Package 'FIREVAT'

February 26, 2019

Type Package

Title FIREVAT, FInding REliable Variants without ArTifacts

Description FIREVAT is a variant filtering tool for cancer sequencing data, which uses mutational signatures to identify sequencing artifacts and low-quality variants.

Version 0.1.5

Authors Andy Jinseok Lee, Hyunbin Kim

Maintainer Andy Jinseok Lee <jinseok.lee@ncc.re.kr>, Hyunbin Kim <khb7840@ncc.re.kr>

Imports data.table,
stringi,
bedr,
GA,
jsonlite,
yaml,
MutationalPatterns,
deconstructSigs,
lsa,
BSgenome.Hsapiens.UCSC.hg19,
BSgenome.Hsapiens.UCSC.hg38,
ggpubr,
caTools,
ggrepel,
gridExtra,
ggplot2,
rmarkdown,
gtable,
dplyr

URL https://github.com/cgab-ncc/FIREVAT

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests knitr

VignetteBuilder knitr

R topics documented:

`AnnotateVCFObj` *AnnotateVCFObj*

Description

Annotates a vcf.obj using df.variants.of.interest (from (`PrepareAnnotationDB`)

Usage

```
AnnotateVCFObj(vcf.obj, df.annotation.db, columns.to.include,
  include.all.columns = FALSE)
```

Arguments

`vcf.obj`	`ReadVCF`
`df.annotation.db`	A data.frame from `PrepareAnnotationDB`. This data.frame must have the columns 'CHROM', 'POS', 'REF', 'ALT'
`columns.to.include`	A character vector of columns to include. Note that existing columns in vcf.obj will not be affected.

include.all.columns
: A boolean value. If TRUE, then annotates vcf.obj with all columns present in df.variants.of.interest. If FALSE, columns.to.include must be supplied.

Value

An annotated vcf.obj

Chromosome.Names *Constant*

Description

Chromosome names for FIREVAT. Chromosome names should be given in the format of "chr" + chromosome number.

Usage

```
Chromosome.Names
```

Format

An object of class `character` of length 25.

ComputeZScore *ComputeZScore*

Description

Returns a z-score of x given a distribution of values

Usage

```
ComputeZScore(values, x)
```

Arguments

values	a numeric vector
x	a numeric value

Value

a numeric value corresponding to the z-score of x

`ComputeZScoreEquiValue` *ComputeZScoreEquiValue*

Description

Returns a numeric value that is equivalent to the specified z.score in the distribution of 'values'

Usage

```
ComputeZScoreEquiValue(z.score, values)
```

Arguments

`z.score`	numeric value
`values`	numeric vector

Value

a numeric value corresponding to the specified z.score in the 'values' distribution

`DecimalCeiling` *DecimalCeiling*

Description

Returns the ceiling of a decimal value e.g. value = 0.15, decimal = 0.1 returns 0.2

Usage

```
DecimalCeiling(value, decimal)
```

Arguments

`value`	numeric value (decimal)
`decimal`	numeric value (e.g. 0.1, 0.001)

Value

a numeric value

DefaultFilterToBinary *Transform default filtering parameters to a binary vector*

Description

This function transforms default filtering parameter to binary vector which can be used as a suggested solution in GA algorithm.

Usage

```
DefaultFilterToBinary(vcf.filter, params.bit.len)
```

Arguments

vcf.filter	A list generated in MakeFilter
params.bit.len	A list with bit lengths of filtering parameters which is generated from ParameterToBits

Value

A binary vector

EnumerateTriNucCounts *EnumerateTriNucCounts*

Description

Returns C>A, C>G, C>T, T>A, T>C, T>G counts

Usage

```
EnumerateTriNucCounts(spectrum)
```

Arguments

spectrum	a numeric vector with 96 numeric values

Details

Please note that this function assumes that 'spectrum' is sorted (i.e. 1:16 –> C>A; 17:32 –> C>G; 33:48 –> C>T; 49:64 –> T>A; 65:80 –> T>C; 81:96 –> T>G)

Value

a numeric vector of length 6 corresponding to the counts of each trinucleotide change (C>A, C>G, C>T, T>A, T>C, T>G)

FilterVCF *FilterVCF*

Description

Filter vcf based on the filter Filtering parameters are saved in config.obj Split vcf.obj into vcf.obj.filtered & vcf.obj.artifact based on vcf.filter

Usage

```
FilterVCF(vcf.obj, vcf.filter, config.obj, include.array = NULL,
  force.include = FALSE, verbose = TRUE)
```

Arguments

vcf.obj	A list from ReadVCF
vcf.filter	A list from MakeMuTect2Filter
config.obj	A list from ParseConfigFile
include.array	A boolean vector
force.include	A boolean value. If TRUE, then uses 'include.array'
verbose	If true, provides process detail

Value

A list with the following elements

- 1) Mutations which passed filteringvcf.obj.filtered = vcf.obj (list with data, header, genome)
- 2) Mutations which did not pass filteringvcf.obj.artifact = vcf.obj (list with data, header, genome)

GenerateConfigObj *Generate config.obj by checking vcf header*

Description

This function generate config.obj by checking vcf header. Users should fill in the information needed in console. In current version, only Integers & Float values can be used in config.obj for running FIREVAT.

Usage

```
GenerateConfigObj(vcf.obj, save.config = TRUE,
  config.path = "../temp/FIREVAT_configure.json")
```

Arguments

vcf.obj	A list from ReadVCF
save.config	If true, save config.obj to config.path
config.path	File path to write config.obj (json or yaml)

Value

config.obj

GetCOSMICMutSigs *GetCOSMICMutSigs*

Description

Returns a data.frame of the COSMIC mutational signature reference file from the data directory

Usage

```
GetCOSMICMutSigs()
```

Value

a data.frame of the COSMIC reference mutational signatures

GetCOSMICMutSigsEtiologiesColors *GetCOSMICMutSigsNames*

Description

Returns all COSMIC mutational signature etiologies and colors

Usage

```
GetCOSMICMutSigsEtiologiesColors()
```

Value

data.frame with following columns: signature, group and color.

GetCOSMICMutSigsNames *GetCOSMICMutSigsNames*

Description

Returns all COSMIC mutational signature names

Usage

```
GetCOSMICMutSigsNames()
```

Value

a character vector

GetOptimizedSignatures *GetOptimizedSignatures*

Description

This function fetches the last row from the optimization iteration log and returns the target and artifactual mutational signatures for the type of mutations ('refined' or 'artifactual')

Usage

```
GetOptimizedSignatures(data, mutations.type = "refined",
  signatures = "all")
```

Arguments

data	A list of main data from RunFIREVAT
mutations.type	A string for type of mutations ('refined' or 'artifact')
signatures	A string ('all', 'target', 'artifact')

Value

A data.frame with the columns 'signature' and 'weight'

GetPCAWGMutSigs *GetPCAWGMutSigs*

Description

Returns the PCAWG mutational signatures data

Usage

```
GetPCAWGMutSigs()
```

Value

a data.frame of the PCAWG mutatioanl signatures

GetPCAWGMutSigsEtiologiesColors *GetPCAWGMutSigsEtiologiesColors*

Description

Returns the PCAWG mutational signatures etiologies and colors

Usage

```
GetPCAWGMutSigsEtiologiesColors()
```

Value

a data.frame with the columns 'signature', 'group', 'color'

GetPCAWGMutSigsNames *GetPCAWGMutSigsNames*

Description

Returns the PCAWG mutational signatures names

Usage

```
GetPCAWGMutSigsNames()
```

Value

a character vector of the PCAWG mutational signatures names

InitializeVCF *InitializeVCF*

Description

Initialize VCF with FIREVAT config file This functions selects point mutations and appends filter values to vcf.obj$data

Usage

```
InitializeVCF(vcf.obj, config.obj, verbose = TRUE)
```

Arguments

vcf.obj	A list from ReadVCF
config.obj	A list from ParseConfigFile
verbose	If true, provides process detail

Value

A list with the following elements

- vcf.obj.filteredvcf.obj (high-quality vcf)
- vcf.obj.artifactvcf.obj (low-quality vcf)

MakeFilter *MakeFilter*

Description

Creates a vcf filter from config.obj

Usage

```
MakeFilter(config.obj)
```

Arguments

config.obj	A list from ParseConfigFile (any filter with "use_in_filter" value declared as FALSE is not considered)

Value

A list with the filter parameters

MutaliskParseVCFObj *MutaliskParseVCFObj*

Description

Parses a vcf.obj and prepares it to run Mutalisk.

Usage

```
MutaliskParseVCFObj(vcf.obj)
```

Arguments

vcf.obj	A list from ReadVCF

Value

A data.frame

MutPatParseRefMutSigs *MutPatParseRefMutSigs*

Description

Parses a df.ref.mut.sigs and prepares it to run Mutational Patterns.

Usage

```
MutPatParseRefMutSigs(df.ref.mut.sigs, target.mut.sigs,
  signature.start.column.index = 4,
  mutation.type.header = "SomaticMutationType")
```

Arguments

df.ref.mut.sigs	A data.frame of reference mutational signatures
target.mut.sigs	A character vector of target mutational signatures names
signature.start.column.index	= An integer value (e.g. column index corresponding to 'SBS1')
mutation.type.header	= A string value (name of header corresponding to column containing 'A[C>A]A' data))

Value

A data.frame of the format deconstructSigs::signatures.cosmic

`MutPatParseVCFObj` *MutPatParseVCFObj*

Description

Parses a vcf.obj and prepares it to run Mutational Patterns.

Usage

```
MutPatParseVCFObj(vcf.obj, bsg, sample.id = "sample")
```

Arguments

`vcf.obj`	A list from ReadVCF
`bsg`	BSgenome.Hsapiens.UCSC.hg19::BSgenome.Hsapiens.UCSC.hg19 or BSgenome.Hsapiens.UCSC.hg38
`sample.id`	A string value

Value

A data.frame with the column sample.id and row names corresponding to 96 substitution types

`ParameterToBits` *ParameterToBits*

Description

Calculate the number of bits needed to conduct FIREVAT GA optimization.

Usage

```
ParameterToBits(vcf.obj, config.obj, vcf.filter, multiplier = 100)
```

Arguments

`vcf.obj`	A list from ReadVCF
`config.obj`	A list from ParseConfigFile
`vcf.filter`	A list from MakeMuTect2Filter
`multiplier`	A multiplier for convert fraction to integer (default = 100)

Details

vcf.obj$data: if max(vcf.obj$data[[param]]) < 1, then multiply multiplier to the vector

Value

A list with the elements 'params.bit.len' containing the bit lengths of each parameter 'vcf.obj' with updated data

ParseConfigFile *ParseConfigFile*

Description

This function returns config.obj from JSON or YAML config file. - Check if the config file is in JSON format or YAML format - Return config.obj

Usage

```
ParseConfigFile(config.path, verbose = TRUE)
```

Arguments

config.path	A string for config file path
verbose	If true, provides process detail

Value

config.obj: list of parameters

Examples

```
## Not run:
ParseConfigFile("example.variant.caller.json")
ParseConfigFile("example.variant.caller.json", verbose=False)

## End(Not run)
```

PCAWG.All.Sequencing.Artifact.Signatures *Constant*

Description

PCAWG mutational signatures reported to be associated with sequencing artifacts

Usage

```
PCAWG.All.Sequencing.Artifact.Signatures
```

Format

An object of class character of length 17.

PCAWG.Known.Sequencing.Artifact.Signatures
Constant

Description

PCAWG mutational signatures reported to be associated with sequencing artifacts

Usage

```
PCAWG.Known.Sequencing.Artifact.Signatures
```

Format

An object of class `character` of length 1.

PCAWG.Likely.Sequencing.Artifact.Signatures
Constant

Description

PCAWG mutational signatures reported to be associated with sequencing artifacts

Usage

```
PCAWG.Likely.Sequencing.Artifact.Signatures
```

Format

An object of class `character` of length 5.

PCAWG.Possible.Sequencing.Artifact.Signatures
Constant

Description

PCAWG mutational signatures reported to be associated with sequencing artifacts

Usage

```
PCAWG.Possible.Sequencing.Artifact.Signatures
```

Format

An object of class `character` of length 11.

PCAWG.Target.Mutational.Signatures

Constant

Description

PCAWG target mutational signatures reported to be unrelated to sequencing artifacts

Usage

```
PCAWG.Target.Mutational.Signatures
```

Format

An object of class `character` of length 49.

PerformStrandBiasAnalysis

PerformStrandBiasAnalysis

Description

Performs strand bias analysis

Usage

```
PerformStrandBiasAnalysis(vcf.obj, ref.forward.strand.var,
  ref.reverse.strand.var, alt.forward.strand.var, alt.reverse.strand.var,
  perform.fdr.correction = TRUE, fdr.correction.method = "BH")
```

Arguments

`vcf.obj` ReadVCF

`ref.forward.strand.var`
: A string value.

`ref.reverse.strand.var`
: A string value.

`alt.forward.strand.var`
: A string value.

`alt.reverse.strand.var`
: A string value.

`perform.fdr.correction`
: If TRUE, then performs false discovery rate correction

`fdr.correction.method`
: A string value. FDR correction method (Refer to p.adjust() function)

Value

An updated vcf.obj

`PlotMutaliskResults` *PlotMutaliskResults*

Description

Plots Mutalisk results

Usage

```
PlotMutaliskResults(mutalisk.results, signatures, trinuc.max.y,
  trinuc.min.y, mut.type.max.y, title)
```

Arguments

`mutalisk.results`	A list obtained from `RunMutalisk`
`signatures`	A character vector of mutational signatures names
`trinuc.max.y`	A numeric value (maximum y-axis value)
`trinuc.min.y`	A numeric value (minimum y-axis value)
`mut.type.max.y`	A numeric value
`title`	A string value

Value

A ggplot object

Examples

```
## Not run:
  df.ref.mut.sigs <- GetPCAWGMutSigs()
  target.mut.sigs <- GetPCAWGMutSigsNames()
  vcf.obj <- ReadVCF(vcf.file = "../data/sample/P-233-CT.final.vcf")
  mutalisk.results <- RunMutalisk(vcf.obj = vcf.obj,
                                  df.ref.mut.sigs = df.ref.mut.sigs,
                                  target.mut.sigs = target.mut.sigs)
  p <- PlotMutaliskResults(mutalisk.results = mutalisk.results)
  print(p)

## End(Not run)
```

PlotMutationTypes *PlotMutationTypes*

Description

Plots a horizontal barplot of mutation types

Usage

```
PlotMutationTypes(mutation.types = c("C>A", "C>G", "C>T", "T>A", "T>C",
  "T>G"), mutation.types.values, mutation.types.colors, max.y.val, title,
  convert.to.percentage = T, show.legend = T, font.size.small = 8,
  font.size.med = 14, plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))
```

Arguments

`mutation.types`	Mutation types; Default = c("C>A", "C>G", "C>T", "T>A", "T>C", "T>G")
`mutation.types.values`	Mutation count for each mutation type
`mutation.types.colors`	A color vector for indicating mutation types
`max.y.val`	y axis maximum value
`title`	Plot title
`convert.to.percentage`	if True convert y values to percentage (x 100); Default = T
`show.legend`	If True, show legend; Default = T
`font.size.small`	Small font size; Default = 8
`font.size.med`	Medium font size; Default = 14
`plot.margin`	Margin vector for drawing plot; Default = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))

Value

A ggplot object

Examples

```
## Not run:
p <- PlotMutationTypes(mutation.types = c("C>A", "C>G", "C>T", "T>A", "T>C", "T>G"),
                  mutation.types.values = c(0.3, 0.3, 0.1, 0.1, 0.1, 0.1),
                  mutation.types.colors = TriNuc.Mutation.Type.Hex.Colors,
                  max.y.val = 0.5,
                  convert.to.percentage = T,
                  show.legend = T,
                  font.size.small = 8,
                  font.size.med = 14,
```

```
                  plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))
print(p)

## End(Not run)
```

`PlotOptimizationIterations` *PlotOptimizationIterations*

Description

Plots multiple scatter plots into one figure

Usage

```
PlotOptimizationIterations(df, columns.to.plot, x.axis.var, x.axis.title,
  x.max, save.file, title, y.axis.title = "", y.max = 1,
  point.size = 1, connect.dots = T, plot.legend = T,
  legend.ncol = 1, font.size.med = 14, font.size.large = 16,
  plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))
```

Arguments

`df`	A data.frame (from reading "FIREVAT_Optimization_Logs.tsv")
`columns.to.plot`	A character vector (of column names to plot)
`x.axis.var`	x axis variable
`x.axis.title`	x axis title
`x.max`	x axis maximum value
`save.file`	Filename (including full path) to which the plot will be saved
`title`	Plot title
`y.axis.title`	y axis title; Default = ""
`y.max`	y axis maximum value; Default = 1
`point.size`	Point size; Default = 1
`connect.dots`	If True draws dots for each iteration; Default = True
`plot.legend`	If True write legend of plot; Default = T
`legend.ncol`	legend.n Default = 1
`font.size.med`	Medium font size; Default = 14
`font.size.large`	Large font size; Default = 16
`plot.margin`	Margin vector for plot; Default = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))

Value

A ggplot object

PlotSignaturesContProbs *PlotSignaturesContProbs*

Description

Plots a horizontal barplot of identified mutational signatures

Usage

```
PlotSignaturesContProbs(df.identified.mut.sigs, df.ref.sigs.groups.colors,
  title, convert.to.percentage = T, font.size.small = 8,
  font.size.med = 14, plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))
```

Arguments

df.identified.mut.sigs	A data.frame of identified mutational signatures
df.ref.sigs.groups.colors	A data.frame with 'signature', 'group', and 'color' columns
title	Plot title
convert.to.percentage	If true, convert y values to percentage (x 100); Default = T,
font.size.small	Small font size; Default = 8,
font.size.med	Medium font size; Default = 14,
plot.margin	Margin vector for drawing plot; Default = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))

Value

A ggplot object

Examples

```
## Not run:
 g <- PlotSignaturesContProbs(sigs = c(mutalisk.results$identified.mut.sigs),
 sigs.probs = c(mutalisk.results$identified.mut.sigs.probs),
 df.ref.sigs.groups.colors = GetPCAWGMutSigsEtiologiesColors())
 print(g)

## End(Not run)
```

PlotTable *PlotTable*

Description

Plots basic statistics table

Usage

```
PlotTable(df, padding = 20, font.size = 14)
```

Arguments

df	= A data.frame where the first column is header and the second column is data value
padding	Padding size; Default = 20
font.size	Font size; Default = 14

Value

A plot

PlotTriNucSpectrum *PlotTriNucSpectrum*

Description

Plots the spectrum of 96 trinucleotide distribution (C>A, C>G, C>T, T>A, T>C, T>G) Please note that this function assumes that both sub.types and spectrum are sorted in the following order: C>A, C>G, C>T, T>A, T>C, T>G

Usage

```
PlotTriNucSpectrum(sub.types, spectrum, max.y.val, min.y.val, y.axis.title,
  draw.top.strip = T, draw.x.axis.labels = T, draw.y.axis.labels = T,
  draw.y.axis.title = T, font.size.small = 8, font.size.med = 14,
  plot.margin.top = 0.5, plot.margin.bottom = 0.5,
  plot.margin.left = 0.5, plot.margin.right = 0.5, title)
```

Arguments

sub.types	A character vector (types of 96 trinucleotide substitutions)
spectrum	A numeric vector (96 elements)
max.y.val	y axis maximum value
min.y.val	y axis minimum value
y.axis.title	y axis title
draw.top.strip	If True then draws top strip; Default = T
draw.x.axis.labels	If True then draws x axis labels; Default = T
draw.y.axis.labels	If True then draws y axis labels; Default = T
draw.y.axis.title	If True then draws y axis title; Default = T
font.size.small	Small font size; Default = 8
font.size.med	Medium font size; Default = 14
plot.margin.top	Top margin; Default = 0.5
plot.margin.bottom	Bottom margin; Default = 0.5
plot.margin.left	Left margin; Default = 0.5
plot.margin.right	Right margin; Default = 0.5
title	Plot title

Value

A ggplot object

PlotVCFStatsBoxPlots *PlotVCFStatsBoxPlots*

Description

Plots multiple (original, refined, artifact vcf) boxplots for single filter parameter

Usage

```
PlotVCFStatsBoxPlots(original.vcf.stat.values, refined.vcf.stat.values,
  artifact.vcf.stat.values, xlab, axis.font.size = 10,
  label.font.size = 10, title.font.size = 12)
```

Arguments

original.vcf.stat.values	A numeric vector corresponding to the original vcf.obj values of single filter parameter
refined.vcf.stat.values	A numeric vector corresponding to the refined vcf.obj values of single filter parameter
artifact.vcf.stat.values	A numeric vector corresponding to the artifact vcf.obj values of single filter parameter
xlab	A string value (x-axis label)
axis.font.size	An integer value (axis font size)
label.font.size	An integer value (label font size)
title.font.size	An integer value (title font size)

Value

A ggboxplot

PlotVCFStatsHistograms

PlotVCFStatsHistograms

Description

Plots multiple VCF stats histograms into one figure

Usage

```
PlotVCFStatsHistograms(plot.values, x.axis.labels, stat.y.max.vals,
  stat.x.max.vals, sample.id, save.file, title, cutoff.values,
  plot.boxplot = F, plot.cutoff.line.color = "#D4012E",
  plot.cutoff.value.lines = F, bin.width = 1, ncol = 4, nrow = 3,
  font.size.med = 10, font.size.large = 12, plot.margin = unit(c(0.5,
  0.5, 0.5, 0.5), "cm"))
```

Arguments

plot.values	A list of multiple numeric vectors
x.axis.labels	A character vector of x axis labels
stat.y.max.vals	A numeric vector of max y-axis values

`stat.x.max.vals`
: A numeric vector of max x-axis values

`sample.id`
: A string value of sample ID

`save.file`
: A string value of file to which the resulting plot will be saved

`title`
: A string value of plot title

`cutoff.values`
: A numeric vector of cutoff values

`plot.boxplot`
: A boolean value (default = False)

`plot.cutoff.line.color`
: A hex string value (default = "#D4012E")

`plot.cutoff.value.lines`
: A boolean value (default = False)

`bin.width`
: An integer value (default = 1; histogram bin width)

`ncol`
: An integer value (default = 4; ggarrange ncol)

`nrow`
: An integer value (default = 3; ggarrange nrow)

`font.size.med`
: An integer value (default = 10)

`font.size.large`
: An integer value (default = 12)

`plot.margin`
: A list (default = unit(c(0.5, 0.5, 0.5, 0.5), "cm"))

Value

A list with the following elements

- f = A ggarrange object
- graphs = A list of length 3; each element is a ggplot histogram

---

`PrepareAnnotationDB` *PrepareAnnotationDB*

Description

Prepares df.genes.of.interest from a vcf.obj (`ReadVCF`) of COSMIC or ClinVar vcf for `AnnotateVCFObj`

Usage

```
PrepareAnnotationDB(annotation.vcf.obj)
```

Arguments

`annotation.vcf.obj`
: vcf.obj of COSMIC or ClinVar vcf file

Value

A data.frame with the columns specified in columns.to.include

PrepareArtifactAnnotationTable
PrepareArtifactAnnotationTable

Description

Prepares artifactual mutations annotation (filtered, queried) table

Usage

```
PrepareArtifactAnnotationTable(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A data.frame

PrepareArtifactStrandBiasTable
PrepareArtifactStrandBiasTable

Description

Prepares artifactual mutations strand biased variants table

Usage

```
PrepareArtifactStrandBiasTable(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A data.frame

`PrepareArtifactualMutsOptimizationIterationsPlot`
PrepareArtifactualMutsOptimizationIterationsPlot

Description

Prepares artifactual mutations optimization iterations plot

Usage

```
PrepareArtifactualMutsOptimizationIterationsPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggplot object

`PrepareFilterCutoffsTable`
PrepareFilterCutoffsTable

Description

Prepares filter cutoffs table for reporting

Usage

```
PrepareFilterCutoffsTable(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A data.frame

PrepareGeneticAlgorithmParametersTable
PrepareGeneticAlgorithmParametersTable

Description

Prepares Genetic Algorithm parameters table

Usage

```
PrepareGeneticAlgorithmParametersTable(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A data.frame

PrepareIdentifiedSignaturesPlot
PrepareIdentifiedSignaturesPlot

Description

Prepares identified signatures plot for reporting

Usage

```
PrepareIdentifiedSignaturesPlot(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A ggarrange object

`PrepareMLEReconstructedSpectrumsPlot` *PrepareMLEReconstructedSpectrumsPlot*

Description

Prepares MLE reconstructed spectrums plot

Usage

```
PrepareMLEReconstructedSpectrumsPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggarrange object

`PrepareNucleotideSubstitutionTypesPlot` *PrepareNucleotideSubstitutionTypesPlot*

Description

Prepares nucleotide substitution types plot

Usage

```
PrepareNucleotideSubstitutionTypesPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggarrange object

`PrepareObservedSpectrumsPlot` *PrepareObservedSpectrumsPlot*

Description

Prepares observed spectrums plot

Usage

```
PrepareObservedSpectrumsPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggarrange object

`PrepareOptimizationResultsTable` *PrepareOptimizationResultsTable*

Description

Prepares optimization results table

Usage

```
PrepareOptimizationResultsTable(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A data.frame

`PrepareOptimizedVCFStatisticsPlot`
PrepareOptimizedVCFStatisticsPlot

Description

Prepares optimized VCF statistics plot

Usage

```
PrepareOptimizedVCFStatisticsPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggarrange object

`PrepareRefinedAnnotationTable`
PrepareRefinedAnnotationTable

Description

Prepares refined mutations annotation (filtered, queried) table

Usage

```
PrepareRefinedAnnotationTable(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A data.frame

PrepareRefinedMutsOptimizationIterationsPlot
PrepareRefinedMutsOptimizationIterationsPlot

Description

Prepares refined mutations optimization iterations plot

Usage

```
PrepareRefinedMutsOptimizationIterationsPlot(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A ggplot object

PrepareRefinedStrandBiasTable
PrepareRefinedStrandBiasTable

Description

Prepares refined mutations strand biased variants table

Usage

```
PrepareRefinedStrandBiasTable(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A data.frame

`PrepareResidualSpectrumsPlot`
PrepareResidualSpectrumsPlot

Description

Prepares residual spectrums plot

Usage

```
PrepareResidualSpectrumsPlot(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A ggarrange object

`PrepareTrinucleotideSpectrumsTable`
PrepareTrinucleotideSpectrumsTable

Description

Prepares trinucleotide spectrums table

Usage

```
PrepareTrinucleotideSpectrumsTable(data)
```

Arguments

`data`	A list of elements returned from `RunFIREVAT`

Value

A data.frame

QueryAnnotatedVCF *FilterAnnotatedVCF*

Description

Annotates a vcf.obj using df.variants.of.interest (from (PrepareAnnotationDB)

Usage

```
QueryAnnotatedVCF(vcf.obj.annotated, filter.key.value.pairs,
  filter.condition = "AND")
```

Arguments

vcf.obj.annotated	AnnotateVCFObj
filter.key.value.pairs	A list with the key as the column name and value as the filtering values. E.g. list("CLNSIG" = c("Pathogenic", "Pathogenic/Likely_pathogenic"))
filter.condition	'AND' or 'OR'.

Value

A vcf.obj

ReadOptimizationIterationReport
ReadOptimizationIterationReport

Description

Read optimization iteration report

Usage

```
ReadOptimizationIterationReport(data)
```

Arguments

data	A list of elements returned from RunFIREVAT

Value

A data.frame of FIREVAT optimization logs

ReadVCF *ReadVCF*

Description

Reads a .vcf file

Usage

```
ReadVCF(vcf.file, genome = "hg19", split.info = FALSE,
  check.chromosome.name = TRUE)
```

Arguments

vcf.file	(full path of a .vcf file)
genome	('hg19' or 'hg38')
split.info	A boolean value. If TRUE, then makes the INFO column in the vcf as a separate column. Default value is FALSE.
check.chromosome.name	A boolean value. If TRUE, then check whether converts 'MT' to 'M' and adds 'chr' to the CHROM column. Default value is TRUE.

Value

A list with elements 'data', 'header', 'genome'

ReportFIREVATResults *ReportFIREVATResults*

Description

Reports FIREVAT results in html format (generated from Rmd)

Usage

```
ReportFIREVATResults(data)
```

Arguments

data	A list of main data from RunFIREVAT

Value

An updated data list

RunFIREVAT *RunFIREVAT*

Description

Runs FIREVAT using configuration data.Filters point mutations in the specified vcf.file based on mutational signature decomposition and outputs the refined and artifact vcf as well as metadata related to the refinement process.

Usage

```
RunFIREVAT(vcf.file, vcf.file.genome, config.file, df.ref.mut.sigs,
  target.mut.sigs, sequencing.artifact.mut.sigs, num.cores, output.dir,
  mode = "ga", use.suggested.soln = TRUE, ga.pop.size = 200,
  ga.max.iter = 200, ga.run = 50, ga.pmutation = 0.25,
  mutalisk.method = "all", mutalisk.random.sampling.count = 20,
  mutalisk.random.sampling.max.iter = 10,
  perform.strand.bias.analysis = TRUE,
  strand.bias.perform.fdr.correction = TRUE,
  strand.bias.fdr.correction.method = "BH",
  ref.forward.strand.var = NULL, ref.reverse.strand.var = NULL,
  alt.forward.strand.var = NULL, alt.reverse.strand.var = NULL,
  report.format = "html", annotate = TRUE, df.annotation.db = NULL,
  annotated.columns.to.display = NULL,
  annotation.filter.key.value.pairs = NULL,
  annotation.filter.condition = "AND", verbose = TRUE)
```

Arguments

vcf.file	String value corresponding to input .vcf file. Please provide the full path.
vcf.file.genome	Genome assembly of the input .vcf file. The value should be eitehr 'hg19' or 'hg38'.
config.file	String value corresponding to input configuration file. For more details please refer to ...
df.ref.mut.sigs	A data.frame of the reference mutational signatures
target.mut.sigs	A character vector of the target mutational signatures from reference mutational signatures.
sequencing.artifact.mut.sigs	A character vector of the sequencing artifact mutational signatures from reference mutational signatures.
num.cores	Number of cores to allocate
output.dir	String value of the desired output directory

mode String value. The value should be either 'ga' or 'manual'.

use.suggested.soln
Boolean value. If TRUE, then FIREVAT passes the default values of filter variables declared as 'use_in_filter' in the config file to the 'suggestions' parameter of the Genetic Algorithm package. If FALSE, then FIREVAT supplies NULL to the GA package 'suggestions' parameter.

ga.pop.size Integer value of the Genetic Algorithm 'population size' parameter. Default: 200. This value should be set based on the number of filter parameters. Recommendation: 40 per filter parameter.

ga.max.iter Integer value of the Genetic Algorithm 'maximum iterations' parameter. Ddefault: 200. This value should be set based on the number of filter parameters. Recommendation: same as 'ga.pop.size'.

ga.run Integer value of the Genetic Algorithm 'run' parameter. Default: 50. This value should be set based on the 'ga.max.iter' parameter. Recommendation: 25 percent of 'ga.max.iter'.

ga.pmutation Float value of the Genetic Algorithm 'mutation probability' parameter. Default: 0.25.

mutalisk.method
Mutalisk signature identification method. Default: 'random.sampling'. The value can be either 'all' or 'random.sampling'. 'all' uses all target.mut.sigs to identify mutational signatures. 'random.sampling' randomly samples from target.mut.sigs to identify mutational signatures.

mutalisk.random.sampling.count
Mutalisk random sampling count. Default: 20. The number of signatures to sample from target.mut.sigs

mutalisk.random.sampling.max.iter
Mutalisk random sampling maximum iteration. Default: 10. The number of times Mutalisk randomly samples from target.mut.sigs before determining the candidate signatures.

perform.strand.bias.analysis
If TRUE, then performs strand bias analysis.

strand.bias.perform.fdr.correction
If TRUE, then performs false discovery rate correction for strand bias analysis.

strand.bias.fdr.correction.method
A string value. Default value is 'BH'. Refer to 'p.adjust()' function method.

ref.forward.strand.var
A string value.

ref.reverse.strand.var
A string value,

alt.forward.strand.var
A string value,

alt.reverse.strand.var
A string value,

report.format The format of FIREVAT report. We currently only support 'html'.

annotate A boolean value. Default value is TRUE.

df.annotation.db
: A data.frame. Please refer to PrepareAnnotationDB

annotated.columns.to.display
: A character vector.

annotation.filter.key.value.pairs
: A list.

annotation.filter.condition
: 'AND' or 'OR'.

verbose
: If TRUE, provides process detail. Default: TRUE.

Value

A list with the following elements

- f = A ggarrange object
- graphs = A list of length 3; each element is a ggplot histogram

---

RunMutalisk *RunMutalisk*

Description

Identifies mutational signatures using Mutalisk

Usage

```
RunMutalisk(vcf.obj, df.ref.mut.sigs, target.mut.sigs,
  random.sampling.candidate.mut.sigs = c(), method = "random.sampling",
  n.sample = 20, n.iter = 10, verbose = TRUE)
```

Arguments

vcf.obj
: A list (from firevat_vcf::ReadVCF)

df.ref.mut.sigs
: A data.frame of reference mutational signatures

target.mut.sigs
: A character vector of target mutational signatures names to identify from

random.sampling.candidate.mut.sigs
: A character vector of mutational signatures names that gets appended to the list of candidate mutational signatures so that these are always considered.

method
: A string value (must be either 'random.sampling' or 'all'). The method 'random.sampling' samples (without replacement) 'n.sample' number of signatures 'n.iter' number of times and runs the candidate signatures one last time. The method 'all' uses all target.mut.sigs

n.sample
: An integer value ('random.sampling' method parameter) Number of signatures to choose for each iteration of random sampling).

n.iter
: An integer value ('random.sampling' method parameter). Number of iterations to perform random sampling.

verbose
: If true, provides process details

Value

A list with the following elements

- num.point.mutationsAn integer value - count of total point mutations
- sub.typesA character vector of length 96
- sub.types.spectrumA numeric vector of length 96
- num.mut.sigsAn integer value (count of unique mutational signatures identified)
- identified.mut.sigsA character vector where each element is a mutational signature identified
- identified.mut.sigs.probsA numeric vector where each element is the weight of mutational signature identified. The ordering follows identified.mut.sigs
- identified.mut.sigs.spectrumA numeric vector of length 96
- residualsA numeric vector of length 96
- rssA numeric value (residual sum of squares)
- cos.sim.scoreA numeric value (cosine similarity score between observed mutational spectrum and reconstructed mutational signatures)
- all.models.sigsA list where each element is a model; a model is a list of signatures identified)
- all.models.sigs.probsA list where each element is a model; a model is a list of contribution probabilities
- all.models.cos.sim.scoresA list where each element is a model; a model is a list of cosine similarity socres

---

`RunMutaliskHelper` *RunMutaliskHelper*

Description

Helper function for RunMutalisk

Usage

```
RunMutaliskHelper(vcf.trinucleotide.data, df.ref.mut.sigs, target.mut.sigs)
```

Arguments

`vcf.trinucleotide.data`
: A data.frame (from firevat_mutalisk::MutaliskParseVCFObj)

`df.ref.mut.sigs`
: A data.frame of reference mutational signatures

`target.mut.sigs`
: A character vector of target mutational signatures names

Value

A list with the following elements

- num.point.mutationsAn integer value - count of total point mutations
- sub.typesA character vector of length 96
- sub.types.spectrumA numeric vector of length 96
- num.mut.sigsAn integer value (count of unique mutational signatures identified)
- identified.mut.sigsA character vector where each element is a mutational signature identified
- identified.mut.sigs.probsA numeric vector where each element is the weight of mutational signature identified. The ordering follows identified.mut.sigs
- identified.mut.sigs.spectrumA numeric vector of length 96
- residualsA numeric vector of length 96
- rssA numeric value (residual sum of squares)
- cos.sim.scoreA numeric value (cosine similarity score between observed mutational spectrum and reconstructed mutational signatures)
- all.models.sigsA list where each element is a model; a model is a list of signatures identified)
- all.models.sigs.probsA list where each element is a model; a model is a list of contribution probabilities
- all.models.cos.sim.scoresA list where each element is a model; a model is a list of cosine similarity socres

---

`RunMutPat` *RunMutPat*

Description

Identifies mutational signatures using Mutational Patterns

Usage

```
RunMutPat(mut.pat.input, df.mut.pat.ref.sigs, target.mut.sigs,
  verbose = TRUE)
```

Arguments

`mut.pat.input`	A list from `MutPatParseVCFObj`
`df.mut.pat.ref.sigs`	A data.frame returned by `MutPatParseRefMutSigs`
`target.mut.sigs`	A character vector of target mutational signatures names
`verbose`	If true, provides process details

Value

A list with the following elements

- tumor.mutation.types.spectrumA numeric vector of length 96 - 'observed' spectrum
- identified.mutation.types.spectrumA numeric vector of length 96 - 'identified' spectrum
- residualsA numeric vector of length 96 - residuals
- mutation.typesA character vector of length 96
- identified.mut.sigsA character vector where each element is a mutational signature identified
- identified.mut.sigs.contribution.weightsA numeric vector where each element is the weight of mutational signature identified. The ordering follows identified.mut.sigs
- cosine.similarity.scoreA numeric value

Examples

```
## Not run:
vcf.obj <- ReadVCF(vcf.file = "../data/sample/HNT-082-BT.final.call.vcf", genome = "hg19")
df.ref.mut.sigs <- GetPCAWGMutSigs()
target.mut.sigs <- GetPCAWGMutSigsNames()
RunMutPat(vcf.obj = vcf.obj,
df.ref.mut.sigs = df.ref.mut.sigs,
target.mut.sigs = target.mut.sigs)

## End(Not run)
```

TriNuc.Mutation.Type.Hex.Colors

Constant

Description

Hex codes for the mutation types (for plotting purposes)

Usage

```
TriNuc.Mutation.Type.Hex.Colors
```

Format

An object of class `character` of length 6.

UpdateFilter *UpdateFilter*

Description

Update filter based on optim parameter values

Usage

```
UpdateFilter(vcf.filter, param.values)
```

Arguments

vcf.filter	A list from MakeFilterFromConfig
param.values	A numeric vector contains filtering value (same length with length(vcf.config.filter))

Value

Updated vcf.filter (list)

WriteVCF *WriteVCF*

Description

Writes a vcf.obj to a .vcf file

Usage

```
WriteVCF(vcf.obj, save.file)
```

Arguments

vcf.obj	(from the function ReadVCF)
save.file	(full path including filename)

Index