Package ‘lrRNAseqBenchmark’

February 20, 2022

Type Package

Title Figure out how to call varaints from Iso-Seq data

Version 0.1.0

Author c(person(``Vladimir'', ``Souza'', role=c(``aut'', ``cre''), email=``vladimir.souza@mls.uzh.ch''),
person(``Mark'', ``Robinson'', role = ``aut'', email = ``mark.robinson@imls.uzh.ch''),
person(``Gloria'', ``Sheynkman'', role = ``aut'', email = ``gs9yr@virginia.edu''),
person(``Elizabeth'', ``Tseng'', role = ``aut'', email = ``etseng@pacificbiosciences.com''))
person(``Ben'', ``Jordan'', role = ``aut'', email = ``bj8th.uva@gmail.com''))

Maintainer Vladimir Souza <vladimir.souza@mls.uzh.ch>

Description Use ground-truth VCF files genereted from short-read data to compare and validate VCF files generated from different variant callers on Iso-Seq data.

Depends R (>= 4.0)

Imports dplyr,
rlang,
BiocParallel,
GenomicAlignments,
Rsamtools,
IRanges,
S4Vectors,
snakecase,
ggplot2,
tibble,
tidyr,
vcfR,
GenomicRanges,
stringr,
Biostrings,
XVector

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

URL https://github.com/vladimirsouza/lrRNAseqBenchmark

BugReports https://github.com/vladimirsouza/lrRNAseqBenchmark/issues

R topics documented:

`add_a_method_indel_information_to_master_table`
Add column that states whether the variant called by a method is an indel or not

---

Description

Add column that states whether the variant called by a method is an indel or not

Usage

```
add_a_method_indel_information_to_master_table(
  input_table,
  vcf_file,
  method_name,
  truth_name
)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`vcf_file` 1-length string. The address of the method VCF file.

`method_name` 1-length string. The name of the method.

`truth_name` 1-length string. The name of the ground-truth.

Value

A data.frame

`add_format_tag_from_vcf` *Add a FORMAT tag from a VCF file to a master table*

Description

The function adds to ‘input_table‘ the new column <tagName>_<methodName>, where <tagName> is ‘tag‘ in lower case and <methodName> is ‘method_name‘.

Usage

```
add_format_tag_from_vcf(input_table, method_name, tag, vcf_file)
```

Arguments

`input_table` A data.frame. The input master table.

`method_name` A 1-length string. The name of the method.

`tag` A 1-length string. The name of the tag to add. It must be exactly how the tag is written in the VCF file (case sensitive).

`vcf_file` A 1-length string. The path to the VCF file from which the tag is extracted.

Value

A data.frame.

add_homopolymer_length_when_indels *Add homopolymer lengths to a master table*

Description

This function adds to the master table a column with the lengths of homopolymers, from the reference fasta, that overlaps positions POS + 1, where POS is the position of a variant. POS + 1 makes sence because minimp2 places homopolymer indels to the left of homopolymers, and , in case of indels, the column POS of VCF files means the position immediately to the left of the indel. In this way, homopolymer lengths of SNPs are meaningless. Moreover, homopolymer lengths of variants that are heterozygous alternatives should be meaningless as well. That is because they could contain alleles that are SNPs.

Usage

```
add_homopolymer_length_when_indels(
  input_table,
  homopolymers,
  ouput_what = "length"
)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`homopolymers` A CompressedIRangesList object. It should store all homopolymers, it's nucleotive types and lengths, of the genome used as the reference to call the variants. It is gerated by the function 'sarlacc::homopolymers'.

`ouput_what` A string equal to "length" (default) or "nts". If "length", the lengths of homopolymers are ouput (1 for non-homopolymers). If "nts", the nucleotide type is output (NA for non-homopolymers).

Value

A data.frame

add_info_tag_from_vcf *Add a INFO tag from a VCF file to a master table*

Description

If a variant in the VCF file doesn'r contain the specified tag, the function return -1 for that variant.

Usage

```
add_info_tag_from_vcf(input_table, col_name = NULL, tag, vcf_file)
```

Arguments

`input_table` A data.frame. The input master table.

`col_name` A 1-length string. the name of the column to add. If NULL (default), ‘col_name‘ is equal to ‘tag‘.

`tag` A 1-length string. The name of the tag to add. It must be exactly how the tag is written in the VCF file (case sensitive).

`vcf_file` A 1-length string. The path to the VCF file from which the tag is extracted.

Value

A data.frame.

`add_method_vs_truth_comparison_to_master_table`

Add classiffication of variants calls by comparing to the ground-truth to a master table

---

Description

Compare variants calls of a method to the ground-truth and classify each one as true-positiove (TP), false-negative (FN), false-positive (FP), or true-negative (TN).

Usage

```
add_method_vs_truth_comparison_to_master_table(
  input_table,
  method_name,
  truth_name,
  replace_column = FALSE
)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`method_name` A 1-length string. The name of the method to compare to the ground-truth.

`truth_name` A 1-length string. The name of the ground-truth.

`replace_column` 1-length boolean (default is FALSE). If the classification column already exists and it is desired to replace it.

Value

A data.frame.

`add_number_of_n_cigar_reads_to_master_table` *Add the count of N-cigar reads to the master table*

Description

This function adds a new column to a master table with the counts of N-cigar reads that overlap each site.

Usage

```
add_number_of_n_cigar_reads_to_master_table(
  input_table,
  input_bam,
  dataset_name = NULL
)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`input_bam` A ‘GenomicAlignments‘ object from which N-cigar-read counts are got.

`dataset_name` A 1-length string used to set the name of the new column.

Value

A data.frame.

`add_qual_from_vcf` *Add QUAL from a VCF into a master table*

Description

Add QUAL to master table.

Usage

```
add_qual_from_vcf(input_table, method_name, vcf_file)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`method_name` A 1-lenght string. The name of the method from which is desirable to get the QUAL values. The new column is named as "qual_<method_name>".

`vcf_file` A 1-lenght string. The path of the VCF file from which the the QUAL values are extracted.

Value

A data.frame

`add_read_coverage_from_bam_to_master_table`

Add read coverage (taken from BAM file) of each dataset to a master table

Description

Add read coverage (taken from BAM file) of each dataset to a master table

Usage

```
add_read_coverage_from_bam_to_master_table(
  input_table,
  dataset_coverage,
  dataset_name
)
```

Arguments

`input_table` A master table to add read-coverage columns.

`dataset_coverage` Object generated by `IRanges::coverage` function.

`dataset_name` A 1-length string. Name of the dataset used to genere ‘dataset_coverage‘.

Value

A data frame.

`add_some_read_statistic_from_bam_metacolumn`

Add some statistic of a tag about reads of a BAM file

Description

Add some statistic of a tag about reads of a BAM file

Usage

```
add_some_read_statistic_from_bam_metacolumn(
  input_table,
  col_name,
  galn,
  tag,
  stat
)
```

Arguments

`input_table`	A data.frame. The input master table.
`col_name`	A 1-length string. The name of the new column to add.
`galn`	A GAlignment object. The alingnment from where the tag is extracted.
`tag`	A 1-length string. The name of tag that is wanted to calculate the statistic ‘stat‘. The name must be exactly the same of one column name from ‘mcols(galn)‘.
`stat`	A function used to calculate the desired statistic.

Value

A data.frame.

`add_splice_site_info_to_master_table`

Add columns about spice sites to a master table

Description

Add columns about spice sites to a master table

Usage

```
add_splice_site_info_to_master_table(
  input_table,
  splice_sites,
  max_dist_from_splice_site = 20,
  multiply_max_dist = 100
)
```

Arguments

`input_table`	A data.frame generated by `initiate_master_table`.
`splice_sites`	A data.frame generated by `get_splice_sites_info`.
`max_dist_from_splice_site`	A 1-length integer (default is 20) that indicates the the maximum distance of a varaint from any splice site to consider it a variant near a splice site.
`multiply_max_dist`	A 1-length integer (default is 100). To create the column ‘ss_shortest_dist‘ from column ‘ss_dist‘, NA values (which means variables far from any splice site) are converted to integers. To make these integers be a high value, they are set to `max_dist_from_splice_site * multiply_max_dist`.

Value

A data.frame.

`add_the_ground_truth_indel_information_to_master_table` *Add column that states whether the variant in the ground-truth is an indel or not*

Description

Add column that states whether the variant in the ground-truth is an indel or not

Usage

```
add_the_ground_truth_indel_information_to_master_table(
  input_table,
  vcf_file,
  truth_name
)
```

Arguments

`input_table`	A data.frame. The master table to add the new column.
`vcf_file`	1-length string. The address of the ground-truth VCF file.
`truth_name`	1-length string. The name of the ground-truth.

Value

A data.frame

`add_two_method_comparison_to_master_table` *Add column to compare two methods in a master table*

Description

This function adds a column in a master table that compares two given methods. The new column informs whether the variant was called only by the first method (‘method1_name‘), only by the second method (‘method2_name‘), by both methods (‘"both"‘), or neither of them (‘"neither"‘).

Usage

```
add_two_method_comparison_to_master_table(
  input_table,
  method1_name,
  method2_name
)
```

Arguments

`input_table` A data.frame. The master table to add the new column.

`method1_name` A 1-length string. The name of the first method.

`method2_name` A 1-length string. The name of the second method.

Value

A data.frame.

`add_variant_density_of_a_method` *Add column of density of variants around each variant called by a method*

Description

Add column of density of variants around each variant called by a method

Usage

```
add_variant_density_of_a_method(input_table, window_size, used_methods)
```

Arguments

`input_table` A data.frame. The input master table.

`window_size` A 1-length integer. The size of the window where variants are used to calculate the density of variants around a each variant called by the specified method.

`used_methods` A vector of strings. The names of the methods to be used.

Value

A data.frame.

add_variant_type_to_master_table

Add a column that states the variant type according a method or the ground truth

Description

The new column stores the values "snp", "insertion", or "deletion", respectively to the variant type. If the variant type can not be defined – because the variant is heterozygous alternative and the alleles are a mix between SNP, insertion, or deletion – the value returned is "mix".

Usage

```
add_variant_type_to_master_table(input_table, vcf_file, method_name)
```

Arguments

`input_table`	A data.frame. The master table to add the new column.
`vcf_file`	1-length string. The address of the VCF file.
`method_name`	1-length string. The name of the ground-truth or method.

Value

A data.frame

calculate_precision_recall_for_multi_master_tables

Calculate accuracy measures for multiple master tables

Description

Create a data.frame with accuracy measures for multiple master tables.

Usage

```
calculate_precision_recall_for_multi_master_tables(
  ...,
  experiment_names,
  method_names,
  output_method_names = NULL,
  data_names,
  truth_names,
  coverage_thresholds = c(3, 15, 40, 100),
  what
)
```

Arguments

`...` Master tables.

`experiment_names` Vector of strings. Name of the experiments in the same order of the master tables.

`method_names` Vector or list of strings. Names of the methods to compare. If all experiments compare the same methods, ‘method_names‘ can be a single vector. Otherwise, ‘method_names‘ must be a list, in which each element specifies the names of the methods to compare for each experiment.

`output_method_names` NULL (default), or a vector or a list of strings. Names of the methods to output.

`data_names` Vector or list of strings.

`truth_names` Vector or list of strings.

`coverage_thresholds` Vector or list of integers.

`what` Vector or list of strings.

Value

A data.frame.

---

calculate_precision_recall_for_n_cigar_read_count_intervarls

For each N-cigar-read-count interval, calculate accuracy measures for multiple master tables

---

Description

Create a data.frame with accuracy measures for multiple master tables and different intervals for N-cigar-read counts.

Usage

```
calculate_precision_recall_for_n_cigar_read_count_intervarls(
  ...,
  experiment_names,
  method_names,
  output_method_names = NULL,
  data_names,
  truth_names,
  start_n_cigar_read_percent_intervals = c(0, 0.05, 0.9),
  end_n_cigar_read_percent_intervals = c(0.05, 0.9, 1),
  what
)
```

Arguments

`...` Master tables.

`experiment_names`
Vector of strings. Name of the experiments in the same order of the master tables.

`method_names` Vector or list of strings. Names of the methods to compare. If all experiments compare the same methods, ‘method_names‘ can be a single vector. Otherwise, ‘method_names‘ must be a list, in which each element specifies the names of the methods to compare for each experiment.

`output_method_names`
NULL (default), or a vector or a list of strings. Names of the methods to output.

`data_names` Vector or list of strings.

`truth_names` Vector or list of strings.

`start_n_cigar_read_percent_intervals`
vector of doubles. The start of each interval of N-cigar-read counts.

`end_n_cigar_read_percent_intervals`
vector of doubles. The end of each interval of N-cigar-read counts.

`what` Vector or list of strings.

Value

A data.frame.

`calc_accuracy_measures` *Calculate precision, sensitivity and F1-score of a method in a master table*

Description

Calculate precision, sensitivity and F1-score of a method in a master table

Usage

```
calc_accuracy_measures(input_table, method_name, truth_name)
```

Arguments

`input_table` A data.frame. The master table.

`method_name` A 1-length string. The name of the method to calculate the accuracy measures.

`truth_name` A 1-length string. The name of the ground-truth to validate the method.

Value

A named vector with the accuracy measures.

check_accuracy_per_coverage

Barplot of accuracy measures for different methods and coverage

Description

Return a ggplot object for visualization.

Usage

```
check_accuracy_per_coverage(
  master_table,
  method_names,
  output_method_names = NULL,
  data_name,
  truth_name,
  coverage_threshold
)
```

Arguments

`master_table` A data.frame. The input master table.

`method_names` A vector of strings. The names of the methods to be compared in the master table.

`output_method_names` A vector of strings. How to output the names of the methods to be compared. The defaut is NULL.

`data_name` A 1-length string. The name of the dataset used with the methods to be compared.

`truth_name` A 1-length string. The name of the ground-truth.

`coverage_threshold` A vector of integers. The minimum thresholds to filer by read coverage.

Value

A ggplot object.

get_splice_sites_info *Get all splice sites positions of a BAM files*

Description

Get the position of all splice sites in a BAM file and the number of reads that support each one of them. Besides that, indicate whether the splice site is acceptor or donor site.

Usage

```
get_splice_sites_info(input_bam, threads)
```

Arguments

input_bam	The input BAM file to extract splice site positions from.
threads	Number of threads.

Details

This function may take several hours to run.

Value

A data.frame in which each row is a splice-site position.

gt_vt_method *get the genotype and the variant type of the calls of a primary and secundary method*

Description

Create a new data.frame from a input master table. The rows correspond to the same variants in ‘input_table‘ and in the same order.

Usage

```
gt_vt_method(input_table, gt_first, gt_second, vt_first, vt_second)
```

Arguments

`input_table` A data.frame. The input master table.

`gt_first` A 1-length string. The name of the column that stores the genotypes called by a method. This method is called the first method.

`gt_second` A 1-length string. If the first method does not call some variants, extract their genotype from column ‘gt_second‘.

`vt_first` A 1-length string. The name of the column that stores the variant type called by a method. This method is called the first method.

`vt_second` A 1-length string. If the first method does not call some variants, extract their variant type from column ‘vt_second‘.

Value

A data.frame.

`igv_batch_screenshots` *Generate igv batch screenshots script*

Description

Generate igv batch screenshots script

Usage

```
igv_batch_screenshots(
  chrm,
  pos,
  output_dir,
  prefix,
  snapshot_path,
  windows_size = 1501,
  screenshot_number = 100,
  output_positions = FALSE
)
```

Arguments

`chrm` A vector of string. Chromossome name, e.g. "chrm1".

`pos` A vector of integers. Positions to center the screenshot.

`output_dir` A 1-length string. Directory where the IGV-batch-screenshot script is saved.

`prefix` A 1-length string.

`snapshot_path` A 1-length string. Directory where the IGV-batch-screenshot script will save the screenshots.

`windows_size` 1-length integer. Window size of the screenshot.

`screenshot_number`
: 1-length integer. Number of screenshot to generate.

`output_positions`
: If TRUE, output a data.frame with variant positions and the reagions to visualize.

initiate_master_table *Initiate master table*

Description

This function create the first columns of the master table. Those columns indicate whether a method could call of not a variant. A master table is used to get information form VCF files generated from different methods and compare to ground-truth VCF files.

Usage

```
initiate_master_table(..., method_names)
```

Arguments

`...`	VCF file addresses.
`method_names`	Vector of strings in which each element is the identification (name) of each input VCF files. The order of these elements must be in accordance with the order of the input files, and its length must be the same as the number of input VCF files.

Value

A data.frame.

is_indel_method *Add a column to state whether the variant is an indel or not*

Description

To classify the variant as an indel, first the function looks at the VCF file of the ‘first_method‘. If the variant is not there, if looks at the VCF file of the ‘second_method‘.

Usage

```
is_indel_method(input_table, first_method, second_method = NULL)
```

Arguments

`input_table`	A data.frame. The master table to add the new column.
`first_method`	A 1-length string. The name of the first method.
`second_method`	A 1-length string. The name of the second method

Details

Output meaning: * ‘-1‘ means that the variant type couldn’t be defined, because it is heterozygous alternative; * ‘0‘ means it is not an indel; * ‘1‘ means it is an indles; * ‘NA‘ means the ‘first_method‘ (and the ‘second_method‘) didn’t call the variant.

Value

A data.frame.

---

make_homopolymer_plot *Make plot to compare accuracy per homopolymer length*

Description

The function uses the output from function ‘method_homopolymer_indels‘ to make a plot to compare how the accuracy vary according to the homopolymer length or not within homopolymers. Those accuracy measures can be either rates of TPs, FNs and FPs, or the precision, recall and F1-score.

Usage

```
make_homopolymer_plot(
  input_hom_table,
  variant_type,
  method_name,
  truth_name,
  hom_length_intervals,
  interval_names,
  to_calculate
)
```

Arguments

`input_hom_table`	A data.frame. The output of the function ‘method_homopolymer_indels‘.
`variant_type`	A 1-length string. The variant type. Possivle values are: "snp", "deletion", or "insertion".
`method_name`	A 1-length string. The name of the method to analyse.
`truth_name`	A 1-length string. The name of the ground-truth.
`hom_length_intervals`	A vector of integers. Must be the same length of ‘interval_names‘. The inferior limit for each interval for homopolymer length.
`interval_names`	A vector of strings. Must be the same length of ‘hom_length_intervals‘. The name of each interval of homopolymer length.
`to_calculate`	A 1-length string. Possible values are: "rates" or "pre_rec_f1".

Value

A list, in which the first element, named ‘p‘, is a ggplot object, and the second element, named ‘interval_counts‘, is a named vector of integers that stores the counts of variants in each interval specified by the x-axis of the plot in ‘p‘.

---

make_homopolymer_table_to_plot

Organize the data used to make plots about homopolymer analysis

---

Description

The user may want to use this function several times to pull information about different combinations between ‘variant_type‘ and ‘method_name‘ from ‘input_hom_table‘. In the future, the function should make the job automatically using loops.

Usage

```
make_homopolymer_table_to_plot(
  input_hom_table,
  variant_type,
  method_name,
  truth_name,
  hom_length_intervals,
  interval_names,
  to_calculate,
  output_method_name
)
```

Arguments

`input_hom_table`
: A data.frame generated by function ‘method_homopolymer_indels‘.

`variant_type`
: A 1-length string. Possible values are "insertion" or "deletion".

`method_name`
: A 1-length string. The name of the method from which is desired to extract information.

`truth_name`
: A 1-length string. The name of the ground-truth.

`hom_length_intervals`
: A vector of integers. The minimum for each interval of homopolymer length. Each interval ‘i‘ ranges from ‘hom_length_intervals[i]‘ to ‘hom_length_intervals[i+1]‘, except the last interval which upper limit is ‘Inf‘.

`interval_names`
: A vector of characters with the same length of ‘hom_length_intervals‘. The name for each interval of homopolymer length.

`to_calculate`
: A 1-length string. Possible values are "rates" or "pre_rec_f1". If "rates", the functin calculates the rates of TPs, FNs and FPs. If "pre_rec_f1", it calculates the precision, the recall and the F1-score.

`output_method_name`
: A 1-length string. The label of the method specified in ‘method_name‘ to be output.

Value

A 2-length list (‘class_counts‘ and ‘dat_text‘).

`method_homopolymer_indels` *Get indels in homopolymers*

Description

Create a new data.frame that stores deletions of inside homopolymers, and insertions of a same nucleotide type that happen inside homopolymers of that same nucleotide type. Homopolymer length equal to 1 means non-homopolymers.

Usage

```
method_homopolymer_indels(
  input_table,
  first_method_name,
  second_method_name,
  vcf_first,
  vcf_second,
  method_dataset_name,
  homopolymers,
  ref_fasta_seqs,
  min_isoseq_coverage,
  genotyped_alt
)
```

Arguments

`input_table` A data.frame. The input master table.

`first_method_name`
: A 1-length string of the name of a method. First, the variant information that is taken is related to this method. If a variant is not called by the method, its information is taken relating to the method specifiec in ‘second_method_name‘.

`second_method_name`
: A 1-length string. The name of a method.

`vcf_first` A 1-length string. The path of the VCF file of the first method.

`vcf_second` A 1-length string. The path of the VCF file of the second method.

`method_dataset_name`
: A 1-length string. Name of the dataset used to call variants by the methods to be compared.

`homopolymers` A CompressedIRangesList object. It should store all homopolymers, it's nucleotive types and lengths, of the genome used as the reference to call the variants. It is gerated by the function 'sarlacc::homopolymerFinder'.

`ref_fasta_seqs` A DNAStringSet object. The sequences of the genome used as the reference to call the variants. It's names must be in the form like "chr1", "chr2", ..., "chrX", "chrY".

`min_isoseq_coverage` min iso-seq read coverage to filter.

`genotyped_alt` One of the strings "find" or "same". If "find", the function finds the correct alternative allele based on the genotype. This option must be chosen if there are multiple values for column ALT, which is the case of VCF files output by DeepVariant and Clair3. Since GATK's VCF files keep only the genotyped alternative allele in column ALT, this argument should be "same".

Value

A data.frame.

precision_recall_curve_per_coverage

Precision-recall curves to compare methods, using different filtering on read coverage

Description

Precision-recall curves to compare methods, using different filtering on read coverage

Usage

```
precision_recall_curve_per_coverage(
  master_table,
  method_names,
  output_method_names = NULL,
  data_name,
  truth_name,
  coverage_thresholds,
  what
)
```

Arguments

`master_table` A data.frame. The input master table.

`method_names` A vector of strings. The names of the methods to be compared.

`output_method_names` A vector of strings. How to output the names of the methods to be compared. The default is NULL.

`data_name` A 1-length string. The name of the dataset used with the methods to be compared.

`truth_name` A 1-length string. The name of the ground-truth.

`coverage_thresholds`
A vector of integers. The minimum thresholds to filer by read coverage. Each element defines a point in the precision-recall curves.

`what` A 1-length string. Possible values are: "snps_indels" (default), to make curves for SNPs and indels separately; "snps", to make curves only for SNPs; "indels", to make curves only for indels; "overall", to make curves without distinguishing SNPs and indels.

Value

A ggplot object.

---

splice_junction_analysis_table

Generate data for splice-junction analysis using multiple master tables

---

Description

Generate data to plot variant performance of sites near to and far from splice junctions comparisons using multiple master table as facets.

Usage

```
splice_junction_analysis_table(
  ...,
  experiment_names,
  truth_names,
  method_dataset_name,
  method_names,
  output_method_names,
  variant_type,
  min_isoseq_coverage
)
```

Arguments

`...` Data.frames. Each data.frame is a master table.

`experiment_names`
A vector of strings with length equal to the number of master tables input in '...'. Names of the datasets for each data table.

`truth_names` A vector of strings. Names of the ground truth in each data table especified in '...'.

`method_dataset_name`
A vector of strings. Name of the datasets used to call variants, by the methods to be compared, in each input master table.

`method_names` A vector of strings. The name of the methods to be compared defined in the input master tables. All master table must have the same ‘method_names‘.

`output_method_names`
A vector of strings. The output name of the methods to be compared defined in the input master tables. The output name of the methods will be the same for all master table.

`variant_type` A 1-length string. Possible values are "snp" or "indel".

`min_isoseq_coverage`
A 1-length integer. The threshod value for the minimum Iso-Seq read coverage.

Value

A list of two elements (‘acc_sj‘ and ‘n_test‘) to be used to draw a chart to analyse the variant-calling performance of sites near to and far from splice junctions. ‘acc_sj‘ stores the calculated performance measures, and ‘n_test‘ stores the number of observed variants in each case.

`standardize_genotype` *Standardize genotypes*

Description

This function rewrite genotype like: * 0|1 to 0/1; * 0/2, 0/3, ..., to 0/1; * 2/2, 3/3, ..., to 1/1; * 1/3, 2/3, ..., to 1/2

Usage

```
standardize_genotype(gt)
```

Arguments

`gt` A vector of strings. The input genotypes.

Value

A vector of strings.

Index