

# Package ‘Uniquorn’

April 7, 2026

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

**Version** 2.31.0

**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 somatic 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

**Date** 2025-10-22

**Author** Raik Otto

**RoxygenNote** 7.2.3

**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** baa8db3

**git\_last\_commit\_date** 2025-10-29

**Repository** Bioconductor 3.23

**Date/Publication** 2026-04-06

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

## 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_ccl_genotype_data . . . . .	10
parse_cosmic_genotype_data . . . . .	11
parse_vcf_file . . . . .	11
parse_vcf_query_into_db . . . . .	12
read_library_names . . . . .	12
read_mutation_grange_objects . . . . .	13
remove_ccls_from_database . . . . .	14
remove_library_from_database . . . . .	14
show_contained_ccls . . . . .	15
show_contained_variants_for_ccl . . . . .	16
show_contained_variants_in_library . . . . .	17
show_which_ccls_contain_variant . . . . .	18

## Index

19

---

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**

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

**Details**

add\_p\_q\_values\_statistics Calculates the p-values

**Value**

R table with a statistic

---

create_bed_file	<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**

match_t	R table which contains the mutations from the training database for the cancer cell lines
vcf_fingerprint	contains the mutations that are present in the query cancer cell line's vcf file
output_file	Path to output file
ref_gen	Reference genome version
manual_identifier	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 $-1 * \log(\text{p-value})$ 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**

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

verbose	Print additional information
ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
vcf_file	Path to vcf_file
output_dir	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**

<code>ccle_file</code>	Path to CCLE file on hard disk
<code>ccle_sample_file</code>	Path to CCLE sample file
<code>ref_gen</code>	Reference genome version

**Value**

The R Table `sim_list` which contains the CCLE fingerprints

---

```
parse_cosmic_genotype_data
      parse_cosmic_genotype_data
```

---

**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      Filter Parsed VCF Files
```

---

**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     *Library Name Reader*

---

### Description

This function procides 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
      read_mutation_grange_objects
```

---

**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_ccls`.

**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 successful.

**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 an 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, 10](#)
  - [parse\\_cosmic\\_genotype\\_data, 11](#)
- [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, 10](#)
- [parse\\_cosmic\\_genotype\\_data, 11](#)
- [parse\\_vcf\\_file, 11](#)
- [parse\\_vcf\\_query\\_into\\_db, 12](#)
- 
- [read\\_library\\_names, 12](#)
- [read\\_mutation\\_grange\\_objects, 13](#)
- [remove\\_ccls\\_from\\_database, 14](#)
- [remove\\_library\\_from\\_database, 14](#)
- 
- [show\\_contained\\_ccls, 15](#)
- [show\\_contained\\_variants\\_for\\_ccl, 16](#)
- [show\\_contained\\_variants\\_in\\_library, 17](#)
- [show\\_which\\_ccls\\_contain\\_variant, 18](#)