Creation of `parathyroidGenesSE`

Michael Love

February 6, 2013

Abstract

This vignette describes the construction of the SummarizedExperiment `parathyroidGenesSE` 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].

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 indexes the BAM file (substituting the SRR number for `file`):

```
tophat2 -o file_tophat_out -p 8 genome file_1.fastq file_2.fastq
samtools index file_tophat_out/accepted_hits.bam
```

4 Counting reads in genes

The genes were downloaded using the *GenomicFeatures* package from Ensembl release 69 on 5 February 2013. 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 the vignette, we load a subset of these genes:

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

For counting reads in genes, we used `summarizeOverlaps` from the *GenomicRanges* and *Rsamtools* packages. The following code demonstrates counting reads from 3 reduced BAM files over a subset of the Ensembl genes. The protocol is not strand specific, so we set `ignore.strand=TRUE`. We counted "singletons" as well, reads with an unmapped mate, and added these counts to produce a total.

[1] `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE37211`
[2] `http://www.ncbi.nlm.nih.gov/books/NBK56560/`
[3] `http://tophat.cbcb.umd.edu/`
[4] `http://tophat.cbcb.umd.edu/igenomes.html`

```
library(Rsamtools)
bamDir <- system.file("extdata",package="parathyroidSE",mustWork=TRUE)
fls <- list.files(bamDir, pattern="bam$",full=TRUE)
bamlst <- BamFileList(fls)
geneHitsPairs <- summarizeOverlaps(exonsByGene, bamlst, mode="Union",
                                   singleEnd=FALSE, ignore.strand=TRUE)
geneHitsSingletons <- summarizeOverlaps(exonsByGene, bamlst, mode="Union",
                                        param=ScanBamParam(flag=scanBamFlag(
                                        isPaired=TRUE, hasUnmappedMate=TRUE)),
                                        singleEnd=TRUE, ignore.strand=TRUE)
parathyroidGenesSE <- geneHitsPairs
assay(parathyroidGenesSE) <- assay(geneHitsPairs) + assay(geneHitsSingletons)
```

5 Preparing exonic parts

For counting reads at the exon-level, we first prepared a *GRanges* object which contains 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. We first retreived the exon-by-transcript information to annotate exonic parts with transcript membership.

```
exonsByTranscript <- exonsBy(hse, by="tx", use.names=TRUE)
```

For the vignette, we import a subset of these transcripts:

```
data(exonsByTranscript)
```

Disjoining the exons into non-overlapping exonic parts:

```
exonicParts <- disjoin(unlist(exonsByGene))
```

Assigning exonic parts to aggregate genes:

```
foGG <- findOverlaps(exonsByGene, exonsByGene)
splitByGene <- split(subjectHits(foGG), queryHits(foGG))
aggregateGeneNames <- sapply(splitByGene, function(i)
                             paste(names(exonsByGene)[i],collapse="+"))
foEG <- findOverlaps(exonicParts, exonsByGene, select="first")
mcols(exonicParts)$aggregate_gene <- aggregateGeneNames[foEG]
```

Assigning exonic parts to transcripts:

```
foET <- findOverlaps(exonicParts, exonsByTranscript)
splitByExonicPart <- split(subjectHits(foET), queryHits(foET))
mcols(exonicParts)$transcripts <- sapply(splitByExonicPart, function(i)
                                   paste(names(exonsByTranscript)[i],collapse="+"))
```

Sorting the exonic parts, and assigning numbers to each exonic part per aggregate gene:

```
exonicParts <- exonicParts[order(mcols(exonicParts)$aggregate_gene)]
mcols(exonicParts)$exonic_part_number <- do.call(c,lapply(split(mcols(exonicParts)$aggregate_gene,
                                        mcols(exonicParts)$aggregate_gene),
                                        function(z) seq(along=z)))
```

The resulting exonic parts look like:

```
exonicParts[101:103]
```

```
GRanges with 3 ranges and 3 metadata columns:
      seqnames           ranges strand |                  aggregate_gene
         <Rle>        <IRanges>  <Rle> |                     <character>
  [1]        1 [238418, 238558]      - | ENSG00000228463+ENSG00000241670
  [2]        1 [238559, 238567]      - | ENSG00000228463+ENSG00000241670
  [3]        1 [257268, 257672]      - | ENSG00000228463+ENSG00000241670
                         transcripts exonic_part_number
                         <character>          <integer>
  [1] ENST00000448958+ENST00000424587                 10
  [2]                 ENST00000424587                 11
  [3]                 ENST00000335577                 12
  ---
  seqlengths:
                  1                2 ...            LRG_98            LRG_99
          249250621        243199373 ...             18750             13294
```

6 Counting reads in exonic parts

We used the `countOverlaps` function as a counting mode, in order to count all overlaps. Otherwise, paired-end reads and junction-spanning reads which hit more than one exonic part would not be counted.

```
myco <- function(reads, features, ignore.strand) countOverlaps(
                  features, reads, ignore.strand=ignore.strand)
exonHitsPairs <- summarizeOverlaps(exonicParts, bamlst, mode=myco,
                                   singleEnd=FALSE, ignore.strand=TRUE)
exonHitsSingletons <- summarizeOverlaps(exonicParts, bamlst, mode=myco,
                                        param=ScanBamParam(flag=scanBamFlag(
                                        isPaired=TRUE, hasUnmappedMate=TRUE)),
                                        singleEnd=TRUE, ignore.strand=TRUE)
parathyroidExonsSE <- exonHitsPairs
assay(parathyroidExonsSE) <- assay(exonHitsPairs) + assay(exonHitsSingletons)
```

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("characteristics_ch1","characteristics_ch1.2",
                               "characteristics_ch1.3","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 *SummarizedExperiment* 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)$run <- sub(".*(SRR.*)_tophat_out.*","\\1",
                                       names(colData(parathyroidGenesSE)$fileName))
matchOrder <- match(colData(parathyroidGenesSE)$run, samplesFull$run)
colData(parathyroidGenesSE) <- cbind(colData(parathyroidGenesSE),
                                     subset(samplesFull[matchOrder,],select=-run))
colData(parathyroidExonsSE)$run <- sub(".*(SRR.*)_tophat_out.*","\\1",
                                       names(colData(parathyroidExonsSE)$fileName))
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.

```
exptData = 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(exptData) <- "23024189"
exptData(parathyroidGenesSE) <- list(MIAME=exptData)
exptData(parathyroidExonsSE) <- list(MIAME=exptData)
```

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 Under development (unstable) (2012-10-31 r61057), `x86_64-unknown-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=C`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.19.2, BiocGenerics 0.5.6, Biostrings 2.27.10, GEOquery 2.25.1, GenomicRanges 1.11.28, IRanges 1.17.31, Rsamtools 1.11.15, parathyroidSE 0.99.1
- Loaded via a namespace (and not attached): RCurl 1.95-3, XML 3.95-0.1, bitops 1.0-5, stats4 2.16.0, tools 2.16.0, zlibbioc 1.5.0

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.