Package 'minSNPs'

April 19, 2023

Title Resolution-Optimised SNPs Searcher

Version 0.0.4

Description This is a R implementation of ``Minimum SNPs'' software as described in ``Price E.P., Inman-Bamber, J., Thiruvenkataswamy, V., Huygens, F and Giffard, P.M." (2007) <doi:10.1186/1471-2105-8-278> ``Computer-aided identification of polymorphism sets diagnostic for groups of bacterial and viral genetic variants."

Depends R (>= 3.4.0)

License MIT + file LICENSE

Imports BiocParallel, data.table

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, testthat, pkgdown, rmarkdown, withr

VignetteBuilder knitr

URL https://github.com/ludwigHoon/minSNPs

NeedsCompilation no

Author Ludwig Kian Soon Hoon [aut, cre] (<https://orcid.org/0000-0002-2310-3403>), Peter Shaw [aut, ctb] (<https://orcid.org/0000-0002-3187-8938>), Phil Giffard [aut, ctb] (<https://orcid.org/0000-0002-3030-9127>)

Maintainer Ludwig Kian Soon Hoon <ldwgkshoon@gmail.com>

Repository CRAN

Date/Publication 2023-04-19 07:00:02 UTC

R topics documented:

`calculate_percent` `calculate_percent`

Description

`calculate_percent` is used to calculate dissimilarity index, proportion of isolates not in goi that have been discriminated against. 1 being all and 0 being none.

Usage

```
calculate_percent(pattern, goi)
```

Arguments

`pattern`	list of sequences
`goi`	group of interest

Value

Will return the dissimilarity index of the list of patterns.

`calculate_simpson` `calculate_simpson`

Description

`calculate_simpson` is used to calculate Simpson's index. Which is in the range of 0-1, where the greater the value, the more diverse the population.

Usage

```
calculate_simpson(pattern)
```

Arguments

`pattern`	list of sequences

Value

Will return the Simpson's index of the list of patterns.

`cal_fn` `cal_fn`

Description

`cal_fn` is used to check if the proportion of false negative fastas and metas are compatible.

Usage

```
cal_fn(pattern, goi, target)
```

Arguments

`pattern`	the pattern from `generate_pattern`
`goi`	the group of interest (names of isolates)
`target`	the target sequence(s)

Value

proportion: no. false negative/number of isolates

cal_fp cal_fp

Description

cal_fp is used to check if the proportion of false positive fastas and metas are compatible.

Usage

```
cal_fp(pattern, goi, target)
```

Arguments

pattern	the pattern from generate_pattern
goi	the group of interest (names of isolates)
target	the target sequence(s)

Value

proportion: no. false positive/number of isolates

check_fasta_meta_mapping check_fasta_meta_mapping

Description

check_fasta_meta_mapping is used to check if fastas and metas are compatible.

Usage

```
check_fasta_meta_mapping(fasta, meta)
```

Arguments

fasta	the fasta read into memory to join
meta	the meta read into memory to join

Value

TRUE/FALSE if the fasta and meta are compatible

check_percent check_percent

Description

check_percent is used to check if parameters needed by calculate_percent are all present.

Usage

```
check_percent(list_of_parameters)
```

Arguments

list_of_parameters	is a list of parameter passed to functions that will perform the calculation

Value

TRUE if goi exists, else FALSE

extend_length extend_length

Description

extend_length extend the search sequence such that there will always be (prev) bases before the SNPs and (after) bases after the SNPs.

Usage

```
extend_length(
  overlaps,
  position_reference,
  genome_position,
  prev,
  after,
  ori_string_start,
  ori_string_end,
  ori_snp_pos,
  genome_max
)
```

Arguments

overlaps	Overlappings
position_reference	the mapping of position in SNP matrix to reference genome
genome_position	the position of the SNP in the reference genome
prev	number of bases before the SNP included in the search string
after	number of bases after the SNP included in the search string
ori_string_start	original starting point of search string
ori_string_end	original ending point of the search string
ori_snp_pos	original SNP position in search string
genome_max	length of the reference genome

Value

a list containing the new ‘string_start‘, ‘string_end‘, ‘snp_pos‘, ‘snps_in_string‘.

---

find_optimised_snps find_optimised_snps

Description

find_optimised_snps is used to find optimised SNPs set.

Usage

```
find_optimised_snps(
  seqc,
  metric = "simpson",
  goi = c(),
  accept_multiallelic = TRUE,
  number_of_result = 1,
  max_depth = 1,
  included_positions = c(),
  excluded_positions = c(),
  iterate_included = FALSE,
  bp = SerialParam(),
  ...
)
```

Arguments

seqc	list of sequences, either passed directly from process_allele or read_fasta or equivalence
metric	either 'simpson' or 'percent'
goi	group of interest, if creteria is percent, must be specified, ignored otherwise
accept_multiallelic	whether include positions with > 1 state in goi
number_of_result	number of results to return, 0 will be coerced to 1
max_depth	maximum depth to go before terminating, 0 means it will only calculate the metric for included position
included_positions	included positions
excluded_positions	excluded positions
iterate_included	whether to calculate index at each level of the included SNPs
bp	BiocParallel backend. Rule of thumbs: use MulticoreParam(workers = ncpus - 2)
...	other parameters as needed

Value

Will return the resolution-optimised SNPs set, based on the metric.

---

full_merge full_merge

Description

full_merge is used to merge 2 fasta, where a position exist only in 1 of the fasta, the fasta without allele in that positions are given reference genome's allele at that position. **Doesn't work for large dataset, hence the need for full_merge_1**

Usage

```
full_merge(
  fasta_1,
  fasta_2,
  meta_1,
  meta_2,
  ref,
  bp = BiocParallel::MulticoreParam(),
  ...
)
```

Arguments

fasta_1	fasta read into memory to join
fasta_2	fasta read into memory to join
meta_1	meta file for 'fasta_1' denoting all positions of SNPs and position in reference genome
meta_2	meta file for 'fasta_2' denoting all positions of SNPs and position in reference genome
ref	name of the reference genome (needs to be in both fasta files)
bp	the BiocParallel backend
...	all other arguments

Value

merged fasta and meta

---

full_merge_1 full_merge_1

Description

full_merge_1 is used to merge 2 fasta, where a position exist only in 1 of the fasta, the fasta without allele in that positions are given reference genome's allele at that position.

Usage

```
full_merge_1(
  fasta_1,
  fasta_2,
  meta_1,
  meta_2,
  ref,
  bp = BiocParallel::SerialParam(),
  ...
)
```

Arguments

fasta_1	fasta read into memory to join
fasta_2	fasta read into memory to join
meta_1	meta file for 'fasta_1' denoting all positions of SNPs and position in reference genome
meta_2	meta file for 'fasta_2' denoting all positions of SNPs and position in reference genome
ref	name of the reference genome (needs to be in both fasta files)
bp	the BiocParallel backend
...	all other arguments

Value

merged fasta and meta

`generate_kmers` `generate_kmers`

Description

`generate_kmers` generate the kmer sequences of the given length

Usage

```
generate_kmers(final_string, k)
```

Arguments

`final_string`	the string to generate kmers
`k`	the length of the kmer

Value

a vector of kmers

`generate_kmer_search_string` `generate_kmer_search_string`

Description

`generate_kmer_search_string` generate the search strings to detect genes' presence

Usage

```
generate_kmer_search_string(
  gene_seq,
  k,
  id_prefix = NULL,
  bp = MulticoreParam()
)
```

Arguments

`gene_seq`	sequences to generate k_mers from
`k`	kmer length
`id_prefix`	prefix for the gene id
`bp`	BiocParallel backend to use

Value

a dataframe containing the search strings

`generate_pattern` `generate_pattern`

Description

`generate_pattern` is used to generate pattern for calculation.

Usage

```
generate_pattern(seqc, ordered_index = c(), append_to = list())
```

Arguments

`seqc`	list of sequences
`ordered_index`	list of indexes for the pattern in the order
`append_to`	existing patterns to append to

Value

Will return concatenated list of string for searching.

`generate_snp_search_string` `generate_snp_search_string`

Description

`generate_snp_search_string` identify the SNPs that will overlap the search strings generated from the targeted SNPs

Usage

```
generate_snp_search_string(
  selected_snps,
  position_reference,
  ref_seq,
  snp_matrix,
  prev,
  after,
  position_type = "fasta",
  extend_length = TRUE,
  bp = MulticoreParam()
)
```

Arguments

selected_snps	list of targeted SNPs
position_reference	the mapping between reference genome positions and orthologous SNP matrix positions
ref_seq	the reference genome sequence
snp_matrix	the orthologous SNP matrix
prev	number of characters before the SNP
after	number of characters after the SNP
position_type	type of SNPs input, "fasta" (orthologous SNP matrix based) or "genome" (reference genome based); Default to "fasta"
extend_length	whether to extend the search string before and after the SNP and ignore overlapping SNPs
bp	BiocParallel backend to use

Value

a dataframe containing the search strings

---

get_metric_fun get_metric_fun

Description

get_metric_fun is used to get the metrics function and required parameters. Additional metric may set by assigning to 'MinSNPs_metrics' variable.

Usage

```
get_metric_fun(metric_name = "")
```

Arguments

metric_name	name of the metric, by default percent/simpson

Value

a list, including the function to calculate the metric based on a position ('calc'), and function to check for additional parameters the function need ('args')

identify_overlaps identify_overlaps

Description

identify_overlaps identify the overlapping SNPs in the sequences

Usage

```
identify_overlaps(position_reference, genome_position, prev, after)
```

Arguments

position_reference	the mapping of position in SNP matrix to reference genome
genome_position	the position of the SNP in the reference genome
prev	number of bases before the SNP included in the search string
after	number of bases after the SNP included in the search string

Value

a list containing 2 dataframes listing the bystander SNPs in the flanking sequence before and after the SNPs

iterate_merge iterate_merge

Description

iterate_merge is used to combine > 2 fastas iteratively.

Usage

```
iterate_merge(
  fastas,
  metas,
  ref,
  method = "full",
  bp = BiocParallel::SerialParam(),
  ...
)
```

Arguments

fastas	list of fastas read into memory to join
metas	list of metas read into memory to join
ref	name of the reference genome (needs to be in both fasta files)
method	how to join the 2 fasta, currently supported methods are: inner, full
bp	the BiocParallel backend
...	all other arguments

Value

Will return a list containing a merged FASTA and a meta.

---

match_count match_count

Description

match_count return the number of matches of the target string in the given sequence

Usage

```
match_count(target, search_from)
```

Arguments

target	the search target
search_from	the sequence to search from

Value

number of matches

merge_fasta merge_fasta

Description

merge_fasta is used to combine 2 fasta.

Usage

```
merge_fasta(
  fasta_1,
  fasta_2,
  meta_1,
  meta_2,
  ref,
  method = "full",
  bp = BiocParallel::SerialParam(),
  ...
)
```

Arguments

fasta_1	fasta read into memory to join
fasta_2	fasta read into memory to join
meta_1	meta file for 'fasta_1' denoting all positions of SNPs and position in reference genome
meta_2	meta file for 'fasta_2' denoting all positions of SNPs and position in reference genome
ref	name of the reference genome (needs to be in both fasta files)
method	how to join the 2 fasta, currently supported methods are: inner, full
bp	the BiocParallel backend
...	all other arguments

Value

Will return a list containing a merged FASTA and a meta.

output_result output_result

Description

output_result is used to present the result and save the result as tsv.

Usage

```
output_result(result, view = "", ...)
```

Arguments

result	is the result from find_optimised_snps
view	how to present the output, "csv" or "tsv" will be saved as a file. Otherwise, printed to console.
...	if view is "tsv" or "csv", file name can be passed, e.g., file_name = "result.tsv", otherwise, file is saved as <timestamp>.tsv.

Value

NULL, result either printed or saved as tsv.

output_to_files output_to_files

Description

output_to_files is write the result to files.

Usage

```
output_to_files(merged_result, filename = "merged")
```

Arguments

merged_result	a list containing the merged fasta and meta.
filename	filename to write to, will output <filename>.fasta and <filename>.csv.

Value

NULL, files written to filesystem

process_allele process_allele

Description

process_allele is used to returned the processed allelic profiles, by removing the allele profile with duplicate name and length different from most. 1st allele profile with the duplicated name is returned, the longer length is taken as normal should there be 2 modes.

Usage

```
process_allele(
  seqc,
  bp = BiocParallel::SerialParam(),
  check_length = TRUE,
  check_bases = TRUE,
  dash_ignore = TRUE,
  accepted_char = c("A", "C", "T", "G"),
  ignore_case = TRUE,
  remove_invariant = FALSE,
  biallelic_only = FALSE
)
```

Arguments

seqc	a list containing list of nucleotides. To keep it simple, use provided read_fasta to import the fasta file.
bp	is the biocparallel backend, default to serialParam, most likely sufficient in most scenario
check_length	Check the length of each sample in the matrix, default to TRUE
check_bases	Check the bases of each sample in the matrix, default to TRUE
dash_ignore	whether to treat '-' as another type
accepted_char	character to accept, default to c("A", "C", "T", "G")
ignore_case	whether to be case insensitive, default to TRUE
remove_invariant	whether to remove invariant positions, default to FALSE
biallelic_only	whether to remove positions with more than 2 alleles, default to FALSE

Value

Will return the processed allelic profiles.

read_fasta read_fasta

Description

read_fasta is used to read fasta file, implementation similar to seqinr, but much simpler and allow for spaces in sample name.

Usage

```
read_fasta(file, force_to_upper = TRUE, bp = SerialParam())
```

Arguments

file	file path
force_to_upper	whether to transform sequences to upper case, default to TRUE
bp	is the biocparallel backend, default to serialParam, most likely sufficient in most scenario

Value

Will return list of named character vectors.

read_sequences_from_fastq read_sequences_from_fastq

Description

read_sequences_from_fastq get the sequences from a fastq file, it completely ignores the quality scores

Usage

```
read_sequences_from_fastq(
  fastq_file,
  force_to_upper = TRUE,
  quality_offset = 33,
  bp = MulticoreParam()
)
```

Arguments

fastq_file	location of the fastq file
force_to_upper	whether to transform sequences to upper case, default to TRUE
quality_offset	the quality offset to use, default to 33
bp	BiocParallel backend to use for parallelization

Value

will return a list of sequences, with qualities as attribute

---

resolve_IUPAC_missing resolve_IUPAC_missing

Description

resolve_IUPAC_missing is used to replace the ambiguity codes found in the sequences.

Usage

```
resolve_IUPAC_missing(
  seqc,
  log_operation = TRUE,
  log_file = "replace.log",
  max_ambiguity = -1,
  replace_method = "random",
  N_is_any_base = FALSE,
  output_progress = TRUE,
  bp = MulticoreParam()
)
```

Arguments

seqc	the sequences to be processed
log_operation	whether to log the operation
log_file	log file to write the operations
max_ambiguity	proportion of ambiguity codes to tolerate, -1 = ignore. Default to -1
replace_method	how to substitute the ambiguity codes, current supported methods:random and most_common, default to "random".
N_is_any_base	whether to treat N as any base or substitute it with one of the alleles found at the position.
output_progress	whether to output progress
bp	the BiocParallel backend

Value

Will return the processed sequences.

`reverse_complement` `reverse_complement`

Description

`reverse_complement` returns the reverse complement of the given sequence

Usage

```
reverse_complement(seq)
```

Arguments

`seq`	the sequence to reverse complement

Value

reverse complemented sequence

`view_percent` `view_percent`

Description

`view_percent` is used to present the result of selected SNPs set based on Simpson's Index.

Usage

```
view_percent(result, ...)
```

Arguments

`result`	is the result from `find_optimised_snps`
`...`	other optional parameters

Value

formatted result list to be saved or presented.

view_simpson view_simpson

Description

view_simpson is used to present the result of selected SNPs set based on Simpson's Index.

Usage

```
view_simpson(result, ...)
```

Arguments

result	is the result from find_optimised_snps
...	other optional parameters

Value

formatted result list to be saved or presented.

write_fasta write_fasta

Description

write_fasta is used to write the named character vectors to fasta file.

Usage

```
write_fasta(seqc, filename)
```

Arguments

seqc	a list containing list of nucleotides. To keep it simple, use provided read_fasta to import the fasta file.
filename	filename of the output file

Value

will write the alignments to file

Index