Package 'maftools'

October 18, 2017

Type Package

Title Summarize, Analyze and Visualize MAF Files

Version 1.2.30

Date 2015-12-14

Author Anand Mayakonda <anand_mt@hotmail.com>

Maintainer Anand Mayakonda <anand_mt@hotmail.com>

Description Analyze and visualize Mutation Annotation Format (MAF) files from large scale sequencing studies. This package provides various functions to perform most commonly used analyses in cancer genomics and to create feature rich customizable visualzations with minimal effort.

URL https://github.com/PoisonAlien/maftools

BugReports https://github.com/PoisonAlien/maftools/issues

License MIT + file LICENSE

LazyData TRUE

Depends R (>= 3.3)

Imports data.table, ggplot2(>= 2.0), cowplot, cometExactTest, RColorBrewer, NMF, ggrepel, methods, ComplexHeatmap, mclust, VariantAnnotation, Biostrings, Rsamtools, rjson, grid, DPpackage, wordcloud, grDevices, changepoint, gridExtra, survival

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

biocViews DataRepresentation, DNASeq, Visualization, DriverMutation, VariantAnnotation, FeatureExtraction, Classification, SomaticMutation, Sequencing, FunctionalGenomics

NeedsCompilation no

R topics documented:

`annovarToMaf` *Converts annovar annotations into MAF.*

Description

Converts variant annotations from Annovar into a basic MAF.

Usage

```
annovarToMaf(annovar, Center = NULL, refBuild = "hg19", tsbCol = NULL,
  table = "refGene", basename = NULL, sep = "\t", MAFobj = FALSE)
```

Arguments

`annovar`	input annovar annotation file.
`Center`	Center field in MAF file will be filled with this value. Default NA.
`refBuild`	NCBI_Build field in MAF file will be filled with this value. Default hg19.
`tsbCol`	column name containing Tumor_Sample_Barcode or sample names in input file.
`table`	reference table used for gene-based annotations. Can be 'ensGene' or 'refGene'. Default 'refGene'
`basename`	If provided writes resulting MAF file to an output file.
`sep`	field seperator for input file. Default tab seperated.
`MAFobj`	If TRUE, returns results as an `MAF` object.

Details

Annovar is one of the most widely used Variant Annotation tools in Genomics. Annovar output is generally in a tabular format with various annotation columns. This function converts such annovar output files into MAF. This function requires that annovar was run with gene based annotation as a first operation, before including any filter or region based annotations. Please be aware that this function performs no transcript prioritization.

e.g, table_annovar.pl example/ex1.avinput humandb/ -buildver hg19 -out myanno -remove -protocol (`refGene`),cytoBand,dbnsfp30a -operation (g),r,f -nastring NA

This function mainly uses gene based annotations for processing, rest of the annotation columns from input file will be attached to the end of the resulting MAF.

Value

MAF table.

References

Wang, K., Li, M. & Hakonarson, H. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. Nucleic Acids Res 38, e164 (2010).

Examples

```
var.annovar <- system.file("extdata", "variants.hg19_multianno.txt", package = "maftools")
var.annovar.maf <- annovarToMaf(annovar = var.annovar, Center = 'CSI-NUS', refBuild = 'hg19',
tsbCol = 'Tumor_Sample_Barcode', table = 'ensGene')
```

coOncoplot — *Draw two oncoplots side by side for cohort comparision.*

Description

Draw two oncoplots side by side for cohort comparision.

Usage

```
coOncoplot(m1, m2, genes = NULL, colors = NULL, removeNonMutated = TRUE,
  m1Name = NULL, m2Name = NULL)
```

Arguments

`m1`	first `MAF` object
`m2`	second `MAF` object
`genes`	draw these genes. Default plots top 5 mutated genes from two cohorts.
`colors`	named vector of colors for each Variant_Classification.
`removeNonMutated`	Logical. If `TRUE` removes samples with no mutations in the oncoplot for better visualization. Default `TRUE`.
`m1Name`	optional name for first cohort
`m2Name`	optional name for second cohort

Details

Draws two oncoplots side by side to display difference between two cohorts.

Value

Returns nothing. Just draws plot.

Examples

```
#' ##Primary and Relapse APL
primary.apl <- system.file("extdata", "APL_primary.maf.gz", package = "maftools")
relapse.apl <- system.file("extdata", "APL_relapse.maf.gz", package = "maftools")
##Read mafs
primary.apl <- read.maf(maf = primary.apl)
relapse.apl <- read.maf(maf = relapse.apl)
##Plot
coOncoplot(m1 = primary.apl, m2 = relapse.apl, m1Name = 'Primary APL', m2Name = 'Relapse APL')
dev.off()
```

`extractSignatures` *Extract mutational signatures from trinucletide context.*

Description

Decompose a matrix of 96 substitution classes into `n` signatures.

Usage

```
extractSignatures(mat, n = NULL, nTry = 6, plotBestFitRes = FALSE,
  parallel = NULL)
```

Arguments

`mat`	Input matrix of diemnsion nx96 generated by `trinucleotideMatrix`
`n`	decompose matrix into `n` signatures. Default NULL. Tries to predict best value for `n` by running NMF on a range of values and chooses based on cophenetic correlation coefficient.
`nTry`	tries upto this number of signatures before choosing best `n`. Default 6.
`plotBestFitRes`	plots consensus heatmap for range of values tried. Default FALSE
`parallel`	calls to .opt argument of `nmf`. e.g, 'P4' for using 4 cores. See note on `nmf` for MAC users.

Details

This function decomposes a non-negative matrix into n signatures. Extracted signatures are compared against 30 experimentally validated signatures by calculating cosine similarity. See http://cancer.sanger.ac.uk/cosm
for details.

Value

a list with decomposed scaled signatures, signature contributions in each sample and a cosine similarity table against validated signatures.

See Also

`trinucleotideMatrix` `plotSignatures`

Examples

```
## Not run:
laml.tnm <- trinucleotideMatrix(maf = laml, ref_genome = 'hg19.fa', prefix = 'chr',
add = TRUE, useSyn = TRUE)
laml.sign <- extractSignatures(mat = laml.tnm, plotBestFitRes = FALSE)

## End(Not run)
```

forestPlot *Draw forest plot for differences betweeen cohorts.*

Description

Draw forest plot for differences betweeen cohorts.

Usage

```
forestPlot(mafCompareRes, pVal = 0.05, show = NULL, color = NULL,
  file = NULL, width = 5, height = 6)
```

Arguments

`mafCompareRes`	results from `mafCompare`
`pVal`	p-value threshold. Default 0.05.
`show`	can be either `stat` or `pval`
`color`	vector of colors for cohorts. Default NULL.
`file`	basename for output file. Plot will saved to an output pdf.
`width`	width of plot to be generated
`height`	height of plot to be generated

Details

Plots results from `link{mafCompare}` as a forest plot with x-axis as log10 converted odds ratio and differentially mutated genes on y-axis.

Value

ggplot object of the plot.

See Also

`mafCompare`

Examples

```
##Primary and Relapse APL
primary.apl <- system.file("extdata", "APL_primary.maf.gz", package = "maftools")
relapse.apl <- system.file("extdata", "APL_relapse.maf.gz", package = "maftools")
##Read mafs
primary.apl <- read.maf(maf = primary.apl)
relapse.apl <- read.maf(maf = relapse.apl)
##Perform analysis and draw forest plot.
pt.vs.rt <- mafCompare(m1 = primary.apl, m2 = relapse.apl, m1Name = 'Primary',
m2Name = 'Relapse', minMut = 5)
forestPlot(mafCompareRes = pt.vs.rt, show = 'stat')
```

`geneCloud` *Plots wordcloud.*

Description

Plots word cloud of mutated genes or altered cytobands with size proportional to the event frequency.

Usage

```
geneCloud(input, minMut = 3, col = NULL, top = NULL,
  genesToIgnore = NULL, ...)
```

Arguments

`input`	an `MAF` or `GISTIC` object generated by `read.maf` or `readGistic`
`minMut`	Minimum number of samples in which a gene is required to be mutated.
`col`	vector of colors to choose from.
`top`	Just plot these top n number of mutated genes.
`genesToIgnore`	Ignore these genes.
`...`	Other options passed to `wordcloud`

Value

nothing.

Examples

```
laml.input <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.input, useAll = FALSE)
geneCloud(input = laml, minMut = 5)
```

`genesToBarcodes` *Extracts Tumor Sample Barcodes where the given genes are mutated.*

Description

Extracts Tumor Sample Barcodes where the given genes are mutated.

Usage

```
genesToBarcodes(maf, genes = NULL, justNames = FALSE)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`genes`	Hogo_Symbol for which sample names to be extracted.
`justNames`	if TRUE, just returns samples names instead of summarized tables.

Value

list of `data.tables` with samples in which given genes are mutated.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
genesToBarcodes(maf = laml, genes = 'DNMT3A')
```

`getCytobandSummary` *extract cytoband summary from GISTIC object*

Description

extract cytoband summary from GISTIC object

Usage

```
getCytobandSummary(x)

## S4 method for signature 'GISTIC'
getCytobandSummary(x)
```

Arguments

x	An object of class GISTIC

Value

summarizied gistic results by altered cytobands.

Examples

```
all.lesions <- system.file("extdata", "all_lesions.conf_99.txt", package = "maftools")
amp.genes <- system.file("extdata", "amp_genes.conf_99.txt", package = "maftools")
del.genes <- system.file("extdata", "del_genes.conf_99.txt", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesF
getCytobandSummary(laml.gistic)
```

getFields *extract available fields from MAF object*

Description

extract available fields from MAF object

Usage

```
getFields(x)

## S4 method for signature 'MAF'
getFields(x)
```

Arguments

x	An object of class MAF

Value

Field names in MAF file

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
getFields(x = laml)
```

getGeneSummary *extract gene summary from MAF or GISTIC object*

Description

extract gene summary from MAF or GISTIC object

Usage

```
getGeneSummary(x)

## S4 method for signature 'MAF'
getGeneSummary(x)

## S4 method for signature 'GISTIC'
getGeneSummary(x)
```

Arguments

x	An object of class MAF or GISTIC

Value

gene summary table

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
getGeneSummary(laml)
```

---

getSampleSummary *extract sample summary from MAF or GISTIC object*

Description

extract sample summary from MAF or GISTIC object

Usage

```
getSampleSummary(x)

## S4 method for signature 'MAF'
getSampleSummary(x)

## S4 method for signature 'GISTIC'
getSampleSummary(x)
```

Arguments

x	An object of class MAF or GISTIC

Value

sample summary table

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
getSampleSummary(x = laml)
```

`GISTIC-class` *Class GISTIC*

Description

S4 class for storing summarized MAF.

Slots

`data` data.table of summarized GISTIC file.

`cnv.summary` table containing alterations per sample

`cytoband.summary` table containing alterations per cytoband

`gene.summary` table containing alterations per gene

`cnMatrix` character matrix of dimension n*m where n is number of genes and m is number of samples

`numericMatrix` numeric matrix of dimension n*m where n is number of genes and m is number of samples

`summary` table with basic GISTIC summary stats

`classCode` mapping between numeric values in numericMatrix and copy number events.

See Also

`getGeneSummary` `getSampleSummary` `getCytobandSummary`

`gisticPlot` *Plot gistic results.*

Description

takes output generated by readGistic and draws a plot similar to oncoplot.

Usage

```
gisticPlot(gistic, top = NULL, showTumorSampleBarcodes = FALSE,
  annotation = NULL, bandsToIgnore = NULL, removeNonAltered = FALSE,
  colors = NULL, fontSize = 10)
```

Arguments

`gistic`	an `GISTIC` object generated by `readGistic`
`top`	how many top cytobands to be drawn. defaults to all.
`showTumorSampleBarcodes`	logical to include sample names.
`annotation`	data.frame with first column containing Tumor_Sample_Barcodes and rest of columns with annotations.
`bandsToIgnore`	do not show these bands in the plot Default NULL.

`removeNonAltered`	Logical. If `TRUE` removes samples with no mutations in the oncoplot for better visualization. Default `FALSE`.
`colors`	named vector of colors Amp and Del events.
`fontSize`	font size for cytoband names. Default 10.

Details

Takes gistic file as input and plots it as a matrix. Any desired annotations can be added at the bottom of the oncoplot by providing `annotation`

Value

None.

See Also

`oncostrip`

Examples

```
all.lesions <- system.file("extdata", "all_lesions.conf_99.txt", package = "maftools")
amp.genes <- system.file("extdata", "amp_genes.conf_99.txt", package = "maftools")
del.genes <- system.file("extdata", "del_genes.conf_99.txt", package = "maftools")
gistic.summary = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGen
gisticPlot(gistic.summary)
```

`icgcSimpleMutationToMAF` *Converts ICGC Simple Somatic Mutation format file to MAF*

Description

Converts ICGC Simple Somatic Mutation format file to Mutation Annotation Format. Basic fields are converted as per MAF specififcations, rest of the fields are retained as in the input file. Ensemble gene IDs are converted to HGNC Symbols. Note that by default Simple Somatic Mutation format contains all affected transcripts of a variant resuting in multiple entries of the same variant in same sample. It is hard to choose a single affected transcript based on annotations alone and by default this program removes repeated variants as duplicated entries. If you wish to keep all of them, set removeDuplicatedVariants to FALSE.

Usage

```
icgcSimpleMutationToMAF(icgc, basename = NA, MAFobj = FALSE,
  removeDuplicatedVariants = TRUE, addHugoSymbol = FALSE)
```

Arguments

`icgc`	Input data in ICGC Simple Somatic Mutation format. Can be gz compressed.
`basename`	If given writes to output file with basename.
`MAFobj`	If TRUE returns results as an `MAF` object.
`removeDuplicatedVariants`	removes repeated variants in a particuar sample, mapped to multiple transcripts of same Gene. See Description. Default TRUE.
`addHugoSymbol`	If TRUE replaces ensemble gene IDs with Hugo_Symbols. Default FALSE.

Details

ICGC Simple Somatic Mutattion format specififcation can be found here: http://docs.icgc.org/submission/guide/icgc-simple-somatic-mutation-format/

Value

tab delimited MAF file.

Examples

```
esca.icgc <- system.file("extdata", "simple_somatic_mutation.open.ESCA-CN.sample.tsv.gz", package = "maftool
esca.maf <- icgcSimpleMutationToMAF(icgc = esca.icgc)
```

`inferHeterogeneity` *Clusters variants based on Variant Allele Frequencies (VAF).*

Description

takes output generated by read.maf and clusters variants to infer tumor heterogeneity. This function requires VAF for clustering and density estimation. VAF can be on the scale 0-1 or 0-100. Optionally if copy number information is available, it can be provided as a segmented file (e.g, from Circular Binary Segmentation). Those variants in copy number altered regions will be ignored.

Usage

```
inferHeterogeneity(maf, tsb = NULL, top = 5, vafCol = NULL,
  dirichlet = FALSE, segFile = NULL, ignChr = NULL, minVaf = 0,
  maxVaf = 1)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`tsb`	specify sample names (Tumor_Sample_Barcodes) for which clustering has to be done.
`top`	if `tsb` is NULL, uses top n number of most mutated samples. Defaults to 5.
`vafCol`	manually specify column name for vafs. Default looks for column 't_vaf'
`dirichlet`	If TRUE uses nonparametric dirichlet process for clustering. Default FALSE, uses finite mixture models.

`segFile`	path to CBS segmented copy number file. Column names should be Sample, Chromosome, Start, End, Num_Probes and Segment_Mean (log2 scale).
`ignChr`	ignore these chromosomes from analysis. e.g, sex chromsomes chrX, chrY. Default NULL.
`minVaf`	filter low frequency variants. Low vaf variants maybe due to sequencing error. Default 0. (on the scale of 0 to 1)
`maxVaf`	filter high frequency variants. High vaf variants maybe due to copy number alterations or impure tumor. Default 1. (on the scale of 0 to 1)

Details

This function clusters variants based on VAF to estimate univariate density and cluster classification. There are two methods available for clustering. Default using parametric finite mixture models and another method using nonparametric inifinite mixture models (Dirichlet process).

Value

list of clustering tables.

References

Chris Fraley and Adrian E. Raftery (2002) Model-based Clustering, Discriminant Analysis and Density Estimation Journal of the American Statistical Association 97:611-631

Jara A, Hanson TE, Quintana FA, Muller P, Rosner GL. DPpackage: Bayesian Semi- and Nonparametric Modeling in R. Journal of statistical software. 2011;40(5):1-30.

Olshen AB, Venkatraman ES, Lucito R, Wigler M. Circular binary segmentation for the analysis of array-based DNA copy number data. Biostatistics. 2004;5(4):557-72.

See Also

plotClusters

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
TCGA.AB.2972.clust <- inferHeterogeneity(maf = laml, tsb = 'TCGA.AB.2972', vafCol = 'i_TumorVAF_WU')
```

`lollipopPlot` *Draws lollipop plot of amino acid changes on to Protein structure.*

Description

Draws lollipop plot of amino acid changes.

Usage

```
lollipopPlot(maf, gene = NULL, AACol = NULL, labelPos = NULL,
  showMutationRate = TRUE, fn = NULL, showDomainLabel = TRUE,
  cBioPortal = FALSE, refSeqID = NULL, proteinID = NULL, repel = FALSE,
  collapsePosLabel = TRUE, legendTxtSize = 10, labPosSize = 2,
  labPosAngle = 0, domainLabelSize = 2.5, printCount = FALSE,
  colors = NULL, domainColors = NULL, labelOnlyUniqueDoamins = TRUE,
  defaultYaxis = TRUE)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`gene`	HGNC symbol for which protein structure to be drawn.
`AACol`	manually specify column name for amino acid changes. Default looks for fields 'HGVSp_Short', 'AAChange' or 'Protein_Change'. Changes can be of any format i.e, can be a numeric value or HGVSp annotations (e.g; p.P459L, p.L2195Pfs*30 or p.Leu2195ProfsTer30)
`labelPos`	Amino acid positions to label. If 'all', labels all variants.
`showMutationRate`	Default TRUE
`fn`	basename for plot file to be saved. If provided a pdf will be generated. Default NULL.
`showDomainLabel`	Label domains within the plot. Default TRUE. If FALSE they will be annotated in legend.
`cBioPortal`	Adds annotations similar to cBioPortals MutationMapper and collapse Variants into Truncating and rest.
`refSeqID`	RefSeq transcript identifier for gene if known.
`proteinID`	RefSeq protein identifier for gene if known.
`repel`	If points are too close to each other, use this option to repel them. Default FALSE. Warning: naive method, might make plot ugly in case of too many variants!
`collapsePosLabel`	Collapses overlapping labels at same position. Default TRUE
`legendTxtSize`	Text size for legend. Default 10
`labPosSize`	Text size for labels. Default 2
`labPosAngle`	angle for labels. Defaults to horizonal 0 degree labels. Set to 90 for vertical; 45 for diagonal labels.
`domainLabelSize`	text size for domain labels. Default 2.
`printCount`	If TRUE, prints number of summarized variants for the given protein.
`colors`	named vector of colors for each Variant_Classification. Default NULL.
`domainColors`	Manual colors for protein domains
`labelOnlyUniqueDoamins`	Default TRUE only labels unique doamins.
`defaultYaxis`	If FALSE, just labels min and maximum y values on y axis.

Details

This function by default looks for fields 'HGVSp_Short', 'AAChange' or 'Protein_Change' in maf file. One can also manually specify field name containing amino acid changes.

Value

ggplot object of the plot, which can be futher modified.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
lollipopPlot(maf = laml, gene = 'KIT', AACol = 'Protein_Change')
```

`MAF-class` *Class MAF*

Description

S4 class for storing summarized MAF.

Slots

`data` data.table of original MAF file.

`variants.per.sample` table containing variants per sample

`variant.type.summary` table containing variant types per sample

`variant.classification.summary` table containing variant classification per sample

`gene.summary` table containing variant classification per gene

`oncoMatrix` character matrix of dimension n*m where n is number of genes and m is number of variants

`numericMatrix` numeric matrix of dimension n*m where n is number of genes and m is number of variants

`summary` table with basic MAF summary stats

`classCode` mapping between numeric values in numericMatrix and Variant Classification

`maf.silent` subset of main MAF containing only silent variants

See Also

`getGeneSummary` `getSampleSummary` `getFields`

mafCompare — *compare two cohorts (MAF).*

Description

compare two cohorts (MAF).

Usage

```
mafCompare(m1, m2, m1Name = NULL, m2Name = NULL, minMut = 5)
```

Arguments

m1	first MAF object
m2	second MAF object
m1Name	optional name for first cohort
m2Name	optional name for second cohort
minMut	Consider only genes with minimum this number of samples mutated in atleast one of the cohort for analysis. Helful to ignore single mutated genes. Default 5.

Details

Performs fisher test on 2x2 contigency table generated from two cohorts to find differentially mutated genes.

Value

result list

See Also

forestPlot

Examples

```
primary.apl <- system.file("extdata", "APL_primary.maf.gz", package = "maftools")
relapse.apl <- system.file("extdata", "APL_relapse.maf.gz", package = "maftools")
primary.apl <- read.maf(maf = primary.apl)
relapse.apl <- read.maf(maf = relapse.apl)
pt.vs.rt <- mafCompare(m1 = primary.apl, m2 = relapse.apl, m1Name = 'Primary',
m2Name = 'Relapse', minMut = 5)
```

mafSurvival *Performs survival analysis*

Description

Performs survival analysis by grouping samples from maf based on mutation status of given gene(s) or manual grouping of samples.

Usage

```
mafSurvival(maf, clinicalData, genes = NULL, samples = NULL,
  time = "Time", Status = "Status", groupNames = c("Mutant", "WT"),
  showConfInt = TRUE, addInfo = TRUE, col = c("maroon", "royalblue"),
  isTCGA = FALSE, textSize = 7, fn = NULL, width = 6, height = 6)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`clinicalData`	data containing events and time to events.
`genes`	gene names for which survival analysis needs to be performed.
`samples`	samples to group by. Genes and samples are mutually exclusive.
`time`	column name contining time in `clinicalData`
`Status`	column name containing status of patients in `clinicalData`. e.g, Dead or Alive, 1 or 0.
`groupNames`	names for groups. Should be of length two. Default c("Mutant", "WT")
`showConfInt`	TRUE. Whether to show confidence interval in KM plot.
`addInfo`	TRUE. Whether to show survival info in the plot.
`col`	colors for plotting.
`isTCGA`	FALSE. Is data is from TCGA.
`textSize`	Text size for surv table. Default 7.
`fn`	NULL. If provided saves pdf plot with basename fn.
`width`	width of plot to be saved. Default 6
`height`	height of plot to be saved. Default 6

Details

This function takes MAF file and groups them based on mutation status associated with given gene(s) and performs survival analysis. Requires dataframe containing survival status and time to event. Make sure sample names match to Tumor Sample Barcodes from MAF file.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.surv <- read.delim(system.file("extdata", "laml_survival.tsv", package = "maftools"))
mafSurvival(maf = laml, clinicalData = laml.surv, genes = 'DNMT3A', time = 'days_to_last_followup', Status =
```

`math.score` *calculates MATH (Mutant-Allele Tumor Heterogeneity) score.*

Description

calcuates MATH scores from variant allele frequencies. Mutant-Allele Tumor Heterogeneity (MATH) score is a measure of intra-tumor genetic heterogeneity. High MATH scores are related to lower survival rates. This function requies vafs.

Usage

```
math.score(maf, plotFile = NULL, vafCol = NULL, sampleName = NULL,
  vafCutOff = 0.075)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`plotFile`	file name for output plot.
`vafCol`	manually specify column name for vafs. Default looks for column 't_vaf'
`sampleName`	sample name for which MATH score to be calculated. If NULL, calculates for all samples.
`vafCutOff`	minimum vaf for a variant to be considered for score calculation. Default 0.075

Value

`data.table` with MATH score for every Tumor_Sample_Barcode

References

Mroz, Edmund A. et al. Intra-Tumor Genetic Heterogeneity and Mortality in Head and Neck Cancer: Analysis of Data from The Cancer Genome Atlas. Ed. Andrew H. Beck. PLoS Medicine 12.2 (2015): e1001786.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.math <- math.score(maf = laml, vafCol = 'i_TumorVAF_WU',
sampleName = c('TCGA.AB.3009', 'TCGA.AB.2849', 'TCGA.AB.3002', 'TCGA.AB.2972'))
```

mutExclusive — *Performs exact test for mutual exclusive events.*

Description

Performs statistical test between given set of genes for mutual excluisiveness.

Usage

```
mutExclusive(maf, genes = NULL, top = 10)
```

Arguments

maf	an MAF object generated by read.maf
genes	A pair of genes between which test should be performed. If its null, test will be performed between all combinations of top ten genes.
top	check for exclusiveness among top 'n' number of genes. Defaults to top 10. genes

Value

table with number of events in all possible combinations and p-value. Column header describes mutation status of gene1 and gene2 respectively. n.00 number of samples where both gene1 and gene2 are not mutated c.01 number of samples where gene1 is not mutated but gene2 is mutated and so on.

References

Leiserson, Mark DM et al. CoMEt: A Statistical Approach to Identify Combinations of Mutually Exclusive Alterations in Cancer. Genome Biology 16.1 (2015): 160.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
mutExclusive(maf = laml, top = 5)
```

oncodrive — *Detect cancer driver genes based on positional clustering of variants.*

Description

Clusters variants based on their position to detect disease causing genes.

Usage

```
oncodrive(maf, AACol = NULL, minMut = 5, pvalMethod = "zscore",
  nBgGenes = 100, bgEstimate = TRUE, ignoreGenes = NULL)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`AACol`	manually specify column name for amino acid changes. Default looks for field 'AAChange'
`minMut`	minimum number of mutations required for a gene to be included in analysis. Default 5.
`pvalMethod`	either zscore (default method for oncodriveCLUST), poisson or combined (uses lowest of the two pvalues).
`nBgGenes`	minimum number of genes required to estimate background score. Default 100. Do not change this unless its necessary.
`bgEstimate`	If FALSE skips background estimation from synonymous variants and uses predifined values estimated from COSMIC synonymous variants.
`ignoreGenes`	Ignore these genes from analysis. Default NULL. Helpful in case data contains large number of variants belonging to polymorphic genes such as mucins and TTN.

Details

This is the re-implimentation of algorithm defined in OncodriveCLUST article. Concept is based on the fact that most of the variants in cancer causing genes are enriched at few specific loci (aka hotspots). This method takes advantage of such positions to identify cancer genes. Cluster score of 1 means, a single hotspot hosts all observed variants. If you use this function, please cite OncodriveCLUST article.

Value

data table of genes ordered according to p-values.

References

Tamborero D, Gonzalez-Perez A and Lopez-Bigas N. OncodriveCLUST: exploiting the positional clustering of somatic mutations to identify cancer genes. Bioinformatics. 2013; doi: 10.1093/bioinformatics/btt395s

See Also

`plotOncodrive`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.sig <- oncodrive(maf = laml, AACol = 'Protein_Change', minMut = 5)
```

oncoplot *draw an oncoplot*

Description

takes output generated by read.maf and draws an oncoplot (aka waterfall plot).

Usage

```
oncoplot(maf, writeMatrix = FALSE, top = 20, genes = NULL,
  drawRowBar = TRUE, drawColBar = TRUE, showTumorSampleBarcodes = FALSE,
  annotation = NULL, annotationColor = NULL, genesToIgnore = NULL,
  removeNonMutated = TRUE, colors = NULL, fontSize = 10,
  sortByMutation = FALSE, sortByAnnotation = FALSE)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`writeMatrix`	writes character coded matrix used to generate the plot to an output file. This can be used as an input for ComplexHeatmap oncoPrint function if you wish to customize the plot.
`top`	how many top genes to be drawn. defaults to 20.
`genes`	Just draw oncoplot for these genes. defaults to NULL.
`drawRowBar`	logical plots barplot for each gene.
`drawColBar`	logical plots barplot for each sample.
`showTumorSampleBarcodes`	logical to include sample names.
`annotation`	data.frame with first column containing Tumor_Sample_Barcodes and rest of columns with annotations.
`annotationColor`	list of colors to use for annotation. Default NULL.
`genesToIgnore`	do not show these genes in Oncoplot. Default NULL.
`removeNonMutated`	Logical. If `TRUE` removes samples with no mutations in the oncoplot for better visualization. Default `TRUE`.
`colors`	named vector of colors for each Variant_Classification.
`fontSize`	font size for gene names. Default 10.
`sortByMutation`	Helpful in case of MAF was read along with copy number data. Default FALSE.
`sortByAnnotation`	logical sort oncomatrix by provided annotations. Defaults to FALSE. This is mutually exclusive with sortByMutation.

Details

Takes maf file as input and plots it as a matrix. Any desired annotations can be added at the bottom of the oncoplot by providing `annotation`. Oncoplot can be sorted either by mutations or annotations using arguments `sortByMutation` and `sortByAnnotation` respectively.

Thanks to Ryan Morin for `sortByAnnotation` code.

Value

None.

See Also

oncostrip

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
oncoplot(maf = laml, top = 3)
```

`oncostrip` *draw an oncostrip similar to cBioportal oncoprinter output.*

Description

draw an oncostrip similar to cBioportal oncoprinter output.

Usage

```
oncostrip(maf, genes = NULL, sort = TRUE, sortByAnnotation = FALSE,
  annotation = NULL, annotationColor = NULL, removeNonMutated = TRUE,
  top = 5, showTumorSampleBarcodes = FALSE, colors = NULL)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`genes`	draw oncoprint for these genes. default NULL. Plots top 5 genes.
`sort`	logical sort oncomatrix for enhanced visualization. Defaults to TRUE.
`sortByAnnotation`	logical sort oncomatrix by provided annotations. Defaults to FALSE. This is mutually exclusive with `sort`.
`annotation`	`data.frame` with first column containing Tumor_Sample_Barcodes and rest of columns with annotations.
`annotationColor`	list of colors to use for annotation. Default NULL.
`removeNonMutated`	Logical. If `TRUE` removes samples with no mutations in the oncoplot for better visualization. Default TRUE.
`top`	how many top genes to be drawn. defaults to 5.
`showTumorSampleBarcodes`	logical to include sample names.
`colors`	named vector of colors for each Variant_Classification.

Value

None.

See Also

oncoplot

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
dev.new()
oncostrip(maf = laml, genes = c('NPM1', 'RUNX1'), removeNonMutated = TRUE)
```

---

oncotate *Annotates given variants using oncotator api.*

Description

Takes variants as input and annotates them using Broad's oncotator api (http://www.broadinstitute.org/oncotator/). Output is a dataframe of annotated variants in maf format.

Input should be a five column file with chr, start, end, ref_allele, alt_allele (and so on, but only first five will used, rest will be attached to resulting maf file). Note: Time consuming if input is huge. Try to include necessary columns such as Tumor_Sample_Barcode along with above 5 fields.

Usage

```
oncotate(maflite, header = FALSE, basename = NULL)
```

Arguments

maflite	input tsv file with chr, start, end, ref_allele, alt_allele columns. (rest of the columns, if present will be attached to the output maf)
header	logical. Whether input has a header line. Default is FALSE.
basename	NULL. if basename is given, annotations will be written to <basename>.maf file.

Value

returns a dataframe in maf format.

Examples

```
sample.var = data.frame(chromsome = c('chr4', 'chr15'), Start = c(55589774, 41961117),
end = c(55589774, 41961117), ref = c('A', 'TGGCTAA'), alt = c('G', '-'),
Tumor_Sample_Barcode = c('fake_1', 'fake2'))
write.table(sample.var, 'sampleVars.txt', sep='\t',quote = FALSE, row.names = FALSE)
##var.maf <- oncotate(maflite = 'sampleVars.txt', header = TRUE)
```

`pancanComparision` *Perform PacCancer analysis*

Description

Takes MutSig results and compares them against PanCancer results.

Usage

```
pancanComparision(mutsigResults, qval = 0.1, cohortName = "input",
  inputSampleSize = NULL, label = 1, normSampleSize = FALSE,
  file = NULL, width = 6, height = 6, pointSize = 3, labelSize = 3)
```

Arguments

`mutsigResults`	MutSig results (usually sig_genes.txt). Can be gz compressed.
`qval`	qvalue threshold to define SMG. Default 0.1
`cohortName`	Input cohort name.
`inputSampleSize`	Sample size from MAF file used to generate mutSig results. Optional.
`label`	Default 1. Can be 1, 2 or 3.
`normSampleSize`	normalizes gene sizes to draw bubble plot. Requires inputSampleSize. i.e, bubble sizes proportional to fraction of samples in which the gene is mutated.
`file`	basename for output file (both raw data and plot are saved)
`width`	width of the file to be saved.
`height`	height of the file to be saved.
`pointSize`	size for scatter plot. Default 1.
`labelSize`	label text size. Default 3

Details

This function takes MutSig results and compares them against panCancer cohort (~5000 tumor samples from 21 cancer types). This analysis can reveal novel genes exclusively mutated in input cohort.

References

Lawrence MS, Stojanov P, Mermel CH, et al. Discovery and saturation analysis of cancer genes across 21 tumor types. Nature. 2014;505(7484):495-501. doi:10.1038/nature12912.

Examples

```
laml.mutsig <- system.file("extdata", "LAML_sig_genes.txt.gz", package = "maftools")
pancanComparision(mutsigResults = laml.mutsig, qval = 0.1, cohortName = 'LAML', inputSampleSize = 200, label
```

`pfamDomains` *pfam domain annotation and summarization.*

Description

Summarizes amino acid positions and annotates them with pfam domain information.

Usage

```
pfamDomains(maf = NULL, AACol = NULL, summarizeBy = "AAPos", top = 5,
  baseName = NULL, varClass = "nonSyn")
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`AACol`	manually specify column name for amino acid changes. Default looks for field 'AAChange'
`summarizeBy`	Summarize domains by amino acid position or conversions. Can be "AAPos" or "AAChange"
`top`	How many top mutated domains to label in the scatter plot. Defaults to 5.
`baseName`	If given writes the results to output file. Default NULL.
`varClass`	which variants to consider for summarization. Can be nonSyn, Syn or all. Default nonSyn.

Value

returns a list two tables summarized by amino acid positions and domains respectively. Also plots top 5 most mutated domains as scatter plot.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
pfamDomains(maf = laml, AACol = 'Protein_Change')
```

`plotCBSsegments` *Plots segmented copy number data.*

Description

Plots segmented copy number data.

Usage

```
plotCBSsegments(cbsFile = NULL, maf = NULL, tsb = NULL, chr = NULL,
  savePlot = FALSE, width = 6, height = 3, labelAll = FALSE,
  genes = NULL, ref.build = "hg19", writeTable = FALSE,
  removeXY = FALSE, color = NULL)
```

Arguments

`cbsFile`	CBS segmented copy number file. Column names should be Sample, Chromosome, Start, End, Num_Probes and Segment_Mean (log2 scale).
`maf`	optional `MAF`
`tsb`	If segmentation file contains many samples (as in gistic input), specify sample name here. Defualt plots all samples. If you are maping maf, make sure sample names in Sample column of segmentation file matches to those Tumor_Sample_Barcodes in MAF.
`chr`	Just plot this chromosome.
`savePlot`	If true plot is saved as pdf.
`width`	width of plot
`height`	height of plot
`labelAll`	If true and if maf object is specified, maps all mutataions from maf onto segments. Default FALSE, maps only variants on copy number altered regions.
`genes`	highlight only these variants
`ref.build`	Reference build for chromosome sizes. Can be hg18, hg19 or hg38. Default hg19.
`writeTable`	If true and if maf object is specified, writes plot data with each variant and its corresponding copynumber to an output file.
`removeXY`	don not plot sex chromosomes.
`color`	Manually specify color scheme for chromosomes. Default NULL.

Details

this function takes segmented copy number data and plots it. If MAF object is specified, all mutations are highlighted on the plot.

Value

ggplot object

Examples

```
tcga.ab.009.seg <- system.file("extdata", "TCGA.AB.3009.hg19.seg.txt", package = "maftools")
plotCBSsegments(cbsFile = tcga.ab.009.seg)
```

`plotClusters` *Plot density plots from clutering results.*

Description

Plots results from inferHeterogeneity.

Usage

```
plotClusters(clusters, tsb = NULL, genes = NULL, showCNvars = FALSE,
  savePlot = FALSE, width = 6, height = 5, colors = NULL)
```

Arguments

`clusters`	clustering results from `inferHeterogeneity`
`tsb`	sample to plot from clustering results. Default plots all samples from results.
`genes`	genes to highlight on the plot. Can be a vector of gene names, `CN_altered` to label copy number altered varinats. or `all` to label all genes. Default NULL.
`showCNvars`	show copy numbered altered variants on the plot. Default FALSE.
`savePlot`	If TRUE saves plot to output pdf
`width`	plot width. Default 6.
`height`	plot height. Default 5.
`colors`	manual colors for clusters. Default NULL.

Value

returns nothing.

See Also

`inferHeterogeneity`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
seg = system.file('extdata', 'TCGA.AB.3009.hg19.seg.txt', package = 'maftools')
TCGA.AB.3009.clust <- inferHeterogeneity(maf = laml, tsb = 'TCGA.AB.3009',
segFile = seg, vafCol = 'i_TumorVAF_WU')
plotClusters(TCGA.AB.3009.clust, genes = c('NF1', 'SUZ12'), showCNvars = TRUE)
```

`plotGisticResults` *Plot gistic results as a bubble plot or as a genomic segment plot.*

Description

Plots significantly altered cytobands as a function of number samples in which it is altered and number genes it contains. Size of each bubble is according to -log10 transformed q values. Optionally if scores.gistic file is provided, a genomic plot is generated with segments highlighting signififcant Amplifications and Deletion regions.

Usage

```
plotGisticResults(gistic = NULL, gis.scores = NULL, fdrCutOff = 0.1,
  markBands = NULL, color = NULL, ref.build = "hg19",
  cytobandOffset = 0.03, file = NULL, width = 6, height = 5,
  txtSize = 3)
```

Arguments

`gistic`	an object of class `GISTIC` generated by `readGistic`
`gis.scores`	optional scores.gistic file generated by GISTIC. If provided plot would be a histogram of amplifications and deletions along the genome. Default NULL
`fdrCutOff`	fdr cutoff to use. Default 0.1
`markBands`	any cytobands to label.
`color`	colors for Amp and Del events.
`ref.build`	reference build. Could be hg18, hg19 or hg38.
`cytobandOffset`	if scores.gistic file is given use this to adjust cytoband size.
`file`	if given saves plot as a pdf.
`width`	width of the file to be saved.
`height`	height of the file to be saved.
`txtSize`	label size for bubbles.

Value

nothing

Examples

```
all.lesions <- system.file("extdata", "all_lesions.conf_99.txt", package = "maftools")
amp.genes <- system.file("extdata", "amp_genes.conf_99.txt", package = "maftools")
del.genes <- system.file("extdata", "del_genes.conf_99.txt", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesF
plotGisticResults(gistic = laml.gistic)
```

`plotmafSummary` *Plots maf summary.*

Description

Plots maf summary.

Usage

```
plotmafSummary(maf, file = NULL, rmOutlier = TRUE, dashboard = TRUE,
  titvRaw = TRUE, width = 6, height = 5, addStat = NULL,
  showBarcodes = FALSE, fs = 10, textSize = 2, color = NULL,
  statFontSize = 3, titvColor = NULL, top = 10)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`file`	If given pdf file will be generated.
`rmOutlier`	If TRUE removes outlier from boxplot.
`dashboard`	If FALSE plots simple summary instead of dashboard style.
`titvRaw`	TRUE. If false instead of raw counts, plots fraction.

`width`	plot parameter for output file.
`height`	plot parameter for output file.
`addStat`	Can be either mean or median. Default NULL.
`showBarcodes`	include sample names in the top bar plot.
`fs`	base size for text. Default 10.
`textSize`	font size if showBarcodes is TRUE. Default 2.
`color`	named vector of colors for each Variant_Classification.
`statFontSize`	font size if addStat is used. Default 3.
`titvColor`	colors for SNV classifications.
`top`	include top n genes dashboard plot. Default 10.

Value

Prints plot.

See Also

`read.maf` `MAF`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, useAll = FALSE)
plotmafSummary(maf = laml, addStat = 'median')
```

`plotOncodrive` *Plots results from* `oncodrive`

Description

Takes results from `oncodrive` and plots them as a scatter plot. Size of the gene shows number of clusters (hotspots), x-axis can either be an absolute number of variants accumulated in these clusters or a fraction of total variants found in these clusters. y-axis is fdr values transformed into -log10 for better representation. Labels indicate Gene name with number clusters observed.

Usage

```
plotOncodrive(res = NULL, fdrCutOff = 0.05, useFraction = FALSE,
  colCode = NULL, labelSize = 2)
```

Arguments

`res`	results from `oncodrive`
`fdrCutOff`	fdr cutoff to call a gene as a driver.
`useFraction`	if TRUE uses a fraction of total variants as X-axis scale instead of absolute counts.
`colCode`	Colors to use for indicating significant and non-signififcant genes. Default NULL
`labelSize`	font size for labelling genes. Default 2.

Value

a ggplot object which can be further modified.

See Also

oncodrive

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.sig <- oncodrive(maf = laml, AACol = 'Protein_Change', minMut = 5)
plotOncodrive(res = laml.sig, fdrCutOff = 0.1)
```

---

plotSignatures *Plots decomposed mutational signatures or APOBEC enrichment plot.*

Description

If input is results from extractSignatures plots decomposed mutational signatures as a barplot. If input is results from trinucleotideMatrix plots APOBEC enrichment plot.

Usage

```
plotSignatures(nmfRes = NULL, contributions = FALSE, color = NULL, ...)
```

Arguments

nmfRes	results from extractSignatures or trinucleotideMatrix
contributions	If TRUE plots contribution of signatures in each sample.
color	colors for each Ti/Tv conversion class. Default NULL
...	further plot options passed to barplot

Value

ggplot object if contributions is TRUE

See Also

trinucleotideMatrix

plotTiTv *Plot Transition and Trasnversion ratios.*

Description

Takes results generated from `titv` and plots the Ti/Tv ratios and contributions of 6 mutational conversion classes in each sample.

Usage

```
plotTiTv(res = NULL, plotType = "both", file = NULL, width = 6,
  height = 5, color = NULL, showBarcodes = FALSE, textSize = 2)
```

Arguments

`res`	results generated by `titv`
`plotType`	Can be 'bar', 'box' or 'both'. Defaults to 'both'
`file`	basename for output file name. If given pdf will be generated.
`width`	width of the plot, in inches.
`height`	height of the plot, in inches.
`color`	named vector of colors for each coversion class.
`showBarcodes`	Whether to include sample names for barplot
`textSize`	fontsize if showBarcodes is TRUE. Deafult 2.

Value

None.

See Also

`titv`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.titv = titv(maf = laml, useSyn = TRUE)
plotTiTv(laml.titv)
```

plotVaf — *Plots vaf distribution of genes*

Description

Plots vaf distribution of genes as a boxplot or violinplot.

Usage

```
plotVaf(maf, vafCol = NULL, genes = NULL, violin = FALSE, top = 10,
  orderByMedian = TRUE, flip = FALSE, fn = NULL, width = 6,
  height = 5)
```

Arguments

maf	an MAF object generated by read.maf
vafCol	manually specify column name for vafs. Default looks for column 't_vaf'
genes	specify genes for which plots has to be generated
violin	if TRUE plots violin plot
top	if genes is NULL plots top n number of genes. Defaults to 5.
orderByMedian	Orders genes by decreasing median VAF. Default TRUE
flip	if TRUE, flips axes. Default FALSE
fn	Filename. If given saves plot as a output pdf. Default NULL.
width	Width of plot to be saved. Default 6
height	Height of plot to be saved. Default 5

Value

ggplot object which can be further modified.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
plotVaf(maf = laml, vafCol = 'i_TumorVAF_WU')
```

prepareMutSig *Prepares MAF file for MutSig analysis.*

Description

Corrects gene names for MutSig compatibility.

Usage

```
prepareMutSig(maf, fn = NULL)
```

Arguments

maf	an MAF object generated by read.maf
fn	basename for output file. If provided writes MAF to an output file with the given basename.

Details

MutSig/MutSigCV is most widely used program for detecting driver genes. However, we have observed that covariates files (gene.covariates.txt and exome_full192.coverage.txt) which are bundled with MutSig have non-standard gene names (non Hugo_Symbols). This discrepancy between Hugo_Symbols in MAF and non-Hugo_symbols in covariates file causes MutSig program to ignore such genes. For example, KMT2D - a well known driver gene in Esophageal Carcinoma is represented as MLL2 in MutSig covariates. This causes KMT2D to be ignored from analysis and is represented as an insignificant gene in MutSig results. This function attempts to correct such gene symbols with a manually curated list of gene names compatible with MutSig covariates list.

Value

returns a MAF with gene symbols corrected.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
prepareMutSig(maf = laml)
```

rainfallPlot *Rainfall plot to display hyper mutated genomic regions.*

Description

Plots inter variant distance as a function of genomic locus.

Usage

```
rainfallPlot(maf, tsb = NULL, detectChangePoints = FALSE,
  ref.build = "hg19", color = NULL, savePlot = FALSE, width = 6,
  height = 3, fontSize = 12, pointSize = 1)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`. Required.
`tsb`	specify sample names (Tumor_Sample_Barcodes) for which plotting has to be done. If NULL, draws plot for most mutated sample.
`detectChangePoints`	If TRUE, detectes genomic change points where potential kataegis are formed. Results are written to an output tab delimted file.
`ref.build`	Reference build for chromosome sizes. Can be hg18, hg19 or hg38. Default hg19.
`color`	named vector of colors for each coversion class.
`savePlot`	If TRUE plot is saved to output pdf. Default FALSE.
`width`	width of plot to be saved.
`height`	height of plot to be saved.
`fontSize`	Default 12.
`pointSize`	Default 2.

Details

Note that detected change points are only loci where the distribution of inter-event distance changes. Segments may have to be manually inferred by adjacent change-points.

Value

returns ggplot object of the plot which can be further modified.

`read.maf` *Read MAF files.*

Description

Takes tab delimited MAF (can be plain text or gz compressed) file as an input and summarizes it in various ways. Also creates oncomatrix - helpful for visualization.

Usage

```
read.maf(maf, removeSilent = TRUE, useAll = TRUE,
  gisticAllLesionsFile = NULL, gisticAmpGenesFile = NULL,
  gisticDelGenesFile = NULL, cnTable = NULL,
  removeDuplicatedVariants = TRUE, isTCGA = FALSE, verbose = TRUE)
```

Arguments

`maf`	tab delimited MAF file. File can also be gz compressed. Required. Alternatively, you can also provide already read MAF file as a dataframe.
`removeSilent`	logical. Whether to discard silent (variants with Low/Modifier consequences) mutations ("3'UTR", "5'UTR", "3'Flank", "Targeted_Region", "Silent", "Intron","RNA", "IGR", "Splice_Region", "5'Flank", "lincRNA"). Default is TRUE.

`useAll`	logical. Whether to use all variants irrespective of values in Mutation_Status. Defaults to TRUE. If FALSE, only uses with values Somatic.
`gisticAllLesionsFile`	All Lesions file generated by gistic. e.g; all_lesions.conf_XX.txt, where XX is the confidence level. Default NULL.
`gisticAmpGenesFile`	Amplification Genes file generated by gistic. e.g; amp_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
`gisticDelGenesFile`	Deletion Genes file generated by gistic. e.g; del_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
`cnTable`	Custom copynumber data if gistic results are not available. Input file should a tab seperated three column table containing gene name, Sample name and copy number status (either 'Amp' or 'Del'). Default NULL.
`removeDuplicatedVariants`	removes repeated variants in a particuar sample, mapped to multiple transcripts of same Gene. See Description. Default TRUE.
`isTCGA`	Is input MAF file from TCGA source.
`verbose`	TRUE logical. Default to be talkative and prints summary.

Details

This function takes MAF file as input and summarizes them. If copy number data is available, e.g from GISTIC, it can be provided too via arguments gisticAllLesionsFile, gisticAmpGenesFile, and gisticDelGenesFile. Copy number data can also be provided as a custom table containing Gene name, Sample name and Copy Number status.

Note that if input MAF file contains multiple affected transcripts of a variant, this function by default removes them as duplicates, while keeping single unique entry per variant per sample. If you wish to keep all of them, set removeDuplicatedVariants to FALSE.

FLAGS - If you get a note on possible FLAGS while reading MAF, it means some of the top mutated genes are fishy. These genes are often non-pathogenic and passengers, but are frequently mutated in most of the public exome studies. Examples of such genes include TTN, MUC16, etc. This note can be ignored without any harm, it's only generated as to make user aware of such genes. See references for details on FLAGS.

Value

An object of class MAF.

References

Shyr C, Tarailo-Graovac M, Gottlieb M, Lee JJ, van Karnebeek C, Wasserman WW. FLAGS, frequently mutated genes in public exomes. BMC Med Genomics 2014; 7: 64.

See Also

`plotmafSummary` `write.mafSummary`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
```

readGistic *Read and summarize gistic output.*

Description

A little function to summarize gistic output files. Summarized output is returned as a list of tables.

Usage

```
readGistic(gisticAllLesionsFile, gisticAmpGenesFile = NULL,
  gisticDelGenesFile = NULL, isTCGA = FALSE)
```

Arguments

`gisticAllLesionsFile`	All Lesions file generated by gistic. e.g; all_lesions.conf_XX.txt, where XX is the confidence level. Default NULL.
`gisticAmpGenesFile`	Amplification Genes file generated by gistic. e.g; amp_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
`gisticDelGenesFile`	Deletion Genes file generated by gistic. e.g; del_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
`isTCGA`	Is the data from TCGA. Default FALSE.

Details

Requires output files generated from GISTIC. Gistic documentation can be found here ftp://ftp.broadinstitute.org/pub/GIS

Value

A list of summarized data.

Examples

```
all.lesions <- system.file("extdata", "all_lesions.conf_99.txt", package = "maftools")
amp.genes <- system.file("extdata", "amp_genes.conf_99.txt", package = "maftools")
del.genes <- system.file("extdata", "del_genes.conf_99.txt", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesF
```

subsetMaf *Subset MAF*

Description

Subsets MAF based on given conditions.

Usage

```
subsetMaf(maf, includeSyn = FALSE, tsb = NULL, genes = NULL,
  fields = NULL, query = NULL, mafObj = FALSE, isTCGA = FALSE)
```

Arguments

`maf`	an MAF object generated by `read.maf`
`includeSyn`	to include sysnonymous variants in output
`tsb`	subset by these samples (Tumor Sample Barcodes)
`genes`	subset by these genes
`fields`	include only these fields along with necessary fields in the output
`query`	query string. e.g, "Variant_Classification == 'Missense_Mutation'" returns only Missense variants.
`mafObj`	returns output as MAF class `MAF-class`. Default FALSE
`isTCGA`	Is input MAF file from TCGA source.

Value

subset table or an object of class `MAF-class`

See Also

`getFields`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
##Select all Splice_Site mutations from DNMT3A and NPM1
subsetMaf(maf = laml, genes = c('DNMT3A', 'NPM1'),
query = "Variant_Classification == 'Splice_Site'")
##Select all variants with VAF above 30%
subsetMaf(maf = laml, query = "i_TumorVAF_WU > 30")
##Extract data for samples 'TCGA.AB.3009' and 'TCGA.AB.2933' but only include vaf filed.
subsetMaf(maf = laml, tsb = c('TCGA.AB.3009', 'TCGA.AB.2933'), fields = 'i_TumorVAF_WU')
```

`tcgaCompare` *Compare mutation load against TCGA cohorts*

Description

Compares mutation load in input MAF against all of 33 TCGA cohorts

Usage

```
tcgaCompare(maf, cohortName = NULL, primarySite = FALSE, col = c("gray70",
  "black"), medianCol = "red", fn = NULL, width = 8, height = 5,
  fontSize = 10)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`cohortName`	name for the input MAF cohort. Default "Input"
`primarySite`	If TRUE uses primary site of cancer as labels instead of TCGA project IDs. Default FALSE.
`col`	color vector for length 2 TCGA cohorts and input MAF cohort. Default gray70 and black.
`medianCol`	color for median line. Default red.
`fn`	If provided saves plot to output pdf with basename fn. Default NULL.
`width`	width for output plot
`height`	height of output plot
`fontSize`	base fontsize. Default 10.

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
tcgaCompare(maf = laml, cohortName = "AML")
```

`titv` *Classifies SNPs into transitions and transversions*

Description

takes output generated by read.maf and classifies Single Nucleotide Variants into Transitions and Transversions.

Usage

```
titv(maf, useSyn = FALSE, plot = TRUE, file = NULL)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`useSyn`	Logical. Whether to include synonymous variants in analysis. Defaults to FALSE.
`plot`	plots a titv fractions. default TRUE.
`file`	basename for output file name. If given writes summaries to output file. Default NULL.

Value

list of `data.frames` with Transitions and Transversions summary.

See Also

`plotTiTv`

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
laml.titv = titv(maf = laml, useSyn = TRUE)
```

`trinucleotideMatrix` *Extract single 5' and 3' bases flanking the mutated site for de-novo signature analysis. Also estimates APOBEC enrichment scores.*

Description

Extract single 5' and 3' bases flanking the mutated site for de-novo signature analysis. Also estimates APOBEC enrichment scores.

Usage

```
trinucleotideMatrix(maf, ref_genome, prefix = NULL, add = TRUE,
  ignoreChr = NULL, useSyn = TRUE, fn = NULL)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`ref_genome`	faidx indexed refrence fasta file.
`prefix`	Prefix to add or remove from contig names in MAF file.
`add`	If prefix is used, default is to add prefix to contig names in MAF file. If false prefix will be removed from contig names.
`ignoreChr`	Chromsomes to remove from analysis. e.g. chrM
`useSyn`	Logical. Whether to include synonymous variants in analysis. Defaults to TRUE
`fn`	If given writes APOBEC results to an output file with basename fn. Default NULL.

Details

Extracts immediate 5' and 3' bases flanking the mutated site and classifies them into 96 substitution classes. This function loads reference genome into memeory. Typical human geneome occupies a peak memory of ~3 gb while extracting bases.

APOBEC Enrichment: Enrichment score is calculated using the same method described by Roberts et al.

E = (n_tcw * background_c) / (n_C * background_tcw)

where, n_tcw = number of mutations within T[C>T]W and T[C>G]W context. (W -> A or T)

n_C = number of mutated C and G

background_C and background_tcw motifs are number of C and TCW motifs occuring around +/- 20bp of each mutation.

One-sided Fisher's Exact test is performed to determine the enrichment of APOBEC tcw mutations over background.

Value

list of 2. A matrix of dimension nx96, where n is the number of samples in the MAF and a table describing APOBEC enrichment per sample.

References

Roberts SA, Lawrence MS, Klimczak LJ, et al. An APOBEC Cytidine Deaminase Mutagenesis Pattern is Widespread in Human Cancers. Nature genetics. 2013;45(9):970-976. doi:10.1038/ng.2702.

See Also

`extractSignatures`

Examples

```
## Not run:
laml.tnm <- trinucleotideMatrix(maf = laml, ref_genome = 'hg19.fa',
prefix = 'chr', add = TRUE, useSyn = TRUE)

## End(Not run)
```

`vcr` *Samll internal function to make complex events.*

Description

Samll internal function to make complex events. Ignore this.

Usage

```
vcr(xstr, gis = FALSE)
```

Arguments

`xstr`	charcter to split
`gis`	Is input from gistic. Logical.

Value

split string

`write.GisticSummary` *Writes GISTIC summaries to output tab-delimited text files.*

Description

Writes GISTIC summaries to output tab-delimited text files.

Usage

```
write.GisticSummary(gistic, basename = NULL)
```

Arguments

`gistic`	an object of class `GISTIC` generated by `readGistic`
`basename`	basename for output file to be written.

Value

None. Writes output as tab delimited text files.

See Also

`readGistic`

Examples

```
all.lesions <- system.file("extdata", "all_lesions.conf_99.txt", package = "maftools")
amp.genes <- system.file("extdata", "amp_genes.conf_99.txt", package = "maftools")
del.genes <- system.file("extdata", "del_genes.conf_99.txt", package = "maftools")
laml.gistic <- readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenes
write.GisticSummary(gistic = laml.gistic, basename = 'laml')
```

`write.mafSummary` *Writes maf summaries to output tab-delimited text files.*

Description

Writes maf summaries to output tab-delimited text files.

Usage

```
write.mafSummary(maf, basename = NULL)
```

Arguments

`maf`	an `MAF` object generated by `read.maf`
`basename`	basename for output file to be written.

Value

None. Writes output as text files.

See Also

read.maf

Examples

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf, removeSilent = TRUE, useAll = FALSE)
write.mafSummary(maf = laml, basename = 'laml')
```

Index