Package ‘mutType’

December 12, 2022

Type Package

Title Determine the mutation type for a set of single nucleotide variants in a genome

Version 0.99.0

Date 2022-12-12

Description Starting from a set of single nucleotide variants in VCF format, the corresponding reference genome and a parameter 'context_length' specified by the user, the package function 'mutation_type' determines for each mutation the corresponding mutation type 'UP[REF>ALT]DOWN' such that all mutation types have C or T as mutated reference base. The overall length of the mutation type is determined by the context_length parameter. The package function 'count_table' summarizes the mutation types for the set of mutations into a count table that reports the number of mutations per mutation type. Eventually the package function 'graphical_summary' generates a pdf file showing the barplot visualization of all the mutation types with a frequency higher than a threshold specified by the user.

Depends R(>= 3.6.0)

Imports VariantAnnotation, ggplot2, Biostrings, grDevices, BSgenome, GenomicRanges, IRanges

Suggests knitr, rmarkdown, BiocStyle, testthat, pdftools, BSgenome.Hsapiens.UCSC.hg19, BiocManager

License GPL-2

Encoding UTF-8

VignetteBuilder knitr

biocViews SNP, VariantAnnotation, BiomedicalInformatics, Genetics, Sequencing, Software

RoxygenNote 7.2.3

NeedsCompilation no

Author Paola Maragno [aut, cre]

Maintainer Paola Maragno <paola.maragno@mail.polimi.it>

R topics documented:

mutType-package *mutType - Determine the mutation type for a set of single nucleotide variants in a genome*

Description

The mutType package - starting from a set of single nucleotide variants in VCF format, the corresponding reference genome and a parameter *context_length* specified by the user - determines for each mutation the corresponding mutation type ‘UP[REF>ALT]DOWN‘ such that all mutation types have C or T as mutated reference base.
The overall length of the mutation type is determined by the context_length parameter specified by the user.
The package also provides a function which summarizes the mutation types for the set of mutations into a count table.
Eventually the package generates a pdf file showing the barplot visualization of all the mutation types with a frequency higher than a threshold specified by the user.

Details

Package:	mutType
Type:	Package
Version:	0.99.0
Date:	2022-12-12
License:	GPL (>=2)

Author(s)

Paola Maragno [aut, cre]
Politecnico di Milano
Maintainer: Paola Maragno
E-Mail: <paola.maragno@mail.polimi.it>

build_mut_type **Build the final mutation type**

Description

`build_mut_type` function takes a GRanges object containing a set of single nucleotide variants retrieved from a VCF file, the corresponding reference genome, the parameter *context_length* specified by the user and the indication whether the reverse complement of the sequence containing the SNV must be computed.
This function returns a vector with the mutation types ‘UP[REF>ALT]DOWN‘ of all the variants in the GRanges object, such that all mutation types have C or T as mutated reference base.
The overall length of the mutation type is determined by the context_length parameter.
This function is not exported to be used by the user but it is defined only to be exploited in the mutation_type function.

Usage

```
build_mut_type(mutgr, reference_genome, context_length, rev)
```

Arguments

`mutgr` GRanges object containing a set of single nucleotide variants

`reference_genome`
Reference genome

`context_length` Parameter specified by the user to indicate the overall length of the mutation type - it is used to compute how many nucleotides upstream and downstream the SNV base include in the mutation type

`rev` Parameter to indicate whether the reverse complement of the sequence containing the SNV must be computed (default is FALSE)

Value

Vector with the mutation types ‘UP[REF>ALT]DOWN‘ of all the variants in the GRanges object with C or T as mutated reference base.

Author(s)

Paola Maragno
Politecnico di Milano
Maintainer: Paola Maragno
E-Mail: <paola.maragno@mail.polimi.it>

`count_table` Count table of mutation types

Description

`count_table` function summarizes the mutation types for the set of mutations into a count table reporting the number of mutations per mutation type.

Usage

```
count_table(mut_types)
```

Arguments

`mut_types` Vector of mutation types ‘UP[REF>ALT]DOWN‘ returned by mutation_type function

Value

Count table reporting the number of mutations per mutation type.

Author(s)

Paola Maragno
Politecnico di Milano
Maintainer: Paola Maragno
E-Mail: <paola.maragno@mail.polimi.it>

Examples

```
mut_types <- c("GT[C>A]CA", "GG[C>G]TC", "CC[T>C]TC", "GT[C>T]GT", "TA[C>A]CG", "GT[C>A]CA",
"CA[C>G]CT", "CC[T>C]CT", "CA[T>C]AT", "TT[C>G]TC", "CC[T>C]CT", "CT[T>C]TG", "CC[T>C]CT", "TA[C>A]CG")
count_table(mut_types)
```

`graphical_summary` Graphical visualization of the mutation types with a frequancy higher than a threshold

---

Description

`graphical_summary` function generates a pdf file showing the barplot visualization of all the mutation types with a frequency higher than a threshold specified by the user.

Usage

```
graphical_summary(c_table, freq, file_name)
```

Arguments

`c_table` Count table returned by count_table function that summarizes the mutation types contained in a VCF file

`freq` Threshold of frequency that a mutation type must have at least in order to be visualized in the barplot

`file_name` Name of the pdf file in which the function will plot its graphical output

Value

The function returns the name of the pdf file - that is named as chosen by the user and it is stored in the working directory - showing the barplot visualization of the frequencies of the mutation types with a frequency at least equal to *freq* threshold.

Author(s)

Paola Maragno
Politecnico di Milano
Maintainer: Paola Maragno
E-Mail: <paola.maragno@mail.polimi.it>

Examples

```
c_table <- data.frame(mut_types = c("AA[C>A]AA", "AA[C>A]AC", "AA[C>A]AG", "AA[C>A]AT", "AA[C>A]CA", "AA[C>A]C
 "AA[C>A]CT", "AA[C>A]GA", "AA[C>A]GC", "AA[C>A]GG", "AA[C>A]GT", "AA[C>A]TA", "AA[C>A]TC"),
 Freq = c(15,  13,  32,  44,  25,  5,  67,  21,  14,  42,  52,  21,  19))
graphical_summary(c_table, 30, "Mut_types_visualization_30.pdf")
```

`mutation_type` **Determine the mutation type of a set of single nucleotide variants in a genome**

Description

`mutation_type` function takes a set of single nucleotide variants in VCF format, the corresponding reference genome and a parameter *context_length* specified by the user - that must be odd - and determines for each mutation the corresponding mutation type ‘UP[REF>ALT]DOWN‘ such that all mutation types have C or T as mutated reference base.
The overall length of the mutation type is determined by the context_length parameter.

Usage

```
mutation_type(vcf_file, reference_genome, context_length)
```

Arguments

`vcf_file` VCF file containing a set of single nucleotide variants (like ‘chr22.vcf‘)

`reference_genome` Reference genome

`context_length` Parameter specified by the user to indicate the overall length of the mutation type - it is used to compute how many nucleotides upstream and downstream the SNV base include in the mutation type

Value

List of two elements: the mutation types ‘UP[REF>ALT]DOWN‘ vector with C or T as mutated reference base and the GRanges object containing first the SNVs with C or T as mutated reference base and then the SNVs with G or A as mutated reference base. In this way the position of each mutation type in the vector is the same of the corresponding SNV in the GRanges object returned by the function.

Author(s)

Paola Maragno
Politecnico di Milano
Maintainer: Paola Maragno
E-Mail: <paola.maragno@mail.polimi.it>

Examples

```
if (!'BSgenome.Hsapiens.UCSC.hg19' %in% installed.packages()) {
if (!requireNamespace("BiocManager", quietly = TRUE)) {
install.packages("BiocManager")
}
BiocManager::install('BSgenome.Hsapiens.UCSC.hg19')
}

library('BSgenome.Hsapiens.UCSC.hg19')
vcf_file <- system.file("extdata", "chr22.vcf.gz", package = "VariantAnnotation")
results <- mutation_type(vcf_file, BSgenome.Hsapiens.UCSC.hg19, 3)
mut_types <- results[[1]]
mutgr_reordered <- results[[2]]
mut_types[1:10]
mutgr_reordered[1:10]
```

Index