Package 'blit'

February 27, 2025

Title Bioinformatics Library for Integrated Tools

Version 0.1.0

Description An all-encompassing R toolkit designed to streamline the process of calling various bioinformatics software and then performing data analysis and visualization in R. With 'blit', users can easily integrate a wide array of bioinformatics command line tools into their workflows, leveraging the power of R for sophisticated data manipulation and graphical representation.

License GPL (>= 3)

Imports cli, data.table, R6, rlang (>= 1.1.0), sys, utils, withr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

SystemRequirements Unix System

URL https://github.com/WangLabCSU/blit

BugReports https://github.com/WangLabCSU/blit/issues

NeedsCompilation no

Author Yun Peng [aut, cre] (<https://orcid.org/0000-0003-2801-3332>), Shixiang Wang [aut] (<https://orcid.org/0000-0001-9855-7357>), Jennifer Lu [cph] (Author of the included scripts from Kraken2 and KrakenTools libraries), Li Song [cph] (Author of included scripts from TRUST4 library), X. Shirley Liu [cph] (Author of included scripts from TRUST4 library)

Maintainer Yun Peng <yunyunp96@163.com>

Repository CRAN

Date/Publication 2025-02-27 16:50:02 UTC

Contents

`allele_counter` *Run alleleCount*

Description

The `alleleCount` program primarily exists to prevent code duplication between some other projects, specifically `AscatNGS` and `Battenberg`.

Usage

```
allele_counter(
  hts_file,
  loci_file,
  ofile,
  ...,
  odir = getwd(),
  alleleCounter = NULL
)
```

Arguments

`hts_file`	A string of path to sample HTS file.
`loci_file`	A string of path to loci file.
`ofile`	A string of path to the output file.

...	<dynamic dots> Additional arguments passed to alleleCounter command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(alleleCounter()).
odir	A string of path to the output directory.
alleleCounter	A string of path to alleleCounter command.

Value

A command object.

See Also

https://github.com/cancerit/alleleCount

---

arg *Deliver arguments of command*

Description

Deliver arguments of command

Usage

```
arg(tag, value, indicator = FALSE, lgl2int = FALSE, format = "%s", sep = " ")
```

Arguments

tag	A string specifying argument tag, like "-i", "-o".
value	Value passed to the argument.
indicator	A logical value specifying whether value should be an indicator of tag. If TRUE, logical value will explain the set or unset of tag.
lgl2int	A logical value indicates whether transfrom value TRUE to 1 or FALSE to 0. If TRUE, format will always be set to "%d".
format	The format of the value, details see sprintf.
sep	A character string used to separate "tag" and "value", usually " " or "=".

Value

A string.

cellranger *Run cellranger*

Description

Run cellranger

Usage

```
cellranger(subcmd = NULL, ..., cellranger = NULL)
```

Arguments

subcmd	Sub-Command of cellranger.
...	<dynamic dots> Additional arguments passed to `cellranger subcmd` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`. Details see: `cmd_help(cellranger subcmd())`.
cellranger	A string of path to `cellranger` command.

Value

A command object.

cmd_run *Execute command*

Description

- `cmd_run`: Run the command.
- `cmd_background`: Run the command in the background.
- `cmd_help`: Print the help document for this command.

Usage

```
cmd_run(
  command,
  stdout = TRUE,
  stderr = TRUE,
  stdin = NULL,
  timeout = NULL,
  verbose = TRUE
)
```

```
cmd_background(
  command,
  stdout = NULL,
  stderr = NULL,
  stdin = NULL,
  verbose = TRUE
)

cmd_help(command, stdout = NULL, stderr = NULL, verbose = TRUE)
```

Arguments

command	A command object.
stdout, stderr	How output streams of the child process are processed. Possible values are: • TRUE: print the child output in R console. • FALSE: suppress output stream • **string**: name or path of file to redirect output • connection: a writable R connection object • function: a callback function (including purrr-like lambda syntax) with one argument accepting a raw vector (use as_text() to convert to text). For cmd_background(), only a string (file path), or a single boolean value can be used. For cmd_help(), only a string (file path), connection, or function can be used.
stdin	should the input be diverted? A character string naming a file.
timeout	Timeout in seconds. This is a limit for the elapsed time running command in the separate process.
verbose	A single boolean value indicating whether the command execution should be verbose.

Value

- cmd_run: Exit status invisiblely.
- cmd_background: Returns the process ID, which can be terminated manually using tools::pskill(). You can also use sys::exec_status() to check the process's exit status.
- cmd_help: the input command invisiblely.

See Also

cmd_wd()/cmd_envvar()/cmd_envpath()

cmd_wd *Define the environment when running the command*

Description

- cmd_wd: define the working directory.
- cmd_envvar: define the environment variables.
- cmd_envpath: define the PATH-like environment variables.

Usage

```
cmd_wd(command, wd = NULL)

cmd_envvar(command, ..., action = "replace", sep = " ")

cmd_envpath(command, ..., action = "prefix", name = "PATH")
```

Arguments

command	A command object.
wd	A string or NULL define the working directory of the command.
...	• cmd_envvar: Named character define the environment variables. • cmd_envpath: Unnamed character to define the PATH-like environment variables name.
action	Should the new values "replace", "prefix" or "suffix" existing environment variables?
sep	A string to separate new and old value when action is "prefix" or "suffix".
name	A string define the PATH environment variable name. You can use this to define other PATH-like environment variable such as PYTHONPATH.

Value

- cmd_wd: The command object itself, with working directory updated.
- cmd_envvar: The command object itself, with running environment variable updated.
- cmd_envpath: The command object self, with running environment variable name updated.

See Also

cmd_run()/cmd_background()/cmd_help()

Command *R6 Class to prepare command parameters*

Description

R6 Class to prepare command parameters

R6 Class to prepare command parameters

Methods

Public methods:

- `Command$new()`
- `Command$evaluate()`
- `Command$build()`
- `Command$print()`
- `Command$clone()`

Method `new()`**:** Create a new `Command` object.

Usage:

```
Command$new(..., .subcmd = NULL)
```

Arguments:

`...` Additional argument passed into command.

`.subcmd` Sub-command string.

Method `evaluate()`**:** Evaluate the parameters to execute command.

Usage:

```
Command$evaluate()
```

Returns: The object itself.

Method `build()`**:** Build parameters to run command.

Usage:

```
Command$build(help = FALSE, verbose = TRUE, envir = caller_env())
```

Arguments:

`help` A boolean value indicating whether to build parameters for help document or not.

`verbose` A boolean value indicating whether the command execution should be verbose.

`envir` An environment used to Execute command.

Returns: An atomic character combine the command and parameters.

Method `print()`**:** Build parameters to run command.

Usage:

```
Command$print(indent = NULL)
```

Arguments:

`indent` A single integer number giving the space of indent.

Returns: The object itself.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Command$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`exec` *Invoke a System Command*

Description

Invoke a System Command

Usage

```
exec(cmd, ...)
```

Arguments

`cmd`	Command to be invoked, as a character string.
`...`	<dynamic dots> Additional arguments passed to `cmd` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`.

Value

A `command` object.

See Also

- `cmd_wd()`/`cmd_envvar()`/`cmd_envpath()`
- `cmd_run()`/`cmd_background()`/`cmd_help()`

Examples

```
cmd_run(exec("echo", "$PATH"))
```

`fastq_pair` *FASTQ PAIR*

Description

Rewrite paired end fastq files to make sure that all reads have a mate and to separate out singletons.

Usually when you get paired end read files you have two files with a /1 sequence in one and a /2 sequence in the other (or a /f and /r or just two reads with the same ID). However, often when working with files from a third party source (e.g. the SRA) there are different numbers of reads in each file (because some reads fail QC). Spades, bowtie2 and other tools break because they demand paired end files have the same number of reads.

Usage

```
fastq_pair(
  fq1,
  fq2,
  ...,
  hash_table_size = NULL,
  max_hash_table_size = NULL,
  fastq_pair = NULL
)

fastq_read_pair(fastq_files)
```

Arguments

`fq1`, `fq2`	A string of fastq file path.
`...`	<dynamic dots> Additional arguments passed to `fastq_pair` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`. Details see: `cmd_help(fastq_pair())`.
`hash_table_size`	Size of hash table to use.
`max_hash_table_size`	Maximal hash table size to use.
`fastq_pair`	A string of path to `fastq_pair` command.
`fastq_files`	A character of the fastq file paths.

Value

A `command` object.

See Also

https://github.com/linsalrob/fastq-pair

`gistic2` *Run GISTIC2*

Description

The GISTIC module identifies regions of the genome that are significantly amplified or deleted across a set of samples. Each aberration is assigned a G-score that considers the amplitude of the aberration as well as the frequency of its occurrence across samples. False Discovery Rate q-values are then calculated for the aberrant regions, and regions with q-values below a user-defined threshold are considered significant. For each significant region, a "peak region" is identified, which is the part of the aberrant region with greatest amplitude and frequency of alteration. In addition, a "wide peak" is determined using a leave-one-out algorithm to allow for errors in the boundaries in a single sample. The "wide peak" boundaries are more robust for identifying the most likely gene targets in the region. Each significantly aberrant region is also tested to determine whether it results primarily from broad events (longer than half a chromosome arm), focal events, or significant levels of both. The GISTIC module reports the genomic locations and calculated q-values for the aberrant regions. It identifies the samples that exhibit each significant amplification or deletion, and it lists genes found in each "wide peak" region.

Usage

```
gistic2(seg, refgene, ..., odir = getwd(), gistic2 = NULL)
```

Arguments

`seg`	A data.frame of segmented data.
`refgene`	Path to reference genome data input file (REQUIRED, see below for file description).
`...`	<dynamic dots> Additional arguments passed to `gistic2` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`. Details see: `cmd_help(gistic2())`.
`odir`	A string of path to the output directory.
`gistic2`	A string of path to `gistic2` command.

Value

A `command` object.

See Also

https://broadinstitute.github.io/gistic2/

kraken2 *Running Kraken2*

Description

Kraken is a taxonomic sequence classifier that assigns taxonomic labels to DNA sequences. Kraken examines the k-mers within a query sequence and uses the information within those k-mers to query a database. That database maps k-mers to the lowest common ancestor (LCA) of all genomes known to contain a given k-mer.

Usage

```
kraken2(
  fq1,
  ...,
  fq2 = NULL,
  ofile = "kraken_output.txt",
  report = "kraken_report.txt",
  classified_out = NULL,
  unclassified_out = NULL,
  odir = getwd(),
  kraken2 = NULL
)
```

Arguments

fq1, fq2	A string of fastq file path.
...	<dynamic dots> Additional arguments passed to kraken2 command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(kraken2()).
ofile	A string of path to save kraken2 output.
report	A string of path to save kraken2 report.
classified_out	A string of path to save classified sequences, which should be a fastq file.
unclassified_out	A string of path to save unclassified sequences, which should be a fastq file.
odir	A string of path to the output directory.
kraken2	A string of path to kraken2 command.

Value

A command object.

See Also

- https://github.com/DerrickWood/kraken2/wiki/Manual
- https://benlangmead.github.io/aws-indexes/k2

`kraken_tools` *KrakenTools is a suite of scripts to be used alongside the Kraken, KrakenUniq, Kraken 2, or Bracken programs.*

Description

These scripts are designed to help Kraken users with downstream analysis of Kraken results.

Usage

```
kraken_tools(script, ..., python = NULL)
```

Arguments

`script`	Name of the kraken2 script. One of `"combine_kreports"`, `"combine_mpa"`, `"extract_kraken_reads"`, `"filter_bracken_out"`, `"fix_unmapped"`, `"kreport2krona"`, `"kreport2mpa"`, `"make_kreport"`, and `"make_ktaxonomy"`.
`...`	<dynamic dots> Additional arguments passed to `kraken_tools` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`. Details see: `cmd_help(kraken_tools())`.
`python`	A string of path to `python` command.

Value

A `command` object.

See Also

https://github.com/jenniferlu717/KrakenTools

make_command *Helper function to create new command*

Description

Helper function to create new command

Usage

```
make_command(name, fun, envir = caller_env())
```

Arguments

name	A string of the function name.
fun	A function used to initialize the Command object.
envir	A environment used to bind the created function.

Value

A function.

perl *Perl is a highly capable, feature-rich programming language with over 36 years of development.*

Description

Perl is a highly capable, feature-rich programming language with over 36 years of development.

Usage

```
perl(..., perl = NULL)
```

Arguments

...	<dynamic dots> Additional arguments passed to perl command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(perl()).
perl	A string of path to perl command.

Value

A command object.

See Also

https://www.perl.org/

`pyscenic` *Run pyscenic*

Description

Run pyscenic

Usage

```
pyscenic(subcmd = NULL, ..., pyscenic = NULL)
```

Arguments

`subcmd`	Sub-Command of pyscenic.
`...`	<dynamic dots> Additional arguments passed to `pyscenic subcmd` command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using `shQuote`. Details see: `cmd_help(pyscenic subcmd())`.
`pyscenic`	A string of path to `pyscenic` command.

Value

A `command` object.

References

https://github.com/aertslab/pySCENIC

`python` *Python is a programming language that lets you work quickly and integrate systems more effectively.*

Description

Python is a programming language that lets you work quickly and integrate systems more effectively.

Usage

```
python(..., python = NULL)
```

Arguments

...	<dynamic dots> Additional arguments passed to python command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(python()).
python	A string of path to python command.

Value

A command object.

See Also

https://www.python.org/

---

seqkit *Run seqkit*

Description

Run seqkit

Usage

```
seqkit(subcmd = NULL, ..., seqkit = NULL)
```

Arguments

subcmd	Sub-Command of seqkit.
...	<dynamic dots> Additional arguments passed to seqkit subcmd command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(seqkit subcmd()).
seqkit	A string of path to seqkit command.

Value

A command object.

See Also

https://bioinf.shenwei.me/seqkit/

trust4 *TRUST4: immune repertoire reconstruction from bulk and single-cell RNA-seq data*

Description

TRUST4: immune repertoire reconstruction from bulk and single-cell RNA-seq data

Usage

```
trust4(
  file1,
  ref_coordinate,
  ...,
  file2 = NULL,
  mode = NULL,
  ref_annot = NULL,
  ofile = NULL,
  odir = getwd(),
  trust4 = NULL
)

trust4_imgt_annot(
  species = "Homo_sapien",
  ...,
  ofile = "IMGT+C.fa",
  odir = getwd(),
  perl = NULL
)

trust4_gene_names(imgt_annot, ofile = "bcr_tcr_gene_name.txt", odir = getwd())
```

Arguments

file1	Path to bam file or fastq file.
ref_coordinate	Path to the fasta file coordinate and sequence of V/D/J/C genes.
...	• trust4: <dynamic dots> Additional arguments passed to run-trust4 command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(run-trust4()). • trust4_imgt_annot: <dynamic dots> Additional arguments passed to trust4_imgt_annot command. Empty arguments are automatically trimmed. If a single argument, such as a file path, contains spaces, it must be quoted, for example using shQuote. Details see: cmd_help(trust4_imgt_annot()).
file2	Path to the second paired-end read fastq file, only used for mode = "fastq".
mode	One of "bam" or "fastq". If NULL, will be inferred from file1.

ref_annot	Path to detailed V/D/J/C gene reference file, such as from IMGT database. (default: not used). (recommended).
ofile	• trust4: Prefix of output files. (default: inferred from file prefix). • trust4_imgt_annot: Output file name. • trust4_gene_names: Output file name.
odir	A string of path to the output directory.
trust4	A string of path to run-trust4 command.
species	Species to extract IMGT annotation, details see https://www.imgt.org//download/V-QUEST/IMGT_V-QUEST_reference_directory/.
perl	A string of path to perl command.
imgt_annot	Path of IMGT annotation file, created via trust4_imgt_annot.

Value

A command object.

See Also

https://github.com/liulab-dfci/TRUST4

Index