Data Structures for Biological Data in *R*

Maarten van Iterson

June 23, 2017

Goals

- Which annotation packages are available within *Bioconductor*[1,2]
- How can we use these packages
- Get some idea of how these packages are implemented
- How to get annotation from online resources in *R*
- Use *Bioconductor* data/infra structure(s) for efficient handling of Biological data in *R*

---

[1]Gentleman, R. et al. (2004). Bioconductor: open software development for computational biology and bioinformatics.
Genome Biol., 5(10):R80

[2]Huber, W. et al. (2015). Orchestrating high-throughput genomic analysis with Bioconductor.
Nat. Methods, 12(2):115–121

Major types of annotation in *Bioconductor*

Meta data/Annotation *AnnotationDbi*

- Organism level: *org.Mm.eg.db*
- Platform level: *hgu133plus2.db*
- System-biology level: *GO.db* or *KEGG.db*
- Transcript centric annotations: *GenomicFeatures*

Range data *IRanges*, *GenomicRanges* and *GenomicFeatures*

- Genomic ranges: *TxDb.Hsapiens.UCSC.hg19.knownGene*

For example: annotation tracks from genome browsers or ChIP-seq data, a peak covering a certain region of the genome

Major types of annotation in *Bioconductor* [CONT]

Sequence data: *Biostrings* and *BSgenome*

- Genomic sequences: *BSgenome.Hsapiens.UCSC.hg19*

For example: DNA/RNA sequences or motifs of transcription factor binding sites

Query web-based resources for annotation and experimental data, e.g., ENCODE, ROADMAP, $\cdots$, tracks

- *biomaRt* or *AnnotationHub*

import of genomic data in various formats like BED, BAM, FASTQ, VCF, $\cdots$

- *rtracklayer*, *Rsamtools*, *VariantAnnotation*

Meta data/Annotation

Bioconductor provides extensive annotation resource

- for associating microarray and other genomic data in real time with biological metadata from web databases such as GenBank, Entrez genes and PubMed
- covering a broad range of model organisms with support for different genomic builds
- updated every 6 months corresponding to the *Bioconductor* release cycle
- customized annotation libraries can also be assembled
- implementations are based on **SQLite** with a number of higher-level interfaces e.g., using a simplified version of SQL queries

Example: Mapping between gene identifiers

```
library(org.Hs.eg.db)
entrez_ids <- head(keys(org.Hs.eg.db, keytype="ENTREZID"))
entrez_ids
```

```
## [1] "1"  "2"  "3"  "9"  "10" "11"
```

```
select(org.Hs.eg.db, keys=entrez_ids, columns=c("SYMBOL","ENSEMBL"), keytype="ENTREZID")
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
##   ENTREZID SYMBOL         ENSEMBL
## 1        1   A1BG ENSG00000121410
## 2        2    A2M ENSG00000175899
## 3        3  A2MP1 ENSG00000256069
## 4        9   NAT1 ENSG00000171428
## 5       10   NAT2 ENSG00000156006
## 6       11   NATP            <NA>
```

Implementation

- gene centric databases (ENTREZ GENE ID)
- out-of-memory data storage (SQLite)
- fast access to data subsets (lower-level interface using SQL)
- general and simple high-level interface `columns`, `keys`, `keytype` and `select`

Further reading: *AnnotationDbi* vignettes:

"AnnotationDbi: Introduction To Bioconductor Annotation Packages" and

"How to use bimaps from the ".db" annotation packages"

Range data[1]

Core packages *IRanges*, *GenomicRanges* and *GenomicFeatures*

- directly supports more than 80 other *Bioconductor* packages, including those for sequence analysis, differential expression analysis and visualization
- provide scalable data structures for representing annotated ranges on the genome, with special support for transcript structures, read alignments and coverage vectors.
- computational facilities include efficient algorithms for overlap and nearest neighbor detection, coverage calculation and other range operations.

---

[1] Lawrence, M., Huber, W., Pages, H., Aboyoun, P., Carlson, M., Gentleman, R., Morgan, M., and Carey, V. (2013). Software for computing and annotating genomic ranges.
PLoS Comput. Biol., 9(8)

Example: Rle, IRanges and GRanges

```
(seqnames <- Rle(rep(c("chr1", "chr2"), c(1, 3))))
```

```
## character-Rle of length 4 with 2 runs
##   Lengths:      1      3
##   Values : "chr1" "chr2"
```

```
(ranges <- IRanges(1:4, end = 11:14, names = head(letters, 4)))
```

```
## IRanges object with 4 ranges and 0 metadata columns:
##         start       end     width
##     <integer> <integer> <integer>
##   a         1        11        11
##   b         2        12        11
##   c         3        13        11
##   d         4        14        11
```

```
GRanges(seqnames = seqnames, ranges = ranges, strand = Rle(strand(c("-", "+")),
    c(1, 3)), GC = seq(1, 0, length = 4))
```

```
## GRanges object with 4 ranges and 1 metadata column:
##     seqnames    ranges strand |                GC
##        <Rle> <IRanges>  <Rle> |         <numeric>
##   a     chr1   [1, 11]      - |                 1
##   b     chr2   [2, 12]      + | 0.666666666666667
##   c     chr2   [3, 13]      + | 0.333333333333333
##   d     chr2   [4, 14]      + |                 0
##   -------
##   seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

Example: Obtain transcript structure

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
kras_gene <- org.Hs.egSYMBOL2EG$KRAS
kras_gene
```

```
## [1] "3845"
```

```
kras_exons <- exons(TxDb.Hsapiens.UCSC.hg19.knownGene,
filter = list(gene_id = kras_gene),
columns = c("tx_id", "exon_id"))
kras_exons
```

```
## GRanges object with 8 ranges and 2 metadata columns:
##       seqnames              ranges strand |             tx_id   exon_id
##          <Rle>           <IRanges>  <Rle> |     <IntegerList> <integer>
##   [1]    chr12 [25358180, 25362845]      - |       47893,47894    168446
##   [2]    chr12 [25368371, 25368494]      - |             47893    168447
##   [3]    chr12 [25378548, 25378707]      - |       47893,47894    168448
##   [4]    chr12 [25380168, 25380346]      - |       47893,47894    168449
##   [5]    chr12 [25386768, 25388160]      - |             47895    168450
##   [6]    chr12 [25398208, 25398329]      - | 47893,47894,47895    168451
##   [7]    chr12 [25403685, 25403854]      - |       47893,47894    168452
##   [8]    chr12 [25403698, 25403863]      - |             47895    168453
##   -------
##   seqinfo: 93 sequences (1 circular) from hg19 genome
```

Example: Obtain transcript structure

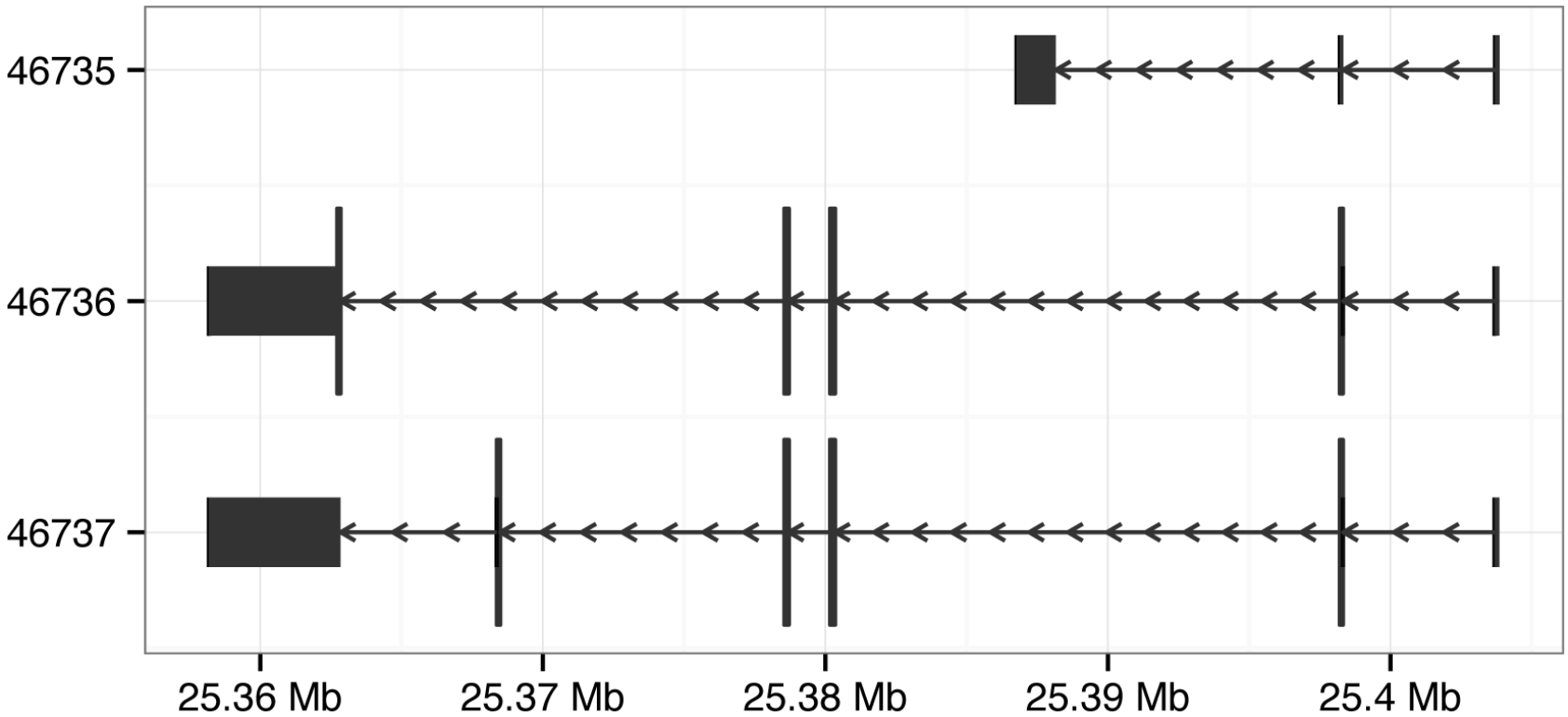

Figure : Representation of the exons for the human KRAS gene, derived from the UCSC known gene annotation.

Implementation

IRanges/GenomicRanges

- Run-length-encoding for efficient storage of range data
- IRange object are derived from IntegerList a list of integer vectors

One of the most power features are finding overlapping regions between `IRanges` or `GRanges`

- `findOverlaps` function uses an efficient interval tree algorithm
- the algorithm supports several types of overlap, including those defined by Allen's Interval Algebra
- the one-time cost of constructing the interval tree is $O(n \log n)$, and queries are performed in logarithmic time

Sequence data

Biostrings DNA, RNA and protein string manipulations

- counting and tabulating i.e., nucleotide frequencies
- sequence transformation and editing, i.e., translate DNA in RNA
- string matching/alignments, i.e., pattern matching
- I/O functions, i.e. read/write FASTA files

BSgenome

- 22 genomes with different builds e.g., H. sapiens has available builds:

```
##  [1] "BSgenome.Hsapiens.1000genomes.hs37d5"
##  [2] "BSgenome.Hsapiens.NCBI.GRCh38"
##  [3] "BSgenome.Hsapiens.UCSC.hg17"
##  [4] "BSgenome.Hsapiens.UCSC.hg17.masked"
##  [5] "BSgenome.Hsapiens.UCSC.hg18"
##  [6] "BSgenome.Hsapiens.UCSC.hg18.masked"
##  [7] "BSgenome.Hsapiens.UCSC.hg19"
##  [8] "BSgenome.Hsapiens.UCSC.hg19.masked"
##  [9] "BSgenome.Hsapiens.UCSC.hg38"
## [10] "BSgenome.Hsapiens.UCSC.hg38.masked"
```

- optionally *BSgenome* package can be generated

Example: Obtain nucleotide frequency of Human chr 1/count number of MseI restriction sites

```
library(BSgenome.Hsapiens.UCSC.hg19)
Hsapiens$chr1
```

```
##   249250621-letter "DNAString" instance
## seq: NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN...NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```

```
alphabetFrequency(Hsapiens$chr1)
```

```
##        A        C        G        T        M        R        W        S
## 65570891 47024412 47016562 65668756        0        0        0        0
##        Y        K        V        H        D        B        N        -
##        0        0        0        0        0        0 23970000        0
##        +        .
##        0        0
```

```
MseI <- "TTAA"
countPattern(MseI, Hsapiens$chr1)
```

```
## [1] 1428207
```

Implementation

1. use R external pointers to store the string data (references to C structures)
2. use bit patterns to encode the string data

Online resources

- genome browsers like UCSC and Ensembl are a rich resource for annotation of biological data
- large Consortia make their data available through genome browser e.g., HapMap, ENCODE, ROADMAP, $\cdots$

biomaRt, *rtracklayer* and *AnnotationHub*

- these packages provide easy access to public data repositories
- *rtracklayer* has functionality to import various genomic data formats: GFF, BED, Bed15, bedGraph, WIG, BigWig

biomaRt example: Get annotation from ENSEMBL

```
library(biomaRt)
ensembl <- useMart("ensembl",dataset="hsapiens_gene_ensembl")
getBM(attributes = c("hgnc_symbol", "ensembl_gene_id") ,
filters = "entrezgene",
values = entrez_ids, mart= ensembl)
```

```
##   hgnc_symbol ensembl_gene_id
## 1        A1BG ENSG00000121410
## 2        NAT2 ENSG00000156006
## 3         A2M ENSG00000175899
## 4       A2MP1 ENSG00000256069
## 5        NAT1 ENSG00000171428
```

rtracklayer example: Genome Segmentations track from ENCODE

```
library(rtracklayer)
genomicSegmentation <- import("wgEncodeAwgSegmentationChromhmmGm12878.bed", format="BED")
head(genomicSegmentation)
```

```
## GRanges object with 6 ranges and 4 metadata columns:
##       seqnames          ranges strand |        name     score     itemRgb
##          <Rle>       <IRanges>  <Rle> | <character> <numeric> <character>
##   [1]     chr1 [    1, 10000]      * |       Quies      1000     #E1E1E1
##   [2]     chr1 [10001, 10400]      * |      FaireW      1000     #FFFC04
##   [3]     chr1 [10401, 15800]      * |         Low      1000     #C2D69A
##   [4]     chr1 [15801, 16000]      * |        Pol2      1000     #00B050
##   [5]     chr1 [16001, 16400]      * |       Gen3'      1000     #00B050
##   [6]     chr1 [16401, 16600]      * |        Elon      1000     #00B050
##                thick
##             <IRanges>
##   [1] [    1, 10000]
##   [2] [10001, 10400]
##   [3] [10401, 15800]
##   [4] [15801, 16000]
##   [5] [16001, 16400]
##   [6] [16401, 16600]
##   -------
##   seqinfo: 23 sequences from an unspecified genome; no seqlengths
```

AnnotationHub example: Obtain ROADMAP chromatin segmentation tracks

```
library(AnnotationHub)
ah <- AnnotationHub()


## snapshotDate(): 2016-03-09

query(ah, c("EpigenomeRoadMap", "coreMarks"))


## AnnotationHub with 127 records
## # snapshotDate(): 2016-03-09
## # $dataprovider: BroadInstitute
## # $species: Homo sapiens
## # $rdataclass: GRanges
## # additional mcols(): taxonomyid, genome, description, tags,
## #   sourceurl, sourcetype
## # retrieve records with, e.g., 'object[["AH46856"]]'
##
##             title
##   AH46856 | E001_15_coreMarks_mnemonics.bed.gz
##   AH46857 | E002_15_coreMarks_mnemonics.bed.gz
##   AH46858 | E003_15_coreMarks_mnemonics.bed.gz
##   AH46859 | E004_15_coreMarks_mnemonics.bed.gz
##   AH46860 | E005_15_coreMarks_mnemonics.bed.gz
##   ...       ...
##   AH46978 | E125_15_coreMarks_mnemonics.bed.gz
##   AH46979 | E126_15_coreMarks_mnemonics.bed.gz
##   AH46980 | E127_15_coreMarks_mnemonics.bed.gz
##   AH46981 | E128_15_coreMarks_mnemonics.bed.gz
##   AH46982 | E129_15_coreMarks_mnemonics.bed.gz
```

AnnotationHub example: Obtain ROADMAP chromatin segmentation tracks[CONT]

```
ah['AH46982']
```

```
## AnnotationHub with 1 record
## # snapshotDate(): 2016-03-09
## # names(): AH46982
## # $dataprovider: BroadInstitute
## # $species: Homo sapiens
## # $rdataclass: GRanges
## # $title: E129_15_coreMarks_mnemonics.bed.gz
## # $description: 15 state chromatin segmentations from EpigenomeRoadMap ...
## # $taxonomyid: 9606
## # $genome: hg19
## # $sourcetype: BED
## # $sourceurl: http://egg2.wustl.edu/roadmap/data/byFileType/chromhmmSeg...
## # $sourcelastmodifieddate: 2013-10-11
## # $sourcesize: 2914216
## # $tags: EpigenomeRoadMap, chromhmmSegmentations, ChmmModels,
## #   coreMarks, E129, ENCODE2012, BONE.OSTEO, Osteoblast Primary
## #   Cells
## # retrieve record with 'object[["AH46982"]]'
```

AnnotationHub example: Obtain ROADMAP chromatin segmentation tracks[CONT]

```
gr <- ah[['AH46982']]


## loading from cache '/home/mvaniterson/.AnnotationHub/52422'


gr


## GRanges object with 511350 ranges and 4 metadata columns:
##             seqnames               ranges strand |        abbr
##                <Rle>            <IRanges>  <Rle> | <character>
##        [1]    chr10     [     1, 119600]      * |    15_Quies
##        [2]    chr10     [119601, 120200]      * |      1_TssA
##        [3]    chr10     [120201, 120400]      * |  2_TssAFlnk
##        [4]    chr10     [120401, 122000]      * |      5_TxWk
##        [5]    chr10     [122001, 122800]      * |      1_TssA
##        ...      ...                  ...    ... .         ...
##   [511346]     chrY [58984401, 58985800]      * |  8_ZNF/Rpts
##   [511347]     chrY [58985801, 58999400]      * |       9_Het
##   [511348]     chrY [58999401, 59001000]      * |    15_Quies
##   [511349]     chrY [59001001, 59033200]      * |       9_Het
##   [511350]     chrY [59033201, 59373400]      * |    15_Quies
##                         name        color_name  color_code
##                  <character>       <character> <character>
##        [1]       Quiescent/Low             White     #FFFFFF
##        [2]          Active TSS               Red     #FF0000
##        [3] Flanking Active TSS        Orange Red     #FF4500
##        [4]  Weak transcription         DarkGreen     #006400
##        [5]          Active TSS               Red     #FF0000
##        ...                 ...               ...         ...
##   [511346] ZNF genes & repeats Medium Aquamarine     #66CDAA
##   [511347]     Heterochromatin     PaleTurquoise     #8A91D0
##   [511348]       Quiescent/Low             White     #FFFFFF
```

Further Reading

- all vignettes: $>$ *vignette("packageName")*

- `http://www.bioconductor.org/help/workflows/annotation/annotation/`
- `http://www.bioconductor.org/help/workflows/variants/`
- `http://www.bioconductor.org/help/workflows/annotation/AnnotatingRanges/`

- `http://www.ebi.ac.uk/training/sites/ebi.ac.uk.training/files/materials/2013/131021_HTS/genesandgenomes.pdf`