

Package ‘Uniquorn’

April 25, 2025

Title Identification of cancer cell lines based on their weighted
mutational/ variational fingerprint

Version 2.29.0

Description 'Uniquorn' enables users to identify cancer cell lines.
Cancer cell line misidentification and cross-
contamination represents a significant challenge for cancer researchers.
The identification is vital and in the frame of this package based on the locations/ loci of so-
matic and germline mutations/ variations.
The input format is vcf/ vcf.gz and the files have to contain a single cancer cell line sample
(i.e. a single member/genotype/gt column in the vcf file).

Imports stringr, R.utils, WriteXLS, stats, doParallel, foreach,
GenomicRanges, IRanges, VariantAnnotation, data.table

Depends R (>= 3.5)

License Artistic-2.0

Type Package

Maintainer 'Raik Otto' <raik.otto@hu-berlin.de>

Date 2022-06-28

Author Raik Otto

RoxygenNote 7.2.1

NeedsCompilation no

Suggests testthat, knitr, rmarkdown, BiocGenerics

biocViews ImmunoOncology, StatisticalMethod, WholeGenome, ExomeSeq

VignetteBuilder knitr

Encoding UTF-8

git_url <https://git.bioconductor.org/packages/Uniquorn>

git_branch devel

git_last_commit 480d043

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-04-24

Contents

add_custom_vcf_to_database	2
add_missing_cls	3
add_penalty_statistics	4
add_p_q_values_statistics	4
create_bed_file	5
identify_vcf_file	6
initiate_canonical_databases	7
init_and_load_identification	8
match_query_ccl_to_database	9
parse_ccle_genotype_data	9
parse_cosmic_genotype_data	10
parse_vcf_file	10
parse_vcf_query_into_db	11
read_library_names	12
read_mutation_grange_objects	12
remove_ccls_from_database	13
remove_library_from_database	14
show_contained_ccls	14
show_contained_variants_for_ccl	15
show_contained_variants_in_library	16
show_which_ccls_contain_variant	16
Index	18

add_custom_vcf_to_database

add_custom_vcf_to_database This function adds the variants of
parsed custom CCLs to a monet DB instance

Description

add_custom_vcf_to_database This function adds the variants of parsed custom CCLs to a monet DB instance

Usage

```
add_custom_vcf_to_database(  
  vcf_input_files,  
  ref_gen = "GRCH37",  
  library_name = "CUSTOM",  
  n_threads = 1,  
  test_mode = FALSE  
)
```

Arguments

vcf_input_files

a character vector containing the input vcf files. This may be one or many vcf files.

ref_gen	a character string specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
library_name	a character string giving the name of the library to add the cancer cell lines to. Default is "CUSTOM". Library name will be automatically added as a suffix to the identifier.
n_threads	an integer specifying the number of threads to be used.
test_mode	Is this a test? Just for internal use

Value

Message wheather the adding was successful

Examples

```
HT29_vcf_file = system.file("extdata/HT29_TEST.vcf", package = "Uniquorn");
add_custom_vcf_to_database(
  vcf_input_files = HT29_vcf_file,
  library_name = "CELLMINER",
  ref_gen = "GRCH37",
  n_threads = 1,
  test_mode = TRUE
)
```

add_missing_cls	<i>add_missing_cls</i>
-----------------	------------------------

Description

add_missing_cls

Usage

```
add_missing_cls(res_table, dif_cls)
```

Arguments

res_table	Table that contains the identification results
dif_cls	Missing CLs

Value

Results table with added missing cls

`add_penalty_statistics`*add_penalty_statistics*

Description

Add penalty statistics to results

Usage

```
add_penalty_statistics(match_t, minimum_matching_mutations)
```

Arguments

`match_t` object that contains the matching variants

`minimum_matching_mutations`
a numerical giving the minimum amount of mutations that has to match between query and training sample for a positive prediction

Value

The updated statistics

`add_p_q_values_statistics`*add_p_q_values_statistics*

Description

A hypergeometric distribution-assumption allows to calculate the p-values for a significant or non-significant overlap in this function

Usage

```
add_p_q_values_statistics(  
  g_query,  
  match_t,  
  p_value,  
  ref_gen,  
  minimum_matching_mutations,  
  top_hits_per_library  
)
```

Arguments

<code>g_query</code>	IRanges object that contains the query variants
<code>match_t</code>	A table that contains the nubmber of matching variants
<code>p_value</code>	Threshold for the significance p-value
<code>ref_gen</code>	Reference genome version
<code>minimum_matching_mutations</code>	Manual lower amount of matching mutations require for a significant match between a query and a reference
<code>top_hits_per_library</code>	limits significant similarities to the first n hits

Details

`add_p_q_values_statistics` Calculates the p-values

Value

R table with a statistic

<code>create_bed_file</code>	<i>create_bed_file</i>
------------------------------	------------------------

Description

Creates BED files from the found and not found annotated mutations

Usage

```
create_bed_file(
  match_t,
  vcf_fingerprint,
  output_file,
  ref_gen,
  manual_identifier

)
```

Arguments

<code>match_t</code>	R table which contains the mutations from the training database for the cancer cell lines
<code>vcf_fingerprint</code>	contains the mutations that are present in the query cancer cell line's vcf file
<code>output_file</code>	Path to output file
<code>ref_gen</code>	Reference genome version
<code>manual_identifier</code>	Manually enter a vector of CL name(s) whose bed files should be created, independently from them passing the detection threshold

Value

Returns a message which indicates if the BED file creation has succeeded

identify_vcf_file	<i>identify_VCF_file</i>
-------------------	--------------------------

Description

Identifies a cancer cell lines contained in a vcf file based on the pattern (start & length) of all contained mutations/ variations.

Usage

```
identify_vcf_file(
    vcf_file,
    output_file,
    ref_gen,
    minimum_matching_mutations,
    mutational_weight_inclusion_threshold,
    write_xls,
    output_bed_file,
    top_hits_per_library,
    manual_identifier,
    verbose,
    p_value,
    confidence_score,
    n_threads,
    write_results
)
```

Arguments

vcf_file	Input vcf file. Only one sample column allowed.
output_file	Path of the output file. If blank, autogenerated as name of input file plus ' _uniquorn_ident.tab' suffix.
ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
minimum_matching_mutations	The minimum amount of mutations that has to match between query and training sample for a positive prediction
mutational_weight_inclusion_threshold	Include only mutations with a weight of at least x. Range: 0.0 to 1.0. 1= unique to CL. ~0 = found in many CL samples.
write_xls	Create identification results additionally as xls file for easier reading
output_bed_file	If BED files for IGV visualization should be created for the Cancer Cell lines that pass the threshold

top_hits_per_library	Limit the number of significant similarities per library to n (default 3) many hits. Is particularly used in contexts when heterogeneous query and reference CCLs are being compared.
manual_identifier	Manually enter a vector of CL name(s) whose bed files should be created, independently from them passing the detection threshold
verbose	Print additional information
p_value	Required p-value for identification. Note that if you set the confidence score, the confidence score overrides the p-value
confidence_score	Cutoff for positive prediction between 0 and 100. Calculated by transforming the p-value by $-1 * \log(p\text{-value})$ Note that if you set the confidence score, the confidence score overrides the p-value
n_threads	Number of threads to be used
write_results	Write identification results to file

Details

identify_vcf_file parses the vcf file and predicts the identity of the sample

Value

R table with a statistic of the identification result

Examples

```
HT29_vcf_file = system.file("extdata/HT29.vcf", package = "Uniquorn");

identification = identify_vcf_file(
  vcf_file = HT29_vcf_file,
  verbose = FALSE,
  write_results = FALSE
)
```

```
initiate_canonical_databases
      initiate_canonical_databases
```

Description

Parses data into r list variable

Usage

```
initiate_canonical_databases(
  cosmic_file = "CosmicCLP_MutantExport.tsv.gz",
  ccle_file = "CCLE_mutations.csv",
  ccle_sample_file = "sample_info.csv",
  ref_gen = "GRCH38"
)
```

Arguments

<code>cosmic_file</code>	The path to the Cosmic CLP file. The Cosmic file can be obtained from " https://cancer.sanger.ac.uk/cell_lines/depmap " and should be labeled "CosmicCLP_MutantExport.tsv.gz". Ensure that the right reference genome is used
<code>ccle_file</code>	The path to the ccle DNA genotype data file. It should be labeled "CCLE_mutations.csv". Ensure that the right reference genome is used
<code>ccle_sample_file</code>	The path to the CCLE sample file. It should be labeled "sample_info.csv" containing both the DepMap ID and corresponding cell line name.
<code>ref_gen</code>	Reference genome version

Value

Returns message if parsing process has succeeded

Examples

```
initiate_canonical_databases(
    cosmic_file = "CosmicCLP_MutantExport.tsv.gz",
    ccle_file = "CCLE_mutations.csv",
    ccle_sample_file = "sample_info.csv",
    ref_gen = "GRCH38"
)
```

```
init_and_load_identification
    init_and_load_identification
```

Description

Initiate the analysis Output basic information

Usage

```
init_and_load_identification(
    verbose,
    ref_gen,
    vcf_file,
    output_dir
)
```

Arguments

<code>verbose</code>	Print additional information
<code>ref_gen</code>	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
<code>vcf_file</code>	Path to vcf_file
<code>output_dir</code>	Output directory for identification results

Details

init_and_load_identification parses vcf file and output basic information

Value

Three file path instances and the fingerprint

```
match_query_ccl_to_database
      match_query_ccl_to_database
```

Description

Matches query ccl to the database

Usage

```
match_query_ccl_to_database(
  g_query,
  ref_gen = "GRCH37",
  library_name,
  mutational_weight_inclusion_threshold
)
```

Arguments

<code>g_query</code>	IRanges object that contains the variants
<code>ref_gen</code>	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
<code>library_name</code>	a character string giving the name of the library
<code>mutational_weight_inclusion_threshold</code>	a numerical giving the lower bound for mutational weight to be included

Value

The R Table `sim_list` which contains the CoSMIC CLP fingerprints

```
parse_ccle_genotype_data
      parse_ccle_genotype_data
```

Description

Parses ccle genotype data

Usage

```
parse_ccle_genotype_data(ccle_file, ccle_sample_file, ref_gen = "GRCH38")
```

Arguments

ccle_file	Path to CCLE file on hard disk
ccle_sample_file	Path to CCLE sample file
ref_gen	Reference genome version

Value

The R Table sim_list which contains the CCLE fingerprints

parse_cosmic_genotype_data	<i>parse_cosmic_genotype_data</i>
----------------------------	-----------------------------------

Description

Parses cosmic genotype data

Usage

```
parse_cosmic_genotype_data(cosmic_file, ref_gen = "GRCH38")
```

Arguments

cosmic_file	Path to cosmic clp file in hard disk
ref_gen	Reference genome version

Value

The R Table sim_list which contains the CoSMIC CLP fingerprints

parse_vcf_file	<i>Filter Parsed VCF Files</i>
----------------	--------------------------------

Description

Intern utility function. Filters the parsed VCF file for all informations except for the start and length of variations/mutations.

Usage

```
parse_vcf_file(
  vcf_file,
  ref_gen,
  library_name
)
```

Arguments

vcf_file	character string giving the path to the vcf file on the operating system.
ref_gen	Reference genome version
library_name	Name of the reference library

Value

Loci-based DNA-mutational fingerprint of the cancer cell line as found in the input VCF file.

parse_vcf_query_into_db

parse_vcf_query_into_db This function adds the variants of parsed custom CCLs to a monet DB instance

Description

parse_vcf_query_into_db This function adds the variants of parsed custom CCLs to a monet DB instance

Usage

```
parse_vcf_query_into_db(
  g_query,
  ref_gen = "GRCH37",
  library_name,
  test_mode = FALSE
)
```

Arguments

g_query	a GenomicRanges object
ref_gen	a character string specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
library_name	a character string giving the name of the library to add the cancer cell lines to. Default is "CUSTOM". Library name will be automatically added as a suffix to the identifier.
test_mode	Is this a test? Just for internal use

Value

Message wheather the adding was successful

read_library_names	<i>Library Name Reader</i>
--------------------	----------------------------

Description

This function provides information on the reference library names

Usage

```
read_library_names(ref_gen)
```

Arguments

ref_gen	a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
---------	--

Value

Returns a character vector of the contained libraries

Examples

```
read_library_names(ref_gen = "GRCH37")
```

read_mutation_grange_objects	<i>read_mutation_grange_objects</i>
------------------------------	-------------------------------------

Description

Read the GRange object for a specific library

Usage

```
read_mutation_grange_objects(
  library_name,
  mutational_weight_inclusion_threshold,
  ref_gen,
  test_mode
)
```

Arguments

library_name	a character string giving the name of the library
mutational_weight_inclusion_threshold	a numerical giving the lower bound for mutational weight to be included
ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
test_mode	Is this a test? Just for internal use

Value

The R Table `sim_list` which contains the CoSMIC CLP fingerprints

`remove_ccls_from_database`*Remove Cancer Cell Line*

Description

This function removes a cancer cell line training fingerprint (VCF file) from the database. The names of all training sets can be seen by using the function `show_contained_cls`.

Usage

```
remove_ccls_from_database(ccl_names, ref_gen = "GRCH37",  
  library_name, test_mode = FALSE)
```

Arguments

<code>ccl_names</code>	A character vector giving the names of the cancer cell line identifiers to be removed. Can be one or many
<code>ref_gen</code>	A character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
<code>library_name</code>	Name of the library from which ccls are to be removed
<code>test_mode</code>	Signifies if this is a test run

Value

Message that indicates whether the removal was succesful.

Examples

```
remove_ccls_from_database(  
  ccl_names = "HT29",  
  ref_gen = "GRCH37",  
  library_name = "CELLMINER",  
  test_mode = TRUE  
)
```

```
remove_library_from_database
```

Remove entire Library from Database

Description

This function removes a entire library from the database by removing all associated cancer cell line fingerprints from the database.

Usage

```
remove_library_from_database(library, ref_gen = "GRCH37", test_mode = FALSE)
```

Arguments

library	a character vector giving the names of the library to be removed.
ref_gen	a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
test_mode	is this a test? Just for internal use.

Value

Message that indicates whether the removal was succesful.

Examples

```
remove_library_from_database(library = "CELLMINER",
                             ref_gen = "GRCH37",
                             test_mode = TRUE)
```

```
show_contained_ccls      show_contained_ccls
```

Description

This function displays the names, amount of mutations and the overall weight of the mutations of all contained cancer cell line fingerprints for a chosen reference genome and optional library.

Usage

```
show_contained_ccls(ref_gen, verbose)
```

Arguments

ref_gen	a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
verbose	Should DB informations be printed?

Value

R table which contains identifiers of all cancer cell line samples which match the specified parameters (reference genome and library).

Examples

```
## Show all contained cancer cell lines for reference GRCH37:
show_contained_ccls(ref_gen = "GRCH37", verbose = TRUE)
```

```
show_contained_variants_for_ccl
      Variants In Cancer Cell Line
```

Description

This function shows all mutations present in the database for a selected cancer cell line and reference genome.

Usage

```
show_contained_variants_for_ccl(
  name_ccl,
  ref_gen,
  library_name,
  mutational_weight_inclusion_threshold
)
```

Arguments

name_ccl	a character vector giving the identifier of the cancer cell line for which mutations will be shown.
ref_gen	a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
library_name	Name of the reference library
mutational_weight_inclusion_threshold	Include only mutations with a weight of at least x. Range: 0.0 to 1.0. 1= unique to CL. ~0 = found in many CCL samples.

Value

GenomicRanges object that contains the ccl's variants

Examples

```
## Show all mutations for Cancer Cell Line 'SK_OV_3'
show_contained_variants_for_ccl(
  name_ccl = "SK_OV_3",
  ref_gen = "GRCH37",
  library_name = "CELLMINER",
  mutational_weight_inclusion_threshold = 0
)
```

```
show_contained_variants_in_library
```

All variants contained in reference library

Description

This function shows all variants contained in a reference library for a given inclusion weight. Default inclusion weight is 0 (all variants).

Usage

```
show_contained_variants_in_library(  
  ref_gen,  
  library_name,  
  mutational_weight_inclusion_threshold  
)
```

Arguments

`ref_gen` a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".

`library_name` Name of the reference library.

`mutational_weight_inclusion_threshold` Include only mutations with a weight of at least x. Range: 0.0 to 1.0. 1 = unique to CL. ~0 = found in many CL samples.

Value

Returns a GenomicRanges object that contains the variants

Examples

```
## Show all variants contained in reference library CELLMINER  
show_contained_variants_in_library(  
  ref_gen = "GRCH37",  
  library_name = "CELLMINER",  
  mutational_weight_inclusion_threshold = 0  
)
```

```
show_which_ccls_contain_variant
```

Cancer cell lines with specific variant

Description

This function displays all cancer cell lines in the database which contain a specified variant. Utilizes closed interval coordinates.

Usage

```
show_which_ccls_contain_variant(  
  start,  
  end,  
  chromosome,  
  ref_gen,  
  library_name,  
  mutational_weight_inclusion_threshold  
)
```

Arguments

start	Start coordinate
end	Stop coordinate
chromosome	Chromosome, 'chr' prefixes are ignored
ref_gen	a character vector specifying the reference genome version. All training sets are associated with a reference genome version. Default is "GRCH37".
library_name	Name of the reference library
mutational_weight_inclusion_threshold	Include only mutations with a weight of at least x. Range: 0.0 to 1.0. 1= unique to CL. ~0 = found in many CCL samples.

Value

Returns a GenomicRanges object that contains the variant if present. Member ccls can be found in the \$Member_ccl vector

Examples

```
show_which_ccls_contain_variant(  
  start = 92030762,  
  end = 92030762,  
  chromosome = 8,  
  ref_gen = "GRCH37",  
  library_name = "CELLMINER",  
  mutational_weight_inclusion_threshold = 0  
)
```

Index

* internal

- [parse_ccle_genotype_data](#), 9
- [parse_cosmic_genotype_data](#), 10

- [add_custom_vcf_to_database](#), 2
- [add_missing_cls](#), 3
- [add_p_q_values_statistics](#), 4
- [add_penalty_statistics](#), 4

- [create_bed_file](#), 5

- [identify_vcf_file](#), 6
- [init_and_load_identification](#), 8
- [initiate_canonical_databases](#), 7

- [match_query_ccl_to_database](#), 9

- [parse_ccle_genotype_data](#), 9
- [parse_cosmic_genotype_data](#), 10
- [parse_vcf_file](#), 10
- [parse_vcf_query_into_db](#), 11

- [read_library_names](#), 12
- [read_mutation_grange_objects](#), 12
- [remove_ccls_from_database](#), 13
- [remove_library_from_database](#), 14

- [show_contained_ccls](#), 14
- [show_contained_variants_for_ccl](#), 15
- [show_contained_variants_in_library](#), 16
- [show_which_ccls_contain_variant](#), 16