MutationType: A package to identify the mutations types of SNVs

Giuseppe Barranco *[1]

[1]Master's Degree student in BCG Polimi/Unimi 10868963

*giuseppe.barranco@studenti.unimi.it

2024-02-04

Contents

1 Introduction

This package, made principally of three functions, aims to find the Single-Nucleotide-Variants, the relative genomic sequence and eventually perform some general visual statistics.

1.1 Biological Background

A change of one base with respect to the reference genome is considered a single nulceotide mutation. Some of these variants however are present in different prcentages in the whole population indeed are also called SN-polymorphism. It can happen that a variant cam be rare and often is labeled as mutation.

1.2 This package

There are three main functions:

- seekSNVs(vcffile, referenceGenome) which takes a vcf file and a reference genome as input and find among all the variants only the single nucleotide. It then return a df which some infomrations (name, chr, ref allele, alternative)
- seekMutAndSeq(df, contextLenght=3, refGen) which takes in input a df, that should be the output of the first function, that has the position of the SNVs; A context length which define the total length of the sequences that has to retrieve on the left and right of the reference genome with respect to the position of the SNVs.
- generalStat compute the frequency of the two types of mutations, (to reduce the redundancy we consider only C,T as the reverse strand) and also the frequency for equal whole sequences.

2 Usage example

2.1 seekSNVs

You can either provide a path to file or a vcf file to this function. To read it it used the function *readVcf*. Using the *GenomicRanges::width* function the vcf file can be filtered. We want to select the variant where there is only one nucleotide for the reference and also for the alternative. This is basically how to distinguish snv, indels, duplication. Here a file provided from the *VariantAnnotation* package will be used.

```
vcffile <- system.file("extdata", "chr22.vcf.gz", package="VariantAnnotation")
vcffile <- readVcf(vcffile,"hg38")
setOfSNVs <- seekSNVs(vcffile, "hg38")
```

id	chr	pos	start	end	ref	alt	stra
rs7410291	chr22	50300078	0	0	A	G	*
rs147922003	chr22	50300086	0	0	C	T	*
rs114143073	chr22	50300101	0	0	G	A	*
rs141778433	chr22	50300113	0	0	C	T	*
rs182170314	chr22	50300166	0	0	C	T	*
rs115145310	chr22	50300187	0	0	G	A	*
rs186769856	chr22	50300268	0	0	T	C	*
rs77627744	chr22	50300346	0	0	G	A	*
rs193230365	chr22	50300423	0	0	G	A	*
rs9627788	chr22	50300438	0	0	T	C	*

2.2 seekMutAndSeq

Here, after you get the SNVs and the relative positions it is possible to run this function which find the genomic sequence with the given length input of the given reference genome file. It can happen that the context_length can produce somehow ranges that are over the actually length of the chromosome. Therefore in this function is implemented a check that ensure this. Also if different chromosome are provided. Here the ref genome is the "Full genomic sequences for Homo sapiens as provided by UCSC (genome hg38, based on assembly GRCh38.p14 since 2023/01/31). The sequences are stored in DNAString objects". [https://bioconductor.org/packages/release/data/annotation/html/BSgenome.Hsapiens.UCSC.hg38.html]

```
setOfSNVsInfo <- seekMutAndSeq(setOfSNVs, contextLength = 3, Hsapiens)
```

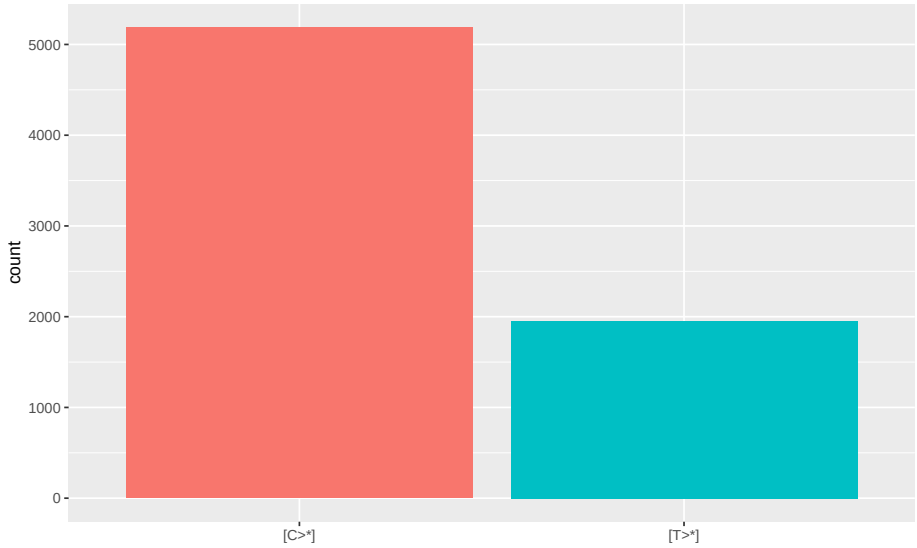
	id	chr	pos	start	end	ref	alt	stra	sequences	mutation_type
chr22.1	rs7410291	chr22	50300078	50300077	50300079	T	C	*	CGC	C[T>C]C
chr22.2	rs147922003	chr22	50300086	50300085	50300087	C	T	*	CGG	C[C>T]G
chr22.3	rs114143073	chr22	50300101	50300100	50300102	C	T	*	CCC	C[C>T]C
chr22.4	rs141778433	chr22	50300113	50300112	50300114	C	T	*	TCG	T[C>T]G
chr22.5	rs182170314	chr22	50300166	50300165	50300167	C	T	*	CAG	C[C>T]G
chr22.6	rs115145310	chr22	50300187	50300186	50300188	C	T	*	GCG	G[C>T]G
chr22.7	rs186769856	chr22	50300268	50300267	50300269	T	C	*	GGG	G[T>C]G
chr22.8	rs77627744	chr22	50300346	50300345	50300347	C	T	*	GGG	G[C>T]G
chr22.9	rs193230365	chr22	50300423	50300422	50300424	C	T	*	AGC	A[C>T]C
chr22.10	rs9627788	chr22	50300438	50300437	50300439	T	C	*	ACG	A[T>C]G

2.3 generalStart

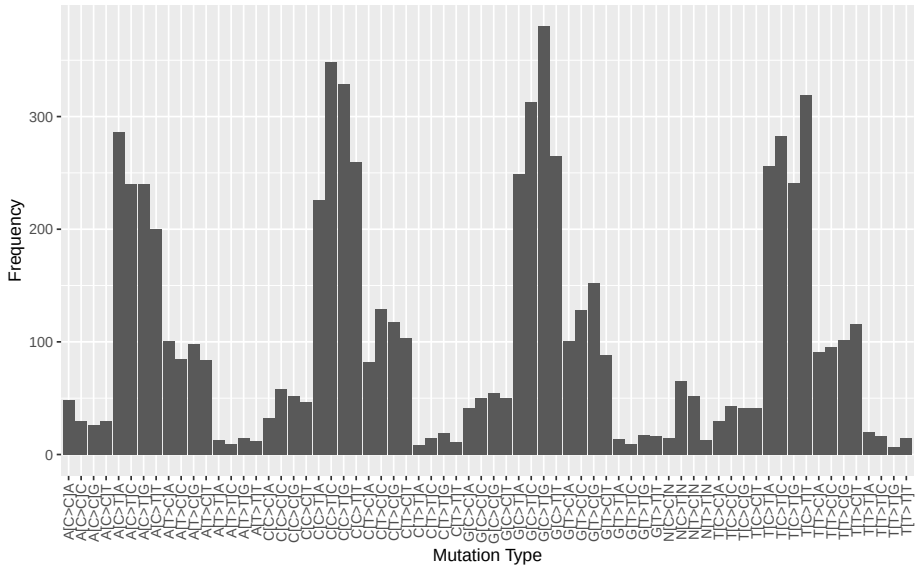
This function returns a list with two tables and two plots. There are frequency for the nucleotide mutations and also for the whole context sequence

```
results <- generalStat(setOfSNVsInfo)
```

refMutation	Freq
C	5189
T	1954



Type	Freq
A[C>C]A	48
A[C>C]C	30
A[C>C]G	26
A[C>C]T	30
A[C>T]A	286
A[C>T]C	240
A[C>T]G	240
A[C>T]T	200
A[T>C]A	101
A[T>C]C	85




```
## R version 4.3.2 (2023-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.2.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK versio
##
## locale:
## [1] C/UTF-8/C/C/C/C
##
## time zone: Europe/Rome
## tzcode source: internal
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] MutationType_1.0.0               BSgenome.Hsapiens.UCSC.hg38_1.4.5
##  [3] BSgenome_1.70.1                  rtracklayer_1.62.0
```

```
##  [5] BiocIO_1.12.0                     VariantAnnotation_1.48.1
##  [7] Rsamtools_2.18.0                  Biostrings_2.70.2
##  [9] XVector_0.42.0                    SummarizedExperiment_1.32.0
## [11] Biobase_2.62.0                    GenomicRanges_1.54.1
## [13] GenomeInfoDb_1.38.5               IRanges_2.36.0
## [15] S4Vectors_0.40.2                  MatrixGenerics_1.14.0
## [17] matrixStats_1.2.0                 BiocGenerics_0.48.1
## [19] ggplot2_3.4.4                     knitr_1.45
## [21] BiocStyle_2.30.0
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0         farver_2.1.1             dplyr_1.1.4
##  [4] blob_1.2.4               filelock_1.0.3           bitops_1.0-7
##  [7] fastmap_1.1.1            RCurl_1.98-1.14          BiocFileCache_2.10.1
## [10] GenomicAlignments_1.38.2 XML_3.99-0.16.1          digest_0.6.34
## [13] lifecycle_1.0.4          KEGGREST_1.42.0          RSQLite_2.3.5
## [16] magrittr_2.0.3           compiler_4.3.2           rlang_1.1.3
## [19] progress_1.2.3           tools_4.3.2              utf8_1.2.4
## [22] yaml_2.3.8               labeling_0.4.3           prettyunits_1.2.0
## [25] S4Arrays_1.2.0           bit_4.0.5                curl_5.2.0
## [28] DelayedArray_0.28.0      xml2_1.3.6               abind_1.4-5
## [31] BiocParallel_1.36.0      withr_3.0.0              grid_4.3.2
## [34] fansi_1.0.6              colorspace_2.1-0         scales_1.3.0
## [37] biomaRt_2.58.2           cli_3.6.2                rmarkdown_2.25
## [40] crayon_1.5.2             generics_0.1.3           rjson_0.2.21
## [43] httr_1.4.7               DBI_1.2.1                cachem_1.0.8
## [46] stringr_1.5.1            zlibbioc_1.48.0          parallel_4.3.2
## [49] AnnotationDbi_1.64.1     restfulr_0.0.15          BiocManager_1.30.22
## [52] vctrs_0.6.5              Matrix_1.6-5             bookdown_0.37
## [55] hms_1.1.3                bit64_4.0.5              GenomicFeatures_1.54.3
## [58] glue_1.7.0               codetools_0.2-19         stringi_1.8.3
## [61] gtable_0.3.4             munsell_0.5.0            tibble_3.2.1
## [64] pillar_1.9.0             rappdirs_0.3.3           htmltools_0.5.7
## [67] GenomeInfoDbData_1.2.11  R6_2.5.1                 dbplyr_2.4.0
## [70] evaluate_0.23            lattice_0.22-5           png_0.1-8
## [73] memoise_2.0.1            SparseArray_1.2.3        xfun_0.41
## [76] pkgconfig_2.0.3
```