Creation of `parathyroidGenesSE` and `parathyroidExonsSE`

Michael Love

October 29, 2020

Abstract

This vignette describes the construction of the RangedSummarizedExperiment `parathyroidGenesSE` and `parathyroidExonsSE` in the *parathyroidSE* package.

Contents

1 Dataset description

We downloaded the RNA-Seq data from the publication of Haglund et al. [1]. The paired-end sequencing was performed on primary cultures from parathyroid tumors of 4 patients at 2 time points over 3 conditions (control, treatment with diarylpropionitrile (DPN) and treatment with 4-hydroxytamoxifen (OHT)). DPN is a selective estrogen receptor β 1 agonist and OHT is a selective estrogen receptor modulator. One sample (patient 4, 24 hours, control) was omitted by the paper authors due to low quality.

2 Downloading the data

The raw sequencing data is publicly available from the NCBI Gene Expression Omnibus under accession number GSE37211[1]. The read sequences in FASTQ format were extracted from the NCBI short read archive file (.sra files), using the sra toolkit[2].

[1] http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE37211

[2] http://www.ncbi.nlm.nih.gov/books/NBK56560/

3 Aligning reads

The sequenced reads in the FASTQ files were aligned using TopHat version 2.0.4[3] with default parameters to the GRCh37 human reference genome using the Bowtie index available at the Illumina iGenomes page[4]. The following code for the command line produces a directory for each run and then sorts resulting BAM files by QNAME, allowing us to read in the paired-end reads in batches using the `yieldSize` argument of `BamFileList`.

[3] http://tophat.cbcb.umd.edu/

[4] http://tophat.cbcb.umd.edu/igenomes.html

```
tophat2 -o file_tophat_out genome file_1.fastq file_2.fastq
samtools sort -n file_tophat_out/accepted_hits.bam _sorted
```

4 Counting reads in genes

The genes were downloaded using the `makeTranscriptDbFromBiomart` of the *GenomicFeatures* package, drawing from Ensembl release 72 on July 30 2013. For stability and reproducibility of results, one might consider to download the GTF files for the appropriate Ensembl release directly from the Ensembl website. The GTF file can be read in using the `makeTranscriptDbFromGFF` function with the argument `format` set to `"gtf"`. The `exonsBy` function produces a *GRangesList* object of all exons grouped by gene.

```
library("GenomicFeatures")
hse <- makeTranscriptDbFromBiomart(biomart="ensembl",
                                   dataset="hsapiens_gene_ensembl")
exonsByGene <- exonsBy(hse, by="gene")
```

For demonstration purposes in the vignette, we load a subset of these genes:

```
library("parathyroidSE")
data(exonsByGene)
```

The following code is used to generate a character vector of the location of the BAM files. The first line specifying `bamDir` would typically be replaced with the directory containing the BAM files.

```
bamDir <- system.file("extdata",package="parathyroidSE",mustWork=TRUE)
fls <- list.files(bamDir, pattern="bam$",full=TRUE)
```

We specified the files using `BamFileList` of the *Rsamtools* package. The BAM files are sorted by QNAME, so there is not an index file, and we set `obeyQname`.

```
library("Rsamtools")
bamLst <- BamFileList(fls, index=character(), obeyQname=TRUE)
```

For counting reads in genes, we used `summarizeOverlaps` from the *GenomicAlignments* package. The following code demonstrates counting reads from 3 reduced BAM files over a subset of the Ensembl genes. We set the counting mode to `"Union"`, which is explained in the di-

agram for htseq-count[5]. The protocol is not strand specific, so we set `ignore.strand=TRUE`. We specified `fragments=TRUE`, in order to count both proper pairs and "singletons" (reads without a mate).

[5] http://www-huber.embl.de/users/anders/HTSeq/doc/count.html

```
library("GenomicAlignments")
parathyroidGenesSE <- summarizeOverlaps(exonsByGene, bamLst,
                                        mode="Union",
                                        singleEnd=FALSE,
                                        ignore.strand=TRUE,
                                        fragments=TRUE)
```

5 Preparing exonic parts

For counting reads at the exon-level, we first prepared a *GRanges* object which contains non-overlapping exonic parts. We used the function `disjointExons` from the *GenomicFeatures* package in order to prepare the non-overlapping exonic parts. By comparing count levels across these exonic parts, we could infer cases of differential exon usage. The resulting exonic parts are identical to those produced by the python script distributed with the *DEXSeq* package (though the aggregated gene names might be in a different order). Note that some of the exonic parts have changed since the preparation of the *parathyroid* package due to the different Ensembl releases.

```
exonicParts <- disjointExons(hse)
```

For the vignette, we import a subset of these exonic parts:

```
data(exonicParts)
```

The resulting exonic parts look like:

```
exonicParts[1:3]
```

```
GRanges object with 3 ranges and 3 metadata columns:
      seqnames            ranges strand |         gene_id         tx_name
         <Rle>         <IRanges>  <Rle> | <CharacterList> <CharacterList>
  [1]        X 99883667-99884983      - | ENSG00000000003 ENST00000373020
  [2]        X 99885756-99885863      - | ENSG00000000003 ENST00000373020
  [3]        X 99887482-99887537      - | ENSG00000000003 ENST00000373020
      exonic_part
        <integer>
  [1]           1
  [2]           2
  [3]           3
  -------
  seqinfo: 580 sequences (1 circular) from an unspecified genome
```

6 Counting reads in exonic parts

We used the `summarizeOverlaps` function again, this time specifying `inter.feature=FALSE` in order to count all overlaps, treating each feature independently. Otherwise, paired-end reads and junction-spanning reads which hit more than one exonic part would not be counted.

```
parathyroidExonsSE <- summarizeOverlaps(exonicParts, bamLst,
                                        mode="Union",
                                        singleEnd=FALSE,
                                        ignore.strand=TRUE,
                                        inter.feature=FALSE,
                                        fragments=TRUE)
```

Note that the metadata about the transcripts is stored in the `rowRanges` of these *RangedSummarizedExperiment* objects. Here, `str` is used to neatly print a list.

```
str(metadata(rowRanges(parathyroidGenesSE)))
```

```
List of 1
 $ genomeInfo:List of 20
  ..$ Db type                                  : chr "TranscriptDb"
  ..$ Supporting package                       : chr "GenomicFeatures"
  ..$ Data source                              : chr "BioMart"
  ..$ Organism                                 : chr "Homo sapiens"
  ..$ Resource URL                             : chr "www.biomart.org:80"
  ..$ BioMart database                         : chr "ensembl"
  ..$ BioMart database version                 : chr "ENSEMBL GENES 72 (SANGER UK)"
  ..$ BioMart dataset                          : chr "hsapiens_gene_ensembl"
  ..$ BioMart dataset description              : chr "Homo sapiens genes (GRCh37.p11)"
  ..$ BioMart dataset version                  : chr "GRCh37.p11"
  ..$ Full dataset                             : chr "yes"
  ..$ miRBase build ID                         : chr NA
  ..$ transcript_nrow                          : chr "213140"
  ..$ exon_nrow                                : chr "737783"
  ..$ cds_nrow                                 : chr "531154"
  ..$ Db created by                            : chr "GenomicFeatures package from Bioconductor"
  ..$ Creation time                            : chr "2013-07-30 17:30:25 +0200 (Tue, 30 Jul 2013)"
  ..$ GenomicFeatures version at creation time: chr "1.13.21"
  ..$ RSQLite version at creation time         : chr "0.11.4"
  ..$ DBSCHEMAVERSION                          : chr "1.0"
```

7 Obtaining sample annotations from GEO

In order to provide phenotypic data for the samples, we used the *GEOquery* package to parse the series matrix file downloaded from the NCBI Gene Expression Omnibus under accession number GSE37211. We included this file as well in the package, and read it in locally in the code below.

```
library("GEOquery")
gse37211 <- getGEO(filename=system.file("extdata/GSE37211_series_matrix.txt",
                                package="parathyroidSE",mustWork=TRUE))
samples <- pData(gse37211)[,c("patient:ch1","agent:ch1",
                              "time:ch1","relation")]
colnames(samples) <- c("patient","treatment","time","experiment")
samples$patient <- sub("patient: (.+)","\\1",samples$patient)
samples$treatment <- sub("agent: (.+)","\\1",samples$treatment)
samples$time <- sub("time: (.+)","\\1",samples$time)
samples$experiment <- sub("SRA: http://www.ncbi.nlm.nih.gov/sra\\?term=(.+)","\\1",
```

```
                        samples$experiment)
samples
```

```
           patient treatment time experiment
GSM913873        1   Control  24h  SRX140503
GSM913874        1   Control  48h  SRX140504
GSM913875        1       DPN  24h  SRX140505
GSM913876        1       DPN  48h  SRX140506
GSM913877        1       OHT  24h  SRX140507
GSM913878        1       OHT  48h  SRX140508
GSM913879        2   Control  24h  SRX140509
GSM913880        2   Control  48h  SRX140510
GSM913881        2       DPN  24h  SRX140511
GSM913882        2       DPN  48h  SRX140512
GSM913883        2       OHT  24h  SRX140513
GSM913884        2       OHT  48h  SRX140514
GSM913885        3   Control  24h  SRX140515
GSM913886        3   Control  48h  SRX140516
GSM913887        3       DPN  24h  SRX140517
GSM913888        3       DPN  48h  SRX140518
GSM913889        3       OHT  24h  SRX140519
GSM913890        3       OHT  48h  SRX140520
GSM913891        4   Control  48h  SRX140521
GSM913892        4       DPN  24h  SRX140522
GSM913893        4       DPN  48h  SRX140523
GSM913894        4       OHT  24h  SRX140524
GSM913895        4       OHT  48h  SRX140525
```

8 Matching GEO experiments with SRA runs

The sample information from GEO must be matched to the individual runs from the Short Read Archive (the FASTQ files), as some samples are spread over multiple sequencing runs. The run information can be obtained from the Short Read Archive using the *SRAdb* package (note that the first step involves a large download of the SRA metadata database). We included the conversion table in the package.

```
library("SRAdb")
sqlfile <- getSRAdbFile()
sra_con <- dbConnect(SQLite(),sqlfile)
conversion <- sraConvert(in_acc = samples$experiment, out_type =
                        c("sra","submission","study","sample","experiment","run"),
                        sra_con = sra_con)
write.table(conversion,file="inst/extdata/conversion.txt")
```

We used the `merge` function to match the sample annotations to the run information. We ordered the *data.frame* `samplesFull` by the run number and then set all columns as factors.

```
conversion <- read.table(system.file("extdata/conversion.txt",
                                     package="parathyroidSE",mustWork=TRUE))
samplesFull <- merge(samples, conversion)
samplesFull <- samplesFull[order(samplesFull$run),]
samplesFull <- DataFrame(lapply(samplesFull, factor))
```

9 Adding column data and experiment data

We combined the information from GEO and SRA to the *RangedSummarizedExperiment* object. First we extracted the run ID, contained in the names of the *BamFileList* in the `fileName` column. We then ordered the rows of `samplesFull` to match the order of the run ID in `parathyroidGenesSE`, and removed the duplicate column of run ID.

```
colData(parathyroidGenesSE)$run <- sub(".*(SRR.*)_tophat_out.*","\\1",
                                       colnames(parathyroidGenesSE))
matchOrder <- match(colData(parathyroidGenesSE)$run, samplesFull$run)
colData(parathyroidGenesSE) <- cbind(colData(parathyroidGenesSE),
                                     subset(samplesFull[matchOrder,],select=-run))
colData(parathyroidExonsSE)$run <- sub(".*(SRR.*)_tophat_out.*","\\1",
                                       colnames(parathyroidExonsSE))
matchOrder <- match(colData(parathyroidExonsSE)$run, samplesFull$run)
colData(parathyroidExonsSE) <- cbind(colData(parathyroidExonsSE),
                                     subset(samplesFull[matchOrder,],select=-run))
```

We included experiment data and PubMed ID from the NCBI Gene Expression Omnibus.

```
metadata = new("MIAME",
  name="Felix Haglund",
  lab="Science for Life Laboratory Stockholm",
  contact="Mikael Huss",
  title="DPN and Tamoxifen treatments of parathyroid adenoma cells",
  url="http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE37211",
  abstract="Primary hyperparathyroidism (PHPT) is most frequently present in postmenopausal women. Althou
pubMedIds(metadata) <- "23024189"
metadata(parathyroidGenesSE) <- list(MIAME=metadata)
metadata(parathyroidExonsSE) <- list(MIAME=metadata)
```

Finally, we saved the object in the data directory of the package.

```
save(parathyroidGenesSE,file="data/parathyroidGenesSE.RData")
save(parathyroidExonsSE,file="data/parathyroidExonsSE.RData")
```

10 Session information

- R version 4.0.3 (2020-10-10), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Ubuntu 18.04.5 LTS`
- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.12-bioc/R/lib/libRblas.so`
- LAPACK: `/home/biocbuild/bbs-3.12-bioc/R/lib/libRlapack.so`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils

- Other packages: Biobase 2.50.0, BiocGenerics 0.36.0, Biostrings 2.58.0, GEOquery 2.58.0, GenomeInfoDb 1.26.0, GenomicAlignments 1.26.0, GenomicRanges 1.42.0, IRanges 2.24.0, MatrixGenerics 1.2.0, Rsamtools 2.6.0, S4Vectors 0.28.0, SummarizedExperiment 1.20.0, XVector 0.30.0, matrixStats 0.57.0, parathyroidSE 1.28.0
- Loaded via a namespace (and not attached): BiocManager 1.30.10, BiocParallel 1.24.0, BiocStyle 2.18.0, DelayedArray 0.16.0, GenomeInfoDbData 1.2.4, Matrix 1.2-18, R6 2.5.0, RCurl 1.98-1.2, assertthat 0.2.1, bitops 1.0-6, cli 2.1.0, compiler 4.0.3, crayon 1.3.4, digest 0.6.27, dplyr 1.0.2, ellipsis 0.3.1, evaluate 0.14, fansi 0.4.1, generics 0.0.2, glue 1.4.2, grid 4.0.3, hms 0.5.3, htmltools 0.5.0, knitr 1.30, lattice 0.20-41, lifecycle 0.2.0, limma 3.46.0, magrittr 1.5, pillar 1.4.6, pkgconfig 2.0.3, ps 1.4.0, purrr 0.3.4, readr 1.4.0, rlang 0.4.8, rmarkdown 2.5, rstudioapi 0.11, tibble 3.0.4, tidyr 1.1.2, tidyselect 1.1.0, tools 4.0.3, vctrs 0.3.4, xfun 0.18, xml2 1.3.2, yaml 2.2.1, zlibbioc 1.36.0

References

[1] Felix Haglund, Ran Ma, Mikael Huss, Luqman Sulaiman, Ming Lu, Inga-Lena Nilsson, Anders Höög, Christofer C. Juhlin, Johan Hartman, and Catharina Larsson. Evidence of a Functional Estrogen Receptor in Parathyroid Adenomas. *Journal of Clinical Endocrinology & Metabolism*, September 2012. URL: http://dx.doi.org/10.1210/jc.2012-2484, doi:10.1210/jc.2012-2484.