Package 'homosapienDEE2CellScore'

July 3, 2025

Title Example Data Package for CellScore

Version 1.5.0

Description This is a data package for normalised homosapien data downloaded from DEE2. The package both downloads, normalises, and filters the data, and provides a way to access the data from a canonical store without needing local processing. This package was built as a way to generate and store canonical test data for CellScore.

Imports Rtsne (>= 0.15), utils(>= 3.5.0), ExperimentHub, BiocGenerics, DESeq2, S4Vectors, SummarizedExperiment, getDEE2, MatrixGenerics

Suggests knitr, rmarkdown, devtools, Biobase (>= 2.39.1), BiocManager, AnnotationHubData, ExperimentHubData, AnnotationHub, CellScore (>= 1.21.4)

License GPL (>= 3)

biocViews RNASeqData, Genome, ExperimentHub, ExpressionData

BugReports https://github.com/flaviusb/homosapienDEE2CellScore/issues

Encoding UTF-8

RoxygenNote 7.3.1

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/homosapienDEE2CellScore

git_branch devel

git_last_commit f5dd740

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-07-03

Author Justin Marsh [aut, cre]

Maintainer Justin Marsh <j.marsh@auckland.ac.nz>

Contents

`buildData` *buildData builds the data included in this package*

Description

This function generates the data set for this package. All parameters are optional; by default the function will generate a normalised dataset based on downloading the accessions in 'inst/hsapiens_colData_transitions_v3 for species "hsapiens", and save the dataset to a file called 'homosapienDEE2Data.rds' in the current directory.

Usage

```
buildData(
  species = "hsapiens",
  name_prefix = "homosapienDEE2Data",
  name_suffix = ".csv",
  build_raw = FALSE,
  build_srx_agg = FALSE,
  build_deseq2 = TRUE,
  build_tsne = TRUE,
  build_rank = TRUE,
  generate_qc_pass = TRUE,
  generate_qc_warn = TRUE,
  base = getwd(),
  quiet = TRUE,
  metadata = if (!(build_raw || build_srx_agg || build_deseq2 || build_tsne ||
    build_rank) || !(generate_qc_pass || generate_qc_warn)) {
     return(list())
 } else
    {
     getDEE2Metadata(species, quiet = quiet)
 },
  counts.cutoff = 10,
  accessions = as.list(unique(cols$SRR_accession)),
 in_data = if (!(build_raw || build_srx_agg || build_deseq2 || build_tsne || build_rank)
    || !(generate_qc_pass || generate_qc_warn)) {
     return(list())
 } else {

   buildRaw(species = species, accessions = accessions, quiet = quiet, metadata =
    metadata)
 },
  dds_design = ~1,
  write_files = TRUE
)
```

Arguments

species	The species to fetch data for; default is "hsapiens".
name_prefix	The output file name prefix; default is "homosapienDEE2Data".
name_suffix	The output file name suffix; default is ".csv"
build_raw	Whether to build the raw normalisation.
build_srx_agg	Whether to build the srx aggregation normalisation.
build_deseq2	Whether to build the deseq2 normalisation.
build_tsne	Whether to build the tsne normalisation.
build_rank	Whether to build the rank normalisation.
generate_qc_pass	Generate output from the input data that passed quality control
generate_qc_warn	Generate output from the the conbination of input data that passed quality control and input data that had warnings in quality control
base	The directory to output the file to; default is the current working directory.
quiet	Whether to suppress notification output where possible; default TRUE.
metadata	If you have already downloaded metadata for the species, you can pass it in here. Otherwise the metadata will be downloaded.
counts.cutoff	Cutoff value for minimum gene expression; default is 10.
accessions	Which sample ids to download from DEE2 (we refer to these as accessions); default is derived from 'hsapiens_colData.csv' in this package. For subsets, you can see the internal 'cols' objects 'SRR_accession' member.
in_data	If you have already downloaded the accession data from DEE2, you can pass it through here. Otherwise this data will be downloaded.
dds_design	The design formula used as part of DESeq2 normalisation. Default is '~ 1'. See the documentation for 'DESeq2::DESeqDataSetFromMatrix' for more details.
write_files	Write out normalised data to files. If this is false, the function will not write out the normalised data, but will only return it.

Value

A named list of SummarizedExperiment objects. The exact set depends on the options you select when calling the function.

See Also

downloadAllTheData

Examples

```
# To build the default, full dataset, and write it out to several csv files:
#homosapienDEE2CellScore::buildData()

# To build a restricted set of data, with a cached metadata file,
# only running deseq2 normalisation, to "data_PASS_deseq2.csv" and "data_WARN_deseq2.csv"
metadata <- getDEE2::getDEE2Metadata("hsapiens", quiet=TRUE)
homosapienDEE2CellScore::buildData(
  metadata=metadata, accessions=as.list(unique(cols$SRR_accession)[c(1,3)]),
```

```
  build_deseq2=TRUE, build_tsne=FALSE, build_rank=FALSE, name_prefix="data")

# Process a subset of the data, but do not write it out into files
processed_data <- homosapienDEE2CellScore::buildData(
  metadata=metadata, accessions=as.list(unique(cols$SRR_accession)[c(1,3)]),
  build_deseq2=TRUE, build_tsne=FALSE, write_files=FALSE)

# Get PCA form of the deseq2 normalised data that passed quality control
pca_form <- prcomp(t(SummarizedExperiment::assay(processed_data$qc_pass_deseq2, "counts")))
```

---

buildRaw *buildRaw gets the raw data in SummarizedExperiment format*

Description

This function gets the raw Data from dee2 and packages it in a SummarizedExperiment

Usage

```
buildRaw(
  species = "hsapiens",
  accessions = unique(cols$SRR_accession),
  quiet = TRUE,
  metadata = getDEE2Metadata(species, quiet = quiet)
)
```

Arguments

species	The species to fetch data for; default is "hsapiens".
accessions	Which sample ids to download from DEE2 (we refer to these as accessions); default is derived from ‘hsapiens_colData.csv‘ in this package. For subsets, you can see the internal ‘cols‘ objects ‘SRR_accession‘ member.
quiet	Whether to suppress notification output where possible; default TRUE.
metadata	If you have already downloaded metadata for the species, you can pass it in here. Otherwise the metadata will be downloaded.

Value

Returns a SummarizedExperiment object containing raw data downloaded from dee2.

Examples

```
# To get a few accessions and package them into a SummarizedExperiment
accessions_of_interest <- buildRaw(accessions=as.list(unique(cols$SRR_accession)[c(1,3)]))
```

cols *hsapiens column data we are targetting*

Description

A dataframe containing metadata for the dataset we are processing; notably the accessions (which tell us the specific chunks of data to download from dee2.io) are in cols$SRR_accession

Usage

```
cols
```

Format

An object of class `DFrame` with 572 rows and 75 columns.

Examples

```
# We can use this for looking up metadata based on other metadata
# For instance, to get all the accessions for liver cells in the dataset we are analysing, run
liver_cell_accessions <-
 homosapienDEE2CellScore::cols$SRR_accession[homosapienDEE2CellScore::cols$cell_type == "liver"]
```

downloadAllTheData *downloadAllTheData in SummarizedExperiment format*

Description

This is a helper function to download all of the processed data from figshare and unpack it into a tagged list of SummarizedExperiment objects.

Usage

```
downloadAllTheData()
```

Value

A named list of SummarizedExperiment objects. The names correspond to the kinds of filtering and processing that object has undergone. The names and what they correspond to are:

- HomosapienDEE2_QC_WARN_Raw Raw data including data that has quality control warnings
- HomosapienDEE2_QC_PASS_Raw Raw data without any quality control warnings
- HomosapienDEE2_QC_WARN_Rank Rank normalised data including data that has quality control warnings
- HomosapienDEE2_QC_PASS_Rank Rank normalised data without any quality control warnings
- HomosapienDEE2_QC_WARN_Agg Aggregated data including data that has quality control warnings

- HomosapienDEE2_QC_PASS_Agg Aggregated data without any quality control warnings
- HomosapienDEE2_QC_WARN_Deseq2 Deseq2 normalised data that has quality control warnings
- HomosapienDEE2_QC_PASS_Deseq2 Deseq2 normalised data without any quality control warnings

Examples

```
# To download all of the preprocessed data from figshare via ExperimentHub, run:

#the_data <- homosapienDEE2CellScore::downloadAllTheData()
```

`HomosapienDEE2_QC_WARN_Raw` *Automatically created ergonomic accessor functions*

Description

Accessor functions for retrieving the data associated with this data package from ExperimentHub. Each of these functions downloads the container file and then returns a path to it. This file can be rehydrated into a SummarizedExperiment by using 'readInSEZip'. Usually you would want to actually use 'downloadAllTheData' instead of using any of these functions.

Details

- HomosapienDEE2_QC_WARN_Raw Raw data including data that has quality control warnings
- HomosapienDEE2_QC_PASS_Raw Raw data without any quality control warnings
- HomosapienDEE2_QC_WARN_Rank Rank normalised data including data that has quality control warnings
- HomosapienDEE2_QC_PASS_Rank Rank normalised data without any quality control warnings
- HomosapienDEE2_QC_WARN_Agg Aggregated data including data that has quality control warnings
- HomosapienDEE2_QC_PASS_Agg Aggregated data without any quality control warnings
- HomosapienDEE2_QC_WARN_Deseq2 Deseq2 normalised data that has quality control warnings
- HomosapienDEE2_QC_PASS_Deseq2 Deseq2 normalised data without any quality control warnings

Value

These functions return a path to the downloaded container file.

See Also

readInSEZip

downloadAllTheData

Examples

```
# The ExperimentHub metadata for the Deseq2 normalised data that passes QC is downloadable like so
the_metadata <- HomosapienDEE2_QC_PASS_Deseq2(metadata=TRUE)

# Or to download all of the data for the Deseq2 normalised data that passes QC do the following
#the_data <- HomosapienDEE2_QC_PASS_Deseq2()
```

`readInSEFolder` *readInSEFolder read a SummarizedExperiment in from a folder*

Description

This function reads in a SummaraizedExperiment from a Zip file generated by writeOutSEZip. It is designed for getting an intact SummarizedExperiment out of ExperimentHub for a data package, so it does not clean up after itself and leaves stray csv files in the data package directory.

Usage

```
readInSEFolder(folder_name = "SE_out/")
```

Arguments

`folder_name`	The path to a folder containing a SummarizedExperiment

Value

A SummarizedExperiment object.

Examples

```
# We can read in a small SummarizedExperiment containing a
# subset of the built data stored directly in the package like so
small_data <- readInSEFolder(
 folder_name=system.file("ExampleSummarisedExperimentFolder", package="homosapienDEE2CellScore"))
```

`readInSEZip` *readInSEZip read a SummarizedExperiment in from a zip file*

Description

This function reads in a SummarizedExperiment from a Zip file generated by writeOutSEZip. It is designed for getting an intact SummarizedExperiment out of ExperimentHub for a data package, so it extracts the intermediate csvs into a temporary folder to get the data into the datastructure.

Usage

```
readInSEZip(zip_name = "SE_out.zip")
```

Arguments

`zip_name`	The path to a zip file containing a SummarizedExperiment

Value

A SummarizedExperiment object.

Examples

```
# We can read in a small SummarizedExperiment containing a
# subset of the built data stored directly in the package like so
small_data <- readInSEZip(
  system.file("ASmallSummarizedExperiment.zip", package="homosapienDEE2CellScore"))
```

`srx_agg_se` *srx_agg_se is a version of srx_agg that works on SummarizedExperiments*

Description

This function aggregates runs that represent the same SRA experiment, and reorganises the coldata in the SummarizedExperiment to to be grouped by SRA experiment in order to preserve necessary SummarizedExperiment internal invariants.

Usage

```
srx_agg_se(x, counts = "GeneCounts")
```

Arguments

`x`	A SummarizedExperiment.
`counts`	What kind of count; "GeneCounts" for STAR based gene counts, "TxCounts" for kallisto transcript level counts or "Tx2Gene" for transcript counts aggregated to gene level. Default is "GeneCounts"

Value

A SummarizedExperiment object, with runs representing the same SRA experiment aggregated, and with coldata grouped by SRA experiment.

Examples

```
# This is a small SummarizedExperiment containing some un-aggregated data
small_data <- readInSEZip(
  system.file("ASmallSummarizedExperiment.zip", package="homosapienDEE2CellScore"))
# We can aggregate it like so:
aggregated_small_data <- srx_agg_se(small_data)
```

writeOutSEZip *writeOutSEZip writes out a SummarizedExperiment into a zip file*

Description

This function writes out a SummarizedExperiment into a group of zipped csvs, with a manifest.csv It is designed for use in persisting the SummarizedExperiments generated by this data package, for upload to ExperimentHub, so it is not built robustly.

Usage

```
writeOutSEZip(
  the_summarized_experiment,
  filename_base = "SE_out",
  filename_ext = ".csv",
 filenames = list(metadata = paste(filename_base, "_metadata", filename_ext, sep = ""),
   assay_counts = paste(filename_base, "_assay_counts", filename_ext, sep = ""),
  assay_calls = paste(filename_base, "_assay_calls", filename_ext, sep = ""), colData =
   paste(filename_base, "_colData", filename_ext, sep = ""), rowData =
   paste(filename_base, "_rowData", filename_ext, sep = "")),
  zip_name = paste(filename_base, ".zip", sep = "")
)
```

Arguments

the_summarized_experiment	A SummarizedExperiment
filename_base	The default base name to use for each of the generated files within the zip file
filename_ext	The default extension to use for each of the generated files within the zip file
filenames	A tagged list of filenames for the files inside the zip file: ‘metadata‘, ‘assay_counts‘, ‘assay_calls‘, ‘colData‘, ‘rowData‘.
zip_name	The name of the generated zip file.

Value

The status value returned by the external command invoked to create the zip file, invisibly.

Examples

```
# First, we download a few accessions of interest into a SummarizedExperiment
accessions_of_interest <- buildData(
  accessions=as.list(unique(cols$SRR_accession)[c(1,3)]), write_files=FALSE,
  build_raw=TRUE, build_deseq2=FALSE, build_tsne=FALSE, build_rank=FALSE,
  generate_qc_pass=FALSE)$qc_warn_raw
# Then we write them out to a zip file for later
writeOutSEZip(accessions_of_interest, filename_base="InterestingAccessionsForLater")
```

Index