Package 'Uniquorn'

May 12, 2017

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

Version 1.4.1

Description Identifies cancer cell lines with their small variant fingerprint.
Cancer cell line misidentification and cross-contamination reprents a significant challenge for cancer researchers.
The identification is vital and in the frame of this package based on the locations or loci of somatic and germline mutations or variations.
The input format is vcf and the files have to contain a single cancer cell line sample.
The implemented method is optimized for the Next-generation whole exome and whole genome DNA-sequencing technology. RNA-seq data is very likely to work as well but hasn't been rigiously tested yet.
Panel-seq will require manual adjustment of thresholds.

Imports DBI, stringr, RSQLite, R.utils, WriteXLS, stats, BiocParallel

Depends R (>= 3.4)

License Artistic-2.0

LazyData TRUE

Type Package

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

Date 2017-05-12

Author Raik Otto

RoxygenNote 6.0.1

NeedsCompilation no

Suggests testthat, knitr, rmarkdown, BiocGenerics, RUnit

biocViews Software, StatisticalMethod, WholeGenome, ExomeSeq

VignetteBuilder knitr

R topics documented:

`add_custom_vcf_to_database` *Adds a custom vcf file to the three existing cancer cell line panels*

Description

Adds a custom vcf file to the three existing cancer cell line panels

Usage

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

Arguments

`vcf_input_files`	Input vcf file.s This may be one or many vcf files
`ref_gen`	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
`library`	The name of the library to add the CCLs to. Standard is '_CUSTOM' will automatically be added as suffix.
`test_mode`	Is this a test? Just for internal use
`n_threads`	Specifies number of threads to be used

Value

Message if the adding has succeeded

Examples

```
HT29_vcf_file = system.file("extdata/HT29.vcf.gz", package="Uniquorn");
add_custom_vcf_to_database(
vcf_input_files = HT29_vcf_file,
library = "",
ref_gen = "GRCH37",
test_mode = TRUE,
n_threads = 1)
```

`add_missing_cls` *add_missing_cls*

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

`calculate_p_and_q_values` *calculate_p_and_q_values*

Description

calculate_p_and_q_values

Usage

```
calculate_p_and_q_values(candidate_hits_abs_all, cl_absolute_mutation_hits,
  sim_list, sim_list_stats, minimum_matching_mutations, list_of_cls, p_value,
  q_value, vcf_fingerprint, panels)
```

Arguments

candidate_hits_abs_all	Maximally possible found variants
cl_absolute_mutation_hits	Matching variants
sim_list	Contains reference mutation data
sim_list_stats	Contains global reference mutation stats
minimum_matching_mutations	Minimal amount of required matching mutations
list_of_cls	List of CLs
p_value	Required maximal p-value
q_value	Required maximal q-value
vcf_fingerprint	The start and end positions of variants in the query
panels	The reference libraries

Value

Results table

---

calculate_similarity_results *calculate_similarity_results*

Description

calculate_similarity_results

Usage

```
calculate_similarity_results(sim_list, sim_list_stats, found_mut_mapping,
  minimum_matching_mutations, p_value, q_value, confidence_score,
  vcf_fingerprint, panels, list_of_cls)
```

Arguments

sim_list	Contains reference mutation data
sim_list_stats	Contains global reference mutation stats
found_mut_mapping	Mapping to mutations from query to reference mutation set
minimum_matching_mutations	Minimal amount of required matching mutations
p_value	Required maximal p-value
q_value	Required maximal q-value
confidence_score	Threshold above which a positive prediction occurs default 3.0
vcf_fingerprint	The start and end positions of variants in the query
panels	The reference libraries
list_of_cls	List of cancer cell lines

Value

Results table

`create_bed_file` *create_bed_file*

Description

Creates BED files from the found and not found annotated mutations

Usage

```
create_bed_file(

sim_list,

vcf_fingerprint,

res_table,

output_file,

ref_gen,

manual_identifier

)
```

Arguments

`sim_list`	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
`res_table`	Table containing the identification results
`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

`filter_for_weights` *filter_for_weights*

Description

Filter the reference set

Usage

```
filter_for_weights(
mutational_weight_inclusion_threshold,
ref_gen,
verbose,
sim_list,
sim_list_stats)
```

Arguments

`mutational_weight_inclusion_threshold`	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
`verbose`	Print additional information
`sim_list`	Contains the mutations
`sim_list_stats`	Contains the overal mutation statistics

Details

`filter_for_weights` parses vcf file and output basic information

Value

Filtered reference sets

`identify_vcf_file` *identify_VCF_file*

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 = "GRCH37",
minimum_matching_mutations = 0,
mutational_weight_inclusion_threshold = 0.5,
only_first_candidate = FALSE,
write_xls = FALSE,
output_bed_file = FALSE,
manual_identifier_bed_file = "",
verbose = FALSE,
p_value = .05,
q_value = .05,
confidence_score = 10.0,
n_threads = 1)
```

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.
only_first_candidate	Only the CL identifier with highest score is predicted to be present in the sample
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
manual_identifier_bed_file	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
q_value	Required q-value for identification
confidence_score	Threshold above which a positive prediction occurs default 10.0
n_threads	Number of threads to be used

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.gz", package="Uniquorn");

identification = identify_vcf_file( HT29_vcf_file )
```

`initiate_canonical_databases`
initiate_canonical_databases

Description

Parses data into r list variable

Usage

```
initiate_canonical_databases(
cosmic_file = "CosmicCLP_MutantExport.tsv",
ccle_file = "CCLE_hybrid_capture1650_hg19_NoCommonSNPs_CDS_2012.05.07.maf",
ref_gen = "GRCH37")
```

Arguments

`cosmic_file`	The path to the cosmic DNA genotype data file. Ensure that the right reference genome is used
`ccle_file`	The path to the ccle DNA genotype data file. Ensure that the right reference genome is used
`ref_gen`	Reference genome version

Value

Returns message if parsing process has succeeded

Examples

```
initiate_canonical_databases(
cosmic_file = "CosmicCLP_MutantExport.tsv",
ccle_file = "CCLE_hybrid_capture1650_hg19_NoCommonSNPs_CDS_2012.05.07.maf",
ref_gen = "GRCH37")
```

initiate_db_and_load_data *initiate_db_and_load_data*

Description

Intern utility function, loads database and return the sim_list and sim_list_stats variables.

Usage

```
initiate_db_and_load_data(
ref_gen,
request_table,
load_default_db )
```

Arguments

ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
request_table	Names of the tables to be extracted from the database
load_default_db	Indicate whether the default db should be used as source for the data

Value

Returns the sim_list and sim_list_stats variable

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_file,
n_threads)
```

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_file	Path to output report file
n_threads	Specifies number of threads to be used

Details

`init_and_load_identification` parses vcf file and output basic information

Value

Three file path instances and the fingerprint

`parse_ccle_genotype_data`
parse_ccle_genotype_data

Description

Parses ccle genotype data

Usage

```
parse_ccle_genotype_data(ccle_file, sim_list)
```

Arguments

ccle_file	Path to CCLE file on hard disk
sim_list	Variable containing mutations and cell line

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, sim_list)
```

Arguments

cosmic_file	Path to cosmic clp file in hard disk
sim_list	Variable containing mutations & cell line

Value

The R Table sim_list which contains the CoSMIC CLP fingerprints

parse_vcf_file *parse_vcf_file*

Description

Parses the vcf file and filters all information except for the start and length of variations/ mutations.

Usage

```
parse_vcf_file( vcf_file_path, n_threads)
```

Arguments

vcf_file_path	Path to the vcf file on the operating system
n_threads	Specifies number of threads to be used

Value

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

remove_custom_vcf_from_database

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.

Description

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_custom_vcf_from_database(
name_cl,
ref_gen = "GRCH37",
test_mode = FALSE)
```

Arguments

name_cl	name of the cancer cell line training fingerprintt
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

Message that indicates if the removal was succesful

Examples

```
remove_custom_vcf_from_database(
name_cl = "HT29_CELLMINER",
ref_gen = "GRCH37",
test_mode = TRUE )
```

re_calculate_cl_weights

Re-calculate sim_list_weights

Description

This function re-calculates the weights of mutation after a change of the training set

Usage

```
re_calculate_cl_weights(sim_list, ref_gen)
```

Arguments

sim_list	R Table which contains a mapping from mutations/ variations to their containing CLs
ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37

Value

A list containing both the sim_list at pos 1 and sim_list_stats at pos 2 data frames.

---

show_contained_cls *show_contained_cls*

Description

Show all cancer cell line identifier present in the database for a selected reference genome: This function shows the names, amount of mutations/ variations, overall weight of the mutations of all contained training CLs for a chosen reference genome.

Usage

```
show_contained_cls(
ref_gen)
```

Arguments

ref_gen	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37

Value

R table which contains the identifier of all cancer cell line samples with the specific reference genome and the weight of all mutations

Examples

```
contained_cls = show_contained_cls(
ref_gen = "GRCH37")
```

show_contained_mutations
show_contained_mutations

Description

Show all mutations present in the database for a selected reference Genome: This function shows all training-set mutations for a selected reference genome, i.e. the mutations that are being used for identification of query cancer cell lines.

Usage

```
show_contained_mutations(
ref_gen )
```

Arguments

ref_gen	Reference genome version

Value

R Table which contains all mutations associated with a particular cancer cell line for a specified reference genome

Examples

```
contained_cls = show_contained_mutations( ref_gen = "GRCH37" )
```

show_contained_mutations_for_cl
show_contained_mutations_for_cl

Description

Show all mutations present in the database for a selected cancer cell line and reference Genome

Usage

```
show_contained_mutations_for_cl(
name_cl,
ref_gen)
```

Arguments

name_cl	Name of the cancer cell line sample stored in the database
ref_gen	Reference genome version

Value

R table which contains all mutations associated with the defined cancer cell line and reference genome

Examples

```
SK_OV_3_CELLMINER_mutations = show_contained_mutations_for_cl(
name_cl = "SK_OV_3_CELLMINER_mutations",
ref_gen = "GRCH37")
```

---

show_which_cls_contain_mutation *show_which_cls_contain_mutation*

Description

Show all cancer cell lines in the database which contained the specified mutation and reference Genome. Closed interval coordinates. Format mutation: CHR_START_STOP, e.g. 1_123_123

Usage

```
show_which_cls_contain_mutation(
mutation_name,
ref_gen)
```

Arguments

mutation_name	Name of the mutation in the format CHROMOSOME_START_STOP, e.g. '11_244501_244510'
ref_gen	Reference genome version

Value

R table which contains all cancer cell line samples which contain the specified mutation with respect to the specified reference genome version

Examples

```
Cls_containing_mutations = show_which_cls_contain_mutation(
mutation_name = "10_103354427_103354427",
ref_gen = "GRCH37")
```

---

split_add *split_add*

Description

split_add

Usage

```
split_add(vcf_matrix_row)
```

Arguments

vcf_matrix_row	row of the vcf file

Value

Transformed entry of vcf file, reduced to start and length

`split_add_parallel` *split_add_parallel*

Description

split_add_parallel

Usage

```
split_add_parallel(para_index, vcf_matrix_row, vcf_handle, MARGIN, n_threads)
```

Arguments

`para_index`	row of the vcf file
`vcf_matrix_row`	A row of the parsed vcf file
`vcf_handle`	Handle to the parsed VCF file
`MARGIN`	Margin of the parse operation
`n_threads`	Specifies number of threads to be used

Value

Transformed entry of vcf file, reduced to start and length

`write_data_to_db` *write_data_to_db*

Description

Intern utility function, writes to database the sim_list and sim_list_stats variables

Usage

```
write_data_to_db(
content_table,
table_name,
ref_gen,
overwrite,
test_mode )
```

Arguments

`content_table`	Tables to be written in db
`table_name`	Name of the table to be written into the DB
`ref_gen`	Reference genome version. All training sets are associated with a reference genome version. Default: GRCH37
`overwrite`	Overwrite the potentially existing table
`test_mode`	Is this a test? Just for internal use

Value

the sim_list and sim_list_stats variable

Index