Rsamtools and Work Flows with Larger Data

Martin Morgan

Fred Hutchinson Cancer Research Center

14-18 June, 2010

A time line

- Yesterday: generating and aligning sequences; wrestling with large data; establishing common work flows (e.g., ChIP-seq).
- Today: revising common work flows; coming to terms with data volume (e.g., multiplexing); analyzing designed experiments.
- Tomorrow: 100,000 genomes (George Church, New York Times, 7 June 2010)

Themes

- Increasing confidence with early stages of the pipeline – reads, their qualities, and alignments *per se* become less important.
- Increasing emphasis on experimental design.
- Increasing use of collections of whole genomes, e.g., as data bases to query against.

Outline

- Yesterday
  - Review: Work Flow
  - Annotation
- Today
  - Rsamtools
  - Gapped alignments
- Tomorrow
  - Genome Collections
- Resources

Work flow

Prior to analysis

- Biological preparation, e.g., ChIP.
- ‘Sequencing’: library preparation, cluster generation, imaging, . . .

Analysis

1. Pre-processing, quality assessment, exploratory analysis
2. Domain-specific analysis (ChIP-seq, Digital gene expression, RNA-seq, Microbial / community structure, . . .)
3. Annotation & integration

ShortRead data input

```
> library(EatonEtAlChIPseq)
> fl <- system.file("extdata",
+   "GSM424494_wt_G2_orc_chip_rep1_S288C_14.mapview.txt.gz"
+   package="EatonEtAlChIPseq")
> aln <- readAligned(fl, type = "MAQMapview")
```

Alphabet by cycle

Expectation: nucleotide use independent of cycle

```
> alnp <- aln[strand(aln) == "+"]
> abc <- alphabetByCycle(sread(alnp))
> class(abc)
```

```
[1] "matrix"
```

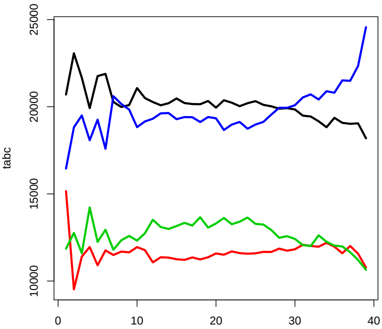
```
> abc[1:6,1:4]
```

```
        cycle
alphabet  [,1]  [,2]  [,3]  [,4]
       A 20701 23067 21668 19920
       C 15159  9523 11402 11952
       G 11856 12762 11599 14220
       T 16454 18818 19501 18078
       M     0     0     0     0
       R     0     0     0     0
```

Alphabet by cycle

`matplot` takes a matrix and plots each column as a set of points

```
> tabc <- t(abc[1:4,])
> matplot(tabc, type="l",
+          lty=1, lwd=3)
```



Annotation

- Gene- (and chip-) centric
- Pathways (KEGG, GO)
- Community resources (e.g., BioMarts, UCSC)

AnnotationDbi

- *R* packages with versioned data.
- Pre-built *org.*.db*, *GO.db*, *KEGG.db* and custom-built.

Example: starts / ends of yeast features, from SGD

```
> library(org.Sc.sgd.db)
> ls('package:org.Sc.sgd.db') # Discovery
> start <- toTable(org.Sc.sgdCHRLOC)
> end <- toTable(org.Sc.sgdCHRLOCEND)
> tbl <- merge(start, end)
```

biomaRt

- Web accessible annotations; from ENSEMBL
- Discovery: `listMarts`, `listDatasets`.
- Use: `useMart`.

```
> library(biomaRt)
> listMarts()
> mart <- useMart("ensembl")
> listDatasets(mart)
> ens <- useMart("ensembl",
+                dataset="scerevisiae_gene_ensembl")
```

Extracting data with *biomaRt*

- Apply *filters* (`listFilters`) and *attributes* (`listAttributes`)

```
> head(listFilters(ens))
> head(listAttributes(ens))
> ## example query
> getBM(attributes=
+         c("ensembl_gene_id","chromosome_name",
+           "strand", "start_position", "end_position"),
+       filters="entrezgene",
+       values=c(1466398,1466399,1466400), mart=ens)
```

rtracklayer

Import UCSC Genome Browser data into *R*

- Create a session: `browserSession`.
- List available genomes from UCSC: `ucscGenomes`.
- Set up a genome object: `genome`.
- List available tracks: `trackNames`.

```
> library(rtracklayer)
> session <- browserSession()
> head(ucscGenomes())
> genome(session) <- "hg19"
> head(trackNames(session))
```

Managing tracks with *rtracklayer*

- Generate a query for UCSC: `ucscTableQuery`.
- Retrieve a UCSC track: `getTable`.

```
> ## generate a query
> query <- ucscTableQuery(session, "refGene")
> ## get the data
> track <- getTable(query)
```

- Also possible to push tracks to UCSC

Demo

```
> system.file('script', 'seq2anno.R', package='CSAMA10')
```

Outline

Yesterday

- Review: Work Flow
- Annotation

Today

- Rsamtools
- Gapped alignments

Tomorrow

- Genome Collections

Resources

Today

Limitations to the *AlignedRead* class and *ShortRead* work flow

- Hard to input an arbitrary subset of reads
- Sequence, quality, identifier and other information included, but not always necessary
- Reads must be aligned without indels or gaps

samtools and *Rsamtools*

samtools

- Data format – text (SAM) and binary (BAM)
- Tools to manipulate (e.g., merge), analyze (e.g., pileup) and view
- Bindings for other languages, e.g., Picard

Rsamtools

- Input and represent BAM files.
- High-level: `readAligned`; with `type="BAM"`; `readPileup`
- Flexible: `scanBam`
- Experiment-wide: `BamViews`

Input

ScanBamParam

which *GRanges* selecting reference, genome coordinates, strand.

flag select paired / mapped / mate mapped reads

what fields to retrieve, e.g., query name, reference name, strand, position, width, cigar

Remote access

- E.g., 1000 genomes individual NA19240, chromosome 6, 'Solexa' reads, aligned with MAQ available via ftp

Gapped alignments

The *GappedAlignments* class in *GenomicRanges*

- `readGappedAlignments` uses `scanBam`
- Genomic coordinates, ‘cigar’, covered intervals
- Cigar: run length encoding; M (match), I, D (insertion, deletion), N (skipped), S, H (soft, hard clip), P (padding). E.g., `35M`, `18M2I15M`
- Accessors, subsetting, narrowing, `pintersect`, `coverage`, . . .

Example

```
> ## reads on chr 6 overlapping 100000-110000
> which <- GRange("6", IRanges(100000, 110000))
> param <- ScanBamParam(which=which)
> ## na19240url <- ftp://ftp-trace.ncbi.nih.gov/1000ge...
> na19240bam <- scanBam(na19240url, param=param)
```

- Index file downloaded, or locally referenced
- `scanBam` returns a nested list
  - One element for each row of `GRanges`
  - Nested elements correspond to `what`

Demo

```
> system.file('script', 'streamBam.R', package='CSAMA10')
```

Outline

- Yesterday
  - Review: Work Flow
  - Annotation
- Today
  - Rsamtools
  - Gapped alignments
- Tomorrow
  - Genome Collections
- Resources

BamViews

- Overall experiment represented by ‘regions of interest’ (rows) in several samples (columns).
- Represent this as a ‘view’ on which coordinated operations can be performed.
- Extended examples: *Rsamtools* vignette, *leeBamViews*

Demo

```
> system.file('script', 'BamViews.R', package='CSAMA10')
```

Outline

Yesterday

Review: Work Flow

Annotation

Today

Rsamtools

Gapped alignments

Tomorrow

Genome Collections

Resources

Resources

Packages

- *Rsamtools*, *GenomicRanges*
- *leeBamViews*