Package ‘Rbowtie2’

April 6, 2026

Type Package

Title An R Wrapper for Bowtie2 and AdapterRemoval

Version 2.16.0

Date 2021-10-5

Description This package provides an R wrapper of the popular bowtie2 sequencing reads aligner and AdapterRemoval, a convenient tool for rapid adapter trimming, identification, and read merging. The package contains wrapper functions that allow for genome indexing and alignment to those indexes. The package also allows for the creation of .bam files via Rsamtools.

License GPL (>= 3)

Depends R (>= 4.1.0)

Encoding UTF-8

Suggests knitr, testthat (>= 3.0.0), rmarkdown

Imports magrittr, Rsamtools

SystemRequirements C++11, GNU make, samtools

Archs x64

RoxygenNote 7.1.1

biocViews Sequencing, Alignment, Preprocessing

VignetteBuilder knitr

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/Rbowtie2

git_branch RELEASE_3_22

git_last_commit ab8fb3e

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-04-05

Author Zheng Wei [aut, cre],
Wei Zhang [aut]

Maintainer Zheng Wei <`wzweizheng@qq.com`>

Contents

.callbinary 2
adapterremoval_usage 3
adapterremoval_version 3
bowtie2 4
bowtie2-build 5
bowtie2_build_usage 7
bowtie2_samtools 7
bowtie2_usage 9
bowtie2_version 10
checkFileCreateable 10
checkFileExist 11
checkIndexType 11
checkPathExist 12
checkSamtoolsExists 12
identify_adapters 13
remove_adapters 14

Index **16**

`.callbinary` *Make system call for binaries*

Description

Function that makes a system call for the bowtie binaries. Note it is not intended to be used outside of the package.

Usage

```
.callbinary(
  bin1,
  args1,
  op = NULL,
  bin2 = NULL,
  args2 = NULL,
  path = NULL,
  lang = NULL
)
```

Arguments

`bin1`	`Character`. The binary used for the system call.
`args1`	`Character`. The arguments to pass to the binary.
`op`	`Character`. Optional: Generally used if needed to pipe to another binary.
`bin2`	`Character`. Optional: Another binary that can be used in the system call. Generally the output of the first binary is piped to the this binary.
`args2`	`Character` Optional: The arguments to pass to the second binary.
`path`	`Character` Optional: If passed to function, returns the path. Needed for Rsamtools to convert from sam to bam.
`lang`	`Character`. The interpreter used to execute the binary

Value

The output of the system call or the path provided.

Author(s)

Zheng Wei

`adapterremoval_usage` *Print available arguments for adapterremoval*

Description

Print available arguments for adapterremoval. Note that some arguments to the adapterremoval are invalid if they are already handled as explicit function arguments.

Usage

```
adapterremoval_usage()
```

Value

AdapterRemoval available arguments and their usage.

Author(s)

Zheng Wei

References

Schubert, Lindgreen, and Orlando (2016). AdapterRemoval v2: rapid adapter trimming, identification, and read merging. BMC Research Notes, 12;9(1):88.

Examples

```
adapterremoval_usage()
```

`adapterremoval_version` *Print version information of adapterremoval*

Description

Print version information of adapterremoval

Usage

```
adapterremoval_version()
```

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistakes

Author(s)

Zheng Wei

References

Schubert, Lindgreen, and Orlando (2016). AdapterRemoval v2: rapid adapter trimming, identification, and read merging. BMC Research Notes, 12;9(1):88.

Examples

```
adapterremoval_version()
```

`bowtie2` *Interface to bowtie2 of bowtie2-2.2.3*

Description

This function can be use to call wrapped `bowtie2` binary.

Usage

```
bowtie2(
  bt2Index,
  samOutput,
  seq1,
  ...,
  seq2 = NULL,
  interleaved = FALSE,
  overwrite = FALSE
)
```

Arguments

`bt2Index`	`Character` scalar. bowtie2 index files prefix: 'dir/basename' (minus trailing '.*.bt2' of 'dir/basename.*.bt2').
`samOutput`	`Character` scalar. A path to a SAM file used for the alignment output.
`seq1`	`Character` vector. For single-end sequencing, it contains sequence file paths. For paired-end sequencing, it can be file paths with #1 mates paired with file paths in seq2. And it can also be interleaved file paths when argument interleaved=`TRUE`
`...`	Additional arguments to be passed on to the binaries. See below for details.
`seq2`	`Character` vector. It contains file paths with #2 mates paired with file paths in seq1. For single-end sequencing files and interleaved paired-end sequencing files(argument interleaved=`TRUE`), it must be `NULL`.
`interleaved`	`Logical`. Set `TRUE` when files are interleaved paired-end sequencing data.
`overwrite`	`Logical`. Force overwriting of existing files if setting `TRUE`.

Details

All additional arguments in ... are interpreted as additional parameters to be passed on to bowtie2. All of them should be `Character` or `Numeric` scalar. You can put all aditional arguments in one `Character`(e.g. "–threads 8 –no-mixed") with white space splited just like command line, or put them in different `Character` (e.g. "–threads","8","–no-mixed"). Note that some arguments("-x","–interleaved","-U","-1","-2","-S") to the bowtie2 are invalid if they are already handled as explicit function arguments. See the output of `bowtie2_usage()` for details about available parameters.

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistakes Otherwise the value is non-zero.

Author(s)

Zheng Wei

References

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature methods, 9(4), 357-359.

Examples

```
td <- tempdir()
## Building a bowtie2 index
refs <- dir(system.file(package="Rbowtie2", "extdata", "bt2","refs"),
full=TRUE)
bowtie2_build(references=refs, bt2Index=file.path(td, "lambda_virus"),
"--threads 4 --quiet",overwrite=TRUE)
## Alignments
reads_1 <- system.file(package="Rbowtie2", "extdata", "bt2", "reads",
"reads_1.fastq")
reads_2 <- system.file(package="Rbowtie2", "extdata", "bt2", "reads",
"reads_2.fastq")
if(file.exists(file.path(td, "lambda_virus.1.bt2"))){
    cmdout<-bowtie2(bt2Index = file.path(td, "lambda_virus"),
       samOutput = file.path(td, "result.sam"),
       seq1=reads_1,seq2=reads_2,overwrite=TRUE,"--threads 3");cmdout
    head(readLines(file.path(td, "result.sam")))
}
```

`bowtie2-build` *Interface to bowtie2-2.4.4 build function*

Description

This function can be use to call the bowtie2-build wrapper which wraps the `bowtie2-build-s` and the `bowtie2-build-l` binaries.

Usage

```
bowtie2_build(references, bt2Index, ..., overwrite = FALSE)
```

Arguments

`references`	`Character` vector. The path to the files containing the references for which to build a bowtie index.
`bt2Index`	`Character` scalar. The path where the bowtie2 index files should be created. Include the basename of the index file at the end of the path (i.e. path_to_index_dir/index_basename).
`...`	Additional arguments to be passed on to the binaries. See below for details.
`overwrite`	`Logical`. Force overwriting of existing files if setting `TRUE`.

Details

All additional arguments in ... are interpreted as additional parameters to be passed on to bowtie2_build wrapper. All of them should be `Character` or `Numeric` scalar. You can put all additional arguments in one `Character` (e.g. "–threads 8 –quiet") with white space separation, or put them in different `Character` (e.g. "–threads","8","–quiet"). See the output of `bowtie2_build_usage()` for details about available parameters.

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistakes Otherwise the value is non-zero.

Author(s)

Zheng Wei

References

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature methods, 9(4), 357-359.

Examples

```
td <- tempdir()

## Building a bowtie2 index

refs <- dir(system.file(package="Rbowtie2", "extdata", "bt2","refs"),
full=TRUE)

bowtie2_build(references=refs, bt2Index=file.path(td, "lambda_virus"),
"--threads 4 --quiet",overwrite=TRUE)

## Use additional arguments in another way

bowtie2_build(references=refs, bt2Index=file.path(td, "lambda_virus"),
"--threads",4,"--quiet",overwrite=TRUE)

## The function will print the output during the process without "--quiet" argument.
bowtie2_build(references=refs, bt2Index=file.path(td, "lambda_virus"),
overwrite=TRUE)
```

`bowtie2_build_usage` *Print available arguments that can be passed to bowtie2_build()*

Description

Calling bowtie2_build_usage() prints the available arguments that can be passed to the ... argument of the bowtie2_build() function of the package. Note that some arguments are invalid if they are already handled as explicit function arguments.

Usage

```
bowtie2_build_usage()
```

Value

Information about available arguments that can be passed to bowtie2_build()

Author(s)

Zheng Wei

References

Langmead B, Salzberg S. Fast gapped-read alignment with Bowtie 2. Nature Methods. 2012, 9:357-359.

Examples

```
bowtie2_build_usage()
```

`bowtie2_samtools` *Interface to bowtie2-2.4.4 align function*

Description

This function can be use to call the bowtie2 wrapper which wraps the `bowtie2-align-s` and the `bowtie2-align-l` binaries.

Usage

```
bowtie2_samtools(
  bt2Index,
  output,
  outputType = "sam",
  seq1 = NULL,
  seq2 = NULL,
  bamFile = NULL,
  ...,
  interleaved = FALSE,
  overwrite = FALSE
)
```

Arguments

`bt2Index`	`Character` scalar. The path where the bowtie2 index files are located. Include the basename of the index files at the end of the path (i.e. path_to_index_dir/index_basename). Do not include the bowtie2 index file extension (.bt2 or .bt2l)
`output`	`Character` scalar. The path where the alignment output file should be created. Include the basename of the alignment file at the end of the path (i.e. path_to_output_dir/output_basename). Do not include the alignment file extension (.sam or .bam).
`outputType`	`Character` scalar. Specify the output alignment file type. Default is set to "sam" but can also be changed to "bam".
`seq1`	`Character` vector. For single-end sequencing, it contains sequence file paths. For paired-end sequencing, it can be file paths with #1 mates paired with file paths in seq2. And it can also be interleaved file paths when argument interleaved=`TRUE`.
`seq2`	`Character` vector. It contains file paths with #2 mates paired with file paths in seq1. For single-end sequencing files and interleaved paired-end sequencing files (argument interleaved=`TRUE`), it must be `NULL`.
`bamFile`	`Character` vector. A path to a bam file that contains unaligned reads. If a bam file is provided then seq1 and seq2 must be set to `NULL`
`...`	Additional arguments to be passed on to the bowtie2 wrapper. See below for details.
`interleaved`	`Logical`. Set `TRUE` when files are interleaved paired-end sequencing data.
`overwrite`	`Logical`. Force overwriting of existing files if setting `TRUE`.

Details

All additional arguments in ... are interpreted as additional parameters to be passed to bowtie2 wrapper. All of them should be `Character` or `Numeric` scalar. You can put all additional arguments in one `Character` (e.g. "–threads 8 –no-mixed") with white space separation, or put them in different `Character` (e.g. "–threads","8","–no-mixed"). Note that some arguments ("-x","–interleaved","-U","-1","-2","-b","-S") are invalid if they are already handled as explicit function arguments. See the output of `bowtie2_usage()` for details about available parameters.

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistakes Otherwise the value is non-zero.

Author(s)

Zheng Wei

References

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature methods, 9(4), 357-359.

Examples

```
td <- tempdir()

## Building a bowtie2 index
refs <- dir(system.file(package="Rbowtie2", "extdata", "bt2","refs"), full=TRUE)
bowtie2_build(references=refs, bt2Index=file.path(td, "lambda_virus"),"--threads 4 --quiet",overwrite=TRUE)

## Alignments
reads_1 <- system.file(package="Rbowtie2", "extdata", "bt2", "reads", "reads_1.fastq")
reads_2 <- system.file(package="Rbowtie2", "extdata", "bt2", "reads", "reads_2.fastq")

## Sam file created
bowtie2_samtools(bt2Index = file.path(td,"lambda_virus"), output = file.path(td,"example"), seq1 = reads_1,
seq2 = reads_2, overwrite = TRUE)

## Bam file created
bowtie2_samtools(bt2Index = file.path(td,"lambda_virus"), output = file.path(td,"example"), outputType = "bam
seq1 = reads_1, seq2 = reads_2, overwrite = TRUE)
```

`bowtie2_usage` *Print available arguments that can be passed to bowtie2()*

Description

Calling bowtie2_usage() prints the available arguments that can be passed to the ... argument of the bowtie2() function of the package. Note that some arguments are invalid if they are already handled as explicit function arguments.

Usage

```
bowtie2_usage()
```

Value

Information about available arguments that can be passed to bowtie2().

Author(s)

Zheng Wei

References

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature methods, 9(4), 357-359.

Examples

```
bowtie2_usage()
```

`bowtie2_version` *Print version information of bowtie2-2.4.4*

Description

Calling bowtie2_version() prints the version information of the bowtie package used.

Usage

```
bowtie2_version()
```

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistakes Otherwise the value is non-zero.

Author(s)

Zheng Wei

References

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature methods, 9(4), 357-359.

Examples

```
bowtie2_version()
```

`checkFileCreateable` *Check if file is creatable*

Description

This is an internal function that is not meant to be used outside of the package. It determines whether the file at the end of the path can be created.

Usage

```
checkFileCreatable(filePath, argname, overwrite)
```

Arguments

`filePath`	`Character` scalar. Path to file to be created.
`argname`	`Character` scalar. Name of the argument passed.
`overwrite`	`Logical` Will warn user of overwriting if set `TRUE`.

Value

Indicates whether the file can be created at the path given.

Author(s)

Zheng Wei

`checkFileExist` *Check if file exists*

Description

This is an internal function that is not meant to be used outside of the package. It determines whether a specific file exists at the end of the path given to the function.

Usage

```
checkFileExist(filePath, argname)
```

Arguments

`filePath`	`Character` scalar. Path to file of interest.
`argname`	`Character` scalar. Name of the argument passed.

Value

Stops the function if the file does not exist.

Author(s)

Zheng Wei

`checkIndexType` *Determine bowtie2 index type*

Description

This is an internal function that is not meant to be used outside of the package. It determines whether the given index library comprises of small indexes (.bt2) or large indexes (.bt2l).

Usage

```
checkIndexType(filePath)
```

Arguments

`filePath`	`Character` scalar. The path where bowtie2 index files are located. Include the basename of the index files at the end of the path (i.e. path_to_index_dir/index_basename). Do not include the bowtie2 index file extension (.bt2 or .bt2l).

Details

The function first tries to determine whether path_to_index_dir/index_basename.1.bt2 exists and if it doesn't exist then it tries to determine whether path_to_index_dir/index_basename.1.bt2l exists. If neither of those files exist then there is an issue with the index files that must be addressed.

Value

`Character` scalar. Returns either "SMALL" if the .bt2 file is found, "LARGE" if the .bt2l file is found, or "ERROR" if neither the .bt2 nor .bt2l file is found.

`checkPathExist` *Check if path exists*

Description

This is an internal function that is not meant to be used outside of the package. It determines whether the path passed to the function exists.

Usage

```
checkPathExist(filePath, argname)
```

Arguments

`filePath`	`Character` scalar. Path of interest.
`argname`	`Character` scalar. Name of the argument passed.

Value

Stops the function if the path does not exist.

Author(s)

Zheng Wei

`checkSamtoolsExists` *Check if samtools exists on the system*

Description

This is an internal function that is not meant to be used outside of the package. It checks whether samtools exists on the system.

Usage

```
checkSamtoolsExists()
```

Value

Returns TRUE if samtools exists on the system, else FALSE

`identify_adapters` *identify adapters for paired-end reads*

Description

This function can be use to call wrapped `AdapterRemoval` binary for adapters identifying.

Usage

```
identify_adapters(file1, file2, ..., basename = NULL, overwrite = FALSE)
```

Arguments

`file1`	`Character` vector. It can be file paths with #1 mates paired with file paths in file2 And it can also be interleaved file paths when argument interleaved=`TRUE`
`file2`	`Character` vector. It contains file paths with #2 mates paired with file paths in file1. For interleaved paired-end sequencing files(argument interleaved=`TRUE`),it must to be setted to `NULL`.
`...`	Additional arguments to be passed on to the binaries. See below for details.
`basename`	`Character`. The outputfile path prefix. Default: your_output
`overwrite`	`Logical`. Force overwriting of existing files if setting `TRUE`.

Details

All additional arguments in ... are interpreted as additional parameters to be passed on to identify_adapters. All of them should be `Character` or `Numeric` scalar. You can put all aditional arguments in one `Character`(e.g. "–threads 8") with white space splited just like command line, or put them in different `Character`(e.g. "–threads","8"). Note that some arguments("–identify-adapters", "–file1","–file2","–basename") to the identify_adapters are invalid if they are already handled as explicit function arguments. See the output of `adapterremoval_usage()` for details about available parameters.

Value

An invisible `Character` vector of adapters for each mate.

Author(s)

Zheng Wei

References

Schubert, Lindgreen, and Orlando (2016). AdapterRemoval v2: rapid adapter trimming, identification, and read merging. BMC Research Notes, 12;9(1):88.

Examples

```
td <- tempdir()
reads_1 <- system.file(package="Rbowtie2", "extdata", "adrm", "reads_1.fq")
reads_2 <- system.file(package="Rbowtie2", "extdata", "adrm", "reads_2.fq")
adapters <- identify_adapters(file1=reads_1,file2=reads_2,
basename = file.path(td,"reads")
,"--threads 2",overwrite=TRUE)
adapters
```

`remove_adapters` *Interface to bowtie2 of adapterremoval-2.2.1a*

Description

This function can be use to call wrapped `AdapterRemoval` binary.

Usage

```
remove_adapters(
  file1,
  ...,
  adapter1 = NULL,
  output1 = NULL,
  file2 = NULL,
  adapter2 = NULL,
  output2 = NULL,
  basename = NULL,
  interleaved = FALSE,
  overwrite = FALSE
)
```

Arguments

`file1`	`Character` vector. For single-end sequencing, it contains sequence file paths. For paired-end sequencing, it can be file paths with #1 mates paired with file paths in file2 And it can also be interleaved file paths when argument interleaved=`TRUE`
`...`	Additional arguments to be passed on to the binaries. See below for details.
`adapter1`	`Character`. It is an adapter sequence for file1. Default: AGATCGGAAGAGCACACGTCTGAACTCCAGTCACNNNNNNATCTCGTATGCCGTCTTCTGCTTG
`output1`	`Character`. The trimmed mate1 reads output file path for file1. Defualt: basename.pair1.truncated (paired-end), basename.truncated (single-end), or basename.paired.truncated (interleaved)
`file2`	`Character` vector. It contains file paths with #2 mates paired with file paths in file1. For single-end sequencing files and interleaved paired-end sequencing files(argument interleaved=`TRUE`), it must be `NULL`.
`adapter2`	`Character`. It is an adapter sequence for file2. Defualt: AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT

`output2`	`Character`. The trimmed mate2 reads output file path for file2. Default: BASENAME.pair2.truncated (only used in PE mode, but not if –interleaved-output is enabled)
`basename`	`Character`. The outputfile path prefix. Default: your_output
`interleaved`	`Logical`. Set `TRUE` when files are interleaved paired-end sequencing data.
`overwrite`	`Logical`. Force overwriting of existing files if setting `TRUE`.

Details

All additional arguments in ... are interpreted as additional parameters to be passed on to remove_adapters. All of them should be `Character` or `Numeric` scalar. You can put all aditional arguments in one `Character`(e.g. "–threads 8") with white space splited just like command line, or put them in different `Character`(e.g. "–threads","8"). Note that some arguments("–file1","–file2","–adapter1","–adapter2","–output1","–output2", "–basename","–interleaved") to the identify_adapters are invalid if they are already handled as explicit function arguments. See the output of `adapterremoval_usage()` for details about available parameters.

Value

An invisible `Integer` of call status. The value is 0 when there is not any mistake. Otherwise the value is non-zero.

Author(s)

Zheng Wei

References

Schubert, Lindgreen, and Orlando (2016). AdapterRemoval v2: rapid adapter trimming, identification, and read merging. BMC Research Notes, 12;9(1):88.

Examples

```
td <- tempdir()

# Identify adapters
reads_1 <- system.file(package="Rbowtie2", "extdata", "adrm", "reads_1.fq")
reads_2 <- system.file(package="Rbowtie2", "extdata", "adrm", "reads_2.fq")
adapters <- identify_adapters(file1=reads_1,file2=reads_2,
basename=file.path(td,"reads"), "--threads 3",overwrite=TRUE)

# Remove adapters
cmdout<-remove_adapters(file1=reads_1,file2=reads_2,adapter1 = adapters[1],
adapter2 = adapters[2],
output1=file.path(td,"reads_1.trimmed.fq"),
output2=file.path(td,"reads_2.trimmed.fq"),
basename=file.path(td,"reads.base"),overwrite=TRUE,"--threads 3");cmdout
```

Index

.callbinary, 2

adapterremoval_usage, 3
adapterremoval_version, 3

bowtie2, 4
bowtie2-build, 5
bowtie2_build *(*bowtie2-build*)*, 5
bowtie2_build_usage, 7
bowtie2_samtools, 7
bowtie2_usage, 9
bowtie2_version, 10

checkFileCreatable
        *(*checkFileCreateable*)*, 10
checkFileCreateable, 10
checkFileExist, 11
checkIndexType, 11
checkPathExist, 12
checkSamtoolsExists, 12

identify_adapters, 13

remove_adapters, 14