

# Package ‘maftools’

April 21, 2026

**Type** Package

**Title** Summarize, Analyze and Visualize MAF Files

**Version** 2.27.0

**Date** 2021-04-30

**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 visualizations with minimal effort.

**License** MIT + file LICENSE

**URL** <https://github.com/PoisonAlien/maftools>

**BugReports** <https://github.com/PoisonAlien/maftools/issues>

**Depends** R (>= 3.3)

**Imports** data.table, grDevices, methods, RColorBrewer, Rhtslib, survival, DNACopy, pheatmap

**Suggests** berryFunctions, Biostrings, BSgenome, BSgenome.Hsapiens.UCSC.hg19, GenomicRanges, IRanges, knitr, mclust, MultiAssayExperiment, NMF, R.utils, RaggedExperiment, rmarkdown, S4Vectors

**LinkingTo** Rhtslib

**VignetteBuilder** knitr

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

**Encoding** UTF-8

**LazyData** TRUE

**NeedsCompilation** no

**RoxygenNote** 7.3.1

**SystemRequirements** GNU make, curl

**git\_url** <https://git.bioconductor.org/packages/maftools>

**git\_branch** devel

**git\_last\_commit** 5abf2c6

**git\_last\_commit\_date** 2025-10-29

**Repository** Bioconductor 3.24

**Date/Publication** 2026-04-20

**Author** Anand Mayakonda [aut, cre] (ORCID:  
<<https://orcid.org/0000-0003-1162-687X>>)

## Contents

annovarToMaf . . . . .	4
bamreadcounts . . . . .	5
cancerhotspots . . . . .	6
cancerhotspotsAggr . . . . .	7
clinicalEnrichment . . . . .	7
coBarplot . . . . .	9
coGisticChromPlot . . . . .	10
compareSignatures . . . . .	12
coOncoplot . . . . .	13
drugInteractions . . . . .	16
estimateSignatures . . . . .	17
extractSignatures . . . . .	18
filterMaf . . . . .	19
forestPlot . . . . .	20
genesToBarcodes . . . . .	21
genotypeMatrix . . . . .	21
getClinicalData . . . . .	22
getCytobandSummary . . . . .	23
getFields . . . . .	23
getGeneSummary . . . . .	24
getSampleSummary . . . . .	25
GISTIC-class . . . . .	25
gisticBubblePlot . . . . .	26
gisticChromPlot . . . . .	27
gisticCompare . . . . .	28
gisticOncoPlot . . . . .	29
gtMarkers . . . . .	31
icgcSimpleMutationToMAF . . . . .	32
inferHeterogeneity . . . . .	33
lollipopPlot . . . . .	34
lollipopPlot2 . . . . .	36
MAF . . . . .	38
MAF-class . . . . .	39
maf2mae . . . . .	40
mafbarplot . . . . .	40
mafCompare . . . . .	41
mafSummary . . . . .	42
mafSurvGroup . . . . .	43
mafSurvival . . . . .	44
math.score . . . . .	45
merge_mafs . . . . .	46
mutCountMatrix . . . . .	46
oncodrive . . . . .	47

oncplot . . . . .	48
oncostrip . . . . .	53
pathways . . . . .	54
pfamDomains . . . . .	55
plotApobecDiff . . . . .	56
plotCBSsegments . . . . .	58
plotClusters . . . . .	59
plotCophenetic . . . . .	60
plotEnrichmentResults . . . . .	60
plotmafSummary . . . . .	61
plotMosdepth . . . . .	63
plotMosdepth_t . . . . .	63
plotOncodrive . . . . .	64
plotPathways . . . . .	65
plotProtein . . . . .	66
plotSignatures . . . . .	67
plotTiTv . . . . .	68
plotVaf . . . . .	69
prepareMutSig . . . . .	70
prepAscat . . . . .	71
prepAscat_t . . . . .	72
rainfallPlot . . . . .	73
read.maf . . . . .	74
readGistic . . . . .	76
sampleSwaps . . . . .	77
segmentLogR . . . . .	78
segSummarize . . . . .	78
setdiffMAF . . . . .	79
signatureEnrichment . . . . .	80
somaticInteractions . . . . .	81
subsetMaf . . . . .	83
survGroup . . . . .	84
tcgaAvailable . . . . .	85
tcgaCompare . . . . .	86
tcgaDriverBP . . . . .	87
tcgaLoad . . . . .	88
titv . . . . .	89
tmb . . . . .	89
trinucleotideMatrix . . . . .	90
vafCompare . . . . .	92
write.GisticSummary . . . . .	93
write.mafSummary . . . . .	93

---

annovarToMaf	<i>Converts annovar annotations into MAF.</i>
--------------	---

---

### Description

Converts variant annotations from Annovar into a basic MAF.

### Usage

```
annovarToMaf(
  annovar,
  Center = NULL,
  refBuild = "hg19",
  tsbCol = NULL,
  table = "refGene",
  ens2hugo = TRUE,
  basename = NULL,
  sep = "\t",
  MAFobj = FALSE,
  sampleAnno = NULL
)
```

### Arguments

annovar	input annovar annotation file. Can be vector of multiple files.
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'
ens2hugo	If 'table' is 'ensGene', setting this argument to 'TRUE' converts all ensemble IDs to hugo symbols.
basename	If provided writes resulting MAF file to an output file.
sep	field separator for input file. Default tab separated.
MAFobj	If TRUE, returns results as an <a href="#">MAF</a> object.
sampleAnno	annotations associated with each sample/Tumor_Sample_Barcode in input annovar file. If provided it will be included in MAF object. Could be a text file or a data.frame. Ideally annotation would contain clinical data, survival information and other necessary features associated with samples. Default NULL.

### 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')
```

---

bamreadcounts	<i>extract nucleotide counts for targeted variants from the BAM file.</i>
---------------	---

---

### Description

Given a BAM file and target loci, 'bamreadcounts' fetches readcounts for A, T, G, C, Ins, and Del. Function name is an homage to <https://github.com/genome/bam-readcount>

### Usage

```
bamreadcounts(
  bam = NULL,
  loci = NULL,
  zerobased = FALSE,
  mapq = 10,
  sam_flag = 1024,
  op = NULL,
  fa = NULL,
  nthreads = 4
)
```

### Arguments

bam	Input bam file(s). Required.
loci	Loci file. Can be a tsv file or a data.frame. First two columns should contain chromosome and position (by default assumes coordinates are 1-based)
zerobased	are coordinates zero-based. Default FALSE.
mapq	Map quality. Default 10
sam_flag	SAM FLAG to filter reads. Default 1024
op	Output file basename. Default parses from BAM file
fa	Indexed fasta file. If provided, extracts and adds reference base to the output tsv.
nthreads	Number of threads to use. Each BAM file will be launched on a separate thread. Works only on Unix and macOS.

cancerhotspots

*Genotype known cancer hotspots from the tumor BAM file***Description**

‘cancerhotspots’ allows rapid genotyping of known somatic variants from the tumor BAM files. This facilitates to get a quick overlook of known somatic hot-spots in a matter of minutes, without spending hours on variant calling and annotation. In simple words, it fetches nucleotide frequencies of known somatic hotspots and prioritizes them based on allele frequency. Output includes a browsable/sharable HTML report of candidate variants. Known cancerhotspots for both GRCh37 and GRCh38 assemblies (3180 variants) are included. This should be sufficient and cover most of the known driver genes/events. See Reference for details.

**Usage**

```
cancerhotspots(
  bam = NULL,
  refbuild = "GRCh37",
  mapq = 10,
  sam_flag = 1024,
  vaf = 0.05,
  t_depth = 30,
  t_alt_count = 8,
  op = NULL,
  fa = NULL,
  browse = FALSE
)
```

**Arguments**

bam	Input bam file. Required.
refbuild	Default "GRCh37". Can be "GRCh37", "GRCh38", "hg19", "hg38"
mapq	Map quality. Default 10
sam_flag	SAM FLAG to filter reads. Default 1024
vaf	VAF threshold. Default 0.05 [Variant filter]
t_depth	Depth of coverage threshold. Default 30 [Variant filter]
t_alt_count	Min. number of reads supporting tumor allele . Default 8 [Variant filter]
op	Output file basename. Default parses from BAM file
fa	Indexed fasta file. If provided, extracts and adds reference base to the output tsv.
browse	If TRUE opens the html file in browser

**References**

Chang MT, Asthana S, Gao SP, et al. Identifying recurrent mutations in cancer reveals widespread lineage diversity and mutational specificity. *Nat Biotechnol.* 2016;34(2):155-163. doi:10.1038/nbt.3391

**See Also**

[cancerhotspotsAggr](#)

---

cancerhotspotsAggr     *Aggregate cancerhotspots reports*

---

### Description

Takes tsv files generated by [cancerhotspots](#) and aggregates them into an MAF for downstream analysis

### Usage

```
cancerhotspotsAggr(  
  tsvs = NULL,  
  minVaf = 0.02,  
  minDepth = 15,  
  sampleNames = NULL,  
  maf = TRUE,  
  ...  
)
```

### Arguments

tsvs	TSV files generated by <a href="#">cancerhotspots</a>
minVaf	Min. VAF threshold. Default 0.02
minDepth	Min. depth of coverage. Default 15
sampleNames	samples for each tsv file. Default NULL. Parses from file names.
maf	Return as an MAF object. Default TRUE.
...	Additional arguments passed to <a href="#">read.maf</a> if 'maf' is TRUE.

### Value

[MAF](#) object

### See Also

[cancerhotspots](#)

---

clinicalEnrichment     *Performs mutational enrichment analysis for a given clinical feature.*

---

### Description

Performs pairwise and groupwise fisher exact tests to find differentially enriched genes for every factor within a clinical feature.

## Usage

```
clinicalEnrichment(  
  maf,  
  clinicalFeature = NULL,  
  annotationDat = NULL,  
  minMut = 5,  
  useCNV = TRUE,  
  pathways = FALSE  
)
```

## Arguments

maf	MAF object
clinicalFeature	columns names from 'clinical.data' slot of MAF to be analysed for.
annotationDat	If MAF file was read without clinical data, provide a custom data.frame or a tsv file with a column containing Tumor_Sample_Barcodes along with clinical features. Default NULL.
minMut	Consider only genes with minimum this number of samples mutated. Default 5.
useCNV	whether to include copy number events if available. Default TRUE. Not applicable when 'pathways = TRUE'
pathways	Summarize genes by pathways before comparing. Default 'FALSE'

## Details

Performs fishers test on 2x2 contingency table for WT/Mutants in group of interest vs rest of the sample. Odds Ratio indicate the odds of observing mutant in the group of interest compared to wild-type

## Value

result list containing p-values

## See Also

[plotEnrichmentResults](#)

## Examples

```
## Not run:  
lam1.maf = system.file('extdata', 'tcga_lam1.maf.gz', package = 'maftools')  
lam1.clin = system.file('extdata', 'tcga_lam1_annot.tsv', package = 'maftools')  
lam1 = read.maf(maf = lam1.maf, clinicalData = lam1.clin)  
clinicalEnrichment(lam1, 'FAB_classification')  
  
## End(Not run)
```

---

`coBarplot`*Draw two barplots side by side for cohort comparison.*

---

**Description**

Draw two barplots side by side for cohort comparison.

**Usage**

```
coBarplot(  
  m1,  
  m2,  
  genes = NULL,  
  orderBy = NULL,  
  m1Name = NULL,  
  m2Name = NULL,  
  colors = NULL,  
  normalize = TRUE,  
  yLims = NULL,  
  borderCol = "gray",  
  titleSize = 1,  
  geneSize = 0.8,  
  showPct = TRUE,  
  pctSize = 0.7,  
  axisSize = 0.8,  
  showLegend = TRUE,  
  legendTxtSize = 1,  
  geneMar = 4  
)
```

**Arguments**

<code>m1</code>	first <a href="#">MAF</a> object
<code>m2</code>	second <a href="#">MAF</a> object
<code>genes</code>	genes to be drawn. Default takes top 5 mutated genes.
<code>orderBy</code>	Order genes by mutation rate in 'm1' or 'm2'. Default 'NULL', keeps the same order of 'genes'
<code>m1Name</code>	optional name for first cohort
<code>m2Name</code>	optional name for second cohort
<code>colors</code>	named vector of colors for each <code>Variant_Classification</code> .
<code>normalize</code>	Default TRUE.
<code>yLims</code>	Default NULL. Auto estimates. Maximum values for 'm1' and 'm2' respectively
<code>borderCol</code>	Default gray
<code>titleSize</code>	Default 1
<code>geneSize</code>	Default 0.8
<code>showPct</code>	Default TRUE

pctSize	Default 0.7
axisSize	Default 0.8
showLegend	Default TRUE.
legendTxtSize	Default 0.8
geneMar	Default 4

### Details

Draws two barplots 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
coBarplot(m1 = primary.apl, m2 = relapse.apl, m1Name = 'Primary APL', m2Name = 'Relapse APL')
dev.off()
```

---

coGisticChromPlot      *Co-plot version of gisticChromPlot()*

---

### Description

Use two GISTIC object or/and two MAF objects to view a vertical arranged version of Gistic Chromosome plot results on the Amp or Del G-scores.

### Usage

```
coGisticChromPlot(
  gistic1 = NULL,
  gistic2 = NULL,
  g1Name = "",
  g2Name = "",
  type = "Amp",
  markBands = TRUE,
  labelGenes = TRUE,
  gLims = NULL,
  maf1 = NULL,
  maf2 = NULL,
  mutGenes = NULL,
  mutGenes1 = NULL,
  mutGenes2 = NULL,
  fdrCutOff = 0.05,
```

```

    symmetric = TRUE,
    color = NULL,
    ref.build = "hg19",
    cytobandOffset = "auto",
    txtSize = 0.8,
    cytobandTxtSize = 1,
    mutGenesTxtSize = 0.6,
    rugTickSize = 0.1
)

```

### Arguments

<code>gistic1</code>	first GISTIC object
<code>gistic2</code>	second GISTIC object
<code>g1Name</code>	the title of the left side
<code>g2Name</code>	the title of the right side
<code>type</code>	default 'Amp', c('Amp', "Del"), choose one to plot, only focal events are shown, 'Amp' only shows the Amplification events, and 'Del' only shows the Deletion events.
<code>markBands</code>	default TRUE, integer of length 1 or 2 or TRUE, mark cytoband names of the outer side of the plot
<code>labelGenes</code>	if you want to label some genes you are interested along the chromosome, set it to TRUE
<code>gLims</code>	Controls the G-score's axis limits. Default NULL.
<code>maf1, maf2</code>	if <code>labelGenes==TRUE</code> , you need to provide <code>MAF</code> object, the genes mutation info collected from the <code>maf1</code> is shown on the left side, while <code>maf2</code> on the right side. the genes selected are controlled by the <code>mutGenes</code> or <code>mutGenes1</code> or <code>mutGenes2</code> parameter, see following.
<code>mutGenes, mutGenes1, mutGenes2</code>	default NULL, could be NULL, number, or character vector of gene symbols which match the corresponding <code>MAF</code> object's <code>Hugo_Symbol</code> column values. <code>mutGenes</code> controls both sides of the annotation, <code>mutGenes1</code> controls only left side and corresponding data is extracted from <code>maf1</code> , and <code>mutGenes2</code> controls only right side annotation and corresponding to <code>maf2</code> . If 'NULL', extract the top 50 mutated genes from <code>maf1</code> and <code>maf2</code> seperately then annotate them on the left side ( <code>maf1</code> genes) and right side ( <code>maf2</code> genes). if integer, say <code>N</code> , only top <code>N</code> genes will be extracted seperately from <code>maf1</code> and <code>maf2</code> . These two condition leads to different genes annotated on both sides. If character vector, then the genes have mutated in <code>maf1</code> and <code>maf2</code> will be annotated on both side of the figure which mean the two sides have the same list of genes. if <code>mutGenes</code> is not NULL and both <code>mutGenes1</code> and <code>mutGenes2</code> are NULL, then the auto set <code>mutGenes1 = mutGenes2 = mutGenes</code> .
<code>fdrCutOff</code>	default 0.05, only items with <code>FDR &lt; fdrCutOff</code> will be colored as Amp or Del (colored 'Red' or 'Blue'), others will be seen as non-significant events (colored gray)
<code>symmetric</code>	default TRUE, If False, when the <code>gistic1</code> and <code>gistic2</code> have different max values of G-scores, the Chrom (0 point of x axis) will not be in the center of the whole plot, if you set <code>symmetric==TRUE</code> , then the one with smaller max(G-score) will be stretched larger to make the 0 of the x axis in the middle which eventually make the plot more symmetric.



**Arguments**

nmfRes	results from <a href="#">extractSignatures</a>
sig_db	can be legacy, SBS, SBS_v34. Default SBS_v34
verbose	Default TRUE

**Details**

SBS signature database was obtained from <https://www.synapse.org/#!/Synapse:syn11738319.7>

**Value**

list containing cosine similarities, aetiologies if available, and best match.

**See Also**

[trinucleotideMatrix](#) [extractSignatures](#) [plotSignatures](#)

---

coOncoplot

*Draw two oncoplots side by side for cohort comparison.*

---

**Description**

Draw two oncoplots side by side for cohort comparison.

**Usage**

```
coOncoplot(
  m1,
  m2,
  genes = NULL,
  m1Name = NULL,
  m2Name = NULL,
  clinicalFeatures1 = NULL,
  clinicalFeatures2 = NULL,
  annotationColor1 = NULL,
  annotationColor2 = NULL,
  annotationFontSize = 1.2,
  sortByM1 = FALSE,
  sortByM2 = FALSE,
  sortByAnnotation1 = FALSE,
  annotationOrder1 = NULL,
  sortByAnnotation2 = FALSE,
  annotationOrder2 = NULL,
  sampleOrder1 = NULL,
  sampleOrder2 = NULL,
  additionalFeature1 = NULL,
  additionalFeaturePch1 = 20,
  additionalFeatureCol1 = "white",
  additionalFeatureCex1 = 0.9,
  additionalFeature2 = NULL,
```

```

additionalFeaturePch2 = 20,
additionalFeatureCol2 = "white",
additionalFeatureCex2 = 0.9,
sepwd_genes1 = 0.5,
sepwd_samples1 = 0.5,
sepwd_genes2 = 0.5,
sepwd_samples2 = 0.5,
colors = NULL,
removeNonMutated = TRUE,
anno_height = 2,
legend_height = 4,
geneNamefont = 0.8,
showSampleNames = FALSE,
SampleNamefont = 0.5,
barcode_mar = 1,
outer_mar = 3,
gene_mar = 1,
legendFontSize = 1.2,
titleFontSize = 1.5,
keepGeneOrder = FALSE,
bgCol = "#ecf0f1",
borderCol = "white"
)

```

### Arguments

m1	first <a href="#">MAF</a> object
m2	second <a href="#">MAF</a> object
genes	draw these genes. Default plots top 5 mutated genes from two cohorts.
m1Name	optional name for first cohort
m2Name	optional name for second cohort
clinicalFeatures1	columns names from ‘clinical.data’ slot of m1 MAF to be drawn in the plot. Default NULL.
clinicalFeatures2	columns names from ‘clinical.data’ slot of m2 MAF to be drawn in the plot. Default NULL.
annotationColor1	list of colors to use for ‘clinicalFeatures1’ Default NULL.
annotationColor2	list of colors to use for ‘clinicalFeatures2’ Default NULL.
annotationFontSize	font size for annotations Default 1.2
sortByM1	sort by mutation frequency in ‘m1’
sortByM2	sort by mutation frequency in ‘m2’
sortByAnnotation1	logical sort oncomatrix (samples) by provided ‘clinicalFeatures1’. Sorts based on first ‘clinicalFeatures1’. Defaults to FALSE. column-sort
annotationOrder1	Manually specify order for annotations for ‘clinicalFeatures1’. Works only for first value. Default NULL.

sortByAnnotation2 same as above but for m2  
 annotationOrder2 Manually specify order for annotations for 'clinicalFeatures2'. Works only for first value. Default NULL.  
 sampleOrder1 Manually specify sample names in m1 for oncolplot ordering. Default NULL.  
 sampleOrder2 Manually specify sample names in m2 for oncolplot ordering. Default NULL.  
 additionalFeature1 a vector of length two indicating column name in the MAF and the factor level to be highlighted.  
 additionalFeaturePch1 Default 20  
 additionalFeatureCol1 Default "white"  
 additionalFeatureCex1 Default 0.9  
 additionalFeature2 a vector of length two indicating column name in the MAF and the factor level to be highlighted.  
 additionalFeaturePch2 Default 20  
 additionalFeatureCol2 Default "white"  
 additionalFeatureCex2 Default 0.9  
 sepwd\_genes1 Default 0.5  
 sepwd\_samples1 Default 0.5  
 sepwd\_genes2 Default 0.5  
 sepwd\_samples2 Default 0.5  
 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.  
 anno\_height Height of clinical margin. Default 2  
 legend\_height Height of legend margin. Default 4  
 geneNamefont font size for gene names. Default 1  
 showSampleNames whether to show sample names. Default FALSE.  
 SampleNamefont font size for sample names. Default 0.5  
 barcode\_mar Margin width for sample names. Default 1  
 outer\_mar Margin width for outer. Default 3  
 gene\_mar Margin width for gene names. Default 1  
 legendFontSize font size for legend. Default 1.2  
 titleFontSize font size for title. Default 1.5  
 keepGeneOrder force the resulting plot to use the order of the genes as specified. Default FALSE  
 bgCol Background grid color for wild-type (not-mutated) samples. Default gray - "#CCCCCC"  
 borderCol border grid color for wild-type (not-mutated) samples. Default 'white'

**Details**

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

**Value**

Invisibly returns a list of sample names in their order of occurrences in M1 and M2 respectively.

**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()
```

---

 drugInteractions

*Drug-Gene Interactions*


---

**Description**

Checks for drug-gene interactions and druggable categories

**Usage**

```
drugInteractions(
  maf,
  top = 20,
  genes = NULL,
  plotType = "bar",
  drugs = FALSE,
  fontSize = 0.8
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
top	Top number genes to check for. Default 20
genes	Manually specify gene list
plotType	Can be bar, pie Default bar plot.
drugs	Check for known/reported drugs. Default FALSE
fontSize	Default 0.8

**Details**

This function takes a list of genes and checks for known/reported drug-gene interactions or Drug-gene categories. All gene-drug interactions and drug claims are compiled from Drug Gene Interaction Database. See reference for details and cite it if you use this function.

## References

Griffith, M., Griffith, O. L., Coffman, A. C., Weible, J. V., McMichael, J. F., Spies, N. C., et. al., 2013. DGIdb - Mining the druggable genome. Nature Methods.

## Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
drugInteractions(maf = lam1)
```

---

estimateSignatures	<i>Estimate number of signatures based on cophenetic correlation metric</i>
--------------------	---

---

## Description

Estimate number of signatures based on cophenetic correlation metric

## Usage

```
estimateSignatures(  
  mat,  
  nMin = 2,  
  nTry = 6,  
  nrun = 10,  
  parallel = 4,  
  pConstant = NULL,  
  verbose = TRUE,  
  plotBestFitRes = FALSE  
)
```

## Arguments

mat	Input matrix of diemnsion nx96 generated by <a href="#">trinucleotideMatrix</a>
nMin	Minimum number of signatures to try. Default 2.
nTry	Maximum number of signatures to try. Default 6.
nrun	numeric giving the number of run to perform for each value in range. Default 5
parallel	Default 4. Number of cores to use.
pConstant	A small positive value to add to the matrix. Use it ONLY if the functions throws an non-conformable arrays error
verbose	Default TRUE
plotBestFitRes	plots consensus heatmap for range of values tried. Default FALSE

## 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 NMF .rank object and summary stats.

**See Also**

[plotCophenetic](#) [extractSignatures](#) [trinucleotideMatrix](#)

**Examples**

```
## Not run:
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
laml.tnm <- trinucleotideMatrix(maf = laml, ref_genome = 'BSgenome.Hsapiens.UCSC.hg19', prefix = 'chr',
add = TRUE, useSyn = TRUE)
library("NMF")
laml.sign <- estimateSignatures(mat = laml.tnm, plotBestFitRes = FALSE, nMin = 2, nTry = 3, nrun = 2, pConstant = 1e-05)

## End(Not run)
```

---

extractSignatures	<i>Extract mutational signatures from trinucleotide context.</i>
-------------------	--

---

**Description**

Decompose a matrix of 96 substitution classes into n signatures.

**Usage**

```
extractSignatures(
  mat,
  n = NULL,
  plotBestFitRes = FALSE,
  parallel = 4,
  pConstant = NULL
)
```

**Arguments**

mat	Input matrix of diemnsion nx96 generated by <a href="#">trinucleotideMatrix</a>
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.
plotBestFitRes	plots consensus heatmap for range of values tried. Default FALSE
parallel	Default 4. Number of cores to use.
pConstant	A small positive value to add to the matrix. Use it ONLY if the functions throws an non-conformable arrays error

**Details**

This function decomposes a non-negative matrix into n signatures.

**Value**

a list with decomposed scaled signatures, signature contributions in each sample and NMF object.

**See Also**

[trinucleotideMatrix](#) [plotSignatures](#) [compareSignatures](#)

**Examples**

```
## Not run:
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
laml.tnm <- trinucleotideMatrix(maf = laml, ref_genome = 'BSgenome.Hsapiens.UCSC.hg19', prefix = 'chr',
add = TRUE, useSyn = TRUE)
library("NMF")
laml.sign <- extractSignatures(mat = laml.tnm, plotBestFitRes = FALSE, n = 2, pConstant = 0.01)

## End(Not run)
```

---

filterMaf

*Filter MAF objects*

---

**Description**

Filter MAF by genes or samples

**Usage**

```
filterMaf(maf, genes = NULL, tsb = NULL, isTCGA = FALSE)
```

**Arguments**

maf	an MAF object generated by <a href="#">read.maf</a>
genes	remove these genes
tsb	remove these samples (Tumor Sample Barcodes)
isTCGA	FALSE

**Value**

Filtered object of class [MAF-class](#)

**See Also**

[subsetMaf](#)

**Examples**

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
#get rid of samples of interest
filterMaf(maf = laml, tsb = c("TCGA-AB-2830", "TCGA-AB-2804"))
#remove genes of intrest
filterMaf(maf = laml, genes =c("TTN", "AHNAK2"))
```

---

`forestPlot`*Draw forest plot for differences between cohorts.*

---

### Description

Draw forest plot for differences between cohorts.

### Usage

```
forestPlot(  
  mafCompareRes,  
  pVal = 0.05,  
  fdr = NULL,  
  color = c("maroon", "royalblue"),  
  geneFontSize = 0.8,  
  titleSize = 1.2,  
  lineWidth = 1  
)
```

### Arguments

<code>mafCompareRes</code>	results from <a href="#">mafCompare</a>
<code>pVal</code>	p-value threshold. Default 0.05.
<code>fdr</code>	fdr threshold. Default NULL. If provided uses adjusted pvalues (fdr).
<code>color</code>	vector of two colors for the lines. Default 'maroon' and 'royalblue'
<code>geneFontSize</code>	Font size for gene symbols. Default 0.8
<code>titleSize</code>	font size for titles. Default 1.2
<code>lineWidth</code>	line width for CI bars. Default 1

### 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

Nothing

### 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.ap1, m2 = relapse.ap1, m1Name = 'Primary',
m2Name = 'Relapse', minMut = 5)
forestPlot(mafCompareRes = pt.vs.rt)
```

---

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, verbose = TRUE)
```

### Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
genes	Hugo_Symbol for which sample names to be extracted.
justNames	if TRUE, just returns samples names instead of summarized tables.
verbose	Default TRUE

### Value

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

### Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
genesToBarcodes(maf = lam1, genes = 'DNMT3A')
```

---

genotypeMatrix      *Creates a Genotype Matrix for every variant*

---

### Description

Creates a Genotype matrix using allele frequencies or by mutation status.

### Usage

```
genotypeMatrix(
  maf,
  genes = NULL,
  tsb = NULL,
  includeSyn = FALSE,
  vafCol = NULL,
  vafCutoff = c(0.1, 0.75)
)
```

**Arguments**

maf	an MAF object generated by <code>read.maf</code>
genes	create matrix for only these genes. Define NULL
tsb	create matrix for only these tumor sample barcodes/samples. Define NULL
includeSyn	whether to include silent mutations. Default FALSE
vafCol	specify column name for vaf's. Default NULL. If not provided simply assumes all mutations are heterozygous.
vafCutoff	specify minimum and maximum vaf to define mutations as heterozygous. Default range 0.1 to 0.75. Mutations above maximum vafs are defined as homozygous.

**Value**

matrix

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools")
lam1 <- read.maf(maf = lam1.maf)
genotypeMatrix(maf = lam1, genes = "RUNX1")
```

---

getClinicalData	<i>extract annotations from MAF object</i>
-----------------	--

---

**Description**

extract annotations from MAF object

**Usage**

```
getClinicalData(x)

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

**Arguments**

x                    An object of class MAF

**Value**

annotations associated with samples in MAF

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools")
lam1 <- read.maf(maf = lam1.maf)
getClinicalData(x = lam1)
```

---

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**

summarized 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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
lam1.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
getCytobandSummary(lam1.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)
getFields(x = laml)
```

---

getGeneSummary	<i>extract gene summary from MAF or GISTIC object</i>
----------------	---

---

**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)
getGeneSummary(laml)
```

---

getSampleSummary	<i>extract sample summary from MAF or GISTIC object</i>
------------------	---

---

**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)
getSampleSummary(x = laml)
```

---

GISTIC-class	<i>Class GISTIC</i>
--------------	---------------------

---

**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  
 gis.scores   gistic.scores  
 summary   table with basic GISTIC summary stats  
 classCode   mapping between numeric values in numericMatrix and copy number events.

**See Also**

[getGeneSummary](#) [getSampleSummary](#) [getCytobandSummary](#)

---

gisticBubblePlot	<i>Plot gistic results as a bubble plot</i>
------------------	---

---

**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  $-\log_{10}$  transformed q values.

**Usage**

```
gisticBubblePlot(
  gistic = NULL,
  color = NULL,
  markBands = NULL,
  fdrCutoff = 0.1,
  log_y = TRUE,
  txtSize = 3
)
```

**Arguments**

gistic	an object of class GISTIC generated by readGistic
color	colors for Amp and Del events.
markBands	any cytobands to label. Can be cytoband labels, or number of top bands to highlight. Default top 5 lowest q values.
fdrCutoff	fdr cutoff to use. Default 0.1
log_y	log10 scale y-axis (# genes affected). Default TRUE
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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
gisticBubblePlot(gistic = laml.gistic, markBands = "")
```

---

`gisticChromPlot`      *Plot gistic results along linearized chromosome*

---

### Description

A genomic plot with segments highlighting significant Amplifications and Deletion regions.

### Usage

```
gisticChromPlot(
  gistic = NULL,
  fdrCutoff = 0.1,
  markBands = NULL,
  color = NULL,
  ref.build = "hg19",
  cytobandOffset = 0.01,
  txtSize = 0.8,
  cytobandTxtSize = 0.6,
  maf = NULL,
  mutGenes = NULL,
  y_lims = NULL,
  mutGenesTxtSize = 0.6
)
```

### Arguments

<code>gistic</code>	an object of class GISTIC generated by <code>readGistic</code>
<code>fdrCutoff</code>	fdr cutoff to use. Default 0.1
<code>markBands</code>	any cytobands to label. Default top 5 lowest q values.
<code>color</code>	colors for Amp and Del events.
<code>ref.build</code>	reference build. Could be hg18, hg19 or hg38.
<code>cytobandOffset</code>	if scores.gistic file is given use this to adjust cytoband size.
<code>txtSize</code>	label size for labels
<code>cytobandTxtSize</code>	label size for cytoband
<code>maf</code>	an optional maf object
<code>mutGenes</code>	mutated genes from maf object to be highlighted
<code>y_lims</code>	Default NULL. A vector upper and lower y-axis limits
<code>mutGenesTxtSize</code>	Default 0.6

### 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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
gisticChromPlot(laml.gistic)

```

---

gisticCompare	<i>compare two GISTIC objects</i>
---------------	-----------------------------------

---

**Description**

compare two GISTIC objects

**Usage**

```

gisticCompare(
  g1,
  g2,
  g1Name = NULL,
  g2Name = NULL,
  minEvent = 5,
  pseudoCount = FALSE
)

```

**Arguments**

g1	first <b>GISTIC</b> object
g2	second <b>GISTIC</b> object
g1Name	optional name for first cohort
g2Name	optional name for second cohort
minEvent	Consider only cytobands with minimum this number of samples altered in at least one of the cohort for analysis. Helpful to ignore single mutated genes. Default 5.
pseudoCount	If TRUE, adds 1 to the contingency table with 0's to avoid 'Inf' values in the estimated odds-ratio.

**Details**

Performs fisher test on 2x2 contingency table generated from two GISTIC objects

**Value**

result list

**See Also**

[forestPlot](#)  
[lollipopPlot2](#)

---

gisticOncoPlot	<i>Plot gistic results.</i>
----------------	-----------------------------

---

### Description

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

### Usage

```
gisticOncoPlot(
  gistic = NULL,
  top = NULL,
  bands = NULL,
  showTumorSampleBarcodes = FALSE,
  gene_mar = 5,
  barcode_mar = 6,
  right_mar = 2.5,
  sepwd_genes = 0.5,
  sepwd_samples = 0.25,
  clinicalData = NULL,
  clinicalFeatures = NULL,
  sortByAnnotation = FALSE,
  sampleOrder = NULL,
  annotationColor = NULL,
  bandsToIgnore = NULL,
  removeNonAltered = TRUE,
  colors = NULL,
  SampleNamefontSize = 0.6,
  fontSize = 0.8,
  legendFontSize = 1.2,
  annotationFontSize = 1.2,
  borderCol = "white",
  bgCol = "#CCCCCC"
)
```

### Arguments

gistic	an <a href="#">GISTIC</a> object generated by <a href="#">readGistic</a>
top	how many top cytobands to be drawn. defaults to all.
bands	draw oncoplot for these bands. Default NULL.
showTumorSampleBarcodes	logical to include sample names.
gene_mar	Default 5
barcode_mar	Default 6
right_mar	Default 2.5
sepwd_genes	Default 0.5
sepwd_samples	Default 0.25
clinicalData	data.frame with columns containing Tumor_Sample_Barcodes and rest of columns with annotations.

clinicalFeatures	columns names from 'clinicalData' to be drawn in the plot. Default NULL.
sortByAnnotation	logical sort oncomatrix (samples) by provided 'clinicalFeatures'. Defaults to FALSE. column-sort
sampleOrder	Manually specify sample names for oncolplot ordering. Default NULL.
annotationColor	list of colors to use for clinicalFeatures. Default NULL.
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.
SampleNamefontSize	font size for sample names. Default 0.6
fontSize	font size for cytoband names. Default 0.8
legendFontSize	font size for legend. Default 1.2
annotationFontSize	font size for annotations. Default 1.2
borderCol	Default "white"
bgCol	Default "#CCCCCC"

## 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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
lam1.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
gisticOncoPlot(lam1.gistic)
```

gtMarkers

*Extract read counts from genetic markers for ASCAT analysis***Description**

The function will generate tsv files ‘<tumor/normal>\_nucleotide\_counts.tsv’ that can be used for downstream analysis. Note that the function will process ~900K loci from Affymetrix Genome-Wide Human SNP 6.0 Array. The process can be sped up by increasing ‘nthreads’ which will launch each chromosome on a separate thread. Currently hg19 and hg38 are supported. Files need to be further processed with [prepAscat](#) for tumor-normal pair, or [prepAscat\\_t](#) for tumor only samples.

**Usage**

```
gtMarkers(
  t_bam = NULL,
  n_bam = NULL,
  build = "hg19",
  prefix = NULL,
  add = TRUE,
  mapq = 10,
  sam_flag = 1024,
  loci = NULL,
  fa = NULL,
  op = NULL,
  zerobased = FALSE,
  nthreads = 4,
  verbose = TRUE
)
```

**Arguments**

t_bam	Tumor BAM file. Required
n_bam	Normal BAM file. Recommended
build	Default hg19. Mutually exclusive with ‘loci’. Currently supported ‘hg19’ and ‘hg38’ and includes ca. 900K SNPs from Affymetrix Genome-Wide Human SNP 6.0 Array. SNP file has no ‘chr’ prefix.
prefix	Prefix to add or remove from contig names in loci file. For example, in case BAM files have ‘chr’ prefix, set prefix = ‘chr’
add	If prefix is used, default is to add prefix to contig names in loci file. If false prefix will be removed from contig names.
mapq	Minimum mapping quality. Default 10
sam_flag	SAM FLAG to filter reads. Default 1024
loci	A tab separated file with chr and position. If not available use ‘build’ argument.
fa	Indexed fasta file. If provided, extracts and adds reference base to the output tsv.
op	Output file basename. Default parses from BAM file
zerobased	are coordinates zero-based. Default FALSE. Use only if ‘loci’ is used.
nthreads	Number of threads to use. Default 4. Each chromosome will be launched on a separate thread. Works only on Unix and macOS.
verbose	Default TRUE

**See Also**

[prepAscat prepAscat\\_t segmentLogR](#)

---

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 specifications, 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 resulting 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,
  clinicalData = NULL,
  removeDuplicatedVariants = TRUE,
  addHugoSymbol = FALSE
)
```

**Arguments**

<code>icgc</code>	Input data in ICGC Simple Somatic Mutation format. Can be gz compressed.
<code>basename</code>	If given writes to output file with <code>basename</code> .
<code>MAFobj</code>	If <code>TRUE</code> returns results as an <a href="#">MAF</a> object.
<code>clinicalData</code>	Clinical data associated with each sample/Tumor_Sample_Barcode in MAF. Could be a text file or a <code>data.frame</code> . Default <code>NULL</code> .
<code>removeDuplicatedVariants</code>	removes repeated variants in a particular sample, mapped to multiple transcripts of same Gene. See Description. Default <code>TRUE</code> .
<code>addHugoSymbol</code>	If <code>TRUE</code> replaces ensemble gene IDs with <code>Hugo_Symbols</code> . Default <code>FALSE</code> .

**Details**

ICGC Simple Somatic Mutation format specification 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 = "maftools")
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,
  segFile = NULL,
  ignChr = NULL,
  minVaf = 0,
  maxVaf = 1,
  useSyn = FALSE,
  dirichlet = FALSE
)
```

**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'
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 chromosomes 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)
useSyn	Use synonymous variants. Default FALSE.
dirichlet	Deprecated! No longer supported. uses nonparametric dirichlet process for clustering. Default FALSE - uses finite mixture models.

**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 infinite 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**

```
## Not run:
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
TCGA.AB.2972.clust <- inferHeterogeneity(maf = lam1, tsb = 'TCGA-AB-2972', vafCol = 'i_TumorVAF_WU')

## End(Not run)
```

---

lollipopPlot

---

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


---

**Description**

Draws lollipop plot of amino acid changes. Protein domains are derived from PFAM database.

**Usage**

```
lollipopPlot(
  maf,
  data = NULL,
  gene = NULL,
  AACol = NULL,
  labelPos = NULL,
  labPosSize = 0.9,
  showMutationRate = TRUE,
  showDomainLabel = TRUE,
  cBioPortal = FALSE,
  refSeqID = NULL,
```

```

proteinID = NULL,
roundedRect = TRUE,
repel = FALSE,
collapsePosLabel = TRUE,
showLegend = TRUE,
legendTxtSize = 0.8,
labPosAngle = 0,
domainLabelSize = 0.8,
axisTextSize = c(1, 1),
printCount = FALSE,
colors = NULL,
domainAlpha = 1,
domainBorderCol = "black",
bgBorderCol = "black",
labelOnlyUniqueDoamins = TRUE,
defaultYaxis = FALSE,
titleSize = c(1.2, 1),
pointSize = 1.5
)

```

### Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
data	Provide a custom two column data frame with pos and counts instead of an <a href="#">MAF</a> . Input data can also contain an additional column 'Variant_Classification' used for color coding the dots.
gene	HGNC symbol for which protein structure to be drawn.
AACol	manually specify column name for amino acid changes. Default looks for fields 'HGVS_Short', 'AACChange' or 'Protein_Change'. Changes can be of any format i.e, can be a numeric value or HGVS annotations (e.g; p.P459L, p.L2195Pfs*30 or p.Leu2195ProfsTer30)
labelPos	Amino acid positions to label. If 'all', labels all variants.
labPosSize	Text size for labels. Default 0.9
showMutationRate	Whether to show the somatic mutation rate on the title. Default TRUE
showDomainLabel	Label domains within the plot. Default TRUE. If 'FALSE' domains are 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.
roundedRect	Default TRUE. If 'TRUE' domains are drawn with rounded corners. Requires <a href="#">berryFunctions</a>
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

showLegend	Default TRUE
legendTxtSize	Text size for legend. Default 0.8
labPosAngle	angle for labels. Defaults to horizontal 0 degree labels. Set to 90 for vertical; 45 for diagonal labels.
domainLabelSize	text size for domain labels. Default 0.8
axisTextSize	text size x and y tick labels. Default c(1,1).
printCount	If TRUE, prints number of summarized variants for the given protein.
colors	named vector of colors for each Variant_Classification. Default NULL.
domainAlpha	Default 1
domainBorderCol	Default "black". Set to NA to remove.
bgBorderCol	Default "black". Set to NA to remove.
labelOnlyUniqueDoamins	Default TRUE only labels unique doamins.
defaultYaxis	If FALSE, just labels min and maximum y values on y axis.
titleSize	font size for title and subtitle. Default c(1.2, 1)
pointSize	size of lollipop heads. Default 1.5

### Details

This function by default looks for fields 'HGVS\_Short', 'AAChange' or 'Protein\_Change' in maf file. One can also manually specify field name containing amino acid changes.

### Value

Nothing

### Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools")
lam1 <- read.maf(maf = lam1.maf)
lollipopPlot(maf = lam1, gene = 'KIT', AACol = 'Protein_Change')
```

---

lollipopPlot2 *Compare two lollipop plots*

---

### Description

Compare two lollipop plots

**Usage**

```

lollipopPlot2(
  m1,
  m2,
  gene = NULL,
  AACol1 = NULL,
  AACol2 = NULL,
  m1_name = NULL,
  m2_name = NULL,
  m1_label = NULL,
  m2_label = NULL,
  refSeqID = NULL,
  proteinID = NULL,
  labPosAngle = 0,
  labPosSize = 0.9,
  colors = NULL,
  alpha = 1,
  axisTextSize = c(1, 1),
  pointSize = 1.2,
  roundedRect = TRUE,
  showDomainLabel = TRUE,
  domainBorderCol = "black",
  domainLabelSize = 1,
  legendTxtSize = 1,
  verbose = TRUE
)

```

**Arguments**

m1	first <a href="#">MAF</a> object
m2	second <a href="#">MAF</a> object
gene	HGNC symbol for which protein structure to be drawn.
AACol1	manually specify column name for amino acid changes in m1. Default looks for fields 'HGVS_Short', 'AAChange' or 'Protein_Change'.
AACol2	manually specify column name for amino acid changes in m2. Default looks for fields 'HGVS_Short', 'AAChange' or 'Protein_Change'.
m1_name	name for m1 cohort. optional.
m2_name	name for m2 cohort. optional.
m1_label	Amino acid positions to label for m1 cohort. If 'all', labels all variants.
m2_label	Amino acid positions to label for m2 cohort. If 'all', labels all variants.
refSeqID	RefSeq transcript identifier for gene if known.
proteinID	RefSeq protein identifier for gene if known.
labPosAngle	angle for labels. Defaults to horizontal 0 degree labels. Set to 90 for vertical; 45 for diagonal labels.
labPosSize	Text size for labels. Default 3
colors	named vector of colors for each Variant_Classification. Default NULL.
alpha	color adjustment. Default 1
axisTextSize	text size for axis labels. Default 1.

pointSize	size of lollipop heads. Default 1.2
roundedRect	Default FALSE. If 'TRUE' domains are drawn with rounded corners. Requires berryFunctions
showDomainLabel	Label domains within the plot. Default TRUE. If FALSE domains are annotated in legend.
domainBorderCol	Default "black". Set to NA to remove.
domainLabelSize	text size for domain labels. Default 1.
legendTxtSize	Default 1.
verbose	Default TRUE

### Details

Draws lollipop plot for a gene from two cohorts

### Value

invisible list of domain overlaps

### See Also

[lollipopPlot](#)

[mafCompare](#)

### 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)
lollipopPlot2(m1 = primary.apl, m2 = relapse.apl, gene = "FLT3", AACol1 = "amino_acid_change", AACol2 = "amino_a
```

---

MAF

*Construct an MAF object*

---

### Description

Constructor function which takes non-synonymous, and synonymous variants along with an optional clinical information and generates an MAF object

### Usage

```
MAF(nonSyn = NULL, syn = NULL, clinicalData = NULL, verbose = TRUE)
```

**Arguments**

nonSyn	non-synonymous variants as a data.table or any object that can be coerced into a data.table (e.g: data.frame, GRanges)
syn	synonymous variants as a data.table or any object that can be coerced into a data.table (e.g: data.frame, GRanges)
clinicalData	Clinical data associated with each sample/Tumor_Sample_Barcode in MAF. Could be a text file or a data.frame. Requires at least a column with the name 'Tumor_Sample_Barcode' Default NULL.
verbose	Default TRUE

**Examples**

```

laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml_dt = data.table::fread(input = laml.maf)
laml.clin = system.file('extdata', 'tcga_laml_annot.tsv', package = 'maftools') #Clinical data
# Just for demonstration
nsyn_vars = laml_dt[Variant_Classification %in% "Missense_Mutation"]
syn_vars = laml_dt[Variant_Classification %in% "Silent"]
maftools::MAF(nonSyn = nsyn_vars, syn = syn_vars, clinicalData = laml.clin)

```

MAF-class

*Class MAF***Description**

S4 class for storing summarized MAF.

**Slots**

data data.table of MAF file containing all non-synonymous variants.  
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  
summary table with basic MAF summary stats  
maf.silent subset of main MAF containing only silent variants  
clinical.data clinical data associated with each sample/Tumor\_Sample\_Barcode in MAF.

**See Also**

[getGeneSummary](#) [getSampleSummary](#) [getFields](#)

---

maf2mae	<i>Convert MAF to MultiAssayExperiment object</i>
---------	---

---

### Description

Generates an object of class MultiAssayExperiment from MAF object

### Usage

```
maf2mae(m = NULL)
```

### Arguments

m                    an MAF object

### Examples

```
laml.maf = system.file('extdata', 'tcga_laml.maf.gz', package = 'maftools')
laml.clin = system.file('extdata', 'tcga_laml_annot.tsv', package = 'maftools')
laml = read.maf(maf = laml.maf, clinicalData = laml.clin)
maf2mae(laml)
```

---

mafbarplot	<i>Creates a bar plot</i>
------------	---------------------------

---

### Description

Takes an MAF object and generates a barplot of mutated genes color coded for variant classification

### Usage

```
mafbarplot(
  maf,
  n = 20,
  genes = NULL,
  color = NULL,
  fontSize = 0.7,
  includeCN = FALSE,
  legendfontSize = 0.7,
  borderCol = "#34495e",
  showPct = TRUE
)
```

**Arguments**

maf	an <a href="#">MAF</a> object
n	Number of genes to include. Default 20.
genes	Manually provide names of genes. Default NULL.
color	named vector of colors for each Variant_Classification. Default NULL.
fontSize	Default 0.7
includeCN	Include copy number events if available? Default FALSE
legendfontSize	Default 0.7
borderCol	Default "#34495e". Set to 'NA' for no border color.
showPct	Default TRUE. Show percent altered samples.

**Examples**

```
lam1.maf = system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools") #MAF file
lam1 = read.maf(maf = lam1.maf)
mafbarplot(maf = lam1)
```

---

mafCompare	<i>compare two cohorts (MAF).</i>
------------	-----------------------------------

---

**Description**

compare two cohorts (MAF).

**Usage**

```
mafCompare(
  m1,
  m2,
  m1Name = NULL,
  m2Name = NULL,
  minMut = 5,
  useCNV = TRUE,
  pathways = NULL,
  custom_pw = NULL,
  pseudoCount = FALSE
)
```

**Arguments**

m1	first <a href="#">MAF</a> object
m2	second <a href="#">MAF</a> 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 at least one of the cohort for analysis. Helpful to ignore single mutated genes. Default 5.

useCNV	whether to include copy number events. Default TRUE if available.. Not applicable when 'pathways = TRUE'
pathways	Summarize genes by pathways before comparing. Can be either 'sigpw' or 'smgbp', 'sigpw' uses known oncogenic signalling pathways (Sanchez/Vega et al) whereas 'smgbp' uses pan cancer significantly mutated genes classified according to biological process (Bailey et al). Default NULL
custom_pw	Optional. Can be a two column data.frame/tsv-file with pathway-name and genes involved in them. Default 'NULL'. This argument is mutually exclusive with pathdb
pseudoCount	If TRUE, adds 1 to the contingency table with 0's to avoid 'Inf' values in the estimated odds-ratio.

### Details

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

### Value

result list

### See Also

[forestPlot](#)

[lollipopPlot2](#)

### 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)
```

---

mafSummary

*Summary statistics of MAF*

---

### Description

Summarizes genes and samples irrespective of the type of alteration. This is different from [getSampleSummary](#) and [getGeneSummary](#) which returns summaries of only non-synonymous variants.

### Usage

```
mafSummary(maf)
```

### Arguments

maf                    an MAF object generated by [read.maf](#)

**Details**

This function takes MAF object as input and returns summary table.

**Value**

Returns a list of summarized tables

**See Also**

[getGeneSummary](#) [getSampleSummary](#)

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools")
lam1 <- read.maf(maf = lam1.maf)
mafSummary(maf = lam1)
```

---

mafSurvGroup	<i>Performs survival analysis for a geneset</i>
--------------	---

---

**Description**

Similar to [mafSurvival](#) but for a geneset

**Usage**

```
mafSurvGroup(
  maf,
  geneSet = NULL,
  minMut = NA,
  clinicalData = NULL,
  time = "Time",
  Status = "Status"
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
geneSet	gene names for which survival analysis needs to be performed.
minMut	minimum number of mutated genes in the 'geneSet' to consider a sample as a mutant. Default, 'NA', samples with all the genes mutated are treated as mutant group.
clinicalData	dataframe containing events and time to events. Default looks for clinical data in annotation slot of <a href="#">MAF</a> .
time	column name containing time in clinicalData
Status	column name containing status of patients in clinicalData. must be logical or numeric. e.g, TRUE or FALSE, 1 or 0.

**Value**

Survival plot

**Examples**

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "mafTools")
laml.clin <- system.file("extdata", "tcga_laml_annot.tsv", package = "mafTools")
laml <- read.maf(maf = laml.maf, clinicalData = laml.clin)
mafSurvGroup(maf = laml, geneSet = c('DNMT3A', 'FLT3'), time = 'days_to_last_followup', Status = 'Overall_Survival')
```

---

mafSurvival	<i>Performs survival analysis</i>
-------------	-----------------------------------

---

**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,
  genes = NULL,
  samples = NULL,
  clinicalData = NULL,
  time = "Time",
  Status = "Status",
  groupNames = c("Mutant", "WT"),
  showConfInt = TRUE,
  addInfo = TRUE,
  col = c("maroon", "royalblue"),
  isTCGA = FALSE,
  textSize = 12
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
genes	gene names for which survival analysis needs to be performed. Samples with mutations in any one of the genes provided are used as mutants.
samples	samples to group by. Genes and samples are mutually exclusive.
clinicalData	dataframe containing events and time to events. Default looks for clinical data in annotation slot of <a href="#">MAF</a> .
time	column name containing time in clinicalData
Status	column name containing status of patients in clinicalData. must be logical or numeric. e.g, TRUE or FALSE, 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.

### 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.

### Value

Survival plot

### Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1.clin <- system.file("extdata", "tcga_lam1_annot.tsv", package = "maftools")
lam1 <- read.maf(maf = lam1.maf, clinicalData = lam1.clin)
mafSurvival(maf = lam1, genes = 'DNMT3A', time = 'days_to_last_followup', Status = 'Overall_Survival_Status', ...)
```

---

math.score	<i>calculates MATH (Mutant-Allele Tumor Heterogeneity) score.</i>
------------	---

---

### Description

calculates 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 requires vafs.

### Usage

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

### Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
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)
laml.math <- math.score(maf = laml, vafCol = 'i_TumorVAF_WU',
sampleName = c('TCGA-AB-3009', 'TCGA-AB-2849', 'TCGA-AB-3002', 'TCGA-AB-2972'))
```

---

merge_mafs	<i>Merge multiple mafs into single MAF</i>
------------	--

---

**Description**

Merges multiple maf files/objects/data.frames into a single MAF.

**Usage**

```
merge_mafs(mafs, verbose = TRUE, ...)
```

**Arguments**

mafs	a list of <a href="#">MAF</a> objects or data.frames or paths to MAF files.
verbose	Default TRUE
...	additional arguments passed <a href="#">read.maf</a>

**Value**

[MAF](#) object

---

mutCountMatrix	<i>Generates count matrix of mutations.</i>
----------------	---

---

**Description**

Generates a count matrix of mutations. i.e, number of mutations per gene per sample.

**Usage**

```
mutCountMatrix(
  maf,
  includeSyn = FALSE,
  countOnly = NULL,
  removeNonMutated = TRUE
)
```

**Arguments**

maf	an MAF object generated by <a href="#">read.maf</a>
includeSyn	whether to include synonymous variants in output matrix. Default FALSE
countOnly	Default NULL - counts all variants. You can specify type of 'Variant_Classification' to count. For e.g, countOnly = 'Splice_Site' will generate matrix for only Splice_Site variants.
removeNonMutated	Logical Default TRUE, removes samples with no mutations from the matrix.

**Value**

Integer Matrix

**See Also**[getFields](#) [getGeneSummary](#) [getSampleSummary](#)**Examples**

```

laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
##Generate matrix
mutCountMatrix(maf = laml)
##Generate count matrix of Splice_Site mutations
mutCountMatrix(maf = laml, countOnly = 'Splice_Site')

```

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 <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
AACol	manually specify column name for amino acid changes. Default looks for field 'AChange'
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 predefined 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)
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

## Usage

```
oncoplot(  
  maf,  
  top = 20,  
  minMut = NULL,  
  genes = NULL,  
  altered = FALSE,  
  drawRowBar = TRUE,  
  drawColBar = TRUE,  
  leftBarData = NULL,  
  leftBarLims = NULL,  
  leftBarVline = NULL,  
  leftBarVlineCol = "gray70",  
  rightBarData = NULL,  
  rightBarLims = NULL,  
  rightBarVline = NULL,
```

```
rightBarVlineCol = "gray70",
topBarData = NULL,
topBarLims = NULL,
topBarHline = NULL,
topBarHlineCol = "gray70",
logColBar = FALSE,
includeColBarCN = TRUE,
clinicalFeatures = NULL,
annotationColor = NULL,
annotationDat = NULL,
pathways = NULL,
topPathways = 3,
path_order = NULL,
selectedPathways = NULL,
collapsePathway = FALSE,
pwLineCol = "#535c68",
pwLineWd = 1,
draw_titv = FALSE,
titv_col = NULL,
showTumorSampleBarcodes = FALSE,
tsbToPIDs = NULL,
barcode_mar = 4,
barcodeSrt = 90,
gene_mar = 5,
anno_height = 1,
legend_height = 4,
sortByAnnotation = FALSE,
groupAnnotationBySize = TRUE,
annotationOrder = NULL,
sortByMutation = FALSE,
keepGeneOrder = FALSE,
GeneOrderSort = TRUE,
sampleOrder = NULL,
additionalFeature = NULL,
additionalFeaturePch = 20,
additionalFeatureCol = "gray70",
additionalFeatureCex = 0.9,
genesToIgnore = NULL,
removeNonMutated = FALSE,
fill = TRUE,
cohortSize = NULL,
colors = NULL,
cBioPortal = FALSE,
bgCol = "#ecf0f1",
borderCol = "white",
annoBorderCol = NA,
numericAnnoCol = NULL,
drawBox = FALSE,
fontSize = 0.8,
SampleNamefontSize = 1,
titleFontSize = 1.5,
legendFontSize = 1.2,
```

```

annotationFontSize = 1.2,
sepwd_genes = 0.5,
sepwd_samples = 0.25,
writeMatrix = FALSE,
colbar_pathway = FALSE,
showTitle = TRUE,
titleText = NULL,
showPct = TRUE
)

```

### Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
top	how many top genes to be drawn. defaults to 20.
minMut	draw all genes with 'min' number of mutations. Can be an integer or fraction (of samples mutated), Default NULL
genes	Just draw oncoplot for these genes. Default NULL.
altered	Default FALSE. Chooses top genes based on mutation status. If TRUE chooses top genes based alterations (CNV or mutation).
drawRowBar	logical. Plots right barplot for each gene. Default TRUE.
drawColBar	logical plots top barplot for each sample. Default TRUE.
leftBarData	Data for leftside barplot. Must be a data.frame with two columns containing gene names and values. Default 'NULL'
leftBarLims	limits for 'leftBarData'. Default 'NULL'.
leftBarVline	Draw vertical lines at these values. Default 'NULL'.
leftBarVlineCol	Line color for 'leftBarVline' Default gray70
rightBarData	Data for rightside barplot. Must be a data.frame with two columns containing to gene names and values. Default 'NULL' which draws distribution by variant classification. This option is applicable when only 'drawRowBar' is TRUE.
rightBarLims	limits for 'rightBarData'. Default 'NULL'.
rightBarVline	Draw vertical lines at these values. Default 'NULL'.
rightBarVlineCol	Line color for 'rightBarVline' Default gray70
topBarData	Default 'NULL' which draws absolute number of mutation load for each sample. Can be overridden by choosing one clinical indicator(Numeric) or by providing a two column data.frame containing sample names and values for each sample. This option is applicable when only 'drawColBar' is TRUE.
topBarLims	limits for 'topBarData'. Default 'NULL'.
topBarHline	Draw horizontal lines at these values. Default 'NULL'.
topBarHlineCol	Line color for 'topBarHline.' Default gray70
logColBar	Plot top bar plot on log10 scale. Default FALSE.
includeColBarCN	Whether to include CN in column bar plot. Default TRUE
clinicalFeatures	columns names from 'clinical.data' slot of MAF to be drawn in the plot. Default NULL.

annotationColor	Custom colors to use for 'clinicalFeatures'. Must be a named list containing a named vector of colors. Default NULL. See example for more info.
annotationDat	If MAF file was read without clinical data, provide a custom data.frame with a column Tumor_Sample_Barcode containing sample names along with rest of columns with annotations. You can specify which columns to be drawn using 'clinicalFeatures' argument.
pathways	Default 'NULL'. Can be 'sigpw', 'smgbp', or a two column data.frame/tsv-file with genes and corresponding pathway mappings.'
topPathways	Top most altered pathways to draw. Default 3. Mutually exclusive with 'selectedPathways'
path_order	Default 'NULL' Manually specify the order of pathways
selectedPathways	Manually provide the subset of pathway names to be selected from 'pathways'. Default NULL. In case 'pathways' is 'auto' draws top 3 altered pathways.
collapsePathway	Shows only rows corresponding to the pathways. Default FALSE.
pwLineCol	Color for the box around the pathways Default #535c68
pwLineWd	Line width for the box around the pathways Default Default 1
draw_titv	logical Includes TiTv plot. FALSE
titv_col	named vector of colors for each transition and transversion classes. Should be of length six with the names "C>T" "C>G" "C>A" "T>A" "T>C" "T>G". Default NULL.
showTumorSampleBarcodes	logical to include sample names.
tsbToPIDs	Custom names for Tumor_Sample_Barcodes. Can be a column name in clinicaldata or a 2 column data.frame of Tumor_Sample_Barcodes to patient ID mappings. Applicable only when 'showTumorSampleBarcodes = TRUE'. Default NULL.
barcode_mar	Margin width for sample names. Default 4
barcodeSrt	Rotate sample labels. Default 90.
gene_mar	Margin width for gene names. Default 5
anno_height	Height of plotting area for sample annotations. Default 1
legend_height	Height of plotting area for legend. Default 4
sortByAnnotation	logical sort oncomatrix (samples) by provided 'clinicalFeatures'. Sorts based on first 'clinicalFeatures'. Defaults to FALSE. column-sort
groupAnnotationBySize	Further group 'sortByAnnotation' orders by their size. Defaults to TRUE. Largest groups comes first.
annotationOrder	Manually specify order for annotations. Works only for first 'clinicalFeatures'. Default NULL.
sortByMutation	Force sort matrix according mutations. Helpful in case of MAF was read along with copy number data. Default FALSE.
keepGeneOrder	logical whether to keep order of given genes. Default FALSE, order according to mutation frequency

GeneOrderSort	logical this is applicable when 'keepGeneOrder' is TRUE. Default TRUE
sampleOrder	Manually speify sample names for oncolplot ordering. Default NULL.
additionalFeature	a vector of length two indicating column name in the MAF and the factor level to be highlighted. Provide a list of values for highlighting more than one features
additionalFeaturePch	Default 20
additionalFeatureCol	Default "gray70"
additionalFeatureCex	Default 0.9
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 FALSE.
fill	Logical. If TRUE draws genes and samples as blank grids even when they are not altered.
cohortSize	Number of sequenced samples in the cohort. Default all samples from Cohort. You can manually specify the cohort size. Default NULL
colors	named vector of colors for each Variant_Classification.
cBioPortal	Adds annotations similar to cBioPortals MutationMapper and collapse Variants into Truncating and rest.
bgCol	Background grid color for wild-type (not-mutated) samples. Default "#ecf0f1"
borderCol	border grid color (not-mutated) samples. Default 'white'.
annoBorderCol	border grid color for annotations. Default NA.
numericAnnoCol	color palette used for numeric annotations. Default 'YlOrBr' from RColorBrewer
drawBox	logical whether to draw a box around main matrix. Default FALSE
fontSize	font size for gene names. Default 0.8.
SampleNameFontSize	font size for sample names. Default 1
titleFontSize	font size for title. Default 1.5
legendFontSize	font size for legend. Default 1.2
annotationFontSize	font size for annotations. Default 1.2
sepwd_genes	size of lines seperating genes. Default 0.5
sepwd_samples	size of lines seperating samples. Default 0.25
writeMatrix	writes character coded matrix used to generate the plot to an output file.
colbar_pathway	Draw top column bar with respect to diplayed pathway. Default FALSE.
showTitle	Default TRUE
titleText	Custom title. Default 'NULL'
showPct	Default TRUE. Shows percent altered to the right side of the plot.

## Details

Takes an [MAF](#) object as an input and plots it as a matrix. Any desired clinical features can be added at the bottom of the oncoplot by providing `clinicalFeatures`. Oncoplot can be sorted either by mutations or by `clinicalFeatures` using arguments `sortByMutation` and `sortByAnnotation` respectively.

By setting `'pathways'` argument either `'sigpw'` or `'smgpb'` - cohort can be summarized by altered pathways. `pathways` argument also accepts a custom pathway list in the form of a two column tsv file or a `data.frame` containing gene names and their corresponding pathway.

## Value

Invisibly returns a list with components 1. `'oncomatrix'` A matrix used for drawing the oncoplot. Values are numeric coded for each variant classification 2. `'vc_legend'` A mapping of variant classification to numeric values in the oncomatrix 3. `'vc_color'` Color coding used for each variant classification

## References

Bailey, Matthew H et al. "Comprehensive Characterization of Cancer Driver Genes and Mutations." *Cell* vol. 173,2 (2018): 371-385.e18. doi:10.1016/j.cell.2018.02.060 Sanchez-Vega, Francisco et al. "Oncogenic Signaling Pathways in The Cancer Genome Atlas." *Cell* vol. 173,2 (2018): 321-337.e10. doi:10.1016/j.cell.2018.03.035

## See Also

[pathways](#)

## Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1.clin = system.file('extdata', 'tcga_lam1_annot.tsv', package = 'maftools')
lam1 <- read.maf(maf = lam1.maf, clinicalData = lam1.clin)
#Basic oncoplot
oncoplot(maf = lam1, top = 3)
#Changing colors for variant classifications (You can use any colors, here in this example we will use a color palette)
col = RColorBrewer::brewer.pal(n = 8, name = 'Paired')
names(col) = c('Frame_Shift_Del', 'Missense_Mutation', 'Nonsense_Mutation', 'Multi_Hit', 'Frame_Shift_Ins',
              'In_Frame_Ins', 'Splice_Site', 'In_Frame_Del')
#Color coding for FAB classification; try getAnnotations(x = lam1) to see available annotations.
fabcolors = RColorBrewer::brewer.pal(n = 8, name = 'Spectral')
names(fabcolors) = c("M0", "M1", "M2", "M3", "M4", "M5", "M6", "M7")
fabcolors = list(FAB_classification = fabcolors)
oncoplot(maf = lam1, colors = col, clinicalFeatures = 'FAB_classification', sortByAnnotation = TRUE, annotation
```

---

oncostrip

*draw an oncostrip similar to cBioportal oncoprinter output.*

---

## Description

draw an oncostrip similar to cBioportal oncoprinter output.

**Usage**

```
oncostrip(maf = NULL, ...)
```

**Arguments**

```
maf          an MAF object generated by read.maf
...          arguments passed oncoplot
```

**Details**

This is just a wrapper around [oncoplot](#) with `drawRowBar` and `drawColBar` set to `FALSE`

**Value**

None.

**See Also**

[oncoplot](#)

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
dev.new()
oncostrip(maf = lam1, genes = c('NPM1', 'RUNX1'))
```

---

pathways

*Enrichment of known oncogenic or custom pathways*

---

**Description**

Checks for enrichment of known or custom pathways

**Usage**

```
pathways(
  maf,
  pathdb = "sigpw",
  pathways = NULL,
  fontSize = 1,
  panelWidths = c(2, 4, 4),
  plotType = NA,
  col = "#f39c12"
)
```

## Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
pathdb	Either 'sigpw' or 'smgbp', 'sigpw' uses known oncogenic signalling pathways (Sanchez/Vega et al) whereas 'smgbp' uses pan cancer significantly mutated genes classified according to biological process (Bailey et al). Default smgbp
pathways	Can be a two column data.frame/tsv-file with gene names and pathway-name involved in them. Default 'NULL'. This argument is mutually exclusive with pathdb
fontSize	Default 1
panelWidths	Default c(2, 4, 4)
plotType	Can be 'treemap' or 'bar'. Set NA to suppress plotting. Default NA
col	Default #f39c12

## Details

Oncogenic signalling and SMG pathways are derived from TCGA cohorts. See references for details.

## Value

fraction of altered pathway. attr genes contain pathway contents

## References

Sanchez-Vega F, Mina M, Armenia J, Chatila WK, Luna A, La KC, Dimitriadoy S, Liu DL, Kantheti HS, Saghafeinia S et al. 2018. Oncogenic Signaling Pathways in The Cancer Genome Atlas. Cell 173: 321-337 e310 Bailey, Matthew H et al. "Comprehensive Characterization of Cancer Driver Genes and Mutations." Cell vol. 173,2 (2018): 371-385.e18.

## See Also

[plotPathways](#)

## Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
pathways(maf = lam1)
```

---

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,
  domainsToLabel = NULL,
  baseName = NULL,
  varClass = "nonSyn",
  width = 5,
  height = 5,
  labelSize = 1
)
```

**Arguments**

maf	an MAF object generated by <code>read.maf</code>
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.
domainsToLabel	Default NULL. Exclusive with top argument.
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.
width	width of the file to be saved.
height	height of the file to be saved.
labelSize	font size for labels. Default 1.

**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**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
pfamDomains(maf = lam1, AACol = 'Protein_Change')
```

---

plotApobecDiff	<i>Plot differences between APOBEC enriched and non-APOBEC enriched samples.</i>
----------------	--

---

**Description**

Plots differences between APOBEC enriched and non-APOBEC enriched samples

## Usage

```
plotApobecDiff(  
  tnm,  
  maf,  
  pVal = 0.05,  
  title_size = 1,  
  axis_lwd = 1,  
  font_size = 1.2  
)
```

## Arguments

tnm	output generated by <a href="#">trinucleotideMatrix</a>
maf	an <a href="#">MAF</a> object used to generate the matrix
pVal	p-value threshold for fisher's test. Default 0.05.
title_size	size of title. Default 1.3
axis_lwd	axis width. Default 1
font_size	font size. Default 1.2

## Details

Plots differences between APOBEC enriched and non-APOBEC enriched samples (TCW). Plot includes differences in mutations load, tCw motif distribution and top genes altered.

## Value

list of table containing differentially altered genes. This can be passed to [forestPlot](#) to plot results.

## See Also

[trinucleotideMatrix](#) [plotSignatures](#)

## Examples

```
## Not run:  
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")  
laml <- read.maf(maf = laml.maf)  
laml.tnm <- trinucleotideMatrix(maf = laml, ref_genome = 'BSgenome.Hsapiens.UCSC.hg19', prefix = 'chr',  
  add = TRUE, useSyn = TRUE)  
plotApobecDiff(laml.tnm)  
  
## End(Not run)
```

---

plotCBSsegments      *Plots segmented copy number data.*

---

### Description

Plots segmented copy number data.

### Usage

```
plotCBSsegments(
  cbsFile = NULL,
  maf = NULL,
  tsb = NULL,
  savePlot = FALSE,
  ylims = NULL,
  seg_size = 0.1,
  width = 6,
  height = 3,
  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 <a href="#">MAF</a>
tsb	If segmentation file contains many samples (as in gistic input), specify sample name here. Default plots head 1 sample. Set 'ALL' for plotting all samples. If you are mapping maf, make sure sample names in Sample column of segmentation file matches to those Tumor_Sample_Barcodes in MAF.
savePlot	If true plot is saved as pdf.
ylims	Default NULL
seg_size	Default 0.1
width	width of plot
height	height of plot
genes	If given and maf object is specified, maps all mutations from maf onto segments. Default NULL
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. i.e, alternating Gray70 and midnightblue

**Details**

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

**Value**

Draws plot

**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 clustering results.*

---

**Description**

Plots results from inferHeterogeneity.

**Usage**

```
plotClusters(
  clusters,
  tsb = NULL,
  genes = NULL,
  showCNvars = FALSE,
  colors = NULL
)
```

**Arguments**

clusters	clustering results from <a href="#">inferHeterogeneity</a>
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 variants. or all to label all genes. Default NULL.
showCNvars	show copy numbered altered variants on the plot. Default FALSE.
colors	manual colors for clusters. Default NULL.

**Value**

returns nothing.

**See Also**

[inferHeterogeneity](#)

## Examples

```
## Not run:
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
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)

## End(Not run)
```

---

plotCophenetic	<i>Draw an elbow plot of cophenetic correlation metric.</i>
----------------	---

---

## Description

Draw an elbow plot of cophenetic correlation metric.

## Usage

```
plotCophenetic(res = NULL, bestFit = NULL)
```

## Arguments

res	output from <a href="#">estimateSignatures</a>
bestFit	rank to highlight. Default NULL

## Details

This function draws an elbow plot of cophenetic correlation metric.

## See Also

[estimateSignatures](#) [plotCophenetic](#)

---

plotEnrichmentResults	<i>Plots results from clinicalEnrichment analysis</i>
-----------------------	---

---

## Description

Plots results from clinicalEnrichment analysis

**Usage**

```
plotEnrichmentResults(  
  enrich_res,  
  pVal = 0.05,  
  ORthr = 1,  
  featureLvls = NULL,  
  cols = NULL,  
  annoFontSize = 0.8,  
  geneFontSize = 0.8,  
  legendFontSize = 0.8,  
  showTitle = TRUE,  
  ylims = c(-1, 1)  
)
```

**Arguments**

enrich_res	results from <a href="#">clinicalEnrichment</a> or <a href="#">signatureEnrichment</a>
pVal	Default 0.05
ORthr	Default 1. Odds ratio threshold. >1 indicates positive enrichment in the group of interest.
featureLvls	Plot results from the selected levels. Default NULL, plots all.
cols	named vector of colors for factor in a clinical feature. Default NULL
annoFontSize	cex for annotation font size. Default 0.8
geneFontSize	cex for gene font size. Default 0.8
legendFontSize	cex for legend font size. Default 0.8
showTitle	Default TRUE
ylims	Default c(-1, 1)

**Value**

returns nothing.

**See Also**

[clinicalEnrichment](#) [signatureEnrichment](#)

---

plotmafSummary	<i>Plots maf summary.</i>
----------------	---------------------------

---

**Description**

Plots maf summary.

**Usage**

```
plotmafSummary(
  maf,
  rmOutlier = TRUE,
  dashboard = TRUE,
  titvRaw = TRUE,
  log_scale = FALSE,
  addStat = NULL,
  showBarcodes = FALSE,
  fs = 1,
  textSize = 0.8,
  color = NULL,
  titleSize = c(1, 0.8),
  titvColor = NULL,
  top = 10
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
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.
log_scale	FALSE. If TRUE log10 transforms Variant Classification, Variant Type and Variants per sample sub-plots.
addStat	Can be either mean or median. Default NULL.
showBarcodes	include sample names in the top bar plot.
fs	base size for text. Default 1
textSize	font size if showBarcodes is TRUE. Default 0.8
color	named vector of colors for each Variant_Classification.
titleSize	font size for title and subtitle. Default c(10, 8)
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')
```

---

plotMosdepth	<i>Plot results from mosdepth output for Tumor/Normal pair</i>
--------------	--

---

**Description**

Plot results from mosdepth output for Tumor/Normal pair

**Usage**

```
plotMosdepth(  
  t_bed = NULL,  
  n_bed = NULL,  
  segment = TRUE,  
  sample_name = NULL,  
  col = c("#95a5a6", "#7f8c8d")  
)
```

**Arguments**

t_bed	mosdepth output from tumor
n_bed	mosdepth output from matched normal
segment	Performs CBS segmentation. Default TRUE
sample_name	sample name. Default parses from 't_bed'
col	Colors. Default c("#95a5a6", "#7f8c8d")

**Value**

Invisibly returns [DNACopy](#) object if 'segment' is 'TRUE'

**References**

Pedersen BS, Quinlan AR. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics*. 2018;34(5):867-868. doi:10.1093/bioinformatics/btx699

---

plotMosdepth_t	<i>Plot results from mosdepth output</i>
----------------	--

---

**Description**

Plot results from mosdepth output

**Usage**

```
plotMosdepth_t(  
  bed = NULL,  
  col = c("#95a5a6", "#7f8c8d"),  
  sample_name = NULL,  
  segment = FALSE  
)
```

**Arguments**

bed	mosdepth output
col	Colors. Default c("#95a5a6", "#7f8c8d")
sample_name	sample name. Default parses from 'bed'
segment	Performs CBS segmentation. Default FALSE

**Value**

Invisibly returns [DNAcopy](#) object if 'segment' is 'TRUE'

**References**

Pedersen BS, Quinlan AR. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics*. 2018;34(5):867-868. doi:10.1093/bioinformatics/btx699

---

plotOncodrive	<i>Plots results from oncodrive</i>
---------------	-------------------------------------

---

**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  $-\log_{10}$  for better representation. Labels indicate Gene name with number clusters observed.

**Usage**

```
plotOncodrive(
  res = NULL,
  fdrCutoff = 0.05,
  useFraction = FALSE,
  colCode = NULL,
  bubbleSize = 1,
  labelSize = 1
)
```

**Arguments**

res	results from <a href="#">oncodrive</a>
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-significant genes. Default NULL
bubbleSize	Size for bubbles. Default 2.
labelSize	font size for labelling genes. Default 1.

**Value**

Nothing

**See Also**[oncodrive](#)**Examples**

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

---

plotPathways	<i>Plot oncogenic pathways</i>
--------------	--------------------------------

---

**Description**

Plot oncogenic pathways

**Usage**

```
plotPathways(
  maf = NULL,
  pathlist = NULL,
  pathnames = NULL,
  removeNonMutated = FALSE,
  fontSize = 1,
  showTumorSampleBarcodes = FALSE,
  sampleOrder = NULL,
  SampleNamefontSize = 0.6,
  mar = c(4, 6, 2, 3)
)
```

**Arguments**

maf	an <a href="#">MAF</a> object
pathlist	Output from <a href="#">pathways</a>
pathnames	Names of the pathways to be drawn. Default NULL, plots everything from input 'pathlist'
removeNonMutated	Default FALSE
fontSize	Default 1
showTumorSampleBarcodes	logical to include sample names.
sampleOrder	Manually specify sample names for oncolplot ordering. Default NULL.
SampleNamefontSize	font size for sample names. Default 0.6
mar	margins Default c(4, 6, 2, 3). Margins to bottom, left, top and right respectively

**Details**

Draws pathway burden123

**References**

Sanchez-Vega F, Mina M, Armenia J, Chatila WK, Luna A, La KC, Dimitriadoy S, Liu DL, Kantheti HS, Saghafeinia S et al. 2018. Oncogenic Signaling Pathways in The Cancer Genome Atlas. Cell 173: 321-337 e310

**See Also**

[pathways](#)

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "mafTools")
lam1 <- read.maf(maf = lam1.maf)
p <- pathways(maf = lam1)
plotPathways(maf = lam1, pathlist = p)
```

---

plotProtein

*Display protein domains*

---

**Description**

Display protein domains

**Usage**

```
plotProtein(  
  gene,  
  refSeqID = NULL,  
  proteinID = NULL,  
  domainAlpha = 0.9,  
  showLegend = FALSE,  
  bgBorderCol = "black",  
  axisTextSize = c(1, 1),  
  roundedRect = TRUE,  
  domainBorderCol = "black",  
  showDomainLabel = TRUE,  
  domainLabelSize = 0.8,  
  titleSize = c(1.2, 1),  
  legendTxtSize = 1,  
  legendNcol = 1  
)
```

**Arguments**

gene	HGNC symbol for which protein structure to be drawn.
refSeqID	RefSeq transcript identifier for gene if known.
proteinID	RefSeq protein identifier for gene if known.
domainAlpha	Default 1
showLegend	Default TRUE
bgBorderCol	Default "black". Set to NA to remove.
axisTextSize	text size x and y tick labels. Default c(1,1).
roundedRect	Default TRUE. If 'TRUE' domains are drawn with rounded corners. Requires berryFunctions
domainBorderCol	Default "black". Set to NA to remove.
showDomainLabel	Default TRUE
domainLabelSize	text size for domain labels. Default 0.8
titleSize	font size for title and subtitle. Default c(1.2, 1)
legendTxtSize	Text size for legend. Default 0.8
legendNcol	Default 1

**Examples**

```
par(mfrow = c(2, 1))
plotProtein(gene = "KIT")
plotProtein(gene = "DNMT3A")
```

---

plotSignatures

*Plots decomposed mutational signatures*


---

**Description**

Takes results from [extractSignatures](#) and plots decomposed mutational signatures as a barplot.

**Usage**

```
plotSignatures(
  nmfRes = NULL,
  contributions = FALSE,
  absolute = FALSE,
  color = NULL,
  patient_order = NULL,
  font_size = 0.6,
  show_title = TRUE,
  sig_db = "SBS_v34",
  axis_lwd = 1,
  title_size = 0.9,
  show_barcodes = FALSE,
  yaxisLim = NA,
  ...
)
```

**Arguments**

nmfRes	results from <a href="#">extractSignatures</a>
contributions	If TRUE plots contribution of signatures in each sample.
absolute	Whether to plot absolute contributions. Default FALSE.
color	colors for each Ti/Tv conversion class. Default NULL
patient_order	User defined ordering of samples. Default NULL.
font_size	font size. Default 0.6
show_title	If TRUE compares signatures to COSMIC signatures and prints them as title
sig_db	Only applicable if show_title is TRUE. can be legacy, SBS, SBS_v34. Default SBS_v34
axis_lwd	axis width. Default 1.
title_size	size of title. Default 1.3
show_barcodes	Default FALSE
yaxisLim	Default NA.
...	further plot options passed to <a href="#">barplot</a>

**Value**

Nothing

**See Also**

[trinucleotideMatrix](#) [plotSignatures](#)

---

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",
  sampleOrder = NULL,
  color = NULL,
  showBarcodes = FALSE,
  textSize = 0.8,
  baseFontSize = 1,
  axisTextSize = c(1, 1),
  plotNotch = FALSE
)
```

**Arguments**

res	results generated by <a href="#">titv</a>
plotType	Can be 'bar', 'box' or 'both'. Defaults to 'both'
sampleOrder	Sample names in which the barplot should be ordered. Default NULL
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.
baseFontSize	font size. Deafult 1.
axisTextSize	text size x and y tick labels. Default c(1,1).
plotNotch	logical. Include notch in boxplot.

**Value**

None.

**See Also**

[titv](#)

**Examples**

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
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. Each dot in the jitter is a variant.

**Usage**

```
plotVaf(
  maf,
  vafCol = NULL,
  genes = NULL,
  top = 10,
  orderByMedian = TRUE,
  keepGeneOrder = FALSE,
  flip = FALSE,
  fn = NULL,
  gene_fs = 0.8,
  axis_fs = 0.8,
  height = 5,
  width = 5,
  showN = TRUE,
  color = NULL
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
vafCol	manually specify column name for vafs. Default looks for column 't_vaf'
genes	specify genes for which plots has to be generated
top	if genes is NULL plots top n number of genes. Defaults to 5.
orderByMedian	Orders genes by decreasing median VAF. Default TRUE
keepGeneOrder	keep gene order. Default FALSE
flip	if TRUE, flips axes. Default FALSE
fn	Filename. If given saves plot as a output pdf. Default NULL.
gene_fs	font size for gene names. Default 0.8
axis_fs	font size for axis. Default 0.8
height	Height of plot to be saved. Default 5
width	Width of plot to be saved. Default 4
showN	if TRUE, includes number of observations
color	manual colors. Default NULL.

**Value**

Nothing.

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
plotVaf(maf = lam1, vafCol = 'i_TumorVAF_WU')
```

---

prepareMutSig	<i>Prepares MAF file for MutSig analysis.</i>
---------------	---

---

**Description**

Corrects gene names for MutSig compatibility.

**Usage**

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

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
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**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
prepareMutSig(maf = lam1)
```

---

```
prepAscat
```

---

*Prepare input files for ASCAT*

---

**Description**

Function takes the output from [gtMarkers](#) and generates 'logR' and 'BAF' files required for ASCAT analysis.

**Usage**

```
prepAscat(
  t_counts = NULL,
  n_counts = NULL,
  sample_name = NA,
  min_depth = 15,
  normalize = TRUE
)
```

**Arguments**

t_counts	read counts from tumor generated by <a href="#">gtMarkers</a>
n_counts	read counts from normal generated by <a href="#">gtMarkers</a>
sample_name	Sample name. Used as a basename for output files. Default 'NA', parses from 't_counts' file.
min_depth	Min read depth required to consider a marker. Default 15
normalize	If TRUE, normalizes for library size. Default TRUE

**Details**

The function will filter SNPs with low coverage (default <15), estimate BAF, logR, and generates the input files for ASCAT. Alternatively, logR file can be segmented with [segmentLogR](#)

**References**

Van Loo P, Nordgard SH, Lingjærde OC, et al. Allele-specific copy number analysis of tumors. Proc Natl Acad Sci U S A. 2010;107(39):16910-16915. doi:10.1073/pnas.1009843107

**See Also**

[gtMarkers](#) [prepAscat\\_t](#) [segmentLogR](#)

---

```
prepAscat_t
```

*Prepare input files for ASCAT tumor only samples*

---

**Description**

Function takes the output from [gtMarkers](#) and generates 'logR' and 'BAF' files required for ASCAT analysis.

**Usage**

```
prepAscat_t(t_counts = NULL, sample_name = NA, min_depth = 15)
```

**Arguments**

t_counts	read counts from tumor generated by <a href="#">gtMarkers</a>
sample_name	Sample name. Used as a basename for output files. Default NA, parses from 't_counts' file.
min_depth	Min read depth required to consider a marker. Default 15

**Details**

The function will filter SNPs with low coverage (default <15), estimate BAF, logR, and generates the input files for ASCAT. Tumor 'logR' file will be normalized for median depth of coverage. Alternatively, logR file can be segmented with [segmentLogR](#)

**Value**

Generates logR and BAF files required by ASCAT

**References**

Van Loo P, Nordgard SH, Lingjærde OC, et al. Allele-specific copy number analysis of tumors. Proc Natl Acad Sci U S A. 2010;107(39):16910-16915. doi:10.1073/pnas.1009843107

**See Also**

[gtMarkers](#) [prepAscat](#) [segmentLogR](#)

---

rainfallPlot	<i>Rainfall plot to display hyper mutated genomic regions.</i>
--------------	--

---

### 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 = 1.2,
  pointSize = 0.4
)
```

### Arguments

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a> . 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, detects genomic change points where potential kataegis are formed. Results are written to an output tab delimited 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 0.8.

### Details

If ‘detectChangePoints‘ is set to TRUE, this function will identify Kataegis loci. Kataegis detection algorithm by Moritz Goretzky at WWU Munster, which exploits the definition of Kataegis (six consecutive mutations with an avg. distance of 1000bp ) to identify hyper mutated genomic loci. Algorithm starts with a double-ended queue to which six consecutive mutations are added and their average intermutation distance is calculated. If the average intermutation distance is larger than 1000, one element is added at the back of the queue and one is removed from the front. If the average intermutation distance is less or equal to 1000, further mutations are added until the average intermutation distance is larger than 1000. After that all mutations in the double-ended queue are written into output as one kataegis and the double-ended queue is reinitialized with six mutations.

**Value**

Results are written to an output file with suffix changePoints.tsv

---

read.maf	<i>Read MAF files.</i>
----------	------------------------

---

**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,
  clinicalData = NULL,
  rmFlags = FALSE,
  removeDuplicatedVariants = TRUE,
  useAll = TRUE,
  gisticAllLesionsFile = NULL,
  gisticAmpGenesFile = NULL,
  gisticDelGenesFile = NULL,
  gisticScoresFile = NULL,
  cnLevel = "all",
  cnTable = NULL,
  isTCGA = FALSE,
  vc_nonSyn = NULL,
  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.
clinicalData	Clinical data associated with each sample/Tumor_Sample_Barcode in MAF. Could be a text file or a data.frame. Default NULL.
rmFlags	Default FALSE. Can be TRUE or an integer. If TRUE removes all the top 20 FLAG genes. If integer, remove top n FLAG genes.
removeDuplicatedVariants	removes repeated variants in a particular sample, mapped to multiple transcripts of same Gene. See Description. Default 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.

<code>gisticDelGenesFile</code>	Deletion Genes file generated by gistic. e.g; del_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
<code>gisticScoresFile</code>	scores.gistic file generated by gistic. Default NULL
<code>cnLevel</code>	level of CN changes to use. Can be 'all', 'deep' or 'shallow'. Default uses all i.e, genes with both 'shallow' or 'deep' CN changes
<code>cnTable</code>	Custom copynumber data if gistic results are not available. Input file or a data.frame should contain three columns in aforementioned order with gene name, Sample name and copy number status (either 'Amp' or 'Del'). Default NULL. Recommended to include additional columns 'Chromosome' 'Start_Position' 'End_Position'
<code>isTCGA</code>	Is input MAF file from TCGA source. If TRUE uses only first 12 characters from Tumor_Sample_Barcode.
<code>vc_nonSyn</code>	NULL. Provide manual list of variant classifications to be considered as non-synonymous. Rest will be considered as silent variants. Default uses Variant Classifications with High/Moderate variant consequences. <a href="https://m.ensembl.org/info/genome/variation/variant_classification.html">https://m.ensembl.org/info/genome/variation/variant_classification.html</a> "Frame_Shift_Del", "Frame_Shift_Ins", "Splice_Site", "Translation_Start_Site", "Nonsense_Mutation", "Nonstop_Mutation", "In_Frame_Del", "In_Frame_Ins", "Missense_Mutation"
<code>verbose</code>	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") #MAF file
laml.clin = system.file('extdata', 'tcga_laml_annot.tsv', package = 'maftools') #clinical data
laml = read.maf(maf = laml.maf, clinicalData = laml.clin)
```

---

readGistic	<i>Read and summarize gistic output.</i>
------------	--

---

**Description**

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

**Usage**

```
readGistic(
  gisticDir = NULL,
  gisticAllLesionsFile = NULL,
  gisticAmpGenesFile = NULL,
  gisticDelGenesFile = NULL,
  gisticScoresFile = NULL,
  cnLevel = "all",
  isTCGA = FALSE,
  verbose = TRUE
)
```

**Arguments**

<code>gisticDir</code>	Directory containing GISTIC results. Default NULL. If provided all relevant files will be imported. Alternatively, below arguments can be used to import required files.
<code>gisticAllLesionsFile</code>	All Lesions file generated by gistic. e.g; all_lesions.conf_XX.txt, where XX is the confidence level. Required. Default NULL.
<code>gisticAmpGenesFile</code>	Amplification Genes file generated by gistic. e.g; amp_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
<code>gisticDelGenesFile</code>	Deletion Genes file generated by gistic. e.g; del_genes.conf_XX.txt, where XX is the confidence level. Default NULL.
<code>gisticScoresFile</code>	scores.gistic file generated by gistic.
<code>cnLevel</code>	level of CN changes to use. Can be 'all', 'deep' or 'shallow'. Default uses all i.e, genes with both 'shallow' or 'deep' CN changes
<code>isTCGA</code>	Is the data from TCGA. Default FALSE.
<code>verbose</code>	Default TRUE

**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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
```

---

sampleSwaps

*Identify sample swaps and similarities*

---

**Description**

Given a list BAM files, the function genotypes known SNPs and identifies potentially related samples. For the source of SNPs, see reference

**Usage**

```
sampleSwaps(
  bams = NULL,
  build = "hg19",
  prefix = NULL,
  add = TRUE,
  min_depth = 30,
  ncores = 4,
  ...
)
```

**Arguments**

bams	Input bam files. Required.
build	reference genome build. Default "hg19". Can be hg19 or hg38
prefix	Prefix to add or remove from contig names in SNP file. If BAM files are aligned GRCh37/38 genome, use prefix 'chr' to 'add'
add	If prefix is used, default is to add prefix to contig names in SNP file. If FALSE prefix will be removed from contig names.
min_depth	Minimum read depth for a SNP to be considered. Default 30.
ncores	Default 4. Each BAM file will be launched on a separate thread. Works only on Unix and macOS.
...	Additional arguments passed to <a href="#">bamreadcounts</a>

**Value**

a list with results summarized

**References**

Westphal, M., Frankhouser, D., Sonzone, C. et al. SMaSH: Sample matching using SNPs in humans. BMC Genomics 20, 1001 (2019). <https://doi.org/10.1186/s12864-019-6332-7>

---

`segmentLogR`*Segment and plot log ratio values with DNACopy*

---

**Description**

The function takes logR file generated by [prepAscat](#) or [prepAscat\\_t](#) and performs segmentation with [DNACopy](#)

**Usage**

```
segmentLogR(tumor_logR = NULL, sample_name = NULL, build = "hg19")
```

**Arguments**

<code>tumor_logR</code>	logR.txt file generated by <a href="#">prepAscat</a> or <a href="#">prepAscat_t</a>
<code>sample_name</code>	Default NULL. Parses from 'tumor_logR' file
<code>build</code>	Reference genome. Default hg19. Can be hg18, hg19, or hg38

**Value**

Invisibly returns [DNACopy](#) object

**See Also**

[gtMarkers prepAscat](#)

---

`segSummarize`*Summarize CBS segmentation results*

---

**Description**

Summarize CBS segmentation results

**Usage**

```
segSummarize(  
  seg = NULL,  
  build = "hg19",  
  cytoband = NULL,  
  thr = 0.3,  
  verbose = TRUE,  
  maf = NULL,  
  genes = NULL,  
  topanno = NULL,  
  topannocols = NA  
)
```

**Arguments**

seg	segmentation results generated from DNACopy package <a href="#">segment</a> . Input should be a multi-sample segmentation file or a data.frame. First six columns should correspond to sample name, chromosome, start, end, Num_Probes, Segment_Mean in log2 scale. (default output format from DNACopy)
build	genome build. Default hg19. Can be hg19, hg38. If other than these, use 'cytoband' argument
cytoband	cytoband data from UCSC genome browser. Only needed if 'build' is other than 'hg19' or 'hg38'
thr	threshold to call amplification and deletion. Any cytobands or chromosomal arms with median logR above or below this will be called. Default 0.3
verbose	Default TRUE
maf	optional MAF
genes	Add mutation status of these genes as an annotation to the heatmap
topanno	annotation for each sample. This is passed as an input to 'annotation_col' of 'pheatmap'
topannocols	annotation cols for 'topanno'. This is passed as an input to 'annotation_colors' of 'pheatmap'

**Details**

A handy function to summarize CBS segmentation results. Takes segmentation results generated by DNACopy package [segment](#) and summarizes the CN for each cytoband and chromosomal arms.

**Value**

List of median CN values for each cytoband and chromosomal arm along with the plotting matrix

**Examples**

```
laml.seg <- system.file("extdata", "LAML_CBS_segments.tsv.gz", package = "maftools")
segSummarize(seg = laml.seg)

#Highlight some genes as annotation
laml.maf = system.file("extdata", "tcga_laml.maf.gz", package = "maftools") #MAF file
laml.clin = system.file('extdata', 'tcga_laml_annot.tsv', package = 'maftools') #clinical data
laml = read.maf(maf = laml.maf, clinicalData = laml.clin)

segSummarize(seg = laml.seg, maf = laml, genes = c("FLT3", "DNMT3A"))
```

---

setdiffMAF

*Set Operations for MAF objects*


---

**Description**

Set Operations for MAF objects

**Usage**

```
setdiffMAF(x, y, mafObj = TRUE, refAltMatch = TRUE, ...)

intersectMAF(x, y, refAltMatch = TRUE, mafObj = TRUE, ...)
```

**Arguments**

x the first ‘MAF’ object.  
y the second ‘MAF’ object.  
mafObj Return output as an ‘MAF’ object. Default ‘TRUE’  
refAltMatch Set operations are done by matching ref and alt alleles in addition to loci (Default). If FALSE only loci (chr, start, end positions) are matched.  
... other parameters passing to ‘subsetMaf’ for subsetting operations.

**Value**

subset table or an object of class [MAF-class](#). If no overlaps found returns ‘NULL’

**Examples**

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml <- read.maf(maf = laml.maf)
x <- subsetMaf(maf = laml, tsb = c('TCGA-AB-3009'))
y <- subsetMaf(maf = laml, tsb = c('TCGA-AB-2933'))
setdiffMAF(x, y)
intersectMAF(x, y) #Should return NULL due to no common variants
```

---

signatureEnrichment	<i>Performs sample stratification based on signature contribution and enrichment analysis.</i>
---------------------	--

---

**Description**

Performs k-means clustering to assign signature to samples and performs enrichment analysis. Note - Do not use this function. This will be removed in future updates.

**Usage**

```
signatureEnrichment(maf, sig_res, minMut = 5, useCNV = FALSE, fn = NULL)
```

**Arguments**

maf an [MAF](#) object used for signature analysis.  
sig\_res Signature results from [extractSignatures](#)  
minMut Consider only genes with minimum this number of samples mutated. Default 5.  
useCNV whether to include copy number events. Only applicable when MAF is read along with copy number data. Default TRUE if available.  
fn basename for output file. Default NULL.

**Value**

result list containing p-values

**See Also**

[plotEnrichmentResults](#)

---

somaticInteractions	<i>Exact tests to detect mutually exclusive, co-occurring and altered gene-sets.</i>
---------------------	--

---

**Description**

Performs Pair-wise Fisher's Exact test to detect mutually exclusive or co-occurring events.

**Usage**

```
somaticInteractions(
  maf,
  top = 25,
  genes = NULL,
  pvalue = c(0.05, 0.01),
  returnAll = TRUE,
  geneOrder = NULL,
  fontSize = 0.8,
  leftMar = 4,
  topMar = 4,
  showSigSymbols = TRUE,
  showCounts = FALSE,
  countStats = "all",
  countType = "all",
  countsFontSize = 0.8,
  countsFontColor = "black",
  colPal = "BrBG",
  revPal = FALSE,
  showSum = TRUE,
  plotPadj = FALSE,
  colNC = 9,
  nShiftSymbols = 5,
  sigSymbolsSize = 2,
  sigSymbolsFontSize = 0.9,
  pvSymbols = c(46, 42),
  limitColorBreaks = TRUE
)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
top	check for interactions among top 'n' number of genes. Defaults to top 25.
genes	List of genes among which interactions should be tested. If not provided, test will be performed between top 25 genes.

pvalue	Default c(0.05, 0.01) p-value threshold. You can provide two values for upper and lower threshold.
returnAll	If TRUE returns test statistics for all pair of tested genes. Default FALSE, returns for only genes below pvalue threshold.
geneOrder	Plot the results in given order. Default NULL.
fontSize	cex for gene names. Default 0.8
leftMar	Left margin. Default 4
topMar	Top margin. Default 4
showSigSymbols	Default TRUE. Highlight significant pairs
showCounts	Default TRUE. Include number of events in the plot
countStats	Default 'all'. Can be 'all' or 'sig'
countType	Default 'cooccur'. Can be 'all', 'cooccur', 'mutexcl'
countsFontSize	Default 0.8
countsFontColor	Default 'black'
colPal	colPalBrewer palettes. See RColorBrewer::display.brewer.all() for details
revPal	Reverse the color palette. Default FALSE
showSum	show [sum] with gene names in plot, Default TRUE
plotPadj	Plot adj. p-values instead
colNC	Number of different colors in the palette, minimum 3, default 9
nShiftSymbols	shift if positive shift SigSymbols by n to the left, default = 5
sigSymbolsSize	size of symbols in the matrix and in legend
sigSymbolsFontSize	size of font in legends
pvSymbols	vector of pch numbers for symbols of p-value for upper and lower thresholds c(upper, lower)
limitColorBreaks	limit color to extreme values. Default TRUE

### Details

This function and plotting is inspired from genetic interaction analysis performed in the published study combining gene expression and mutation data in MDS. See reference for details.

### Value

list of data.tables

### References

Gerstung M, Pellagatti A, Malcovati L, et al. Combining gene mutation with gene expression data improves outcome prediction in myelodysplastic syndromes. *Nature Communications*. 2015;6:5901. doi:10.1038/ncomms6901.

### Examples

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
somaticInteractions(maf = lam1, top = 5)
```

---

subsetMaf	<i>Subset MAF objects</i>
-----------	---------------------------

---

## Description

Subsets MAF based on given conditions.

## Usage

```
subsetMaf(
  maf,
  tsb = NULL,
  genes = NULL,
  query = NULL,
  clinQuery = NULL,
  ranges = NULL,
  keepNA = FALSE,
  mult = "first",
  fields = NULL,
  mafObj = TRUE,
  includeSyn = TRUE,
  isTCGA = FALSE,
  dropLevels = TRUE,
  restrictTo = "all",
  verbose = TRUE
)
```

## Arguments

<code>maf</code>	an MAF object generated by <a href="#">read.maf</a>
<code>tsb</code>	subset by these samples (Tumor Sample Barcodes)
<code>genes</code>	subset by these genes
<code>query</code>	query string. e.g, "Variant_Classification == 'Missense_Mutation'" returns only Missense variants.
<code>clinQuery</code>	query by clinical variable.
<code>ranges</code>	subset by ranges. data.frame with 3 column (chr, start, end). Overlaps are identified by <a href="#">foverlaps</a> function with arguments 'type = within', 'mult = all', 'no-match = NULL'
<code>keepNA</code>	Keep NAs while sub-setting for ranges. Default 'FALSE' - removes rows with missing loci prior to overlapping. Set to TRUE to keep them as is.
<code>mult</code>	When multiple loci in 'ranges' match to the variants maf, mult=. controls which values are returned - "all" , "first" (default) or "last". This value is passed to 'mult' argument of <a href="#">foverlaps</a>
<code>fields</code>	include only these fields along with necessary fields in the output
<code>mafObj</code>	returns output as MAF class <a href="#">MAF-class</a> . Default TRUE
<code>includeSyn</code>	Default TRUE, only applicable when mafObj = FALSE. If mafObj = TRUE, synonymous variants will be stored in a seperate slot of MAF object.
<code>isTCGA</code>	Is input MAF file from TCGA source.

dropLevels	Default TRUE.
restrictTo	restrict subset operations to these. Can be 'all', 'cnv', or 'mutations'. Default 'all'. If 'cnv' or 'mutations', subset operations will only be applied on copy-number or mutation data respectively, while retaining other parts as is.
verbose	Default TRUE

**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)
##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')
##Subset by ranges
ranges = data.frame(chr = c("2", "17"), start = c(25457000, 7571720), end = c(25458000, 7590868))
subsetMaf(laml, ranges = ranges)

```

---

survGroup

*Predict genesets associated with survival*

---

**Description**

Predict genesets associated with survival

**Usage**

```

survGroup(
  maf,
  top = 20,
  genes = NULL,
  geneSetSize = 2,
  minSamples = 5,
  clinicalData = NULL,
  time = "Time",
  Status = "Status",
  verbose = TRUE,
  plot = FALSE
)

```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
top	If genes is NULL by default used top 20 genes
genes	Manual set of genes
geneSetSize	Default 2
minSamples	minimum number of samples to be mutated to be considered for analysis. Default 5
clinicalData	dataframe containing events and time to events. Default looks for clinical data in annotation slot of <a href="#">MAF</a> .
time	column name containing time in clinicalData
Status	column name containing status of patients in clinicalData. must be logical or numeric. e.g, TRUE or FALSE, 1 or 0.
verbose	Default TRUE
plot	Default FALSE If TRUE, generate KM plots of the genesets combinations.

**Examples**

```
laml.maf <- system.file("extdata", "tcga_laml.maf.gz", package = "maftools")
laml.clin <- system.file("extdata", "tcga_laml_annot.tsv", package = "maftools")
laml <- read.maf(maf = laml.maf, clinicalData = laml.clin)
survGroup(maf = laml, top = 20, geneSetSize = 1, time = "days_to_last_followup", Status = "Overall_Survival_Stat")
```

---

tcgaAvailable	<i>Prints available TCGA datasets</i>
---------------	---------------------------------------

---

**Description**

Prints available TCGA cohorts

**Usage**

```
tcgaAvailable(repo = c("github", "gitee"))
```

**Arguments**

repo	can be "github" (default) or "gitee". If 'github' fails to fetch, switch to 'gitee'
------	---

**See Also**

[tcgaLoad](#)

**Examples**

```
tcgaAvailable()
```

tcgaCompare

*Compare mutation load against TCGA cohorts***Description**

Compares mutation load in input MAF against all of 33 TCGA cohorts derived from MC3 project.

**Usage**

```
tcgaCompare(
  maf,
  capture_size = NULL,
  tcga_capture_size = 35.8,
  cohortName = NULL,
  tcga_cohorts = NULL,
  primarySite = FALSE,
  col = c("gray70", "black"),
  bg_col = c("#EDF8B1", "#2C7FB8"),
  medianCol = "red",
  decreasing = FALSE,
  logscale = TRUE,
  rm_hyper = FALSE,
  rm_zero = TRUE,
  cohortFontSize = 0.8,
  axisFontSize = 0.8
)
```

**Arguments**

maf	MAF object(s) generated by <a href="#">read.maf</a>
capture_size	capture size for input MAF in MBs. Default NULL. If provided plot will be scaled to mutations per mb. TCGA capture size is assumed to be 35.8 mb.
tcga_capture_size	capture size for TCGA cohort in MB. Default 35.8. Do NOT change. See details for more information.
cohortName	name for the input MAF cohort. Default "Input"
tcga_cohorts	restrict tcga data to these cohorts.
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.
bg_col	background color. Default '#EDF8B1', '#2C7FB8'
medianCol	color for median line. Default red.
decreasing	Default FALSE. Cohorts are arranged in increasing mutation burden.
logscale	Default TRUE
rm_hyper	Remove hyper mutated samples (outliers)? Default FALSE
rm_zero	Remove samples with zero mutations? Default TRUE
cohortFontSize	Default 0.8
axisFontSize	Default 0.8

**Details**

Tumor mutation burden for TCGA cohorts is obtained from TCGA MC3 study. For consistency TMB is estimated by restricting variants within Agilent Sureselect capture kit of size 35.8 MB.

**Value**

data.table with median mutations per cohort

**Source**

TCGA MC3 file was obtained from <https://api.gdc.cancer.gov/data/1c8cfe5f-e52d-41ba-94da-f15ea1337efc>. See TCGAmutations R package for more details. Further downstream script to estimate TMB for each sample can be found in 'inst/scripts/estimate\_tcga\_tmb.R'

**References**

Scalable Open Science Approach for Mutation Calling of Tumor Exomes Using Multiple Genomic Pipelines Kyle Ellrott, Matthew H. Bailey, Gordon Saksena, et. al. Cell Syst. 2018 Mar 28; 6(3): 271–281.e7. <https://doi.org/10.1016/j.cels.2018.03.002>

**Examples**

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

---

 tcgaDriverBP

---

*Compare genes to known TCGA drivers and their biological pathways*


---

**Description**

A small function which uses known cancer driver genes and their associated pathways from TCGA cohorts. See reference for details

**Usage**

```
tcgaDriverBP(m, genes = NULL, top = 20, fontSize = 0.7)
```

**Arguments**

m	an <a href="#">MAF</a> object
genes	genes to compare. Default 'NULL'.
top	Top number of genes to use. Mutually exclusive with 'genes' argument. Default 20
fontSize	Default 0.7

**References**

Bailey MH, Tokheim C, Porta-Pardo E, et al. Comprehensive Characterization of Cancer Driver Genes and Mutations. Cell. 2018;173(2):371–385.e18. doi:10.1016/j.cell.2018.02.060

---

`tcgaLoad`*Loads a TCGA cohort*

---

**Description**

Loads the user mentioned TCGA cohorts

**Usage**

```
tcgaLoad(  
  study = NULL,  
  source = c("MC3", "Firehose"),  
  repo = c("github", "gitee")  
)
```

**Arguments**

<code>study</code>	Study names to load. Use <a href="#">tcgaAvailable</a> to see available options.
<code>source</code>	Source for MAF files. Can be MC3 or Firehose. Default MC3. Argument may be abbreviated (M or F)
<code>repo</code>	one of "github" (default) and "gitee".

**Details**

The function loads curated and pre-compiled MAF objects from TCGA cohorts. TCGA data are obtained from two sources namely, Broad Firehose repository, and MC3 project.

**Value**

An object of class MAF.

**References**

Scalable Open Science Approach for Mutation Calling of Tumor Exomes Using Multiple Genomic Pipelines Kyle Ellrott, Matthew H. Bailey, Gordon Saksena, et. al. Cell Syst. 2018 Mar 28; 6(3): 271–281.e7.

**See Also**

[tcgaAvailable](#)

**Examples**

```
# Loads TCGA LAML cohort (default from MC3 project)  
tcgaLoad(study = "LAML")  
# Loads TCGA LAML cohort (from Borad Firehose)  
tcgaLoad(study = "LAML", source = "Firehose")
```

---

titv	<i>Classifies SNPs into transitions and transversions</i>
------	---

---

### 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 <a href="#">MAF</a> object generated by <code>read.maf</code>
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

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
lam1.titv = titv(maf = lam1, useSyn = TRUE)
```

---

tmb	<i>Estimate Tumor Mutation Burden</i>
-----	---------------------------------------

---

### Description

Estimates Tumor Mutation Burden in terms of per megabases

**Usage**

```
tmb(
  maf,
  captureRegions = NULL,
  captureSize = 50,
  logScale = TRUE,
  ignoreCNV = TRUE,
  plotType = "classic",
  pointcol = "#2c3e50",
  verbose = TRUE
)
```

**Arguments**

maf	maf <a href="#">MAF</a> object
captureRegions	capture regions. Default NULL. If provided sub-sets variants within the capture regions for TMB estimation. Can be a data.frame or a tsv with first three columns containing chromosome, start and end position.
captureSize	capture size for input MAF in MBs. Default 50MB. Mutually exclusive with captureRegions
logScale	Default TRUE. For plotting purpose only.
ignoreCNV	Default TRUE. Ignores all the variants annotated as 'CNV' in the 'Variant_Type' column of MAF
plotType	Can be "classic" or "boxplot". Set to 'NA' for no plot.
pointcol	Default #2c3e50
verbose	Default TRUE

**Value**

data.table with TMB for every sample

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
tmb(maf = lam1)
```

---

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 = NULL,
  prefix = NULL,
  add = TRUE,
  ignoreChr = NULL,
  useSyn = TRUE,
  fn = NULL
)
```

**Arguments**

maf	an MAF object generated by <a href="#">read.maf</a>
ref_genome	BSgenome object or name of the installed BSgenome package. Example: BSgenome.Hsapiens.UCSC Default NULL, tries to auto-detect from installed genomes.
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	Chromosomes to ignore 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. Requires BSgenome data packages for sequence extraction.

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 occurring 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](#) [plotApobecDiff](#)

**Examples**

```
## Not run:
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
lam1.tnm <- trinucleotideMatrix(maf = lam1, ref_genome = 'BSgenome.Hsapiens.UCSC.hg19',
prefix = 'chr', add = TRUE, useSyn = TRUE)

## End(Not run)
```

---

vafCompare

*compare VAF across two cohorts*


---

**Description**

Draw boxplot distribution of VAFs across two cohorts

**Usage**

```
vafCompare(
  m1,
  m2,
  genes = NULL,
  top = 5,
  vafCol1 = NULL,
  vafCol2 = NULL,
  m1Name = "M1",
  m2Name = "M2",
  cols = c("#2196F3", "#4CAF50"),
  sigvals = TRUE,
  nrows = NULL,
  ncols = NULL
)
```

**Arguments**

m1	first <a href="#">MAF</a> object. Required.
m2	second <a href="#">MAF</a> object. Required.
genes	specify genes for which plot has to be generated. Default NULL.
top	if genes is NULL plots top n number of genes. Defaults to 5.
vafCol1	manually specify column name for vafs in m1. Default looks for column 't_vaf'
vafCol2	manually specify column name for vafs in m2. Default looks for column 't_vaf'
m1Name	optional name for first cohort
m2Name	optional name for second cohort
cols	vector of colors corresponding to m1 and m2 respectively.
sigvals	Estimate and add significance stars. Default TRUE.

nrows	Number of rows in the layout. Default NULL - estimated automatically
ncols	Number of genes drawn per row. Default 4

---

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")
scores.gistic <- system.file("extdata", "scores.gistic", package = "maftools")
laml.gistic = readGistic(gisticAllLesionsFile = all.lesions, gisticAmpGenesFile = amp.genes, gisticDelGenesFi
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, compress = FALSE)
```

**Arguments**

maf	an <a href="#">MAF</a> object generated by <a href="#">read.maf</a>
basename	basename for output file to be written.
compress	If 'TRUE' files will be gz compressed. Default 'FALSE'

**Details**

Writes MAF and related summaries to output files.

**Value**

None. Writes output as text files.

**See Also**

[read.maf](#)

**Examples**

```
lam1.maf <- system.file("extdata", "tcga_lam1.maf.gz", package = "maftools")
lam1 <- read.maf(maf = lam1.maf)
write.mafSummary(maf = lam1, basename = 'lam1')
```

# Index

annovarToMaf, 4

bamreadcounts, 5, 77

barplot, 68

cancerhotspots, 6, 7

cancerhotspotsAggr, 6, 7

clinicalEnrichment, 7, 61

coBarplot, 9

coGisticChromPlot, 10

compareSignatures, 12, 19

coOncoplot, 13

DNAcopy, 63, 64, 78

drugInteractions, 16

estimateSignatures, 17, 60

extractSignatures, 12, 13, 18, 18, 67, 68, 80, 92

filterMaf, 19

forestPlot, 20, 28, 42, 57

foverlaps, 83

genesToBarcodes, 21

genotypeMatrix, 21

getClinicalData, 22

getClinicalData,MAF-method (getClinicalData), 22

getCytobandSummary, 23, 26

getCytobandSummary,GISTIC-method (getCytobandSummary), 23

getFields, 23, 39, 47, 84

getFields,MAF-method (getFields), 23

getGeneSummary, 24, 26, 39, 42, 43, 47

getGeneSummary,GISTIC-method (getGeneSummary), 24

getGeneSummary,MAF-method (getGeneSummary), 24

getSampleSummary, 25, 26, 39, 42, 43, 47

getSampleSummary,GISTIC-method (getSampleSummary), 25

getSampleSummary,MAF-method (getSampleSummary), 25

GISTIC, 28, 29

GISTIC (GISTIC-class), 25

GISTIC-class, 25

gisticBubblePlot, 26

gisticChromPlot, 27

gisticCompare, 28

gisticOncoPlot, 29

gtMarkers, 31, 71, 72, 78

icgcSimpleMutationToMAF, 32

inferHeterogeneity, 33, 59

intersectMAF (setdiffMAF), 79

lollipopPlot, 34, 38

lollipopPlot2, 28, 36, 42

MAF, 4, 7–9, 11, 14, 16, 21, 22, 32, 33, 35, 37, 38, 41, 43–47, 50, 53–58, 62, 65, 70, 73, 80, 81, 85–87, 89–92, 94

MAF-class, 39

maf2mae, 40

mafbarplot, 40

mafCompare, 20, 38, 41

mafSummary, 42

mafSurvGroup, 43

mafSurvival, 43, 44

math.score, 45

merge\_mafs, 46

mutCountMatrix, 46

oncodrive, 47, 64, 65

oncoplot, 48, 54

oncostrip, 30, 53

pathways, 53, 54, 65, 66

pfamDomains, 55

plotApobecDiff, 56, 92

plotCBSsegments, 58

plotClusters, 34, 59

plotCophenetic, 18, 60, 60

plotEnrichmentResults, 8, 60, 81

plotmafSummary, 61, 75

plotMosdepth, 63

plotMosdepth\_t, 63

plotOncodrive, 48, 64

plotPathways, 55, 65

plotProtein, 66  
plotSignatures, 13, 19, 57, 67, 68  
plotTiTv, 68, 89  
plotVaf, 69  
prepareMutSig, 70  
prepAscat, 31, 32, 71, 72, 78  
prepAscat\_t, 31, 32, 72, 72, 78  
  
rainfallPlot, 73  
read.maf, 7, 16, 19, 21, 22, 33, 35, 42–47, 50,  
55, 56, 62, 70, 73, 74, 81, 83, 85, 86,  
89, 91, 94  
readGistic, 29, 76, 93  
  
sampleSwaps, 77  
segment, 79  
segmentLogR, 32, 71, 72, 78  
segSummarize, 78  
setdiffMAF, 79  
signatureEnrichment, 61, 80  
somaticInteractions, 81  
subsetMaf, 19, 83  
survGroup, 84  
  
tcgaAvailable, 85, 88  
tcgaCompare, 86  
tcgaDriverBP, 87  
tcgaLoad, 85, 88  
titv, 69, 89  
tmb, 89  
trinucleotideMatrix, 13, 17–19, 57, 68, 90  
  
vafCompare, 92  
  
write.GisticSummary, 93  
write.mafSummary, 75, 93