Package 'tximeta'

October 17, 2020

Version 1.6.3

Title Transcript Quantification Import with Automatic Metadata

Description Transcript quantification import from Salmon and alevin with automatic attachment of transcript ranges and release information, and other associated metadata. De novo transcriptomes can be linked to the appropriate sources with linkedTxomes and shared for computational reproducibility.

Maintainer Michael Love <michaelisaiahlove@gmail.com>

License GPL-2

VignetteBuilder knitr

Imports SummarizedExperiment, tximport, jsonlite, S4Vectors, IRanges, GenomicRanges, AnnotationDbi, GenomicFeatures, ensembldb, BiocFileCache, AnnotationHub, Biostrings, tibble, GenomeInfoDb, rappdirs, utils, methods, Matrix

Suggests knitr, rmarkdown, testthat, tximportData, org.Dm.eg.db, DESeq2, edgeR, limma, devtools

URL https://github.com/mikelove/tximeta

biocViews Annotation, GenomeAnnotation, DataImport, Preprocessing, RNASeq, SingleCell, Transcriptomics, Transcription, GeneExpression, FunctionalGenomics, ReproducibleResearch, ReportWriting, ImmunoOncology

RoxygenNote 7.1.0

Encoding UTF-8

git_url https://git.bioconductor.org/packages/tximeta

git_branch RELEASE_3_11

git_last_commit 65c2753

git_last_commit_date 2020-08-20

Date/Publication 2020-10-16

Author Michael Love [aut, cre],
Charlotte Soneson [aut, ctb],
Peter Hickey [aut, ctb],
Rob Patro [aut, ctb]

R topics documented:

`tximeta-package` *Import transcript-level quantification data with automatic metadata*

Description

The tximeta package imports abundances (TPM), estimated counts, and effective lengths from Salmon, alevin, or other quantification tools, and will output a SummarizedExperiment object. For Salmon and alevin quantification directories, `tximeta` will try to identify the correct provenance of the reference transcripts and automatically attach the transcript ranges to the SummarizedExperiment, to facilitate downstream integration with other datasets. The automatic identification of reference transcripts should work out-of-the-box for human or mouse transcriptomes from the sources: GENCODE, Ensembl, or RefSeq.

Details

The main functions are:

- `tximeta` - with key argument: `coldata`
- `summarizeToGene` - summarize quants to gene-level
- `retrieveDb` - retrieve the transcript database
- `addIds` - add transcript or gene ID (see `gene` argument)
- `addExons` - convert from GRanges to GRangesList

All software-related questions should be posted to the Bioconductor Support Site:

`https://support.bioconductor.org`

The code can be viewed at the GitHub repository, which also lists the contributtor code of conduct:

`https://github.com/mikelove/tximeta`

Author(s)

Michael I. Love, Charlotte Soneson, Peter Hickey, Rob Patro

References

tximeta reference:

Michael I. Love, Charlotte Soneson, Peter F. Hickey, Lisa K. Johnson N. Tessa Pierce, Lori Shepherd, Martin Morgan, Rob Patro (2020) Tximeta: reference sequence checksums for provenance identification in RNA-seq. PLOS Computational Biology. https://doi.org/10.1371/journal.pcbi.1007664

tximport reference (the effective length offset and counts-from-abundance):

Charlotte Soneson, Michael I. Love, Mark D. Robinson (2015) Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. F1000Research. http://doi.org/10.12688/f1000research.7563

`addExons` *Add exons to rowRanges of a transcript-level SummarizedExperiment*

Description

After running `tximeta`, the `SummarizedExperiment` output will have `GRanges` representing the transcript locations attached as `rowRanges` to the object. These provide the start and end of the transcript in the genomic coordiantes, and strand information. However, the exonic locations are not provided. This function, `addExons`, swaps out the `GRanges` with a `GRangesList`, essentially a list along the rows of the `SummarizedExperiment`, where each element of the list is a `GRanges` providing the locations of the exons for that transcript.

Usage

```
addExons(se)
```

Arguments

`se`	the SummarizedExperiment

Details

This function is designed only for transcript-level objects. This "lack of a feature" reflects a belief on the part of the package author that it makes more sense to think about exons belonging to transcripts than to genes. For users desiring exonic information alongside gene-level objects, for example, which exons are associated with a particular gene, it is recommended to pull out the relevant `GRangesList` for the transcripts of this gene, while the object represents transcript-level data, such that the exons are still associated with transcripts.

For an example of `addExons`, please see the tximeta vignette.

Value

a SummarizedExperiment

addIds *Add IDs to rowRanges of a SummarizedExperiment*

Description

For now this just works with SummarizedExperiments with Ensembl gene or transcript IDs. See example of usage in tximeta vignette. For obtaining multiple matching IDs for each row of the SummarizedExperiment set `multiVals="list"`. See `select` for documentation on use of `multiVals`.

Usage

```
addIds(se, column, gene = FALSE, ...)
```

Arguments

se	the SummarizedExperiment
column	the name of the new ID to add (a `column` of the org database)
gene	logical, whether to map by genes or transcripts (default is FALSE). if rows are genes, and easily detected as such (ENSG or ENSMUSG), it will automatically switch to TRUE. if rows are transcripts and `gene=TRUE`, then it will try to use a `gene_id` column to map IDs to `column`
...	arguments passed to `mapIds`

Value

a SummarizedExperiment

Examples

```
example(tximeta)
library(org.Dm.eg.db)
se <- addIds(se, "REFSEQ", gene=FALSE)
```

getTximetaBFC *Get or set the directory of the BiocFileCache used by tximeta*

Description

Running `getTximetaBFC` will report the saved directory, if it has been determined, or will return NULL. Running `setTximetaBFC` will ask the user to specify a BiocFileCache directory for accessing and saving TxDb sqlite files.

Usage

```
getTximetaBFC()

setTximetaBFC(dir)
```

Arguments

dir	the location for tximeta's BiocFileCache. can be missing in which case the function will call `file.choose` for choosing location interactively

Value

the directory of the BiocFileCache used by tximeta (or nothing, in the case of `setTximetaBFC`)

Examples

```
# getting the BiocFileCache used by tximeta
# (may not be set, which uses BiocFileCache default or temp directory)
getTximetaBFC()

# don't want to actually change user settings so this is not run:
# setTximetaBFC()
```

---

linkedTxome *Make and load linked transcriptomes ("linkedTxome")*

Description

For now, for details please see the vignette `inst/script/linked.Rmd`

Usage

```
makeLinkedTxome(
  indexDir,
  source,
  organism,
  release,
  genome,
  fasta,
  gtf,
  write = TRUE,
  jsonFile
)

loadLinkedTxome(jsonFile)
```

Arguments

indexDir	the local path to the Salmon index
source	the source of transcriptome (e.g. "GENCODE", "Ensembl", "de-novo")
organism	organism (e.g. "Homo sapiens")
release	release number (e.g. "27")
genome	genome (e.g. "GRCh38", or "none")

fasta	location(s) for the FASTA transcript sequences (of which the transcripts used to build the index is equal or a subset). This can be a local path, or an HTTP or FTP URL
gtf	location for the GTF/GFF file (of which the transcripts used to build the index is equal or a subset). This can be a local path, or an HTTP or FTP URL While the `fasta` argument can take a vector of length greater than one (more than one FASTA file containing transcripts used in indexing), the `gtf` argument has to be a single GTF/GFF file. If transcripts were added to a standard set of reference transcripts (e.g. fusion genes, or pathogen transcripts), it is recommended that the tximeta user would manually add these to the GTF/GFF file, and post the modified GTF/GFF publicly, such as on Zenodo. This enables consistent annotation and downstream annotation tasks, such as by `summarizeToGene`.
write	logical, should a JSON file be written out which documents the transcriptome checksum and metadata? (default is TRUE)
jsonFile	the path to the json file for the linkedTxome

Value

nothing, the function is run for its side effects

Examples

```
# point to a Salmon quantification file with an additional artificial transcript
dir <- system.file("extdata/salmon_dm", package="tximportData")
file <- file.path(dir, "SRR1197474.plus", "quant.sf")
coldata <- data.frame(files=file, names="SRR1197474", sample="1",
                      stringsAsFactors=FALSE)

# now point to the Salmon index itself to create a linkedTxome
# as the index will not match a known txome
indexDir <- file.path(dir, "Dm.BDGP6.22.98.plus_salmon-0.14.1")

# point to the source FASTA and GTF:
fastaFTP <- c("ftp://ftp.ensembl.org/pub/release-98/fasta/drosophila_melanogaster/cdna/Drosophila_melanogas
         "ftp://ftp.ensembl.org/pub/release-98/fasta/drosophila_melanogaster/ncrna/Drosophila_melanogaster
              "extra_transcript.fa.gz")
gtfPath <- file.path(dir, "Drosophila_melanogaster.BDGP6.22.98.plus.gtf.gz")

# now create a linkedTxome, linking the Salmon index to its FASTA and GTF sources
makeLinkedTxome(indexDir=indexDir, source="Ensembl", organism="Drosophila melanogaster",
               release="98", genome="BDGP6.22", fasta=fastaFTP, gtf=gtfPath, write=FALSE)

# to clear the entire linkedTxome table
# (don't run unless you want to clear this table!)
# bfcloc <- getTximetaBFC()
# bfc <- BiocFileCache(bfcloc)
# bfcremove(bfc, bfcquery(bfc, "linkedTxomeTbl")$rid)
```

makeDGEList *Make a DGEList from tximeta output*

Description

A simple wrapper function for constructing a DGEList for use with edgeR. See vignette for an example. Requires installation of the edgeR package from Bioconductor.

Usage

```
makeDGEList(se)
```

Arguments

se	a SummarizedExperiment produced by tximeta

Value

a DGEList

retrieveDb *Retrieve the TxDb or EnsDb associated with a SummarizedExperiment*

Description

SummarizedExperiment objects returned by `tximeta` have associated TxDb or EnsDb databases which are cached locally and used to perform various metadata related tasks. This helper function retrieves the database itself for the user to perform any additional operations.

Usage

```
retrieveDb(se)
```

Arguments

se	the SummarizedExperiment

Value

a database object

Examples

```
example(tximeta)
edb <- retrieveDb(se)
```

splitSE *Split SummarizedExperiment by gene categories*

Description

Construct a new SummarizedExperiment by splitting one of the assays into a list of assays, each of which contains features of a given 'type'. It is assumed that there is a one-to-one correspondence between feature sets of different types; for example, these can be spliced and unspliced variants of the same transcripts. The type of each feature in the original SummarizedExperiment, and the correspondence between the features of different types, are given in a `data.frame`.

Usage

```
splitSE(se, splitDf, assayName)
```

Arguments

`se`	A `SummarizedExperiment` object.
`splitDf`	A `data.frame` with feature IDs. Each column represents a separate feature type, and the features in a given row are considered representatives of the same feature (and will be represented as one feature in the output object).
`assayName`	A character scalar, indicating the assay of `se` that will be split. Must be one of `assayNames(se)`.

Value

A `SummarizedExperiment` object with the same columns as the input object, and the same number of assays as the number of columns in `splitDf`. The assays will be named by the column names of `splitDf`. The `colData` and `metadata` of the input `SummarizedExperiment` object are copied to the output object. The row names are set to the feature IDs in the first column of `splitDf`.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = S4Vectors::SimpleList(
    counts = as(matrix(1:15, nrow = 5), "sparseMatrix"),
    logcounts = log2(matrix(1:15, nrow = 5))
  ),
  colData = S4Vectors::DataFrame(sID = paste0("S", 1:3),
                                 condition = c("A", "A", "B")),
  metadata = list(md1 = "annotation")
)
rownames(se) <- paste0("G", 1:5)
colnames(se) <- paste0("P", 1:3)
splitDf <- data.frame(spliced = c("G1", "G2", "G6"),
                      unspliced = c("G3", "G5", "G4"),
                      stringsAsFactors = FALSE)

splse <- splitSE(se = se, splitDf = splitDf, assayName = "counts")
```

summarizeToGene,SummarizedExperiment-method *Summarize estimated quantitites to gene-level*

Description

Summarizes abundances, counts, lengths, (and inferential replicates or variance) from transcript- to gene-level. This function operates on SummarizedExperiment objects, and will automatically access the relevant TxDb (by either finding it in the BiocFileCache or by building it from an ftp location). #' This function uses the tximport package to perform summarization, where a method is defined that works on simple lists.

Usage

```
## S4 method for signature 'SummarizedExperiment'
summarizeToGene(object, varReduce = FALSE, ...)
```

Arguments

`object`	a SummarizedExperiment produced by `tximeta`
`varReduce`	whether to reduce per-sample inferential replicates information into a matrix of sample variances `variance` (default FALSE)
`...`	arguments passed to `tximport`

Value

a SummarizedExperiment with summarized quantifications

Examples

```
example(tximeta)
gse <- summarizeToGene(se)
```

tximeta *tximeta: Transcript quantification import with automatic metadata*

Description

`tximeta` leverages the hashed checksum of the Salmon index, in addition to a number of core Bioconductor packages (GenomicFeatures, ensembldb, AnnotationHub, GenomeInfoDb, BiocFileCache) to automatically populate metadata for the user, without additional effort from the user. Note that `tximeta` requires that the entire output directory of Salmon or alevin is present and unmodified in order to identify the provenance of the reference transcripts.

Usage

```
tximeta(
  coldata,
  type = "salmon",
  txOut = TRUE,
  skipMeta = FALSE,
  skipSeqinfo = FALSE,
  useHub = TRUE,
  markDuplicateTxps = FALSE,
  cleanDuplicateTxps = FALSE,
  customMetaInfo = NULL,
  ...
)
```

Arguments

coldata	a data.frame with at least two columns (others will propogate to object): • files - character, paths of quantification files • names - character, sample names if coldata is a vector, it is assumed to be the paths of quantification files and unique sample names are created
type	what quantifier was used (see tximport)
txOut	whether to output transcript-level data. tximeta is designed to have transcript-level output with Salmon, so default is TRUE, and it's recommended to use summarizeToGene following tximeta for gene-level summarization. For an alevin file, tximeta will import the gene level counts ignoring this argument (alevin produces only gene-level quantification).
skipMeta	whether to skip metadata generation (e.g. to avoid errors if not connected to internet). This calls tximport directly and so either txOut=TRUE or tx2gene should be specified.
skipSeqinfo	whether to skip the addition of Seqinfo, which requires an internet connection to download the relevant chromosome information table from UCSC
useHub	whether to first attempt to download an EnsDb object from AnnotationHub, rather than creating from a GTF file from FTP (default is TRUE). If FALSE, it will force tximeta to download and parse the GTF
markDuplicateTxps	whether to mark the status (hasDuplicate) and names of duplicate transcripts (duplicates) in the rowData of the SummarizedExperiment output. Subsequent summarization to gene level will keep track of the number of transcripts sets per gene (numDupSets)
cleanDuplicateTxps	whether to try to clean duplicate transcripts (exact sequence duplicates) by replacing the transcript names that do not appear in the GTF with those that do appear in the GTF
customMetaInfo	the relative path to a custom metadata information JSON file, relative to the paths in files of coldata. For example, customMetaInfo="meta_info.json" would indicate that in the same directory as the quantification files in files, there are custom metadata information JSON files. These should contain the SHA-256 hash of the reference transcripts with the index_seq_hash tag (see details in vignette).

... arguments passed to tximport

Details

Most of the code in tximeta works to add metadata and transcript ranges when the quantification was performed with Salmon. However, tximeta can be used with any quantification type that is supported by tximport, where it will return an non-ranged SummarizedExperiment.

tximeta performs a lookup of the hashed checksum of the index (stored in an auxilary information directory of the Salmon output) against a database of known transcriptomes, which lives within the tximeta package and is continually updated on Bioconductor's release schedule. In addition, tximeta performs a lookup of the checksum against a locally stored table of linkedTxome's (see link{makeLinkedTxome}). If tximeta detects a match, it will automatically populate, e.g. the transcript locations, the transcriptome release, the genome with correct chromosome lengths, etc. It allows for automatic and correct summarization of transcript-level quantifications to the gene-level via summarizeToGene without the need to manually build a tx2gene table.

tximeta on the first run will ask where the BiocFileCache for this package should be kept, either using a default location or a temporary directory. At any point, the user can specify a location using setTximetaBFC and this choice will be saved for future sessions. Multiple users can point to the same BiocFileCache, such that transcript databases (TxDb or EnsDb) associated with certain Salmon indices and linkedTxomes can be accessed by different users without additional effort or time spent downloading and building the relevant TxDb / EnsDb. Note that, if the EnsDb is present in AnnotationHub, tximeta will use this object instead of downloading and building an EnsDb from GTF (to disable this set useHub=FALSE).

In order to allow that multiple users can read and write to the same location, one should set the BiocFileCache directory to have group write permissions (g+w).

Value

a SummarizedExperiment with metadata on the rowRanges. (if the hashed checksum in the Salmon or Sailfish index does not match any known transcriptomes, or any locally saved linkedTxome, tximeta will just return a non-ranged SummarizedExperiment)

Examples

```
# point to a Salmon quantification file:
dir <- system.file("extdata/salmon_dm", package="tximportData")
files <- file.path(dir, "SRR1197474", "quant.sf")
coldata <- data.frame(files, names="SRR1197474", condition="A", stringsAsFactors=FALSE)

# normally we would just run the following which would download the appropriate metadata
# se <- tximeta(coldata)

# for this example, we instead point to a local path where the GTF can be found
# by making a linkedTxome:
indexDir <- file.path(dir, "Dm.BDGP6.22.98_salmon-0.14.1")
fastaFTP <- c("ftp://ftp.ensembl.org/pub/release-98/fasta/drosophila_melanogaster/cdna/Drosophila_melanogas
         "ftp://ftp.ensembl.org/pub/release-98/fasta/drosophila_melanogaster/ncrna/Drosophila_melanogaster
gtfPath <- file.path(dir, "Drosophila_melanogaster.BDGP6.22.98.gtf.gz")
makeLinkedTxome(indexDir=indexDir, source="Ensembl", organism="Drosophila melanogaster",
              release="98", genome="BDGP6.22", fasta=fastaFTP, gtf=gtfPath, write=FALSE)
se <- tximeta(coldata)

# to clear the entire linkedTxome table
```

```
# (don't run unless you want to clear this table!)
# bfcloc <- getTximetaBFC()
# bfc <- BiocFileCache(bfcloc)
# bfcremove(bfc, bfcquery(bfc, "linkedTxomeTbl")$rid)
```

Index