Package 'RNASeqAnalysis'

January 15, 2017

Title Analysis of RNA-Seq data: QC filtering, expression analysis, differential expression pipeline

Version 1.0.0

Author Nicky Thrupp [aut, cre]

Maintainer Nicky Thrupp <nixthrupp@gmail.com>

Description The package implements a pipeline for the analysis of RNA-Seq data. The pipeline is designed to simplify the analysis as much as possible, thus requiring minimal user input.

Depends R (>= 3.2.1),
lattice,
readr,
ShortRead,
tximport,
tximportData

Imports grDevices, graphics, stats

License GPL-3

LazyData true

RoxygenNote 5.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

addKallistoResToTable *Adds the output kallisto directory to the original datafile.*

Description

GOES HERE

Usage

```
addKallistoResToTable(dataFile, countDat)
```

Arguments

`dataFile`	MORE
`countDat`	MORE

Value

dataframe MORE

See Also

`collectKallistoCounts`

availableReferences *A number of transcriptome indices have already been built, for use with* `kallistoQuant`

Description

A number of transcriptome index files (.idx files) have been pre-built and provided with the package. These files may be used as input to the `kallsitoQuant` pseudoalignment function (or the user may provide his/her own transcriptome file). The `availableReferences` function provides a list of these ready-available transcriptomes.

Usage

```
availableReferences()
```

Arguments

`none`	No parameters.

Value

list A list of available transcriptome indices (.idx files).

Examples

```
availableReferences()
```

`checkDataFile` *Checks that the format of the "data file" (which is used as input in many functions) is correct.*

Description

The function takes as input a data file (see `datafileTemplate`), and checks whether it is in a format acceptable to be used as input to other functions.

Usage

```
checkDataFile(x)
```

Arguments

`x.`	A data frame containing: the name of the file to be processed, whether it is single or paried-end data, the sample and replicate ID, and (optional) the name of an output file which results from the `runQAandFilter` function will be written to.

Value

A data frame which has been modified - the column names and classes have been verified (and corrected if necessary), all white space has been removed, observations relating to files which do not exist in the directory have been removed. The function also prints out any duplicated files - these should be removed by the user to prevent complications downstream.

See Also

`datafileTemplate`

Examples

```
dirPath <- system.file("extdata", "RNA-Seq-data", package = "RNASeqAnalysis")
setwd(dirPath)
dataF <- read.csv("data3.csv")
checkDataFile(dataF)
```

`collectKallistoCounts` *Collects raw counts of gene (or transcript) data generated by kallisto.*

Description

Extracts raw counts from each of the "abundance.tsv" files generated by kallisto (`https://pachterlab.github.io/kallisto/about`) expression analysis. The counts are put into a matrix (genes by sample), and rounded. This step collects the raw count data ready for further expression analysis, or for differential expression, using `DESeq2`.

Usage

```
collectKallistoCounts(baseDir = "./kallistoResults")
```

Arguments

`baseDir`	A character string of the directory where kallisto results are stored. For example, baseDir = "kallistoResults" will search "kallistoResults/*/abundance.tsv".

Value

matrix. A matrix of (rounded) raw counts of genes (or transcripts) for each sample.

See Also

kallisto (`https://pachterlab.github.io/kallisto/about`)

`collectKallistoCountsTXItoGene` *Collects raw counts of genes, based on transcript data generated by kallisto.*

Description

Extracts raw counts of transcripts from each of the "abundance.tsv" files generated by kallisto (`https://pachterlab.github.io/kallisto/about`) expression analysis and converts this to gene counts. The counts are put into a matrix (genes by sample). This step collects the raw count data ready for further expression analysis, or for differential expression, using `DESeq2`.

Usage

```
collectKallistoCountsTXItoGene(baseDir = "./kallistoResults")
```

Arguments

`baseDir`	A character string of the directory where kallisto results are stored. For example, baseDir = "kallistoResults" will search "kallistoResults/*/abundance.tsv".

Value

matrix. A matrix of raw counts of genes for each sample.

`compareQA` *Compares quality before and after filtering fastq files. Porvides a graphic of the change in quality of fastq files, before and after running the `filterBadSeqs` function.*

Description

Compares quality before and after filtering fastq files. Porvides a graphic of the change in quality of fastq files, before and after running the `filterBadSeqs` function.

Usage

```
compareQA(dataFile, prefiltData, postfiltData)
```

Arguments

`dataFile`	An R data frame with the data to be processed. The R object is a standard format, and must contain the following headings: File, PE, Sample, Replicate, FilteredFile. More information about the file is available at `datafileTemplate`. The `checkDataFile` function should be run on the file before use.
`prefiltData`	A FastqQA object resulting from `runQA`. The object contains QA data on fastq files before the filtering procedure (`filterBadSeqs`).
`postfiltData`	A FastqQA object resulting from `runQA`. The object contains QA data on fastq files after the filtering procedure (`filterBadSeqs`).

Value

a datafrome containing densities and quality scores for each fastq file (both before and after filtering). It also outputs a graph which shows comparisons of quality before and after filtering.

`datafileTemplate` *(Empty) template data file which summarises the data to be analysed*

Description

This is an empty dataset, the purpose of which is to act as a template for users. In the data file, the name of the file, the name of the sample and the technical replicate, and whether the sample is paired-end or single-end should be included. In addition, if the user is planning to filter the input file (see `runQAandFilter`), a column is provided for the user to add the name of the output fastq file.

This data file acts as input to many of the package's functions. The format of the data file should not be altered, save to add observations (rows).

The data set `datafileExample` is an example of a data file containing observations.

Usage

```
datafileTemplate
```

Format

a dataframe with 0 observations of 5 vairables

Details

- File. The name of the fastq (fastq.gz) file to be processed
- PE. "PE" should be entered into this column if the sample is from a paired-end run, or "SE" if the sample is from a single-end run.
- Sample. The name of the sample (e.g. "control", "lung").
- Replicate. The name of the replicate (e.g. "a", "2").
- FilteredFile. The name of the fastq (fastq.gz) file that the results from `runQAandFilter` function are written to.

Author(s)

Nicky Thrupp <`nixthrupp@gmail.com`>

filterBadSeqs *Performs quality checks, then filters reads for quality*

Description

The function trims poor-quality bases and unknown bases from the ends of the sequences. Any reads which are too short, or contain any unknown bases (N), are removed from the file.

Usage

```
filterBadSeqs(dataFile, minlength = 30, Phred = 25, blockSize = 1e+08,
  readerBlockSize = 1e+05, mc.cores = 1)
```

Arguments

`dataFile`	An R data frame with the data to be processed. The R object is a standard format, and must contain the following headings: File, PE, Sample, Replicate, FilteredFile. More information about the file is available at `datafileTemplate`.
`Phred`	An integer which specifies Phred (ascii) quality score. Any two consecutive nucleotides with a quality score lower than this threshold will be discarded. Default score is 30.
`blockSize`	An integer which specifies the number of reads to be read at a time when processing. Default is 1e8.
`mc.cores`	The number of cores to use when parallelizing. Default is 1 (i.e. no parallelisation)
`minLength`	An integer which specifies the minimum length for a read. Reads shorter than this length will be discarded. Default is 30 nucleotides.
`readBlockSize`	An integer which specifies the number of bytes (characters) to be read at one time. Smaller `readBlockSize` reduces memory requirements, but is less efficient. Default is 1e5.

Details

The function should be run in the working directory, where all fastq files are found.

`filterBadSeqs` iterates over each file specified in the "datafile", and filters and trims the reads for quality. This is done by iterating over chunks of reads in the fastq files at a time. The size of the chunks are decided by the "blockSize" and "readerBlockSize" parameters. More information about how this is done is available in the `ShortRead` package.

* it removes any trailing or leadining N's from each sequence,

* it removes any reads wich still contain N's,

* it trims the trailing end when it finds a minimum of 2 poor-quality bases in a window of 5. The threshold for poor quality is determined by the parameter "Phred", where the Phred score is logarithmically related to the probability of errors at each base,

* it removes any reads shorter than a minimum length (this is specified by the "minLength" parameter).

The function produces a new set of fastq files which have been filtered. The user must specify in the "FILTEREDFILE" column of the data file the output file. The user may specify the same output file for multiple input files - this will append new output to existing files, thereby allowing de-multiplexing of samples which have been run on different lanes. A new R object (`QualityFilterResults`) is created, which contains pointers to the input and output fastq files, as well as a summary of how many reads have been trimmed or removed.

Value

A data frame summarising for each file how many sequences have been trimmed or removed.

See Also

`https://en.wikipedia.org/wiki/Phred_quality_score` for more about quality scores.

`ShortRead` for more information about `blockSize` (n) and `readerBlockSize`.

`kallistoIndex` *Builds a transcriptome index (.idx) from a FASTA-formatted transcriptome.*

Description

`kallistoIndex` builds an index of the transcriptome. Once built, the transcriptome index (in the form of a De Bruijn graph) may be used for pseudoalignment of the target RNA-Seq data in order to quantify expression (see `kallistoQuant`). A transcriptome De Bruijn graph (T-DBG) is constructed from k-mers in the transcriptome, where each k-mer in the transcriptome is a node in the T-DBG. Each transcript is represented by a different colour, and whenever a node appears in a transcript, it is given the colour of that transcript. MORE. Kallisto must be installed locally. Kallisto can be downloaded from `https://pachterlab.github.io/kallisto/download.html`.

Usage

```
kallistoIndex(refTranscriptome, indexName = "./transcripts.idx")
```

Arguments

`refTranscriptome`	A character string of the transcriptome file (in fasta format) to be indexed.
`indexName`	A character string of the output index file to be created. Default is "./transcripts.idx").

Value

An index file (.idx) which can be used to quantify expression data using `kallistoQuant` pseudoalignment.

See Also

`http://pachterlab.github.io/kallisto/` for more about Kallisto.

Examples

```
kallistoIndex("./transcripts.fasta")
kallistoIndex("./transcripts.fa.gz", "./index.idx")
```

`kallistoQuant` *Kallisto quantifies transcript abundance from RNA-Seq data.*

Description

The data is broken down into k-mers and each k-mer is pseudoaligned to k-mers in the index. Because kallisto doesn't rely on full alignment, it is much quicker than other methods, without losing accuracy.

Usage

```
kallistoQuant(dataFile, preFilt = FALSE, refIndex,
  refIndexFromPackage = FALSE, bootstrap = 0, fragmentLength = 200,
  fragmentSD = 10, biasCor = FALSE)
```

Arguments

`refIndex`	A character string of the name of the index file (usually, .idx) against which the fastq files are pseudoaligned. This file can be generated from a fasta-formatted transcriptome file using `kallistoIndex`.
`refIndexFromPackage`	A logical, false by default. If the user would like to use an index provided by the package (these can be viewed using `availableReferences()`, he/she should specifiy "true" here. Otherwise, the function will assume the index is provided by the user.
`bootstrap`	An integer specifying the number of times to bootstrap. The output will provide a measure of variance.
`fragmentLength`	An integer. Estimated fragment length. Only required for single-end data (for paired-end data, Kallisto is able to calculate the fragment length). Default is 200 bp.
`fragmentSD`	A numeric. The standard deviation of the fragment length. Only required for single-end data (for paired-end data, Kallisto is able to calculate the standard deviation). Default is 10 bp.
`file1`	A character string of the name of the RNA-Seq data file (fastq.gz) to be processed.
`file2`	A character string of the RNA-Seq data file (fastq.gz) to be processed - in the case there is paired-end data.
`pairedEnd`	is a logical. If `true` (default), a paired end protocol is chosen (for this, the file1 and file2 parameters must be specified). If `false`, a single end protocol will be run, and only file1 will be processed.

Value

A data frame of the estimated abundances of each transcript specified in the input file(s). The data frame is also saved to a folder, which is given the title of the files, exlcuding the extensions. The folder contains these abundances (in text format and compressed format), as well as information about the run.

runQA — *Performs quality checks*

Description

The function checks each fastq file specified in the "data file" for quality, and writes findings to a report.

Usage

```
runQA(dataFile, preFilter = TRUE)
```

Arguments

`dataFile`	An R data frame with the data to be processed. The R object is a standard format, and must contain the following headings: File, PE, Sample, Replicate, FilteredFile. More information about the file is available at `datafileTemplate`.
`preFilter`	A logical - if true (default), the function will select and analyse files which have not yet been processed for quality. If false, the function will select and analyse those files which have been processed for quality, i.e. the "filtered file" in the data file.

Details

The function should be run in the working directory, where all fastq files are found.

`runQA` iterates over each file specified in the "datafile". It runs a quality assessment from the `ShortRead` package. The `ShortRead` package (`https://bioconductor.org/packages/release/bioc/html/ShortRead.html`) contains more information about this step. The quality assessment may be performed before and after the filtering step, by setting the "pre-filter" parameter to true or to false, respectively. All quality assessment data is output to the "QA" directory. Quality reports are output to the working directory (under the QA directory). R objects of the raw data used to generate the reports are also saved to this directory.

Value

FastqQA object. Outputs quality results in the form of raw data (an R FastqQA object) and HTML format (saved to "QA" directory).

summariseFilteringStats — *Collects summary statistics on filtering*

Description

DESCRIPTION GOES HERE

Usage

```
summariseFilteringStats(x)
```

Arguments

`x`	An object of class QualityFilterResults (e.g. returned from `filterBadSeqs`).

Value

A dataframe. The dataframe contains the number of reads which have been trimmed (on leading or trailing tails) for poor quality or unknown bases, the number of reads which have been removed entirely due to poor quality, possession of unknown bases, or length too short. Also produces graphs of absolute number of reads output, and relative numbers of reads which have been trimmed or removed (in PDF format).

See Also

`filterBadSeqs`

Index