High-level S4 containers for HTS data

Hervé Pagès
hpages@fhcrc.org

Fred Hutchinson Cancer Research Center
Seattle, WA

July 2013
Introduction

Most frequently seen low-level containers
  - Rle objects
  - IRanges objects
  - DataFrame objects
  - Other frequently seen low-level containers

GRanges objects
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects
  - Splitting a GRanges object
  - Exercise 1

GRangesList objects
  - GRangesList constructor and accessors
  - Vector operations on GRangesList objects
  - List operations on GRangesList objects
  - Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
  - GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
  - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

Advanced operations
  - Coverage and slicing
  - Finding/counting overlaps
  - Exercise 3

Final notes
Introduction

Most frequently seen low-level containers
  - Rle objects
  - IRanges objects
  - DataFrame objects

Other frequently seen low-level containers

GRanges objects
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects
  - Splitting a GRanges object
  - Exercise 1

GRangesList objects
  - GRangesList constructor and accessors
  - Vector operations on GRangesList objects
  - List operations on GRangesList objects
  - Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
  - GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
  - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

Advanced operations
  - Coverage and slicing
  - Finding/counting overlaps
  - Exercise 3

Final notes
There is only a small nb of high-level containers for HTS data (< 10)...

High-level containers for HTS data

Covered in this presentation (and all defined in the GenomicRanges package):

- GRanges
- GRangesList
- GAAlignments
- GAlignmentPairs

Not covered in this presentation:

- In the GenomicRanges package: SummarizedExperiment
- In the GenomicFeatures package: TranscriptDb
- In the ShortRead package: ShortRead, AlignedRead
- In the VariantAnnotation package: VCF

... but they are built on top of 100+ low-level containers!

Some of the most frequently seen low-level containers

Covered in this presentation (and all defined in the IRanges package):

- Rle
- IRanges
- DataFrame
- CharacterList, IntegerList
- RleList, RleViews, RleViewsList

Not covered in this presentation:

- In the IRanges package: IRangesList, Hits, SplitDataFrameList, and many more...
- In the GenomicRanges package: Seqinfo
- In the Biostrings package: DNAString, DNAStringSet, and many more...
About the implementation

S4 classes (a.k.a. *formal* classes) → relies heavily on the *methods* package.

Current implementation tries to provide an API that is as consistent as possible. In particular:

- The end-user should never need to use `new()`: a *constructor*, named as the container, is provided for each container. E.g. `GRanges()`.
- The end-user should never need to use `@` (a.k.a. *direct slot access*): slot *accessors* (*getters* and *setters*) are provided for each container. Not all getters have a corresponding setter!
- Standard functions/operators like `length()`, `names()`, `[`, `c()`, `[]`, `$`, etc... work almost everywhere and behave “as expected”.
- Additional functions that work almost everywhere: `mcols()`, `elementLengths()`, `seqinfo()`, etc...
- Consistent display (*show* methods).
Basic operations

Vector operations
Operate on vector-like objects
(e.g. on Rle, IRanges, GRanges, DNAStringSet, etc... objects)

- Accessors: length(), names(), mcols()
- Single-bracket subsetting: [
- Combining: c()
- Splitting/relisting: split(), relist()
- Comparing: ==, !=, match(), %in%, duplicated(), unique()
- Ordering: <=, >=, <, >, order(), sort(), rank()

List operations
Operate on list-like objects
(e.g. on IRangesList, GRangesList, DNAStringSetList, etc... objects)

- Double-bracket subsetting: [[
- elementLengths(), unlist()
- lapply(), sapply(), endoapply()
- mendoapply() (not covered in this presentation)

Coercion methods
- as()
- S3-style form: as.vector(), as.character(), as.factor(), etc...

*list-like objects are also vector-like objects*
Range-based operations

Range-based operations operate on range-based objects
(e.g. on IRanges, IRangesList, GRanges, GRangesList, etc... objects)

Intra range transformations
shift(), narrow(), flank(), resize()

Inter range transformations
disjoin(), range(), reduce(), gaps()

Range-based set operations
union(), intersect(), setdiff(), punion(), pintersect(), psetdiff(), pgap()

Coverage and slicing
coverage(), slice()

Finding/counting overlapping ranges
findOverlaps(), countOverlaps()

Finding the nearest range neighbor
nearest(), precede(), follow()

and more...
Introduction

Most frequently seen low-level containers
  - Rle objects
  - IRanges objects
  - DataFrame objects
  - Other frequently seen low-level containers

GRanges objects
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects
  - Splitting a GRanges object
  - Exercise 1

GRangesList objects
  - GRangesList constructor and accessors
  - Vector operations on GRangesList objects
  - List operations on GRangesList objects
  - Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
  - GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
  - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

Advanced operations
  - Coverage and slicing
  - Finding/counting overlaps
  - Exercise 3

Final notes
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects
- Other frequently seen low-level containers

GRanges objects

- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects

- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects

- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations

- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
Rle objects

Rle: Run Length Encoding

A substitute for ordinary atomic vectors that is (in most situations) more compact in memory.

In general, can be manipulated just like ordinary atomic vectors.

Supported operations

- Vector operations: **YES** (splitting/relisting produces an RleList object)
- List operations: **NO**
- Coercion methods: **YES** (to atomic vector, factor, or IRanges)
- Range-based operations: **NO**
Rle objects (continued)

> library(IRanges)
> set.seed(2013)
> rle1 <- Rle(sample(c(-4.9, 0), 20, replace=TRUE))
> rle1

numeric-Rle of length 20 with 7 runs
  Lengths:  1  3  1  5  2  6  2
  Values : -4.9  0  -4.9  0  -4.9  0  -4.9

> runLength(rle1)
[1] 1 3 1 5 2 6 2
> runValue(rle1)
[1] -4.9  0.0 -4.9  0.0 -4.9  0.0 -4.9

> as.vector(rle1)
[1] -4.9  0.0  0.0  0.0 -4.9  0.0  0.0  0.0  0.0  0.0 -4.9 -4.9  0.0  0.0  0.0  0.0
[17] 0.0  0.0 -4.9 -4.9
> rle1[c(TRUE, FALSE)]

numeric-Rle of length 10 with 7 runs
  Lengths:  1  1  1  1  2  1  3  1
  Values : -4.9  0  -4.9  0  -4.9  0  -4.9
Rle objects (continued)

```r
> sort(rle1)
numeric-Rle of length 20 with 2 runs
  Lengths:  6   14
  Values: -4.9  0

> rle1 * 5
numeric-Rle of length 20 with 7 runs
  Lengths:  1   3   1   5   2   6   2
  Values: -24.5  0 -24.5  0 -24.5  0 -24.5

> sum(rle1)
[1] -29.4

> cumsum(rle1)
numeric-Rle of length 20 with 6 runs
  Lengths:  4   6   1   7   1   1
  Values: -4.9 -9.8 -14.7 -19.6 -24.5 -29.4

> cumsum(rle1) <= -20
logical-Rle of length 20 with 2 runs
  Lengths: 18   2
  Values: FALSE  TRUE

> rle1[cumsum(rle1) <= -20]
numeric-Rle of length 2 with 1 run
  Lengths:  2
  Values: -4.9
```
Rle objects (continued)

```r
> rle2 <- Rle(c("ch1", "chMT", "ch1", "ch2", "chMT"), c(4, 2, 1, 5, 1))
> rle2
character-Rle of length 13 with 5 runs
   Lengths: 4 2 1 5 1
   Values : "ch1" "chMT" "ch1" "ch2" "chMT"

> as.vector(rle2)
[1] "ch1" "ch1" "ch1" "ch1" "chMT" "chMT" "ch1" "ch2" "ch2" "ch2" "ch2" "ch2" "chMT"

> c(rle2, c("chMT", "chX"))
character-Rle of length 15 with 6 runs
   Lengths: 4 2 1 5 2 1
   Values : "ch1" "chMT" "ch1" "ch2" "ch2" "ch2" "ch2" "ch2" "ch2" "chX"
```
> runValue(rle2) <- factor(runValue(rle2))
> rle2

factor-Rle of length 13 with 5 runs
  Lengths:  4  2  1  5  1
  Values:  ch1 chMT ch1 ch2 chMT
Levels(3): ch1 ch2 chMT

> runValue(rle2)
[1] ch1  chMT ch1  ch2  chMT
Levels: ch1 ch2 chMT

> as.vector(rle2)
[1] "ch1" "ch1" "ch1" "ch1" "chMT" "chMT" "ch1" "ch2" "ch2" "ch2" "ch2"
[12] "ch2" "chMT"

> as.factor(rle2)
[1] ch1  ch1  ch1  ch1  chMT chMT ch1  ch2  ch2  ch2  ch2  ch2  chMT
Levels: ch1 ch2 chMT
Rle objects (continued)

> rle1 == 0

logical-Rle of length 20 with 7 runs
Lengths:  1  3  1  5  2  6  2
Values : FALSE  TRUE  FALSE  TRUE  FALSE  TRUE  FALSE

> as(rle1 == 0, "IRanges")

IRanges of length 3
start end width
[1]  2  4  3
[2]  6 10  5
[3] 13 18  6
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects
- Other frequently seen low-level containers

GRanges objects
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
The purpose of the IRanges container is...

... to store a set of integer ranges (a.k.a. integer intervals).

- Each range can be defined by a start and an end value: both are included in the interval (except when the range is empty).
- The width of the range is the number of integer values in it: width = end - start + 1.
- end is always >= start, except for empty ranges (a.k.a. zero-width ranges) where end = start - 1.

**Supported operations**

- **Vector operations**: YES (splitting/relisting produces an IRangesList object)
- **List operations**: YES (not covered in this presentation)
- **Coercion methods**: YES (from logical or integer vector to IRanges)
- **Range-based operations**: YES
IRanges objects (continued)

```r
> ir1 <- IRanges(start=c(12, -9, NA, 12),
+                end=c(NA, 0, 15, NA),
+                width=c(4, NA, 4, 3))
> ir1  # "show" method not yet consistent with the other "show" methods (TODO)

IRanges of length 4
   start end width
[1]  12  15   4
[2] - 9   0  10
[3]  12  15   4
[4]  12  14   3

> start(ir1)
[1]  12 - 9  12  12

> end(ir1)
[1]  15   0  15  14

> width(ir1)
[1]   4  10   4  3

> successiveIRanges(c(10, 5, 38), from=101)

IRanges of length 3
   start end width
[1] 101 110  10
[2] 111 115  5
[3] 116 153 38
```
IRanges objects (continued)

```r
> ir1[-2]
IRanges of length 3
  start end width
[1]  12  15   4
[2]  12  15   4
[3]  12  14   3

> ir2 <- c(ir1, IRanges(-10, 0))
> ir2
IRanges of length 5
  start end width
[1]  12  15   4
[2]  -9  10  10
[3]  12  15   4
[4]  12  14   3
[5] -10  11  11

> duplicated(ir2)
[1] FALSE FALSE TRUE FALSE FALSE

> unique(ir2)
IRanges of length 4
  start end width
[1]  12  15   4
[2]  -9  10  10
[3]  12  14   3
[4] -10  11  11

> order(ir2)
[1]  5  2  4  1  3

> sort(ir2)
IRanges of length 5
  start end width
[1] -10  10  11
[2] -9  10  10
[3]  12  14  10
[4]  12  15  11
[5]  12  15  11

> ok <- c(FALSE, FALSE, TRUE, TRUE, TRUE, FALSE, FALSE, TRUE)
> ir4 <- as(ok, "IRanges") # from logical vector to IRanges
> ir4
IRanges of length 2
  start end width
[1]  3  5   3
[2]  8  8   1

> as.data.frame(ir4)
   start end width
1   3   5   3
2   8   8   1

> rle2[ir4] # IRanges subscript
factor-Rle of length 4 with 3 runs
  Lengths:  2  1  1
  Values: ch1 chMT ch2
  Levels(3): ch1 ch2 chMT
```
Range-based operations

ir0

shift(ir0, 5)

disjoin(ir0)

reduce(ir0)
IRanges objects (continued)

```r
> ir1
IRanges of length 4
  start end width
[1]  12  15   4
[2]  -9   0  10
[3]  12  15   4
[4]  12  14   3

> shift(ir1, -start(ir1))
IRanges of length 4
  start end width
[1]   0   3   4
[2]   0   9  10
[3]   0   3   4
[4]   0   2   3

> flank(ir1, 10, start=FALSE)
IRanges of length 4
  start end width
[1]  16  25  10
[2]   1  10  10
[3]  16  25  10
[4]  15  24  10
```
IRanges objects (continued)

> ir1

IRanges of length 4
   start   end   width
[1]   12    15     4
[2]    -9     0    10
[3]   12    15     4
[4]   12    14     3

> range(ir1)

IRanges of length 1
   start   end   width
[1]   -9    15    25

> reduce(ir1)

IRanges of length 2
   start   end   width
[1]   -9     0    10
[2]   12    15     4

> union(ir1, IRanges(-2, 6))

IRanges of length 2
   start   end   width
[1]   -9     6    16
[2]   12    15     4

> intersect(ir1, IRanges(-2, 13))

IRanges of length 2
   start   end   width
[1]    -2     0     3
[2]   12    13     2

> setdiff(ir1, IRanges(-2, 13))

IRanges of length 2
   start   end   width
[1]   -9    -3     7
[2]   14    15     2
IRanges objects (continued)

```r
> ir3 <- IRanges(5:1, width=12)
> ir3

IRanges of length 5

<table>
<thead>
<tr>
<th>start</th>
<th>end</th>
<th>width</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>16</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

> ir2

IRanges of length 5

<table>
<thead>
<tr>
<th>start</th>
<th>end</th>
<th>width</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>15</td>
<td>4</td>
</tr>
<tr>
<td>-9</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>15</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>-10</td>
<td>0</td>
<td>11</td>
</tr>
</tbody>
</table>

> pintersect(ir3, ir2, resolve.empty="max.start")

IRanges of length 5

<table>
<thead>
<tr>
<th>start</th>
<th>end</th>
<th>width</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>15</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```
Introduction

**Most frequently seen low-level containers**
- Rle objects
- IRanges objects
- **DataFrame objects**
- Other frequently seen low-level containers

**GRanges objects**
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

**GRangesList objects**
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

**Advanced operations**
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
DataFrame objects

DataFrame: An S4 version of `data.frame` that can hold almost anything it its columns.

**Supported operations**

- **Vector/List methods**: All the `data.frame` operations. Just manipulate a DataFrame as a `data.frame`!
- **Coercion methods**: from almost anything to DataFrame, and from DataFrame to `data.frame`.
- **Splitting/relisting**: **YES** (produces a `SplitDataFrameList` object)

```r
> library(Biostrings)
> dna <- DNAStringSet(c("AAA", "CATTNGAGC", "TAATAG"))
> af <- alphabetFrequency(dna, baseOnly=TRUE)
> df <- DataFrame(dna, af)
> df

DataFrame with 3 rows and 6 columns

<table>
<thead>
<tr>
<th>dna</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>other</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DNAStrIngSet&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;integer&gt;</td>
</tr>
<tr>
<td>1</td>
<td>AAA</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>CATTNGAGC</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>TAATAG</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

> df$G

[1] 0 2 1
```
DataFrame objects (continued)

```r
> df$cds_id <- paste("CDS", 1:3, sep="")
> df$cds_range <- successiveIRanges(width(dna), from=51)
> df

Data frame with 3 rows and 8 columns

<table>
<thead>
<tr>
<th>dna</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>other</th>
<th>cds_id</th>
<th>cds_range.start</th>
<th>cds_range.end</th>
<th>cds_range.width</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CDS1</td>
<td>51</td>
<td>53</td>
<td>3</td>
</tr>
<tr>
<td>CATTNGAGC</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>CDS2</td>
<td>54</td>
<td>62</td>
<td>9</td>
</tr>
<tr>
<td>TAATAG</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>CDS3</td>
<td>63</td>
<td>68</td>
<td>6</td>
</tr>
</tbody>
</table>

> as.data.frame(df)

<table>
<thead>
<tr>
<th>dna</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>other</th>
<th>cds_id</th>
<th>cds_range.start</th>
<th>cds_range.end</th>
<th>cds_range.width</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>CDS1</td>
<td>51</td>
<td>53</td>
<td>3</td>
</tr>
<tr>
<td>CATTNGAGC</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>CDS2</td>
<td>54</td>
<td>62</td>
<td>9</td>
</tr>
<tr>
<td>TAATAG</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>CDS3</td>
<td>63</td>
<td>68</td>
<td>6</td>
</tr>
</tbody>
</table>
```
Introduction

**Most frequently seen low-level containers**
- `Rle` objects
- `IRanges` objects
- `DataFrame` objects

**Other frequently seen low-level containers**

**GRanges objects**
- `GRanges` constructor and accessors
- Vector operations on `GRanges` objects
- Range-based operations on `GRanges` objects
- Splitting a `GRanges` object
- Exercise 1

**GRangesList objects**
- `GRangesList` constructor and accessors
- Vector operations on `GRangesList` objects
- List operations on `GRangesList` objects
- Range-based operations on `GRangesList` objects

**GAlignments and GAlignmentPairs objects**
- `GAlignments`
- `GAlignments` constructor and accessors
- Exercise 2
- From `GAlignments` to `GRanges` or `GRangesList`
- `GAlignmentPairs`

**Advanced operations**
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
CharacterList objects

An S4 virtual class for representing a list of character vectors.

```r
> x <- CharacterList(one=c("aaa", "bb", "c"), two=c("dd", "e", "fff", "gggg"))

> x
CharacterList of length 2
[["one"]]
[1] "aaa" "bb" "c"
[["two"]]
[1] "dd" "e" "fff" "gggg"

> length(x)
[1] 2

> names(x)
[1] "one" "two"

> as.list(x)

$one
[1] "aaa" "bb" "c"

$two
[1] "dd" "e" "fff" "gggg"

> x[[2]]
[1] "dd" "e" "fff" "gggg"

Exist in 2 flavors (i.e. 2 different internal representations):

▶ CompressedCharacterList
▶ SimpleCharacterList

> class(x)
[1] "CompressedCharacterList" attr("package")
[1] "IRanges"
CharacterList objects (continued)

```r
> toupper(x)

CharacterList of length 2
[['one']] AAA BB C
[['two']] DD E FFF GGGG

> elementLengths(x) # fast version of sapply(x, length)
one two
  3  4

> unlist(x)

  one one one two two two two two
"aaa" "bb" "c" "dd" "e" "fff" "gggg"

> unlist(x, use.names=FALSE)

[1] "aaa" "bb" "c" "dd" "e" "fff" "gggg"
```
**IntegerList objects**

An S4 virtual class for representing a list of integer vectors.

```r
> x <- IntegerList(6:-2, 5, integer(0), 14:21)
> x

IntegerList of length 4
[[1]] 6 5 4 3 2 1 0 -1 -2
[[2]] 5
[[3]] integer(0)
[[4]] 14 15 16 17 18 19 20 21

> x * x

IntegerList of length 4
[[1]] 36 25 16 9 4 1 0 1 4
[[2]] 25
[[3]] integer(0)
[[4]] 196 225 256 289 324 361 400 441
```

Exist in 2 flavors (i.e. 2 different internal representations):

- CompressedIntegerList
- SimpleIntegerList

```r
> class(x)
```
2 different ways to obtain the same result:

```r
> x * 100L - 2L

IntegerList of length 4
[[1]] 598 498 398 298 198 98 -2 -102 -202
[[2]] 498
[[3]] integer(0)
[[4]] 1398 1498 1598 1698 1798 1898 1998 2098

> relist(unlist(x) * 100L - 2L, x)

IntegerList of length 4
[[1]] 598 498 398 298 198 98 -2 -102 -202
[[2]] 498
[[3]] integer(0)
[[4]] 1398 1498 1598 1698 1798 1898 1998 2098
```

But the above trick would not work here:

```r
> cumsum(x)

NumericList of length 4
[[1]] 6 11 15 18 20 21 21 20 18
[[2]] 5
[[3]] numeric(0)
[[4]] 14 29 45 62 80 99 119 140
```
RleList, RleViews and RleViewsList objects

Typically seen when doing *Coverage and slicing* (more on this later).

**RleList**: An S4 virtual class for representing a list of Rle objects. Exist in 2 flavors (i.e. 2 different internal representations):
- CompressedRleList
- SimpleRleList

**RleViews**: An S4 class for representing a set of *views* (i.e. ranges) defined on an Rle subject.

**RleViewsList**: An S4 virtual class for representing a list of RleViews objects. Exist only in 1 flavor: SimpleRleViewsList.
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

**GRanges objects**
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

**GRangesList objects**
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
  - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

**Advanced operations**
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
The purpose of the GRanges container is...

... to store a set of genomic ranges (a.k.a. genomic regions or genomic intervals).

- Like for IRanges objects, each range can be defined by a *start* and an *end* value.
- In addition, each range is also assigned a chromosome name and a strand.
- *start* and *end* are both 1-based positions relative to the 5’ end of the plus strand of the chromosome (a.k.a. reference sequence), even when the range is on the minus strand.
- So the *start* is always the leftmost position and the *end* the rightmost, even when the range is on the minus strand.
- As a consequence, if a genomic range represents a gene on the minus strand, the gene “starts” (biologically speaking) at the end of it.

**Supported operations**

- Vector operations: **YES** (splitting/relisting produces a GRangesList object)
- List operations: **NO**
- Coercion methods: to IRangesList (not covered in this presentation)
- Range-based operations: **YES**
Introduction
Most frequently seen low-level containers
    Rle objects
    IRanges objects
    DataFrame objects
Other frequently seen low-level containers

GRanges objects
    GRanges constructor and accessors
    Vector operations on GRanges objects
    Range-based operations on GRanges objects
    Splitting a GRanges object
    Exercise 1

GRangesList objects
    GRangesList constructor and accessors
    Vector operations on GRangesList objects
    List operations on GRangesList objects
    Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
    GAlignments
    GAlignments constructor and accessors
    Exercise 2
    From GAlignments to GRanges or GRangesList
    GAlignmentPairs

Advanced operations
    Coverage and slicing
    Finding/counting overlaps
    Exercise 3

Final notes
GRanges constructor

```r
> library(GenomicRanges)
> gr1 <- GRanges(seqnames=Rle(c("ch1", "chMT"), lengths=c(2, 4)),
+    ranges=IRanges(start=16:21, end=20),
+    strand=rep(c("+", "-", "*"), 2))
> gr1

GRanges with 6 ranges and 0 metadata columns:

  seqnames ranges strand
  <Rle> <IRanges> <Rle>
     [1]    ch1    [16, 20]     +
     [2]    ch1    [17, 20]     -

---

seqlengths:
  ch1  chMT
    NA   NA
```
GRanges accessors

> length(gr1)
[1] 6

> seqnames(gr1)

factor-Rle of length 6 with 2 runs
  Lengths:  2  4
  Values:  ch1 chMT
Levels(2):  ch1 chMT

> ranges(gr1)

IRanges of length 6
  start end width
  [1]  16  20   5
  [2]  17  20   4
  [3]  18  20   3
  [4]  19  20   2
  [5]  20  20   1
  [6]  21  20   0
> start(gr1)
[1] 16 17 18 19 20 21
> end(gr1)
[1] 20 20 20 20 20 20
> width(gr1)
[1] 5 4 3 2 1 0
> strand(gr1)

factor-Rle of length 6 with 6 runs
  Lengths: 1 1 1 1 1 1
  Values : + - * + - *
Levels(3): + - *

> strand(gr1) <- c("-", "-", "+")
> strand(gr1)

factor-Rle of length 6 with 4 runs
  Lengths: 2 1 2 1
  Values : - + - +
Levels(3): + - *
GRanges accessors (continued)

> names(gr1) <- LETTERS[1:6]
> names(gr1)

[1] "A" "B" "C" "D" "E" "F"

> mcols(gr1) <- DataFrame(score=11:16, GC=seq(1, 0, length=6))
> mcols(gr1)

Dataframe with 6 rows and 2 columns

<table>
<thead>
<tr>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>1.0</td>
</tr>
<tr>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>15</td>
<td>0.2</td>
</tr>
<tr>
<td>16</td>
<td>0.0</td>
</tr>
</tbody>
</table>

> gr1

GRanges with 6 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1</td>
<td>[-]</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>[-]</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>[+]</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>[-]</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>E</td>
<td>chMT</td>
<td>[-]</td>
<td>15</td>
<td>0.2</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>[+]</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>
> seqinfo(gr1)
Seqinfo of length 2
seqnames seqlengths isCircular genome
  ch1     NA     NA  <NA>
  chMT    NA     NA  <NA>

> seqlevels(gr1)
[1] "ch1" "chMT"

> seqlengths(gr1)
  ch1  chMT
  NA   NA

> seqlengths(gr1) <- c(50000, 800)
> seqlengths(gr1)
  ch1  chMT
  50000 800
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

**GRanges objects**

- GRanges constructor and accessors

**Vector operations on GRanges objects**

- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

**GRangesList objects**

- GRangesList constructor and accessors

**Vector operations on GRangesList objects**

- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**

- GAlignments

**GAlignments constructor and accessors**

- Exercise 2

- From GAlignments to GRanges or GRangesList

**GAlignmentPairs**

**Advanced operations**

- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
Vector operations on GRanges objects

```r
> gr1[c("F", "A")]

GRanges with 2 ranges and 2 metadata columns:
  seqnames ranges strand | score  GC
   <Rle> <IRanges> <Rle> | <integer> <numeric>
    F   chMT [21, 20]  + |   16 0
    A   ch1  [16, 20]  - |   11 1

---

seqlengths:
   ch1   chMT
    50000   800

> gr1[strand(gr1) == "+"]

GRanges with 2 ranges and 2 metadata columns:
  seqnames ranges strand | score  GC
   <Rle> <IRanges> <Rle> | <integer> <numeric>
    C   chMT [18, 20]  + |   13 0.6
    F   chMT [21, 20]  + |   16 0

---

seqlengths:
   ch1   chMT
    50000   800
```
> gr1 <- gr1[-5]
> gr1

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [16, 20]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 20]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [21, 20]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
Vector operations on GRanges objects (continued)

```r
> gr2 <- GRanges(seqnames="ch2",
+                 ranges=IRanges(start=c(2:1,2), width=6),
+                 score=15:13,
+                 GC=seq(0, 0.4, length=3))
> gr12 <- c(gr1, gr2)
> gr12

GRanges with 8 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
```
Vector operations on GRanges objects (continued)

> gr12[length(gr12)] == gr12

[1] FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE

> duplicated(gr12)

[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE

> unique(gr12)

GRanges with 7 ranges and 2 metadata columns:

```
  seqnames ranges strand | score   GC
        <Rle> <IRanges> <Rle> | <integer> <numeric>
A    ch1     [16, 20] - | 11   1
B    ch1     [17, 20] - | 12   0.8
C   chMT    [18, 20] + | 13   0.6
D   chMT    [19, 20] - | 14   0.4
F   chMT    [21, 20] + | 16   0
     ch2     [ 2, 7] * | 15   0
     ch2     [ 1, 6] * | 14   0.2
```

---

seqlengths:
```
  ch1    chMT    ch2
50000   800      NA
```
> sort(gr12)

GRanges with 8 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1</td>
<td>[16, 20]</td>
<td>-</td>
<td>11</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>[17, 20]</td>
<td>-</td>
<td>12</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>[18, 20]</td>
<td>+</td>
<td>13</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>[21, 20]</td>
<td>+</td>
<td>16</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>[19, 20]</td>
<td>-</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>ch2</td>
<td>[ 1, 6]</td>
<td>*</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>ch2</td>
<td>[ 2, 7]</td>
<td>*</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>ch2</td>
<td>[ 2, 7]</td>
<td>*</td>
<td>13</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th></th>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch1</td>
<td>50000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chMT</td>
<td>800</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ch2</td>
<td>NA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

**GRanges objects**
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

**GRangesList objects**
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

**Advanced operations**
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
> gr2

GRanges with 3 ranges and 2 metadata columns:

```
<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>
```

---

seqlengths:

```
ch2
NA
```

> shift(gr2, 50)

GRanges with 3 ranges and 2 metadata columns:

```
<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[52, 57]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[51, 56]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[52, 57]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>
```

---

seqlengths:

```
ch2
NA
```

> narrow(gr2, start=2, end=-2)

GRanges with 3 ranges and 2 metadata columns:

```
<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[3, 6]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 5]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[3, 6]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>
```

---

seqlengths:

```
ch2
NA
```
> gr1

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [16, 20]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 20]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [21, 20]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>

> resize(gr1, 12)

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [ 9, 20]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [ 9, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 29]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [ 9, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [21, 32]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
Range-based operations on GRanges objects (continued)

> gr1

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;integer&gt;</td>
</tr>
<tr>
<td>A</td>
<td>ch1</td>
<td>[16, 20]</td>
<td>-</td>
<td>11</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>[17, 20]</td>
<td>-</td>
<td>12</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>[18, 20]</td>
<td>+</td>
<td>13</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>[19, 20]</td>
<td>-</td>
<td>14</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>[21, 20]</td>
<td>+</td>
<td>16</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>

> flank(gr1, 3)

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;integer&gt;</td>
</tr>
<tr>
<td>A</td>
<td>ch1</td>
<td>[21, 23]</td>
<td>-</td>
<td>11</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>[21, 23]</td>
<td>-</td>
<td>12</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>[15, 17]</td>
<td>+</td>
<td>13</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>[21, 23]</td>
<td>-</td>
<td>14</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>[18, 20]</td>
<td>+</td>
<td>16</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
Range-based operations on GRanges objects (continued)

```r
> gr3 <- shift(gr1, c(35000, rep(0, 3), 100))
> width(gr3)[c(3,5)] <- 117
> gr3

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [ 17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [ 18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [ 19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [ 121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>

> range(gr3)

GRanges with 3 ranges and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [17, 35020]</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>chMT [18, 237]</td>
<td>+</td>
</tr>
<tr>
<td>C</td>
<td>chMT [19, 20]</td>
<td>-</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
```
Range-based operations on GRanges objects (continued)

```
> gr3

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;integer&gt;</td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th></th>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>

> disjoin(gr3)

GRanges with 6 ranges and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
</tr>
<tr>
<td>[1]</td>
<td>ch1 [17, 20]</td>
<td>-</td>
</tr>
<tr>
<td>[2]</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
</tr>
<tr>
<td>[3]</td>
<td>chMT [18, 120]</td>
<td>+</td>
</tr>
<tr>
<td>[4]</td>
<td>chMT [121, 134]</td>
<td>+</td>
</tr>
<tr>
<td>[6]</td>
<td>chMT [19, 20]</td>
<td>-</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th></th>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
Range-based operations on GRanges objects (continued)

> gr3

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt; &lt;Rle&gt;</td>
<td></td>
<td>&lt;integer&gt;</td>
<td>&lt;numeric&gt;</td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>

> reduce(gr3)

GRanges with 4 ranges and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt;</td>
<td></td>
</tr>
<tr>
<td>[1]</td>
<td>ch1 [17, 20]</td>
<td>-</td>
</tr>
<tr>
<td>[2]</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
</tr>
<tr>
<td>[3]</td>
<td>chMT [18, 237]</td>
<td>+</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
</tr>
</tbody>
</table>
> gr3

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqlengths:</th>
<th>ch1 chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000 800</td>
<td></td>
</tr>
</tbody>
</table>

---

> gaps(gr3)

GRanges with 10 ranges and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqlengths:</th>
<th>ch1 chMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000 800</td>
<td></td>
</tr>
</tbody>
</table>
Introduction

Most frequently seen low-level containers
  - Rle objects
  - IRanges objects
  - DataFrame objects
  - Other frequently seen low-level containers

**GRanges objects**
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects

**Splitting a GRanges object**

Exercise 1

**GRangesList objects**
  - GRangesList constructor and accessors
  - Vector operations on GRangesList objects
  - List operations on GRangesList objects
  - Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
  - GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
    - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

**Advanced operations**
  - Coverage and slicing
  - Finding/counting overlaps

Exercise 3

**Final notes**
Splitting a GRanges object

```r
> split(gr3, seqnames(gr3))

GRangesList of length 2:
$ch1
GRanges with 2 ranges and 2 metadata columns:
  seqnames ranges strand | score GC
    <Rle>   <IRanges> <Rle> | <integer> <numeric>
    A ch1 [35016, 35020] - | 11 1
    B ch1 [ 17, 20] - | 12 0.8

$chMT
GRanges with 3 ranges and 2 metadata columns:
  seqnames ranges strand | score GC
    <Rle>   <IRanges> <Rle> | <integer> <numeric>
    C chMT [ 18, 134] + | 13 0.6
    D chMT [ 19, 20] - | 14 0.4
    F chMT [121, 237] + | 16 0

---

seqlengths:
  ch1  chMT
  50000  800
```
Introduction
Most frequently seen low-level containers
Rle objects
IRanges objects
DataFrame objects
Other frequently seen low-level containers

**GRanges objects**
GRanges constructor and accessors
Vector operations on GRanges objects
Range-based operations on GRanges objects
Splitting a GRanges object

**Exercise 1**

GRangesList objects
GRangesList constructor and accessors
Vector operations on GRangesList objects
List operations on GRangesList objects
Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
GAlignments
GAlignments constructor and accessors
Exercise 2
From GAlignments to GRanges or GRangesList
GAlignmentPairs

**Advanced operations**
Coverage and slicing
Finding/counting overlaps

**Exercise 3**

**Final notes**
Exercise 1

a. Load the *GenomicRanges* package.
b. Open the man page for the *GRanges* class and run the examples in it.
c. Shift the ranges in *gr* by 1000 positions to the right.
d. What method is called when doing `shift()` on a *GRanges* object? Find the man page for this method.
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

**GRanges objects**

- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

**GRangesList objects**

- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**

- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

**Advanced operations**

- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

**Final notes**
The purpose of the GRangesList container is...

... to store a list of compatible GRanges objects.

compatible means:

▶ they are relative to the same genome,
▶ AND they have the same metadata columns (accessible with the mcols() accessor).

Supported operations

▶ Vector operations: partially supported (no splitting/relisting, no comparing or ordering)
▶ List operations: YES
▶ Coercion methods: to IRangesList (not covered in this presentation)
▶ Range-based operations: partially supported (some operations like gaps() are missing but they could/will be added)
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects
Other frequently seen low-level containers

GRanges objects
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
- GAlignments constructor and accessors
Exercise 2
- From GAlignments to GRanges or GRangesList
GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps
Exercise 3

Final notes
> grl <- GRangesList(gr3, gr2)
> grl

GRangesList of length 2:
[[1]]
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

[[2]]
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
GRangesList accessors

> length(grl)
[1] 2

> seqnames(grl)
RleList of length 2
[[1]]
factor-Rle of length 5 with 2 runs
  Lengths:  2  3
  Values : ch1 chMT
Levels(3): ch1 chMT ch2

[[2]]
factor-Rle of length 3 with 1 run
  Lengths:  3
  Values : ch2
Levels(3): ch1 chMT ch2

> strand(grl)
RleList of length 2
[[1]]
factor-Rle of length 5 with 4 runs
  Lengths:  2  1  1  1
  Values : - + - +
Levels(3): + - *

[[2]]
factor-Rle of length 3 with 1 run
  Lengths:  3
  Values : *
Levels(3): + - *
GRangesList accessors (continued)

\[
\begin{array}{lllll}
\text{start} & \text{end} & \text{width} & \text{names} \\
35016 & 35020 & 5 & A \\
17 & 20 & 4 & B \\
18 & 134 & 117 & C \\
19 & 20 & 2 & D \\
121 & 237 & 117 & F \\
\end{array}
\]

\[
\begin{array}{llll}
\text{start} & \text{end} & \text{width} \\
2 & 7 & 6 \\
1 & 6 & 6 \\
2 & 7 & 6 \\
\end{array}
\]

> ranges(grl)

IRangesList of length 2
[[1]]
IRanges of length 5
,start end width names
[1] 35016 35020 5 A
[2] 17 20 4 B
[3] 18 134 117 C
[4] 19 20 2 D
[5] 121 237 117 F

[[2]]
IRanges of length 3
,start end width names
[1] 2 7 6
[2] 1 6 6
[3] 2 7 6

> start(grl)

IntegerList of length 2
[[1]] 35016 17 18 19 121
[[2]] 2 1 2

> end(grl)

IntegerList of length 2
[[1]] 35020 20 134 20 237
[[2]] 7 6 7

> width(grl)

IntegerList of length 2
[[1]] 5 4 117 2 117
[[2]] 6 6 6
> names(grl) <- c("TX1", "TX2")
> grl

GRangesList of length 2:

$TX1

GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [   17,  20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [   18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [   19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [ 121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2

GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>ch2 [1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
GRangesList accessors (continued)

```r
> mcols(grl)$geneid <- c("GENE1", "GENE2")
> mcols(grl)

DataFrame with 2 rows and 1 column
geneid
  <character>
1  GENE1
2  GENE2

> grl

GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td></td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2 [1, 6]</td>
<td>*</td>
<td></td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td></td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:
<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
```
GRangesList accessors (continued)

```r
> seqinfo(grl)

Seqinfo of length 3
seqnames  seqlengths  isCircular  genome
  ch1         50000      NA      <NA>
  chMT         800       NA      <NA>
  ch2         NA       NA      <NA>
```
Introduction

Most frequently seen low-level containers
  Rle objects
  IRanges objects
  DataFrame objects
Other frequently seen low-level containers

GRanges objects
  GRanges constructor and accessors
  Vector operations on GRanges objects
  Range-based operations on GRanges objects
  Splitting a GRanges object
  Exercise 1

GRangesList objects
  GRangesList constructor and accessors
  Vector operations on GRangesList objects
  List operations on GRangesList objects
  Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
  GAlignments
  GAlignments constructor and accessors
  Exercise 2
  From GAlignments to GRanges or GRangesList
  GAlignmentPairs

Advanced operations
  Coverage and slicing
  Finding/counting overlaps
  Exercise 3

Final notes
Vector operations on GRangesList objects

> grl[c("TX2", "TX1")]

GRangesList of length 2:
$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
Vector operations on `GRangesList` objects (continued)

```r
> c(gr1, GRangesList(gr3))

GRangesList of length 3:

$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ch2 [1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

[[3]]
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
Introduction

Most frequently seen low-level containers
  - Rle objects
  - IRanges objects
  - DataFrame objects

Other frequently seen low-level containers

GRanges objects
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects
  - Splitting a GRanges object

Exercise 1

GRangesList objects
  - GRangesList constructor and accessors
  - Vector operations on GRangesList objects

List operations on GRangesList objects
  - Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
  - GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
    - From GAlignments to GRanges or GRangesList
  - GAlignmentPairs

Advanced operations
  - Coverage and slicing
  - Finding/counting overlaps

Exercise 3

Final notes
List operations on GRangesList objects

```r
> grl[[2]]

GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;numeric&gt;</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>

> elementLengths(grl)

TX1 TX2
5 3

> unlisted <- unlist(grl, use.names=FALSE)  # same as c(grl[[1]], grl[[2]])
> unlisted

GRanges with 8 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
<td>&lt;integer&gt;</td>
<td>&lt;numeric&gt;</td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>ch2 [1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
```
List operations on GRangesList objects (continued)

> grl100 <- relist(shift(unlisted, 100), grl)
> grl100

GRangesList of length 2:

$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35116, 35120]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [ 117, 120]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [ 118, 234]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [ 119, 120]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [ 221, 337]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2</td>
<td>[102, 107]</td>
<td>*</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>ch2</td>
<td>[101, 106]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
</tr>
<tr>
<td>ch2</td>
<td>[102, 107]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
List operations on GRangesList objects (continued)

```r
> grl100b <- endoapply(grl, shift, 100)
> grl100b

GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:
  seqnames ranges strand score GC
   <Rle> <IRanges> <Rle> | <integer> <numeric>
A  ch1 [ 35116, 35120] - |  11  1
B  ch1 [  117,  120] - |  12  0.8
C  chMT [  118,  234] + |  13  0.6
D  chMT [  119,  120] - |  14  0.4
F  chMT [  221,  337] + |  16  0

$TX2
GRanges with 3 ranges and 2 metadata columns:
  seqnames ranges strand score GC
   <Rle> <IRanges> <Rle> | <integer> <numeric>
B  ch2 [  102,  107] * |  15  0
C  ch2 [  101,  106] * |  14  0.2
D  ch2 [  102,  107] * |  13  0.4

---

seqlengths:
  ch1  chMT  ch2
50000 800   NA

> mcols(grl100)

DataFrame with 2 rows and 0 columns

> mcols(grl100b)

DataFrame with 2 rows and 1 column
geneid
  <character>
1   GENE1
2   GENE2
```
Introduction
Most frequently seen low-level containers
   Rle objects
   IRanges objects
   DataFrame objects
Other frequently seen low-level containers
GRanges objects
   GRanges constructor and accessors
   Vector operations on GRanges objects
   Range-based operations on GRanges objects
   Splitting a GRanges object
   Exercise 1
GRangesList objects
   GRangesList constructor and accessors
   Vector operations on GRangesList objects
   List operations on GRangesList objects
   Range-based operations on GRangesList objects
GAlignments and GAlignmentPairs objects
   GAlignments
   GAlignments constructor and accessors
   Exercise 2
   From GAlignments to GRanges or GRangesList
   GAlignmentPairs
Advanced operations
   Coverage and slicing
   Finding/counting overlaps
   Exercise 3
Final notes
Range-based operations on GRangesList objects

```
> grl

GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt; &lt;Rle&gt;</td>
<td></td>
<td>&lt;integer&gt; &lt;numeric&gt;</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35016, 35020] -</td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ch1 [ 17, 20] -</td>
<td>12</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>chMT [ 18, 134] +</td>
<td>13</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>chMT [ 19, 20] -</td>
<td>14</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>chMT [ 121, 237] +</td>
<td>16</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt; &lt;Rle&gt;</td>
<td></td>
<td>&lt;integer&gt; &lt;numeric&gt;</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35116, 35120] -</td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ch1 [117, 120] -</td>
<td>12</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>chMT [118, 234] +</td>
<td>13</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>chMT [119, 120] -</td>
<td>14</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>chMT [221, 337] +</td>
<td>16</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
```

```
> shift(grl, 100)

GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt; &lt;Rle&gt;</td>
<td></td>
<td>&lt;integer&gt; &lt;numeric&gt;</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>ch1 [35116, 35120] -</td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ch1 [ 117, 120] -</td>
<td>12</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>chMT [ 118, 134] +</td>
<td>13</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>chMT [ 119, 120] -</td>
<td>14</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>chMT [ 221, 237] +</td>
<td>16</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;Rle&gt; &lt;IRanges&gt; &lt;Rle&gt;</td>
<td></td>
<td>&lt;integer&gt; &lt;numeric&gt;</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>ch2 [102, 107] *</td>
<td>15</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>ch2 [101, 106] *</td>
<td>14</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>ch2 [102, 107] *</td>
<td>13</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
</tr>
</tbody>
</table>
```

shift(grl, 100) is equivalent to endoapply(grl, shift, 100)
Range-based operations on GRangesList objects (continued)

```r
> grl
GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35021, 35030]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [21, 30]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [8, 17]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [21, 30]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [111, 120]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

seqlengths:

<table>
<thead>
<tr>
<th>seqlengths</th>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
<td></td>
</tr>
</tbody>
</table>
```

```r
> flank(grl, 10)
GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35021, 35030]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [21, 30]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [8, 17]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [21, 30]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [111, 120]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>15</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ch2 [1, 6]</td>
<td>*</td>
<td>14</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>ch2 [2, 7]</td>
<td>*</td>
<td>13</td>
<td>0.4</td>
<td></td>
</tr>
</tbody>
</table>

seqlengths:

<table>
<thead>
<tr>
<th>seqlengths</th>
<th>ch1</th>
<th>chMT</th>
<th>ch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>800</td>
<td>NA</td>
<td></td>
</tr>
</tbody>
</table>
```

flank(grl, 10) is equivalent to endoapply(grl, flank, 10)
Range-based operations on GRangesList objects (continued)

```r
> grl
GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:

```

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>
```

$TX2
GRanges with 3 ranges and 2 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>score</th>
<th>GC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ch1 [35016, 35020]</td>
<td>-</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>ch1 [17, 20]</td>
<td>-</td>
<td>12</td>
<td>0.8</td>
</tr>
<tr>
<td>C</td>
<td>chMT [18, 134]</td>
<td>+</td>
<td>13</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>chMT [19, 20]</td>
<td>-</td>
<td>14</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>chMT [121, 237]</td>
<td>+</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

```
ch1 50000
chMT 800
ch2 NA
```

> range(grl)
GRangesList of length 2:
$TX1
GRanges with 3 ranges and 0 metadata columns:

```
[1] ch1 [17, 35020] -
[2] chMT [18, 237] +
[3] chMT [19, 20] -
```

$TX2
GRanges with 1 range and 0 metadata columns:

```
[1] ch2 [1, 7] *
```

---

seqlengths:

```
ch1 50000
chMT 800
ch2 NA
```

range(grl) is equivalent to endoapply(grl, range)
Range-based operations on GRangesList objects (continued)

```r
> grl
GRangesList of length 2:
$TX1
GRanges with 5 ranges and 2 metadata columns:
       seqnames ranges strand  |  score  GC
         <Rle>  <IRanges>  <Rle>  | <integer> <numeric>
A       ch1 [35016, 35020] - |  11  1
B       ch1 [ 17,  20]     - |  12  0.8
C       chMT [ 18,  134]  + |  13  0.6
D       chMT [ 19,  20]  - |  14  0.4
F       chMT [ 121,  237] + |  16  0

$TX2
GRanges with 3 ranges and 2 metadata columns:
       seqnames ranges strand  |  score  GC
         <Rle>  <IRanges>  <Rle>  | <integer> <numeric>
        ch2 [ 2,  7]  *  |  15  0
        ch2 [ 1,  6]  *  |  14  0.2
        ch2 [ 2,  7]  *  |  13  0.4

---
seqlengths:
   ch1  chMT  ch2
     50000  800  NA
```

```r
> reduce(grl)
GRangesList of length 2:
$TX1
GRanges with 4 ranges and 0 metadata columns:
       seqnames ranges strand
         <Rle>  <IRanges>  <Rle>
[1]   ch1 [ 17,  20]  -
[2]   ch1 [35016, 35020]  -
[3]  chMT [ 18,  237]  +

$TX2
GRanges with 1 range and 0 metadata columns:
      seqnames ranges strand
        [1]    ch2 [ 1,  7]  *

---
seqlengths:
   ch1  chMT  ch2
     50000  800  NA
```

**reduce(grl) is equivalent to endoapply(grl, reduce)**
Range-based operations on GRangesList objects (continued)

```r
> grl2
GRangesList of length 2:
$TX1
GRanges with 1 range and 2 metadata columns:
  seqnames  ranges  strand  |  score  GC
  <Rle>  <IRanges>  <Rle>  |  <integer>  <numeric>
  C  chMT [18, 134]  +  |  13  0.6

$TX2
GRanges with 1 range and 2 metadata columns:
  seqnames  ranges  strand  |  score  GC
  ch2 [2, 7]  *  |  15  0

---
seqlengths:
  ch1  chMT  ch2
  50000  800  NA

> grl3
GRangesList of length 2:
[[1]]
GRanges with 1 range and 2 metadata columns:
  seqnames  ranges  strand  |  score  GC
  <Rle>  <IRanges>  <Rle>  |  <integer>  <numeric>
  chMT [22, 130]  +  |  13  0.6

[[2]]
GRanges with 1 range and 2 metadata columns:
  seqnames  ranges  strand  |  score  GC
  ch2 [2, 7]  *  |  15  0

---
seqlengths:
  ch1  chMT  ch2
  50000  800  NA
```

```r
> psetdiff(grl2, grl3)
GRangesList of length 2:
$TX1
GRanges with 2 ranges and 0 metadata columns:
  seqnames  ranges  strand
  <Rle>  <IRanges>  <Rle>
  [1]  chMT [ 18, 21]  +
  [2]  chMT [131, 134]  +

$TX2
GRanges with 0 ranges and 0 metadata columns:
  seqnames  ranges  strand

---
seqlengths:
  ch1  chMT  ch2
  50000  800  NA
```

```r
> psetdiff(grl2, grl)
GRangesList of length 2:
$TX1
GRanges with 2 ranges and 0 metadata columns:
  seqnames  ranges  strand
  <Rle>  <IRanges>  <Rle>
  [1]  chMT [ 18, 21]  +
  [2]  chMT [131, 134]  +

$TX2
GRanges with 0 ranges and 0 metadata columns:
  seqnames  ranges  strand

---
seqlengths:
  ch1  chMT  ch2
  50000  800  NA
```
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

GRanges objects
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object

Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
  - From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps

Exercise 3

Final notes
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers
- GRanges objects
  - GRanges constructor and accessors
  - Vector operations on GRanges objects
  - Range-based operations on GRanges objects
  - Splitting a GRanges object
  - Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
  - From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
The purpose of the GAlignments container is...

... to store a set of genomic alignments (aligned reads, typically).

The alignments can be loaded from a BAM file with readGAlignments(). By default, only the following information is loaded for each alignment:

- **RNAME field**: name of the reference sequence to which the query is aligned.
- **strand bit** (from FLAG field): strand in the reference sequence to which the query is aligned.
- **CIGAR field**: a string in the "Extended CIGAR format" describing the "geometry" of the alignment (i.e. locations of insertions, deletions and gaps). See the SAM Spec for the details.
- **POS field**: **1-based** position of the leftmost mapped base.

In particular, the query sequences (SEQ) and qualities (QUAL) are not loaded by default.

### Supported operations

- **Vector operations**: **partially supported** (no splitting/relisting, no comparing or ordering)
- **List operations**: **NO**
- **Ranges operations**: only narrow() and qnarrow() (GAlignments specific) are supported
- **Coercion methods**: to GRanges or GRangesList
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects
- Other frequently seen low-level containers

GRanges objects

- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects

- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects

- GAlignments
  - GAlignments constructor and accessors
  - Exercise 2
  - From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations

- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
GAlignments constructor

Typically not used directly!

```r
> gal0 <- GAlignments(seqnames=Rle(c("ch1", "ch2"), c(3, 1)),
+                     pos=1L + 10L*0:3,
+                     cigar=c("36M", "20M3D16M", "20M703N16M", "14M2I20M"),
+                     strand=strand(c("+", "-", "-", "+"))
> gal0
```

GAlignments with 4 alignments and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>strand</th>
<th>cigar</th>
<th>qwidth</th>
<th>start</th>
<th>end</th>
<th>width</th>
<th>ngap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch1</td>
<td>+</td>
<td>36M</td>
<td>36</td>
<td>1</td>
<td>36</td>
<td>36</td>
<td>0</td>
</tr>
<tr>
<td>ch1</td>
<td>-</td>
<td>20M3D16M</td>
<td>36</td>
<td>11</td>
<td>49</td>
<td>39</td>
<td>0</td>
</tr>
<tr>
<td>ch1</td>
<td>-</td>
<td>20M703N16M</td>
<td>36</td>
<td>21</td>
<td>759</td>
<td>739</td>
<td>1</td>
</tr>
<tr>
<td>ch2</td>
<td>+</td>
<td>14M2I20M</td>
<td>36</td>
<td>31</td>
<td>64</td>
<td>34</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:
- ch1
- ch2
  - ch1
  - ch2
  
An N in the cigar indicates a gap (≠ deletion).
```r
> library(pasillaBamSubset)
> U1gal <- readGAlignments(untreated1_chr4())
> length(U1gal)

[1] 204355

> head(U1gal)

GAlignments with 6 alignments and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>strand</th>
<th>cigar</th>
<th>qwidth</th>
<th>start</th>
<th>end</th>
<th>width</th>
<th>ngap</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr4</td>
<td>-</td>
<td>75M</td>
<td>75</td>
<td>892</td>
<td>966</td>
<td>75</td>
<td>0</td>
</tr>
<tr>
<td>chr4</td>
<td>-</td>
<td>75M</td>
<td>75</td>
<td>919</td>
<td>993</td>
<td>75</td>
<td>0</td>
</tr>
<tr>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>924</td>
<td>998</td>
<td>75</td>
<td>0</td>
</tr>
<tr>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>936</td>
<td>1010</td>
<td>75</td>
<td>0</td>
</tr>
<tr>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>949</td>
<td>1023</td>
<td>75</td>
<td>0</td>
</tr>
<tr>
<td>chr4</td>
<td>-</td>
<td>75M</td>
<td>75</td>
<td>967</td>
<td>1041</td>
<td>75</td>
<td>0</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>chr2L</th>
<th>chr2R</th>
<th>chr3L</th>
<th>chr3R</th>
<th>chr4</th>
<th>chrM</th>
<th>chrX</th>
<th>chrYHet</th>
</tr>
</thead>
<tbody>
<tr>
<td>23011544</td>
<td>21146708</td>
<td>24543557</td>
<td>27905053</td>
<td>1351857</td>
<td>19517</td>
<td>22422827</td>
<td>347038</td>
</tr>
</tbody>
</table>
> seqnames(U1gal)

factor-Rle of length 204355 with 1 run
  Lengths: 204355
  Values : chr4
Levels(8): chr2L chr2R chr3L chr3R chr4 chrM chrX chrYHet

> table(as.factor(seqnames(U1gal)))

chr2L  chr2R  chr3L  chr3R  chr4    chrM    chrX chrYHet
  0      0      0      0  204355     0      0      0

> strand(U1gal)

factor-Rle of length 204355 with 53763 runs
  Lengths: 2 3 3 1 2 2 4 1 4 2 2 1 ... 13 1 17 1 20 3 3 40 2
  Values : - + - + - + - + - + ... - + - + - + - + - + -
Levels(3): + - *

> table(as.factor(strand(U1gal)))

   +   -   *
102101 102254  0

> head(cigar(U1gal))

[1] "75M" "75M" "75M" "75M" "75M" "75M"

> head(qwidth(U1gal))

[1] 75 75 75 75 75 75

> table(qwidth(U1gal))

   75
204355
GAlignments accessors (continued)

```r
> head(start(U1gal))
[1]  892  919  924  936  949  967
> head(end(U1gal))
[1]  966  993  998 1010 1023 1041
> head(width(U1gal))
[1]  75  75  75  75  75  75
> head(ngap(U1gal))
[1]  0  0  0  0  0  0
> table(ngap(U1gal))
   0  1  2 184039  20169    147

> mcols(U1gal)
Dataframe with 204355 rows and 0 columns

> seqinfo(U1gal)
Seqinfo of length 8
seqnames seqlengths isCircular genome
chr2L 23011544 NA <NA>
chr2R 21146708 NA <NA>
chr3L 24543557 NA <NA>
chr3R 27905053 NA <NA>
chr4  1351857 NA <NA>
chrM  19517 NA <NA>
chrX 22422827 NA <NA>
chrYHet 347038 NA <NA>
```
Loading additional information from the BAM file

```r
> param <- ScanBamParam(what=c("flag", "mapq"), tag=c("NH", "NM"))
> U1gal <- readGAlignments(untreated1_chr4(),
+    use.names=TRUE, param=param)
> U1gal[1:5]

GAlignments with 5 alignments and 4 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>strand</th>
<th>cigar</th>
<th>qwidth</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRR031729.3941844</td>
<td>chr4</td>
<td>-</td>
<td>75M</td>
<td>75</td>
<td>892</td>
</tr>
<tr>
<td>SRR031728.3674563</td>
<td>chr4</td>
<td>-</td>
<td>75M</td>
<td>75</td>
<td>919</td>
</tr>
<tr>
<td>SRR031729.8532600</td>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>924</td>
</tr>
<tr>
<td>SRR031729.2779333</td>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>936</td>
</tr>
<tr>
<td>SRR031728.2826481</td>
<td>chr4</td>
<td>+</td>
<td>75M</td>
<td>75</td>
<td>949</td>
</tr>
</tbody>
</table>

width | ngap | flag | mapq | NH | NM
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>75</td>
<td>0</td>
<td>16</td>
<td>&lt;NA&gt;</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>16</td>
<td>&lt;NA&gt;</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

---

seqlengths:

- chr2L: 23011544
- chr2R: 21146708
- chr3L: 24543557
- chr3R: 27905053
- chr4: 1351857
- chrM: 19517
- chrX: 22422827
- chrYHet: 347038

> any(duplicated(names(U1gal)))

[1] TRUE
```
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

**GRanges objects**
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
  
Exercise 1

**GRangesList objects**
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

**GAlignments and GAlignmentPairs objects**
- GAlignments
  
**Exercise 2**
- From GAlignments to GRanges or GRangesList
  
**Advanced operations**
- Coverage and slicing
- Finding/counting overlaps
  
Exercise 3

**Final notes**
Exercise 2

a. Find the SAM Spec online and investigate the meaning of predefined tags NH and NM.

b. Load BAM file untreated1_chr4.bam into a GAlignments object and subset this object to keep only the alignments satisfying the 2 following conditions:
   ▶ The alignment corresponds to a query with a *unique alignment* (a.k.a. *unique match* or *unique hit*).
   ▶ The alignment is a *perfect match* (i.e. no insertion, no deletion, no mismatch).

c. Do those alignments have gaps?
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

GRanges objects

- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object

Exercise 1

GRangesList objects

- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects

- GAlignments
- GAlignments constructor and accessors

Exercise 2

From GAlignments to GRanges or GRangesList

GAlignmentPairs

Advanced operations

- Coverage and slicing
- Finding/counting overlaps

Exercise 3

Final notes
From GAlignments to GRanges

**Gaps are ignored**, that is, each alignment is converted into a *single* genomic range defined by the *start* and *end* of the alignment.

```r
> as(U1gal, "GRanges")

GRanges with 204355 ranges and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRR031729.3941844</td>
<td>chr4 [892, 966]</td>
<td>-</td>
</tr>
<tr>
<td>SRR031728.3674563</td>
<td>chr4 [919, 993]</td>
<td>-</td>
</tr>
<tr>
<td>SRR031729.8532600</td>
<td>chr4 [924, 998]</td>
<td>+</td>
</tr>
<tr>
<td>SRR031729.2779333</td>
<td>chr4 [936, 1010]</td>
<td>+</td>
</tr>
<tr>
<td>SRR031728.2826481</td>
<td>chr4 [949, 1023]</td>
<td>+</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SRR031728.1789947</td>
<td>chr4 [1348268, 1348342]</td>
<td>+</td>
</tr>
<tr>
<td>SRR031728.4528492</td>
<td>chr4 [1348268, 1348342]</td>
<td>+</td>
</tr>
<tr>
<td>SRR031729.5150849</td>
<td>chr4 [1348268, 1348342]</td>
<td>+</td>
</tr>
<tr>
<td>SRR031729.9070096</td>
<td>chr4 [1348449, 1348523]</td>
<td>-</td>
</tr>
<tr>
<td>SRR031729.9070096</td>
<td>chr4 [1350124, 1350198]</td>
<td>-</td>
</tr>
</tbody>
</table>

---

seqlengths:

<table>
<thead>
<tr>
<th>chr2L</th>
<th>chr2R</th>
<th>chr3L</th>
<th>chr3R</th>
<th>chr4</th>
<th>chrM</th>
<th>chrX</th>
<th>chrYHet</th>
</tr>
</thead>
<tbody>
<tr>
<td>23011544</td>
<td>21146708</td>
<td>24543557</td>
<td>27905053</td>
<td>1351857</td>
<td>19517</td>
<td>22422827</td>
<td>347038</td>
</tr>
</tbody>
</table>
From GAlignments to GRangesList

**Gaps are NOT ignored**, that is, each alignment is converted into one or more genomic ranges (one more range than the number of gaps in the alignment).

```r
> U1grl <- as(U1gal, "GRangesList")
> U1grl

GRangesList of length 204355:
$SRR031729.3941844
GRanges with 1 range and 0 metadata columns:
  seqnames    ranges strand
    <Rle> <IRanges> <Rle>
[1] chr4 [892, 966] -

$SRR031728.3674563
GRanges with 1 range and 0 metadata columns:
  seqnames    ranges strand
    <Rle> <IRanges> <Rle>
[1] chr4 [919, 993] -

$SRR031729.8532600
GRanges with 1 range and 0 metadata columns:
  seqnames    ranges strand
    <Rle> <IRanges> <Rle>
[1] chr4 [924, 998] +

...<204352 more elements>
---
seqlengths:
  chr2L  chr2R  chr3L  chr3R  chr4  chrM  chrX  chrYHet
 23011544 21146708 24543557 27905053 1351857 19517 22422827 347038
```
One more range than the number of gaps in the alignment:

```r
> all(elementLengths(U1grl) == ngap(U1gal) + 1)
[1] TRUE
```
Introduction

Most frequently seen low-level containers

- Rle objects
- IRanges objects
- DataFrame objects

Other frequently seen low-level containers

GRanges objects

- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects

- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects

- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList

GAlignmentPairs

Advanced operations

- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
The purpose of the GAlignmentPairs container is...

... to store a set of aligned paired-end reads.

- Implemented on top of the GAlignments class.
- The alignments can be loaded from a BAM file with readGAlignmentPairs().
- first(x), last(x): extract the first and last ends in 2 separate GAlignments objects of the same length.

**Supported operations**

- **Vector operations**: partially supported (no splitting/relisting, no comparing or ordering)
- **List operations**: YES
- **Ranges operations**: NO
- **Coercion methods**: to GRanges or GRangesList
```r
> library(pasillaBamSubset)
> U3galp <- readGAlignmentPairs(untreated3_chr4())
> length(U3galp)

[1] 75346

> head(U3galp)

GAlignmentPairs with 6 alignment pairs and 0 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>strand</th>
<th>ranges</th>
<th>ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>&lt;Rle&gt;</td>
<td>&lt;Rle&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;IRanges&gt;</td>
<td>&lt;IRanges&gt;</td>
</tr>
</tbody>
</table>
[1] chr4  | +      | [169, 205]   | [326, 362]   |
[2] chr4  | +      | [943, 979]   | [1086, 1122] |
[3] chr4  | +      | [944, 980]   | [1119, 1155] |
[4] chr4  | +      | [946, 982]   | [986, 1022]  |
[5] chr4  | +      | [966, 1002]  | [1108, 1144] |
[6] chr4  | +      | [966, 1002]  | [1114, 1150] |

---

seqlengths:

<table>
<thead>
<tr>
<th>chr2L</th>
<th>chr2R</th>
<th>chr3L</th>
<th>chr3R</th>
<th>chr4</th>
<th>chrM</th>
<th>chrX</th>
<th>chrYHet</th>
</tr>
</thead>
<tbody>
<tr>
<td>23011544</td>
<td>21146708</td>
<td>24543557</td>
<td>27905053</td>
<td>1351857</td>
<td>19517</td>
<td>22422827</td>
<td>347038</td>
</tr>
</tbody>
</table>
```
Currently, `readGAlignmentPairs()` drops pairs where the first and last ends have incompatible sequence names and/or strands (a rare situation).
> seqnames(U3galp)

factor-Rle of length 75346 with 1 run
   Lengths: 75346
   Values : chr4
Levels(8): chr2L chr2R chr3L chr3R chr4 chrM chrX chrYHet

> strand(U3galp)

factor-Rle of length 75346 with 18999 runs
   Lengths: 6 6 3 1 6 1 1 2 2 1 1 3 ... 3 2 3 1 2 1 5 6 2 7 3
   Values : + - + - + - + - + - + - ... + - + - + - + - + - +
Levels(3): + - *

> head(ngap(U3galp))

[1] 0 0 0 0 0 0

> table(ngap(U3galp))

        0 1 2
72949 2291 106
From GAlignmentPairs to GRangesList

```r
> U3grl <- as(U3galp, "GRangesList")
> U3grl

GRangesList of length 75346:
[[1]]
GRanges with 2 ranges and 0 metadata columns:
  seqnames   ranges strand
      <Rle>   <IRanges> <Rle>
    [1]   chr4 [169, 205]   +
    [2]   chr4 [326, 362]   +

[[2]]
GRanges with 2 ranges and 0 metadata columns:
  seqnames   ranges strand
      <Rle>   <IRanges> <Rle>
    [1]   chr4 [ 943, 979]   +
    [2]   chr4 [1086, 1122]   +

[[3]]
GRanges with 2 ranges and 0 metadata columns:
  seqnames   ranges strand
      <Rle>   <IRanges> <Rle>
    [1]   chr4 [ 944, 980]   +
    [2]   chr4 [1119, 1155]   +

...  
75343 more elements
---

seqlengths:
  chr2L  chr2R  chr3L  chr3R  chr4  chrM  chrX  chrYHet
23011544  21146708  24543557  27905053  1351857  19517  22422827  347038
```
> `U3gr1[ngap(U3galp) != 0]

**GRangesList of length 2397:**

[[1]]

**GRanges with 3 ranges and 0 metadata columns:**

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
</tr>
<tr>
<td>[1]</td>
<td>chr4 [74403, 74435]</td>
<td>-</td>
</tr>
<tr>
<td>[2]</td>
<td>chr4 [77050, 77053]</td>
<td>-</td>
</tr>
<tr>
<td>[3]</td>
<td>chr4 [13711, 13747]</td>
<td>-</td>
</tr>
</tbody>
</table>

[[2]]

**GRanges with 3 ranges and 0 metadata columns:**

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
</tr>
<tr>
<td>[1]</td>
<td>chr4 [56932, 56968]</td>
<td>+</td>
</tr>
<tr>
<td>[2]</td>
<td>chr4 [57072, 57083]</td>
<td>+</td>
</tr>
<tr>
<td>[3]</td>
<td>chr4 [57142, 57166]</td>
<td>+</td>
</tr>
</tbody>
</table>

[[3]]

**GRanges with 3 ranges and 0 metadata columns:**

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Rle&gt;</td>
<td>&lt;IRanges&gt;</td>
<td>&lt;Rle&gt;</td>
</tr>
<tr>
<td>[1]</td>
<td>chr4 [56932, 56968]</td>
<td>+</td>
</tr>
<tr>
<td>[2]</td>
<td>chr4 [57065, 57083]</td>
<td>+</td>
</tr>
<tr>
<td>[3]</td>
<td>chr4 [57142, 57159]</td>
<td>+</td>
</tr>
</tbody>
</table>

...<2394 more elements>

**seqlengths:**

| chr2L chr2R chr3L chr3R chr4 chrM chrX chrYHet |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|--------------|
| 23011544        | 21146708         | 24543557         | 27905053         | 1351857         | 19517           | 22422827        | 347038       |
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects
Other frequently seen low-level containers

GRanges objects
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
Introduction
Most frequently seen low-level containers
  Rle objects
  IRanges objects
  DataFrame objects
Other frequently seen low-level containers
GRanges objects
  GRanges constructor and accessors
  Vector operations on GRanges objects
  Range-based operations on GRanges objects
  Splitting a GRanges object
  Exercise 1
GRangesList objects
  GRangesList constructor and accessors
  Vector operations on GRangesList objects
  List operations on GRangesList objects
  Range-based operations on GRangesList objects
GAlignments and GAlignmentPairs objects
  GAlignments
  GAlignments constructor and accessors
  Exercise 2
  From GAlignments to GRanges or GRangesList
  GAlignmentPairs
Advanced operations
  Coverage and slicing
  Finding/counting overlaps
  Exercise 3
Final notes
```r
> U1cvg <- coverage(U1grl)
> U1cvg

RleList of length 8
$chr2L
  integer-Rle of length 23011544 with 1 run
  Lengths: 23011544
  Values : 0

$chr2R
  integer-Rle of length 21146708 with 1 run
  Lengths: 21146708
  Values : 0

$chr3L
  integer-Rle of length 24543557 with 1 run
  Lengths: 24543557
  Values : 0

$chr3R
  integer-Rle of length 27905053 with 1 run
  Lengths: 27905053
  Values : 0

$chr4
  integer-Rle of length 1351857 with 122061 runs
  Lengths: 891 27 5 12 13 45 5 ... 3 106 75 1600 75 1659
  Values : 0 1 2 3 4 5 4 ... 6 0 1 0 1 0

...
<3 more elements>
```
Coverage (continued)

```r
> mean(U1cvg)
  chr2L  chr2R  chr3L  chr3R  chr4  chrM  chrX  chrYHet
0.00000 0.00000 0.00000 11.33746 0.00000 0.00000 0.00000

> max(U1cvg)
  chr2L  chr2R  chr3L  chr3R  chr4  chrM  chrX  chrYHet
0      0      0      0      5627  0      0      0
```
Slicing the coverage

> U1sl <- slice(U1cvg, lower=10)
> U1sl

RleViewsList of length 8
 names(8): chr2L chr2R chr3L chr3R chr4 chrM chrX chrYHet

> elementLengths(U1sl)

<table>
<thead>
<tr>
<th></th>
<th>chr2L</th>
<th>chr2R</th>
<th>chr3L</th>
<th>chr3R</th>
<th>chr4</th>
<th>chrM</th>
<th>chrX</th>
<th>chrYHet</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1sl</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1183</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

> head(U1sl$chr4)

Views on a 1351857-length Rle subject

views:

<table>
<thead>
<tr>
<th></th>
<th>start</th>
<th>end</th>
<th>width</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>4936</td>
<td>5077</td>
<td>142</td>
</tr>
<tr>
<td>[2]</td>
<td>5211</td>
<td>5245</td>
<td>35</td>
</tr>
<tr>
<td>[3]</td>
<td>5334</td>
<td>5337</td>
<td>4</td>
</tr>
<tr>
<td>[4]</td>
<td>5736</td>
<td>5744</td>
<td>9</td>
</tr>
<tr>
<td>[5]</td>
<td>5752</td>
<td>5754</td>
<td>3</td>
</tr>
<tr>
<td>[6]</td>
<td>5756</td>
<td>5882</td>
<td>127</td>
</tr>
</tbody>
</table>

> head(mean(U1sl$chr4))


> head(max(U1sl$chr4))

[1] 39 13 10 10 10 38
Introduction

Most frequently seen low-level containers
- Rle objects
- IRanges objects
- DataFrame objects
Other frequently seen low-level containers

GRanges objects
- GRanges constructor and accessors
- Vector operations on GRanges objects
- Range-based operations on GRanges objects
- Splitting a GRanges object
- Exercise 1

GRangesList objects
- GRangesList constructor and accessors
- Vector operations on GRangesList objects
- List operations on GRangesList objects
- Range-based operations on GRangesList objects

GAlignments and GAlignmentPairs objects
- GAlignments
- GAlignments constructor and accessors
- Exercise 2
- From GAlignments to GRanges or GRangesList
- GAlignmentPairs

Advanced operations
- Coverage and slicing
- Finding/counting overlaps
- Exercise 3

Final notes
Finding/counting overlaps

A typical use case: count the number of *hits* (a.k.a. *overlaps*) per transcript.

**Typical input**

- A BAM file with the aligned reads (*single-* or *paired-end*).
- Transcript annotations *for the same reference genome* that was used to align the reads.

**Typical tools**

- `readGAlignments()` or `readGAlignmentPairs()` to load the reads in a `GAAlignments` or `GAlignmentPairs` object.
- A `TranscriptDb` object containing the transcript annotations.
- The `exonsBy()` extractor (defined in the *GenomicFeatures* package) to extract the exons ranges grouped by transcript from the `TranscriptDb` object. The exons ranges are returned in a `GRangesList` object with 1 top-level element per transcript.
- The `findOverlaps()` and/or `countOverlaps()` functions.
Load the transcripts

```r
> library(TxDB.Dmelanogaster.UCSC.dm3.ensGene)
> txdb <- TxDb.Dmelanogaster.UCSC.dm3.ensGene
> exbytx <- exonsBy(txdb, by="tx", use.names=TRUE)
> exbytx

GRangesList of length 23017:
$FBtr0300689
GRanges with 2 ranges and 3 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>exon_id</th>
<th>exon_name</th>
<th>exon_rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr2L</td>
<td>[7529, 8116]</td>
<td>+</td>
<td>1</td>
<td>&lt;NA&gt;</td>
<td>1</td>
</tr>
<tr>
<td>chr2L</td>
<td>[8193, 9484]</td>
<td>+</td>
<td>3</td>
<td>&lt;NA&gt;</td>
<td>2</td>
</tr>
</tbody>
</table>

$FBtr0300690
GRanges with 3 ranges and 3 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>exon_id</th>
<th>exon_name</th>
<th>exon_rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr2L</td>
<td>[7529, 8116]</td>
<td>+</td>
<td>1</td>
<td>&lt;NA&gt;</td>
<td>1</td>
</tr>
<tr>
<td>chr2L</td>
<td>[8193, 8589]</td>
<td>+</td>
<td>2</td>
<td>&lt;NA&gt;</td>
<td>2</td>
</tr>
<tr>
<td>chr2L</td>
<td>[8668, 9484]</td>
<td>+</td>
<td>4</td>
<td>&lt;NA&gt;</td>
<td>3</td>
</tr>
</tbody>
</table>

$FBtr0078100
GRanges with 5 ranges and 3 metadata columns:

<table>
<thead>
<tr>
<th>seqnames</th>
<th>ranges</th>
<th>strand</th>
<th>exon_id</th>
<th>exon_name</th>
<th>exon_rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr2L</td>
<td>[67044, 67507]</td>
<td>+</td>
<td>5</td>
<td>&lt;NA&gt;</td>
<td>1</td>
</tr>
<tr>
<td>chr2L</td>
<td>[67569, 67762]</td>
<td>+</td>
<td>6</td>
<td>&lt;NA&gt;</td>
<td>2</td>
</tr>
<tr>
<td>chr2L</td>
<td>[67892, 68023]</td>
<td>+</td>
<td>7</td>
<td>&lt;NA&gt;</td>
<td>3</td>
</tr>
<tr>
<td>chr2L</td>
<td>[68085, 70549]</td>
<td>+</td>
<td>8</td>
<td>&lt;NA&gt;</td>
<td>4</td>
</tr>
<tr>
<td>chr2L</td>
<td>[70607, 71390]</td>
<td>+</td>
<td>9</td>
<td>&lt;NA&gt;</td>
<td>5</td>
</tr>
</tbody>
</table>
```

---

seqlengths:

<table>
<thead>
<tr>
<th>chr2L</th>
<th>chr2R</th>
<th>chr3L</th>
<th>chr3R</th>
<th>...</th>
<th>chrXHet</th>
<th>chrYHet</th>
<th>chrUextra</th>
</tr>
</thead>
<tbody>
<tr>
<td>23011544</td>
<td>21146708</td>
<td>24543557</td>
<td>27905053</td>
<td>...</td>
<td>204112</td>
<td>347038</td>
<td>29004656</td>
</tr>
</tbody>
</table>
Single-end overlaps

```r
> U1txhits <- countOverlaps(exbytx, U1grl)
> length(U1txhits)
[1] 23017
> head(U1txhits)
FBtr0300689 FBtr0300690 FBtr0078100 FBtr0078101 FBtr0302164 FBtr0301733
   0       0       0       0       0       0
> sum(U1txhits)  # total nb of hits
[1] 250802
> head(sort(U1txhits, decreasing=TRUE))
FBtr0089175 FBtr0089176 FBtr0089177 FBtr0112904 FBtr0289951 FBtr0089243
 20380     20380     20380     6018     5982     5979
```

Rough counting!

- More than 1 alignment per read can be reported in the BAM file (sometimes the same read hits the same transcript many times).
- A hit is counted even if it's not compatible with the splicing of the transcript.
Paired-end overlaps

```r
U3txhits <- countOverlaps(exbytx, U3grl)
length(U3txhits)
[1] 23017
head(U3txhits)
FBtr0300689 FBtr0300690 FBtr0078100 FBtr0078101 FBtr0302164 FBtr0301733
 0 0 0 0 0 0
sum(U3txhits) # total nb of hits
[1] 95587
head(sort(U3txhits, decreasing=TRUE))
FBtr0089175 FBtr0089176 FBtr0089177 FBtr0112904 FBtr0289951 FBtr0089243
 6799 6799 6790 2617 2610 2609
```

Note that exons that fall within the *inter-read* gap are NOT considered to overlap.
Introduction
Most frequently seen low-level containers
   Rle objects
   IRanges objects
   DataFrame objects
Other frequently seen low-level containers
GRanges objects
   GRanges constructor and accessors
   Vector operations on GRanges objects
   Range-based operations on GRanges objects
   Splitting a GRanges object
   Exercise 1
GRangesList objects
   GRangesList constructor and accessors
   Vector operations on GRangesList objects
   List operations on GRangesList objects
   Range-based operations on GRangesList objects
GAlignments and GAlignmentPairs objects
   GAlignments
   GAlignments constructor and accessors
   Exercise 2
   From GAlignments to GRanges or GRangesList
   GAlignmentPairs
Advanced operations
   Coverage and slicing
   Finding/counting overlaps
   Exercise 3
Final notes
Exercise 3

Use the *TxDb.Dmelanogaster.UCSC.dm3.ensGene* package and the result of Exercise 2 to count the number of *unique hits* per transcript, that is, the number of hits from reads with a *unique alignment*. 
Introduction
Most frequently seen low-level containers
   Rle objects
   IRanges objects
   DataFrame objects
Other frequently seen low-level containers
GRanges objects
   GRanges constructor and accessors
   Vector operations on GRanges objects
   Range-based operations on GRanges objects
   Splitting a GRanges object
   Exercise 1
GRangesList objects
   GRangesList constructor and accessors
   Vector operations on GRangesList objects
   List operations on GRangesList objects
   Range-based operations on GRangesList objects
GAlignments and GAlignmentPairs objects
   GAlignments
   GAlignments constructor and accessors
   Exercise 2
   From GAlignments to GRanges or GRangesList
   GAlignmentPairs
Advanced operations
   Coverage and slicing
   Finding/counting overlaps
   Exercise 3
Final notes
Final notes

Where to look next

- `findCompatibleOverlaps()` and `countCompatibleOverlaps()` functions in the *GenomicRanges* package for finding/counting hits (from *single- or paired-end reads*) that are compatible with the splicing of the transcript.

- `assignReads()` and `countReads()` functions in the *SplicingGraphs* package for assigning RNA-seq reads (*single- or paired-end*) to the exons and introns of a gene model, and to summarize them in different ways. (Still a work-in-progress.)

- `summarizeOverlaps()` function in the *GenomicRanges* package for counting overlaps between reads and genomic features, and resolve reads that overlap multiple features.

- Vignettes in the *GenomicRanges* package (browseVignettes("GenomicRanges")).

- `GRanges`, `GRangesList`, `GAlignments`, and `GAlignmentPairs` man pages in the *GenomicRanges* package.

- SAMtools website: http://samtools.sourceforge.net/

- *Bioconductor* mailing lists: http://bioconductor.org/help/mailing-list/

Further developments

- More optimization on the containers. Improve their documentation.

- Paired-end reads: convenience functions for extracting the *inter-read gap* and computing the *observed template length* (a.k.a. TLEN in BAM/SAM jargon).

- On user request...

THANKS!