Package 'TraceQC'

October 4, 2021

Type Package

Title Quality Control of CRISPR Lineage Tracing Seqence Data

Version 1.0.1

Date 2020-06-18

Description Provides a generalized quantify control functions and data processing pipeline across several CRISPR-based lineage tracing platforms that can be directly used in data analysis.

Depends R (>= 3.5.0)

Imports circlize,
tidyr,
RColorBrewer,
ggplot2,
readr,
purrr,
fastqcr,
tictoc,
magrittr,
dplyr,
stringr,
grid,
rmarkdown,
tibble

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

`circular_chordgram` *Display a circos plot with links for a given data frame.*

Description

Display a circos plot with links for a given data frame.

Usage

```
circular_chordgram(df, title, ref, use_log_count = TRUE, count_cutoff = 1)
```

Arguments

`df`	a data frame that contains data to be visualized on the plot
`title`	The main title of the plot
`ref`	A reference object.
`use_log_count`	Color links use log(count + 1).
`count_cutoff`	A cutoff to remove link whose log10-count are less than the value.

Value

It doesn't generate any specific output.

circular_histogram *Display a circos plot with a histgoram for a given data frame.*

Description

Display a circos plot with a histgoram for a given data frame.

Usage

```
circular_histogram(df, title, ref)
```

Arguments

df	a data frame that contains data to be visualized on the plot.
title	The main title of the plot.
ref	A reference object

Value

It doesn't generate any specific output.

filter_mutations *Filter mutations based on read count per UMI*

Description

Filter mutations based on read count per UMI

Usage

```
filter_mutations(data, include_max = TRUE, freq_threshold)
```

Arguments

data	A data frame.
include_max	include the mutations with maximum read count.
freq_threshold	threshold of mutation frequency.

Value

A filtered data frame.

`filter_mutations_per_cell` *Filter mutations based on read count per cell*

Description

Filter mutations based on read count per cell

Usage

```
filter_mutations_per_cell(data, freq_threshold)
```

Arguments

`data`	A data frame.
`freq_threshold`	threshold of mutation frequency.

Value

A filtered data frame.

`filter_mutations_per_UMI` *Filter mutations based on read count per UMI*

Description

Filter mutations based on read count per UMI

Usage

```
filter_mutations_per_UMI(data, include_max = TRUE, freq_threshold)
```

Arguments

`data`	A data frame.
`include_max`	include the mutations with maximum read count.
`freq_threshold`	threshold of mutation frequency.

Value

A filtered data frame.

find_position *Creating a data frame of mutation events.*

Description

Creating a data frame of mutation events.

Usage

```
find_position(insertions, deletions, mutations, target_seq, score, read_count)
```

Arguments

insertions	A list that contains insertion events.
deletions	A list that contains deletion events.
mutations	A list that contains mutation (substitution) events.
target_seq	A list that contains the alignment for each event.
score	A list that contains the alignment score for each event.
read_count	A vector that contains counts for each event.

Value

A data frame that contains the event information.

format_mutation_df *format a mutation data frame for output.*

Description

format a mutation data frame for output.

Usage

```
format_mutation_df(mutation_df, is_singlecell)
```

Arguments

mutations	A data frame of mutations. The output of seq_to_character.

Value

A formatted data frame of mutations.

get_abspath *Get absolute path of a file.*

Description

Get absolute path of a file.

Usage

```
get_abspath(f)
```

Arguments

f	A relative or absolute file path.

Value

It returns an absolute path for a file.

get_read_count_per_UB *Get read count per UMI.*

Description

Get read count per UMI.

Usage

```
get_read_count_per_UB(df)
```

Arguments

df	aligned reads

Value

A data frame of UMI

get_UMI_count_per_CB *Get UMI count per cell.*

Description

Get UMI count per cell.

Usage

```
get_UMI_count_per_CB(df)
```

Arguments

df	aligned reads

Value

A data frame of Cells.

mutation_type_donut *A pie chart that shows a summary of mutation types.*

Description

A pie chart that shows a summary of mutation types.

Usage

```
mutation_type_donut(mutations)
```

Arguments

mutations	A mutation dataframe

Value

A ggplot2 object that shows the pie chart

Examples

```
data(example_obj)
mutation_type(example_obj)
```

num_mutation_histogram *A barplot to show distribution of the number of mutations per barcode*

Description

A barplot to show distribution of the number of mutations per barcode

Usage

```
num_mutation_histogram(mutations)
```

Arguments

mutations	A mutation dataframe

Value

A ggplot2 object that shows the barplot

Examples

```
data(example_obj)
num_mutation_histogram(example_obj)
```

parse_ref_file *Parsing reference sequence file*

Description

Parsing reference sequence file

Usage

```
parse_ref_file(ref_file)
```

Arguments

ref_file	A path of a reference sequence file.

Value

A list with those four elements.

- 'refseq': The reference sequence.
- 'regions': Detailed information about the reference sequence.

plot_alignment_permutation *Visualization of alignment permutation.*

Description

Visualization of alignment permutation.

Usage

```
plot_alignment_permutation(alignment_permutation)
```

Arguments

alignment_permutation	an data frame of permutation sequence, output of ‘sequence_permutation‘

Value

it returns A ggplot2 object that shows the permutation.

plot_construct *Visualization of the construct (reference sequence) information.*

Description

Visualization of the construct (reference sequence) information.

Usage

```
plot_construct(ref, chr_per_row = 50, chr_size = 10)
```

Arguments

ref	an reference object, output of ‘parse_ref_file‘
chr_per_row	number of characters per row.
chr_size	the text size of character.

Value

it returns A ggplot2 object that shows the construct information.

`plot_deletion_hotspot` *Display a circos plot that shows overall deletion pattern across the barcodes.*

Description

Display a circos plot that shows overall deletion pattern across the barcodes.

Usage

```
plot_deletion_hotspot(mutations, ref, use_log_count = TRUE, count_cutoff = 1)
```

Arguments

`mutations`	A mutations dataframe.
`ref`	A reference object.
`use_log_count`	Color links use log(count + 1).
`count_cutoff`	A cutoff to remove link whose log10-count are less than the value.

Value

It doesn't generate any specific output.

Examples

```
data(example_obj)
plot_deletion_hotspot(example_obj)
```

`plot_insertion_hotspot` *Display a circos plot that shows overall insertion pattern across the barcodes.*

Description

Display a circos plot that shows overall insertion pattern across the barcodes.

Usage

```
plot_insertion_hotspot(mutations, ref, use_log_count = TRUE, count_cutoff = 1)
```

Arguments

mutations	A mutations dataframe.
ref	A reference object.
use_log_count	Color links use log(count + 1).
count_cutoff	A cutoff to remove link whose log10-count are less than the value.

Value

It won't return any specific object.

Examples

```
data(example_obj)
plot_insertion_hotspot(example_obj)
```

---

plot_lorenz_curve *Drawing Lorenz Curve*

Description

The Lorenz curve shows an inequality of barcode distribution of the sample.

Usage

```
plot_lorenz_curve(aligned_reads)
```

Arguments

aligned_reads	A aligned_reads dataframe.

Value

A ggplot2 object that shows Lorenz Curve

Examples

```
data(example_obj)
plot_lorenz_curve(example_obj)
```

plot_point_substitution_hotspot

Display a mutation hotspot circos plot.

Description

The circos plot shows the frequency of mutation events for each nucleotide.

Usage

```
plot_point_substitution_hotspot(mutations, ref)
```

Arguments

mutations	A mutations dataframe
ref	A reference object

Value

It won't return any specific object.

Examples

```
data(example_obj)
plot_point_mutation_hotspot(example_obj)
```

plot_score_distribution

Drawing a score distribution plot

Description

Drawing a score distribution plot

Usage

```
plot_score_distribution(aligned_reads)
```

Arguments

aligned_reads	A aligned_reads dataframe.

Value

A ggplot2 object that shows alignment score distribution.

Examples

```
data(example_obj)
plot_score_distribution(example_obj)
```

`sequence_alignment` *Function for a sequence alignment between the reference file and sample.*

Description

The function is an wrapper of a python function which performs a global pairwise sequence alignment by biopython package.

Usage

```
sequence_alignment(
  input_file,
  ref_file,
  output_file = "aligned_reads.txt",
  python_path = "python3",
  match = 2,
  mismatch = -2,
  gapopen = -6,
  gapextension = -0.1,
  ncores = 4,
  penalize_end_gaps = 1,
  return_df = FALSE
)
```

Arguments

`input_file`	A FASTQ file path
`ref_file`	A path of a reference sequence file.
`output_file`	The output path. An output of the alignment will be stored at the path.
`python_path`	The path to Python interpreter.
`match`	The score for a correct basepair matching.
`mismatch`	The penalty score for a basepair mismatching.
`gapopen`	The gap opening score for the alignment.
`gapextension`	The gap extension score for the alignment.
`ncores`	The number of cores for the parallel processing
`penalize_end_gaps`	If penalizing the end gap when performing alignment (0 or 1).

Value

It returns a data frame of the alignment result if 'return_df' is 'T' and 'NULL' otherwise.

Examples

```
## Not run:
library(TraceQC)
input_file <- system.file("extdata", "test_data",
                          "raw_sequences", "hgRNA_example.fastq.gz", package="TraceQC")
ref_file <- system.file("extdata", "test_data", "ref",
                        "ref_hgRNA_invitro.txt", package="TraceQC")
output_file <- tempfile()
sequence_alignment(input_file=input_file,
                   ref_file=ref_file,
                   output_file=output_file)
aligned_reads <- read_tsv(output_file)

## End(Not run)
```

`sequence_alignment_for_10x`

Function for a sequence alignment between the reference file and sample for 10x data.

Description

The function is an wrapper of a python function which performs a global pairwise sequence alignment by biopython package.

Usage

```
sequence_alignment_for_10x(
  input_file,
  ref_file,
  output_file = "aligned_reads.txt",
  python_path = "python3",
  match = 2,
  mismatch = -2,
  gapopen = -6,
  gapextension = -0.1,
  penalize_end_gaps = 1,
  ncores = 4
)
```

Arguments

input_file	A file path of possorted_genome_bam.bam out put by cellranger
ref_file	A path of a reference sequence file.
output_file	The output path. An output of the alignment will be stored at the path.
python_path	The path to Python interpreter.
match	The score for a correct basepair matching.
mismatch	The penalty score for a basepair mismatching.
gapopen	The gap opening score for the alignment.
gapextension	The gap extension score for the alignment.
penalize_end_gaps	If penalizing the end gap when performing alignment (0 or 1).
ncores	The number of cores for the parallel processing

Value

It returns a data frame of the alignment result if 'return_df' is 'T' and 'NULL' otherwise.

---

sequence_permutation *Function for finding threshold of sequence alignment. The function randomly permutate certain percentage reference sequence and perform global alignment with the original reference sequence. By use the permutated sequence alignment score, users can filter the TraceQC alignment result.*

Description

Function for finding threshold of sequence alignment. The function randomly permutate certain percentage reference sequence and perform global alignment with the original reference sequence. By use the permutated sequence alignment score, users can filter the TraceQC alignment result.

Usage

```
sequence_permutation(
  ref_file,
  python_path = "python3",
  match = 2,
  mismatch = -2,
  gapopen = -6,
  gapextension = -0.1,
  penalize_end_gaps = 1,
  read_length = 0,
  permutate_percent = seq(0, 1, length.out = 101),
  n = 2,
  output_file = "alignment_threshold.txt"
)
```

Arguments

ref_file	A path of a reference sequence file.
python_path	The path to Python interpreter.
match	The score for a correct basepair matching.
mismatch	The penalty score for a basepair mismatching.
gapopen	The gap opening score for the alignment.
gapextension	The gap extension score for the alignment.
penalize_end_gaps	If penalizing the end gap when performing alignment (0 or 1).
read_length	The read length of each sequence. Use 0 if each read convers entire reference sequence.
permutate_percent	The percentage of sequence to permutate.
n	number of random permutation used for each percentage
output_file	The output path. An output dataframe will be stored at the path.

Value

It returns a data frame of the alignment result

---

seq_split *Split a string by a fixed length and joined with
 HTML tag.*

Description

Split a string by a fixed length and joined with
 HTML tag.

Usage

```
seq_split(s, len = 50)
```

Arguments

s	The input string
len	The fixed length

Value

A string splited by the 'len' then joined by '
'

`seq_to_character` *Identifying mutation events.*

Description

Identifying mutation events.

Usage

```
seq_to_character(
  aligned_reads,
  use_CPM,
  alignment_score_cutoff = 0,
  abundance_cutoff = 0
)
```

Arguments

`use_CPM`	Use count per million
`alignment_score_cutoff`	Minimum cutoff for alignment score
`abundance_cutoff`	Minimum cutoff for read count. This parameter are used with use_CPM.
`traceQC_input`	A TraceQC object.

Value

A data frame that contains the columns:

- 'type': The type of mutation.
- 'start': The starting position of mutation event.
- 'length': The length of mutation.
- 'mutation_to': A string that shows what mutation is occurred.
- 'count': The total number of the mutation events from the sample.

Index