGAGGCCCT...TTGCATCTGTTTTCTTTTTCTAA ENST00000581068
```

- Preferred way: use `getGenomeFaFile` method to get the *correct* genomic sequence.

Plotting support

- ggbio and Gviz: plot data along genomic coordinates.
- ggbio: support for EnsDb objects and filters integrated.
- Example: use ggbio and ensembldb to plot a chromosomal region.

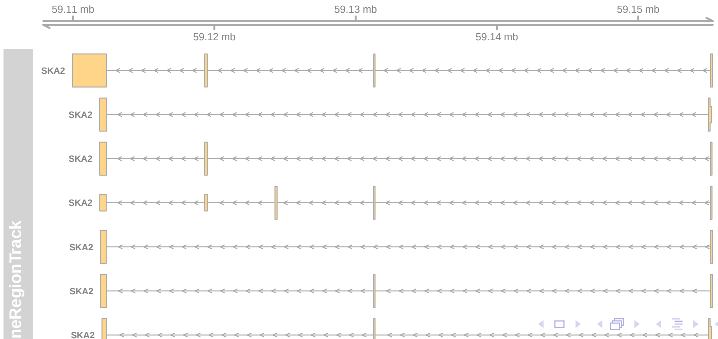
```r
library(ggbio)

## Plot the SKA2 gene model by passing a filter to the function.
autoplot(edb, GenenameFilter("SKA2"))
```



Plotting support

- Gviz: getGeneRegionTrackForGviz method to extract Gviz-formatted data.
- Example: plot genes encoded on a chromosomal region using Gviz.

```r
library(Gviz)
## Get all genes encoded in the same genomic region (same strand)
ska2 <- genes(edb, filter=GenenameFilter("SKA2"))
grt <- getGeneRegionTrackForGviz(edb, filter=GRangesFilter(ska2,
                                                   condition="overlapping"))
geneTrack <- GeneRegionTrack(grt)
plotTracks(list(GenomeAxisTrack(), geneTrack), transcriptAnnotation="symbol")
```



The ensembldb shiny app

- The ensembldb shiny app allows interactive annotation look-up.
- Example: search for a gene using the shiny app and return the result to R.

```r
## Run the shiny app:
Result <- runEnsDbApp()

## Inspect the result:
Result
```

Building annotation databases

The easiest way: with AnnotationHub

- ensDbFromAH: build an EnsDb database from an AnnotationHub (gtf) resource.

```r
library(AnnotationHub)
ah <- AnnotationHub()
## Query for available Ensembl gtf files for release 83.
query(ah, pattern=c("ensembl", "release-83", "gtf"))

## Select one; in this case: Anolis carolinensis (lizard)
edbSql83 <- ensDbFromAH(ah=ah["AH7537"])

## Use the database right away.
db <- EnsDb(edbSql83)
genes(db, filter=SeqnameFilter("2"))

## Make a package from the database.
makeEnsembldbPackage(ensdb=edbSql83, version="1.0.0",
                     maintainer="Johannes Rainer <johannes.rainer@eurac.edu>",
                     author="J Rainer")
```

- But: no NCBI Entrez Gene IDs available.

Building annotation databases

The easy way: from gtf and gff files

- ensDbFromGtf: create an EnsDb from a *gtf* or *gff* file.
- *Should* work with all gtf and gff files from Ensembl.
- But: gtf files don't provide NCBI Entrez Gene IDs.
- Example: create an EnsDb from a GTF file downloaded from `ftp://ftp.ensembl.org`.

```r
## Create an EnsDb from an Ensembl GTF file.

## Create the SQLite database file:
##  o Eventually define 'organism' and 'genomeVersion'.
##  o Needs also an internet connection to retrieve the 'seqlengths'.
edbSql <- ensDbFromGtf("data/gtf/Canis_familiaris.CanFam3.1.84.gtf.gz")

edbSql

## Use the makeEnsembldbPackage to create a package, or load and use it.
dogDb <- EnsDb(edbSql)

dogDb

## Fully functional, except we don't have Entrez gene ids.
head(genes(dogDb, filter=SeqnameFilter("X")))
```

Building annotation databases

The hard way: using Ensembl's Perl API

- Requires:
  - Perl.
  - Ensembl Perl API (and Bioperl).
- `fetchTablesFromEnsembl` to fetch the annotations from Ensembl.
- `makeEnsemblSQLiteFromTables` to create the SQLite database from the tables.
- `makeEnsembldbPackage` to create a package containing and providing the annotation.
- Example: create an EnsDb using the Perl API.

```
## Create an EnsDb using the Ensembl Perl API:
## This takes quite some time...
fetchTablesFromEnsembl(version="81",
                       ensemblapi="/Users/jo/ensembl/81/API/ensembl/modules",
                       species="dog")

## Create an SQLite database from the generated txt files
dbf <- makeEnsemblSQLiteFromTables()

## Finally, create the package
```

Finally. . .

Thank you for your attention!